Algorithmic and Combinatorial Results in Selection and Computational Geometry

Ke Chen
University of Wisconsin-Milwaukee

ALGORITHMIC AND COMBINATORIAL RESULTS IN SELECTION AND COMPUTATIONAL GEOMETRY

by

Ke Chen

A Dissertation Submitted in

Partial Fulfillment of the

Requirements for the Degree of

Doctor of Philosophy

in Engineering

at

The University of Wisconsin-Milwaukee

May 2021

ABSTRACT

ALGORITHMIC AND COMBINATORIAL RESULTS IN SELECTION AND COMPUTATIONAL GEOMETRY

by

Ke Chen

The University of Wisconsin-Milwaukee, 2021
Under the Supervision of Professor Adrian Dumitrescu

This dissertation investigates two sets of algorithmic and combinatorial problems. The first part focuses on the selection problem under the pairwise comparison model. For the classic "median of medians" scheme, contrary to the popular belief that smaller group sizes cause superlinear behavior, several new linear time algorithms that utilize small groups are introduced. Then the exact number of comparisons needed for an optimal selection algorithm is studied. In particular, the implications of a long standing conjecture known as Yao's hypothesis are explored. For the multiparty model, we designed low communication complexity protocols for selecting an exact or an approximate median of data that is distributed among multiple players.

In the second part, three computational geometry problems are studied. For the longest spanning tree with neighborhoods, approximation algorithms are provided. For the stretch factor of polygonal chains, upper bounds are proved and almost matching lower bound constructions in $\mathbb{R}^2$ and higher dimensions are developed. For the piercing number τ and independence number ν of a family of axis-parallel rectangles in the plane, a lower bound construction for $\nu = 4$ that matches Wegner's conjecture is analyzed. The previous matching construction for $\nu = 3$, due to Wegner himself, dates back to 1968.

To my parents

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ACKNOWLEDGMENTS

I am deeply indebted to my advisor, Prof. Adrian Dumitrescu, without whose unwavering support, invaluable advice, and patient guidance the completion of this dissertation would not have been possible. I would also like to extend my sincere thanks to Prof. Christine Cheng for her insightful suggestions in research and generous help in teaching; to Prof. Ichiro Suzuki for his continuous inspiration and profound trust; to Prof. Guangwu Xu for kindly sharing his valuable expertise multiple times when we were faced with difficulties.

I am grateful to professors Craig Guilbault, Jeb Willenbring, and Yi Ming Zou for guiding me around the beautiful world of math which provided the necessary tools and confidence for me to tackle the research projects.

Special thanks to Prof. John Boyland who not only offered tremendous support as the department chair, but also took great care of me as the host. The hospitality of him and his family made the intense graduate school life enjoyable.

Thanks also to the College of Engineering and Department of Computer Science for supporting my research. Last but not least, I'd like to acknowledge the assistance of Mr. Shane Haensgen and Ms. Therese Crary for coordinating my defense.

Chapter 0

Introduction

This thesis is divided into two parts. Chapters 1, 2, and 3 focus on several aspects of the selection problem. Chapters 4, 5, and 6 investigate three different flavors of computational geometry problems. The following provides an overview of each topic studied.

0.1 Selection Problems

Given a set A of n objects with an unknown total order, the *selection problem* asks for the ith smallest element for a prespecified integer $i \in [1, \ldots, n]$. Although under certain models it is possible to determine the order in one shot (for instance, in a 100-meter dash of 8 athletes), in this thesis we focus on the more common regime where only pairwise comparisons are allowed. Algorithms are then evaluated by the number of comparisons performed to find the target.

One special case is the min (resp. max) problem where $i = 1$ (resp. $i = n$). Many natural approaches used since ancient times, knockout tournament as an example, can easily be proved optimal. The story becomes much more complicated for $i = 2$. It is believed that C. L. Dodgson, better known as Lewis Carroll, was the first to write about awarding second prize to the defeated in the final round of a tennis tournament can be unfair. In his 1883 essay, Dodgson proposed a refined tournament that determines the true second-best player.

Although correct, his method turned out to be suboptimal (and unfortunately is never tried in tennis games despite being quite interesting). In 1932, J. Schreier provided an algorithm that uses at most $n - 2 + \lceil \log n \rceil$ comparisons to find the second best among n players. Thirty years later, S. S. Kislitsyn finally proved that this number is indeed the best possible. When $i \geq 3$ (and $i \leq n - 2$ by symmetry), the optimal algorithm remains unknown.

For larger i, in particular $i = \lceil n/2 \rceil$ (i.e., selecting the median), the asymptotic complexity of the best algorithm for selection was comparable to sorting for a while until 1973. In that year, Blum, Floyd, Pratt, Rivest, and Tarjan published their now classic "median of medians" deterministic algorithm Select that only requires $O(n)$ number of comparisons.

0.1.1 Selection algorithms with small groups

The central idea of Select is to find a good pivot element to partition the input set so that a large chunk of elements can be ruled out (because they cannot be the target). This is achieved by arranging elements into small groups where medians can be easily found, then the median of these medians is determined recursively and used as the partition pivot.

In their original paper, Blum et al. proved linearity of this algorithm when the group size is an odd number at least 5. Since then, it has been perpetuated in the literature that using smaller group sizes will force the worst-case number of comparisons to become superlinear, namely $\Omega(n \log n)$. In Chapter 1, we first point out that the usual arguments found in the literature justifying the superlinear worst case fall short of proving this claim. We further prove that it is possible to use group size smaller than 5 while maintaining the linear number of comparisons. To this end, three simple variants of Select are introduced: the repeated step algorithm, the shifting target algorithm, and the hyperpair algorithm, all of which use linear number of comparisons.

Relevant paper.

K. Chen and A. Dumitrescu, Selection algorithms with small groups, *International Journal of Foundations of Computer Science*, Vol. 31, No. 3 (2020), 355–369. A preliminary version in *Proceedings of the 29th International Workshop on Algorithms and Data Structures* (WADS 2015), Victoria, Canada, August 2015; LNCS 9214, Springer, 2015, pp. 189–199.

0.1.2 Yao's hypothesis

The selection problem can be viewed as a special case of the *poset production problem*. Let P be a partially ordered set (poset). A sequence of comparisons on an n-element set X ($n \geq |P|$) is said to produce P if the obtained poset contains a subposet that is isomorphic to P. The poset production problem asks for the smallest number of comparisons needed to produce a given target poset P. Selecting the ith smallest element from an n-element set X is equivalent to producing the "star poset" S_{i-1}^{n-i} where one element (the target) is known to be greater than $i-1$ elements and less than the remaining $n-i$ elements.

Frances Yao conjectured in 1974 that having extra elements does not help in producing the star posets (i.e., the minimum number of comparisons required remains the same). This conjecture, known as the *Yao's hypothesis*, has many compelling implications for the selection problem. Take the median selection as an example, the current best upper bound is slightly lower than $3n$, and the best lower bound is marginally higher than $2n$; whereas Schönhage, Paterson, and Pippenger showed that Yao's hypothesis implies a median selection algorithm for n elements that uses at most $2.5n + o(n)$ comparisons.

Chapter 2 focuses on $V_i(n)$, the exact number of comparisons needed for selecting the ith smallest from n elements. A brief survey is provided on the known upper and lower bounds. Further implications of Yao's hypothesis on $V_i(n)$, both structurally and computationally, are explored. Then we carried out an exhaustive search to calculate the values of $V_i(n)$ for small i and n, aiming at potential contradictions to Yao's hypothesis. Our search produced a few new values for n up to 13. Although no direct contradiction was found, we believe that this framework is a feasible way to tackle Yao's hypothesis once more computing power is available.

0.1.3 Multiparty selection

Being a widely used subroutine for many tasks, the selection problem naturally appears in distributed systems. In 1979, Andrew Yao first considered the two-party model where the two players Alice and Bob each holds a subset of $\{1, 2, \ldots, n\}$. The goal is to determine the median of the multiset $A \cup B$ (A and B are disjoint). In this setup, algorithms are often referred to as protocols, and the main objective is to minimize the amount of information communicated (in bits).

A trivial solution for the two-party model is Alice send her entire set to Bob who will then compute the median of the union and send it back. The total communication complexity is $O(n \log n)$ bits. Several intuitive classic protocols dating back to 1980s achieve a much better bound $O(\log^2 n)$ which has been subsequently reduced to $O(\log n)$. This is optimal in the sense that merely sharing the result (a single number) between the two players require exchanging $\log n$ bits.

In Chapter 3, this model is generalized to a multiparty setting with broadcasting (each message can be seen by all players). For k players, we show a deterministic protocol for finding the median with $O(k \log^2 n)$ communication complexity.

0.1.4 Approximate median

Among the selection problems, it is commonly believed (although no proof is known) that selecting the median is the hardest. On the other hand, finding the exact median is not always necessary in practice. In 1974, Frances Yao suggested the concept of a *mediocre* element. For a set A with $|A| = n$, an element is (i, j)-mediocre if it is neither among the top i nor among the bottom j elements of A. Observe that $(\lfloor (n-1)/2 \rfloor, \lfloor n/2 \rfloor)$-mediocre elements are exact medians of A. Intuitively, smaller i and j means wider tolerance for inaccuracy, which makes the task easier.

In Chapter 3, the approximate median problem under the multiparty model is studied. We provide a protocol for selecting a mediocre element near the median among k players with communication complexity $O(k \log n)$. Somewhat surprisingly, we show that in the two-party model (under suitable additional assumptions), the communication complexity of finding a mediocre element in the vicinity of the median is bounded from above by a constant and is therefore independent of n.

Relevant paper.

K. Chen and A. Dumitrescu, Multiparty selection, *Proceedings of the 31st International Symposium on Algorithms and Computation* (ISAAC 2020), Online, December 2020, to appear.

0.2 Computational Geometry

For the classroom definition, computational geometry is the study of algorithms that compute certain geometric objects. Typical examples include computing convex hull of a set of points, polygon triangulation, and Euclidean shortest path. The first topic in this part of the thesis falls under this definition.

More broadly, computational geometry is the interconnection between computer science and geometry. Often times, geometric results are motivated by network design questions; meanwhile, computational tools are also widely used in solving geometric problems – our second and third topics are respective examples.

0.2.1 Longest spanning tree with neighborhoods

The *Euclidean maximum spanning tree problem* seeks a maximum length tree that connects a given set of n points in the Euclidean space $\mathbb{R}^d$, $d \geq 2$. It finds applications in cluster analysis and network visualization where a set of points are partitioned into homogeneous classes (clusters) such that the maximum distance between elements of the same cluster is minimized (known as the complete-linkage clustering). The problem is easily solvable in polynomial time by Prim's or Kruskal's algorithm.

In Chapter 4, this problem is generalized where the input points are replaced by n compact neighborhoods in $\mathbb{R}^d$. We need to select a point in each neighborhood so that the longest spanning tree on these points has maximum length. Similar generalizations have been studied in the literature for the minimum spanning tree and the traveling salesman problem. The neighborhoods can model uncertainty or inaccuracy of the inputs. It can also be used for hierarchical networks where a "backbone" is constructed before we zoom into each neighborhood to expand local networks.

With the neighborhood setup, the greedy approach becomes suboptimal. We suspect that the generalized problem is already NP-hard in the plane, so an approximation algorithm with ratio 0.511 is provided. It is the first, albeit small, improvement beyond the simple 1/2 approximation.

Relevant paper.

K. Chen and A. Dumitrescu, On the longest spanning tree with neighborhoods, *Discrete Mathematics, Algorithms and Applications*, Vol. 12, No. 5 (2020). A preliminary version in *Proceedings of the 12th International Frontiers of Algorithmics Workshop* (FAW 2018), Guangzhou, China, May 2018; in LNCS.

0.2.2 Stretch factor of polygonal chains

Let $P = (p_1, p_2, \ldots, p_n)$ be a polygonal chain in $\mathbb{R}^d$. The *stretch factor* of P is the ratio between the total length of P and the distance of its endpoints, $\sum_{i=1}^{n-1} |p_i p_{i+1}| / |p_1 p_n|$. For a parameter $c \geq 1$, we call P a *c-chain* if $|p_i p_j| + |p_j p_k| \leq c|p_i p_k|$, for every triple (i, j, k), $1 \leq i < j < k \leq n$. The stretch factor is a global property: it measures how close P is to a straight line, and it involves all the vertices of P; being a c-chain, on the other hand, is a *fingerprint*-property: it only depends on subsets of $O(1)$ vertices of the chain.

In Chapter 5, we investigate how the c-chain property influences the stretch factor. In the Euclidean plane $\mathbb{R}^2$, we show that for every $\varepsilon > 0$, there is a noncrossing c-chain that has stretch factor $\Omega(n^{1/2-\varepsilon})$, for sufficiently large constant $c = c(\varepsilon)$. From the other direction, the stretch factor of a c-chain P is shown to be $O\left(n^{1/2}\right)$, for every constant $c \geq 1$, regardless of whether P is crossing or noncrossing. These results generalize to $\mathbb{R}^d$. For every dimension $d \geq 2$ and every $\varepsilon > 0$, a noncrossing c-chain that has stretch factor $\Omega\left(n^{(1-\varepsilon)(d-1)/d}\right)$ is constructed; on the other hand, the stretch factor of any c-chain is $O\left((n-1)^{(d-1)/d}\right)$.

From the algorithmic perspective, it is trivial to test whether an n-vertex chain in $\mathbb{R}^d$ is a c-chain in $O(n^3)$ time by examining all triples $1 \leq i < j < k \leq n$. Our coauthors, W. Mulzer and Cs. D. Tóth, give a randomized subcubic algorithm based on recent results from geometric range searching that runs in $O\left(n^{3-1/d} \text{ polylog } n\right)$ expected time and $O(n \log n)$ space.

Relevant paper.

K. Chen, A. Dumitrescu, W. Mulzer, and Cs. D. Tóth, On the Stretch factor of polygonal chains, *SIAM Journal on Discrete Mathematics*, submitted. A preliminary version in *Proceedings of the 44th International Symposium on Mathematical Foundations of Computer Science* (MFCS 2019), Aachen, Germany, August 2019; LIPIcs series, Schloss Dagstuhl.

0.2.3 Wegner's inequality for axis-parallel rectangles

Given a family $\mathcal{F}$ of sets, a *piercing set* is a set of elements from $\bigcup_{F\in\mathcal{F}} F$ intersecting every set in $\mathcal{F}$. The *piercing number* of $\mathcal{F}$, denoted by $\tau(\mathcal{F})$, is the size of a minimum piercing set. The *independence number* of $\mathcal{F}$, namely the maximum number of pairwise disjoint sets in $\mathcal{F}$, is denoted by $\nu(\mathcal{F})$. According to an old conjecture of Wegner, if $\mathcal{F}$ is a set of axis-parallel rectangles in the plane, then its piercing number is at most twice its independence number minus 1, that is, $\tau(\mathcal{F}) \le 2\nu(\mathcal{F}) - 1$. In 2015, Corea et al. came up with an elegant lower bound construction with $\tau(\mathcal{F}) \ge 2\nu(\mathcal{F}) - 4$. On the other hand, the current best upper bound, due to the same group of authors, is a $O\left((\log\log \nu(\mathcal{F}))^2\right)$ factor away.

In Chapter 6, we exhibit families of axis-parallel rectangles in the plane with $\tau = 7$ and $\nu = 4$ and thereby show that Wegner's inequality, if true, cannot be improved for $\nu = 4$. The analogous result for $\nu = 3$, due to Wegner, dates back to 1968. A key element in our proof is establishing a connection with the *maximum empty box problem*: Given a set P of n points inside an axis-parallel box U in $\mathbb{R}^d$, find a maximum-volume axis-parallel box that is contained in U but contains no points of P in its interior. Whereas our construction can be extended to any larger independence number ($\nu = 5, 6, \ldots$), its analysis remains open.

Relevant paper.

K. Chen and A. Dumitrescu, On Wegner's conjecture for axis-parallel rectangles, *Discrete Mathematics*, Vol. 343, Issue 12 (2020) 112091.

Part I

Selection Problems

Chapter 1

Selection Algorithms with Small Groups

1.1 Introduction

Together with sorting, selection is one of the most widely used procedures in computer algorithms. Indeed, it is easy to find numerous algorithms (documented in at least as many research articles) that use selection as a subroutine. Two classic examples from computational geometry are [24, 27].

Given a sequence A of n numbers (usually stored in an array), and an integer (target) parameter $1 \leq i \leq n$, the selection problem asks to find the ith smallest element in A. Sorting the numbers trivially solves the selection problem, but if one aims at a linear time algorithm, a higher level of sophistication is needed. A now classic approach for selection [7, 15, 20, 30, 33] from the 1970s is to use an element in A as a pivot to partition A into two smaller subsequences and recurse on one of them with a (possibly different) selection parameter i.

The time complexity of this kind of algorithms is sensitive to the pivots used. For example, if a good pivot is used, many elements in A can be discarded; whereas if a bad pivot is used, in the worst case, the size of the problem may be only reduced by a constant, leading to a quadratic worst-case running time. But choosing a good pivot can be time consuming.

Randomly choosing the pivots yields a well-known randomized algorithm with expected linear running time (see e.g., [8, Ch. 9.2], [25, Ch. 13.5], or [28, Ch. 3.4]), however its worst case running time is quadratic in n.

The first deterministic linear time selection algorithm SELECT (called PICK by the authors), in fact a theoretical breakthrough at the time, was introduced by Blum et al. [7]. By using the median of medians of small (constant size) disjoint groups of A, good pivots that guarantee reducing the size of the problem by a constant fraction can be chosen with low costs. The authors [7, page 451, proof of Theorem 1] required the group size to be at least 5 for the SELECT algorithm to run in linear time. It has been perpetuated in the literature the idea that SELECT with groups of 3 or 4 does not run in linear time: an exercise of the book by Cormen et al. [8, page 223, exercise 9.3-1] asks the readers to argue that "SELECT does not run in linear time if groups of 3 are used".

We first point out that the argument for the $\Omega(n \log n)$ lower bound in the solution to this exercise [9, page 23] is incomplete by failing to provide an input sequence with one third of the elements being discarded in each recursive call in both the current sequence and its sequence of medians; the difficulty in completing the argument lies in the fact that these two sequences are not disjoint thus cannot be constructed or controlled independently. The question whether the original SELECT algorithm runs in linear time with groups of 3 remains open at the time of this writing.

Further, we show that this restriction on the group size is unnecessary, namely that group sizes smaller than 5 can be used by a linear time deterministic algorithm for the selection problem. Since selecting the median in smaller groups is easier to implement and requires fewer comparisons (e.g., 3 comparisons for group size 3 versus 6 comparisons for group size 5), it is attractive to have linear time selection algorithms that use smaller groups. Our main result concerning selection with small group size is summarized in the following theorem.

Theorem 1.1. *There exist suitable variants of* Select *with groups of* 2, 3, *and* 4 *running in* $O(n)$ *time.*

Historical background. The interest in selection algorithms has remained high over the years with many exciting developments (e.g., lower bounds, parallel algorithms, etc) taking place; we only cite a few here [2, 6, 10, 12–19, 21, 22, 29, 32, 33]. We also refer the reader to the dedicated book chapters on selection in [1, 4, 8, 11, 25, 26] and the more recent articles [3, 23], including experimental work.

Outline. In Section 1.2, the classic Select algorithm is introduced (rephrased) under standard simplifying assumptions. In Section 1.3, we introduce a variant of select, the *repeated step* algorithm, which runs in linear time with either group size 3 and 4. With groups of 3, the algorithm executes a certain step, "group by 3 and find the medians of the groups", twice in a row. In Section 1.4, we introduce another variant of select, the *shifting target* algorithm, a linear time selection algorithm with group size 4. In each iteration, upper or lower medians are used based on the current rank of the target, and the shift in the target parameter i is controlled over three consecutive iterations. In Section 1.5, we introduce yet another variant of select, the *hyperpair* algorithm, a linear time selection algorithm with group size 2. The algorithm performs the "group by pairs" step four times in a row to form hyperpairs. In Section 1.6, we briefly introduce three other variants of select with group size 4, including one due to Zwick [34], all running in linear time.

In Section 1.7, we compare our algorithms (with group size 3 and 4) with the original Select algorithm (with group size 5) by deriving upper bounds on the exact numbers of comparisons used by each algorithm. We also present experimental results that verify our numeric calculations. In Section 1.8, we summarize our results and formulate a conjecture on the running time of the original Select algorithm from [7] with groups of 3 and 4, as suggested by our study.

1.2 Preliminaries

Without affecting the results, the following two standard simplifying assumptions are convenient: (i) the input sequence A contains n distinct numbers; and (ii) the floor and ceiling functions are omitted in the descriptions of the algorithms and their analyses. We also assume that all the grouping steps are carried out using the "natural" order, i.e., given a sequence $A = \{a_1, a_2, \ldots, a_n\}$, "arrange A into groups of size m" means that group 1 contains $a_1, a_2, \ldots, a_m$, group 2 contains $a_{m+1}, a_{m+2}, \ldots, a_{2m}$ and so on. Under these assumptions, Select with groups of 5 (from [7]) can be described as follows (using this group size has become increasingly popular, see e.g., [8, Ch. 9.2]):

1. If $n \leq 5$, sort A and return the ith smallest number.
2. Arrange A into groups of size 5. Let M be the sequence of medians of these $n/5$ groups. Select the median of M recursively, let it be m.
3. Partition A into two subsequences $A_1 = \{x | x < m\}$ and $A_2 = \{x | x > m\}$ (the order of elements is preserved). If $i = |A_1| + 1$, return m. If $i < |A_1| + 1$, go to step 1 with $A \leftarrow A_1$ and $n \leftarrow |A_1|$. If $i > |A_1| + 1$, go to step 1 with $A \leftarrow A_2$, $n \leftarrow |A_2|$ and $i \leftarrow i - |A_1| - 1$.

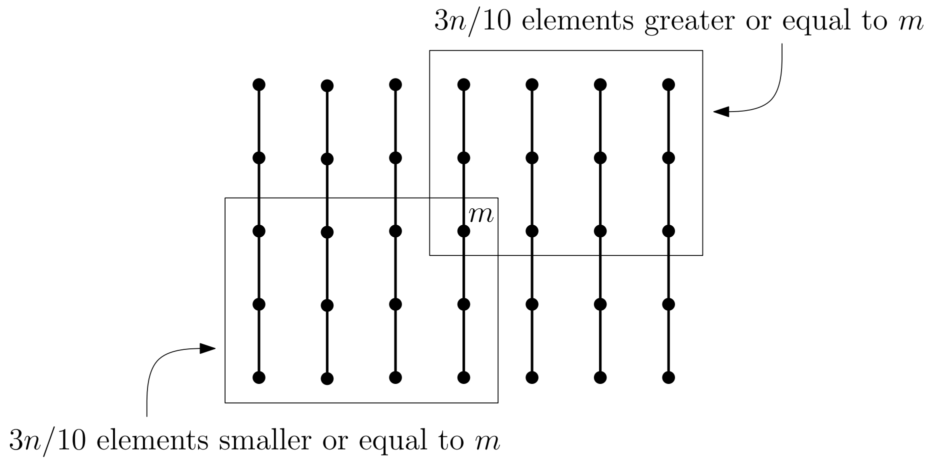

$3n/10$ elements greater or equal to m
m
$3n/10$ elements smaller or equal to m

Figure 1.1: One iteration of the Select algorithm with group size 5. At least $3n/10$ elements can be discarded.

Denote the worst case running time of the recursive selection algorithm on an n-element input by $T(n)$. As shown in Figure 1.1, at least $3 * (n/5)/2 = 3n/10$ elements are discarded at each iteration, which yields the recurrence

$$T(n) \leq T(n/5) + T(7n/10) + O(n). \tag{1.1}$$

This recurrence is one of the following generic form:

$$T(n) \leq \sum_{i=1}^{k} T(a_i\, n) + O(n), \text{ where } a_i > 0 \text{ for } i = 1, \ldots, k \text{ and } \sum_{i=1}^{k} a_i \leq 1. \tag{1.2}$$

It is well-known [8, Ch. 4] (and can be verified by direct substitution) that the solution of (1.2) is

$$T(n) = \begin{cases} O(n) & \text{if } \sum_{i=1}^{k} a_i < 1, \\ O(n \log n) & \text{if } \sum_{i=1}^{k} a_i = 1. \end{cases} \tag{1.3}$$

As such, since the coefficients in (1.1) sum to $1/5 + 7/10 = 9/10 < 1$, we see that the original Select algorithm with group size 5 runs in $T(n) = \Theta(n)$ (as it is well-known).

1.3 The Repeated Step Algorithm

Using group size 3 directly in the Select algorithm in [7] yields

$$T(n) \leq T(n/3) + T(2n/3) + O(n), \tag{1.4}$$

which solves to $T(n) = O(n \log n)$. Here a large portion (at least one third) of A is discarded in each iteration but the cost of finding such a good pivot is too high, namely $T(n/3)$. The idea of our *repeated step* algorithm, inspired by the algorithm in [5], is to find a weaker pivot in a faster manner by performing the operation "group by 3 and find the medians" twice in a row (as illustrated in Figure 1.2). It is worth noting that this method is akin to using the

Tukey's ninther [31]. More precisely, M' as defined in step 3 below is the sequence formed by the Tukey's ninthers of groups of 9 elements in A.

Algorithm

1. If $n \leq 3$, sort A and return the ith smallest number.
2. Arrange A into groups of size 3. Let M be the sequence of medians of these $n/3$ groups.
3. Arrange M into groups of size 3. Let M' be the sequence of medians of these $n/9$ groups.
4. Select the median of M' recursively, let it be m.
5. Partition A into two subsequences $A_1 = \{x|x < m\}$ and $A_2 = \{x|x > m\}$. If $i = |A_1| + 1$, return m. If $i < |A_1| + 1$, go to step 1 with $A \leftarrow A_1$ and $n \leftarrow |A_1|$. If $i > |A_1| + 1$, go to step 1 with $A \leftarrow A_2$, $n \leftarrow |A_2|$ and $i \leftarrow i - |A_1| - 1$.

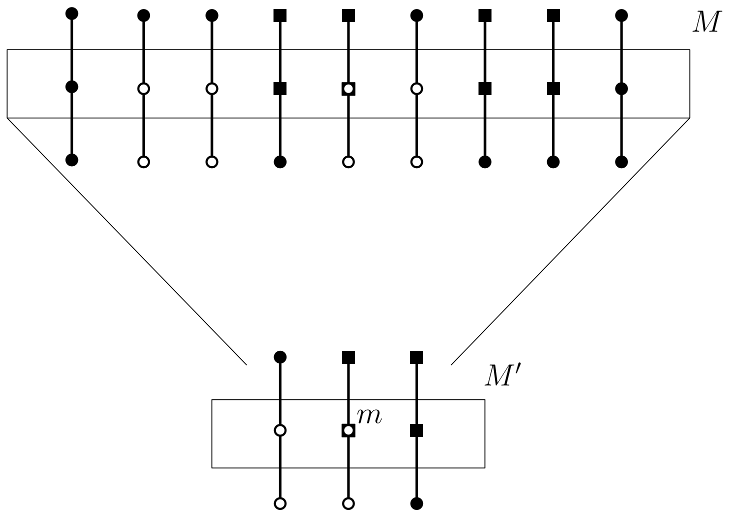


Figure 1.2: One iteration of the *repeated step* algorithm with groups of 3. Empty disks represent elements that are guaranteed to be smaller than or equal to m. Filled squares represent elements that are guaranteed to be greater than or equal to m.

Analysis. Since elements are discarded if and only if they are too large or too small to be the ith smallest element, the correctness of the algorithm is implied. Regarding the time complexity of this algorithm, we have the following lemma:

Lemma 1.2. *The repeated step algorithm with groups of* 3 *runs in* $\Theta(n)$ *time on an* n*-element input.*

Proof. By finding the median of medians of medians instead of the median of medians, the cost of selecting the pivot m reduces from $T(n/3) + O(n)$ to $T(n/9) + O(n)$. We need to determine how well m partitions A in the worst case. In step 4, m is guaranteed to be greater than or equal to $2 * (n/9)/2 = n/9$ elements in M. Each element in M is a median of a group of size 3 in A, so it is greater than or equal to 2 elements in its group. All the groups of A are disjoint, thus m is greater than or equal to $2n/9$ elements in A. Similarly, m is smaller than or equal to $2n/9$ elements in A. Thus, in the last step, at least $2n/9$ elements can be discarded. The recursive call in step 4 takes $T(n/9)$ time. So the resulting recurrence is

$$T(n) \leq T(n/9) + T(7n/9) + O(n),$$

and since the coefficients on the right side sum to $8/9 < 1$, by (1.3), we have $T(n) = \Theta(n)$, as required. $\square$

Note that grouping by 3 twice and finding the median of medians of medians is different from grouping by 9 and finding the median of medians. The number of comparisons required for grouping by 3 twice is $3n/3 + 3n/9 = 12n/9$, while for grouping by 9 the number is $14n/9$ (14 comparisons for selecting the median of 9). The number of elements guaranteed to be discarded is also different: for grouping by 3 twice, at least $2n/9$ elements can be discarded, while for grouping by 9, this number is $5n/18$. So our method trades some of the quality of the pivots for speed (discards fewer elements than the median of 9 approach) by doing fewer comparisons.

1.4 The Shifting Target Algorithm

In the SELECT algorithm introduced in [7], the group size is restricted to odd numbers, where the median of a group has a privileged symmetric position. For group size 4, depending on the choice of upper, lower, or average median, there are three possible partial orders to be considered (see Figure 1.3).

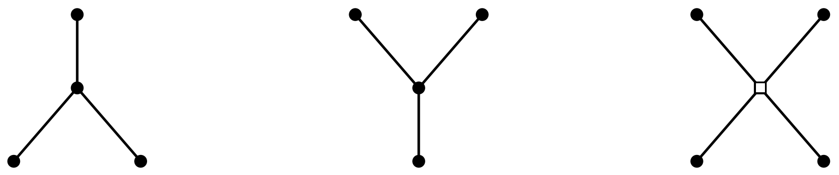


Figure 1.3: Three partial orders of 4 elements based on the upper (left), lower (middle), and average (right) medians. The empty square represents the average of the upper and lower median, which is not necessarily part of the 4-element sequence.

If the upper (or lower) median is always used, only $2 * (n/4)/2 = n/4$ elements are guaranteed to be discarded in each iteration (see Figure 1.4), which gives the recurrence

$$T(n) \leq T(n/4) + T(3n/4) + O(n). \tag{1.5}$$

The term $T(n/4)$ is for the recursive call to find the median of all $n/4$ medians. This recursion solves to $T(n) = O(n \log n)$. Even if we use the average of the two medians, the recursion remains the same since only 2 elements from each of the $(n/4)/2 = n/8$ groups are guaranteed to be discarded.

Observe that if the target parameter satisfies $i \leq n/2$ (resp., $i \geq n/2$), using the lower (resp., upper) median gives a better chance to discard more elements and thus obtain a better recurrence; detailed calculations are given in the proof of Lemma 1.3. Inspired by this idea, we propose the *shifting target* algorithm as follows:

Algorithm

1. If $n \leq 4$, sort A and return the ith smallest number.

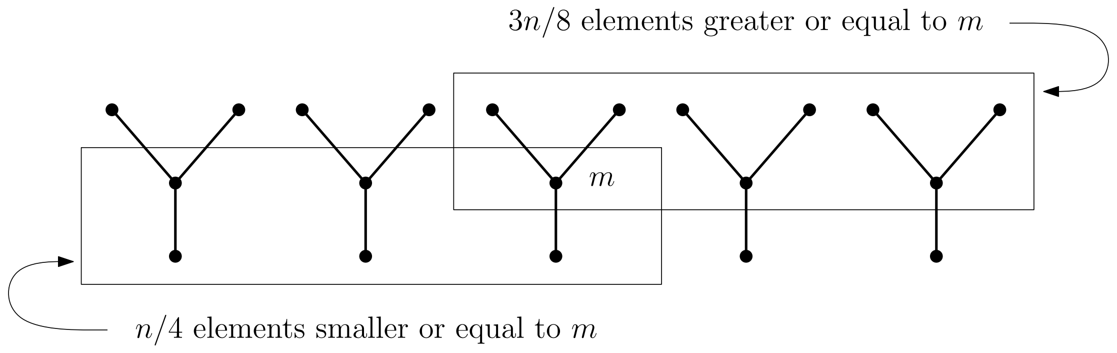
2. Arrange A into groups of size 4. Let M be the sequence of medians of these $n/4$ groups. If $i \leq n/2$, the lower medians are used; otherwise the upper medians are used. Select the median of M recursively, let it be m.

3. Partition A into two subsequences $A_1 = \{x | x < m\}$ and $A_2 = \{x | x > m\}$. If $i = |A_1| + 1$, return m. If $i < |A_1| + 1$, go to step 1 with $A \leftarrow A_1$ and $n \leftarrow |A_1|$. If $i > |A_1| + 1$, go to step 1 with $A \leftarrow A_2$, $n \leftarrow |A_2|$ and $i \leftarrow i - |A_1| - 1$.

Analysis. Regarding the time complexity, we have the following lemma.

Lemma 1.3. *The shifting target algorithm with group size* 4 *runs in* $\Theta(n)$ *time on an n-element input.*

Proof. We shall prove that in at most three consecutive iterations, the size of the problem is reduced by a large enough fraction so that the resulting recurrence is of the form in (1.2) with $\sum_{i=1}^{k} a_i < 1$.



$3n/8$ elements greater or equal to m
m
$n/4$ elements smaller or equal to m

Figure 1.4: Group size 4 with lower medians used.

If in some iteration, we have $i \leq n/4$, then the lower medians are used (see Figure 1.4). Recall that m is guaranteed to be greater than or equal to $2 * (n/4)/2 = n/4$ elements of A. So either m is the ith smallest element in A or at least $3 * (n/4)/2 = 3n/8$ largest elements are discarded, see Figure 1.5. Hence the worst-case running time recurrence is

$$T(n) \leq T(n/4) + T(5n/8) + O(n). \tag{1.6}$$

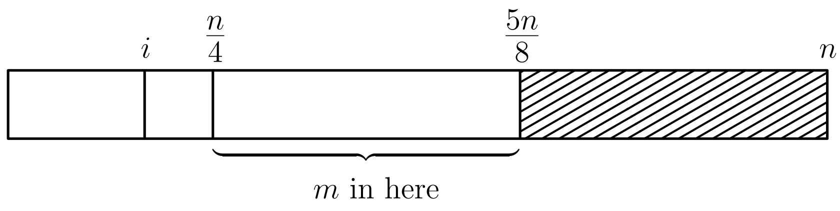


Figure 1.5: When $i \leq n/4$, at least $3n/8$ largest elements (shaded) can be discarded.

Observe that in this case the coefficients on the right side sum to $7/8 < 1$, yielding a linear solution, as required.

Now consider the case $n/4 < i \leq n/2$, again the lower medians are used. If $|A_1| \geq i$, i.e., the rank of m is higher than i, at least $3 * (n/4)/2 = 3n/8$ largest elements are discarded and (1.6) applies. Otherwise, suppose that only $t = |A_1| \geq 2 * (n/4)/2 = n/4$ smallest elements are discarded. Then in the next iteration, $i' = i - t$, $n' = n - t$.

If $i' \leq n'/4$, at least $3n'/8$ elements are discarded, see Figure 1.6. The first iteration

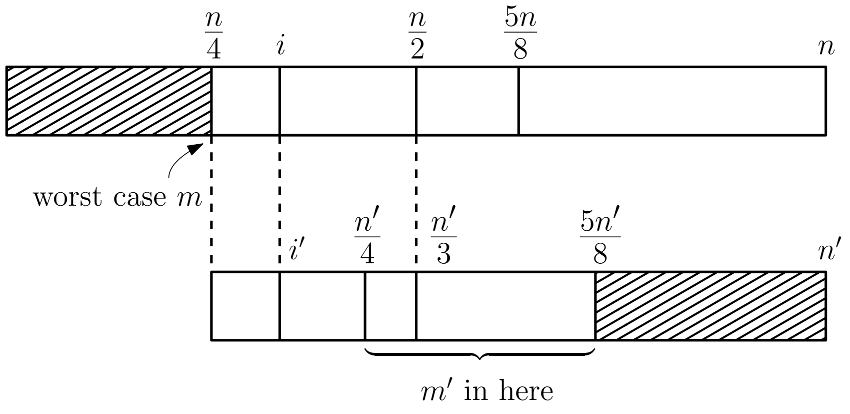


Figure 1.6: When $n/4 < i \leq n/2$ and the rank of m is lower than i, at least $n/4$ smallest elements (shaded on top) are discarded. In the next iteration, if $i' \leq n'/4$, at least $3n'/8$ largest (shaded on bottom) elements can be discarded.

satisfies recurrence (1.5) and we can use recurrence (1.6) to bound the term $T(3n/4)$ from above. We deduce that in two iterations the worst case running time satisfies the recurrence:

$$\begin{aligned} T(n) &\leq T(n/4) + T(3n/4) + O(n) \\ &\leq T(n/4) + T((3n/4)/4) + T((3n/4) * 5/8) + O(n) \\ &= T(n/4) + T(3n/16) + T(15n/32) + O(n). \end{aligned} \tag{1.7}$$

Observe that the coefficients on the right side sum to $29/32 < 1$, yielding a linear solution, as required. Subsequently, we can therefore assume that $i' \geq n'/4$. We have

$$\begin{aligned} i'/n' &= (i-t)/(n-t) \leq (i-n/4)/(n-n/4) \\ &\leq (n/2-n/4)/(n-n/4) = 1/3. \end{aligned}$$

Since $1/4 < i'/n' \leq 1/3 \leq 1/2$, the lower medians will be used. As described above, if at least $3n'/8$ largest elements are discarded, in two iterations, the worst case running time satisfies the same recurrence (1.7).

So suppose that only $t' \geq 2 * (n'/4)/2 = n'/4$ smallest elements are discarded. Let $i'' = i' - t'$, $n'' = n' - t'$. We have

$$\begin{aligned} i''/n'' &= (i'-t')/(n'-t') \leq (i'-n'/4)/(n'-n'/4) \\ &\leq (n'/3-n'/4)/(n'-n'/4) = 1/9. \end{aligned}$$

Since $i''/n'' \leq 1/9 < 1/4$, in the next iteration, at least $3n''/8$ elements will be discarded, see Figure 1.7.

The first two iterations satisfy recurrence (1.5) and we can use recurrence (1.6) to bound the term $T(9n/16)$ from above. We deduce that in three iterations the worst case running time satisfies the recurrence:

$$\begin{aligned} T(n) &\leq T(n/4) + T(3n/4) + O(n) \\ &\leq T(n/4) + T((3n/4)/4) + T((3n/4) * 3/4) + O(n) \\ &= T(n/4) + T(3n/16) + T(9n/16) + O(n) \\ &\leq T(n/4) + T(3n/16) + T((9n/16)/4) + T((9n/16) * 5/8) + O(n) \\ &= T(n/4) + T(3n/16) + T(9n/64) + T(45n/128) + O(n). \end{aligned}$$

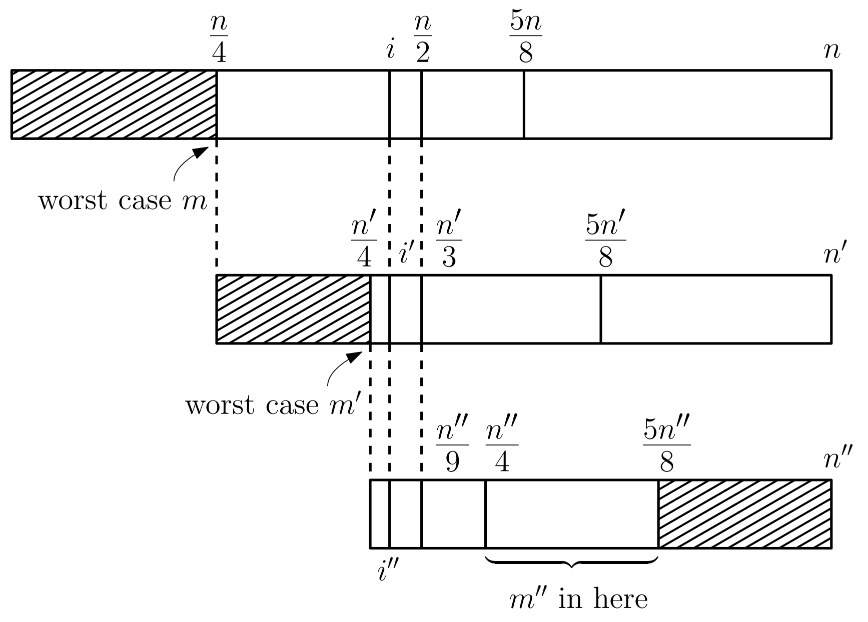


Figure 1.7: In at most three consecutive iterations, the size of the problem is guaranteed to reduce by a large fraction.

The sum of the coefficients on the right side is $119/128 < 1$, so again by (1.3), the solution is $T(n) = \Theta(n)$.

By symmetry, the analysis also holds for the case $i \geq n/2$, and the proof of Lemma 1.3 is complete. $\square$

1.5 The Hyperpair Algorithm

For completeness, we consider the ultimate group size 2, i.e., each group contains a pair of elements. The upper (resp. lower) median of a pair is the larger (resp. smaller) element in that pair. In the original SELECT algorithm, if pairs were used, only $1 * (n/4)$ elements are guaranteed to be discarded in each iteration, which gives the recurrence

$$T(n) \leq T(n/2) + T(3n/4) + O(n). \tag{1.8}$$

The term $T(n/2)$ is for the recursive call to find the median of the $n/2$ upper (or lower) medians. However, the above recursion does not yield a solution linear in n. Now, one can make the following adjustment: instead of taking the median of half the input recursively, let the algorithm recursively compute the jth smallest element among the $n/2$ upper medians, where $j = n/6$. Then $2j = n/2 - j = n/3$ elements can be discarded in each iteration, thus the size of the largest remaining recursive call is $n - n/3 = 2n/3$. However, even with this adjustment, the resulting recurrence (1.9) does not yield a solution linear in n.

$$T(n) \leq T(n/2) + T(2n/3) + O(n). \tag{1.9}$$

The key for obtaining a linear running time in this setting seems to be to use groups of 2 in a repeated manner. The following algorithm has the same flavor as the repeated step algorithm in section 1.3 but uses group size 2. Its name, the *hyperpair* algorithm, will be justified in the analysis.

Algorithm

1. If $n \leq 2$, sort A and return the ith smallest number.
2. Arrange A into groups of size 2. Let M_1 be the sequence of upper medians of these $n/2$ pairs.
3. Arrange M_1 into pairs. Let M_2 be the sequence of lower medians of these $n/4$ pairs.
4. Arrange M_2 into pairs. Let M_3 be the sequence of upper medians of these $n/8$ pairs.
5. Arrange M_3 into pairs. Let M_4 be the sequence of lower medians of these $n/16$ pairs.
6. Select the median of M_4 recursively, let it be m.
7. Partition A into two subsequences $A_1 = \{x | x < m\}$ and $A_2 = \{x | x > m\}$. If $i = |A_1| + 1$, return m. If $i < |A_1| + 1$, go to step 1 with $A \leftarrow A_1$ and $n \leftarrow |A_1|$. If $i > |A_1| + 1$, go to step 1 with $A \leftarrow A_2$, $n \leftarrow |A_2|$ and $i \leftarrow i - |A_1| - 1$.

Analysis. In order to calculate the time complexity of this algorithm, we need to estimate how well m partitions the sequence A. Observe that steps 2–5 can be viewed as constructing *hyperpairs*, as in the non-recursive selection algorithm of Schönhage et al. [30]. In their definition, a single element is a hyperpair with itself as the *center*; given two disjoint copies of a hyperpair, we can combine them to form a larger hyperpair by comparing their centers and taking the upper or lower of these as the new center. The hyperpairs P constructed in our algorithm are illustrated in Figure 1.8. Observe that in P, three elements are guaranteed

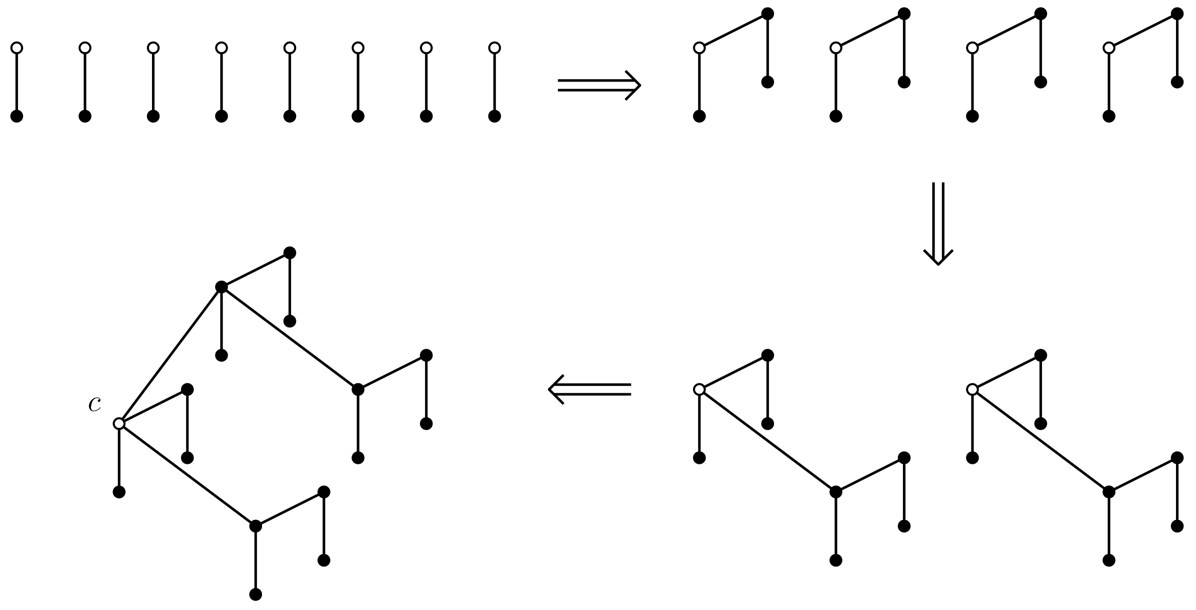


Figure 1.8: Construction of a hyperpair P with 16 elements; the center of each hyperpair is marked by an empty circle.

to be greater than its center c and three are guaranteed to be smaller than c. We are now ready to establish the time complexity of this algorithm:

Lemma 1.4. *The hyperpair algorithm runs in* $\Theta(n)$ *time on an* n*-element input.*

Proof. Steps 2–5 take $n/2 + n/4 + n/8 + n/16 = 15n/16$ comparisons to form the hyperpairs P. The pivot m is the median of the centers of these $n/16$ hyperpairs. So the cost of selecting the pivot is $T(n/16) + 15n/16$. By the above observation about the center c of P, m is guaranteed to be greater than or equal to $4 * (n/16)/2 = n/8$ elements in A. Similarly, m is guaranteed to be smaller than or equal to $n/8$ elements in A. Thus, in the last step, at least $n/8$ elements can be discarded. The resulting recurrence is

$$T(n) \leq T(n/16) + T(7n/8) + O(n),$$

and since the coefficients on the right side sum to $15/16 < 1$, by (1.3), we have $T(n) = \Theta(n)$, as required. □

Note that larger hyperpairs can also be used to obtain linear-time algorithms. If the "group into pairs" step is repeated $2k$ times, $k \geq 2$, where upper and lower medians are used alternatively, then $n/2^{2k}$ hyperpairs of size 2^{2k} are built. Each center is guaranteed to be greater than or equal to 2^k elements in its hyperpair and is also guaranteed to be smaller than or equal to 2^k elements in its hyperpair. So using the median of these centers as pivot, at least $2^k * \left(n/2^{2k}\right)/2 = n/2^{k+1}$ elements can be discarded. The resulting recurrence is

$$T(n) \leq T\left(n/2^{2k}\right) + T\left(\left(1 - 1/2^{k+1}\right) n\right) + O(n),$$

where the $O(n)$ term involves $\sum_{j=1}^{2k} n/2^j = n - n/2^{2k}$ comparisons to build the hyperpairs and at most n comparisons to partition the sequence. Since the coefficients on the right side sum to $1 - \left(2^{k-1} - 1\right)/2^{2k} < 1$, by (1.3), we have $T(n) = \Theta(n)$.

1.6 Other Variants

A similar idea of repeating the group step (from Section 1.3) also applies to the case of groups of 4 and yields

$$T(n) \leq T(n/16) + T(7n/8) + O(n),$$

and thereby another linear time selection algorithm with group size 4.

A hybrid algorithm. Yet another variant of SELECT with group size 4 (we refer to it as the hybrid algorithm), can be obtained by using the ideas of both algorithms together, i.e., repeat the grouping by 4 step twice in a row while M contains the lower medians and M' contains the upper medians (or vice versa). Recursively selecting the median m of M' takes time $T(n/16)$. Notice that m is greater than or equal to $3*(n/16)/2 = 3n/32$ elements in M of which each is greater than or equal to 2 elements in its group in A. So m is greater than or equal to $3n/16$ elements of A. Also, m is smaller than or equal to $2*(n/16)/2 = n/16$ elements in M of which each is smaller than or equal to 3 elements in its group of A. So m is smaller than or equal to $3n/16$ elements of A, thus the resulting recurrence is

$$T(n) \leq T(n/16) + T(13n/16) + O(n),$$

again with a linear solution, as desired.

Zwick's variant. The fact that the Select algorithm can be modified so that it works with groups of 4 in linear time was observed prior to this writing. The following variant, from 2010, is due to Zwick [34]. Split the elements of A into quartets. Find the 2nd smallest element of each quartet (i.e., the lower median), and let M be this subset of $n/4$ elements. Recursively find the $(3/5)(n/4)$th smallest element m of M. Now $(3/5)(n/4)$ groups of A have 2 elements smaller than or equal to m, so m is greater than or equal to $2(3/5)(n/4) = 3n/10$ elements in A. Similarly, $(2/5)(n/4)$ groups of A have 3 elements greater than or equal to m, so m is smaller than or equal to $3(2/5)(n/4) = 3n/10$ elements in A. Thus, the remaining recursive call involves at most $7n/10$ elements, and the resulting recurrence is

$$T(n) \leq T(n/4) + T(7n/10) + O(n).$$

Since $1/4 + 7/10 < 1$, the solution is linear.

1.7 Comparison of the Algorithms and Experimental Results

To compare our algorithms with the original Select algorithm, we first derive upper bounds on the exact numbers of comparisons for each variant in the same manner as in Section 2 of [7]. It should be noted that all recurrent formulas and all proofs do not provide (nor aim to provide) tight bounds or expected number of comparisons. Tighter analytical bounds might exist than those shown. Let now $T(n)$ denote the total number of comparisons performed. For the original Select algorithm with group size 5, we have

$$T(n) \leq T(n/5) + T(7n/10) + 6n/5 + n,$$

in which the term $6n/5$ is for computing the $n/5$ medians (each takes at most 6 comparisons) and the term n is for partitioning the sequence around the selected pivot. Solving the recurrence yields $T(n) \leq 22n$. Similarly, for the repeated step algorithm, we have

$$T(n) \leq T(n/9) + T(7n/9) + 3n/3 + 3n/9 + n,$$

and consequently, $T(n) \leq 21n$. For the hybrid algorithm, we have

$$T(n) \leq T(n/16) + T(13n/16) + 4n/4 + 4n/16 + n,$$

and consequently, $T(n) \leq 18n$. For Zwick's algorithm, we have

$$T(n) \leq T(n/4) + T(7n/10) + 4n/4 + n,$$

and consequently, $T(n) \leq 40n$. For the hyperpair algorithm, we have

$$T(n) \leq T(n/16) + T(7n/8) + 15n/16 + n,$$

and consequently, $T(n) \leq 31n$. For the shifting target algorithm, the analysis is more involved; it yields $T(n) \leq 66n$.

Algorithm	Group	Upper Bound	Average Time	Comparisons		Swaps	
				Average	Max	Average	Max
Hybrid	4	$18n$	364.3ms	4.1	4.2	1.2	1.2
Repeated step	3	$21n$	446.9ms	4.3	4.4	1.8	1.8
Original	5	$22n$	468.9ms	5.7	5.8	1.5	1.5
Hyperpair(4)	2	$31n$	480.6ms	2.9	2.9	3.0	3.0
Zwick's	4	$40n$	541.1ms	6.3	6.3	2.0	2.0
Shifting target	4	$66n$	558.0ms	6.6	6.7	2.0	2.1
Original	4	$O(n \log n)$	560.2ms	6.7	6.7	2.0	2.0
Original	3	$O(n \log n)$	813.4ms	8.2	8.5	3.4	3.5
Hyperpair(6)	2	$127n/3$	452.4ms	2.8	2.8	2.8	2.8
Hyperpair(8)	2	$73n$	456.0ms	2.8	2.8	2.8	2.9
Hyperpair(10)	2	$2047n/15$	458.8ms	2.9	2.9	2.9	2.9

Table 1.1: Experimental results. The last four columns are values per element. The numbers in parentheses for the hyperpair algorithms indicate the numbers of times the "group into pairs" step is repeated. The "Upper Bound" column shows the leading term in the solution of the corresponding recurrence for the worst-case number of comparisons.

We note that sharper upper bounds are possible by taking extra care in avoiding comparisons with known outcomes against the pivot; however, for simplicity of implementation we opted to forego this saving. In order to avoid the overhead of repeated array copying, all the algorithms were implemented in-place, in the sense that, with the exception of the recursion, only $O(1)$ extra space is used in addition to the input array. This requires minor modifications of the algorithms; however, their running time analyses remain unchanged. We carried out 1000 experiments[1] on selecting medians in arrays of 10 million randomly permuted distinct integers. The results are summarized in Table 1.1.

We observed that the experimental results agree with the worst-case estimates in the number of comparisons, in the sense that they show roughly the same speed ranking. One

[1]The experiments were performed on a desktop with 64bits operating system, 7.8GB memory and Intel® Core$^{\text{TM}}$ i7-2600 3.4GHz processor. The C code used can be downloaded at `https://drive.google.com/file/d/0B7USj6ZPkysnMjNwV014RDJGMWc/view?usp=sharing`.

reason why the experimental speed ranking does not fully match the analytical bounds derived is the existence of other operations performed during the selection process that are unaccounted for by the recurrences, such as data copying (shown in the last two columns of the table as swaps). It is worth noting that optimizations introduced in Section 3 of [7], or others discussed in [3], may be used to reduce the multiplicative constant factors.

1.8 Conclusion

The question whether the original selection algorithm introduced in [7] (outlined in Section 1.2) runs in linear time with group size 3 and 4 remains unsettled. Although the recurrences

$$T(n) \leq T(n/3) + T(2n/3) + O(n), \text{ and}$$
$$T(n) \leq T(n/4) + T(3n/4) + O(n)$$

(see (1.4) and (1.5)) for its worst-case running time with these group sizes both solve to $T(n) = O(n \log n)$, we believe that they only give non-tight upper bounds on the worst case scenarios. In any case and against popular belief we think that $\Theta(n \log n)$ is *not* the answer in regard to the time complexity of selection with these group sizes:

Conjecture 1.5. *The* Select *algorithm introduced by Blum et al. [7] runs in* $o(n \log n)$ *time with groups of* 3 *or* 4.

Bibliography

[1] A. V. Aho, J. E. Hopcroft, and J. D. Ullman, *Data Structures and Algorithms*, Addison–Wesley, Reading, Massachusetts, 1983.

[2] M. Ajtai, J. Komlós, W. L. Steiger, and E. Szemerédi, Optimal parallel selection has complexity $O(\log \log n)$, *Journal of Computer and System Sciences* **38(1)** (1989), 125–133.

[3] A. Alexandrescu, Fast deterministic selection, *Proceedings of the 16th International Symposium on Experimental Algorithms* (SEA 2017), June 2017, London, pp. 24:1–24:19.

[4] S. Baase, *Computer Algorithms: Introduction to Design and Analysis*, 2nd edition, Addison-Wesley, Reading, Massachusetts, 1988.

[5] S. Battiato, D. Cantone, D. Catalano, G. Cincotti, and M. Hofri, An efficient algorithm for the approximate median selection problem, *Proceedings of the 4th Italian Conference on Algorithms and Complexity* (CIAC 2000), LNCS vol. 1767, Springer, 2000, pp. 226–238.

[6] S. W. Bent and J. W. John, Finding the median requires $2n$ comparisons, *Proceedings of the 17th Annual ACM Symposium on Theory of Computing* (STOC 1985), ACM, 1985, pp. 213–216.

[7] M. Blum, R. W. Floyd, V. Pratt, R. L. Rivest, and R. E. Tarjan, Time bounds for selection, *Journal of Computer and System Sciences* **7(4)** (1973), 448–461.

[8] T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein, *Introduction to Algorithms*, 3rd edition, MIT Press, Cambridge, 2009.

[9] T. H. Cormen, C. Lee, and E. Lin, *Instructor's Manual*, to accompany *Introduction to Algorithms*, 3rd edition, MIT Press, Cambridge, 2009.

[10] W. Cunto and J. I. Munro, Average case selection, *Journal of ACM* **36(2)** (1989), 270–279.

[11] S. Dasgupta, C. Papadimitriou, and U. Vazirani, *Algorithms*, Mc Graw Hill, New York, 2008.

[12] D. Dor, J. Håstad, S. Ulfberg, and U. Zwick, On lower bounds for selecting the median, *SIAM Journal on Discrete Mathematics* **14(3)** (2001), 299–311.

[13] D. Dor and U. Zwick, Finding the αnth largest element, *Combinatorica* **16(1)** (1996), 41–58.

[14] D. Dor and U. Zwick, Selecting the median, *SIAM Journal on Computing* **28(5)** (1999), 1722–1758.

[15] R. W. Floyd and R. L. Rivest, Expected time bounds for selection, *Communications of ACM* **18(3)** (1975), 165–172.

[16] F. Fussenegger and H. N. Gabow, A counting approach to lower bounds for selection problems, *Journal of ACM* **26(2)** (1979), 227–238.

[17] W. Gasarch, W. Kelly, and W. Pugh, Finding the ith largest of n for small i, n, *SIGACT News* **27(2)** (1996), 88–96.

[18] A. Hadian and M. Sobel, Selecting the t-th largest using binary errorless comparisons, *Combinatorial Theory and Its Applications* **4** (1969), 585–599.

[19] C. A. R. Hoare, Algorithm 63 (PARTITION) and algorithm 65 (FIND), *Communications of the ACM* **4(7)** (1961), 321–322.

[20] L. Hyafil, Bounds for selection, *SIAM Journal on Computing* **5(1)** (1976), 109–114.

[21] J. W. John, A new lower bound for the set-partitioning problem, *SIAM Journal on Computing* **17(4)** (1988), 640–647.

[22] D. G. Kirkpatrick, A unified lower bound for selection and set partitioning problems, *Journal of ACM* **28(1)** (1981), 150–165.

[23] D. G. Kirkpatrick, Closing a long-standing complexity gap for selection: $V_3(42) = 50$, in *Space-Efficient Data Structures, Streams, and Algorithms – Papers in Honor of J. Ian Munro on the Occasion of His 66th Birthday* (A. Brodnik, A. López-Ortiz, V. Raman, and A. Viola, editors), LNCS vol. 8066, Springer, 2013, pp. 61–76.

[24] D. G. Kirkpatrick and R. Seidel, The ultimate planar convex hull algorithm?, *SIAM Journal on Computing* **15(1)** (1986), 287–299.

[25] J. Kleinberg and É. Tardos, *Algorithm Design*, Pearson & Addison–Wesley, Boston, Massachusetts, 2006.

[26] D. E. Knuth, *The Art of Computer Programming, Vol. 3: Sorting and Searching*, 2nd edition, Addison–Wesley, Reading, Massachusetts, 1998.

[27] N. Megiddo, Partitioning with two lines in the plane, *Journal of Algorithms* **6(3)** (1985), 430–433.

[28] M. Mitzenmacher and E. Upfal, *Probability and Computing: Randomized Algorithms and Probabilistic Analysis*, Cambridge University Press, 2005.

[29] M. Paterson, Progress in selection, *Proceedings of the 5th Scandinavian Workshop on Algorithm Theory (SWAT 1996)*, LNCS vol. 1097, Springer, 1996, pp. 368–379.

[30] A. Schönhage, M. Paterson, and N. Pippenger, Finding the median, *Journal of Computer and System Sciences* **13(2)** (1976), 184–199.

[31] J. W. Tukey, The ninther, a technique for low-effort robust (resistant) location in large samples, *Contributions to Survey Sampling and Applied Statistics* (1978), 251–257.

[32] A. Yao and F. Yao, On the average-case complexity of selecting the kth best, *SIAM Journal on Computing* **11(3)** (1982), 428–447.

[33] C. K. Yap, New upper bounds for selection, *Communications of the ACM* **19(9)** (1976), 501–508.

[34] U. Zwick, Personal communication, Sept. 2014.

Chapter 2

Exact Number of Comparisons and Yao's Hypothesis

2.1 Introduction

How many comparisons does it need to guarantee to find the median score of five students? If that is too easy, how about trying sixteen students? Surprisingly, the answer to this latter simple question remains unknown up to date.

Formally, we are given a totally ordered set X of n distinct elements (with the order unknown) and an integer parameter $1 \leq i \leq n$, the *selection problem* asks to find the ith smallest element in X using only pairwise comparisons. Let $V_i(n)$ be the minimum number of comparisons required in the worst case to solve this problem. By symmetry, we shall assume that $i \leq \lceil n/2 \rceil$ (unless otherwise noted). Together with sorting, selection is one of the most widely used procedure in computer science. Numerous applications, including many selection algorithms themselves, recursively involve some small-scale selection subroutines like the ones we ask at the beginning. The exact values of $V_i(n)$ (and the corresponding optimal algorithm) for small i and n are therefore critical for the performance of these algorithms, see [3, 11] for some examples.

It is well known since the 70's that $V_i(n) = \Theta(n)$. However, after decades of efforts, the leading coefficients are still not determined. Take the median selection for example, the current best upper bound is slightly lower than $3n$ while the best lower bound is marginally higher than $2n$. Section 2.1.1 gives a brief historical survey on this gap. Frances Yao [21] has a long-standing conjecture (still open) which will be discussed in Section 2.1.2. One of the major implication is that the upper bound for median selection can be dramatically lowered to about $2.5n$.

Our research will focus on getting a better understanding, both computationally and structurally, of $V_i(n)$ for small i and n. This will in turn help us to tackle Yao's hypothesis, and further improve the general bounds for selection problems.

2.1.1 Gap for Median Selection

Trivially, we have $V_i(n) = O(n \log n)$ by sorting the set X; and $V_i(n) = \Omega(n)$ by a connectivity argument (or by observing that each element must be compared at least once). In 1969, Hadian and Sobel [10] gave an upper bound $V_i(n) \leq n - i + (i-1)\lceil \log_2(n-i+2) \rceil$ which is tight for $i = 1, 2$. Their algorithm was successively improved by Kirkpatrick [13], Yap [22] and Hyafil [12], for various ranges of i within n (see Section 2.2.1 for more details). Note that these algorithms are asymptotically optimal (i.e., $V_i(n) = O(n)$) for a fixed i, but when $i = O(n)$, the above formula still gives $V_i(n) = O(n \log n)$. In particular, the median selection is assumed to be the hardest (although no proof is known) and has drawn the most attention in the literature. For simplicity, we write $M(n) = V_{\lceil n/2 \rceil}(n)$.

In 1973, Blum, Floyd, Pratt, Rivest, and Tarjan [2] were the first to close the asymptotic gap by showing that $M(n) \leq 5.44n$. Their SELECT algorithm partitions X into small groups (subsets of constant size at least 5), then uses the median of medians of these groups as a pivot to rule out a constant fraction of X and only recurses on the remaining. More recently, suitable variants of SELECT with group size 3 and 4 still running in $O(n)$ time have also been put forward [3, 23].

By introducing the concepts of "factory" and "recycling", Schönhage, Paterson, and Pippenger [20] established an algorithm that finds the median of an n element set X with at most $3n + O\left((n \log n)^{3/4}\right)$ comparisons. After more than twenty years, Dor and Zwick [7] utilized some more sophisticated recycle schemes to decrease the upper bound for median selection to $M(n) \le 2.95n$.

On the other hand, Blum et al. [2] were also the first to prove a general non-trivial lower bound $V_i(n) \ge n + i - 2$, which implies $M(n) \ge 1.5n - O(1)$, by using a simple adversary argument. Several groups of authors [12, 13] improved this bound to around $1.8n$ with more complicated adversaries. In 1979, Fussenegger and Gabow [8] introduced a new counting approach and slightly raised the lower bound for some ranges of $V_i(n)$. Bent and John [1] strengthened this counting technique to achieve $M(n) \ge 2n + O(\sqrt{n})$. This bound held the record for over a decade until Dor and Zwick [6] proved that $M(n) \ge (2 + \epsilon)n + o(n)$ for some $\epsilon > 2^{-40}$ which was further improved in [4] to the current best lower bound for median selection $M(n) \ge 2.01n + o(n)$.

2.1.2 Yao's Hypothesis

Both selection and sorting are special cases of a more general problem called poset production. Let P be a partially ordered set (poset). A sequence of comparisons on an n-element set X ($n \ge |P|$) is said to produce P if the obtained poset contains a subposet that is isomorphic to P. Following the notations of Schönhage, Paterson, and Pippenger [20], we define $g(P, n)$ to be the minimum number of comparisons required in the worst case to produce P in X, and let $g(P)$ denote $g(P, |P|)$.

Let S_l^k be the star-shaped poset of $k + l + 1$ elements (see Figure 2.1), i.e., one element is known to be greater than l elements and less than k elements, all the other relations can be inferred from these relations. Note that selecting the ith smallest element from an n-element set X is the same as producing S_{i-1}^{n-i} on X. So $V_i(n) = g\left(S_{i-1}^{n-i}, n\right) = g\left(S_{i-1}^{n-i}\right)$.

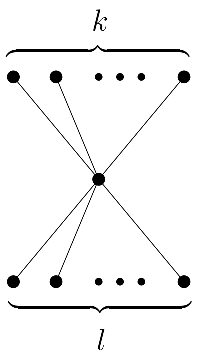


Figure 2.1: The Hasse diagram of S_l^k.

Yao conjectured [21] in 1974 that:

$$g\left(S_l^k\right) = g\left(S_l^k, n\right) \quad \text{for } n \geq l + k + 1, \tag{2.1}$$

i.e., having extra elements would not help in producing S_l^k. It is not hard to see that equation (2.1) holds when $l = 0$ (or $k = 0$). Yao [21] also proved that it holds when $l = 1$. The conjecture remains open for other values of l. Let $Y_i(n)$ denote the minimum number of comparisons needed in the worst case to select the ith smallest element from n elements *assuming Yao's hypothesis*. Clearly, $Y_i(n) \leq V_i(n)$. Furthermore, if Yao's hypothesis is true, then $Y_i(n) = V_i(n)$ for all suitable pairs (n, i).

Schönhage, Paterson, and Pippenger [20] showed that Yao's hypothesis implies a median selection algorithm for n elements that uses at most $2.5n + o(n)$ comparisons. We shall prove a slightly generalized version such that the intermediate results are useful for our research on small i and n.

Theorem 2.1. *For $1 \leq i \leq n$, we have:*

$$Y_i(2n - i + 1) \leq Y_i(n) + n, \tag{2.2}$$

$$Y_{2i}(n + i) \leq Y_i(n) + n, \tag{2.3}$$

$$Y_{\lceil n/2 \rceil}(n) \leq 2.5n + o(n). \quad \text{(Theorem 3.1 in [20])}$$

The theorem is an easy consequence of the following lemma:

Lemma 2.2. *Assuming Yao's hypothesis, for $k, l \in \mathbb{N}$, we have:*

$$g\left(S_l^{2k+1}\right) \leq g\left(S_l^k\right) + k + l + 1, \tag{2.4}$$

$$g\left(S_{2l+1}^k\right) \leq g\left(S_l^k\right) + k + l + 1. \tag{2.5}$$

Proof. Suppose that we want to produce S_l^{2k+1} for some nonnegative integers k and l. Start with $2k+2l+2$ elements, we first use $k+l+1$ comparisons to form pairs. Then apply $g\left(S_l^k\right)$ comparisons on the smaller elements of all the pairs (see Figure 2.2 Left) to get S_l^{2k+1} as a subposet. According to Yao's hypothesis (2.1), we have:

$$g\left(S_l^{2k+1}\right) = g\left(S_l^{2k+1}, 2k+2l+2\right) \leq g\left(S_l^k\right) + k + l + 1.$$

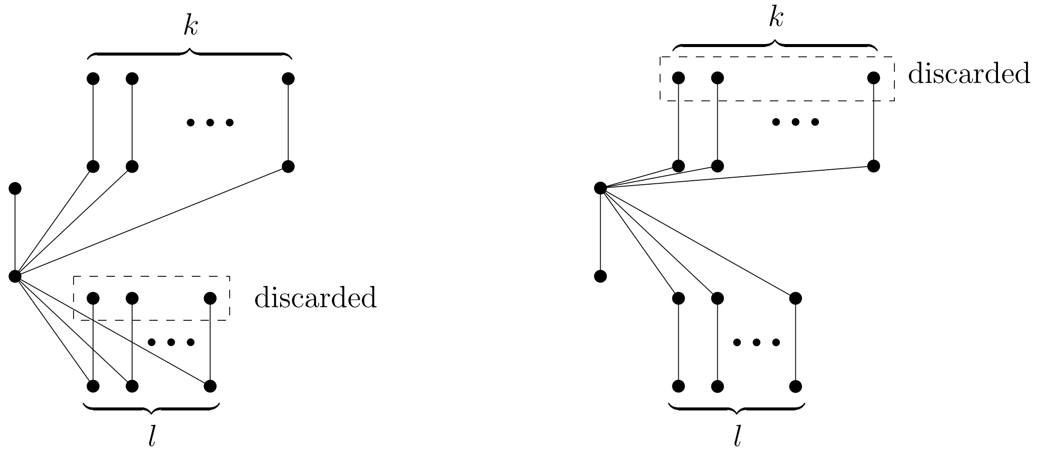


Figure 2.2: Left: illustration of inequality (2.2). Right: illustration of inequality (2.3).

Similarly, first forming $k+l+1$ pairs, then applying $g\left(S_l^k\right)$ comparisons on the larger elements of all the pairs (see Figure 2.2 Right) yields:

$$g\left(S_{2l+1}^k\right) = g\left(S_{2l+1}^k, 2k+2l+2\right) \leq g\left(S_l^k\right) + k + l + 1.$$ □

Proof of Theorem 2.1. Recall that under Yao's hypothesis, $Y_i(n) = g\left(S_{i-1}^{n-i}\right)$. Let $n = k + l + 1$ and $i = l + 1$, then inequality (2.4) can be rewritten as inequality (2.2). Similarly, inequality (2.5) yields inequality (2.3).

For the last inequality in the theorem, let $l = 2k + 1$ in inequality (2.4), we have

$$g\left(S_{2k+1}^{2k+1}\right) \leq g\left(S_{2k+1}^{k}\right) + 3k + 2.$$

Apply inequality (2.5) on the right hand side yields

$$g\left(S_{2k+1}^{2k+1}\right) \leq g\left(S_{k}^{k}\right) + 5k + 3.$$

Since $g\left(S_0^0\right) = 0$ and $g\left(S_{2k}^{2k}\right) \leq g\left(S_{2k+1}^{2k+1}\right)$, an iteration of this inequality yields

$$g\left(S_k^k\right) \leq 5k + O\left(\log k\right).$$

Set $n = 2k + 1$ gives the final bound in the theorem. □

Schönhage et al. [20] observed that the generalization of Yao's hypothesis to an arbitrary poset does not hold. Let P be the poset with seven elements whose Hasse diagram is shown in Figure 2.3 Left. The goal is to produce P. By an exhaustive search, one can show that $g(P) = g(P, 7) = 8$. On the other hand, if we have eight elements, the poset shown in Figure 2.3 Right can be produced with just 7 symmetric comparisons. But the right poset contains P as a subposet, so $g(P, 8) \leq 7 < g(P, 7)$.

2.2 Bounds for Small i and n

Dor and Zwick [5] attempted to disprove Yao's hypothesis by extending Theorem 2.1 to get upper bound of $Y_i(n)$ for $i = \alpha n$, $0 < \alpha < 1$. Although the asymptotic bound they got does not contradict the known lower bounds of $V_i(n)$, we believe that inequalities (2.2) and (2.3)

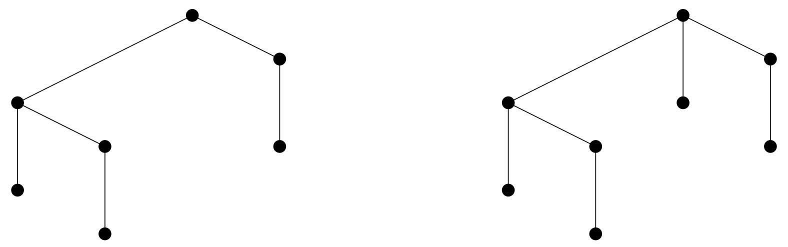


Figure 2.3: A counter example to the generalized Yao's hypothesis. Left: poset P with 7 elements that needs 8 comparisons to produce. Right: a poset with 8 elements that contains P can be produced with 7 comparisons.

can help us find a counterexample with small i and n: Using these inequalities, we can calculate numerical upper bounds of $Y_i(n)$. If Yao's hypothesis is true, these bounds should not contradict the known lower bounds of $V_i(n)$. This enables us to examine the hypothesis by direct computation. To assist the computation, we first list a few easy observations.

- By padding $-\infty/+\infty$ elements, we have:

$$V_i(n) \leq V_{i+1}(n+1), \quad V_i(n) \leq V_i(n+1). \tag{2.6}$$

- Suppose that we are looking for the $(i+1)$th smallest element out of an $(n+1)$-element set X. One strategy is as follows. Take an arbitrary element a out and call the remaining set Y, i.e., $X = Y \sqcup \{a\}$. Apply $V_i(n)$ comparisons on Y to get the ith smallest element, say b. Then compare a and b. If a is smaller than b, b is the $(i+1)$th smallest element in X that we are searching for. Otherwise, we need at most $n-i$ comparisons to find the minimum one from $\{a\}$ union the $n-i$ elements bigger than b. See Figrue 2.4 for a demonstration. Therefore,

$$V_{i+1}(n+1) \leq V_i(n) + n - i + 1. \tag{2.7}$$

- Similar to inequality (2.7). We can first find the ith smallest element b of an n-element subset, then compare it with the remaining element a. We are done if $a > b$, otherwise we need at most additional $i-1$ comparisons to find the biggest element from $\{a\}$

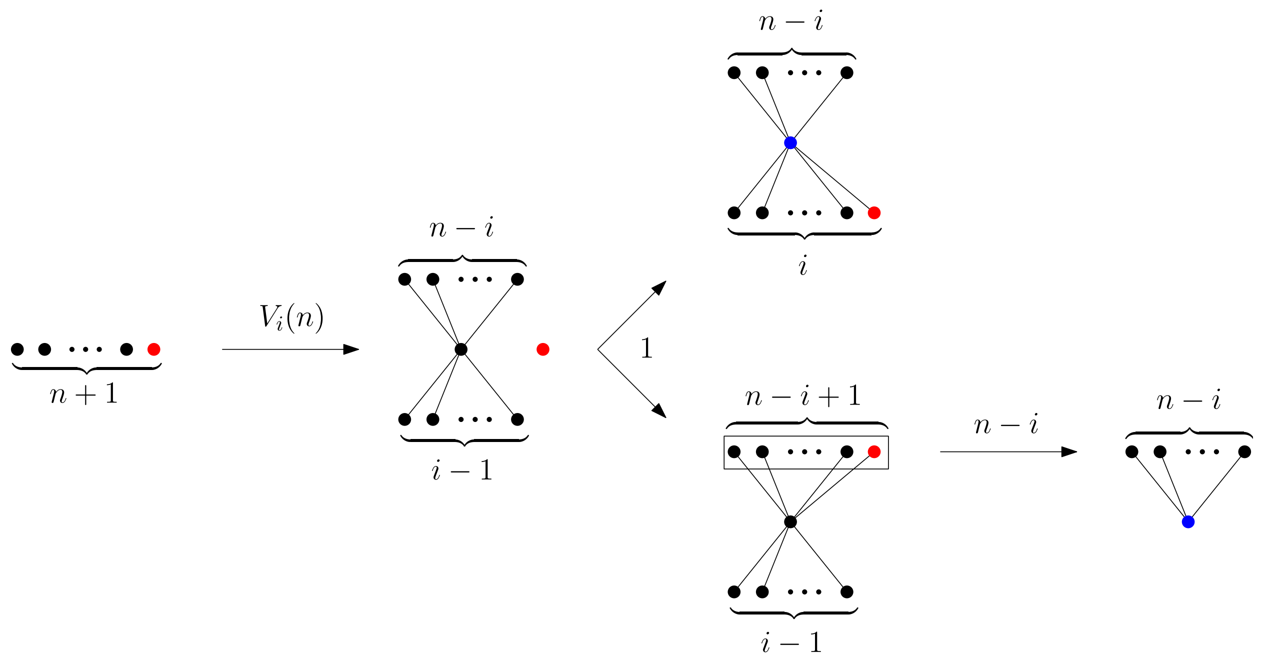


Figure 2.4: The arbitrarily chosen element a is shown in red. The identified target is shown in blue. Number of comparisons needed is shown on each arrow.

union the $i-1$ elements smaller than b.

$$V_i(n+1) \leq V_i(n) + i. \tag{2.8}$$

Note that the observations (2.6), (2.7), and (2.8) all apply to $Y_i(n)$.

2.2.1 Known Bounds for $V_i(n)$

The following list summarizes the known upper bounds for $V_i(n)$.

- Hadian and Sobel (1969) [10], see also Knuth [16]:

$$V_i(n) \leq n - i + (i-1)\lceil \log_2(n-i+2) \rceil, \text{ for } 1 \leq i \leq \left\lceil \frac{n}{2} \right\rceil. \tag{2.9}$$

- Kirkpatrick's improvement (1974) [13] to the Hadian-Sobel algorithm:

$$\text{when } \left(2^{\lceil \log(i+1) \rceil}\right) 2^s < n - i + 2 \leq \left(2^{\lceil \log(i+1) \rceil} + 1\right) 2^s, \text{ for some } s \geq 0, \tag{2.10}$$

$$V_i(n) \leq n - i + (i-1)\lceil \log_2(n-i+2) \rceil - \lfloor (i-1)/2 \rfloor. \tag{2.11}$$

- Yap's improvement (1976) [22] to the Kirkpatrick-Hadian-Sobel algorithm:

$$\text{when } \left(2^{\lceil \log(i) \rceil} + k\right) 2^s < n - i + 2 \leq \left(2^{\lceil \log(i) \rceil} + k + 1\right) 2^s, \tag{2.12}$$

$$\text{and } \lfloor (i-1)/2 \rfloor > k \lfloor \log i \rfloor, \text{ for some } s \geq 0, k \geq 0, \tag{2.13}$$

$$V_i(n) \leq n - i + (i-1)\lceil \log_2(n-i+2) \rceil - \lfloor (i-1)/2 \rfloor + k \lfloor \log i \rfloor. \tag{2.14}$$

The following list summarizes the known lower bounds for $V_i(n)$.

- Hyafil (1976) [12]:

$$V_i(n) \geq n - i + (i-1)\left\lceil \log_2 \frac{n}{i-1} \right\rceil, \text{ for } 1 < i \leq n. \tag{2.15}$$

- Fussenegger and Gabow (1979) [8]:

$$V_i(n) \geq n - i + \left\lceil \log_2 \binom{n}{i-1} \right\rceil, \text{ for } 1 \leq i \leq n. \tag{2.16}$$

- Kirkpatrick (1981) [13, 14]:

$$V_i(n) \geq \begin{cases} n + i - 3 + \sum\limits_{j=0}^{i-2} \left\lceil \log_2 \frac{n-i+2}{i+j} \right\rceil, & 2 \leq i \leq \frac{n}{3} \\ \left\lfloor \frac{3n+i+1}{2} \right\rfloor - 3, & \frac{n}{3} < i \leq \left\lceil \frac{n}{2} \right\rceil. \end{cases} \tag{2.17}$$

- Bent and John (1985) [1]:

$$V_i(n) \geq \left\lceil n + R_i(n) - 2\sqrt{R_i(n)} \right\rceil, \text{ for } \sqrt{R_i(n)} \leq i \leq \left\lceil \frac{n}{2} \right\rceil. \tag{2.18}$$

$$\text{where} \quad R_i(n) = \log_2 \binom{n}{i} - \log_2(n - i + 1) + 3. \tag{2.19}$$

2.2.2 Calculating $V_i(n)$ for Small i and n

As noted before, $Y_i(n) \leq V_i(n)$. So all the upper bounds on $V_i(n)$ are also upper bounds on $Y_i(n)$. Using these as the initial values, we can then apply recursive relations (2.2), (2.3), (2.6), (2.7), and (2.8) to get sharper upper bounds on $Y_i(n)$. The results are summarized in table 2.1. Observe that in some entries (for example, $n = 16$, $i = 7$), the upper bound u of $Y_i(n)$ lies strictly in between the best known upper bound and lower bound of $V_i(n)$. This motivates us to calculate exact values of $V_i(n)$ for some suitable pairs (n, i). If $V_i(n) > u$, then Yao's hypothesis is disproved; otherwise, we can analyze the generated algorithm to have a better understanding of why the hypothesis helps in improving the upper bound.

The worst case selection problem can be viewed as a game played by an algorithm and an adversary. In each step, the algorithm is in charge of picking an optimal pair to compare, while the adversary determines the worse outcome of this comparison. Clearly, this process can be modeled by a minimax tree. Therefore, calculating exact values of $V_i(n)$ is straightforward: simply traverse the entire tree. Unfortunately, these trees are gigantic, even for relatively small values of n. Gasarch, Kelly, and Pugh [9] wrote a program in 1996 to do such computation. With some refinements (described below), they were able to get up to the median of 10 elements, $V_5(10) = 16$ (and $V_4(11) = V_3(12) = 17$). Their program is no longer available[1]. Oksanen [18] posted another implementation on his website that claims to find the optimal selection algorithms for small n and i. Computer generated algorithms for up to $n = 15$ are also provided, some of which are claimed to be optimal without proof. Therefore,

[1]Prof. Gasarch does not have the program anymore but kindly directed us to another author, Dr. Kelly who did not respond to our request.

in the Table 2.1, we list his results as experimental upper bounds for $V_i(n)$. The program is not compilable, so we developed our own with similar refinements to those described in [9] as follows.

1. Apply the classic alpha-beta pruning technique on minimax tree search. The basic idea is to keep updating a lower bound and upper bound pair with the current search result so that some future searches can be determined to be "useless" and therefore can be pruned.

2. After some comparisons, an element may be determined to be not the target. For example, if an element is greater than three other elements then it cannot possibly be the third smallest one. It is easy to see that any further comparison with such elements does not help in finding the target. Therefore, these elements (together with the known relations on them) can be eliminated to get a smaller poset that requires the same number of further comparisons as the original poset. Note that the target may be different in the smaller poset.

3. For $i = 1, 2$, there are formulas for the minimum number of comparisons needed to find the ith smallest element starting with an arbitrary poset (see Knuth [16, Ex. 5.3.3.6]). So these subproblems can be answered directly without a brute force search.

4. If the two outcomes of a comparison result in isomorphic posets, we only need to proceed with one of them. Similarly, if two comparisons result in isomorphic posets (in both outcomes), we only need to continue with one of them.

5. Intuitively, many different sequences of comparisons can end up with the same poset (up to isomorphism). So we store all the encountered posets (and the searching result on them) in a database. After each comparison, the database is queried for the resulting poset. The brute force search will continue only if the answer is not known.

Note that refinements 4 and 5 both require isomorphism tests on posets. We use the C package `nauty` by McKay and Piperno [17] for this purpose. Although `nauty` is a general purpose graph isomorphism test package that may have non-polynomial performance on certain input, our program seems to have not encountered such problem (according to a consistent running time for every ten thousand recursive calls). However, the program may benefit from a better algorithm specific for poset isomorphism test.

The complete search tree for $V_2(4) = 4$ using this scheme is illustrated in Figure 2.5. The number of non-isomorphic posets increases rapidly with respect to n, the number of vertices in the poset. For example, when $n = 10$, the number is 2,567,284; when $n = 13$, the number is 33,823,827,452; and when $n = 16$ (the largest currently known), it reaches 4,483,130,665,195,087. Thanks to refinement 2, not all of them need to be stored in the database. When $n = 10$, our database has (accumulatively) 122,301 posets which takes about 130MB space; when $n = 13$ (up to $i = 6$), the number of stored posets increases to 134,784,532, which takes over 134GB space. The fast increasing nature of this problem, not surprisingly, sets a hard limit on the range of $V_i(n)$ that we can compute. The current record is $V_5(13) = 21$. The computation took over 60 hours to finish[2]. Table 2.1 summarizes the numerical values of the various bounds introduced in this section.

[2]On a computing node at UWM mortimer faculty research cluster, using 200GB RAM and a single core of an Intel®Xeon®E5-2650 v2 processor @ 2.60GHz.

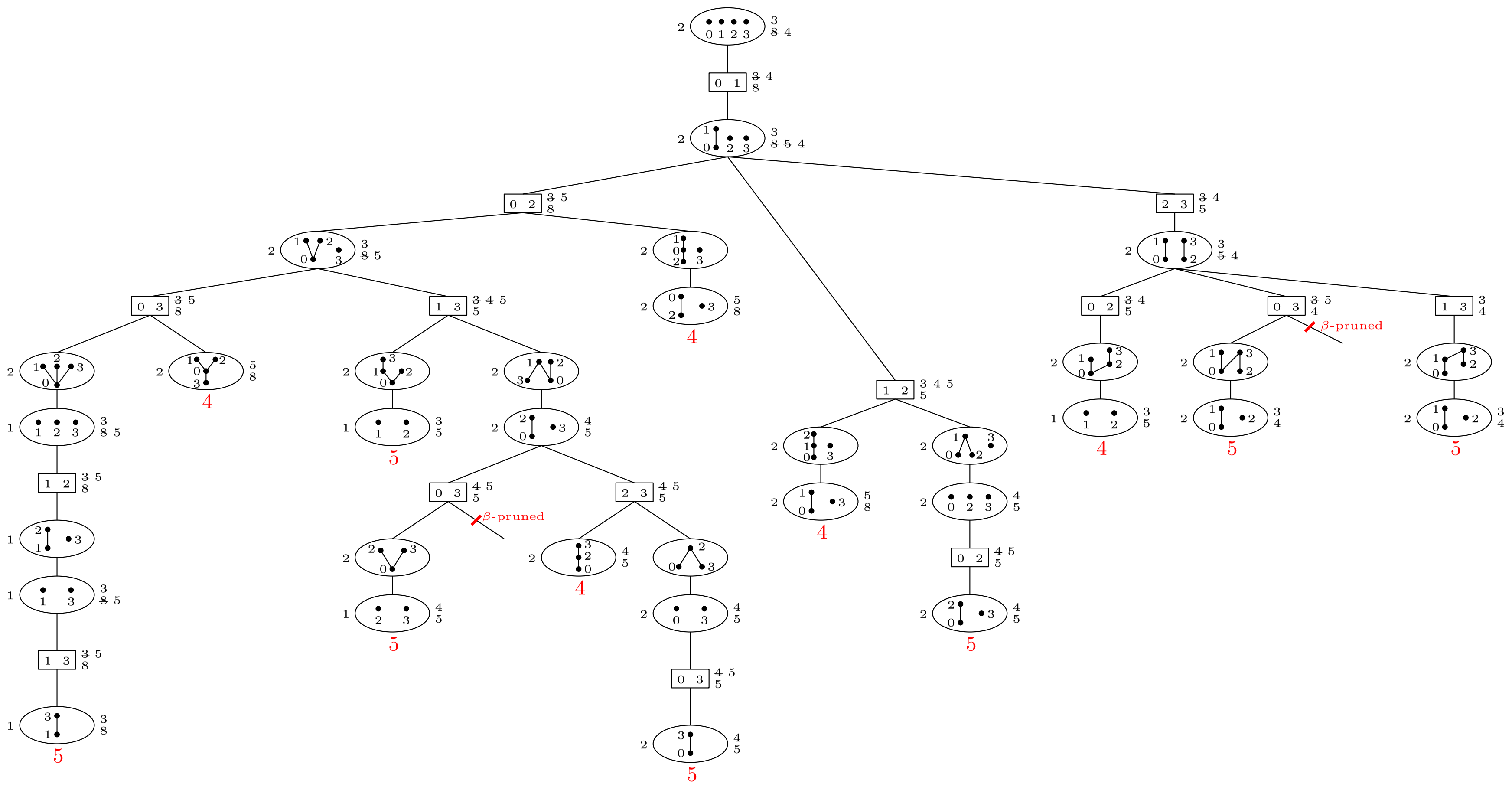


Figure 2.5: Search tree for $V_2(4) = 4$. Each circle node contains the current poset with the target i shown to the left. Each square node represents a choice for the next comparison. For both kinds, the alpha-beta values (with updating history) are shown to the right. The total number of comparisons used when the target is reached is shown in red at the end of each branch. To make sense of the pruning operations, the tree should be read in the same order as it was traversed, i.e., depth-first.

n	i	best known theoretical upper bound of $V_i(n)$	upper bound of $Y_i(n)$	$V_i(n)$	best known theoretical lower bound of $V_i(n)$
3	2	3 (2.9)	3	3	3 (2.15)
4	2	4 (2.9)	4	4	4 (2.15)
5	2	6 (2.9)	6	6	6 (2.15)
	3	6 (2.9)	6	6	6 (2.15)
6	2	7 (2.9)	7	7	7 (2.15)
	3	8 (2.11)	8	8	8 (2.17)
7	2	8 (2.9)	8	8	8 (2.15)
	3	10 (2.9)	10	10	9 (2.16)
	4	11 (2.14)	10	10	10 (2.17)
8	2	9 (2.9)	9	9	9 (2.15)
	3	11 (2.9)	11	11	11 (2.17)
	4	13 (2.9)	12	12	11 (2.17)
9	2	11 (2.9)	11	11	11 (2.15)
	3	12 (2.9)	12	12	12 (2.15)
	4	14 (2.9)	14	14	13 (2.17)
	5	16 (2.9)	14	14	13 (2.17)
10	2	12 (2.9)	12	12	12 (2.15)
	3	14 (2.11)	14	14	14 (2.17)
	4	15 (2.9)	15	15	14 (2.17)
	5	17 (2.9)	16	16	15 (2.17)
11	2	13 (2.9)	13	13	13 (2.15)
	3	15 (2.11)	15	15	15 (2.17)
	4	18 (2.11)	17	17	16 (2.17)
	5	18 (2.9)	18	18*	16 (2.17)

	6	20 (2.9)	18	18*	17 (2.17)
12	2	14 (2.9)	14	14	14 (2.15)
	3	17 (2.9)	17	17	16 (2.16)
	4	19 (2.14)	18	18*	17 (2.17)
	5	21 (2.11)	19	19*	18 (2.17)
	6	21 (2.9)	20	20*	18 (2.17)
13	2	15 (2.9)	15	15	15 (2.15)
	3	18 (2.9)	18	18	17 (2.16)
	4	21 (2.9)	20	20*	19 (2.17)
	5	24 (2.9)	21	21*	19 (2.17)
	6	25 (2.11)	22	$\leq$ 22	20 (2.17)
	7	24 (2.9)	23	$\leq$ 23	20 (2.17)
14	2	16 (2.9)	16	16	16 (2.15)
	3	19 (2.9)	19	19	19 (2.17)
	4	22 (2.9)	21	$\leq$ 21	20 (2.17)
	5	25 (2.9)	23	$\leq$ 23	21 (2.17)
	6	28 (2.9)	24	$\leq$ 24	21 (2.17)
	7	28 (2.11)	25	$\leq$ 25	22 (2.17)
15	2	17 (2.9)	17	17	17 (2.15)
	3	20 (2.9)	20	20	20 (2.17)
	4	23 (2.9)	23	$\leq$ 23	22 (2.17)
	5	26 (2.9)	25	$\leq$ 25	22 (2.17)
	6	29 (2.9)	26	$\leq$ 26	23 (2.17)
	7	31 (2.14)	28	$\leq$ 28	23 (2.17)
	8	32 (2.14)	28	$\leq$ 28	24 (2.17)
	2	18 (2.9)	18	18	18 (2.15)
	3	21 (2.9)	21	21	21 (2.17)

16

	4	24 (2.9)	24	≤ 24	23 (2.17)
	5	27 (2.9)	26	≤ 26	24 (2.17)
	6	30 (2.9)	29	≤ 29	24 (2.17)
	7	33 (2.9)	29 (by (2.3) on $Y_5(11)$)	≤ 30	25 (2.17)
	8	36 (2.9)	30 (by (2.3) on $Y_4(12)$)	≤ 31	25 (2.17)
17	2	20 (2.9)	20	20	20 (2.15)
	3	22 (2.9)	22	22	22 (2.15)
	4	25 (2.9)	25	≤ 25	24 (2.17)
	5	28 (2.9)	28	≤ 28	25 (2.17)
	6	31 (2.9)	29 (by (2.2) on $Y_6(11)$)	≤ 31	26 (2.17)
	7	34 (2.9)	32 (by (2.6) on $Y_7(18)$)	≤ 33	26 (2.17)
	8	37 (2.9)	31 (by (2.3) on $Y_5(12)$)	≤ 34	27 (2.17)
	9	40 (2.9)	34 (by (2.6) on $Y_9(18)$)	≤ 35	27 (2.17)
18	2	21 (2.9)	21	21	21 (2.15)
	3	24 (2.11)	24	24	24 (2.17)
	4	26 (2.9)	26	≤ 26	25 (2.17)
	5	29 (2.9)	29	≤ 29	27 (2.17)
	6	32 (2.9)	32	≤ 32	27 (2.17)
	7	35 (2.9)	32 (by (2.3) on $Y_6(12)$)	≤ 35	28 (2.17)
	8	38 (2.9)	35 (by (2.3) on $Y_4(14)$)	≤ 37	28 (2.17)
	9	41 (2.9)	34 (by (2.3) on $Y_5(13)$)	≤ 37	29 (2.17)
19	2	22 (2.9)	22	22	22 (2.15)
	3	25 (2.11)	25	25	25 (2.17)
	4	29 (2.11)	28 (by (2.2) on $Y_4(11)$)	≤ 29	27 (2.17)
	5	30 (2.9)	30	≤ 30	28 (2.17)
	6	33 (2.9)	32 (by (2.2) on $Y_6(12)$)	≤ 33	29 (2.17)
	7	36 (2.9)	36	≤ 36	29 (2.17)

	8	39 (2.9)	35 (by (2.3) on $Y_6(13)$)	≤ 39	30 (2.17)
	9	42 (2.9)	38 (by (2.6) on $Y_9(20)$)	≤ 40	30 (2.17)
	10	45 (2.9)	37 (by (2.3) on $Y_5(14)$)	≤ 41	31 (2.17)
20	2	23 (2.9)	23	23	23 (2.15)
	3	26 (2.11)	26	26	26 (2.17)
	4	30 (2.11)	30	≤ 30	28 (2.17)
	5	33 (2.11)	31 (by (2.2) on $Y_5(12)$)	≤ 33	30 (2.17)
	6	34 (2.9)	34	≤ 34	30 (2.17)
	7	37 (2.9)	36 (by (2.2) on $Y_7(13)$)	≤ 37	31 (2.17)
	8	40 (2.9)	40	≤ 40	31 (2.17)
	9	43 (2.9)	38 (by (2.3) on $Y_6(14)$)	≤ 43	32 (2.17)
	10	46 (2.9)	40 (by (2.3) on $Y_5(15)$)	≤ 46	32 (2.17)

Table 2.1: Summary of known bounds on $V_i(n)$ for $n \leq 20$. Entries marked by an asterisk in the $V_i(n)$ column are new exact values obtained by our program. Entries prefixed with $\leq$ are computer program generated upper bounds by Gasarch et al. [9] and Oksanen [18].

2.3 Related Problems and Directions

We would like to concentrate on some of the problems listed below.

1. Prove that $V_i(n) \leq V_{i+1}(n)$ for $1 \leq i \leq \lfloor n/2 \rfloor$. Note that the upper bound (2.9) by Hadian and Sobel [10] gives $V_6(13) \leq 26$ but $V_7(13) \leq 24$. With Kirkpatrick's improvement [13] given in (2.11), $V_6(13) \leq 25$. The upper bound for $V_6(13)$ is still higher than it for $V_7(13)$. Similarly, it is unknown whether $Y_i(n) \leq Y_{i+1}(n)$. Proving this inequality in general could improve some bounds given by Yao's hypothesis. For example, $Y_8(17) \leq 31$, but for $Y_7(17)$, we only know an upper bound of 32 which can be reduced to $Y_7(17) \leq 31$ if the above inequality holds.

2. Determine $V_3(n)$ exactly (or further reduce the remaining gap); see [16, Ch. 5.3.3] for

current best bounds and [15] for a new development.

3. A variant of the selection problem asks to find the i smallest elements of X as a set (i.e., the order among these i elements are not necessarily determined). Let $U_i(n)$ be the minimum number of comparisons needed in the worst case to solve this problem. Clearly, $U_i(n) \leq V_i(n)$. Similar to inequality (2.8), we can first apply $V_i(n-1)$ comparisons on an arbitrary subset of X with $n-1$ elements to find the ith smallest one. Then comparing the remaining element with the ith smallest element gives the desired set. So $U_i(n) \leq V_i(n-1)+1$. Hadian and Sobel [10] conjectured that this holds with equality for every pair (n,i). This conjecture is still unsettled after over 40 years.

4. Determine the validity status of Yao's hypothesis. In particular, is there an algorithm for finding the median of n elements in at most $2.49n + o(n)$ comparisons under Yao's hypothesis? The current best upper bound, $2.50n + O(\log n)$ comparisons, is due to Schönhage, Paterson, and Pippenger [20]. The existence of such an algorithm using at most $2.40n + o(n)$ comparisons would already have major consequences and would invalidate a conjecture of Paterson [19] who tentatively placed the comparison complexity for finding the median of n elements at about $\log_{4/3} n = 2.4094\ldots n$ comparisons.

5. A few target values under investigation are: $V_7(16) \overset{?}{=} 30$, $V_6(17) \overset{?}{=} 31$, $V_4(19) \overset{?}{=} 29$. Equality in any of them would disprove Yao's hypothesis.

Bibliography

[1] S. W. Bent and J. W. John, Finding the median requires $2n$ comparisons, *Proceedings of the 17th Annual ACM Symposium on Theory of Computing* (STOC 1985), ACM, 1985, pp. 213–216.

[2] M. Blum, R. W. Floyd, V. Pratt, R. L. Rivest, and R. E. Tarjan, Time bounds for selection, *Journal of Computer and System Sciences* **7(4)** (1973), 448–461.

[3] K. Chen and A. Dumitrescu, Select with groups of 3 or 4, *Proceedings of the 29th International Symposium on Algorithms and Data Structures*, (WADS 2015), Victoria, Canada, August 2015, Vol. 9214 of LNCS, Springer, pp. 189–199. Also available at `arXiv.org/abs/1409.3600`.

[4] D. Dor, J. Håstad, S. Ulfberg, and U. Zwick, On lower bounds for selecting the median, *SIAM Journal on Discrete Mathematics* **14(3)** (2001), 299–311.

[5] D. Dor and U. Zwick, Finding the αnth largest element, *Combinatorica* **16(1)** (1996), 41–58.

[6] D. Dor and U. Zwick, Median selection requires $(2+\epsilon)n$ comparisons, Technical report 312/96, Department of Computer Science, Tel Aviv University, April 1996.

[7] D. Dor and U. Zwick, Selecting the median, *SIAM Journal on Computing* **28(5)** (1999), 1722–1758.

[8] F. Fussenegger and H. N. Gabow, A counting approach to lower bounds for selection problems, *Journal of ACM* **26(2)** (1979), 227–238.

[9] W. Gasarch, W. Kelly, and W. Pugh, Finding the ith largest of n for small i, n, *SIGACT News* **27(2)** (1996), 88–96.

[10] A. Hadian and M. Sobel, Selecting the t-th largest using binary errorless comparisons, *Combinatorial Theory and Its Applications* **4** (1969), 585–599.

[11] M. Hofri, Optimal selection and sorting via dynamic programming, *ACM Journal of Experimental Algorithmics* **18(2)** (2013), Article 2.3, 14 pages.

[12] L. Hyafil, Bounds for selection, *SIAM Journal on Computing* **5(1)** (1976), 109–114.

[13] D. G. Kirkpatrick, Topics in the complexity of combinatorial algorithms, *Technical report 74*, Dept. of Computer Science, Univ. of Toronto, 1974.

[14] D. G. Kirkpatrick, A unified lower bound for selection and set partitioning problems, *Journal of ACM* **28(1)** (1981), 150–165.

[15] D. G. Kirkpatrick, Closing a long-standing complexity gap for selection: $V_3(42) = 50$, in *Space-Efficient Data Structures, Streams, and Algorithms – Papers in Honor of J. Ian Munro on the Occasion of His 66th Birthday* (A. Brodnik, A. López-Ortiz, V. Raman, and A. Viola, editors), LNCS vol. 8066, Springer, 2013, pp. 61–76.

[16] D. E. Knuth, *The Art of Computer Programming, Vol. 3: Sorting and Searching*, 2nd edition, Addison–Wesley, Reading, Massachusetts, 1998.

[17] B. D. McKay, and A. Piperno, Practical Graph Isomorphism, II, *Journal of Symbolic Computation* **60** (2014), 94–112.

[18] K. Oksanen, another implementation of the brute force method for finding lower bounds, `http://www.cs.hut.fi/~cessu/selection/`

[19] M. Paterson, Progress in selection, *Proceedings of the 5th Scandinavian Workshop on Algorithm Theory (SWAT 1996)*, LNCS vol. 1097, Springer, 1996, pp. 368–379.

[20] A. Schönhage, M. Paterson, and N. Pippenger, Finding the median, *Journal of Computer and System Sciences* **13(2)** (1976), 184–199.

[21] F. Yao, On lower bounds for selection problems, Technical report MAC TR-121, Massachusetts Institute of Technology, Cambridge, 1974.

[22] C. K. Yap, New upper bounds for selection, *Communications of the ACM* **19(9)** (1976), 501–508.

[23] U. Zwick, personal communication, September 2014.

Chapter 3

Multiparty Selection

3.1 Introduction

Given a sequence A of n numbers and an integer (selection) parameter $1 \leq i \leq n$, the selection problem asks to find the i-th smallest element in A. If the n elements are distinct, the i-th smallest is larger than $i-1$ elements of A and smaller than the other $n-i$ elements of A. By symmetry, the problems of determining the i-th smallest and the i-th largest are equivalent. Together with sorting, the selection problem is one of the most fundamental problems in computer science. Whereas sorting trivially solves the selection problem in $O(n \log n)$ time, Blum et al. [7] gave an $O(n)$-time algorithm for this problem.

The selection problem, and computing the median in particular, are in close relation with the problem of finding the quantiles of a set. The h-th *quantiles* of an n-element set are the $h-1$ order statistics that divide the sorted set in h equal-sized groups (to within 1); see, e.g., [10, p. 223]. The h-th quantiles of a set can be computed by a recursive algorithm running in $O(n \log h)$ time.

The selection problem, determining the median in particular, has been also considered from the perspective of communication complexity in the *two-party* model introduced by Andrew Yao [38]. Suppose that Alice and Bob hold subsets A and B of $[n] = \{1, 2, \ldots, n\}$, respectively, and wish to determine the median of the multiset $A \cup B$ while keeping their communication close to a minimum. Several classic protocols going back to 1980s achieve this task by exchanging $O(\log^2 n)$ bits [29, 36]. The communication complexity for this task has been subsequently reduced to $O(\log n)$ bits [9, 29, 31, 35].

Mediocre elements. Following Frances Yao [39], an element is said to be (i, j)-*mediocre* if it is neither among the top (i.e., largest) i nor among the bottom (i.e., smallest) j of a totally ordered set S of n elements. As remarked by Yao, finding a mediocre element is closely related to finding the median, in the sense that the common goal is selecting an element that is not too close to either extreme. In particular, (i, j)-mediocre elements where $i = \lfloor \frac{n-1}{2} \rfloor$, $j = \lfloor \frac{n}{2} \rfloor$ (and symmetrically exchanged), are medians of S. Previous work on *approximate selection* (in this sense) includes [5, 16].

In Section 3.3 we provide a protocol to find a mediocre element near the median among k players with communication complexity $O(k \log n)$. To our best knowledge, this is the first result on the mediocre element finding problem, in terms of communication complexity. In Section 3.4 we outline a scenario in which computing a mediocre element near the median in the two-party model can be accomplished with communication complexity $O(1)$—which is very attractive.

Background and related problems. Due to its primary importance, the selection problem has been studied extensively; see for instance [2, 6, 11, 13–15, 19–26, 34, 37, 40]. A comprehensive review of early developments in selection is provided by Knuth [28]. The reader is also referred to dedicated book chapters on selection, such as those in [1, 4, 10, 12, 27] and the more recent articles [8, 17], including experimental work [3].

In many applications (e.g., sorting), it is not important to find an exact median, or any other precise order statistic, for that matter, and an approximate median suffices [18]. For instance, quick-sort type algorithms aim at finding a balanced partition rather quickly; see e.g., [22, 32, 33].

Studying the multiparty communication complexity of exact and approximate selection is relevant in the context of distributed computing [9, 31, 36, 38].

Our results. Our main results are summarized in the three theorems stated below. We first study the communication complexity of finding the median in the multiparty setting. In this model we assume that every message by one of the players is seen by all the players (i.e., it is a broadcast); as in [29, p. 83].

Theorem 3.1. *For $i = 1, \dots, k$, let player i hold a sequence (i.e., a multiset) A_i whose support is a subset of $[n]$ and $|A_i| = O(poly(n))$. There is a deterministic protocol for finding the median of $\uplus_{i=1}^{k} A_i$ (i.e., their multiset sum) with $O(k \log^2 n)$ communication complexity.*

We then study the communication complexity of finding an approximate median in the multiparty setting (under slightly stronger assumptions on the input sets).

Theorem 3.2. *Let $\alpha = p/q$, where $p, q \in \mathbb{N}$, $p < q/2$, q is fixed and $0 < c \leq 1$ be a positive constant. For $i = 1, \dots, k$, let player i hold a set $A_i \subset [n]$ that is disjoint from any other player's set. Assume that $t = |\cup_{i=1}^{k} A_i| \geq cn$. Put $\ell = \lceil \log \frac{2q}{c} \rceil$. Then an $(\alpha t, \alpha t)$-mediocre element of $\cup_{i=1}^{k} A_i$ can be found with $O(\ell \cdot k \log n) = O(k \log n)$ communication complexity.*

In particular, a $(t/3, t/3)$-mediocre element, or a $(0.49\,t, 0.49\,t)$-mediocre element, among k players can be determined with $O(k \log n)$ communication complexity.

In the final part of this chapter, somewhat surprisingly, we show that (under suitable additional assumptions and a somewhat relaxed requirement) the communication complexity of finding a mediocre element in the vicinity of the median is bounded from above by a constant and is therefore independent of n.

Theorem 3.3. *Let $\alpha = p/q$, where $p, q \in \mathbb{N}$, $p < q/2$, q is fixed and $0 < c \leq 1$ be a positive constant. Let Alice and Bob hold disjoint sets A and B of elements from $[n]$, where $s = |A| \leq |B| = m$. Let $t = s + m$ denote the total number of elements in $A \cup B$, where $t \geq cn$. Assume that t, c, and α are known to both players. Put $h = \lceil \frac{2q}{q-2p} \rceil$ and $\ell = \lceil \log \frac{12h}{c} \rceil$.*

Then an $(\alpha t, \alpha t)$-mediocre element can be found (by at least one player) with $O(\ell \log h) = O(1)$ communication complexity. If both players return, each element returned is $(\alpha t, \alpha t)$-mediocre; the elements found by the players need not be the same.

In particular, a $(t/3, t/3)$-mediocre element, or a $(0.49\,t, 0.49\,t)$-mediocre element, between 2 players can be determined (by at least one player) with $O(1)$ communication complexity. A simple example that falls under the scenario in Theorem 3.3 is one where A consists of distinct odd numbers and B consists of distinct even numbers. It is worth noting that since $m/2t \geq 1/4$, if $\alpha < 1/4$, the median of B is guaranteed to be an $(\alpha t, \alpha t)$-mediocre element of $A \cup B$. In this case, no communication is needed.

Preliminaries. A simple but effective procedure reduces the selection problem for finding the i-th smallest element out of n to one for finding the median in a slightly larger sequence. The target is the i-th smallest element in an input sequence A of size n. Assume first that $i < n/2$; in this case pad the input A with $n - 2i$ elements that are less than or equal to the minimum in the input sequence; call A' resulting sequence. Note that $|A'| = n + (n - 2i) = 2(n - i)$. It suffices to observe that the median of A' is the i-th smallest element in A: indeed, $n - 2i + i = n - i$, as required. The case $i > n/2$ is symmetric; in this case pad the input A with $2i - n$ elements that are larger than or equal to the maximum in the input sequence; call A' resulting sequence. Note that $|A'| = n + (2i - n) = 2i$. Observe that the median of A' is the i-th smallest element in A, as required. We therefore restrict our attention to the median selection problem.

Notation. Without affecting the results, the floor and ceiling functions are omitted in some instances where they are not essential. For example, we frequently write the αn-th element instead of the more precise $\lfloor \alpha n \rfloor$-th element. Unless specified otherwise, all logarithms are in base 2.

For an s-bit number x and a positive integer ℓ, where $s \geq \ell$, $\texttt{prefix}_\ell(x)$ denotes the *ℓ-bit binary prefix* of x, i.e., the number formed by the first (i.e., most significant) ℓ bits of x.

If x belongs to a sorted list and is not the minimum, $\texttt{pred}(x)$ denotes its predecessor. If x belongs to a sorted list and is not the maximum, $\texttt{succ}(x)$ denotes its successor.

3.2 Exact selection

In this section we prove Theorem 3.1. First, we set up the problem in the context of two-party communication complexity; we start with some background. In this section, each player's input is allowed to contain duplicates. Following the literature, we refer to these (potential) multisets as sets, and the union operation should be understood as multiset sum [29, Example 1.6, p. 6]. (An equivalent formulation is *merging of sequences.*)

3.2.1 Two players

Alice and Bob hold multisets A and B whose supports are subsets of $[n] = \{1, 2, \dots, n\}$, respectively. It is assumed that $|A|, |B| = O(\text{poly}(n))$. (In a standard setup [29, Example 1.6, p. 6], A and B are subsets of $[n]$; here we extend this setup for potentially larger multisets.) The median of the multiset $A \cup B$ is denoted by $\xi = \texttt{Med}(A, B)$; as usual, the median of X is the $\lceil |X|/2 \rceil$-th smallest element of X.

There is a simple binary-search type protocol due to M. Karchmer that takes $O(\log^2 n)$ bits of communication; see [29, Example 1.6, p. 6]. At each round Alice and Bob have an interval $[i, j]$, $i, j \in \mathbb{N}$, that contains the median. They halve the interval (repeatedly) by deciding whether the median is less than, equal to, or larger than $m = (i + j)/2$. This is done by Alice sending to Bob the number of elements in A that are less than m, equal to m, and larger than m, using $O(\log n)$ bits. Bob can now determine whether the median is less than, equal to, or larger than m, and sends this information to Alice using $O(1)$ bits. The protocol has $O(\log n)$ rounds, each requiring $O(\log n)$ bits of communication, so the overall communication complexity is $O(\log^2 n)$.

An alternative binary-search type protocol that takes $O(\log^2 n)$ bits of communication, also due to Karchmer [29, p. 168], works as follows. Assume, without loss of generality that $|A| = |B|$ and that the common size is a power of 2: this can be achieved by exchanging the sizes of their inputs ($O(\log n)$ bits) and padding them with the appropriate number of the minimal element (1) and the maximal element (n). The protocol works in rounds. During the protocol, Alice maintains a set $A' \subset A$ of elements that may still be the median (initially $A' = A$) and Bob maintains a set $B' \subset B$ of elements that may still be the median (initially $B' = B$). At each round, Alice sends Bob the value a, which is the median of A', and Bob sends Alice the value b, which is the median of B'. At this point we have $\min(a, b) \leq \xi \leq \max(a, b)$. If $a < b$, then Alice discards the lower half of A' (note that a is part of it) and Bob discards the upper half of B'. If $b < a$, then Bob discards the lower half of B' (note that b is part of it) and Alice discards the upper half of A'. In either case, this operation maintains the median of $A' \cup B'$ as the desired median of $A \cup B$. It should be noted that the size of $A' \cup B'$ is reduced (exactly) by a factor of 2. If $a = b$, this value is the median, and if $|A'| = |B'| = 1$, then the smaller number is the median. The protocol has $O(\log n)$ rounds, each requiring $O(\log n)$ bits of communication, and so the communication complexity is $O(\log^2 n)$.

The communication complexity of finding the median can be further reduced. A subtle refinement of the above protocol, due to Karchmer [29, Example 1.7, p. 6 and p. 168], and revised by Gasarch [30], works with $O(\log n)$ communication complexity: its key idea is to make comparisons in a bit-by-bit manner, but this requires careful bookkeeping of the progress and here we omit the technical details.

We next describe a different (folklore) protocol, running with $O(\log n)$ communication complexity, that we find simpler and subsequently refine for computing a mediocre element. The protocol implements a binary-search strategy and works in rounds. Alice maintains a set $A' \subset A$ of elements that may still be the median (initially $A' = A$) and Bob maintains a set $B' \subset B$ of elements that may still be the median (initially $B' = B$). Alice and Bob compute the medians of their current inputs (a and b, respectively). At this point we have

$\min(a,b) \leq \xi \leq \max(a,b)$. Alice and Bob aim to determine the order relation between a and b in order to halve their input in an appropriate manner.

The protocol avoids sending these $\log n$-bit numbers at each round by avoiding making a direct comparison between a and b. The players have an interval $[i,j]$, $i,j \in \mathbb{N}$, that contains the median (initially, $[i,j] = [1,n]$). The medians a and b are compared to the middle element $h = \lfloor (i+j)/2 \rfloor$, If $a = b = h$, this element is the median of $A \cup B$ and the protocol terminates. Otherwise, if a and b are split by h, i.e., $a \leq h \leq b$ or $b \leq h \leq a$, then (by transitivity of $\leq$), the relation between a and b is determined, and Alice and Bob halves their input accordingly (as in the earlier $O(\log^2 n)$ protocol). Otherwise, if a and b are on the same side of h, i.e., $a,b \leq h$ or $h \leq a,b$. For example, in the first case, the elements in the lower half of $A' \cup B'$ are $\leq h$ and the same holds for the median of $A' \cup B'$. As such, both players shrink their common interval $[i,j]$ by (roughly) half: the resulting interval is $[i,h]$ or $[h,j]$, respectively. The sets A' and B' remain unchanged. Alice and Bob communicate each of the outcomes of the above tests in $O(1)$ bits. Each halving operation for A' and B' maintains the property that $\xi = \texttt{Med}(A \cup B) = \texttt{Med}(A' \cup B')$.

Let $\ell = \lceil \log n \rceil$. Note that after $2\ell - 1$ tests, either Alice and Bob hold singleton sets (i.e., $|A'| = |B'| = 1$), or the common interval $[i,j]$ consists of a single integer $i = j$. If $|A'| = |B'| = 1$, the smaller number is the median (or either, for equality), whereas if $i = j$, this number is the median. The number of bits exchanged before the last round of the protocol is $O(\log n)$ and is $O(\log n)$ in the last round. The resulting communication complexity is $O(\log n)$.

3.2.2 k players

In this subsection we show the protocol that proves Theorem 3.1. It is worth noting that the number of players, k is independent of n. The protocol maintains the invariant: the median of $\cup_{i=1}^{k} A_i$ in one round is the same for the updated sets in the next round. It is possible that the number of sets drops from k to a lower number; the protocol remains unchanged until the value $k = 2$ is reached, when the respective players apply the protocol in Subsection 3.2.1; recall that padding with extra elements may be needed. If the value $k = 1$ is reached, the remaining player computes the median in his/her own set and the game ends.

Initially, each player sorts his/her input set locally. The sorted order is used by each player in the pruning process, and if such action occurs, the sorted order is locally maintained. Each set pruning discards elements at one of the two ends of the chain (either low elements below some threshold, or high elements above some threshold).

The protocol roughly halves the size of at least one of the current participating sets; more precisely, for some $X \in \{A_1, \ldots, A_k\}$, we have $|X'| \leq \lfloor |X|/2 \rfloor$ by the end of each round. Since the size of each set is initially $O(\text{poly}(n))$, the size of each of the k sets drops to 0 in at most $O(\log n)$ iterations and consequently, the number of rounds is at most $O(k \log n)$. (Padding with extra elements when $k = 2$ is reached conforms with this bound.)

Each round of the protocol works as follows. Each player (locally) finds the median of his/her current set: $x_i \in A_i$, $i = 1, \ldots, k$. The following scheme regarding medians is used: assume that there are x sets of even size and y sets of odd size in the current round, where $x + y = k$; for the x sets of even size the first $\lceil x/2 \rceil$ use the lower median and the remaining $\lfloor x/2 \rfloor$ use the upper median (in some fixed, e.g., alphabetical, order). The idea of intermixing upper and lower medians is also present in [8]. (A scheme that uses only lower medians or only upper medians fails to guarantee that the median of the union is maintained after pruning, for instance if $k = 3$ and all three sets have even size; the smallest example of this kind is $|A_1| = |A_2| = |A_3| = 2$.)

In the first round, each player posts his/her median and set size on the communication board; this involves $O(k \log n)$ bits of communication. In the remaining rounds, two players whose sets got pruned (as further explained below) need to update their median on the communication board. Depending on the parities of the sets of these two players before and after the pruning, at most one more player may need to update his/her median to maintain the balanced scheme adopted earlier which requires $\lceil x/2 \rceil$ use the lower median and the remaining $\lfloor x/2 \rfloor$ use the upper median. Therefore, in each round, the communication complexity is $O(\log n)$.

All players are now able to determine the sorted order of the k medians. For simplicity, assume that after relabeling, this order is

$$x_1 \leq x_2 \leq \ldots \leq x_k. \tag{3.1}$$

It is convenient to refer to the players holding the minimum and maximum of these medians as Alice and Bob and to their corresponding sets as A and B: $x_A \equiv x_1$ and $x_B \equiv x_k$ (this relabeling is only done for the purpose of analysis).

Let P denote the poset made by the k chains $A_1, \ldots, A_k$, together with the relations in (3.1). Write $a = |A|$, $b = |B|$, and $t = \sum_{i=1}^{k} |A_i|$. The player holding the smaller set between Alice and Bob is in charge of the pruning operation in the current round: the same number of elements is discarded by Alice and Bob as specified below. Refer to Figure 3.1.

If $\min(a, b) = a$, Alice discards $\lceil a/2 \rceil$ elements in A (all $x \leq x_A$ when a is odd or x_A is the lower median, or all $x < x_A$ when x_A is the upper median), and Bob discards the highest $\lceil a/2 \rceil$ elements in B. Such operation is *charged* to Alice. Otherwise, if $\min(a, b) = b$, Bob discards $\lceil b/2 \rceil$ elements in B (all $x \geq x_B$ when b is odd or x_B is the upper median, or all $x > x_B$ when x_B is the lower median), and Alice discards the lowest $\lceil b/2 \rceil$ elements in A. Such operation is *charged* to Bob. It is worth noting that this scheme is feasible: i.e., if the indicated player discards the specified number of elements, the other player can also discard the same number of elements. Then the protocol continues with the next round. Each player keeps track of the players that are still in the game and their set cardinalities, as these can be deduced from the actions of the algorithm.

It remains to show that the same number of elements is discarded from each side of the median in each round. Let u be the number of elements in P that are above the highest discarded element of A, and v be the number of elements in P that are below the lowest discarded element of B. By slightly abusing notation, let k denote the number of players in the current round of the protocol (which may differ from the initial number). Specifically we prove the following.

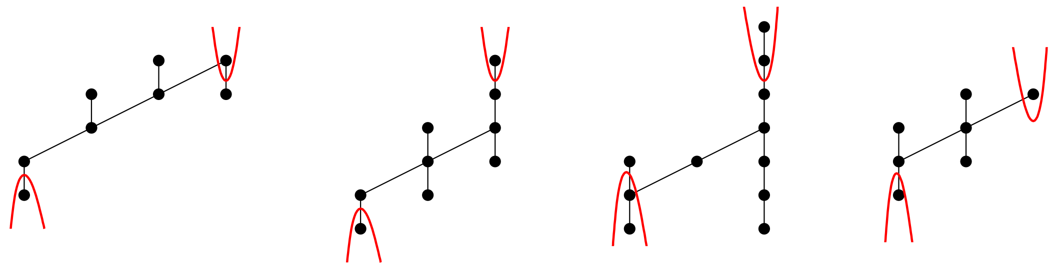


Figure 3.1: Pruning the poset P in the protocol for finding the median; Alice is the leftmost player and Bob is the rightmost player. (i) $k = 4$, $t = 8$, $u = 6$, $v = 5$; operation is charged to Alice. (ii) $k = 3$, $t = 9$, $u = 6$, $v = 7$; operation is charged to Alice. (iii) $t = 11$, $u = 6$, $v = 8$; operation is charged to Alice. (iv) $t = 7$, $u = 5$, $v = 4$; operation is charged to Bob.

Lemma 3.4. *Consider a round of the protocol and assume that $k \geq 3$ and $t = \sum_{i=1}^{k} |A_i|$. The following inequalities for u and v hold: $u \geq \lceil \frac{t+1}{2} \rceil$ and $v \geq \lceil \frac{t}{2} \rceil$.*

Proof. For u, we start by including $|A_i|/2$ corresponding to the upper half elements in the set A_i, for $i = 1, \ldots, k$; this contributes $t/2$ to the sum. In addition we add $1/2$ for each set of odd size, thus $y/2$ over all odd sets. Then we add 1 for each set of even size that uses the lower median, thus $\lceil x/2 \rceil$ over all even sets. This procedure overcounts by 1 if the median x_A is the highest discarded element of A. Therefore, we have

$$u \geq \frac{t}{2} + \frac{y}{2} + \left\lceil \frac{x}{2} \right\rceil - 1 \geq \frac{t}{2} + \frac{y}{2} + \frac{x}{2} - 1 = \frac{t+x+y-2}{2} = \frac{t+k-2}{2} \geq \frac{t+1}{2}.$$

Similarly, for v, we start by including $|A_i|/2$ corresponding to the lower half elements in the set A_i, for $i = 1, \ldots, k$; this contributes $t/2$ to the sum. In addition we add $1/2$ for each set of odd size, thus $y/2$ over all odd sets. Then we add 1 for each set of even size that uses the upper median, thus $\lfloor x/2 \rfloor$ over all even sets. This procedure overcounts by 1 if the median x_B is the lowest discarded element of B. Therefore, we have

$$v \geq \frac{t}{2} + \frac{y}{2} + \left\lfloor \frac{x}{2} \right\rfloor - 1 \geq \frac{t}{2} + \frac{y}{2} + \frac{x-1}{2} - 1 = \frac{t+x+y-3}{2} = \frac{t+k-3}{2} \geq \frac{t}{2}.$$

Since both u and v are integers, we have thereby proved that $u \geq \lceil \frac{t+1}{2} \rceil$ and $v \geq \lceil \frac{t}{2} \rceil$, as required. □

Proof of Theorem 3.1. By Lemma 3.4, all the elements discarded from A are below the median (of the union), and all elements discarded from B are above the median. Thus in each round, the protocol preserves the median and discards the same number of elements from each side of it. This proves the invariant of the protocol. Since the protocol takes $O(k \log n)$ rounds and the communication complexity of each round is $O(\log n)$, the overall communication complexity is $O(k \log^2 n)$, as claimed. □

3.3 Approximate selection with k players

In this section we consider the problem of finding an $(\alpha t, \alpha t)$-mediocre element among k players, where $\alpha \in (0, 1/2)$ is a fixed constant. Recall that in the setting of Theorem 3.2, the sets A_i, $i = 1, \ldots, k$, are pairwise disjoint. But we do *not* assume that they have the same cardinality.

The protocol works in rounds. Let $a_1 = 1$ and $b_1 = n$; and note that $[a_1, b_1]$ contains the median m, i.e., the $\lceil t/2 \rceil$-th smallest element of $\cup_{i=1}^k A_i$. For round $j = 1, 2, \ldots$, the interval $[a_{j+1}, b_{j+1}]$ is obtained from the interval $[a_j, b_j]$ by halving while maintaining the following:

Invariant: For $j = 1, 2, \ldots$, the interval $[a_j, b_j]$ contains the median m.

Equivalently, the invariant can be stated as follows. For $j = 1, 2, \ldots$,

- the number of elements in $\cup_{i=1}^k A_i$ that are $\leq a_j$ is less than $\lceil t/2 \rceil$, and
- the number of elements in $\cup_{i=1}^k A_i$ that are $\leq b_j$ is at least $\lceil t/2 \rceil$.

Specifically, in round j, let

$$c_j = \left\lfloor \frac{a_j + b_j}{2} \right\rfloor.$$

Each player communicates the number of elements in his/her set that are $\leq c_j$. Since there are k players, this takes $O(k \log n)$ bits. Once this is done, each player can compute independently (by adding the k individual counts) the total number of elements in $\cup_{i=1}^k A_i$ that are $\leq c_i$. If the number is less than $\lceil t/2 \rceil$, then we set $[a_{j+1}, b_{j+1}] := [c_j, b_j]$, otherwise, i.e., the number is at least $\lceil t/2 \rceil$, then we set $[a_{j+1}, b_{j+1}] := [a_j, c_j]$. This setting maintains the invariant.

The protocol repeatedly halves the current interval until

$$b_j - a_j \leq \left(\frac{1}{2} - \alpha\right) t. \tag{3.2}$$

When this occurs, since $\cup_{i=1}^k A_i$ consists of distinct elements, $[a_j, b_j]$ contains a continuous range of no more than $\left(\frac{1}{2} - \alpha\right) t$ elements of $\cup_{i=1}^k A_i$, with m being one of them. If $(0.5-\alpha)t < 1$, then the protocol stops when $b_j - a_j = 1$ and returns b_j as the median.

Let z be any element of $\cup_{i=1}^k A_i$ contained in $[a_j, b_j]$. (The protocol will return one such element, as explained below.) Observe that

$$\frac{t}{2} - \left(\frac{1}{2} - \alpha\right) t \leq \operatorname{rank}_{\cup A_i}(z) \leq \frac{t}{2} + \left(\frac{1}{2} - \alpha\right) t, \text{ or}$$

$$\alpha t \leq \operatorname{rank}_{\cup A_i}(z) \leq (1-\alpha)\, t. \tag{3.3}$$

The number of halving rounds needed to achieve the interval-length in (3.2) is at most

$$\begin{aligned}\left\lceil \log \frac{n}{\left(\frac{1}{2} - \alpha\right) t} \right\rceil \leq \left\lceil \log \frac{n}{\left(\frac{1}{2} - \alpha\right) cn} \right\rceil = \left\lceil \log \frac{1}{\left(\frac{1}{2} - \alpha\right) c} \right\rceil = \left\lceil \log \frac{2q}{(q-2p)c} \right\rceil \\ \leq \left\lceil \log \frac{2q}{c} \right\rceil = \ell = O(1).\end{aligned}$$

In each round, the k players communicate their counts, $O(k \log n)$ bits in total. Each player independently computes the total count for the midpoint of the current interval, and all players take the same decision on how to set the next interval in the halving process (with no further communication needed).

In the last round (i.e., when inequality (3.2) is satisfied), the players report in turn. If the player does not hold any element in the interval $[a_j, b_j]$, he/she outputs a zero bit and the report continues; otherwise the player outputs such an element (from his/her set) in $O(\log n)$ bits and the protocol ends. The output element is a valid choice, as justified by (3.3).

The total communication complexity is therefore $O(\ell\, k \log n) = O(k \log n)$ bits, as claimed. This concludes the proof of Theorem 3.2. □

3.4 Approximate selection with two players under special conditions

Let $t = s + m$ denote the total number of elements in $A \cup B$. Here we consider the problem of finding an $(\alpha t, \alpha t)$-mediocre element between two players, where $\alpha \in (0, 1/2)$ is a fixed constant. The protocol described in Subsection 3.2.1 immediately yields the following.

Corollary 3.5. *The deterministic communication complexity of finding an $(\alpha t, \alpha t)$-mediocre element in $A \cup B \subset [n]$, where $t = |A| + |B|$ and $\alpha \in (0, 1/2)$ is a fixed constant, is $O(\log n)$.*

Interestingly enough, this communication complexity can be brought down to a constant under slightly stronger assumptions: (i) A and B have no duplicates or common elements, and (ii) $|A \cup B| \geq cn$, for some constant $c > 0$; and a somewhat relaxed requirement: at least one of the players returns an element to the process that has invoked his/her service; each element returned is $(\alpha t, \alpha t)$-mediocre. Note that this is a natural relaxation — if the set of one player does not contain any suitable element, it is impossible to communicate the final answer to this player within $O(1)$ complexity.

A natural protocol to consider would be to choose one of the median-finding protocols and execute a constant number of rounds from it. However, this seemingly promising idea does not appear to work. It is possible that one of the two sets, say A, does not contain any desired elements, namely $(\alpha t, \alpha t)$-mediocre for the given α and so at the end of the modified protocol only B' contains desired elements (and not A'). More importantly, the players apparently have no indication of which player is the lucky one. We therefore resort to a different idea of using quantiles (more precisely, a sampling technique with a similar effect).

Proof of Theorem 3.3. We may assume, without loss of generality that n and $1/c$ are powers of 2 (in particular, $4n$ is also a power of 2). For $n < 8q^2/c$ Alice and Bob use the earlier $O(\log n)$-protocol for finding the median; we therefore subsequently assume that $n \geq 8q^2/c$. In particular, since $q \geq 3$, we have $n \geq 24q/c$. We further assume, without loss of generality that $|A| = |B| = m$: this can be achieved by padding the smaller size set with the appropriate numbers of small elements and large elements as described below. In particular, the padding elements need also be distinct. (It is *not* assumed that the common size is a power of 2: since our protocol does not exactly halve the current set of each player at each round, such an assumption would be of no use.)

To illustrate the padding process for arbitrary set sizes, we may assume without loss of generality that the given input satisfies: $s = |A| \leq |B| = m$. Recall that s and m are known to both players. We need to pad Alice's input with $m - \lceil \frac{m+s}{2} \rceil$ small elements and $\lceil \frac{m+s}{2} \rceil - s$ large elements. Alice and Bob replace their inputs by $A + n$ and $B + n$, respectively; as a result, the elements they hold are now in the range $\{n+1, \ldots, 2n\}$. Then Alice pads her input with $\{1, 2, \ldots, m - \lceil \frac{m+s}{2} \rceil\} \subset [n]$ and $\{2n+1, \ldots, 2n + \lceil \frac{m+s}{2} \rceil - s\} \subset [3n] \setminus [2n]$. (Note that $\lceil \frac{m+s}{2} \rceil - s = m - \lfloor \frac{m+s}{2} \rfloor$.) The resulting sets have the same size m and $A \cup B$ consists of distinct elements in the range $[3n] \subset [4n]$. By subtracting n, the element(s) returned by the protocol are shifted back to the original range $[n]$ in the end (without explicitly mentioning it there).

A and B below denote the (new) padded sets (of size m). Set $h = \lceil \frac{2q}{q-2p} \rceil$ (recall that

$\alpha = p/q$) and $\ell = \lceil \log \frac{12h}{c} \rceil$. By the assumption $n \geq 24q/c$ we have

$$cn \geq 24q \geq 12 \left\lceil \frac{2q}{q-2p} \right\rceil = 12h.$$

Let Q_A be the set consisting of the $i\lfloor m/h \rfloor$-th elements of A, for $i = 1, 2, \ldots, h$. Similarly, let Q_B be the set consisting of the $i\lfloor m/h \rfloor$-th elements of B, for $i = 1, 2, \ldots, h$. (These sets resemble the h-th quantiles of A and B). Note that $|Q_A| = |Q_B| = h$. Since A and B consist of pairwise distinct elements, between any two elements in Q_A (or Q_B), there are at least

$$\left\lfloor \frac{m}{h} \right\rfloor \geq \frac{m}{h} - 1 \geq \frac{t}{2h} - 1 \geq \frac{cn}{2h} - 1 \geq \frac{cn}{3h} \geq \frac{4n}{2^\ell}$$

elements. Represent each element x in Q_A (and Q_B) with $\log(4n) = \log n + 2$ bits; it follows that the elements in $\{\texttt{prefix}_\ell(x) : x \in Q_A\}$ are pairwise distinct; similarly the elements in $\{\texttt{prefix}_\ell(y) : y \in Q_B\}$ are pairwise distinct.

The protocol implements a binary-search strategy aimed at finding the median of $Q_A \cup Q_B$. Note that $|Q_A| = |Q_B| \leq h$. Alice maintains a set $Q'_A \subset Q_A$ of elements that may still be the median quantile (initially $Q'_A = Q_A$) and Bob maintains a set $Q'_B \subset Q_B$ of elements that may still be the median quantile (initially $Q'_B = Q_B$). The invariant $|Q'_A| = |Q'_B|$ will be maintained. At each round, Alice and Bob compute the medians of their current sets (x_A and x_B, respectively). If $\texttt{prefix}_\ell(x_A) < \texttt{prefix}_\ell(x_B)$ or $\texttt{prefix}_\ell(x_A) > \texttt{prefix}_\ell(x_B)$ the protocol continues with Alice and Bob halving their input as in the median-finding protocol. Specifically, if $\texttt{prefix}_\ell(x_A) < \texttt{prefix}_\ell(x_B)$ the protocol discards the $\lfloor |Q'_A|/2 \rfloor$ lower elements of Q'_A and the $\lfloor |Q'_B|/2 \rfloor$ upper elements of Q'_B. The equality case $\texttt{prefix}_\ell(x_A) = \texttt{prefix}_\ell(x_B)$ is addressed below. Observe that the above comparison can be resolved by exchanging ℓ bits in each round.

If $\texttt{prefix}_\ell(x_A) = \texttt{prefix}_\ell(x_B)$, and $|Q'_A| = |Q'_B| \geq 3$, we have $\texttt{prefix}_\ell(\texttt{pred}(x_A)) < \texttt{prefix}_\ell(x_B)$, and the protocol discards the $\lfloor (|Q'_A| - 1)/2 \rfloor$ lower elements of Q'_A and the $\lfloor (|Q'_B| - 1)/2 \rfloor$ upper elements of Q'_B. Note that this is a slight but important deviation from the standard median-finding protocol; it is aimed at handling prefix equality by discarding possibly one fewer element by each player. With this choice, the median of $Q_A \cup Q_B$ remains the median of $Q'_A \cup Q'_B$; and the invariant $|Q'_A| = |Q'_B|$ is maintained. Since the sets the players hold are almost halved at each round, the protocol terminates in $O(\log h)$ rounds, as specified below.

If $|Q'_A| = |Q'_B| = 2$, and $\texttt{prefix}_\ell(x_A) \neq \texttt{prefix}_\ell(x_B)$, the protocol continues with each player halving his/her own current set accordingly. If $|Q'_A| = |Q'_B| = 2$, and $\texttt{prefix}_\ell(x_A) = \texttt{prefix}_\ell(x_B)$, the protocol terminates with each player output his/her number (x_A and x_B, respectively). Observe that in this case, the median of $Q_A \cup Q_B$ is x_A or x_B and it will be shown below, see (3.7), that both elements are $(\alpha t, \alpha t)$-mediocre.

If $|Q'_A| = |Q'_B| = 1$ and $\texttt{prefix}_\ell(x_A) \neq \texttt{prefix}_\ell(x_B)$, the protocol terminates with the player that holds the smaller of x_A and x_B output that number. If $|Q'_A| = |Q'_B| = 1$ and $\texttt{prefix}_\ell(x_A) = \texttt{prefix}_\ell(x_B)$, the protocol terminates with each player output his/her number (x_A and x_B, respectively). It will be shown below, see (3.7), that both elements are $(\alpha t, \alpha t)$-mediocre.

Recall that $\ell = \lceil \log \frac{12h}{c} \rceil$. If $x, y \in [3n]$ and $\texttt{prefix}_\ell(x) = \texttt{prefix}_\ell(y)$ then

$$|x - y| \leq \frac{3n}{2^\ell} \leq \frac{cn}{4h} \leq \frac{t}{4h}. \tag{3.4}$$

Recall that the median of $Q_A \cup Q_B$ is in $Q'_A \cup Q'_B$ in the last round of the protocol. Since all elements are distinct, for x_A and x_B above, if $\texttt{prefix}_\ell(x_A) = \texttt{prefix}_\ell(x_B)$, Inequality (3.4) implies

$$|\text{rank}_{A\cup B}(x_A) - \text{rank}_{A\cup B}(x_B)| \leq \frac{t}{4h}. \tag{3.5}$$

Assume that the median of $Q_A \cup Q_B$ is $x_A \in Q_A$; then Alice returns x_A. In addition, if $\texttt{prefix}_\ell(x_A) = \texttt{prefix}_\ell(x_B)$, Bob also returns $x_B \in Q_B$. Since x_A is the median of $Q_A \cup Q_B$, it is the h-th smallest element of $Q_A \cup Q_B$. As such (by construction): (i) x_A is $\geq$ than at least

$$h\left\lfloor \frac{m}{h} \right\rfloor \geq h\left(\frac{m}{h} - 1\right) = m - h$$

elements of $A \cup B$; and similarly, (ii) x_A is $\leq$ than at least $m - h$ elements of $A \cup B$. Note that the median of $A \cup B$ has rank m and is the same as the median of the original union of the two sets. See Figure 3.2.

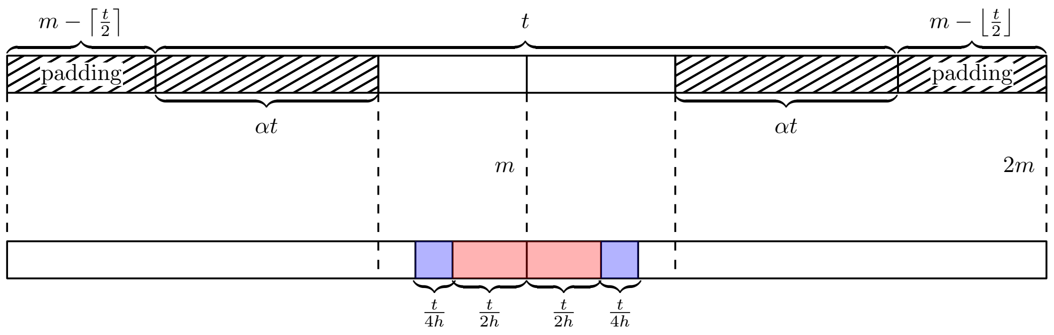


Figure 3.2: Above: Illustration of the original union of the two input sets with padding elements. The players need to find elements from the unshaded region in the middle. Below: The median x of $Q_A \cup Q_B$ lies within the red region. If the other player has an element y such that $\texttt{prefix}_\ell(y) = \texttt{prefix}_\ell(x)$, then y lies in the union of the red and blue regions, therefore it is also a valid output.

Observe that $h = \lceil \frac{2q}{q-2p} \rceil \leq 2q$ which yields $2h^2 \leq 8q^2 \leq cn \leq t$ (recall that $n \geq 8q^2/c$). This implies

$$|\text{rank}_{A\cup B}(x_A) - m| \leq h \leq \frac{t}{2h}. \tag{3.6}$$

Recall that if $\texttt{prefix}_\ell(x_A) = \texttt{prefix}_\ell(x_B)$, Bob also returns $x_B \in Q_B$ and Inequality (3.5) applies. From (3.5) and (3.6) we deduce that the rank of any output element z satisfies (recall that $t = s + m$):

$$|\text{rank}_{A\cup B}(z) - m| \leq \frac{t}{4h} + \frac{t}{2h} \leq \frac{t}{h} \leq \frac{(q-2p)t}{2q} = \left(\frac{1}{2} - \alpha\right) t. \tag{3.7}$$

As such, each output element z is an $(\alpha t, \alpha t)$-mediocre element of the original union of the two sets. The elements returned are x_A or x_B (or both). Alice may return x_A and Bob may return x_B to the processes that have invoked their service; the elements returned by the players could be different. Since $q = O(1)$, we have $h, \ell = O(1)$. The number of bits exchanged is $\ell + O(1) = O(1)$ in each of the $O(\log h)$ rounds of the protocol. The overall communication complexity is $O(\ell \log h) = O(1)$, as claimed. □

3.5 Conclusion

An obvious question is whether the three-party communication complexity of median computation can be reduced to $O(\log n)$. A more general question is whether the k-party communication complexity of median computation, $k \geq 3$, can be reduced to $O(k \log n)$. We believe that the answers to both questions are in the negative. Another interesting question regarding the two-party communication complexity of approximate selection is whether the conditions in Theorems 3.2 and 3.3 can be relaxed.

Bibliography

[1] Alfred V. Aho, John E. Hopcroft, and Jeffrey D. Ullman. *Data Structures and Algorithms.* Addison-Wesley, 1983.

[2] Miklós Ajtai, János Komlós, William L. Steiger, and Endre Szemerédi. Optimal parallel selection has complexity $o(\log \log n)$. *Journal of Computer and System Sciences*, 38(1):125–133, 1989.

[3] Andrei Alexandrescu. Fast deterministic selection. In Costas S. Iliopoulos, Solon P. Pissis, Simon J. Puglisi, and Rajeev Raman, editors, *Proceedings of the 16th International Symposium on Experimental Algorithms, SEA 2017, London, UK, June 21-23, 2017*, volume 75 of *LIPIcs*, pages 24:1–24:19. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2017.

[4] Sara Baase. *Computer algorithms - introduction to design and analysis.* Addison-Wesley, 1988.

[5] Sebastiano Battiato, Domenico Cantone, Dario Catalano, Gianluca Cincotti, and Micha Hofri. An efficient algorithm for the approximate median selection problem. In Gian Carlo Bongiovanni, Giorgio Gambosi, and Rossella Petreschi, editors, *Proceedings of the 4th Italian Conference on Algorithms and Complexity, CIAC 2000, Rome, Italy, March 2000*, volume 1767 of *Lecture Notes in Computer Science*, pages 226–238. Springer, 2000.

[6] Samuel W. Bent and John W. John. Finding the median requires $2n$ comparisons. In Robert Sedgewick, editor, *Proceedings of the 17th Annual ACM Symposium on Theory of Computing, STOC 1985, Providence, Rhode Island, USA, May 6-8, 1985*, pages 213–216. ACM, 1985.

[7] Manuel Blum, Robert W. Floyd, Vaughan R. Pratt, Ronald L. Rivest, and Robert Endre Tarjan. Time bounds for selection. *Journal of Computer and System Sciences*, 7(4):448–461, 1973.

[8] Ke Chen and Adrian Dumitrescu. Selection algorithms with small groups. *International Journal of Foundations of Computer Science*, 31(3):355–369, 2020.

[9] Francis Y. L. Chin and Hing fung Ting. An improved algorithm for finding the median distributively. *Algorithmica*, 2:235–249, 1987.

[10] Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest, and Clifford Stein. *Introduction to Algorithms, 3rd Edition.* MIT Press, 2009.

[11] Walter Cunto and J. Ian Munro. Average case selection. *Journal of ACM*, 36(2):270–279, 1989.

[12] Sanjoy Dasgupta, Christos H. Papadimitriou, and Umesh V. Vazirani. *Algorithms.* McGraw-Hill, 2008.

[13] Dorit Dor, Johan Håstad, Staffan Ulfberg, and Uri Zwick. On lower bounds for selecting the median. *SIAM Journal on Discrete Mathematics*, 14(3):299–311, 2001.

[14] Dorit Dor and Uri Zwick. Finding the αn-th largest element. *Combinatorica*, 16(1):41–58, 1996.

[15] Dorit Dor and Uri Zwick. Selecting the median. *SIAM Journal on Computing*, 28(5):1722–1758, 1999.

[16] Adrian Dumitrescu. Finding a mediocre player. In Pinar Heggernes, editor, *Proceedings of the 11th International Conference on Algorithms and Complexity, CIAC 2019, Rome, Italy, May 27-29, 2019*, volume 11485 of *Lecture Notes in Computer Science*, pages 212–223. Springer, 2019.

[17] Adrian Dumitrescu. A selectable sloppy heap. *Algorithms, special issue on efficient data structures*, 12(3):58, 2019.

[18] Stefan Edelkamp and Armin Weiß. Worst-case efficient sorting with quickmergesort. In Stephen G. Kobourov and Henning Meyerhenke, editors, *Proceedings of the 21st Workshop on Algorithm Engineering and Experiments, ALENEX 2019, San Diego, CA, USA, January 7-8, 2019*, pages 1–14. SIAM, 2019.

[19] Robert W. Floyd and Ronald L. Rivest. Expected time bounds for selection. *Communications of ACM*, 18(3):165–172, 1975.

[20] Frank Fussenegger and Harold N. Gabow. A counting approach to lower bounds for selection problems. *Journal of ACM*, 26(2):227–238, 1979.

[21] Abdollah Hadian and Milton Sobel. Selecting the t-th largest using binary errorless comparisons. Technical Report No. 121, School of Statistics, University of Minnesota, 1969.

[22] Charles Antony Richard Hoare. Algorithm 63: Partition and algorithm 65: Find. *Communications of ACM*, 4(7):321–322, 1961.

[23] Laurent Hyafil. Bounds for selection. *SIAM Journal on Computing*, 5(1):109–114, 1976.

[24] John W. John. A new lower bound for the set-partitioning problem. *SIAM Journal on Computing*, 17(4):640–647, 1988.

[25] Haim Kaplan, László Kozma, Or Zamir, and Uri Zwick. Selection from heaps, row-sorted matrices, and $x+y$ using soft heaps. In Jeremy T. Fineman and Michael Mitzenmacher, editors, *Proceedings of the 2nd Symposium on Simplicity in Algorithms, SOSA 2019, San Diego, CA, USA, January 8-9, 2019*, volume 69 of *OASICS*, pages 5:1–5:21. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2019.

[26] David G. Kirkpatrick. A unified lower bound for selection and set partitioning problems. *Journal of ACM*, 28(1):150–165, 1981.

[27] Jon M. Kleinberg and Éva Tardos. *Algorithm design*. Addison-Wesley, 2006.

[28] Donald E. Knuth. *The art of computer programming, Volume III: Sorting and Searching, 2nd Edition*. Addison-Wesley, 1998.

[29] Eyal Kushilevitz and Noam Nisan. *Communication complexity*. Cambridge University Press, 1997.

[30] Eyal Kushilevitz, Noam Nisan, and Bill Gasarch. Errata of communication complexity. `http://www.cs.technion.ac.il/~eyalk/book.html`.

[31] S. L. Mantzaris. On "an improved algorithm for finding the median distributively". *Algorithmica*, 10(6):501–504, 1993.

[32] Conrado Martínez and Salvador Roura. Optimal sampling strategies in quicksort and quickselect. *SIAM Journal on Computing*, 31(3):683–705, 2001.

[33] Catherine C. McGeoch and J. Doug Tygar. Optimal sampling strategies for quicksort. *Random Structures & Algorithms*, 7(4):287–300, 1995.

[34] Mike Paterson. Progress in selection. In Rolf G. Karlsson and Andrzej Lingas, editors, *Proceedings of the 5th Scandinavian Workshop on Algorithm Theory SWAT 1996, Reykjavík, Iceland, July 3-5, 1996*, volume 1097 of *Lecture Notes in Computer Science*, pages 368–379. Springer, 1996.

[35] Anup Rao and Amir Yehudayoff. *Communication complexity and applications.* Cambridge University Press, 2020.

[36] Michael Rodeh. Finding the median distributively. *Journal of Computer and System Sciences*, 24(2):162–166, 1982.

[37] Arnold Schönhage, Mike Paterson, and Nicholas Pippenger. Finding the median. *Journal of Computer and System Sciences*, 13(2):184–199, 1976.

[38] Andrew Chi-Chih Yao. Some complexity questions related to distributive computing (preliminary report). In Michael J. Fischer, Richard A. DeMillo, Nancy A. Lynch, Walter A. Burkhard, and Alfred V. Aho, editors, *Proceedings of the 11th Annual ACM Symposium on Theory of Computing, STOC 1979, Atlanta, Georgia, USA, April 30 - May 2, 1979*, pages 209–213. ACM, 1979.

[39] Frances F. Yao. On lower bounds for selection problems. Technical Report MAC TR-121, Massachusetts Institute of Technology, Cambridge, Massachusetts, USA, 1974.

[40] Chee-Keng Yap. New upper bounds for selection. *Communications of ACM*, 19(9):501–508, 1976.

Part II

Computational Geometry

Chapter 4

Longest Spanning Tree with Neighborhoods

4.1 Introduction

In the *Euclidean Maximum Spanning Tree Problem* (EMST), given a set of points in the Euclidean space $\mathbb{R}^d$, $d \geq 2$, one seeks a tree that connects these points (as vertices) and has maximum length. The problem is easily solvable in polynomial time by Prim's algorithm or by Kruskal's algorithm; algorithms that take advantage of the geometry are also available [14].

In this chapter we study a natural generalization of the above problem. In the *Longest Spanning Tree with Neighborhoods* (Max-St-N), each point is replaced by a point-set, called *neighborhood* (or *region*), and the tree must connect n representative points, one chosen from each neighborhood (duplicate representatives are allowed), and the tree has maximum length. The tree edges are straight line segments connecting pairs of points in distinct neighborhoods. For obvious reasons we refer to these edges as *bichromatic*. As one would expect, the difficulty lies in choosing the representative points; once these points are selected, the problem is reduced to the graph setting and is therefore easily solvable. An example of a spanning tree for 10 neighborhoods is shown in Figure 4.1.

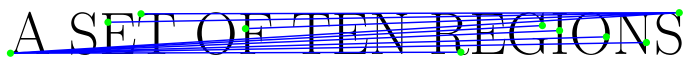


Figure 4.1: An example of a long (still suboptimal) spanning tree for 10 neighborhoods $\mathcal{N} = \{\mathrm{A}, \mathrm{S}\cup\mathrm{S}, \mathrm{E}\cup\mathrm{E}\cup\mathrm{E}, \mathrm{T}\cup\mathrm{T}, \mathrm{O}\cup\mathrm{O}, \mathrm{F}, \mathrm{N}\cup\mathrm{N}, \mathrm{R}, \mathrm{G}, \mathrm{I}\}$ (five neighborhoods are disconnected). The blue segments form a spanning tree on $\mathcal{N}$ and the green dots are the chosen representative points.

The input $\mathcal{N}$ consists of n (possibly disconnected) neighborhoods. For simplicity, it is assumed that each neighborhood is a union of polyhedra and the total vertex complexity of the input is N.

The greedy algorithm. A (natural) greedy algorithm chooses two points attaining a maximum inter-point distance with points in distinct neighborhoods as representatives, and then repeatedly chooses a point in another neighborhood as far as possible from some representative point. The above algorithm does not necessarily find an optimal tree. Let $\mathcal{N} = \{X_1, X_2, X_3\}$, where $X_1 = \{a, c\}$, $X_2 = \{b, c\}$, $X_3 = \{d\}$, Δabc is a unit equilateral triangle and d is the midpoint of ab; see Figure 4.2. Here the selection $a \in X_1$, $b \in X_2$, $d \in X_3$ yields a spanning tree in the form of a star centered at a of length $|ab| + |ad| = 3/2$ (the edge lengths are 1, 1/2, and 1/2 in the underlying complete graph). On the other hand, selecting vertices $c \in X_1$, $c \in X_2$, $d \in X_3$ yields a spanning tree in the form of a 2-edge star centered at d of length $|dc| + |dc| = 2 \times \sqrt{3}/2 = \sqrt{3}$ which is better (the edge lengths are $\sqrt{3}/2$, $\sqrt{3}/2$, and 0 in the underlying complete graph).

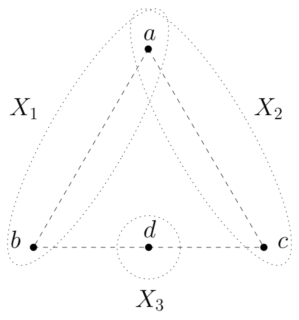


Figure 4.2: Left: an example on which the greedy algorithm is suboptimal.

Definitions and notations. A *geometric graph* G is a graph whose vertex set is a finite set of points in $\mathbb{R}^d$ and whose edges consist of straight line segments [15, p. 223]. For two points $p, q \in \mathbb{R}^d$, the Euclidean distance between them is denoted by $|pq|$. The *length* of G, denoted by $\text{len}(G)$, is the sum of the Euclidean lengths of all edges in G.

For a neighborhood $X \in \mathcal{N}$, let $V(X)$ denote its set of vertices. Let $V = \cup_{X \in \mathcal{N}} V(X)$ denote the union of vertices of all neighborhoods in $\mathcal{N}$ and $N = |V|$.

Given a set $\mathcal{N}$ of n neighborhoods, we define the following parameters. A *monochromatic diameter* pair is a pair of points in the same neighborhood attaining a maximum distance. A *bichromatic diameter* pair is a pair of points from two neighborhoods attaining a maximum distance, i.e., $p_i \in X_i$, $p_j \in X_j$, where $X_i, X_j \in \mathcal{N}$, $i \neq j$, and $|p_i p_j|$ is maximum. A *diameter* pair is a pair of points (in the same neighborhood or in different neighborhoods) attaining an overall maximum distance. See Figure 4.3 for an illustration.

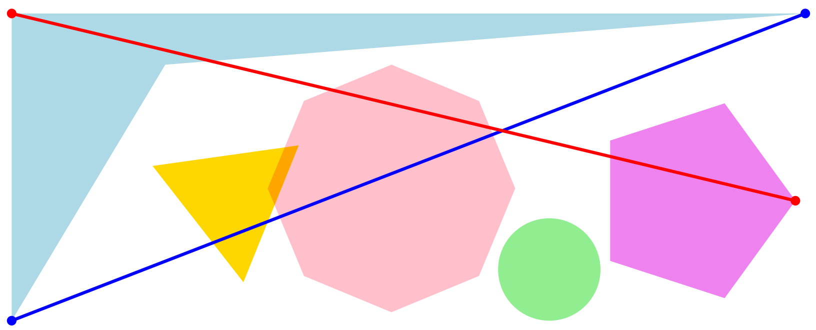


Figure 4.3: A monochromatic diameter pair (in blue) and a bichromatic diameter pair (in red) for a set of 5 neighborhoods. The blue one is also a diameter pair.

For $X \in \mathcal{N}$ and $p \in X$, let $d_{\max}(p)$ denote the maximum distance between p and any point of a neighborhood $Y \in \mathcal{N} \setminus \{X\}$. It is well known and easy to prove that both a monochromatic diameter and a bichromatic diameter pair are attained by pairs of vertices in the input instance. An optimal (longest) spanning tree with neighborhoods is denoted by T_{OPT}; it is a geometric graph whose vertices are the representative points of the n neighborhoods.

Our results. We start by providing a factor 1/2 approximation to MAX-ST-N. We then offer two refinement steps achieving a better ratio. The last refinement step proves the following.

Theorem 4.1. *Given a set $\mathcal{N}$ of n neighborhoods in $\mathbb{R}^d$ (with total vertex complexity N), a ratio 0.511 approximation for the maximum spanning tree for the neighborhoods in $\mathcal{N}$ can be computed in polynomial time.*

It is natural to try to include long edges (with endpoints in different neighborhoods) when constructing a long spanning tree. However, in this regard we show that every algorithm that always includes a bichromatic diameter pair in the solution is bound to have an approximation ratio at most $\sqrt{2-\sqrt{3}} = 0.517\ldots$ (via Figure 4.12 in Section 4.3).

Background and related work. Computing the minimum or maximum Euclidean spanning trees of a point set are classical problems in a geometric setting [14, 16]. The Traveling Salesman Problem (TSP) is yet another related problem with a rich history of research in combinatorial optimization. Several variants of the TSP including the *Euclidean Traveling Salesman Problem* (ETSP) and *Maximum Traveling Salesman Problem* (MAX TSP) are surveyed in [10, 12, 13].

While past research has primarily focused on minimization problems, the maximization variants usually require different techniques and so they are interesting in their own right and pose many unmet challenges. See for instance the section devoted to longest subgraph problems in the survey of Bern and Eppstein [5]. The results obtained in this area in the last 20 years are rather sparse; the few articles [4, 9, 11] make a representative sample. Recently, Biniaz et al. [7] gave several approximation algorithms for computing a longest noncrossing spanning tree in a multipartite geometric graph.

Spanning trees for systems of neighborhoods have also been studied. For instance, given a set of n (possibly disconnected) compact neighborhoods in $\mathbb{R}^d$, select a point in each neighborhood so that the minimum spanning tree on these points has minimum length [8, 20], or maximum length [8], respectively. In the cycle version first studied by Arkin and Hassin [3], called *TSP with neighborhoods* (TSPN), given a set of neighborhoods in $\mathbb{R}^d$, one needs to find a shortest closed curve (tour) intersecting each neighborhood.

Organization. The rest of the chapter is organized as follows. Section 4.2 presents two approximation algorithms: one with ratio 0.5 (in Subsection 4.2.1) and one with ratio 0.511 (in Subsection 4.2.2). The analysis of the latter algorithm is carried out in Section 4.3. We conclude in Section 4.4 with a summary of the results in the context of related problems and some future research directions.

4.2 Approximation Algorithms

For simplicity, we present our algorithms for the plane i.e., $d = 2$. The extension to higher dimensions is straightforward, and is briefly discussed at the end.

Let $S = \{p_1, \dots, p_n\}$, where $p_i = (x_i, y_i)$. Given a point $p \in S$, the *star centered at* p, denoted by S_p, is the spanning tree on S whose edges connect p to the other points. Figure 4.4 shows a star S_p for 10 points. Using a technique developed in [9] (in fact a simplification

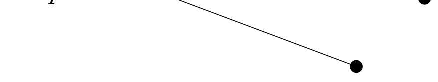


Figure 4.4: A star centered at p for 10 points.

of an earlier approach from [2]), we first obtain an approximation algorithm with ratio 1/2 (`Algorithm A1`). `Algorithm A2` described later in this section implements a refinement of this technique.

4.2.1 A Simple 0.5-Approximation Algorithm

`Algorithm A1.` Compute a bichromatic diameter of the point set V, pick an arbitrary point (vertex) from each of the other $n-2$ neighborhoods, and output the longest of the two stars centered at one of the endpoints of the diameter. Figure 4.5 illustrates the two candidate stars computed on a set of 5 neighborhoods.

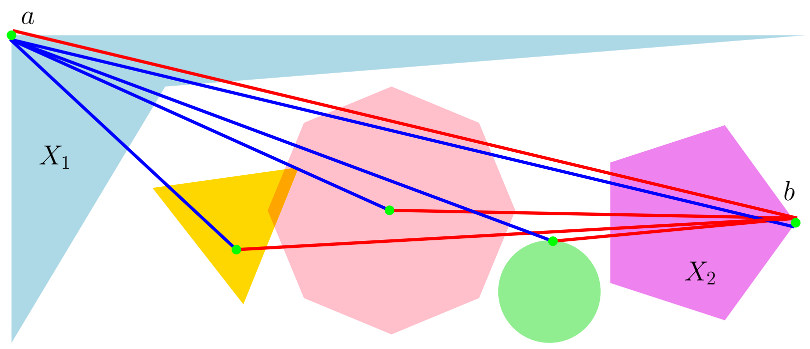


Figure 4.5: Two candidate stars (one in red, one in blue) centered at a bichromatic diameter pair a and b of 5 neighborhoods. The other three points are chosen arbitrarily, one from each neighborhood.

Analysis. Let ab be a bichromatic diameter pair, and assume without loss of generality that ab is a horizontal unit segment, where $a = (0, 0)$ and $b = (1, 0)$. We may assume that $a \in X_1$ and $b \in X_2$; refer to Figure 4.6.

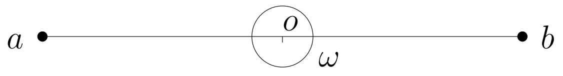


Figure 4.6: A bichromatic diameter pair a, b and the disk ω.

The ratio $1/2$ (or $\frac{n}{2n-2}$ which is slightly better) follows from the next lemma in conjunction with the obvious upper bound

$$\text{len}(T_{\text{OPT}}) \leq n - 1. \tag{4.1}$$

The latter is implied by the fact that each edge of T_{OPT} is bichromatic and thus of length at most 1.

Lemma 4.2. *Let S_a and S_b be the stars centered at the points a and b, respectively. Then* $\mathrm{len}(S_a) + \mathrm{len}(S_b) \geq n$.

Proof. Assume that $a = p_1$, $b = p_2$. For each $i = 3, \ldots, n$, the triangle inequality for the triple a, b, p_i gives

$$|ap_i| + |bp_i| \geq |ab| = 1.$$

By summing up we have

$$\mathrm{len}(S_a) + \mathrm{len}(S_b) = \sum_{i=3}^{n} (|ap_i| + |bp_i|) + 2|ab| \geq (n-2) + 2 = n. \qquad \square$$

4.2.2 An Improved 0.511-Approximation Algorithm

We next refine the previous algorithm to achieve an approximation ratio of 0.511. The setting is the same, where ab is a bichromatic diameter pair of unit length. The technique uses two parameters x and y, introduced below. The smallest value of the ratio obtained over the entire range of admissible x and y is determined and yields the approximation ratio of `Algorithm A2`.

Let o be the midpoint of ab, and ω be the disk centered at o, of minimum radius, say, x, containing at least $\lfloor n/2 \rfloor$ of the neighborhoods $X_3, \ldots, X_n$. In particular, this implies that we can consider $\lfloor n/2 \rfloor$ neighborhoods as contained in ω and $\lceil n/2 \rceil$ neighborhoods having points on the boundary $\partial\omega$ or in the exterior of ω. We first argue that for $x \geq 0.2$, the 0.511 approximation ratio easily follows (with room to spare). Observe that for each of the neighborhoods not contained in ω, there is a point p in it such that one of the connections from p to a or b is at least $\sqrt{\frac{1}{4} + x^2}$, see Figure 4.7. Let T be the spanning tree consisting of all such longer connections together with ab. Then,

$$\mathrm{len}(T) \geq 1 + \left\lfloor \frac{n}{2} \right\rfloor \frac{1}{2} + \left(\left\lceil \frac{n}{2} \right\rceil - 2 \right) \sqrt{\frac{1}{4} + x^2}. \tag{4.2}$$

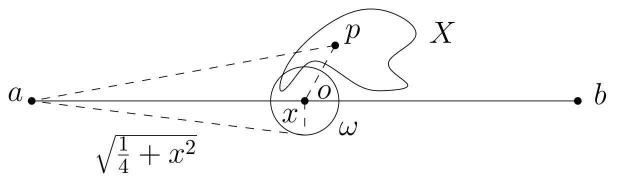


Figure 4.7: For each neighborhood X not contained in the disk ω, pick an arbitrary point $p \in X - \omega$, then one of the segments pa and pb is at least $\sqrt{\frac{1}{4}+x^2}$.

The above expression can be simplified as follows:

$$\operatorname{len}(T) \geq \begin{cases} 1+\dfrac{n}{4}+\left(\dfrac{n}{4}-1\right)\sqrt{1+4x^2}, & \text{if } n \text{ is even,} \\ 1+\dfrac{n-1}{4}+\left(\dfrac{n+1}{4}-1\right)\sqrt{1+4x^2}, & \text{if } n \text{ is odd.} \end{cases} \tag{4.3}$$

Assume first that $x \geq 1/2$. Then,

$$\begin{aligned} \operatorname{len}(T) &\geq 1+\frac{n}{4}+\left(\frac{n}{4}-1\right)\sqrt{2} = \left(\frac{1}{4}+\frac{\sqrt{2}}{4}\right)(n-1)+\left(\frac{5-3\sqrt{2}}{4}\right) \\ &\geq \left(\frac{1+\sqrt{2}}{4}\right)(n-1), \text{ if } n \text{ is even, and} && (4.4) \\ \operatorname{len}(T) &\geq 1+\frac{n-1}{4}+\left(\frac{n+1}{4}-1\right)\sqrt{2} = \left(\frac{1}{4}+\frac{\sqrt{2}}{4}\right)(n-1)+\left(1-\frac{\sqrt{2}}{2}\right) \\ &\geq \left(\frac{1+\sqrt{2}}{4}\right)(n-1), \text{ if } n \text{ is odd.} && (4.5) \end{aligned}$$

Together with (4.1), we have $\operatorname{len}(T) \geq 0.6 \operatorname{len}(T_{\mathrm{OPT}})$, for every $n \geq 2$ and $x \geq 1/2$.

Assume next that $x \leq 1/2$. If n is even, (4.3) yields

$$\begin{aligned} \operatorname{len}(T) &\geq 1+\frac{n}{4}+\left(\frac{n}{4}-1\right)\sqrt{1+4x^2} \\ &= \frac{n-1}{4}\left(1+\sqrt{1+4x^2}\right)+\left(\frac{5}{4}-\frac{3}{4}\sqrt{1+4x^2}\right) \\ &\geq \frac{n-1}{4}\left(1+\sqrt{1+4x^2}\right). \end{aligned}$$

Indeed, the last term in the second line is non-negative for $x \leq 1/2$.

If n is odd, (4.3) yields

$$\begin{aligned}\operatorname{len}(T) &\geq 1 + \frac{n-1}{4} + \left(\frac{n+1}{4} - 1\right)\sqrt{1+4x^2} \\ &= \frac{n-1}{4}\left(1+\sqrt{1+4x^2}\right) + \left(1 - \frac{1}{2}\sqrt{1+4x^2}\right) \\ &\geq \frac{n-1}{4}\left(1+\sqrt{1+4x^2}\right).\end{aligned}$$

Again, the last term in the second line is non-negative for $x \leq 1/2$.

Consequently, for every $n \geq 2$ and $x \leq 1/2$ we have

$$\operatorname{len}(T) \geq \frac{n-1}{4}\left(1+\sqrt{1+4x^2}\right). \tag{4.6}$$

It is easy to check that

$$\frac{1+\sqrt{1+4x^2}}{4} \geq \frac{5+\sqrt{29}}{20} = 0.519\ldots, \text{ for } x \geq 0.2. \tag{4.7}$$

Hence the approximation ratio is at least $0.519\ldots$ if $x \geq 0.2$. We therefore subsequently assume that $x \leq 0.2$. Let the monochromatic diameter of V be $1+y$, for some $y \in [-1, \infty)$. The next lemma shows that $y \leq 1$, and so the monochromatic diameter of V is $1+y$, for some $y \in [-1, 1]$.

Lemma 4.3. *For every $X \in \mathcal{N}$,* $\operatorname{diam}(X) \leq 2$.

Proof. Let pq be a diameter pair of neighborhood X. Let r be an arbitrary point of an arbitrary neighborhood $Y \in \mathcal{N} \setminus \{X\}$. By the triangle inequality, we have $|pq| \leq |pr| + |rq| \leq 1 + 1 = 2$, as required. □

If $y \geq 0.2$, let $a_1, b_1 \in X$ be a corresponding diameter pair. Choose a point in every other neighborhood and connect it to a_1 and b_1. Since $|a_1 b_1| = 1 + y \geq 1.2$, the longer of the two stars centered at a_1 and b_1 has length at least $(n-1)(1+y)/2 \geq 0.6(n-1)$; this candidate spanning tree offers thereby this ratio of approximation. We will subsequently assume that $y \in [-1, 0.2]$.

We have thus shown that a constant approximation ratio better than 0.511 can be obtained if x or y is sufficiently large. In the complementary case, i.e., both x and y are small, we apply the following algorithm.

`Algorithm A2.` The algorithm computes two candidate solutions T_1 and T_2 and returns the best of the two. The setting is the same as in Subsection 4.2.1. In particular, it is assumed that $x \in [0, 0.2]$ and $y \in [-1, 0.2]$ (outside this range, the approximation ratio exceeds 0.519).

The first candidate solution T_1 for the spanning tree is only relevant for the range $y \geq 0$ (if $y < 0$ its length could be smaller than $(n-1)/2$ and T_1 will be ignored). Suppose that a monochromatic diameter pair in V is achieved by a pair $a_1, b_1 \in X$. Recall that $|a_1 b_1| = 1+y$. Choose an arbitrary point in every other neighborhood and connect it to a_1 and b_1. Let T_1 be the longer of the two stars centered at a_1 and b_1. See Figure 4.8 for an illustration.

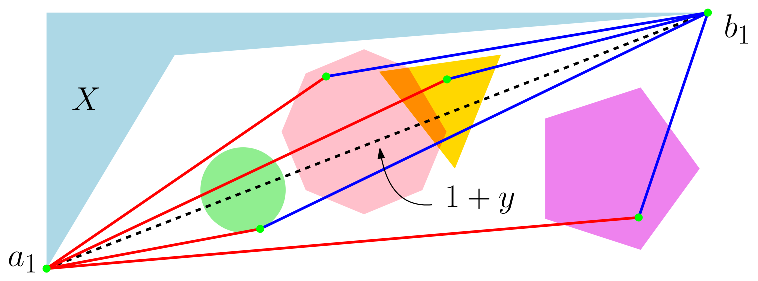


Figure 4.8: The first candidate T_1 computed by `Algorithm A2` on 5 neighborhoods.

The second candidate solution T_2 for the spanning tree connects each of the neighborhoods contained in ω with either a or b at a cost of at least $1/2$ (based on the fact that $\max\{|ap_i|, |bp_i|\} \geq |ab|/2 = 1/2$). For each neighborhood X_i, $i \geq 3$, select the vertex of X_i that is farthest from o and connect it with a or b, whichever yields the longer connection. As such, if X_i is not contained in ω, the connection length is at least $\sqrt{\frac{1}{4} + x^2}$. Finally add the unit segment ab. See Figure 4.9 for an illustration.

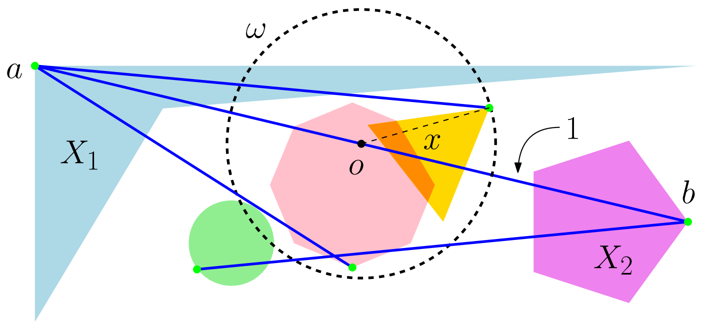


Figure 4.9: The second candidate T_2 computed by `Algorithm A2` on 5 neighborhoods.

Lower bounds on the lengths of candidate solutions. By the triangle inequality, the length of the star T_1 is bounded from below as follows:

$$\mathrm{len}(T_1) \geq (n-1)\,\frac{1+y}{2}. \tag{4.8}$$

The length of T_2 is bounded from below by (4.6). As such, we have

$$\mathrm{len}(T_2) \geq \frac{n-1}{4}\left(1+\sqrt{1+4x^2}\right). \tag{4.9}$$

In order to prove the claimed approximation ratio 0.511 for `Algorithm A2`, we first derive a sharper upper bound on the length of T_{OPT} when both x and y are smaller than 0.2.

4.2.3 Upper bound on $\mathrm{len}(T_{\mathbf{OPT}})$

Let Ω be the disk of radius $R(y)$ centered at o, see Figure 4.10, where

$$R(y) = \begin{cases} \dfrac{\sqrt{3}}{2} & \text{if } y \leq 0, \\[2ex] \dfrac{\sqrt{3}}{2} + \dfrac{2y}{\sqrt{3}} & \text{if } y \geq 0. \end{cases}$$

Lemma 4.4. *V is contained in Ω.*

Proof. Assume for contradiction that there exists a point $p_i \in X_i$ at distance larger than

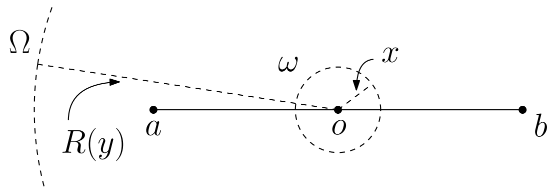


Figure 4.10: All the neighborhoods are contained in Ω with radius $R(y)$.

$R(y)$ from o. By symmetry, we may assume that $|ap_i| \le |bp_i|$ and that p_i lies in the closed halfplane above the line containing ab.

First consider the case $y \le 0$; it follows that $|bp_i| > \sqrt{\frac{1}{4}+\frac{3}{4}} = 1$. If $i = 2$, then $b, p_i \in X_2$, which contradicts the definition of y. Otherwise, $b \in X_2$ and $p_i \in X_i$ are points in different neighborhoods at distance larger than 1, in contradiction with the original assumption on the bichromatic diameter of V.

Next consider the case $y \ge 0$; it follows that $|bp_i| \ge \sqrt{\frac{1}{4} + \left(\frac{\sqrt{3}}{2} + \frac{2}{\sqrt{3}}\,y\right)^2} > 1+y$. If $i = 2$, then $b, p_i \in X_2$, which contradicts the definition of y. Otherwise, $b \in X_2$ and $p_i \in X_i$ are points in different neighborhoods at distance larger than 1, in contradiction with the original assumption on the bichromatic diameter of V.

In either case (for any y) we have reached a contradiction, and this concludes the proof. □

Recall that for a point $p \in X \in \mathcal{N}$, $d_{\max}(p)$ is the maximum distance between p and a point of a neighborhood $Y \in \mathcal{N} \setminus \{X\}$.

Lemma 4.5. *Let $\mathcal{N} = \{X_1, \ldots, X_n\}$ be a set of n neighborhoods and T_{OPT} be an optimal spanning tree assumed to connect points (vertices) $p_i \in X_i$ for $i = 1, \ldots, n$. For every $j \in [n]$, we have*

$$\mathrm{len}(T_{OPT}) \le \sum_{i \ne j} d_{max}(p_i).$$

Proof. Consider T_{OPT} rooted at p_j. Let $\pi(v)$ denote the parent of a (non-root) vertex v. Uniquely assign each edge $\pi(v)v$ of T_{OPT} to vertex v. The inequality $\mathrm{len}(\pi(v)v) \le d_{\max}(v)$ holds for each edge of the tree. By adding up the above inequalities, the lemma follows. □

Lemma 4.6. *If $X \in \mathcal{N}$ is contained in ω, and $p \in X$, then $d_{max}(p) \leq \min(1, x + R(y))$.*

Proof. By definition, $d_{\max}(p) \leq 1$. By Lemma 4.4, the vertex set V is contained in Ω and thus all neighborhoods in $\mathcal{N}$ are contained in Ω. By the triangle inequality, $d_{\max}(p) \leq |po| + R(y) \leq x + R(y)$, as claimed. $\square$

Lemma 4.7. *The following inequality holds:*

$$\operatorname{len}(T_{OPT}) \leq (n-1) \cdot \min\left(1, \frac{1 + x + R(y)}{2}\right). \tag{4.10}$$

Proof. Let T_{OPT} be a longest spanning tree of $p_1, \ldots, p_n$, where $p_i \in X_i$, for $i = 1, \ldots, n$. View T_{OPT} as rooted at $p_1 \in X_1$; recall that $a \in X_1$. By Lemma 4.5,

$$\operatorname{len}(T_{\mathrm{OPT}}) \leq \sum_{i=2}^{n} d_{\max}(p_i).$$

If X_i is not contained in ω, $d_{\max}(p_i) \leq 1$; otherwise, by Lemma 4.6, $d_{\max}(p_i) \leq \min(1, x + R(y))$. By the setting of x in the definition of ω, we have

$$\operatorname{len}(T_{\mathrm{OPT}}) \leq \left(\left\lceil \frac{n}{2} \right\rceil - 1\right) \cdot 1 + \left\lfloor \frac{n}{2} \right\rfloor \cdot \min(1, x + R(y)). \tag{4.11}$$

If n is even, inequality (4.11) yields (since the second term in the second line is ≤ 0)

$$\begin{aligned}
\operatorname{len}(T_{\mathrm{OPT}}) &\leq \left(\frac{n}{2} - 1\right) + \frac{n}{2} \cdot \min(1, x + R(y)) \\
&\leq \frac{n-1}{2}\,(1 + x + R(y)) + \frac{\min(1, x + R(y)) - 1}{2} \\
&\leq \frac{n-1}{2}\,(1 + x + R(y))\,.
\end{aligned}$$

If n is odd, Inequality (4.11) yields

$$\operatorname{len}(T_{\mathrm{OPT}}) \leq \frac{n-1}{2} + \frac{n-1}{2}\,(x + R(y)) = \frac{n-1}{2}\,(1 + x + R(y))\,.$$

Therefore the above inequality holds for every $n \geq 2$. The lemma follows by adjoining the trivial upper bound in equation (4.1). $\square$

4.3 Analysis of `Algorithm A2`

We start with a preliminary argument for ratio 0.506 that comes with a simpler proof. We then give a sharper analysis for ratio 0.511.

A preliminary estimate on the approximation ratio of `Algorithm A2`. First consider the case $y < 0$. Then $R(y) = \sqrt{3}/2$, so the ratio of `Algorithm A2` is at least

$$\min_{\substack{0 \leq x \leq 0.2 \\ y < 0}} \frac{\operatorname{len}(T_2)}{\operatorname{len}(T_{\text{OPT}})} \geq \min_{0 \leq x \leq 0.2} \frac{1 + \sqrt{1 + 4x^2}}{\min\left(4, 2 + \sqrt{3} + 2x\right)}.$$

A standard analysis shows that this ratio achieves its minimum $\left(1 + 2\sqrt{2 - \sqrt{3}}\right)\Big/ 4 = 0.508\ldots$ when $x = 1 - \sqrt{3}/2$.

When $y \geq 0$, the ratio of `Algorithm A2` is at least

$$\min_{0 \leq x, y \leq 0.2} \max\left(\frac{\operatorname{len}(T_1)}{\operatorname{len}(T_{\text{OPT}})}, \frac{\operatorname{len}(T_2)}{\operatorname{len}(T_{\text{OPT}})}\right).$$

The inequalities (4.8), (4.9), (4.10) imply that this ratio is at least

$$\frac{\max\left(1 + y, (1 + \sqrt{1 + 4x^2})/2\right)}{\min\left(2, 1 + x + R(y)\right)} = \frac{\max\left(1 + y, (1 + \sqrt{1 + 4x^2})/2\right)}{\min\left(2, 1 + \frac{\sqrt{3}}{2} + x + \frac{2}{\sqrt{3}}\, y\right)}.$$

Since the analysis is similar to that for deriving the refined bound we give next, we state without providing details that this piecewise function reaches its minimum value

$$\left(4\sqrt{3} - 1 - 2\sqrt{9 - 3\sqrt{3}}\right) \Big/ 4 = 0.506\ldots,$$

when

$$x = \sqrt{3}/2 - 3 + 2\sqrt{3 - \sqrt{3}} = 0.1180\ldots, \text{ and } y = \left(4\sqrt{3} - 3 - 2\sqrt{9 - 3\sqrt{3}}\right) \Big/ 2 = 0.0137\ldots.$$

This provides a preliminary ratio 0.506 in Theorem 4.1.

A refined bound. Let $m = \lfloor n/2 \rfloor$. Assume for convenience that the neighborhoods $X_3, \ldots, X_n$ are relabeled so that $X_3, \ldots, X_{m+2}$ are contained in ω and $X_{m+3}, \ldots, X_n$ are not contained in the interior of ω. Recall that $p_i \in X_i$ are the representative points in an optimal solution T_{OPT}. Let $x_i = |op_i|$, for $i = 3, \ldots, m+2$; as such, $x_3, \ldots, x_{m+2} \leq x$. Denote the average of $x_3, \ldots, x_{m+2}$ by z, i.e., $\sum_{i=3}^{m+2} x_i = mz$, and note that $z \leq x$.

As in the proof of Lemma 4.6, by the triangle inequality we have

$$d_{\max}(p_i) \leq |op_i| + R(y) = x_i + R(y), \text{ for } i = 3, \ldots, m+2.$$

Consequently, the upper bound in (4.10) can be improved to

$$\mathrm{len}(T_{\mathrm{OPT}}) \leq (n-1) \cdot \min\left(1, \frac{1 + z + R(y)}{2}\right). \tag{4.12}$$

We next obtain an improved lower bound on $\mathrm{len}(T_2)$. Recall that `Algorithm A2` selects the vertex of X_i that is farthest from o for every $i \geq 3$, and connects it with a or b, whichever yields the longer connection. In particular, the length of this connection is at least $\sqrt{\frac{1}{4} + x_i^2}$ for $i = 3, \ldots, m+2$. Let $h\colon [0, \infty) \to \mathbb{R}$ be defined as follows:

$$h(x) = \sqrt{1 + 4x^2}. \tag{4.13}$$

It is easy to check that

$$h'(x) = \frac{4x}{\sqrt{1+4x^2}} \geq 0 \text{ and } h''(x) = \frac{4}{(1+4x^2)^{3/2}} > 0. \tag{4.14}$$

As such, the function $h(x)$ is convex, i.e.,

$$h\left(\frac{x+y}{2}\right) \leq \frac{h(x)+h(y)}{2}, \text{ for every } x, y \geq 0, \tag{4.15}$$

and Jensen's inequality yields:

$$\sum_{i=3}^{m+2} \sqrt{1+4x_i^2} \geq m\sqrt{1+4z^2}. \tag{4.16}$$

We thereby obtain (noting that $z \leq x$) the following sharpening of the lower bound in (4.9):

$$\text{len}(T_2) \geq \frac{n-1}{4}\left(\sqrt{1+4z^2} + \sqrt{1+4x^2}\right) \geq \frac{n-1}{2}\sqrt{1+4z^2}. \tag{4.17}$$

To analyze the approximation ratio we relate the upper bound (4.12) to the lower bound (4.17) and distinguish two cases:

Case 1: $y \leq 0$. Then $R(y) = \sqrt{3}/2$, so the ratio of `Algorithm A2` is at least

$$\min_{0 \leq z \leq 0.2} \frac{\text{len}(T_2)}{\text{len}(T_{\text{OPT}})} \geq \min_{0 \leq z \leq 0.2} \frac{2\sqrt{1+4z^2}}{\min\left(4, 2+2z+\sqrt{3}\right)}.$$

When $4 \leq 2+2z+\sqrt{3}$, we have $z \geq 1-\sqrt{3}/2$. Then

$$\frac{\sqrt{1+4z^2}}{2} \geq \frac{\sqrt{8-4\sqrt{3}}}{2} = \sqrt{2-\sqrt{3}} = 0.517\ldots.$$

When $2+2z+\sqrt{3} \leq 4$, i.e., $z \leq 1-\sqrt{3}/2$, let

$$f(z) = \frac{2\sqrt{1+4z^2}}{2+\sqrt{3}+2z}.$$

Then

$$f'(z) = \frac{8\left(2+\sqrt{3}\right)z - 4}{\sqrt{1+4z^2}\left(2+\sqrt{3}+2z\right)^2}.$$

Since $8\left(2+\sqrt{3}\right)z - 4 \le 4\left(2+\sqrt{3}\right)\left(2-\sqrt{3}\right) - 4 = 0$, the function is non-increasing on $[0, 1-\sqrt{3}/2]$, and so

$$f(z) \ge f\left(1-\sqrt{3}/2\right) = \sqrt{2-\sqrt{3}} = 0.517\ldots.$$

This concludes the proof for the first case.

Case 2: $y \ge 0$, then the ratio of `Algorithm A2` is at least

$$\min_{0 \le y, z \le 0.2} \max\left(\frac{\operatorname{len}(T_1)}{\operatorname{len}(T_{\mathrm{OPT}})}, \frac{\operatorname{len}(T_2)}{\operatorname{len}(T_{\mathrm{OPT}})}\right).$$

For $0 \le y, z \le 0.2$, let

$$g(z,y) = \frac{\max\left(1+y, \sqrt{1+4z^2}\right)}{\min\left(2, 1+z+R(y)\right)} = \frac{\max\left(1+y, \sqrt{1+4z^2}\right)}{\min\left(2, 1+\frac{\sqrt{3}}{2}+z+\frac{2}{\sqrt{3}}\,y\right)}.$$

The inequalities (4.8), (4.12), (4.17) imply that the ratio of `Algorithm A2` is at least

$$\min_{0 \le y, z \le 0.2} g(z,y).$$

The curve $\gamma : 1+y = \sqrt{1+4z^2}$ and the line $\ell : 2 = 1+\frac{\sqrt{3}}{2}+z+\frac{2}{\sqrt{3}}\,y$ split the feasible region $[0,0.2]\times[0,0.2]$ into four subregions; see Figure 4.11. The curve γ intersects line ℓ at point $p = (z_0, y_0)$, where

$$z_0 = \left(8\sqrt[4]{3}-\sqrt{3}-6\right)\Big/26 = 0.1075\ldots, \text{ and } y_0 = \left(8\sqrt{3}-2\sqrt[4]{27}-9\right)\Big/13 = 0.0228\ldots.$$

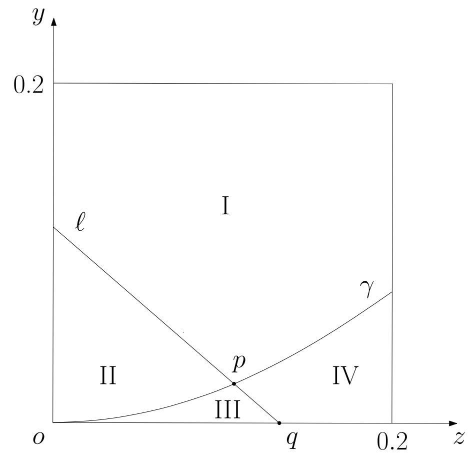


Figure 4.11: The feasible region of the function $g(z, y)$.

Set

$$\rho := (1 + y_0)/2 = \left(4\sqrt{3} + 2 - \sqrt[4]{27}\right) \Big/ 13 = 0.511\ldots. \tag{4.18}$$

In region I, $g(z, y) = (1 + y)/2$. It reaches the minimum value ρ when y is minimized, i.e., $y = y_0$.

In region II, $g(z, y) = \dfrac{1 + y}{1 + \sqrt{3}/2 + z + 2y/\sqrt{3}}$. Its partial derivative is positive, i.e.,

$$\frac{\partial g}{\partial y} = \frac{1 - \sqrt{3}/6 + z}{\left(1 + \sqrt{3}/2 + z + 2y/\sqrt{3}\right)^2} > 0,$$

so $g(z, y)$ reaches its minimum value on the curve γ. On this curve, let

$$G(z) = g\left(z, y(z)\right) = \frac{\sqrt{1 + 4z^2}}{1 - \sqrt{3}/6 + z + 2\sqrt{1 + 4z^2}/\sqrt{3}}.$$

Its derivative is

$$G'(z) = \frac{\left(4 - 2\sqrt{3}/3\right) z - 1}{\sqrt{1 + 4z^2}\left(1 - \sqrt{3}/6 + z + 2\sqrt{1 + 4z^2}/\sqrt{3}\right)^2}.$$

Note that the numerator of $G'(z)$ is negative, i.e., $\left(4 - 2\sqrt{3}/3\right) z - 1 < 4z - 1 < 0$ for $z \in [0, 0.2]$, thus $G'(z) < 0$. So the minimum value is ρ, and is achieved when z is maximized, i.e., $z = z_0$.

In region IV, $g(z, y) = \sqrt{1 + 4z^2}/2$ which increases monotonically with respect to z. So the minimum value is again ρ and is achieved when z is minimized, i.e., $z = z_0$.

In region III,

$$g(z, y) = \frac{\sqrt{1 + 4z^2}}{1 + \sqrt{3}/2 + z + 2y/\sqrt{3}}.$$

Its partial derivative is negative, i.e.,

$$\frac{\partial g}{\partial y} = \frac{-2\sqrt{1 + 4z^2}}{\sqrt{3}\left(1 + \sqrt{3}/2 + z + 2y/\sqrt{3}\right)^2} < 0,$$

so $g(z, y)$ reaches its minimum value on the arc $op \subset \gamma$ or the segment $pq \subset \ell$, where $q = (1 - \sqrt{3}/2, 0)$ is the intersection point of ℓ and the z-axis. Since these two curves are shared with region II and IV respectively, by previous analyses, $g(z, y)$ reaches its minimum value ρ at point p.

In summary, we showed that

$$\min_{0 \le y, z \le 0.2} g(z, y) \ge \rho = 0.511\ldots,$$

establishing the approximation ratio in Theorem 4.1. □

Remarks. 1. The algorithm can be adapted to work in $\mathbb{R}^d$ for any $d \ge 3$. In the analysis, the disk ω becomes the ball of radius x with the same defining property and the disk Ω becomes the ball of radius $R(y)$. All arguments and relevant bounds still hold since they only rely on the triangle inequality; the verification is left to the reader. Consequently, the approximation guarantee remains the same.

2. It is apparent from the context that our methods extend to a broader class of neighborhoods, namely those that are approximable within a prescribed accuracy by unions of polyhedra (this class includes curved objects, for instance balls of arbitrary radii).

An almost tight example. Let Δabc be an isosceles triangle with $|ca| = |cb| = 1 - \varepsilon$, $|ab| = 1$, for a small $\varepsilon > 0$, e.g., set $\varepsilon = 1/(n-1)$. Let $\mathcal{N} = \{X_1, \ldots, X_n\}$, where $X_1 = ac$, $X_2 = bc$, and $X_3, \ldots, X_n$ are $n-2$ points at distance $1-\varepsilon$ from c, below ab and whose projections onto ab are close to the midpoint of ab; see Figure 4.12. Note that ab is the unique (bichromatic) diameter of V (the set of vertices in $\mathcal{N}$). `Algorithm A2` selects $a \in X_1$, $b \in X_2$, and $X_3, \ldots, X_n$.

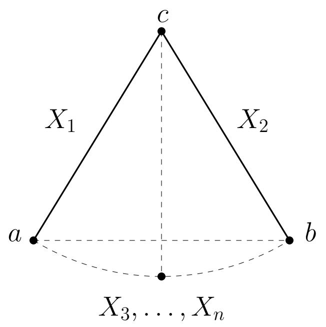


Figure 4.12: A tight example.

The edge ab has unit length and each of the remaining edges has length close to $\sqrt{2-\sqrt{3}}$. Therefore, the spanning tree constructed by `Algorithm A2` is of length close to $1+\sqrt{2-\sqrt{3}}(n-2)$, while the longest spanning tree has length at least $(1-\varepsilon)(n-1) = n-2$. As such, the approximation ratio of `Algorithm A2` approaches $\sqrt{2-\sqrt{3}} = 0.517\ldots$ for large n. Note that this is a tight example for the case $y \leq 0$, for which the ratio of `Algorithm A2` is at least $\sqrt{2-\sqrt{3}}$; and an almost tight example in general, since the overall approximation ratio of `Algorithm A2` is 0.511. Moreover, the example shows that every algorithm that always includes a bichromatic diameter pair in the solution (as the vertices of the corresponding neighborhoods) is bound to have an approximation ratio at most $\sqrt{2-\sqrt{3}}$.

Time complexity of `Algorithm A2`. It is straightforward to implement the algorithm to run in quadratic time for any fixed d. All interpoint distances can be easily computed in $O(N^2)$ time. Similarly the farthest point from o in each neighborhood (over all neighborhoods) can all be computed in $O(N)$ time. Subquadratic algorithms for computing the diameter and farthest bichromatic pairs in higher dimensions can be found in [1, 6, 17–19];

see also the two survey articles [10, 12].

4.4 Conclusion

We gave two simple approximation algorithms for MAX-ST-N: one with ratio 0.5 and one with ratio 0.511 (the latter with a slightly more elaborate analysis but equally simple principles). The first algorithm outputs a star centered at one of the endpoints of a bichromatic diameter. The second algorithm outputs either a star centered at one of the endpoints of a monochromatic diameter or a 2-star with the endpoints of a bichromatic diameter as its centers. A 2-*star* with *centers* a, b consists of an edge connecting a, b and $n-2$ edges connecting every other vertex with one of the two centers.

The following variants represent extensions of the Euclidean maximum TSP for the neighborhood setting. In the *Euclidean Maximum Traveling Salesman Problem*, given a set of points in the Euclidean space $\mathbb{R}^d$, $d \geq 2$, one seeks a cycle (a.k.a. *tour*) that visits these points (as vertices) and has maximum length; see [4]. In the *Maximum Traveling Salesman Problem with Neighborhoods* (MAX-TSP-N), each point is replaced by a point-set, called *neighborhood* (or *region*), and the cycle must connect n representative points, one chosen from each neighborhood (duplicate representatives are allowed), and the cycle has maximum length. Since the original variant with points is NP-hard when $d \geq 3$ (as shown in [4]), the variant with neighborhoods is also NP-hard for $d \geq 3$. The complexity of the original problem in the plane is unsettled, although the problem is believed to be NP-hard [11]. In the *path* variant, one seeks a path of maximum length.

The following problems remain open for future investigation:

1. What is the computational complexity of MAX-ST-N?
2. Can a better approximation be obtained by constructing candidate spanning trees in the form of a 3-star? (A 3-*star* with *centers* a, b, c consists of two edges connecting a, b, c and $n-3$ edges connecting every other vertex with one of the 3 centers.)

3. What approximations can be obtained for the *cycle* or *path* variants of Max-Tsp-N?

Bibliography

[1] P. K. Agarwal, J. Matoušek and S. Suri, Farthest neighbors, maximum spanning trees and related problems in higher dimensions, *Computational Geometry: Theory and Applications* **1** (1992), 189–201.

[2] N. Alon, S. Rajagopalan and S. Suri, Long non-crossing configurations in the plane, *Fundamenta Informaticae* **22** (1995), 385–394.

[3] E. M. Arkin and R. Hassin, Approximation algorithms for the geometric covering salesman problem, *Discrete Applied Mathematics* **55** (1994), 197–218.

[4] A. I. Barvinok, S. Fekete, D. S. Johnson, A. Tamir, G. J. Woeginger, and R. Woodroofe, The geometric maximum traveling salesman problem, *Journal of the ACM* **50(5)** (2003), 641–664.

[5] M. Bern and D. Eppstein, Approximation algorithms for geometric problems, in *Approximation Algorithms for NP-hard Problems* (D. S. Hochbaum, editor), PWS Publishing Company, Boston, MA, 1997, pp. 296–345.

[6] B. K. Bhattacharya and G. T. Toussaint, Efficient algorithms for computing the maximum distance between two finite planar sets, *Journal of Algorithms* **4** (1983), 121–136.

[7] A. Biniaz, P. Bose, K. Crosbie, J.-L. De Carufel, D. Eppstein, A. Maheshwari, and M. Smid, Maximum plane trees in multipartite geometric graphs, *Algorithmica* **81(4)** (2019), 1512–1534.

[8] R. Dorrigiv, R. Fraser, M. He, S. Kamali, A. Kawamura, A. López-Ortiz, and D. Seco, On minimum- and maximum-weight minimum spanning trees with neighborhoods, *Theory of Computing Systems* **56(1)** (2015), 220–250.

[9] A. Dumitrescu and Cs. D. Tóth, Long non-crossing configurations in the plane, *Discrete and Computational Geometry* **44(4)** (2010), 727–752.

[10] D. Eppstein, Spanning trees and spanners, in *Handbook of Computational Geometry* (J.-R. Sack and J. Urrutia, editors), Elsevier Science, Amsterdam, 2000, pp. 425–461.

[11] S. Fekete, Simplicity and hardness of the maximum traveling salesman problem under geometric distances, *Proc. 10th Annual ACM-SIAM Symposium on Discrete Algorithms*, ACM/SIAM, 1999, pp. 337–345.

[12] J. S. B. Mitchell, Geometric shortest paths and network optimization, in *Handbook of Computational Geometry* (J.-R. Sack and J. Urrutia, editors), Elsevier Science, Amsterdam, 2000, pp. 633–701.

[13] J. S. B. Mitchell, Shortest paths and networks, in *Handbook of Computational Geometry* (J. E. Goodman, J. O'Rourke, and C. D. Tóth, editors), 3rd edition, CRC Press, Boca Raton, FL, 2017, pp. 811–848.

[14] C. Monma, M. Paterson, S. Suri and F. Yao, Computing Euclidean maximum spanning trees, *Algorithmica* **5** (1990), 407–419.

[15] J. Pach and P. K. Agarwal, *Combinatorial Geometry*, John Wiley, New York, 1995.

[16] F. P. Preparata and M. I. Shamos, *Computational Geometry*, Springer-Verlag, New York, 1985.

[17] E. A. Ramos, An optimal deterministic algorithm for computing the diameter of a three-dimensional point set, *Discrete and Computational Geometry* **26(2)** (2001), 233–244.

[18] J. M. Robert, Maximum distance between two sets of points in $\mathbb{R}^d$, *Pattern Recognition Letters* **14** (1993), 733–735.

[19] G. T. Toussaint and M. A. McAlear, A simple $O(n \log n)$ algorithm for finding the maximum distance between two finite planar sets, *Pattern Recognition Letters* **1** (1982), 21–24.

[20] Y. Yang, M. Lin, J. Xu, and Y. Xie, Minimum spanning trees with neighborhoods, In *Proc. 3rd Int. Conf. on Algor. Aspects in Information and Management* (AAIM), vol. 4508 of LNCS, Springer, 2007, pp. 306–316.

Chapter 5

Stretch Factor of Polygonal Chains[1]

5.1 Introduction

Given a set S of n point sites in a Euclidean space $\mathbb{R}^d$, what is the best way to connect S into a *geometric network (graph)*? This question has motivated researchers for a long time, going back as far as the 1940s, and beyond [20, 36]. Numerous possible criteria for a good geometric network have been proposed, perhaps the most basic being the *length*. In 1955, Few [21] showed that for any set of n points in a unit square, there is a traveling salesman tour of length at most $\sqrt{2n} + 7/4$. This was improved to at most $0.984\sqrt{2n} + 11$ by Karloff [24]. Similar bounds hold for the shortest spanning tree and the shortest rectilinear spanning tree [14, 17, 22]. Besides length, two further key factors in the quality of a geometric network are the *vertex dilation* and the *geometric dilation* [32], both of which measure how closely shortest paths in a network approximate the Euclidean distances between their endpoints.

The *dilation* (also called *stretch factor* [30] or *detour* [2]) between two points p and q in a geometric graph G is defined as the ratio between the length of a shortest path from p to q and the Euclidean distance $|pq|$. The *dilation* of the graph G is the maximum dilation over all pairs of vertices in G. A graph in which the dilation is bounded above by $t \geq 1$ is also called a *t-spanner* (or simply a *spanner* if t is a constant). A complete graph in

[1]Joint work with Wolfgang Mulzer and Csaba D. Tóth

Euclidean space is clearly a 1-spanner. Therefore, researchers focused on the dilation of graphs with certain additional constraints, for example, noncrossing (i.e., plane) graphs. In 1989, Das and Joseph [16] identified a large class of plane spanners (characterized by two simple local properties). Bose et al. [7] gave an algorithm that constructs for any set of planar sites a plane 11-spanner with bounded degree. On the other hand, Eppstein [19] analyzed a fractal construction showing that *β-skeletons*, a natural class of geometric networks, can have arbitrarily large dilation.

The study of dilation also raises algorithmic questions. Agarwal et al. [2] described randomized algorithms for computing the dilation of a given path (on n vertices) in $\mathbb{R}^2$ in $O(n \log n)$ expected time. They also presented randomized algorithms for computing the dilation of a given tree, or cycle, in $\mathbb{R}^2$ in $O(n \log^2 n)$ expected time. Previously, Narasimhan and Smid [31] showed that an $(1+\varepsilon)$-approximation of the stretch factor of any path, cycle, or tree can be computed in $O(n \log n)$ time. Klein et al. [25] gave randomized algorithms for a path, tree, or cycle in $\mathbb{R}^2$ to count the number of vertex pairs whose dilation is below a given threshold in $O(n^{3/2+\varepsilon})$ expected time. Cheong et al. [13] showed that it is NP-hard to determine the existence of a spanning tree on a planar point set whose dilation is at most a given value. More results on plane spanners can be found in the monograph dedicated to this subject [32] or in several surveys [9, 18, 30].

We investigate a basic question about the dilation of polygonal chains. We ask how the dilation between the endpoints of a polygonal chain (which we will call the *stretch factor*, to distinguish it from the more general notion of dilation) is influenced by *fingerprint* properties of the chain, i.e., by properties that are defined on $O(1)$-size subsets of the vertex set. Such fingerprint properties play an important role in geometry; classic examples include the *Carathéodory property*[2] [27, Theorem 1.2.3] or the *Helly property*[3] [27, Theorem 1.3.2]. In general, determining the effect of a fingerprint property may prove elusive—given n points

[2]Given a finite set S of points in d dimensions, if every $d+2$ points in S are in convex position, then S is in convex position.

[3]Given a finite collection of convex sets in d dimensions, if every $d+1$ sets have nonempty intersection, then all sets have nonempty intersection.

in the plane, consider the simple property that every 3 points determine 3 distinct distances. It is unknown [10, p. 203] whether this property implies that the total number of distinct distances grows superlinearly in n. Furthermore, fingerprint properties appear in the general study of *local versus global properties of metric spaces*, which is highly relevant to combinatorial approximation algorithms based on mathematical programming relaxations [6].

In the study of dilation, interesting fingerprint properties have also been found. For example, a (continuous) curve C is said to have the *increasing chord property* [15, 26] if for any points a, b, c, d that appear on C in this order, we have $|ad| \geq |bc|$. The increasing chord property implies that C has (geometric) dilation at most $2\pi/3$ [34]. A weaker property is the *self-approaching property*: a (continuous) curve C is self-approaching if for any points a, b, c that appear on C in this order, we have $|ac| \geq |bc|$. Self-approaching curves have dilation at most 5.332 [23] (see also [4]), and they have found interesting applications in the field of graph drawing [5, 8, 33].

We introduce a new natural fingerprint property and see that it can constrain the stretch factor of a polygonal chain, but only in a weaker sense than one may expect; we also provide algorithmic results on this property. Before providing details, we give a few basic definitions.

Definitions A *polygonal chain* P in $\mathbb{R}^d$ is specified by a sequence of n points $(p_1, p_2, \ldots, p_n)$, called *vertices*. The chain P consists of $n-1$ line segments between consecutive vertices. We say P is *simple* if only consecutive line segments intersect and they only intersect at their endpoints. Given a polygonal chain P in $\mathbb{R}^d$ with n vertices and a parameter $c \geq 1$, we call P a *c-chain* if for all $1 \leq i < j < k \leq n$, we have

$$|p_i p_j| + |p_j p_k| \leq c|p_i p_k|. \tag{5.1}$$

Observe that the c-chain condition is a fingerprint condition that is not really a local dilation condition—it is more a combination between the local chain substructure and the distribution of the points in the subchains.

The *stretch factor* δ_P of P is defined as the dilation between the two end points p_1 and p_n of the chain:

$$\delta_P = \frac{\sum_{i=1}^{n-1} |p_i p_{i+1}|}{|p_1 p_n|}.$$

Note that this definition is different from the more general notion of dilation (also called *stretch factor* [30]) of a graph which is the maximum dilation over all pairs of vertices. Since there is no ambiguity in this chapter, we will just call δ_P the stretch factor of P.

For example, the polygonal chain $P = ((0,0),(1,0),\ldots,(n,0))$ in $\mathbb{R}^2$ is a 1-chain with stretch factor 1; and $Q = ((0,0),(0,1),(1,1),(1,0))$ is a $(\sqrt{2}+1)$-chain with stretch factor 3.

Without affecting the results, the floor and ceiling functions are omitted in our calculations. For a positive integer t, let $[t] = \{1,2,\ldots,t\}$. For a point set S, let $\operatorname{conv}(S)$ denote the convex hull of S. All logarithms are in base 2, unless stated otherwise.

Our results In the Euclidean plane $\mathbb{R}^2$, we deduce three upper bounds on the stretch factor of a c-chain P with n vertices (Section 5.2). In particular, we have (i) $\delta_P \le c(n-1)^{\log c}$, (ii) $\delta_P \le c(n-2)+1$, and (iii) $\delta_P = O\left(c^2\sqrt{n-1}\right)$.

From the other direction, we obtain the following lower bound in $\mathbb{R}^2$ (Section 5.3): For every $c \ge 4$, there is a family $\mathcal{P}_c = \{P^m\}_{m\in\mathbb{N}}$ of simple c-chains, so that P^m has $n = 4^m + 1$ vertices and stretch factor $(n-1)^{\frac{1+\log(c-2)-\log c}{2}}$, where the exponent converges to $1/2$ as c tends to infinity. The lower bound construction does not extend to the case of $1 < c < 4$, which remains open.

Then we generalize the results to higher dimensional Euclidean spaces (Section 5.4): For all integers $d \ge 2$, we show that any c-chain P with n vertices in $\mathbb{R}^d$ has stretch factor $\delta_P = O\left((n-1)^{(d-1)/d}\right)$. On the other hand, for any constant $\varepsilon > 0$ and sufficiently large $c = \Omega(d)$, we construct a c-chain in $\mathbb{R}^d$ with n vertices and stretch factor at least $(n-1)^{(1-\varepsilon)(d-1)/d}$.

Finally, we present two algorithmic results (Section 5.5) for all fixed dimensions $d \ge 2$: (i) A randomized algorithm that decides, given a polygonal chain P in $\mathbb{R}^d$ with n vertices and a threshold $c > 1$, whether P is a c-chain in $O\left(n^{3-1/d} \text{ polylog } n\right)$ expected time and $O(n \log n)$ space. (ii) As a corollary, there is a randomized algorithm that finds, for a polygonal chain P with n vertices, the minimum $c \geq 1$ for which P is a c-chain in $O\left(n^{3-1/d} \text{ polylog } n\right)$ expected time and $O(n \log n)$ space.

5.2 Upper Bounds in the Plane

At first glance, one might expect the stretch factor of a c-chain, for $c \geq 1$, to be bounded by some function of c. For example, the stretch factor of a 1-chain is necessarily 1. We derive three upper bounds on the stretch factor of a c-chain with n vertices in terms of c and n (cf. Theorems 5.1–5.3); see Figure 5.1 for a visual comparison between the bounds. For large n, the bound in Theorem 5.1 is the best for $1 \leq c \leq 2^{1/2}$, while the bound in Theorem 5.3 is the best for $c > 2^{1/2}$. In particular, the bound in Theorem 5.1 is tight for $c = 1$. When n is comparable with c, more specifically, for $c \geq 2$ and $n \leq 64c^2 + 2$, the bound in Theorem 5.2 is the best.

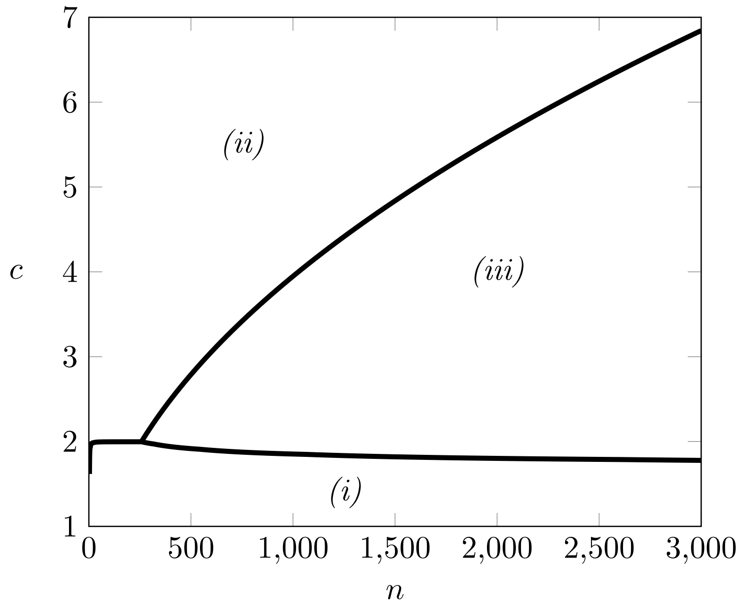


Figure 5.1: The values of n and c for which (i) Theorem 5.1: $\delta_P \leq c(n-1)^{\log c}$, (ii) Theorem 5.2: $\delta_P \leq c(n-2)+1$, and (iii) Theorem 5.3: $\delta_P \leq 8c^2\sqrt{n-1}$ give the current best upper bound.

Our first upper bound is obtained by a recursive application of the c-chain property. It holds for any positive distance function that need not even satisfy the triangle inequality.

Theorem 5.1. *For a c-chain P with n vertices, we have $\delta_P \leq c(n-1)^{\log c}$.*

Proof. We prove, by induction on n, that

$$\delta_P \leq c^{\lceil \log(n-1) \rceil}, \tag{5.2}$$

for every c-chain P with $n \geq 2$ vertices. In the base case, $n = 2$, we have $\delta_P = 1$ and $c^{\lceil \log(2-1) \rceil} = 1$. Now let $n \geq 3$, and assume that (5.2) holds for every c-chain with fewer than n vertices. Let $P = (p_1, \ldots, p_n)$ be a c-chain with n vertices. Then, applying (5.2) to the first and second half of P, followed by the c-chain property for the first, middle, and last vertex of P, we get

$$\begin{aligned}
\sum_{i=1}^{n-1} |p_i p_{i+1}| &\leq \sum_{i=1}^{\lceil n/2 \rceil - 1} |p_i p_{i+1}| + \sum_{i=\lceil n/2 \rceil}^{n-1} |p_i p_{i+1}| \\
&\leq c^{\lceil \log(\lceil n/2 \rceil - 1) \rceil} \left(|p_1 p_{\lceil n/2 \rceil}| + |p_{\lceil n/2 \rceil} p_n| \right) \\
&\leq c^{\lceil \log(\lceil n/2 \rceil - 1) \rceil} \cdot c|p_1 p_n| \\
&\leq c^{\lceil \log(n-1) \rceil} |p_1 p_n|,
\end{aligned}$$

so (5.2) holds also for P. Consequently,

$$\delta_P \leq c^{\lceil \log(n-1) \rceil} \leq c^{\log(n-1)+1} = c \cdot c^{\log(n-1)} = c\,(n-1)^{\log c},$$

as required. ☐

Our second bound combines the c-chain property with the triangle inequality, and it holds in any metric space.

Theorem 5.2. *For a c-chain P with n vertices, we have $\delta_P \leq c(n-2)+1$.*

Proof. Without loss of generality, assume that $|p_1p_n| = 1$. By the c-chain property, every point p_i, $1 < i < n$, lies in an ellipse with foci p_1 and p_n whose major axis has length c; see Figure 5.2. Hence,

$$|p_1p_i| \le c - |p_ip_n|. \tag{5.3}$$

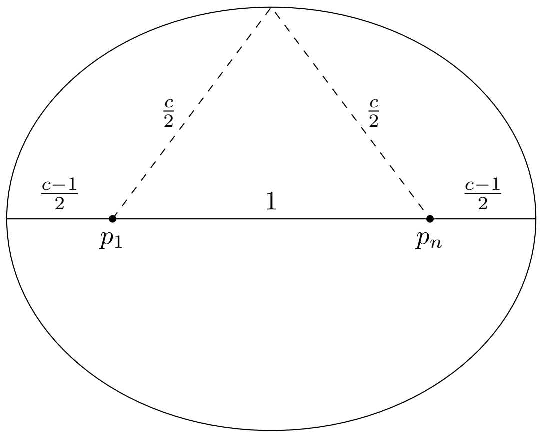


Figure 5.2: The entire chain P lies in an ellipse with foci p_1 and p_n.

The triangle inequality yields

$$|p_1p_i| \le |p_1p_n| + |p_np_i| = 1 + |p_ip_n|. \tag{5.4}$$

The combination of (5.3) and (5.4) gives $|p_1p_i| \le \frac{c+1}{2}$. Analogous argument for p_n (in place of p_1) yields $|p_ip_n| \le \frac{c+1}{2}$.

For every pair $1 < i < j < n$, the triangle inequality implies

$$2|p_ip_j| \le (|p_ip_1| + |p_1p_j|) + (|p_ip_n| + |p_np_j|) = (|p_1p_i| + |p_ip_n|) + (|p_1p_j| + |p_jp_n|) \le 2c,$$

hence $|p_ip_j| \le c$. Overall, the stretch factor of P is bounded above by

$$\begin{aligned}\delta_P = \frac{\sum_{j=1}^{n-1} |p_jp_{j+1}|}{|p_1p_n|} &= |p_1p_2| + |p_{n-1}p_n| + \sum_{j=2}^{n-2} |p_jp_{j+1}| \\ &\le \frac{c+1}{2} + \frac{c+1}{2} + c(n-3) = c(n-2) + 1,\end{aligned}$$

as claimed. $\square$

Our third upper bound[4] uses properties of the Euclidean plane to bound the number of long edges in P. The key observation is that the starting points of two long edges cannot be too close to each other for the c-chain property to hold. Since the entire chain lies in an ellipse with diameter c, a volume argument provides an upper bound of the number of long edges.

Theorem 5.3. *For a c-chain P with n vertices, we have $\delta_P = O\left(c^2\sqrt{n-1}\right)$.*

5.3 Lower Bounds in the Plane

We now present our lower bound construction, showing that the dependence on n for the stretch factor of a c-chain cannot be avoided.

Theorem 5.4. *For every constant $c \geq 4$, there is a set $\mathcal{P}_c = \{P^m\}_{m\in\mathbb{N}}$ of simple c-chains, so that P^m has $n = 4^m + 1$ vertices and stretch factor $(n-1)^{\frac{1+\log(c-2)-\log c}{2}}$.*

By Theorem 5.3, the stretch factor of a c-chain in the plane is $O\left((n-1)^{1/2}\right)$ for every constant $c \geq 1$. Since

$$\lim_{c\to\infty} \frac{1+\log(c-2)-\log c}{2} = \frac{1}{2},$$

our lower bound construction shows that the limit of the exponent cannot be improved. Indeed, for every $\varepsilon > 0$, we can set $c = \frac{2^{2\varepsilon+1}}{2^{2\varepsilon}-1}$, and then the chains above have stretch factor

$$(n-1)^{\frac{1+\log(c-2)-\log c}{2}} = (n-1)^{1/2-\varepsilon} = \Omega(n^{1/2-\varepsilon}).$$

We first construct a family $\mathcal{P}_c = \{P^m\}_{m\in\mathbb{N}}$ of polygonal chains. Then we show, in Lemmata 5.5 and 5.7, that every chain in $\mathcal{P}_c$ is simple and indeed a c-chain. The theorem follows since the claimed stretch factor is a consequence of the construction.

[4]Theorem 5.3 is mainly the work of Csaba D. Tóth.

Construction of $\mathcal{P}_c$ The construction here is a generalization of the iterative construction of the *Koch curve*; when $c = 6$, the result is the original Cesàro fractal (which is a variant of the Koch curve) [11]. We start with a unit line segment P^0, and for $m = 0, 1, \dots,$ we construct P^{m+1} by replacing each segment in P^m by four segments such that the middle three points achieve a stretch factor of $c_* = \frac{c-2}{2}$ (this choice will be justified in the proof of Lemma 5.7). Note that $c_* \geq 1$, since $c \geq 4$.

We continue with the details. Let P^0 be the unit line segment from $(0, 0)$ to $(1, 0)$; see Figure 5.3 (left). Given the polygonal chain P^m ($m = 0, 1, \dots$), we construct P^{m+1} by replacing each segment of P^m by four segments as follows. Consider a segment of P^m, and denote its length by ℓ. Subdivide this segment into three segments of lengths $(\frac{1}{2} - \frac{a}{c_*})\ell$, $\frac{2a}{c_*}\ell$, and $(\frac{1}{2} - \frac{a}{c_*})\ell$, respectively, where $0 < a < \frac{c_*}{2}$ is a parameter to be determined later. Replace the middle segment with the top part of an isosceles triangle of side length $a\ell$. The chains P^0, P^1, P^2, and P^4 are depicted in Figures 5.3 and 5.4.

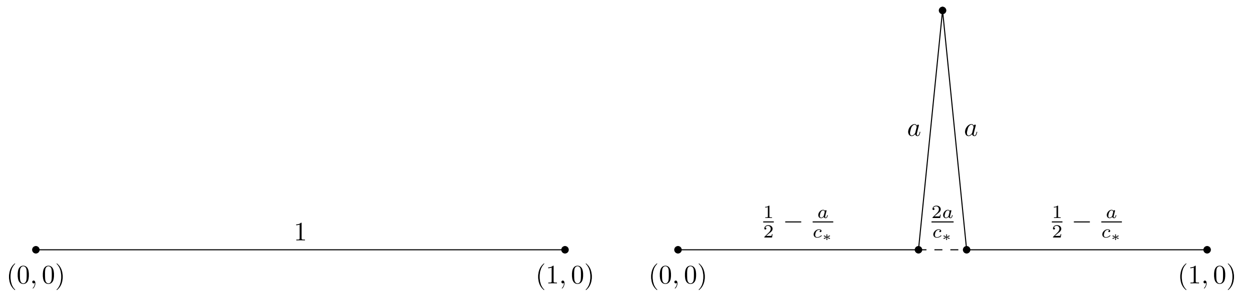


Figure 5.3: The chains P^0 (left) and P^1 (right).

Note that each segment of length ℓ in P^m is replaced by four segments of total length $(1 + \frac{2a(c_*-1)}{c_*})\ell$. After m iterations, the chain P^m consists of 4^m line segments of total length $\left(1 + \frac{2a(c_*-1)}{c_*}\right)^m$.

By construction, the chain P^m (for $m \geq 1$) consists of four scaled copies of P^{m-1}. For $i = 1, 2, 3, 4$, let the *ith subchain of* P^m be the subchain of P^m consisting of 4^{m-1} segments starting from the $((i-1)4^{m-1} + 1)$th segment. By construction, the ith subchain of P^m is similar to the chain P^{m-1}, for $i = 1, 2, 3, 4$.[5] The following functions allow us to refer to

[5]Two geometric shapes are *similar* if one can be obtained from the other by translation, rotation, and

these subchains formally. For $i = 1, 2, 3, 4$, define a function $f_i^m : P^m \to P^m$ as the identity on the ith subchain of P^m that sends the remaining part(s) of P^m to the closest endpoint(s) along this subchain. So $f_i^m(P^m)$ is similar to P^{m-1}. Let $g_i : \mathcal{P}_c \setminus \{P^0\} \to \mathcal{P}_c$ be a piecewise defined function such that $g_i(C) = \sigma^{-1} \circ f_i^m \circ \sigma(C)$ if C is similar to P^m, where $\sigma : C \to P^m$ is a similarity transformation. Applying the function g_i on a chain P^m can be thought of as "cutting out" its ith subchain.

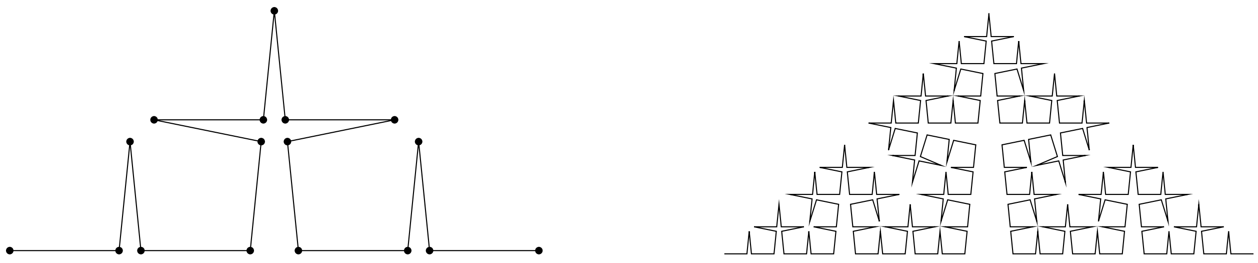


Figure 5.4: The chains P^2 (left) and P^4 (right).

Clearly, the stretch factor of the chain monotonically increases with the parameter a. However, if a is too large, the chain is no longer simple. The following lemma gives a sufficient condition for the constructed chains to avoid self-crossings.

Lemma 5.5. *For every constant $c \geq 4$, if $a \leq \frac{c-2}{2c}$, then every chain in $\mathcal{P}_c$ is simple.*

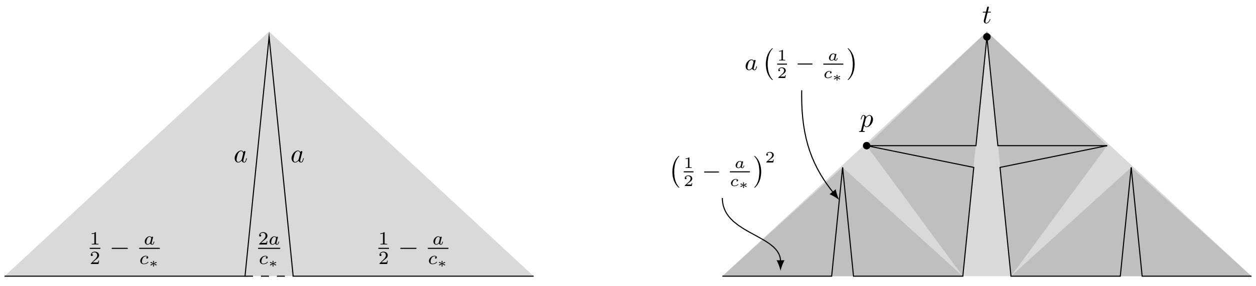
Proof. Let $T = \text{conv}(P^1)$. Observe that T is an isosceles triangle; see Figure 5.5 (left). We first show the following:

Claim 5.6. *If $a \leq \frac{c-2}{2c}$, then* $\text{conv}(P^m) = T$ *for all $m \geq 1$.*

Proof. We prove the claim by induction on m. It holds for $m = 1$ by definition. For the induction step, assume that $m \geq 2$ and that the claim holds for $m - 1$. Consider the chain P^m. Since it contains all the vertices of P^1, $T \subset \text{conv}(P^m)$. So we only need to show that $\text{conv}(P^m) \subset T$.

By construction, $P^m \subset \bigcup_{i=1}^4 \text{conv}(g_i(P^m))$; see Figure 5.5 (right). By the inductive hypothesis, $\text{conv}(g_i(P^m))$ is an isosceles triangle similar to T, for $i = 1, 2, 3, 4$. Since the

scaling; and are *congruent* if one can be obtained from the other by translation and rotation.



a
a
$\frac{1}{2}-\frac{a}{c_*}$
$\frac{2a}{c_*}$
$\frac{1}{2}-\frac{a}{c_*}$
t
$a\left(\frac{1}{2}-\frac{a}{c_*}\right)$
p
$\left(\frac{1}{2}-\frac{a}{c_*}\right)^2$

Figure 5.5: Left: Convex hull T of P^1 in light gray; Right: Convex hulls of $g_i(P^2)$, $i = 1, 2, 3, 4$, in dark gray, are contained in T.

bases of $\text{conv}(g_1(P^m))$ and $\text{conv}(g_4(P^m))$ are collinear with the base of T by construction, due to similarity, they are contained in T. The base of $\text{conv}(g_2(P^m))$ is contained in T. In order to show $\text{conv}(g_2(P^m)) \subset T$, by convexity, it suffices to ensure that its apex p is also in T. Note that the coordinates of the top point is $t = \left(1/2, a\sqrt{c_*^2-1}/c_*\right)$, so the supporting line ℓ of the left side of T is

$$y = \frac{2a\sqrt{c_*^2-1}}{c_*}x, \text{ and}$$
$$p = \left(\frac{1}{2} - \frac{a}{2c_*} - \frac{a^2\left(c_*^2-1\right)}{c_*^2}, \left(\frac{a}{2c_*} + \frac{a^2}{c_*^2}\right)\sqrt{c_*^2-1}\right).$$

By the condition of $a \le \frac{c-2}{2c} = \frac{c_*}{2(c_*+1)}$ in the lemma, p lies on or below ℓ. Under the same condition, we have $\text{conv}(g_3(P^m)) \subset T$ by symmetry. Then $P^m \subset \bigcup_{i=1}^4 \text{conv}(g_i(P^m)) \subset T$. Since T is convex, $\text{conv}(P^m) \subset T$. So $\text{conv}(P^m) = T$, as claimed. □

We can now finish the proof of Lemma 5.5 by induction. Clearly, P^0 and P^1 are simple. Assume that $m \ge 2$, and P^{m-1} is simple. Consider the chain P^m. For $i = 1, 2, 3, 4$, $g_i(P^m)$ is similar to P^{m-1}, hence simple by the inductive hypothesis. Since $P^m = \bigcup_{i=1}^4 g_i(P^m)$, it is sufficient to show that for all $i, j \in \{1, 2, 3, 4\}$, where $i \ne j$, a segment in $g_i(P^m)$ does not intersect any segments in $g_j(P^m)$, unless they are consecutive in P^m and they intersect at a common endpoint. This follows from the above claim together with the observation that for $i \ne j$, the intersection $g_i(P^m) \cap g_j(P^m)$ is either empty or contains a single vertex which is the common endpoint of two consecutive segments in P^m. □

In the remainder of this section, we assume that

$$a = \frac{c-2}{2c} = \frac{c_*}{2(c_*+1)}. \tag{5.5}$$

Under this assumption, all segments in P^1 have the same length a. Therefore, by construction, all segments in P^m have the same length

$$a^m = \left(\frac{c_*}{2(c_*+1)}\right)^m.$$

There are 4^m segments in P^m, with 4^m+1 vertices, and its stretch factor is

$$\delta_{P^m} = 4^m \left(\frac{c_*}{2(c_*+1)}\right)^m = \left(\frac{2c_*}{c_*+1}\right)^m.$$

Consequently, $m = \log_4(n-1) = \frac{\log(n-1)}{2}$, and

$$\delta_{P^m} = \left(\frac{2c_*}{c_*+1}\right)^{\frac{\log(n-1)}{2}} = \left(\frac{2c-4}{c}\right)^{\frac{\log(n-1)}{2}} = (n-1)^{\frac{1+\log(c-2)-\log c}{2}},$$

as claimed. To finish the proof of Theorem 5.4, it remains to show the constructed polygonal chains are indeed c-chains.

Lemma 5.7. *For every constant $c \geq 4$, $\mathcal{P}_c$ is a family of c-chains.*

We first prove a couple of facts that will be useful in the proof of Lemma 5.7. We defer an intuitive explanation until after the formal statement of the following lemma.

Lemma 5.8. *Let $m \geq 1$ and let $P^m = (p_1, p_2, \ldots, p_n)$, where $n = 4^m+1$. Then the following hold:*

(i) There exists a sequence $(q_1, q_2, \ldots, q_\ell)$ of $\ell = 2 \cdot 4^{m-1}$ points in $\mathbb{R}^2$ such that the chain $R^m = (p_1, q_1, p_2, q_2, \ldots, p_\ell, q_\ell, p_{\ell+1})$ is similar to P^m.

(ii) For $m \geq 2$, define $g_5 : \mathcal{P}_c \setminus \{P^0, P^1\} \to \mathcal{P}_c$ by

$$g_5(P^m) = (g_3 \circ g_2(P^m)) \cup (g_4 \circ g_2(P^m)) \cup (g_1 \circ g_3(P^m)) \cup (g_2 \circ g_3(P^m)) \,.$$

Then $g_5(P^m)$ is similar to P^{m-1}.

Part (i) of Lemma 5.8 says that given P^m, we can construct a chain R^m similar to P^m by inserting one point between every two consecutive points of the left half of P^m, see Figure 5.6 (left). Part (ii) says that the "top" subchain of P^m that consists of the right half of $g_2(P^m)$ and the left half of $g_3(P^m)$, see Figure 5.6 (right), is similar to P^{m-1}.

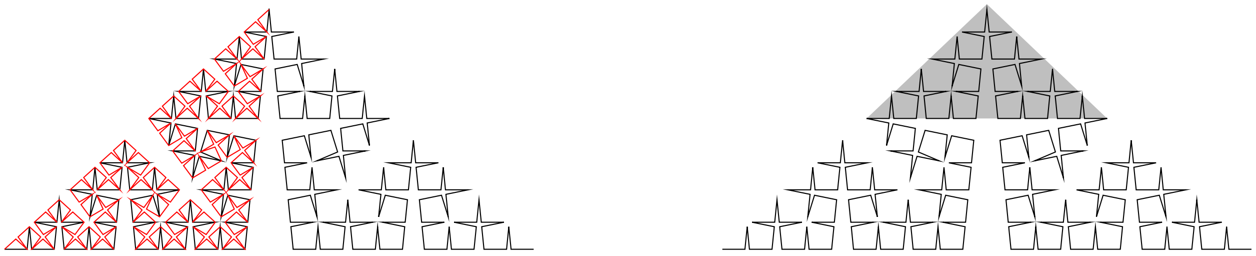


Figure 5.6: Left: Chain P^m with the scaled copy of itself R^m (in red); Right: Chain P^m with its subchain $g_5(P^m)$ marked by its convex hull.

Proof of Lemma 5.8. For part (i), we review the construction of P^m, and show that R^m and P^m can be constructed in a coupled manner. In Figure 5.7 (left), consider $P^1 = (p_1, p_2, p_3, p_4, p_5)$. Recall that all segments in P^1 are of the same length $a = \frac{c_*}{2(c_*+1)}$. The isosceles triangles $\Delta p_1p_2p_3$ and $\Delta p_1p_3p_5$ are similar. Let $\sigma : \Delta p_1p_3p_5 \to \Delta p_1p_2p_3$ be the similarity transformation. Let $q_1 = \sigma(p_2)$ and $q_2 = \sigma(p_4)$. By construction, the chain $R^1 = (p_1, q_1, p_2, q_2, p_3)$ is similar to P^1. In particular, all of its segments have the same length, and so the isosceles triangle $\Delta p_1q_1p_2$ is similar to $\Delta p_1p_3p_5$. Moreover, its base is the segment p_1p_2, so $\Delta p_1q_1p_2$ is precisely $\text{conv}(g_1(P^2))$, see Figure 5.7 (right).

Write $P^2 = (v_1, v_2, \ldots, v_{17})$, then $v_3 = q_1$ by the above argument and $v_7 = q_2$ by symmetry. Now $\Delta v_1v_2v_3$, $\Delta v_3v_4v_5$, $\Delta v_5v_6v_7$, and $\Delta v_7v_8v_9$ are four congruent isosceles triangles, all of which are similar to $\Delta v_1v_9v_{17}$, since the angles are the same. Repeat the above procedure

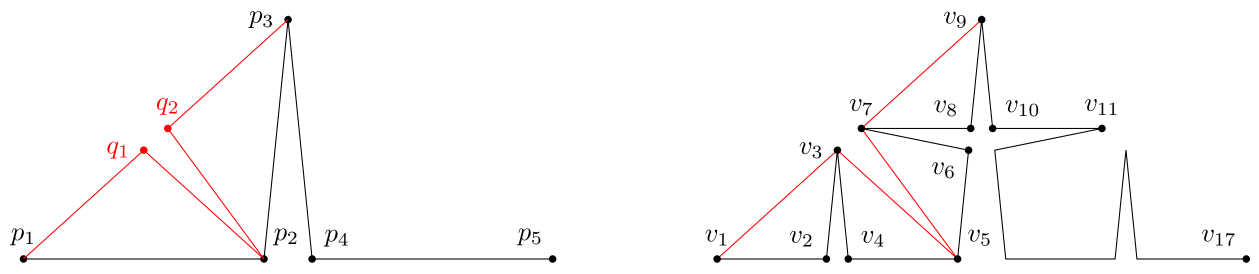


Figure 5.7: Left: the chains P^1 and R^1 (red); Right: the chains P^2 and R^1 (red).

on each of them to obtain $R^2 = (v_1, u_1, v_2, u_2, \ldots, v_8, u_8, v_9)$, which is similar to P^2. Continue this construction inductively to get the desired chain R^m for any $m \geq 1$.

For part (ii), see Figure 5.7 (right). By definition, $g_5(P^2)$ is the subchain $(v_7, v_8, v_9, v_{10}, v_{11})$. Observe that the segments v_7v_8 and $v_{10}v_{11}$ are collinear by symmetry. Moreover, they are parallel to v_1v_{17} since $\angle v_7v_8v_9 = \angle v_1v_5v_9$. So $g_5(P^2)$ is similar to P^1; see Figure 5.7 (left). Then for $m \geq 2$, $g_5(P^m)$ is the subchain of P^m starting at vertex v_7, ending at vertex v_{11}. By the construction of P^m, $g_5(P^m)$ is similar to P^{m-1}. $\square$

Proof of Lemma 5.7. We proceed by induction on m again. The claim is vacuously true for P^0. For P^1, among all ten choices of $1 \leq i < j < k \leq 5$, $\frac{|p_2p_3|+|p_3p_4|}{|p_2p_4|} = c_* = \frac{c-2}{2} < c$ is the largest, and so P^1 is also a c-chain. Assume that $m \geq 2$ and P^{m-1} is a c-chain. We need to show that P^m is also a c-chain. Consider a triplet of vertices $\{p_i, p_j, p_k\} \subset P^m$, where $1 \leq i < j < k \leq n = 4^m + 1$.

Recall that P^m consists of four copies of the subchain P^{m-1}, namely $g_1(P^m)$, $g_2(P^m)$, $g_3(P^m)$, and $g_4(P^m)$, see Figure 5.8 (left). If $\{p_i, p_j, p_k\} \subset g_l(P^m)$ for any $l = 1, 2, 3, 4$, then by the induction hypothesis,

$$\frac{|p_ip_j| + |p_jp_k|}{|p_ip_k|} \leq c.$$

So we may assume that p_i and p_k belong to two different $g_l(P^m)$'s. There are four cases to consider up to symmetry:

Case 1. $p_i \in g_1(P^m)$ and $p_k \in g_2(P^m)$;

Case 2. $p_i \in g_1(P^m)$ and $p_k \in g_3(P^m)$;

Case 3. $p_i \in g_1(P^m)$ and $p_k \in g_4(P^m)$;

Case 4. $p_i \in g_2(P^m)$ and $p_k \in g_3(P^m)$.

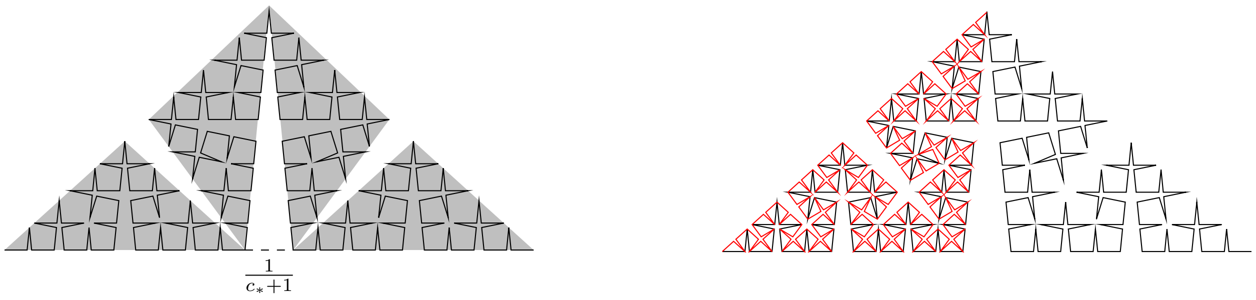


Figure 5.8: Left: Chain P^m with its four subchains of type P^{m-1} marked by their convex hulls; Right: Chain P^m with the scaled copy of itself R^m (in red) constructed in Lemma 5.8 (i).

By Lemma 5.8 (i), the vertex set of $g_1(P^m) \cup g_2(P^m)$ is contained in the chain R^m shown in Figure 5.8 (right). If we are in Case 1, i.e., $p_i \in g_1(P^m)$ and $p_k \in g_2(P^m)$, then p_i, p_j, p_k can be thought of as vertices of R^m. The similarity between R^m and P^m, maps points p_i, p_j, p_k to suitable points $p'_i, p'_j, p'_k \in P^m$ such that

$$\frac{|p'_i p'_j| + |p'_j p'_k|}{|p'_i p'_k|} = \frac{|p_i p_j| + |p_j p_k|}{|p_i p_k|}.$$

Since $p_i \in g_1(R^m) \cup g_2(R^m)$ while $p_k \in g_3(R^m) \cup g_4(R^m)$, the triplet (p'_i, p'_j, p'_k) does not belong to Case 1. In other words, Case 1 can be represented by other cases.

Recall that in Lemma 5.5, we showed that $\operatorname{conv}(P^m)$ is an isosceles triangle T of diameter 1. Observe that if $|p_i p_k| \geq \frac{1}{c_*+1}$, then

$$\frac{|p_i p_j| + |p_j p_k|}{|p_i p_k|} \leq \frac{1+1}{\frac{1}{c_*+1}} = 2c_* + 2 = c,$$

as required. So we may assume that $|p_i p_k| < \frac{1}{c_*+1}$, therefore only Case 4 remains, i.e., $p_i \in g_2(P^m)$ and $p_k \in g_3(P^m)$.

By Lemma 5.8 (ii), the "top" subchain $g_5(P^m)$ of P^m is also similar to P^{m-1}, see Figure 5.9 (left). If p_i and p_k are both in $g_5(P^m)$, i.e., $p_i \in (g_3 \circ g_2(P^m)) \cup (g_4 \circ g_2(P^m))$ and

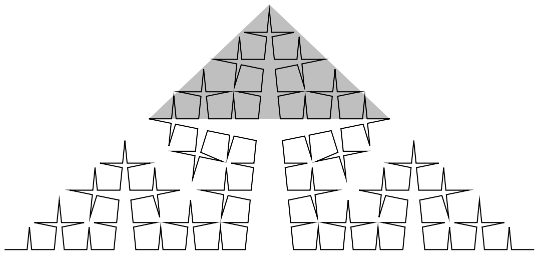
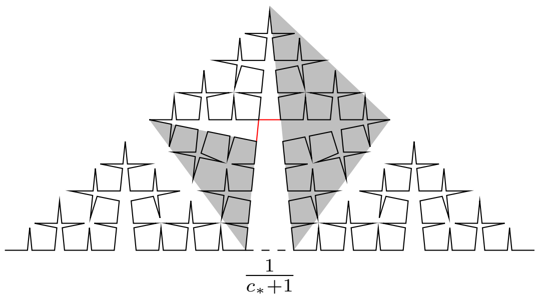



Figure 5.9: Left: Chain P^m with its subchain $g_5(P^m)$ marked by its convex hull; Right: The last case where p_i is in the left shaded subchain and p_k is in the right shaded subchain.

$p_k \in (g_1 \circ g_3(P^m)) \cup (g_2 \circ g_3(P^m))$, then so is p_j.

By the induction hypothesis, we have

$$\frac{|p_ip_j| + |p_jp_k|}{|p_ip_k|} \leq c.$$

So we may assume that at least one of p_i and p_k is not in $g_5(P^m)$. Without loss of generality, let $p_i \in g_2(P^m)\backslash g_5(P^m)$. The similarity that maps P^{m-1} to $g_2(P^m)$ and $g_5(P^m)$, respectively, have the same scaling factor of $a = \frac{c_*}{2(c_*+1)}$, and they carry the bottom dashed segment in Figure 5.9 (right), to the two red segments.

Claim 5.9. *If $p_i \in g_2(P^m) \setminus g_5(P^m)$ and $p_k \in g_3(P^m)$, then $|p_ip_k| > \frac{c_*}{2(c_*+1)^2}$.*

Proof. As noted above, we assume that p_i is in $\text{conv}(g_2(P^m) \setminus g_5(P^m)) = \Delta q_1q_2q_3$ in Figure 5.10. If $p_k \in g_5(P^m) \cap g_3(P^m) = \Delta q_7q_6q_5$, then the configuration is illustrated in Figure 5.10 (left). Note that $\Delta q_1q_2q_3$ and $\Delta q_7q_6q_5$ are reflections of each other with respect to the bisector of $\angle q_3q_4q_5$. Hence the shortest distance between $\Delta q_1q_2q_3$ and $\Delta q_7q_6q_5$ is $\min\{|q_3q_5|, |q_2q_6|, |q_1q_7|\}$. Since $c_* \geq 1$, we have

$$|q_1q_7| > |q_7q_9| = |q_3q_5| = a^{3/2} = \left(\frac{c_*}{2(c_*+1)}\right)^{3/2} \geq \frac{c_*}{2(c_*+1)^2}.$$

Further note that $q_2q_4q_6q_8$ is an isosceles trapezoid, so the length of its diagonal is bounded by $|q_2q_6| > |q_2q_4| = \frac{c_*}{2(c_*+1)^2}$. Therefore the claim holds when $p_k \in \Delta q_7q_6q_5$.

Otherwise $p_k \in g_3(P^m) \setminus g_5(P^m) = \Delta q_9 q_8 q_7$: see Figure 5.10 (right). Note that $\Delta q_1 q_2 q_3$ and $\Delta q_9 q_8 q_7$ are reflections of each other with respect to the bisector of $\angle q_4 q_5 q_6$. So the shortest distance between the shaded triangles is the minimum between $|q_3 q_7|$, $|q_2 q_8|$, and $|q_1 q_9|$. However, all three candidates are strictly larger than $|q_4 q_6| = \frac{c_*}{2(c_*+1)^2}$. This completes the proof of the claim. □

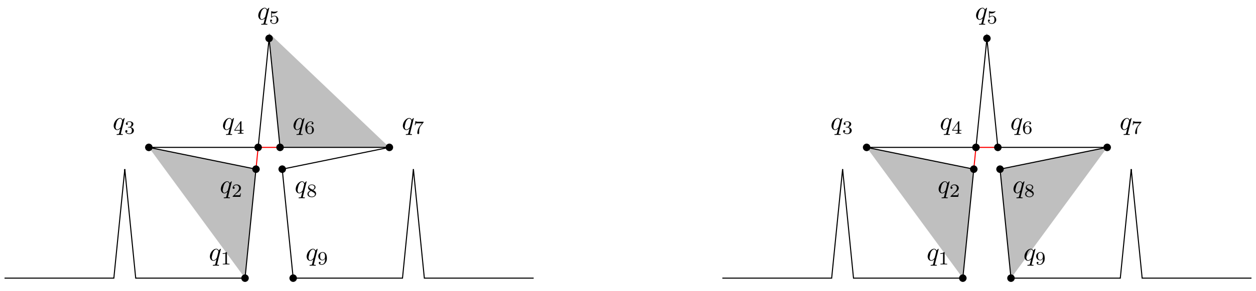


Figure 5.10: $p_i \in \Delta q_1 q_2 q_3$, Left: $p_k \in \Delta q_7 q_6 q_5$; Right: $p_k \in \Delta q_9 q_8 q_7$.

Now the diameter of $g_2(P^m) \cup g_3(P^m)$ is $a = \frac{c_*}{2(c_*+1)}$ (note that there are three diameter pairs), so

$$\frac{|p_i p_j| + |p_j p_k|}{|p_i p_k|} < \frac{2 \cdot \frac{c_*}{2(c_*+1)}}{\frac{c_*}{2(c_*+1)^2}} = 2c_* + 2 = c,$$

as required. This concludes the proof of Lemma 5.7 and Theorem 5.4. □

Remarks

1. For $m \geq 1$, let $P_*^m = g_2(P^m) \cup g_3(P^m)$, see Figure 5.11 (right). Observe that P_*^m is a c-chain with $n = 4^m/2 + 1$ vertices and stretch factor

$$\sqrt{c(c-2)/8}(n-1)^{\frac{1+\log(c-2)-\log c}{2}}.$$

   Since $\sqrt{c(c-2)/8} \geq 1$ for $c \geq 4$, this improves the result of Theorem 5.4 by a constant factor. Since this construction does not improve the exponent, and the analysis would be longer (requiring a case analysis without new insights), we omit the details.

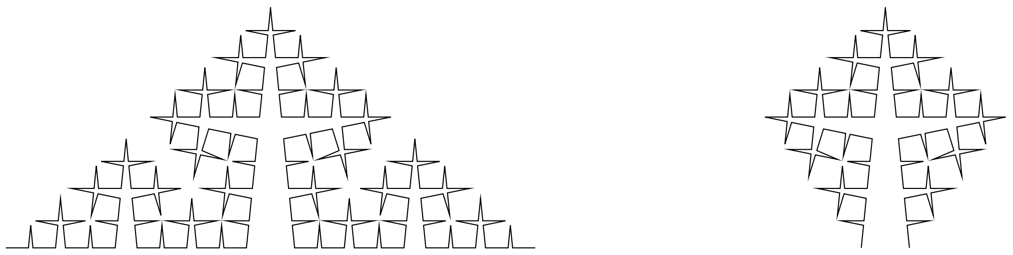


Figure 5.11: The chains P^4 (left) and P^4_* (right).

2. If c were used instead of $c_* = (c-2)/2$ in the lower bound construction, then the condition $c \geq 4$ in Theorem 5.4 could be replaced by $c \geq 1$, and the bound could be improved from
$$(n-1)^{\frac{1+\log(c-2)-\log c}{2}} \quad \text{to} \quad (n-1)^{\frac{1+\log c-\log(c+1)}{2}}.$$
Although we were unable to prove that the resulting P^m's, $m \in \mathbb{N}$, are c-chains, a computer program has verified that the first few generations of them are indeed c-chains.

3. The upper bounds in Theorem 5.1–5.3 are valid regardless of whether the chain is crossing or not. On the other hand, the lower bound in Theorem 5.4 is given by noncrossing chains. A natural question is whether a sharper upper bound holds if the chains are required to be noncrossing. Specifically, can the exponent of n in the upper bound be reduced to $1/2 - \varepsilon$, where $\varepsilon > 0$ depends on c?

5.4 Generalizations to Higher Dimensions

A c-chain P with n vertices and its stretch factor δ_P can be defined in any metric space, not just the Euclidean plane. We now discuss how our results generalize to other metric spaces, with a particular focus on the high-dimensional Euclidean space $\mathbb{R}^d$. First, we examine the upper bounds from Section 5.2.

As already noted in Section 5.2, the upper bound $\delta_P \leq c(n-1)^{\log c}$ of Theorem 5.1 holds for any positive distance function that need not even satisfy the triangle inequality.

Theorem 5.2 uses only the triangle inequality, and the bound $\delta_P \leq c(n-2)+1$ holds in any metric space. This bound cannot be improved, in the following sense: For every $c \geq 2+\sqrt{5}$ and even n, we can define a finite metric space on the vertex set of P by $|p_1p_n| = 1$; for $1 < i < n$,

$$|p_1p_i| = \begin{cases} \dfrac{c+1}{2} & \text{if } i \text{ is even} \\ \\ \dfrac{c-1}{2} & \text{if } i \text{ is odd} \end{cases}, \quad |p_ip_n| = \begin{cases} \dfrac{c-1}{2} & \text{if } i \text{ is even} \\ \\ \dfrac{c+1}{2} & \text{if } i \text{ is odd} \end{cases},$$

and $|p_ip_j| = c$ for all $1 < i < j < n$. It is easy to verify that P is a c-chain (the case that puts the strongest constraint on c in the c-chain property (5.1) occurs if, e.g., $i=1$, $1<j<n$ is even, and $j<k<n$ is odd) and that P has stretch factor

$$\delta_P = \frac{\sum_{i=1}^{n-1}|p_ip_{i+1}|}{|p_1p_n|} = |p_1p_2| + |p_{n-1}p_n| + \sum_{i=2}^{n-2}|p_ip_{i+1}| = c(n-2)+1.$$

The proof of Theorem 5.3 uses a volume argument in the plane. The argument[6] extends to $\mathbb{R}^d$, for all constant dimensions $d \geq 2$, and yields $\delta_P = O\left((n-1)^{(d-1)/d}\right)$.

Theorem 5.10. *For a c-chain P with n vertices in $\mathbb{R}^d$, for some constant $d \geq 2$, we have*

$$\delta_P = O\left((n-1)^{(d-1)/d}\right).$$

Lower bounds in $\mathbb{R}^d$ We show that the exponent $(d-1)/d$ in Theorem 5.10 cannot be improved. More precisely, for every $\varepsilon > 0$, we construct a family of axis-parallel chains in $\mathbb{R}^d$ whose stretch factor is $n^{(1-\varepsilon)(d-1)/d}$ for sufficiently large $n(\varepsilon)$. For the higher-dimensional case, we focus on axis-parallel chains, as they are easier to analyze. In the plane ($d=2$), this construction is also possible, but it yields weaker bounds than Theorem 5.4.

Theorem 5.11. *Let $d \geq 2$ be an integer. For all constants $\varepsilon > 0$ and sufficiently large*

[6]Theorem 5.10 is mainly the work of Csaba D. Tóth.

$c = \Omega(d)$, there is a positive integer n_0 such that for every $n \geq n_0$, there exists an axis-parallel c-chain in $\mathbb{R}^d$ with n vertices and stretch factor at least $(n-1)^{(1-\varepsilon)(d-1)/d}$.

Proof. Let $d \geq 2$, $\varepsilon > 0$, and $c = \Omega(d)$ be given. We describe a recursive construction in terms of an even integer parameter

$$r > 3^{(1-\varepsilon)/(d\varepsilon)}. \tag{5.6}$$

We recursively define a family $\mathcal{Q}_c = \{Q^m\}_{m\in\mathbb{N}}$ of axis-parallel c-chains in $\mathbb{R}^d$, where each chain Q^m has $n_m \leq 3^{m+1}r^{dm}$ vertices. Then, we show that the stretch factor of every Q^m is at least $(n_m - 1)^{(1-\varepsilon)(d-1)/d}$ for sufficiently large $m \in \mathbb{N}$.

Construction of $\mathcal{Q}_c$ For each chain in $\mathcal{Q}_c$, we maintain a subset of *active* directed edges, which are disjoint, have the same length, and are parallel to the same coordinate axis. In a nutshell, the recursion works as follows. We start with a chain Q^0 that consists of a single segment that is labeled active; then for $m = 1, 2, \ldots$, we obtain Q^m by replacing each active edge in a fixed chain π by a homothetic copy of Q^{m-1}. The chain π is defined below; it consists of $6r^d + 1$ edges, $3r^d$ of which are active.

We define the chain π in four steps, see Figure 5.12 for an illustration. Let $\mathbf{e_i}$, $i = 1, \ldots, d$, be the standard basis vectors in $\mathbb{R}^d$.

(1) Consider the $(d-1)$-dimensional hyperrectangle $A = [0,1] \times [0, r-1]^{d-2}$. Let γ_0 be an axis-parallel Hamiltonian cycle on the $2r^{d-2}$ integer points that lie in A such that the origin is incident to an edge parallel to the x_1-axis. We label the vertices of γ_0 by v_i, for $i = 1, \ldots, 2r^{d-2}$, in order, where v_1 is the origin.

(2) Let $a = (3r^2+1)/(3r) = r + 1/(3r)$, and consider the d-dimensional hyperrectangle $A \times [0,a] = [0,1] \times [0, r-1]^{d-2} \times [0,a]$. We construct a Hamiltonian cycle γ_1 on the $4r^{d-2}$ points in

$$\{v_i \times \{0,a\} \mid i = 1, \ldots, 2r^{d-2}\}$$

by replacing every edge (v_{2i-1}, v_{2i}) in γ_0 with three edges

$$((v_{2i-1}, 0), (v_{2i-1}, a)),\ ((v_{2i-1}, a), (v_{2i}, a)),\ \text{and } ((v_{2i}, a), (v_{2i}, 0)).$$

Note that γ_1 has $4r^{d-2}$ edges, such that $2r^{d-2}$ edges have length a and are parallel to the x_d-axis. Also note that the origin v_1 is incident to a unit edge parallel to the x_1-axis, and to an edge of length a parallel to the x_d-axis.

(3) Delete the edge of γ_1 that is incident to the origin v_1 and parallel to the x_1-axis. This turns γ_1 into a Hamiltonian chain γ_2 from the origin to the vertex $\mathbf{e}_1$ in the hyperrectangle $A \times [0, a] = [0, 1] \times [0, r-1]^{d-2} \times [0, a]$.

(4) Consider the hyperrectangle $B(\pi) = [0, 3r^2 + 1] \times [0, r-1]^{d-2} \times [0, a]$. Let π be the chain from the origin to $(3r^2 + 1) \cdot \mathbf{e}_1$ that is obtained by the concatenation of $3r^2/2$ copies of γ_2, translated by vectors $(2j-1) \cdot \mathbf{e}_1$ for $j = 1, 2, \ldots, 3r^2/2$, interlaced with $3r^2/2 + 1$ unit segments parallel to $\mathbf{e}_1$. Note that π has $(3r^2/2) \cdot \left(4r^{d-2} - 1\right) + 3r^2/2 + 1 = 6r^d + 1$ edges, $(3r^2/2) \cdot 2r^{d-2} = 3r^d$ of which have length a and are parallel to the x_d-axis. We label all these edges as active, so that π has $3r^d$ active edges. Observe that $B(\pi)$ is the minimum axis-parallel bounding box of π.

Lemma 5.12. *The chain π is a c'-chain for $c' = 8 + 2r\sqrt{d-1}$. Furthermore, if the points q_1, q_2, and q_3 are contained in active edges, in this order along π and not all in the same edge, then*

$$\frac{|q_1 q_2| + |q_2 q_3|}{|q_1 q_3|} \leq 8 + 2r\sqrt{d-1}.$$

Proof. We extend π to a chain π' by attaching a parallel copy of γ_2 to each end of π. We prove the lemma for π'. Then, the lemma also follows for π, as π is a subchain of π'. Write $\pi' = (p_1, \ldots, p_n)$. Since p_i, p_j, and p_k are endpoints of active edges, for any choice of $1 \leq i < j < k \leq n$, the second claim in the lemma implies that π' is a c'-chain.

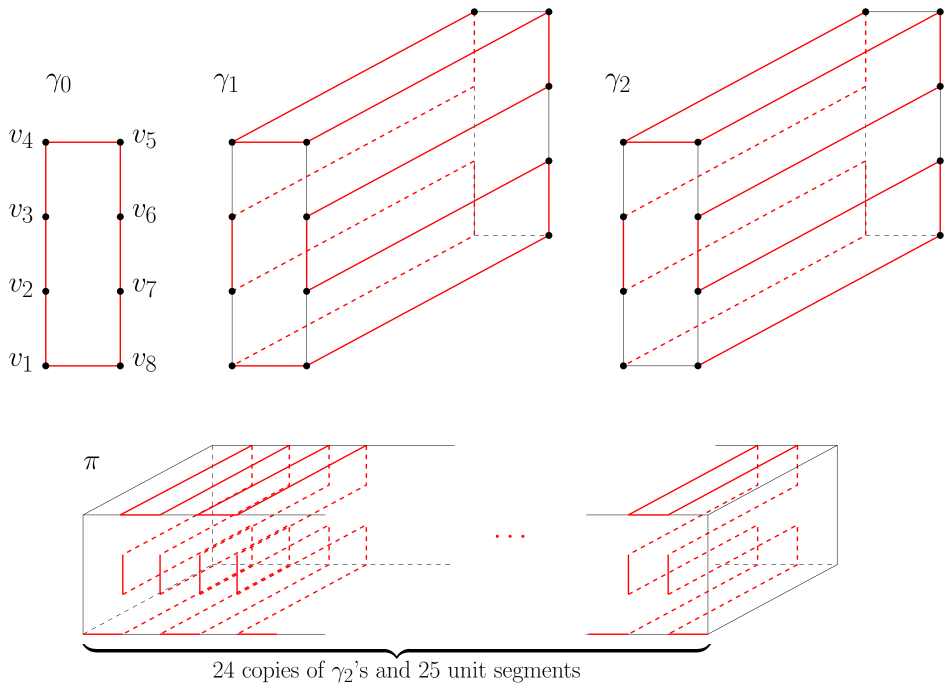


Figure 5.12: The cycles γ_0 (top left), γ_1 (top middle), and the chains γ_2 (top right), π (bottom) for $d = 3$ and $r = 4$. The cycles and chains are in red, their bounding boxes are outlined in black.

We give an upper bound for the ratio $(|q_1q_2|+|q_2q_3|)/|q_1q_3|$. Recall that all the active edges in π' come from the $3r^2/2+2$ translated copies of the chain γ_2; and γ_2 has vertices in an axis-aligned bounding box $B = [0,1] \times [0, r-1]^{d-2} \times [0, a]$. Denote by $B_0, B_1, \ldots, B_{3r^2/2}, B_{3r^2/2+1}$ the minimum axis-aligned bounding boxes of the $3r^2/2 + 2$ translates of γ_2 in π'. Suppose that q_1, q_2, and q_3 are in B_{i_1}, B_{i_2}, and B_{i_3}, respectively. By assumption, $i_1 \le i_2 \le i_3$.

If $i_1 = i_3$, then q_1, q_2, and q_3 are in B_{i_1}. Since q_1 and q_3 are not on the same active edge, and since γ_0 has integer coordinates, we have $|q_1q_3| \ge 1$. Consequently,

$$\begin{aligned}
\frac{|q_1q_2| + |q_2q_3|}{|q_1q_3|} &\le \frac{2 \cdot \operatorname{diam}(B_{i_1})}{1} \\
&\le 2\sqrt{1^2 + (d-2)(r-1)^2 + a^2} \\
&= 2\sqrt{1 + (d-2)(r-1)^2 + (r + 1/(3r))^2} \\
&\le 2\sqrt{2 + (d-1)r^2} \\
&< 2\sqrt{2} + 2r\sqrt{d-1}.
\end{aligned}$$

Otherwise $i_1 < i_3$, and the first coordinates of q_1 and q_3 differ by at least $2(i_3 - i_1) - 1 \geq i_3 - i_1$, hence $|q_1q_3| \geq i_3 - i_1$. In this case,

$$\begin{aligned}\frac{|q_1q_2| + |q_2q_3|}{|q_1q_3|} &\leq \frac{2 \cdot \mathrm{diam}(B_{i_1} \cup B_{i_3})}{i_3 - i_1} \\ &\leq \frac{2 \cdot \sqrt{(2(i_3 - i_1) + 1)^2 + (d-2)(r-1)^2 + a^2}}{i_3 - i_1} \\ &\leq \frac{4(i_3 - i_1) + 4 + 2r\sqrt{d-1}}{i_3 - i_1} \\ &\leq 8 + 2r\sqrt{d-1},\end{aligned}$$

as claimed. This completes the proof of Lemma 5.12. $\square$

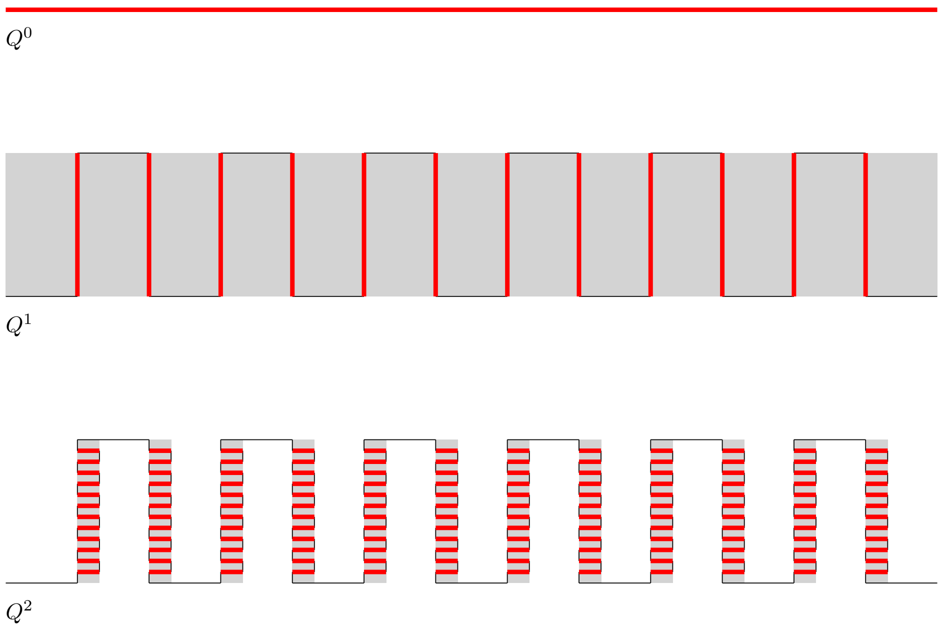


Figure 5.13: The chains Q^0 (top), Q^1 (middle), and Q_2 (bottom) for $d = r = 2$. The active edges are highlighted by red bold lines. The bounding box B of Q^1 and bounding boxes B' of homothetic copies of Q^1 in Q^2 are shaded.

Now the axis-parallel chains Q^m can be defined recursively (see Figure 5.13 for an illustration). Let Q^0 be a line segment of length $3r^2 + 1$, parallel to the x_1-axis, labeled active. Let Q^1 be π and let $B = B(\pi)$ be its minimum axis-parallel bounding box. Recall that $B = [0, 3r^2 + 1] \times [0, r-1]^{d-2} \times [0, a]$.

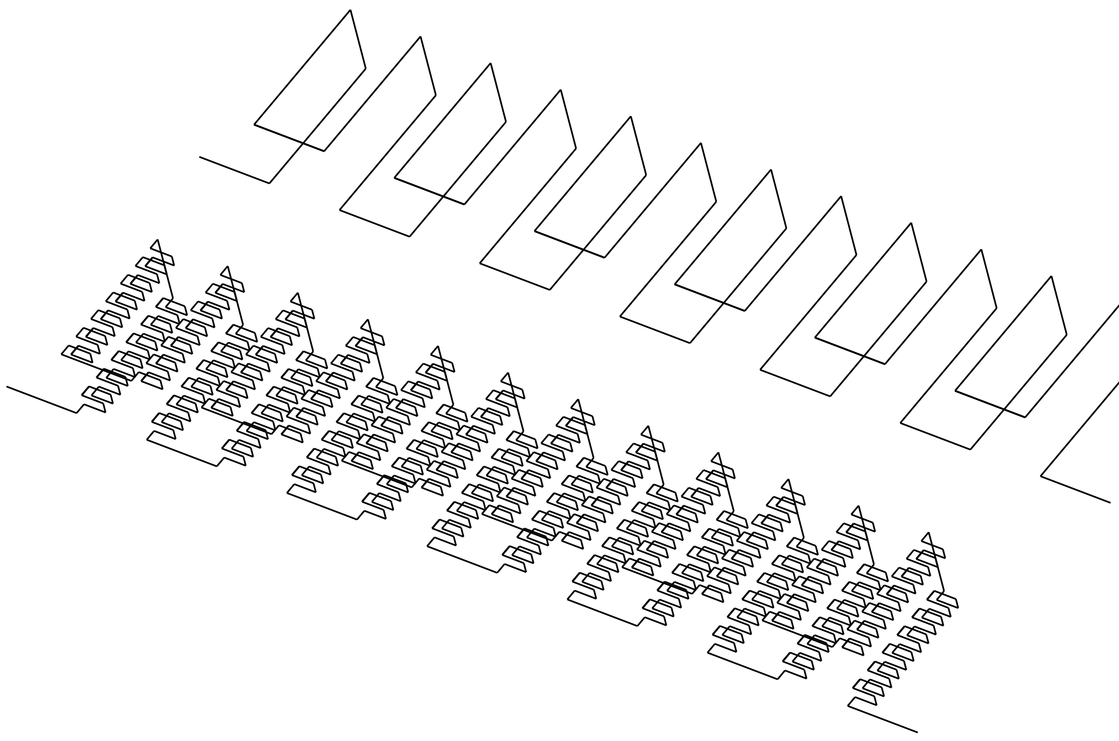


Figure 5.14: The chains Q^1 (top) and Q^2 (bottom) for $d = 3$ and $r = 2$.

We maintain the invariant that each chain Q^m ($m \in \mathbb{N}$) is contained in B. In order to do this, let B' be a hyperrectangle obtained from B by a rotation of 90 degrees in the $\langle \mathbf{e}_1, \mathbf{e}_d \rangle$ plane, and scaling by a factor of $a/(3r^2+1) = 1/(3r)$; i.e., $B' = [0, a/(3r)] \times [0, (r-1)/(3r)]^{d-2} \times [0, a]$. In particular, the longest edges of B' are parallel to the active edges in B, and they all have length a. Place a translate of B' along each active edge in Q^1 such that all such translates are contained in B. Note that the distance between any two translates is at least $1 - 2a/(3r) = 1/3 - 2/(9r^2) \geq 5/18$.

For all $m \geq 1$, we construct Q^{m+1} by replacing the active edges of Q^1 with a scaled (and rotated) copy of Q^m in each translate of B'; and we let the active edges of Q^{m+1} be the active edges in these new copies of Q^m.

Instead of keeping track of the total length of Q^m, we analyze the total length of the active edges of Q^m. In each iteration, the number of active edges increases by a factor of $3r^d$ and the length of an active edge decreases by a factor of $a/(3r^2+1) = 1/(3r)$. Overall the total length of active edges increases by a factor of r^{d-1}. It follows that for all $m \in \mathbb{N}$, the chain Q^m has $3^m r^{dm}$ active edges, and their total length is $(3r^2+1)\cdot r^{(d-1)m}$. Next we estimate the number of vertices in Q^m. Recall that the recursive construction replaces each active edge with $3r^d$ active edges and $3r^d+1$ inactive edges (which are never replaced). Consequently, for $m \geq 1$, the number of inactive edges in Q^m is $(3r^d+1)\sum_{i=0}^{m-1} 3^i r^{di}$, and the total number of vertices is

$$n_m = 1 + 3^m r^{dm} + (3r^d+1)\sum_{i=0}^{m-1} 3^i r^{di} = 1 + 3^m r^{dm} + (3r^d+1)\frac{3^m r^{dm}-1}{3r^d-1}.$$

Note that

$$3^m r^{dm} < n_m \leq 3 \cdot 3^m r^{dm}. \tag{5.7}$$

Since the distance between the two endpoints of Q^m remains $3r^2+1$, using (5.6) and (5.7), the stretch factor of Q^m is at least

$$\frac{|Q^m|}{3r^2+1} \geq r^{(d-1)m} \geq \left(\frac{n_m}{3^{m+1}}\right)^{\frac{d-1}{d}} \geq n_m^{(1-\varepsilon)\frac{d-1}{d}},$$

for sufficiently large m.

It remains to show that $\mathcal{Q}_c = \{Q^m : m \in \mathbb{N}\}$ is a family of c-chains, where $c = \Omega(d)$. We proceed by induction on m. The claim is trivial for $m = 0$, and it follows from Lemma 5.12 for $m = 1$.

Now, let $m \geq 2$. Write $Q^m = (p_1, \ldots, p_n)$, and let $1 \leq i < j < k \leq n$. We shall derive an upper bound for the ratio $(|p_i p_j| + |p_j p_k|)/|p_i p_k|$. Recall that Q^m is obtained by replacing each active edge of $Q^1 = \pi$ by a scaled copy of Q^{m-1}. If p_i and p_k are in the same copy of Q^{m-1}, then so is p_j and induction completes the proof.

Otherwise let B'_i, B'_j, and B'_k be the bounding boxes of the copies of Q^{m-1} that contain p_i, p_j, and p_k, respectively. Let a_i, a_j, and a_k be the active segments in Q^1 that are replaced by B'_i, B'_j, and B'_k; and let $q_i \in a_i$, $q_j \in a_j$, and $q_k \in a_k$ be the orthogonal projections of p_i,

p_j, and p_k onto a_i, a_j, and a_k, respectively. (If $i = 1$, then let $q_i = p_1$; if $k = n$, then let $q_k = p_n$. Since the proof of Lemma 5.12 works on the extended chain π', it applies to q_i, q_j, and q_k regardless of this special condition.)

Since each projection happens within a hyperplane orthogonal to the x_d-axis onto an active edge in a translated copy of $[0, a/(3r)] \times [0, (r-1)/(3r)]^{d-2} \times [0, a]$, we have that $|p_iq_i|$, $|p_jq_j|$, and $|p_kq_k|$ are each bounded above by

$$\sqrt{a^2/(3r)^2 + (d-2)(r-1)^2/(3r)^2} \leq \sqrt{d-1}/3 + 1/3r \leq \sqrt{d-1}/3 + 1/6.$$

As there are at least two distinct active edges among a_i, a_j, and a_k (and as the distance between p_1 or p_n and any active edge in π is at least 1), we have

$$|q_iq_j| + |q_jq_k| \geq \max\{|q_iq_j|, |q_jq_k|\} \geq 1.$$

Combining these two bounds with the triangle inequality, we get

$$\begin{aligned} |p_ip_j| + |p_jp_k| &\leq (|p_iq_i| + |q_iq_j| + |q_jp_j|) + (|p_jq_j| + |q_jq_k| + |q_kp_k|) \\ &\leq |q_iq_j| + |q_jq_k| + \frac{4}{3}\sqrt{d-1} + \frac{2}{3} \\ &\leq \left(\frac{5}{3} + \frac{4}{3}\sqrt{d-1}\right)(|q_iq_j| + |q_jq_k|). \end{aligned}$$

On the other hand, we have $|p_ip_k| \geq \frac{5}{18}|q_iq_k|$, as this lower bound holds for the projections of the edges to each coordinate axis. Now Lemma 5.12 yields

$$\begin{aligned} \frac{|p_ip_j| + |p_jp_k|}{|p_ip_k|} &\leq \frac{5/3 + 4\sqrt{d-1}/3}{5/18} \cdot \frac{|q_iq_j| + |q_jq_k|}{|q_iq_k|} \\ &\leq (6 + 24\sqrt{d-1}/5) \cdot (8 + 2r\sqrt{d-1}) \\ &= O(r(d-1)). \end{aligned}$$

This completes the proof of Theorem 5.11. □

5.5 Algorithm for Recognizing c-Chains[7]

In this section, we design a randomized Las Vegas algorithm to recognize c-chains in d-dimensional Euclidean space. More precisely, given a polygonal chain $P = (p_1, \ldots, p_n)$ in $\mathbb{R}^d$, and a parameter $c \geq 1$, the algorithm decides whether P is a c-chain, in $O\left(n^{3-1/d} \text{ polylog } n\right)$ expected time. By definition, $P = (p_1, \ldots, p_n)$ is a c-chain if $|p_ip_j| + |p_jp_k| \leq c\ |p_ip_k|$ for all $1 \leq i < j < k \leq n$; equivalently, p_j lies in the ellipsoid of major axis c with foci p_i and p_k. Consequently, it suffices to test, for every pair $1 \leq i < k \leq n$, whether the ellipsoid of major axis $c|p_ip_k|$ with foci p_i and p_k contains p_j, for all j, $i < j < k$. For this, we can apply recent results from geometric range searching.

Theorem 5.13. *For every integer $d \geq 2$, there are randomized algorithms that can decide, for a polygonal chain $P = (p_1, \ldots, p_n)$ in $\mathbb{R}^d$ and a threshold $c > 1$, whether P is a c-chain in $O\left(n^{3-1/d} \text{ polylog } n\right)$ expected time and $O(n \log n)$ space.*

Agarwal, Matoušek and Sharir [3, Theorem 1.4] constructed, for a set S of n points in $\mathbb{R}^d$, a data structure that can answer semi-algebraic range searching queries; in particular, it can report the number of points in S that are contained in a query ellipsoid. Specifically, they showed that, for every $d \geq 2$ and $\varepsilon > 0$, there is a constant B and a data structure with $O(n)$ space, $O\left(n^{1+\varepsilon}\right)$ expected preprocessing time, and $O\left(n^{1-1/d} \log^B n\right)$ query time. The construction was later simplified by Matoušek and Patáková [28]. Using this data structure, we can quickly decide whether a given polygonal chain is a c-chain.

The idea is to repeatedly subdivide the chain into two equal-sized subchains until single vertices are reached and construct an ellipsoid range searching data structure for each of the subchains obtained. Then for each pair of indices $1 \leq i < k \leq n$, we only need to query the data structures that correspond to disjoint maximal subchains that make up the chain $(p_i, \ldots, p_n)$.

In this decision algorithm only the construction of the data structures uses randomization,

[7]This section is mainly the work of Wolfgang Mulzer and Csaba D. Tóth.

which is independent of the value of c. The parameter c is used for defining the ellipses $E_{i,k}$, and the queries to the data structures; this part is deterministic. Hence, we can find the optimal value of c by Meggido's parametric search [29] in the second part of the algorithm.

Meggido's technique reduces an optimization problem to a corresponding decision problem at a polylogarithmic factor increase in the running time. An optimization problem is amenable to this technique if the following three conditions are met [35]: (1) the objective function is monotone in the given parameter; (2) the decision problem can be solved by evaluating bounded-degree polynomials, and (3) the decision problem admits an efficient parallel algorithm (with polylogarithmic running time using a polynomial number of processors). All three conditions hold in our case: The area of each ellipse with foci in S monotonically increases with c; the data structure of [28] answers ellipse range counting queries by evaluating polynomials of bounded degree; and the $\binom{n}{2}$ queries can be performed in parallel. Alternatively, Chan's randomized optimization technique [12] is also applicable. Both techniques yield the following result.

Corollary 5.14. *There are randomized algorithms that can find, for a polygonal chain $P = (p_1, \ldots, p_n)$ in $\mathbb{R}^d$, the minimum $c \geq 1$ for which P is a c-chain in $O\left(n^{3-1/d} \operatorname{polylog} n\right)$ expected time and $O(n \log n)$ space.*

We note that, for $c = 1$, the test takes $O(n)$ time: it suffices to check whether points $p_3, \ldots, p_n$ lie on the line spanned by p_1p_2, in that order.

Remark Recently, Agarwal et al. [1, Theorem 13] designed a data structure for semi-algebraic range searching queries that supports $O(\log n)$ query time, at the expense of higher space and preprocessing time. The size and preprocessing time depend on the number of free parameters that describe the semi-algebraic set. An ellipsoid in $\mathbb{R}^d$ is defined by $2d + 1$ parameters: the coordinates of its foci and the length of its major axis. Specifically, they showed that, for every $d \geq 2$ and $\varepsilon > 0$, there is a data structure with $O(n^{2d+1+\varepsilon})$ space and $O(n^{2d+1+\varepsilon})$ expected preprocessing time that can report the number of points in S

contained in a query ellipsoid in $O(\log n)$ time. This data structure allows for a tradeoff between preprocessing time and overall query time in the algorithm above. However the resulting tradeoff does not seem to yield an improvement over the expected running time in Theorem 5.13 for any $d \geq 2$.

Bibliography

[1] Pankaj K. Agarwal, Boris Aronov, Esther Ezra, and Joshua Zahl. An efficient algorithm for generalized polynomial partitioning and its applications. In Gill Barequet and Yusu Wang, editors, *35th International Symposium on Computational Geometry, SoCG 2019, June 18-21, 2019, Portland, Oregon, USA*, volume 129 of *LIPIcs*, pages 5:1–5:14. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2019.

[2] Pankaj K. Agarwal, Rolf Klein, Christian Knauer, Stefan Langerman, Pat Morin, Micha Sharir, and Michael A. Soss. Computing the detour and spanning ratio of paths, trees, and cycles in 2D and 3D. *Discrete & Computational Geometry*, 39(1-3):17–37, 2008.

[3] Pankaj K. Agarwal, Jiří Matoušek, and Micha Sharir. On range searching with semialgebraic sets. II. *SIAM J. Computing*, 42(6):2039–2062, 2013.

[4] Oswin Aichholzer, Franz Aurenhammer, Christian Icking, Rolf Klein, Elmar Langetepe, and Günter Rote. Generalized self-approaching curves. *Discrete Applied Mathematics*, 109(1-2):3–24, 2001.

[5] Soroush Alamdari, Timothy M. Chan, Elyot Grant, Anna Lubiw, and Vinayak Pathak. Self-approaching graphs. In Walter Didimo and Maurizio Patrignani, editors, *Proc. 20th Symposium on Graph Drawing (GD)*, volume 7704 of *LNCS*, pages 260–271, Berlin, 2012. Springer.

[6] Sanjeev Arora, László Lovász, Ilan Newman, Yuval Rabani, Yuri Rabinovich, and Santosh Vempala. Local versus global properties of metric spaces. *SIAM J. Computing*, 41(1):250–271, 2012.

[7] Prosenjit Bose, Joachim Gudmundsson, and Michiel H. M. Smid. Constructing plane spanners of bounded degree and low weight. *Algorithmica*, 42(3-4):249–264, 2005.

[8] Prosenjit Bose, Irina Kostitsyna, and Stefan Langerman. Self-approaching paths in simple polygons. *Computational Geometry: Theory and Applications*, 87:101595, 2020.

[9] Prosenjit Bose and Michiel H. M. Smid. On plane geometric spanners: A survey and open problems. *Computational Geometry: Theory and Applications*, 46(7):818–830, 2013.

[10] Peter Brass, William O. J. Moser, and János Pach. *Research Problems in Discrete Geometry*. Springer, New York, 2005.

[11] Ernesto Cesàro. Remarques sur la courbe de von Koch. *Atti della R. Accad. della Scienze fisiche e matem. Napoli*, 12(15), 1905. Reprinted as §228 in *Opere scelte, a cura dell'Unione matematica italiana e col contributo del Consiglio nazionale delle ricerche*, Vol. 2: Geometria, analisi, fisica matematica, Rome, dizioni Cremonese, pp. 464–479, 1964.

[12] Timothy M. Chan. Geometric applications of a randomized optimization technique. *Discrete & Computational Geometry*, 22(4):547–567, 1999.

[13] Otfried Cheong, Herman J. Haverkort, and Mira Lee. Computing a minimum-dilation spanning tree is NP-hard. *Computational Geometry: Theory and Applications*, 41(3):188–205, 2008.

[14] Fan R. K. Chung and Ron L. Graham. On Steiner trees for bounded point sets. *Geometriae Dedicata*, 11(3):353–361, 1981.

[15] Hallard T. Croft, Kenneth J. Falconer, and Richard K. Guy. *Unsolved Problems in Geometry*, volume 2 of *Unsolved Problems in Intuitive Mathematics*. Springer, New York, 1991.

[16] Gautam Das and Deborah Joseph. Which triangulations approximate the complete graph? In Hristo Djidjev, editor, *Proc. International Symposium on Optimal Algorithms*, volume 401 of *LNCS*, pages 168–192, Berlin, 1989. Springer.

[17] Adrian Dumitrescu and Minghui Jiang. Minimum rectilinear Steiner tree of n points in the unit square. *Computational Geometry: Theory and Applications*, 68:253–261, 2018.

[18] David Eppstein. Spanning trees and spanners. In Jörg-Rüdiger Sack and Jorge Urrutia, editors, *Handbook of Computational Geometry*, chapter 9, pages 425–461. Elsevier, Amsterdam, 2000.

[19] David Eppstein. Beta-skeletons have unbounded dilation. *Computational Geometry: Theory and Applications*, 23(1):43–52, 2002.

[20] László Fejes Tóth. Über einen geometrischen Satz. *Mathematische Zeitschrift*, 46:83–85, 1940.

[21] Leonard Few. The shortest path and the shortest road through n points. *Mathematika*, 2(2):141–144, 1955.

[22] Edgar N. Gilbert and Henry O. Pollak. Steiner minimal trees. *SIAM Journal on Applied Mathematics*, 16(1):1–29, 1968.

[23] Christian Icking, Rolf Klein, and Elmar Langetepe. Self-approaching curves. *Mathematical Proceedings of the Cambridge Philosophical Society*, 125(3):441–453, 1999.

[24] Howard J. Karloff. How long can a Euclidean traveling salesman tour be? *SIAM Journal on Discrete Mathematics*, 2(1):91–99, 1989.

[25] Rolf Klein, Christian Knauer, Giri Narasimhan, and Michiel H. M. Smid. On the dilation spectrum of paths, cycles, and trees. *Computational Geometry: Theory and Applications*, 42(9):923–933, 2009.

[26] David G. Larman and Peter McMullen. Arcs with increasing chords. *Mathematical Proceedings of the Cambridge Philosophical Society*, 72(2):205–207, 1972.

[27] Jiří Matoušek. *Lectures on Discrete Geometry*, volume 212 of *Graduate Texts in Mathematics*. Springer-Verlag, New York, 2002.

[28] Jiří Matoušek and Zuzana Patáková. Multilevel polynomial partitions and simplified range searching. *Discrete & Computational Geometry*, 54(1):22–41, 2015.

[29] Nimrod Megiddo. Linear-time algorithms for linear programming in $\mathbb{R}^3$ and related problems. *SIAM J. Computing*, 12(4):759–776, 1983.

[30] Joseph S. B. Mitchell and Wolfgang Mulzer. Proximity algorithms. In Jacob E. Goodman, Joseph O'Rourke, and Csaba D. Tóth, editors, *Handbook of Discrete and Computational Geometry*, chapter 32, pages 849–874. CRC Press, Boca Raton, 3rd edition, 2017.

[31] Giri Narasimhan and Michiel H. M. Smid. Approximating the stretch factor of Euclidean graphs. *SIAM J. Comput.*, 30(3):978–989, 2000.

[32] Giri Narasimhan and Michiel H. M. Smid. *Geometric Spanner Networks*. Cambridge University Press, 2007.

[33] Martin Nöllenburg, Roman Prutkin, and Ignaz Rutter. On self-approaching and increasing-chord drawings of 3-connected planar graphs. *Journal of Computational Geometry*, 7(1):47–69, 2016.

[34] Günter Rote. Curves with increasing chords. *Mathematical Proceedings of the Cambridge Philosophical Society*, 115(1):1–12, 1994.

[35] Jeffrey S. Salowe. Parametric search. In Jacob E. Goodman and Joseph O'Rourke, editors, *Handbook of Discrete and Computational Geometry*, chapter 43, pages 969–982. CRC Press, Boca Raton, 2nd edition, 2004.

[36] Samuel Verblunsky. On the shortest path through a number of points. *Proceedings of the American Mathematical Society*, 2:904–913, 1951.

Chapter 6

Wegner's Inequality for Axis-Parallel Rectangles

6.1 Introduction

Given a collection of sets E, a *piercing set* is a set of elements from $\cup_{F\in E}F$ intersecting every set in E. The *piercing number* of E is the minimal size of a piercing set. Given a hypergraph $H=(X,E)$, a *cover* of H is a set $C\subseteq X$ such that every edge of H contains a point in C, namely, for every $e\in E$ we have $e\cap C\neq\emptyset$. As such, a cover is precisely a piercing set of E. The *piercing number* $\tau(H)$ of a hypergraph $H=(X,E)$ is the piercing number of its edge set E. It is sometimes also called the *covering number* or *stabbing number* of the hypergraph.

Given integers $p\geq q>1$, a family $\mathcal{F}$ of sets is said to satisfy the (p,q)*-property* if among every p sets in $\mathcal{F}$ there exist q sets with a non-empty intersection. The *independence number* or *matching number* of $\mathcal{F}$, namely the maximum number of pairwise disjoint sets in $\mathcal{F}$, is denoted by $\alpha(\mathcal{F})$ or $\nu(\mathcal{F})$. Clearly, $\nu(\mathcal{F})\leq\tau(\mathcal{F})$. If $\nu(\mathcal{F})=1$ then we say that $\mathcal{F}$ is an *intersecting* family.

In the above terminology, Helly's theorem [22] says that if a family $\mathcal{F}$ of convex sets in $\mathbb{R}^d$ satisfies the $(d+1,d+1)$-property then $\tau(\mathcal{F})=1$. Finding the piercing number of families of sets in $\mathbb{R}^d$ satisfying the (p, q)-property has been known in the literature as the (p, q)-*problem.* In particular, a collection of pairwise-intersecting intervals (i.e., an intersecting family of intervals) must have a point that belongs to all the intervals.

Hadwiger and Debrunner [20, 21] conjectured in 1957 that the (p, q)-property in a family $\mathcal{F}$ of convex sets in $\mathbb{R}^d$ implies that $\tau(\mathcal{F})$ is bounded by a constant depending on d, p, and q. They proved this under the condition that $(d-1)p < d(q-1)$ in the following stronger form:

Theorem 6.1 (Hadwiger–Debrunner [20])**.** *Let $\mathcal{F}$ be a finite family of convex sets in $\mathbb{R}^d$ satisfying the (p, q)-property for $p \geq q > 1$. If $(d-1)p < d(q-1)$ then $\tau(\mathcal{F}) \leq p - q + 1$.*

In 1992 Alon and Kleitman [3] resolved the Hadwiger–Debrunner conjecture, proving that in a family of convex sets in $\mathbb{R}^d$ that satisfies the (p, q)-property, the piercing number is bounded by a constant:

Theorem 6.2 (Alon–Kleitman [3])**.** *Let $p \geq q \geq d+1$ be integers. Then there exists a constant $c = c(d; p, q)$ depending only on d, p, q, such that if a family $\mathcal{F}$ of convex sets in $\mathbb{R}^d$ satisfies the (p, q)-property then $\tau(\mathcal{F}) \leq c$.*

In many cases the upper bounds on the piercing number improve significantly if we deal with families of "nice" sets. One such example is a result by Danzer, who proved:

Theorem 6.3 (Danzer [10])**.** *If a family of disks in $\mathbb{R}^2$ satisfies the $(2, 2)$-property, then $\tau(\mathcal{F}) \leq 4$.*

In this chapter we restrict our attention to axis-parallel *hyper-rectangles* (or *boxes*) in $\mathbb{R}^d$. The following inequality [16, Ineq. (3.4), p. 355] applies to any such family $\mathcal{F}$ that satisfies the (p, q)-property:

$$\tau(\mathcal{F}) \leq \binom{p-q+d}{d}, \quad p \geq q \geq 2. \tag{6.1}$$

Many have examined the case $q = 2$. The main unsettled question here is whether $\tau(\mathcal{F}) = O(\nu(\mathcal{F}))$. The following is a long-standing conjecture in dimension 2:

Conjecture 6.4 (Wegner [32], Gýarfás–Lehel [19])**.** *If a family $\mathcal{F}$ of axis-parallel rectangles in $\mathbb{R}^2$ satisfies the $(p,2)$-property, then $\tau(\mathcal{F}) \leq 2p-3$.*

The $(p,2)$-property can be rephrased as a family $\mathcal{F}$ of axis-parallel rectangles in $\mathbb{R}^2$ with $\nu(\mathcal{F}) = p-1$. As such, Conjecture 6.4 can be formulated as follows:

Conjecture 6.5. *If $\mathcal{F}$ is a family of axis-parallel rectangles in $\mathbb{R}^2$, then $\tau(\mathcal{F}) \leq 2\nu(\mathcal{F})-1$.*

Károlyi [26] proved that if $\mathcal{F}$ is a family of axis-parallel boxes in $\mathbb{R}^d$, then

$$\tau(\mathcal{F}) \leq \nu(\mathcal{F})(1+\log(\nu(\mathcal{F})))^{d-1}. \tag{6.2}$$

For the planar case, Eckhoff [16] gives the following upper-bound inequality based on a recurrence relation found independently by Wegner [33] and by Fon-Der-Flaass and Kostochka [17].

$$\tau(\mathcal{F}) \leq (\nu(\mathcal{F})+1)\left\lceil \log(\nu(\mathcal{F})+1)\right\rceil - 2^{\lceil \log(\nu(\mathcal{F})+1)\rceil} + 1. \tag{6.3}$$

After about 25 years, Correa et al. [9] improved Károlyi's bound for the plane by combining results of [4] and [6].

$$\tau(\mathcal{F}) = O\left(\nu(\mathcal{F}) \cdot (\log\log \nu(\mathcal{F}))^2\right). \tag{6.4}$$

From the other direction, Jelínek found an elegant construction with $\tau(\mathcal{F}) = 2\nu(\mathcal{F})-4$, for every $\nu \geq 4$ [9], and thereby showed that the factor 2 in Wegner's conjecture cannot be improved. On the other hand, one may note that this bound is not competitive for small ν, e.g., $\nu = 4, 5$. Our Theorem 6.6 below is relevant in this case.

For the special case of squares, better bounds are in effect. It is known that $\tau(\mathcal{F}) \leq 4\nu(\mathcal{F})$ for families of squares and $\tau(\mathcal{F}) \leq 2\nu(\mathcal{F})-1$ for families of unit squares [1, 12, 13]. The current state of the art for the ratio $\limsup \tau(\mathcal{F})/\nu(\mathcal{F})$ depending on the rectangle-type in the family is summarized in Table 6.1.

	Rectangles	Squares	Unit squares
Upper bound	$O\left((\log\log\nu)^2\right)$	4	2
Lower bound	2	3/2	3/2

Table 6.1: Bounds on the ratio $\limsup \tau(\mathcal{F})/\nu(\mathcal{F})$.

The function $f(n)$. In order to study the dependence between $\tau(\mathcal{F})$ and $\nu(\mathcal{F})$ it is convenient to define an integer function. As in [17], define $f(n)$ as the minimum integer such that every family $\mathcal{F}$ of axis-parallel rectangles with $\nu(\mathcal{F}) \le n$ can be pierced by $f(n)$ points. (Alternatively, the condition $\nu(\mathcal{F}) \le n$ can be replaced by $\nu(\mathcal{F}) = n$ in the definition.) A line sweep argument found independently by Wegner [33] and Fon-Der-Flaass and Kostochka [17] (mentioned previously; see also [16, Ineq. (3.6), p. 356]) yields the following recurrence:

$$f(n) \le f\left(\left\lfloor \frac{n-1}{2} \right\rfloor\right) + f\left(\left\lceil \frac{n-1}{2} \right\rceil\right) + n. \tag{6.5}$$

Taking into account that $f(0) = 0$ and $f(1) = 1$, one immediately obtains

$$\begin{aligned}
f(2) &\le f(0) + f(1) + 2 = 3,\\
f(3) &\le f(1) + f(1) + 3 = 5,\\
f(4) &\le f(1) + f(2) + 4 \le 8,\\
f(5) &\le f(2) + f(2) + 5 \le 11,\\
&\vdots
\end{aligned}$$

The general solution to the recurrence (6.5) (equivalent to (6.3)) is:

$$f(n) \le (n+1)\lceil \log(n+1) \rceil - 2^{\lceil \log(n+1) \rceil} + 1. \tag{6.6}$$

Lower bound constructions yield $f(1) = 1$, $f(2) = 3$, and $f(3) = 5$, and these are the only exact values known [16]. For small n, the resulting bounds are recorded in Table 6.2.

n	2	3	4	5
Upper bound on $f(n)$	3	5	8	11
Lower bound on $f(n)$	3	5	7(∗)	8(∗)

Table 6.2: Bounds on $f(n)$ for small n. The starred entries are proved in this chapter.

Our results. (i) Our main result—the first starred entry in Table 6.2—is Theorem 6.6 below (its proof appears in Section 6.4). It gives $f(4) \geq 7$; recall that $f(4) \leq 8$ is known. (ii) A lower bound on the ratio $\tau(\mathcal{F})/\nu(\mathcal{F})$ in higher dimensions is given by Theorem 6.7 in Section 6.2 (its proof appears in Section 6.5). Both results rely on the connection between piercing numbers for families of axis-parallel boxes in $\mathbb{R}^d$ and the Maximum Empty Box problem in $[0,1]^d$, introduced in Section 6.2.

Theorem 6.6. *There exists a finite family $\mathcal{S}$ of axis-parallel rectangles with $\nu(\mathcal{S}) = 4$ and $\tau(\mathcal{S}) = 7$. That is, $f(4) \geq 7$.*

Related work. Among the many variants of the (p,q)-property and Helly's theorem in particular, we can only mention a few here. Danzer and Grünbaum [11] investigated the following problem: if d and n are positive integers, what is the smallest $h = h(d,n)$ such that a family of boxes in $\mathbb{R}^d$ is n-pierceable if each of its h-member subfamilies is n-pierceable? The showed that $h(d,n)$ is infinite for all (d,n) with $d \geq 2$ and $n \geq 3$ except for $(d,n) = (2,3)$ when it is 16.

Larman et al. [29] showed that any collection of n axis-parallel rectangles contains $\sqrt{n/\log n}$ of them which are pairwise intersecting or pairwise disjoint; on the other hand, there are trivial examples with at most $\sqrt{n}$ in each of the two classes. If the conjecture $\tau(\mathcal{F}) = O(\nu(\mathcal{F}))$ were true, then the lower bound $\sqrt{n/\log n}$ could be improved to $\Omega(\sqrt{n})$; see also [5, p. 410].

Hadwiger had asked whether any collection of closed convex sets where every four have a triple that has a nonempty intersection (i.e., has at least one point in common) can be pierced by two points. Danzer exhibited six congruent triangles in the plane that can only be pierced by three points. The current best bound on the piercing number for such a family with the $(4, 3)$-property is 13; this bound is due to Kleitman et al. [28]. As such, the current gap for this problem is between 3 and 13.

Károlyi and Tardos [27] studied transversal numbers of hypergraphs related to multiple intervals and axis-parallel rectangles. Kaiser and Rabinovich [24] formulated a multicomponent generalization of Helly's theorem to convex (n, d)-bodies. Karasev [25] considered the problem of piercing families of convex sets in $\mathbb{R}^d$ such that every d or fewer sets in the family have a common point. Chan and Har-Peled [7] proved that for every family $\mathcal{F}$ of axis-parallel rectangles in $\mathbb{R}^2$ in which for every two intersecting rectangles, one of them contains a corner of the other, we have $\tau(\mathcal{F}) = O(\nu(\mathcal{F}))$.

Aronov, Ezra, and Sharir [4] have studied the of size of ε-nets for axis-parallel rectangles and boxes. Chalermsook and Chuzhoy [6] gave a $O(\log \log n)$-approximation algorithm for the problem of computing a Maximum Independent Set of Rectangles (MISR). Correa et al. [9] have used the above-mentioned results in combination. Besides combinatorial results, they have obtained several approximation algorithms for piercing various classes of rectangles, e.g., diagonal-pierced rectangles.

Govindarajan and Nivasch [18] studied a strengthening of the (p, q)-property by requiring that, among every p members of $\mathcal{S}$, at least q meet at a point of X, where X is a fixed convex curve in the plane; they showed that the piercing number can be substantially reduced in that case. Chudnovsky, Spirkl, and Zerbib [8] showed that if for each two intersecting boxes in a family $\mathcal{F}$ of boxes in $\mathbb{R}^d$, a corner of one is contained in the other, then $\mathcal{F}$ can be pierced by at most $O(k \log \log k)$ points, where $k = \nu(\mathcal{F})$, and in the special case where $\mathcal{F}$ contains only cubes this bound improves to $O(k)$.

Su and Zerbib [31] recently showed that results on piercing numbers have a natural interpretation in voting theory. For a survey of piercing in the context of geometric transversals, the reader is referred to the survey article [23].

6.2 Setup

In this section, we demonstrate an idea of constructing lower bound examples (i.e., families of axis-parallel rectangles) that support Wegner's Conjecture 6.5. By applying this idea, examples with $\tau(\mathcal{F}) = 2\nu(\mathcal{F}) - 1$ for $\nu(\mathcal{F}) = 2, 3, 4$ are obtained respectively. The last result in this sequence proves Theorem 6.6 (in Section 6.4). Whereas examples for the previous two ratios were previously known, we include ours to help the reader understand the construction better; as it illustrates the main ideas at a smaller scale.

6.2.1 Maximum Empty Box

A *box* in $\mathbb{R}^d$, $d \geq 2$, is a closed axis-parallel hyperrectangle $[a_1, b_1] \times \cdots \times [a_d, b_d]$ with $a_i \leq b_i$ for $1 \leq i \leq d$. Given a set S of n points in the unit cube $U_d = [0,1]^d$, a box $B \subset U_d$ is *empty* if it contains no points of S in its interior. Let $A_d(S)$ be the maximum volume of an empty box contained in U_d, and let $A_d(n)$ be the minimum value of $A_d(S)$ over all sets S of n points in U_d.

For a fixed d, it is known [30] that $A_d(n)$ is of the order $\Theta(\frac{1}{n})$. The following upper bound holds for any $d \geq 2$:

$$A_d(n) < \left(2^{d-1} \prod_{i=1}^{d-1} p_i \right) \cdot \frac{1}{n}, \tag{6.7}$$

where p_i is the ith prime, as shown in [14, 30]. In particular, $A_2(n) < \frac{4}{n}$.

A sharper upper bound has been recently obtained for larger d. The current best upper and lower bounds on $A_d(n)$, for $d \geq 54$, are as follows:

$$\frac{\log d}{4(n + \log d)} \leq A_d(n) \leq \frac{2^{7d+1}}{n}. \tag{6.8}$$

The lower bound is due to Aistleitner, Hinrichs, and Rudolf [2], and the upper bound is due to Larcher [2, Section 4]. For $d = 2$ the current best bounds (see [14, 15]) are

$$\big(5A_2(4) - o(1)\big) \cdot \frac{1}{n} = \big(1.25 - o(1)\big) \cdot \frac{1}{n} \leq A_2(n) < \frac{4}{n}. \tag{6.9}$$

Following Aistleitner et al. [2], define

$$c_d = \liminf_{n \to \infty} n \cdot A_d(n). \tag{6.10}$$

Taking (6.8) into account, we have

$$\frac{\log d}{4} \leq c_d \leq 2^{7d+1}, \text{ for } d \geq 2. \tag{6.11}$$

6.2.2 Discretization and connection with Maximum Empty Box

A long-standing open question—appearing in Eckhoff's survey [16, p. 359]—is whether $\tau = O(\nu)$ for systems of axis-parallel boxes in a fixed dimension d. Whereas we cannot answer this question, here we show that the ratio τ/ν must grow with the dimension d and further elaborate on the rate of this growth. It follows from (6.1) that $\tau(\mathcal{F}) \leq d + 1$, for any family of boxes in $\mathbb{R}^d$ having the $(3, 2)$-property. In particular, $\tau(\mathcal{R}) \leq d + 1$, for systems of axis-parallel boxes in $\mathbb{R}^d$ with $\nu(\mathcal{R}) = 2$. On the other hand, we have

$$\tau(\mathcal{R})/\nu(\mathcal{R}) = \Omega(\sqrt{d}/\log d) \tag{6.12}$$

for systems of axis-parallel boxes in $\mathbb{R}^d$ with $\nu(\mathcal{R}) = 2$ (this can be derived from a classical result of Erdős on k-chromatic triangle-free graphs, see [16, 17]). By taking multiple copies of this construction, it follows that there exist families of axis-parallel boxes in $\mathbb{R}^d$ with any given ν for which (6.12) holds.

One may wonder if there is any relation between (6.11) and (6.12). Observe that the large gap in (6.11) for the key parameter c_d leaves plenty of room for improvement. We next show that if $c_d = \omega(\sqrt{d}/\log d)$ were to hold, then one would obtain systems of axis-parallel boxes in $\mathbb{R}^d$ with $\tau(\mathcal{R})/\nu(\mathcal{R}) = \omega(\sqrt{d}/\log d)$, and thereby improve the lower bound in (6.12). The following result can be derived from Lemma 6.10 in combination with the aforementioned result of Aistleitner et al. [2].

Theorem 6.7. *For every $d \geq 1024$, there exists a system of axis-parallel boxes in $\mathbb{R}^d$ where $\tau(\mathcal{R})/\nu(\mathcal{R}) \geq c_d/2$.*

In particular, by the current state of the art, we have $\tau(\mathcal{R})/\nu(\mathcal{R}) = \Omega(\log d)$, a bound that grows with d but is inferior to the bound in (6.12).

We start with a technical lemma that provides a discretization mechanism for extracting a *finite* family of hyper-rectangles from an *infinite* family. For a finite point set $P \subset U_d$, $a, \delta > 0$, where $A_d(P) \geq a + 2\delta$, $a + 2\delta < 1$, and $1/\delta \in \mathbb{N}$, let $\mathcal{R}'(P, a, \delta; d)$ denote the *infinite* family of axis-parallel empty boxes of volume at least $a + 2\delta$ in U_d. Observe that if $P \subset P'$, then $\mathcal{R}'(P', a, \delta; d) \subset \mathcal{R}'(P, a, \delta; d)$.

Lemma 6.8. *For $P \subset U_d$, $a, \delta > 0$, where $A_d(P) \geq a + 2\delta$, $a + 2\delta < 1$, and $1/\delta \in \mathbb{N}$, there exists a finite family of axis-parallel empty boxes in U_d, denoted by $\mathcal{R}(P, a, \delta; d)$, so that:*

(i) *for each box $r \in \mathcal{R}(P, a, \delta; d)$ we have $r \cap P = \emptyset$ and $\mathrm{Vol}(r) = a + \delta$,*

(ii) *for every $r' \in \mathcal{R}'(P, a, \delta; d)$, there exists $r \in \mathcal{R}(P, a, \delta; d)$, with $r \subseteq r'$.*

(In particular, every box in $\mathcal{R}(P, a, \delta; d)$ has no points of P on its boundary.)

Proof. Let $j = 4d/\delta + 1$, and consider the $j \times \cdots \times j$ d-dimensional grid contained in U_d:

$$x_i = \frac{0}{j-1}, \frac{1}{j-1}, \ldots, 1, \text{ for } i = 1, \ldots, d.$$

Let $\mathcal{R}_1$ be the set of non-degenerate boxes determined by this grid. Note that $\mathcal{R}_1$ is a *finite* set whose cardinality is $|\mathcal{R}_1| = \binom{j}{2}^d$. Let $\mathcal{R}_2 \subset \mathcal{R}_1$ be the subset of grid boxes with volume at least $a + 1.5\delta$. Let $\mathcal{R}_3$ be the set of concentric scaled down homothetic copies of boxes in $\mathcal{R}_2$ of volume exactly $a + \delta$. Observe that $\mathcal{R}_1$, $\mathcal{R}_2$, and $\mathcal{R}_3$ are independent of P. Set $\mathcal{R}(P, a, \delta; d) = \{r \in \mathcal{R}_3 \mid r \cap P = \emptyset\}$. It is not clear a priori, whether this set is non-empty; we argue below that it is.

Consider any hyper-rectangle $r' \in \mathcal{R}'(P, a, \delta; d)$; assume that $r' = [s_1, t_1] \times \cdots \times [s_d, t_d] \subseteq U_d$, where $\mathrm{Vol}(r') \geq a + 2\delta$. Put $\Delta_j = t_j - s_j$ for $j = 1, \ldots, d$; we have $\mathrm{Vol}(r') = \prod_{j=1}^{d} \Delta_j$ and

$$a + 2\delta \leq \Delta_j \leq 1, \text{ for } j = 1, \ldots, d.$$

By construction, each interval $[s_j, t_j]$ contains a *grid*-interval I_j of length

$$|I_j| \geq \Delta_j - \frac{2}{j-1} = \Delta_j - \frac{2\delta}{4d} = \Delta_j - \frac{\delta}{2d}.$$

Note that

$$|I_j| \geq a + 2\delta - \frac{\delta}{2d} \geq a + 1.75\delta.$$

Then $r_2 := \prod_{j=1}^{d} I_j \in \mathcal{R}_1$ is a *grid* hyper-rectangle (box) whose volume is

$$\begin{aligned} \mathrm{Vol}(r_2) &\geq \prod_{j=1}^{d} \left(\Delta_j - \frac{\delta}{2d}\right) \geq \prod_{j=1}^{d} \Delta_j - \sum_{j=1}^{d} \frac{\delta}{2d} \\ &= \prod_{j=1}^{d} \Delta_j - \frac{\delta}{2} \geq a + 2\delta - 0.5\,\delta = a + 1.5\delta. \end{aligned} \tag{6.13}$$

The first inequality above follows from Lemma 6.9 below with $k = d$, $a_i = \Delta_i$, and $\delta_i = \delta/(2d)$ for $i = 1, \ldots, k$. This implies $r_2 \in \mathcal{R}_2$. Furthermore, since r_2 is contained in r', it is empty of points of P in its interior.

By construction, the smaller concentric homothetic copy of r_2 of volume exactly $a + \delta$, denoted here by r, belongs to $\mathcal{R}_3$. Since r lies strictly inside r_2, we have $r \cap P = \emptyset$ and thus $r \in \mathcal{R}(P, a, \delta; d)$ as required. $\square$

Lemma 6.9. *Let $a, \delta \in (0, 1)$, where $a + 2\delta \leq \prod_{i=1}^{k} a_i \leq a_1, \ldots, a_k \leq 1$ and $\delta_1, \ldots, \delta_k > 0$,* *where* $\sum_{i=1}^{k} \delta_i \leq \delta$. *Then*

$$\prod_{i=1}^{k}(a_i - \delta_i) \geq \prod_{i=1}^{k} a_i - \sum_{i=1}^{k} \delta_i.$$

Proof. By the hypothesis, we have $a_i \geq a + 2\delta > \delta > \delta_i$, for every i. We prove the inequality by induction on k. For $k = 1$ there is nothing to prove. Assume that the inequality holds for $k - 1$:

$$\prod_{i=1}^{k-1}(a_i - \delta_i) \geq \prod_{i=1}^{k-1} a_i - \sum_{i=1}^{k-1} \delta_i. \tag{6.14}$$

The left hand-side of (6.14) is clearly positive. Observe that

$$\prod_{i=1}^{k-1} a_i - \sum_{i=1}^{k-1} \delta_i \geq \prod_{i=1}^{k} a_i - \sum_{i=1}^{k} \delta_i \geq a + 2\delta - \delta = a + \delta,$$

thus the right hand-side of (6.14) is also positive. We can multiply the inequality by $(a_k - \delta_k) \geq a + \delta > 0$. This yields

$$\begin{aligned}\prod_{i=1}^{k}(a_i - \delta_i) &\geq \left(\prod_{i=1}^{k-1} a_i - \sum_{i=1}^{k-1} \delta_i\right)(a_k - \delta_k) \\ &\geq \prod_{i=1}^{k} a_i - \sum_{i=1}^{k} \delta_i + \delta_k \left(\sum_{i=1}^{k-1} \delta_i\right) \\ &\geq \prod_{i=1}^{k} a_i - \sum_{i=1}^{k} \delta_i.\end{aligned}$$

Indeed, the next to last inequality is equivalent to

$$\sum_{i=1}^{k} \delta_i \geq a_k \sum_{i=1}^{k-1} \delta_i + \delta_k \prod_{i=1}^{k-1} a_i,$$

which is implied by $a_k \leq 1$ and $\prod_{i=1}^{k-1} a_i \leq 1$.

We have thus shown that the inequality holds for k and this completes the induction proof. $\square$

The connection between piercing numbers and Maximum Empty Box is highlighted by the following two lemmas.

Lemma 6.10. *For $a, \delta > 0$, where $a + 2\delta < 1$, and $1/\delta \in \mathbb{N}$, if $A_d(n) \geq a + 2\delta$ holds for some $n \in \mathbb{N}$, then $\tau(\mathcal{R}(\emptyset, a, \delta; d)) \geq n + 1$.*

Proof. Assume for contradiction that there exists a piercing set P with n points for $\mathcal{R}(\emptyset, a, \delta; d)$. Since $A_d(n) \geq a + 2\delta$ by assumption, there exists a hyper-rectangle r' amidst the points in P that is empty in its interior, whose volume is at least $a + 2\delta$. By Lemma 6.8, there exists a hyper-rectangle $r \in \mathcal{R}(\emptyset, a, \delta; d)$, with $r \subset r'$ and $\mathrm{Vol}(r) = a + \delta$ and $r \cap P = \emptyset$, in contradiction to our assumption that P is a piercing set for $\mathcal{R}(\emptyset, a, \delta; d)$. This concludes the proof. $\square$

Lemma 6.11. *Let $P \subset U_d$ be a finite point set and $a, \delta > 0$, where $a + 2\delta < 1$, $1/\delta \in \mathbb{N}$, and $A_d(P) \geq a + 2\delta$. Let $r'_1, \ldots, r'_j \in \mathcal{R}'(P, a, \delta; d)$ be j empty hyper-rectangles that require j piercing points in $U_d \setminus P$. Then $\tau(\mathcal{R}(P, a, \delta; d)) \geq j$.*

Proof. By Lemma 6.8, for every hyper-rectangle r'_i, $1 \leq i \leq j$, there exists a hyper-rectangle $r_i \in \mathcal{R}(P, a, \delta; d)$, with $r_i \subset r'_i$ and $\mathrm{Vol}(r_i) = a + \delta$ and $r_i \cap P = \emptyset$. Since piercing $r'_1, \ldots, r'_j$ requires j piercing points in $U_d \setminus P$, piercing $r_1, \ldots, r_j$ also requires j piercing points in $U_d \setminus P$. Consequently, $\tau(\mathcal{R}(P, a, \delta; d)) \geq j$. $\square$

Piercing a set of rectangles (contained in $[0, 1]^2$) whose areas are above some threshold is dual to the problem of finding a large empty rectangle (beyond this threshold) amidst the points in the piercing set. This insight could be used directly in the pursuit of a better lower bound for Wegner's inequality, it may, however, be ineffective. Here we extend the system of rectangles by adding a *grid* of segments (i.e., degenerate rectangles) as explained below. The main idea is that piercing the grid segments imposes constraints on the position of the piercing points and this allows the existence of a large empty rectangle.

6.2.3 Construction

All rectangles in our construction are axis-parallel and contained in the unit square $U = [0,1]^2$. Let $k \geq 2$ be a fixed integer; here we will work with $k = 2, 3, 4$. Let $a = 1/(k+1)$ and $\delta = 10^{-3}$. Note that $1/\delta \in \mathbb{N}$, as required by Lemma 6.8. Let $\mathcal{R}' = \mathcal{R}'(\emptyset, a, \delta; 2)$ and let $\mathcal{R} = \mathcal{R}(\emptyset, a, \delta; 2)$ be the system obtained from $\mathcal{R}'$ as in Lemma 6.8. Recall that $\mathcal{R}'$ is the set of all rectangles contained in U with area at least $1/(k+1) + 2\delta$; and that the area of each rectangle in (the finite family) $\mathcal{R}$ is $1/(k+1) + \delta$.

Our construction is the following finite family of rectangles (see Figure 6.2 (left) for $k = 2$):

$$\mathcal{S} = \mathcal{R} \cup \mathcal{G}, \tag{6.15}$$

where $\mathcal{G}$ is the $k \times k$ *grid* described below.

$$\begin{aligned}
\mathcal{H} &= \{[1/(k+1), k/(k+1)] \times \{i/(k+1)\}, i = 1, 2, \ldots, k\},\\
\mathcal{V} &= \{\{i/(k+1)\} \times [1/(k+1), k/(k+1)], i = 1, 2, \ldots, k\},\\
\mathcal{G} &= \mathcal{H} \cup \mathcal{V}. \qquad (6.16)
\end{aligned}$$

We show that (for every $k \geq 2$) the matching number of this family is equal to k.

Lemma 6.12. *$\nu(\mathcal{S}) = k$.*

Proof. The k (degenerate) rectangles in $\mathcal{H}$ immediately yield $\nu(\mathcal{S}) \geq k$. It remains to prove the upper bound. Let $\mathcal{I}$ be an independent set of rectangles in $\mathcal{S}$. If $\mathcal{I}$ consists only of segments in $\mathcal{G}$, it is clear that $|\mathcal{I}| \leq k$. If $\mathcal{I}$ consists only of rectangles in $\mathcal{R}$, a simple area argument implies that

$$|\mathcal{I}| \leq \left\lfloor \frac{1}{1/(k+1) + \delta} \right\rfloor \leq k.$$

Assume now that $\mathcal{I}$ consists of rectangles in $\mathcal{R}$ and grid segments in $\mathcal{G}$. Observe that any segment $s \in \mathcal{H}$ divides U into a top and a bottom region in the sense that any rectangle from $\mathcal{R}$ whose vertical extent intersects s must intersect s. A similar observation applies to segments in $\mathcal{V}$. If multiple segments are in $\mathcal{I}$, these segments must be members of the same family ($\mathcal{H}$ or $\mathcal{V}$), and the regions are further subdivided in the same manner. For each resulting region, the same area argument gives an upper bound on the number of independent rectangles from $\mathcal{R}$ in that region.

Specifically, let $h = |\mathcal{I} \cap \mathcal{H}|$; without loss of generality, we may assume that $h > 0$. Let $i_1, \ldots, i_h \in \{1, 2, \ldots, k\}$ be the subscripts of the segments in $\mathcal{I} \cap \mathcal{H}$ in ascending order. For convenience, put $i_0 = 0$ and $i_{h+1} = k + 1$. The area argument yields:

$$\begin{aligned}|\mathcal{I} \cap \mathcal{R}| &\leq \sum_{j=0}^{h} \left\lfloor \frac{(i_{j+1} - i_j)/(k+1)}{1/(k+1) + \delta} \right\rfloor = \sum_{j=0}^{h} \left\lfloor \frac{i_{j+1} - i_j}{1 + (k+1)\delta} \right\rfloor \\ &\leq \sum_{j=0}^{h} (i_{j+1} - i_j - 1) = i_{h+1} - i_0 - (h+1) = (k+1) - (h+1) = k - h.\end{aligned}$$

Therefore, $|\mathcal{I}| = |\mathcal{I} \cap \mathcal{G}| + |\mathcal{I} \cap \mathcal{R}| \leq h + (k - h) = k$, as required. □

Key terms used in bounding the piercing number. Let P be a piercing set for $\mathcal{S}$. Let X denote the set of k^2 grid points in $\mathcal{H} \cap \mathcal{V}$. A subset of X is *independent* if no two points are on the same grid segment (that is, no two coordinates are the same). Let J denote a maximal set of independent points in $X \cap P$. Obviously $0 \leq |J| \leq k$. Assume in what follows that $|J| < k$. Let $\mathcal{H}(J)$ and $\mathcal{V}(J)$ be the grid segments pierced by J. Consider any pair of segments h, v, where $h \in \mathcal{H} \setminus \mathcal{H}(J)$ and $v \in \mathcal{V} \setminus \mathcal{V}(J)$; then h and v cannot be pierced by the common point $h \cap v$, because $J \cup \{h \cap v\}$ would be an independent set of larger cardinality than J, a contradiction. Therefore, we can view this pair of segments as "disjoint" although they share a common point. Broadly, we refer to any set of t rectangles that are pierced in $P \setminus J$ by t distinct piercing points as *quasi-disjoint* (with respect to the given J).

In our analyses we distinguish several cases depending on the size of J, as defined above, and use quasi-disjointness to infer the possible structure of a piercing set (Subsection 6.3.2 is the first such use).

6.3 Preliminary constructions

The simplest lower bound example with $\tau/\nu \geq 3/2$ is the "5-cycle" from the hypergraph setting (also mentioned in [26]): five rectangles forming a cycle where each rectangle only intersects its two neighbors in the cycle. It is worth noting that the 5-cycle can be realized with (axis-aligned) unit squares; see Figure 6.1. Our construction here is more complex but the relatively simple proofs in this section pave the way for the proof of Theorem 6.6.

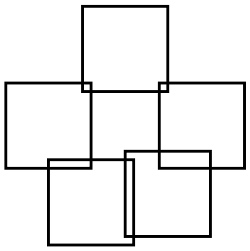


Figure 6.1: A 5-cycle made from unit squares.

6.3.1 $k = 2$

According to (6.15) and (6.16), our construction consists of four segments that make $\mathcal{G}$, see Figure 6.2 (left), and a finite number of rectangles with area $1/3 + \delta$.

By Lemma 6.12 for $k = 2$ we have $\nu(\mathcal{S}) = 2$. For example, in Figure 6.2 (right), the segment $s = [1/3, 2/3] \times \{2/3\}$ divides U into the top region with area $1/3$ and the bottom region with area $2/3$. The top region can have at most $\left\lfloor \frac{1/3}{1/3+\delta} \right\rfloor = 0$ rectangles from $\mathcal{R}$ in an independent set $\mathcal{I}$, and the bottom region can have at most $\left\lfloor \frac{1/3}{1/3+\delta} \right\rfloor = 1$ rectangle from $\mathcal{R}$ in $\mathcal{I}$. Thus, any independent set containing s has size at most 2; and the same bound holds for any other case.

To see that this construction gives the ratio $\tau/\nu = 3/2$, it suffices to prove the following:

Claim 6.13. $\tau(\mathcal{S}) = 3$.

Proof. To see that $\tau(\mathcal{S}) \leq 3$, consider the three points shown in Figure 6.3 (left), namely $(1/3, 2/3)$, $(1/2, 1/2)$, and $(2/3, 1/3)$. First note that all segments in $\mathcal{G}$ are pierced. It is easy

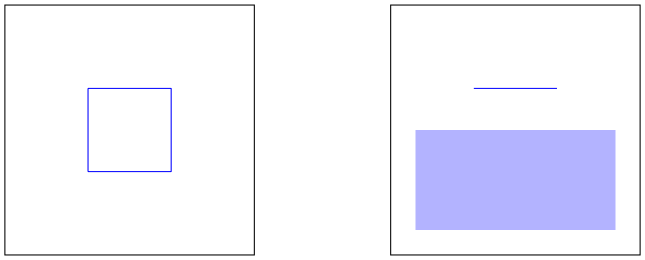


Figure 6.2: Left: the four segments in $\mathcal{G}$. Right: an independent set of size 2 in $\mathcal{S} = \mathcal{R} \cup \mathcal{G}$.

to check that with the aforementioned three points, the maximum empty rectangle in U has area $1/3$. Therefore, all rectangles in $\mathcal{R}$ are pierced.

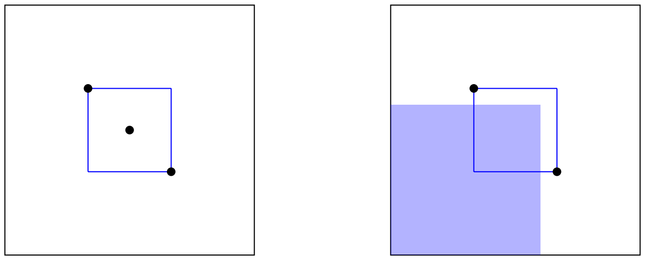


Figure 6.3: Left: three points piercing all rectangles in $\mathcal{S} = \mathcal{R} \cup \mathcal{G}$. Right: two points required to pierce all segments in $\mathcal{G}$ leave a rectangle from $\mathcal{R}$ unpierced.

We now prove the lower bound. Assume for contradiction that $\tau(\mathcal{S}) \leq 2$, i.e., there exist two points in U that collectively pierce the rectangles in $\mathcal{S}$. Observe that at least two points are required to pierce all segments in $\mathcal{G}$. Up to symmetry by rotation and reflection of U, there is only one case, shown in Figure 6.3 (right), where the two points are $(1/3, 2/3)$ and $(2/3, 1/3)$. Consider the rectangle $r' = [0, 0.6] \times [0, 0.6] \in \mathcal{R}'$. We have $\text{Area}(r') = 0.36 > 1/3 + 2\delta$, so by Lemma 6.8 there is a rectangle $r \in \mathcal{R}$ that is contained in r'. Since r' is not pierced by the two points, neither is r. $\square$

6.3.2 $k = 3$

The first lower bound example with $\tau/\nu \geq 5/3$ was found by Wegner [33] (see also [16]) in 1968. It has 23 rectangles. A slight variation of this example was later independently found by Fon-Der-Flaass and Kostochka [17].

According to (6.15) and (6.16), our construction consists of six segments making the

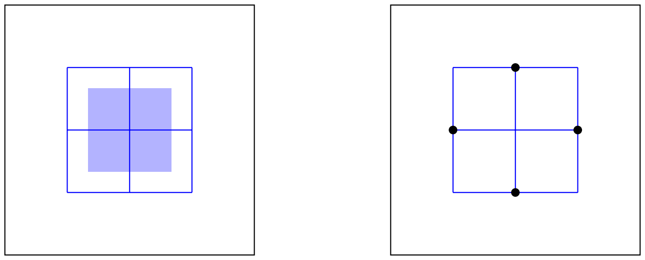


Figure 6.4: Left: the six segments in $\mathcal{G}$ and the center square Q. Right: $\mathcal{R} \cup \mathcal{G}$ can be pierced by 4 points.

3×3 grid $\mathcal{G}$ and a finite number of rectangles with area $1/4 + \delta$; see Figure 6.4 (left). By Lemma 6.12 with $k = 3$, the system $\mathcal{S} = \mathcal{G} \cap \mathcal{R}$ has $\nu = 3$; however, $\tau < 5$. In fact, $\mathcal{S}$ can be pierced by the four points $(1/4, 1/2)$, $(1/2, 1/4)$, $(1/2, 3/4)$, and $(3/4, 1/2)$, depicted in Figure 6.4 (right). We therefore add the center square $Q = [1/3, 2/3] \times [1/3, 2/3]$ and redefine $\mathcal{S} := \mathcal{R} \cup \mathcal{G} \cup \{Q\}$.

Adding Q introduces a few more cases in the proof of $\nu(\mathcal{S}) = 3$, but the idea is the same as in the proof of Lemma 6.12 in Section 6.2.3; here we omit the details. To show $\tau(\mathcal{S}) \leq 5$, observe that adding the point $(1/2, 1/2)$ to the earlier set of four points is enough to pierce all rectangles in $\mathcal{S}$.

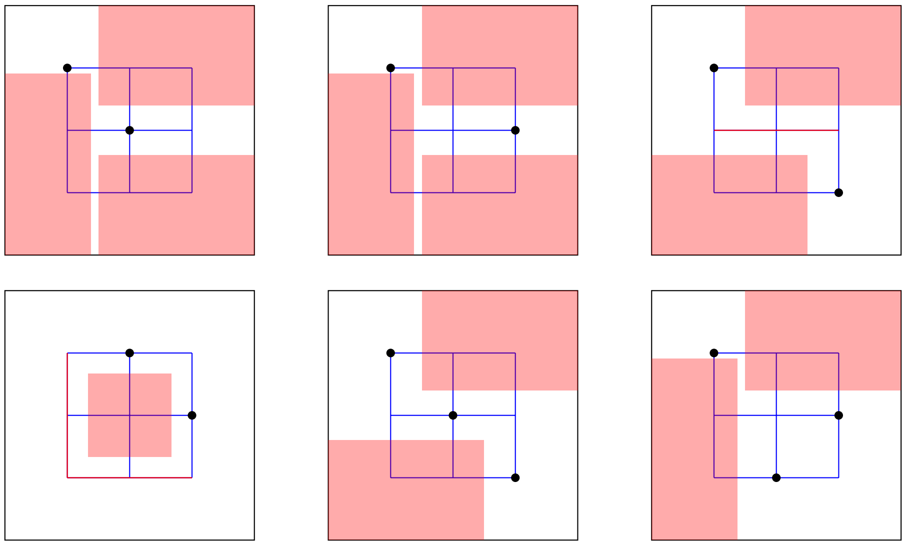


Figure 6.5: Four cases for $|J| = 2$ and two cases for $|J| = 3$; disjoint unpierced rectangles are shown in red. (In the 4th case for $|J| = 2$, the point $h_1 \cap v_1$ cannot be used to pierce the two red segments!)

Assume for contradiction that $\tau(\mathcal{S}) = 4$, i.e., P is a 4-point piercing set for $\mathcal{S}$. In particular, P is a piercing set for $\mathcal{G}$. Let X denote the set of 9 grid points in $\mathcal{H} \cap \mathcal{V}$. Let J denote a maximal set of independent points in $X \cap P$. Obviously $0 \leq |J| \leq 3$. Since points in J are independent, they together cover $2|J|$ grid segments. By the maximality of J, each of the remaining $|P| - |J|$ points in P can cover at most one new grid segment from the remaining $6 - 2|J|$. To cover all the segments in $\mathcal{G}$, we have $|P| - |J| \geq 6 - 2|J|$ which yields $|J| \geq 2$. Up to symmetry by rotation and reflection of U, if $|J| = 2$, there are four cases. In each of them, we exhibit three unpierced quasi-disjoint rectangles from $\mathcal{R}' \cup \mathcal{G} \cup \{Q\}$, thus by Lemma 6.11, at least three more points are needed; however, there are only two available points in P, a contradiction. If $|J| = 3$, there are two cases. In each of them, we exhibit two disjoint unpierced rectangles from $\mathcal{R}'$, thus by Lemma 6.11, at least two more points are needed, see Figure 6.5; however, there is only one available point in P, a contradiction.

6.4 Main construction: $k = 4$

In this section we prove Theorem 6.6. Recall that $\mathcal{S} = \mathcal{R} \cup \mathcal{G}$ where $\mathcal{R}$ consists of a finite number of rectangles with area $1/5 + \delta$ and $\mathcal{G}$ is a 4×4 grid, see Figure 6.6 (left).

$$\begin{aligned}
&\mathcal{H} = \{h_i = [1/5, 4/5] \times \{i/5\}, i = 1, 2, 3, 4\}, \\
&\mathcal{V} = \{v_i = \{i/5\} \times [1/5, 4/5], i = 1, 2, 3, 4\}, \\
&\mathcal{G} = \mathcal{H} \cup \mathcal{V}.
\end{aligned}$$

By Lemma 6.12 for $k = 4$ we have $\nu(\mathcal{S}) = 4$. Figure 6.6 (middle) shows an independent set of size 4 in $\mathcal{S}$. To obtain Theorem 6.6, we need the following.

Lemma 6.14. $\tau(\mathcal{S}) = 7$.

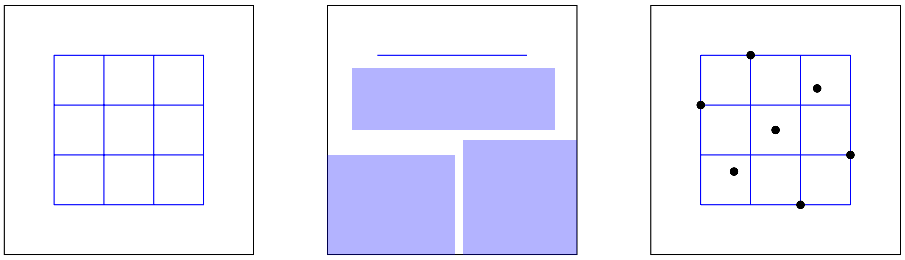


Figure 6.6: Left: the eight segments in $\mathcal{G}$. Middle: an independent set of size 4 in $\mathcal{S} = \mathcal{R} \cup \mathcal{G}$. Right: a set of 7 points piercing $\mathcal{S}$.

Proof. The following set of 7 piercing points shows that $\tau(\mathcal{S}) \leq 7$; see Figure 6.6 (right).

$$P = \left\{ \left(\frac{1}{5}, \frac{3}{5}\right), \left(\frac{1}{3}, \frac{1}{3}\right), \left(\frac{2}{5}, \frac{4}{5}\right), \left(\frac{1}{2}, \frac{1}{2}\right), \left(\frac{3}{5}, \frac{1}{5}\right), \left(\frac{2}{3}, \frac{2}{3}\right), \left(\frac{4}{5}, \frac{2}{5}\right) \right\}.$$

Observe first that all the segments in $\mathcal{G}$ are pierced. It is not hard to verify that the area of the largest empty rectangle in U amidst these 7 points is equal to $1/5$. Recall that the area of every rectangle in $\mathcal{R}$ is $1/5 + \delta$, therefore all rectangles in $\mathcal{R}$ are pierced.

The proof of the lower bound $\tau(\mathcal{S}) \geq 7$ is more involved, but the idea is the same as in the earlier proofs for $k = 2$ and 3. Assume for contradiction that there exists a set P of 6 points in U that collectively pierce all the rectangles in $\mathcal{S}$. We show that piercing the grid segments in $\mathcal{G}$ imposes constraints on the position of the piercing points and this allows the existence of a large empty rectangle, i.e., one whose area is at least $1/5 + 2\delta$. By Lemma 6.11, this further implies the existence of an unpierced rectangle whose area is $1/5 + \delta$ in the system $\mathcal{R}$ (and thus in $\mathcal{S}$), which contradicts the assumption that P is a piercing set for $\mathcal{S}$.

Let X denote the set of 16 grid points in $\mathcal{H} \cap \mathcal{V}$. Recall that $\mathcal{H} = \{h_1, h_2, h_3, h_4\}$ and $\mathcal{V} = \{v_1, v_2, v_3, v_4\}$ (labeled in ascending order of y- and x-coordinates, respectively). Let J denote a maximal set of independent points in $X \cap P$. Obviously $0 \leq |J| \leq 4$. Since points in J are independent, they together cover $2|J|$ grid segments. By the maximality of J, each of the remaining $|P| - |J|$ points in P can cover at most one new grid segment from the remaining $8 - 2|J|$. To cover all the segments in $\mathcal{G}$, we have $|P| - |J| \geq 8 - 2|J|$ which yields $|J| \geq 2$. Henceforth, we distinguish three cases depending on the size of J.

Case $|J| = 4$. Up to symmetry by rotation and reflection of U, there are 7 configurations for these 4 points, see Figure 6.7. For each configuration, we provide 3 unpierced disjoint rectangles, each of area at least $1/5 + 2\delta$. This shows that there are 3 unpierced disjoint rectangles in $\mathcal{R}$ which cannot all be pierced by the remaining $|P| - |J| = 6 - 4 = 2$ points.

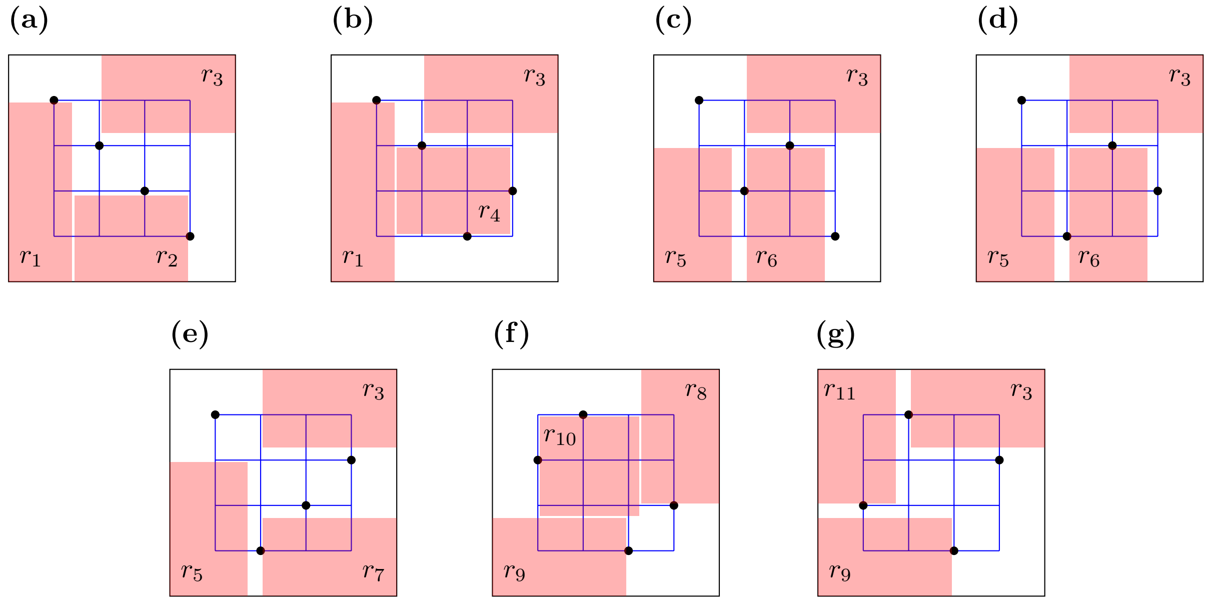


Figure 6.7: Seven configurations for $|J| = 4$, the unpierced disjoint rectangles are shown in red.

The following 11 rectangles, $r_1, \ldots, r_{11}$ are used. Observe that all entries in the third column of Table 6.3, representing *excess areas*, are nonnegative for $\delta \leq 10^{-3}$ (the bottleneck entry is r_{10}, its excess area vanishes for $\delta \approx 1/430$).

Rectangle	Dimensions	Area $-$ $(1/5 + 2\delta)$
r_1	$[0, 1/4 + 3\delta] \times [0, 4/5 - \delta]$	$3\delta/20 - 3\delta^2$
r_2	$[1/4 + 4\delta, 4/5 - \delta] \times [0, 2/5 - 2\delta]$	$1/50 - 51\delta/10 + 10\delta^2$
r_3	$[2/5 + \delta, 1] \times [2/3 - 4\delta, 1]$	$\delta/15 - 4\delta^2$
r_4	$[1/4 + 4\delta, 4/5 - \delta] \times [1/5 + \delta, 3/5 - \delta]$	$1/50 - 51\delta/10 + 10\delta^2$
r_5	$[0, 1/3 + 4\delta] \times [0, 3/5 - \delta]$	$\delta/15 - 4\delta^2$

r_6	$[2/5+\delta, 11/15+5\delta] \times [0, 3/5-\delta]$	$\delta/15 - 4\delta^2$
r_7	$[2/5+\delta, 1] \times [0, 1/3+4\delta]$	$\delta/15 - 4\delta^2$
r_8	$[2/3-4\delta, 1] \times [2/5+\delta, 1]$	$\delta/15 - 4\delta^2$
r_9	$[0, 3/5-\delta] \times [0, 1/3+4\delta]$	$\delta/15 - 4\delta^2$
r_{10}	$[1/5+\delta, 2/3-5\delta] \times [1/3+5\delta, 4/5-\delta]$	$4/225 - 38\delta/5 + 36\delta^2$
r_{11}	$[0, 1/3+4\delta] \times [2/5+\delta, 1]$	$\delta/15 - 4\delta^2$

Table 6.3: List of rectangles used for the case $|J| = 4$.

The argument is summarized in the following table:

Configuration	J	3 unpierced disjoint rectangles
(a)	$\{(1/5, 4/5), (2/5, 3/5), (3/5, 2/5), (4/5, 1/5)\}$	r_1, r_2, r_3
(b)	$\{(1/5, 4/5), (2/5, 3/5), (3/5, 1/5), (4/5, 2/5)\}$	r_1, r_3, r_4
(c)	$\{(1/5, 4/5), (2/5, 2/5), (3/5, 3/5), (4/5, 1/5)\}$	r_3, r_5, r_6
(d)	$\{(1/5, 4/5), (2/5, 1/5), (3/5, 3/5), (4/5, 2/5)\}$	r_3, r_5, r_6
(e)	$\{(1/5, 4/5), (2/5, 1/5), (3/5, 2/5), (4/5, 3/5)\}$	r_3, r_5, r_7
(f)	$\{(1/5, 3/5), (2/5, 4/5), (3/5, 1/5), (4/5, 2/5)\}$	r_8, r_9, r_{10}
(g)	$\{(1/5, 2/5), (2/5, 4/5), (3/5, 1/5), (4/5, 3/5)\}$	r_3, r_9, r_{11}

Table 6.4: $|J| = 4$. All 7 configurations are handled by providing three unpierced disjoint rectangles.

Case $|J| = 3$. Up to symmetry by rotation and reflection of U, there are 16 configurations for these 3 points, see Figure 6.8. For each configuration, there are $|\mathcal{G}| - 2|J| = 8 - 6 = 2$ unpierced grid segments. These two segments must intersect otherwise J is not independent. However they cannot be pierced by one point because otherwise the configuration would have been handled in the previous case where $|J| = 4$; as such, they are quasi-disjoint. Out of the 16 configurations, we handle 13 using the same technique as in the previous case. That is, we exhibit 4 unpierced quasi-disjoint rectangles, each of them either is a grid segment (i.e., in $\mathcal{G}$) or has area at least $1/5+2\delta$ (i.e., in $\mathcal{R}'$). This shows that there are 4 unpierced quasi-disjoint rectangles in $\mathcal{G} \cup \mathcal{R}$ which cannot all be pierced by the remaining $|P| - |J| = 6 - 3 = 3$ points. The other three configurations are handled differently.

The following 6 additional rectangles are used. Observe that all entries in the third column of Table 6.5, representing excess areas, are nonnegative for $\delta \leq 10^{-3}$ (the bottleneck entries are r_{12}, r_{15}, r_{16}, and r_{17}, their excess areas vanish for $\delta = 1/60$).

Rectangle	Dimensions	Area $-$ $(1/5 + 2\delta)$
r_{12}	$[0, 3/5 - \delta] \times [1/5 + \delta, 8/15 + 5\delta]$	$\delta/15 - 4\delta^2$
r_{13}	$[1/5 + \delta, 4/5 - \delta] \times [2/3 - 5\delta, 1]$	$\delta/3 - 10\delta^2$
r_{14}	$[0, 1/3 + 5\delta] \times [1/5 + \delta, 4/5 - \delta]$	$\delta/3 - 10\delta^2$
r_{15}	$[0, 3/5 - \delta] \times [2/5 + \delta, 11/15 + 5\delta]$	$\delta/15 - 4\delta^2$
r_{16}	$[2/3 - 4\delta, 1] \times [0, 3/5 - \delta]$	$\delta/15 - 4\delta^2$
r_{17}	$[7/15 - 5\delta, 4/5 - \delta] \times [2/5 + \delta, 1]$	$\delta/15 - 4\delta^2$

Table 6.5: List of additional rectangles used for the case $|J| = 3$.

Arguments for the first 13 configurations are summarized in the following table:

Configuration	J	4 unpierced quasi-disjoint rectangles
(a)	$\{(1/5, 4/5), (2/5, 3/5), (3/5, 2/5)\}$	h_1, v_4, r_{12}, r_{13}
(b)	$\{(1/5, 4/5), (2/5, 3/5), (4/5, 2/5)\}$	h_1, v_3, r_8, r_{12}
(c)	$\{(1/5, 4/5), (2/5, 3/5), (4/5, 1/5)\}$	h_2, v_3, r_8, r_9
(d)	$\{(1/5, 4/5), (3/5, 3/5), (4/5, 2/5)\}$	h_1, v_2, r_3, r_{14}

(e)	$\{(1/5,4/5),(2/5,1/5),(3/5,3/5)\}$	h_2, r_7, r_8, r_{15}
(f)	$\{(1/5,4/5),(3/5,3/5),(4/5,1/5)\}$	v_2, r_3, r_5, r_6
(g)	$\{(1/5,4/5),(3/5,2/5),(4/5,3/5)\}$	v_2, r_3, r_5, r_{16}
(h)	$\{(1/5,4/5),(3/5,1/5),(4/5,3/5)\}$	v_2, r_3, r_5, r_{16}
(i)	$\{(1/5,4/5),(3/5,1/5),(4/5,2/5)\}$	h_3, v_2, r_3, r_5
(j)	$\{(1/5,3/5),(2/5,4/5),(4/5,2/5)\}$	h_1, v_3, r_8, r_{12}
(k)	$\{(1/5,2/5),(2/5,4/5),(3/5,3/5)\}$	r_3, r_9, r_{11}, r_{16}
(l)	$\{(2/5,4/5),(3/5,3/5),(4/5,2/5)\}$	h_1, v_1, r_3, r_4
(m)	$\{(1/5,2/5),(2/5,4/5),(4/5,3/5)\}$	v_3, r_9, r_{11}, r_{16}

Table 6.6: $|J| = 3$. The first 13 configurations are handled by exhibiting 4 unpierced quasi-disjoint rectangles.

For each of the remaining three configurations, we exhibit 7 unpierced rectangles, each of which is either a grid segment (i.e., in $\mathcal{G}$) or has area at least $1/5 + 2\delta$ (i.e., in $\mathcal{R}'$). Observe that each point in $U \setminus X$ (recall that X is the set of grid points) can cover at most 2 of these 7 rectangles. Therefore, at least 4 more piercing points are needed. The arguments are summarized in the following table:

Configuration	J	7 unpierced rectangles
(n)	$\{(1/5,4/5),(2/5,2/5),(3/5,3/5)\}$	$h_1, v_4, r_3, r_9, r_{14}, r_{15}, r_{16}$
(o)	$\{(1/5,4/5),(2/5,1/5),(4/5,3/5)\}$	$h_2, v_3, r_3, r_5, r_7, r_{15}, r_{16}$
(p)	$\{(1/5,3/5),(2/5,4/5),(3/5,2/5)\}$	$h_1, v_4, r_3, r_5, r_7, r_{12}, r_{17}$

Table 6.7: $|J| = 3$. The remaining 3 configurations are handled by exhibiting 7 unpierced rectangles that need at least 4 additional piercing points.

Case $|J| = 2$. Up to symmetry by rotation and reflection of U, there are 13 configurations

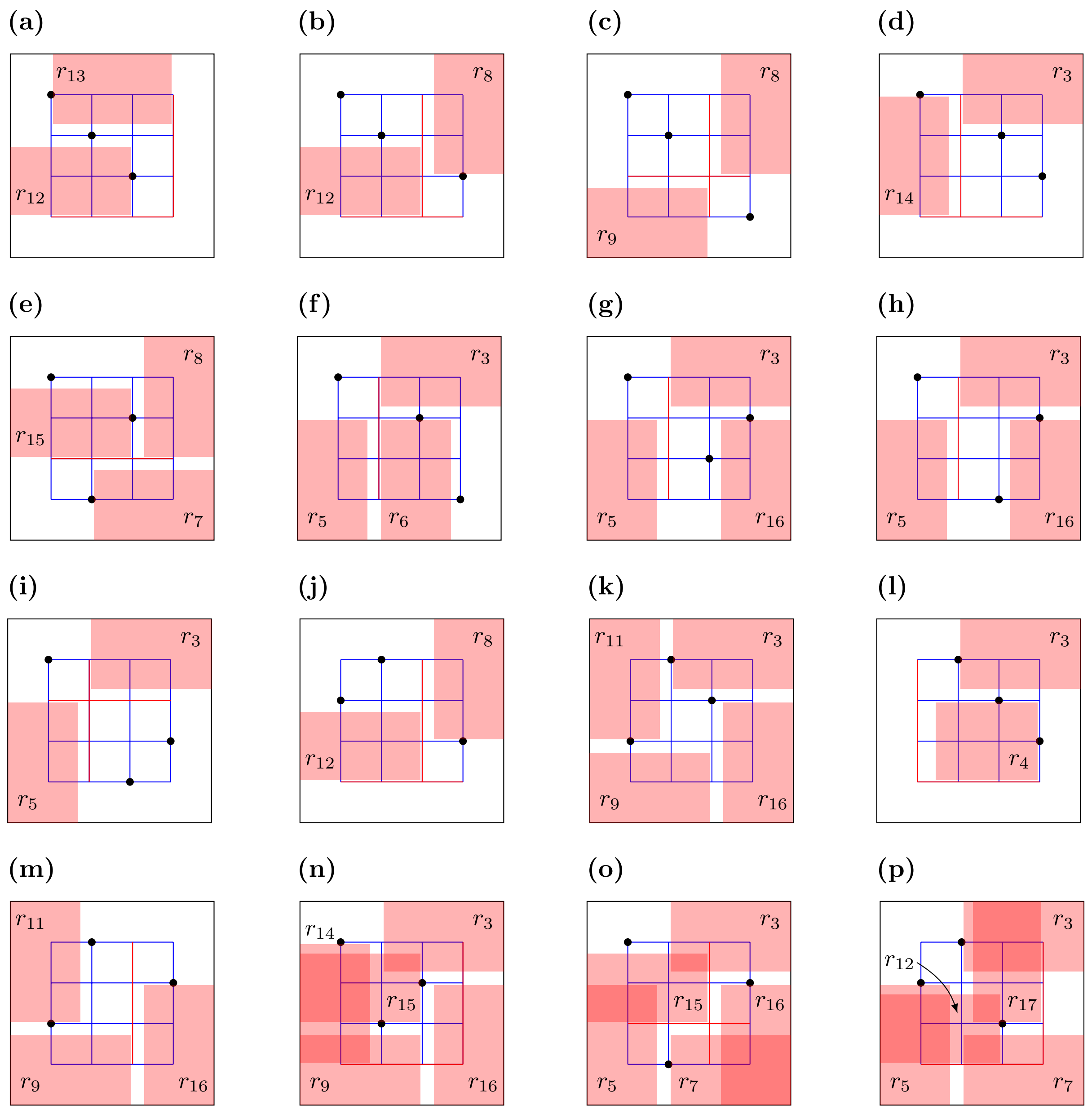


Figure 6.8: 16 configurations for $|J| = 3$. For the first 13 configurations, the unpierced quasi-disjoint rectangles are shown in red; for the last 3 configurations, the 7 unpierced rectangles that need at least 4 additional piercing points are shown in red.

for these 2 points, see Figure 6.9. For each configuration, there are $|G| - 2|J| = 8 - 4 = 4$ unpierced grid segments. As in the previous case, these segments are quasi-disjoint (i.e., cannot be pierced at their intersections) because otherwise the configuration would have been handled in the previous cases where $|J| = 3$ or 4. Note that there remain $|P| - |J| = 6 - 2 = 4$ points, so they must all be placed on segments in $\mathcal{G}$, one for each unpierced segment.

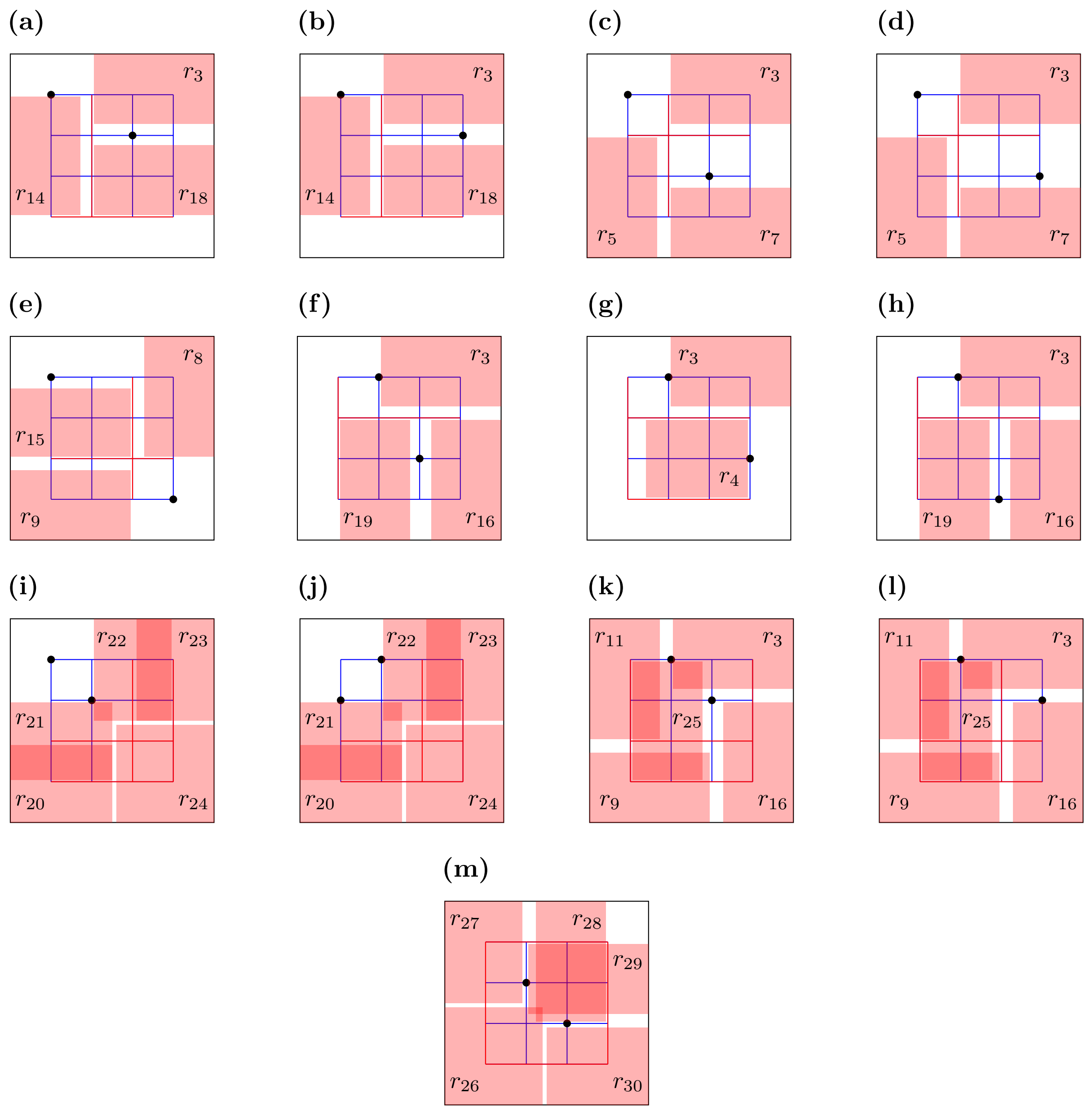

(a)
r_3 r_{14} r_{18}
(b)
r_3 r_{14} r_{18}
(c)
r_3 r_5 r_7
(d)
r_3 r_5 r_7
(e)
r_8 r_{15} r_9
(f)
r_3 r_{19} r_{16}
(g)
r_3 r_4
(h)
r_3 r_{19} r_{16}
(i)
r_{22} r_{23} r_{21} r_{20} r_{24}
(j)
r_{22} r_{23} r_{21} r_{20} r_{24}
(k)
r_{11} r_3 r_{25} r_9 r_{16}
(l)
r_{11} r_3 r_{25} r_9 r_{16}
(m)
r_{27} r_{28} r_{29} r_{26} r_{30}

Figure 6.9: 13 configurations for $|J| = 2$. For the first 8 configurations, the unpierced quasi-disjoint rectangles are shown in red; for the last 5 configurations, the 9 unpierced rectangles that cannot be pierced by 4 points in $\mathcal{G} \setminus X$ are shown in red.

Out of the 13 configurations, we handle the first 8 using the same technique as in the previous cases. That is, we provide 5 unpierced quasi-disjoint rectangles, each of them either is a grid segment (i.e., in $\mathcal{G}$) or has area at least $1/5 + 2\delta$ (i.e., in $\mathcal{R}'$). This shows that there are 5 unpierced quasi-disjoint rectangles in $\mathcal{G} \cup \mathcal{R}$ which cannot all be pierced by the remaining 4 points. The remaining 5 configurations are handled differently.

The following 13 additional rectangles are used. Observe that all entries in the third column of Table 6.8 are nonnegative for $\delta \leq 10^{-3}$ (the bottleneck entries are $r_{20}, r_{21}, r_{22}, r_{23}, r_{27}$, and r_{30}, their excess areas vanish for $\delta \approx 1/255$).

Rectangle	Dimension	Area $-(1/5+2\delta)$
r_{18}	$[2/5+\delta, 1] \times [1/5+\delta, 8/15+5\delta]$	$\delta/15 - 4\delta^2$
r_{19}	$[1/5+\delta, 8/15+5\delta] \times [0, 3/5-\delta]$	$\delta/15 - 4\delta^2$
r_{20}	$[0, 11/20-5\delta] \times [0, 2/5-2\delta]$	$1/50 - 51\delta/10 + 10\delta^2$
r_{21}	$[0, 11/20-5\delta] \times [1/5+\delta, 3/5-\delta]$	$1/50 - 51\delta/10 + 10\delta^2$
r_{22}	$[2/5+\delta, 4/5-\delta] \times [9/20+5\delta, 1]$	$1/50 - 51\delta/10 + 10\delta^2$
r_{23}	$[3/5+2\delta, 1] \times [9/20+5\delta, 1]$	$1/50 - 51\delta/10 + 10\delta^2$
r_{24}	$[11/20-3\delta, 1] \times [0, 9/20+3\delta]$	$1/400 + 7\delta/10 + 9\delta^2$
r_{25}	$[1/5+\delta, 8/15+6\delta] \times [1/5+\delta, 4/5-\delta]$	$\delta/3 - 10\delta^2$
r_{26}	$[0, 9/20+3\delta, 1] \times [0, 9/20+3\delta]$	$1/400 + 7\delta/10 + 9\delta^2$
r_{27}	$[0, 2/5-2\delta] \times [9/20+5\delta, 1]$	$1/50 - 51\delta/10 + 10\delta^2$
r_{28}	$[7/15-5\delta, 4/5-\delta] \times [2/5+\delta, 1]$	$\delta/15 - 4\delta^2$
r_{29}	$[2/5+\delta, 1] \times [7/15-5\delta, 4/5-\delta]$	$\delta/15 - 4\delta^2$
r_{30}	$[9/20+5\delta, 1] \times [0, 2/5-2\delta]$	$1/50 - 51\delta/10 + 10\delta^2$

Table 6.8: List of additional rectangles used for the case $|J| = 2$.

Arguments for the first 8 configurations are summarized in the following table:

Configuration	J	5 unpierced quasi-disjoint rectangles
(a)	$\{(1/5, 4/5), (3/5, 3/5)\}$	$h_1, v_2, r_3, r_{14}, r_{18}$
(b)	$\{(1/5, 4/5), (4/5, 3/5)\}$	$h_1, v_2, r_3, r_{14}, r_{18}$

(c)	$\{(1/5, 4/5), (3/5, 2/5)\}$	h_3, v_2, r_3, r_5, r_7
(d)	$\{(1/5, 4/5), (4/5, 2/5)\}$	h_3, v_2, r_3, r_5, r_7
(e)	$\{(1/5, 4/5), (4/5, 2/5)\}$	$h_2, v_3, r_8, r_9, r_{15}$
(f)	$\{(2/5, 4/5), (3/5, 2/5)\}$	$h_3, v_1, r_3, r_{16}, r_{19}$
(g)	$\{(2/5, 4/5), (4/5, 2/5)\}$	h_1, h_3, v_1, r_3, r_4
(h)	$\{(2/5, 4/5), (3/5, 1/5)\}$	$h_3, v_1, r_3, r_{16}, r_{19}$

Table 6.9: $|J| = 2$. The first 8 configurations are handled by providing 5 unpierced quasi-disjoint rectangles.

There are 5 more configurations. For each of them, we exhibit 9 unpierced rectangles, each of which is either a grid segment (i.e., in $\mathcal{G}$) or has area at least $1/5 + 2\delta$ (i.e., in $\mathcal{R}'$). Recall that if $|P| = 6$, the remaining 4 piercing points must all lie on the grid segments $\mathcal{G} \setminus X$ (X is the set of grid points). But each point in $\mathcal{G} \setminus X$ can cover at most 2 of these 9 rectangles, which is a contradiction. The arguments are summarized in the following table:

Configuration	J	9 unpierced rectangles
(i)	$\{(1/5, 4/5), (2/5, 3/5)\}$	$h_1, h_2, v_3, v_4, r_{20}, r_{21}, r_{22}, r_{23}, r_{24}$
(j)	$\{(1/5, 3/5), (2/5, 4/5)\}$	$h_1, h_2, v_3, v_4, r_{20}, r_{21}, r_{22}, r_{23}, r_{24}$
(k)	$\{(2/5, 4/5), (3/5, 3/5)\}$	$h_1, h_2, v_1, v_4, r_3, r_9, r_{11}, r_{16}, r_{25}$
(l)	$\{(2/5, 4/5), (4/5, 3/5)\}$	$h_1, h_2, v_1, v_3, r_3, r_9, r_{11}, r_{16}, r_{25}$
(m)	$\{(2/5, 3/5), (3/5, 2/5)\}$	$h_1, h_4, v_1, v_4, r_{26}, r_{27}, r_{28}, r_{29}, r_{30}$

Table 6.10: $|J| = 2$. The remaining 5 configurations are handled by providing 9 unpierced rectangles that cannot be pierced by 4 points in $\mathcal{G} \setminus X$.

This concludes the proof of Lemma 6.14. □

Now Theorem 6.6 immediately follows from Lemma 6.12 (with $k = 4$) and Lemma 6.14.

Remarks.

1. An alternative proof of Theorem 6.6 could be obtained by restricting rectangles in $\mathcal{R}$ to those that are used in the proof of the lower bound on τ (Lemma 6.14) and their images under rotation and reflection of U. This would make the resulting lower bound example smaller with regard to number of rectangles in it. But its description would be tedious and not as enlightening as our presentation here.

2. A natural question is whether our construction can be used to create lower bound examples for larger k. Interestingly enough, for $k = 5$ the rectangles in $\mathcal{S}$ can be pierced by 8 points. In Figure 6.10, all the segments in $\mathcal{G}$ are pierced and the area of the largest empty rectangle is 1/6 so all rectangles in $\mathcal{R}$ are also pierced. Recall that in the construction for $k = 3$, we added a center square to increase τ by 1. A similar fix might be possible (but in view of Figure 6.10, a center square clearly won't work). Another major difficulty is that the number of cases required to prove a lower bound on τ grows rapidly with respect to k.

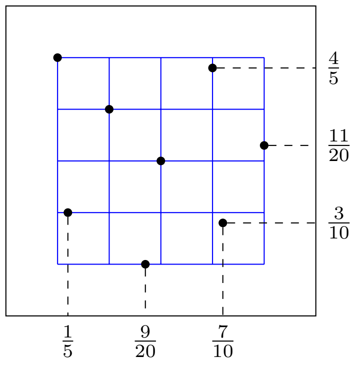


Figure 6.10: The system $\mathcal{S} = \mathcal{G} \cup \mathcal{R}$ for $k = 5$ can be pierced by 8 points.

3. The straightforward method to construct lower bounds by taking unions yields the following:

$$f(m + n) \geq f(m) + f(n), \text{ for every } m, n \geq 0. \tag{6.17}$$

By (6.17) and Theorem 6.6 we have $f(5) \geq f(4) + f(1) \geq 7 + 1 = 8$; alternatively, the result follows from $f(5) \geq f(3) + f(2) = 5 + 3 = 8$.

6.5 Higher dimensions

Proof of Theorem 6.7. By (6.11), we have $c_d \geq \frac{\log d}{4} \geq \frac{10}{4}$. By the definition of c_d, see (6.10), there exist arbitrarily large integers n such that $nA_d(n) \geq c_d - 1$. Let $k = \lfloor \frac{n}{c_d - 1} \rfloor + 1$. Since n can be arbitrarily large, we may assume that $k \geq 10$. On one hand, we have $k > \frac{n}{c_d-1}$ and thus $n < (c_d - 1)k$. On the other hand, we have $k \leq \frac{n}{c_d-1} + 1$, and therefore $n \geq (k-1)(c_d-1)$. Note that $c_d \geq 10/4$ and $k \geq 10$ imply that $kc_d/2 \geq k + c_d$.

Consider the system of boxes

$$\mathcal{R}' := \mathcal{R}'\left(\emptyset, \frac{1}{k+1}, \frac{1}{4k^2}; d\right).$$

Observe that the preconditions for Lemma 6.8 are trivially met; and so the system

$$\mathcal{R} := \mathcal{R}\left(\emptyset, \frac{1}{k+1}, \frac{1}{4k^2}; d\right)$$

exists. Note that $\nu(\mathcal{R}) = k$. Since

$$A_d(n) \geq \frac{c_d - 1}{n} > \frac{1}{k} > \frac{1}{k+1} + \frac{2}{4k^2} \quad \text{(for } k \geq 10\text{)},$$

Lemma 6.10 yields

$$\begin{aligned}\tau(\mathcal{R}) &= \tau\left(\mathcal{R}\left(\emptyset, \frac{1}{k+1}, \frac{1}{4k^2}; d\right)\right) \\ &\geq n + 1 \geq (k-1)(c_d - 1) + 1 \\ &= kc_d - (k + c_d) + 2 > \frac{kc_d}{2}.\end{aligned}$$

Consequently, we have $\tau(\mathcal{R}) \geq \nu(\mathcal{R})c_d/2$, or $\tau(\mathcal{R})/\nu(\mathcal{R}) \geq c_d/2$, as required. $\square$

Bibliography

[1] R. Ahlswede and I. Karapetyan, Intersection graphs of rectangles and segments, in *General Theory of Information Transfer and Combinatorics*, Ahlswede R. et al. (editors), LNCS vol. 4123. Springer, Berlin, Heidelberg, 2006, pp. 1064–1065.

[2] C. Aistleitner, A. Hinrichs, and D. Rudolf, On the size of the largest empty box amidst a point set, *Discrete Applied Mathematics* **230** (2017), 146–150.

[3] N. Alon and D.J. Kleitman, Piercing convex sets and the Hadwiger-Debrunner (p, q)-problem, *Advances in Mathematics* **96(1)** (1992), 103 – 112.

[4] B. Aronov, E. Ezra, and M. Sharir, Small-Size ε-nets for axis-parallel rectangles and boxes, *SIAM Journal on Computing* **39(7)** (2010), 3248–3282.

[5] P. Braß, W. Moser, and J. Pach, *Research Problems in Discrete Geometry*, Springer, New York, 2005.

[6] P. Chalermsook and J. Chuzhoy, Maximum independent set of rectangles, *Proc. 20th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA 2009)*, pp. 892–901.

[7] T. M. Chan and S. Har-Peled, Approximation algorithms for maximum independent set of pseudo-disks, *Discrete & Computational Geometry* **48(2)** (2012), 373–392.

[8] M. Chudnovsky, S. Spirkl, and S. Zerbib, Piercing axis-parallel boxes, *Electr. J. Comb.* **25(1)** (2018), #P1.70.

[9] J. R. Correa, L. Feuilloley, P. Pérez-Lantero, and J. A. Soto, Independent and hitting sets of rectangles intersecting a diagonal line: algorithms and complexity, *Discrete & Computational Geometry* **53(2)** (2015), 344–365.

[10] L. Danzer, Zur Lösung des Gallaischen Problems über Kreisscheiben in der Euklidischen Ebene, *Studia Sci. Math. Hungar.* **21(1-2)** (1986), 111–134.

[11] L. Danzer and B. Grünbaum, Intersection properties of boxes in $\mathbb{R}^d$, *Combinatorica* **2(3)** (1982), 237–246.

[12] V. L. Dol'nikov, A coloring problem, *Siberian Mathematical Journal* **13** (1972), 886–894. Translation of *Sibirsk Math. Zh.* **13** (1972), 1272–1283.

[13] A. Dumitrescu and M. Jiang, Piercing translates and homothets of a convex body, *Algorithmica* **61(1)** (2011), 94–115.

[14] A. Dumitrescu and M. Jiang, On the largest empty axis-parallel box amidst n points, *Algorithmica* **66(2)** (2013), 225–248.

[15] A. Dumitrescu and M. Jiang, Computational Geometry Column 69, *SIGACT News Bulletin* **50(3)** (2019), 75–90.

[16] J. Eckhoff, A survey of the Hadwiger–Debrunner (p, q)-problem, *Discrete & Computational Geometry*, 347–377, Algorithms and Combinatorics, 25, Springer, Berlin (2003).

[17] D. Fon-Der-Flaass and A. V. Kostochka, Covering boxes by points, *Discrete Mathematics* **120(1-3)** (1993), 269–275.

[18] S. Govindarajan and G. Nivasch, A variant of the Hadwiger-Debrunner (p, q)-problem in the plane, *Discrete & Computational Geometry* **54(3)** (2015), 637–646.

[19] A. Gyárfás and J. Lehel, Covering and coloring problems for relatives of intervals, *Discrete Mathematics* **55(2)** (1985), 167–180.

[20] H. Hadwiger and H. Debrunner, Über eine Variante zum Hellyschen Satz, *Archiv der Mathematik* (Basel) **8(4)** (1957), 309–313.

[21] H. Hadwiger and H. Debrunner, *Combinatorial Geometry in the Plane* (English translation by Victor Klee), Holt, Rinehart and Winston, New York, 1964.

[22] E. Helly, Über Mengen konvexer Körper mit gemeinschaftlichen Punkten, *Jahresbericht der Deutschen Mathematiker-Vereinigung* **32** (1923), 175–176.

[23] A. Holmsen and R. Wenger, Helly-type theorems and geometric transversals, in *Handbook of Discrete and Computational Geometry* (J. E. Goodman, J. O'Rourke, and C. D. Tóth, editors), pp. 91–123, 3rd edition, CRC Press, Boca Raton, 2017.

[24] T. Kaiser and Y. Rabinovich, Intersection properties of families of convex (n, d)-bodies, *Discrete & Computational Geometry* **21(2)** (1999), 275–287.

[25] R. N. Karasev, Piercing families of convex sets with the d-intersection property in $\mathbb{R}^d$, *Discrete & Computational Geometry* **39(4)** (2008), 766–777.

[26] G. Károlyi, On point covers of parallel rectangles, *Period. Math. Hungar.* **23(2)** (1991), 105–107.

[27] G. Károlyi and G. Tardos, On point covers of multiple intervals and axis-parallel rectangles, *Combinatorica* **16(2)** (1996), 213–222.

[28] D. J. Kleitman, A. Gyárfás, and G. Tóth, Convex sets in the plane with three of every four meeting, *Combinatorica* **21(2)** (2001), 221–232.

[29] D. Larman, J. Matoušek, J. Pach, and J. Törőcsik, A Ramsey-type result for convex sets, *Bulletin of the London Mathematical Society* **26(2)** (1994), 132–136.

[30] G. Rote and R. F. Tichy, Quasi-Monte-Carlo methods and the dispersion of point sequences, *Mathematical and Computer Modelling*, **23** (1996), 9–23.

[31] F. E. Su and S. Zerbib, Piercing numbers in approval voting, *Mathematical Social Sciences* **101** (2019), 65–71.

[32] G. Wegner, Über eine kombinatorisch-geometrische Frage von Hadwiger und Debrunner, *Israel J. Math.* **3** (1965), 187–198.

[33] G. Wegner, Anmerkungen zu 'Über eine kombinatorisch-geometrische Frage von Hadwiger und Debrunner', Unpublished notes (4 pages), Göttingen, 1968.

Curriculum Vitae

Ke Chen

Education

- B.S. in *Computer Science*, Southeast University, 2011
- M.S. in *Computer Science*, University of Wisconsin-Milwaukee, 2014
- Ph.D, University of Wisconsin-Milwaukee, 2021

Dissertation Title

Algorithmic and Combinatorial Results in Selection and Computational Geometry

Publications

1. K. Chen and A. Dumitrescu, Nonconvex cases for carpenter's rulers, *Theoretical Computer Science*, 2015. Special issue with invited papers from the FUN 2014 conference. A preliminary version in *Proceedings of the 7th International Conference on Fun with Algorithms* (FUN 2014), Lipari Island, Sicily, Italy, July 2014; Vol. 8496 of LNCS, pp. 89–99.

2. K. Chen and A. Dumitrescu, Selection algorithms with small groups, *International Journal of Foundations of Computer Science*, Vol. 31, No. 3 (2020), 355–369. A preliminary version in *Proceedings of the 29th International Workshop on Algorithms and Data Structures* (WADS 2015), Victoria, Canada, August 2015; LNCS 9214, Springer, 2015, pp. 189–199.

3. K. Chen and A. Dumitrescu, On the longest spanning tree with neighborhoods, *Discrete Mathematics, Algorithms and Applications*, Vol. 12, No. 5 (2020). A preliminary version in *Proceedings of the 12th International Frontiers of Algorithmics Workshop* (FAW 2018), Guangzhou, China, May 2018; in LNCS.

4. K. Chen, A. Dumitrescu, W. Mulzer, and Cs. D. Tóth, On the Stretch factor of polygonal chains, *SIAM Journal on Discrete Mathematics*, submitted. A preliminary version in *Proceedings of the 44th International Symposium on Mathematical Foundations of Computer Science* (MFCS 2019), Aachen, Germany, August 2019; LIPIcs series, Schloss Dagstuhl.

5. K. Chen and A. Dumitrescu, On Wegner's conjecture for axis-parallel rectangles, *Discrete Mathematics*, Vol. 343, Issue 12 (2020) 112091.

6. K. Chen and A. Dumitrescu, Multiparty selection, *Proceedings of the 31st International Symposium on Algorithms and Computation* (ISAAC 2020), Online, December 2020, to appear.