Average-case analysis for combinatorial problems

A Thesis
Presented to
The Academic Faculty

by

Abraham D. Flaxman

In Partial Fulfillment
of the Requirements for the Degree
Doctor of Philosophy

Department of Mathematical Sciences
Carnegie Mellon University
May 2006

Abstract

This thesis considers the average case analysis of algorithms, focusing primarily on NP-hard combinatorial optimization problems. It includes a catalog of distributions frequently used in average-case analysis and a collection of mathematical tools that have been useful in studying these distributions. The bulk of the thesis consists of case-studies in average-case analysis of algorithms. Algorithms for 3-SAT, Subset Sum, Strong Connectivity, Stochastic Minimum Spanning Tree, and Uncapacitated Facility Location are analyzed on random instances.

Acknowledgements

There are so many people to acknowledge and thank that I have trouble attempting this section.

First, I must sincerely thank my advisor Alan Frieze. It is said over and over and over again, all that matters to the graduate school experience is your advisor. I could not have hoped for anything better than the advising I received from Alan. From his encyclopedaic knowledge of probabilistic combinatorics to his uncanny ability to identify *tractable* interesting research problems, Alan has been a inspiring role model.

I also owe a great debt to the ACO and Theory community at Carnegie Mellon University, including the students, the faculty, and the staff. Juan Carlos Vera, Tom Bohman, Alan Frieze, R. Ravi, Avrim and Manuel Blum, Luis von Ahn, Maverick Woo, Bartosz Przydatek, Cliff Smyth, Konstantin Andreev, Anupam Gupta, Ryan Williams, Mor Harchol-Balter, Gary Miller, Steven Rudich, Danny Sleator, Reha Tutuncu, Jochen Konemann, Ojas Parekh, Ke Yang, Amitabh Sinha, Nina Balcan, and Javier Pena just begins listing the folks that I have had many happy chats with about research and otherwise.

Special thanks are due to Santosh Vempala, who recommended I go to CMU for grad school in the first place. And for that matter, thanks also to David Kelly, who showed me what real math was about to begin with.

There are a number of researchers outside the CMU circle that I met during my summer travels whom I would like to thank as well. Thanks to the folks I worked with at IBM, especially Greg Sorkin, David Gamarnik, Don Coppersmith, John Lee, and Marcos Goycoolea. Thanks to Adam Kalai, Eric Vigoda, David McAllester, and Lance Fortnow, and the others who shared their minds with me at TTI-C. Thanks to everyone from MSR-SVC, especially Jason Hartline and Andrew Goldberg.

I would also like to thank my family and the friends I have not mentioned already. Vaugely chronologically, big thanks to Isaac, Trevor, Helen, Nicole, Jen, Betty, Ian, Steven, Ksenjia, Chad, Chuck, Larry, Devon, Laura, Lauren, John, Sarah, Daniel, Paul, Bin, Andy, Joel, Katie, Dug, Matt, and all from Rustbelt Radio and the Big Idea Bookstore. Over the years, you have all been there to share a story or a drink, go see a movie or have dinner, to take part in the little day-to-day things that end up being everything.

I'd like to thank most-of-all my girlfriend Jessi. For our discussions of everything, for eating and cooking together, for giving me attention, and for leaving me alone. Thank you for sharing life with me.

Table of Contents

List of Figures

Chapter 1

Introduction

This thesis deals with the average-case analysis of algorithms for combinatorial search and optimization problems.

The phrases *combinatorial search* and *combinatorial optimization* are used here to refer to a wide range of problems. Some examples of combinatorial search and optimization problems are: finding a minimum weight spanning tree, finding a minimum weight traveling salesperson tour, finding a subset of a given set of integers that sums to a given number, and finding a satisfying assignment to a Boolean formula.

The development of algorithms for combinatorial search and optimization problems has been underway for the last century. However, *algorithm* is a somewhat informal term. Is exhaustive search an algorithm? Should a procedure that fails or finds the wrong answer on some inputs be considered an algorithm? For the purposes of this thesis, an algorithm is a systematic procedure for solving some problem. It is something specific enough to be translated into a computer language or a Turing machine. It doesn't have to be correct on all inputs, and it doesn't have to run fast enough to find an answer. But it is good if it does.

1.1 When good algorithms exist

This thesis is a small piece of the theoretical study of when good algorithms exist. The most popular formalization is in terms of recognizing sets with a polynomial-time Turing machine, and the famous question of **P** vs. **NP**. Though this beautiful formalism is simple enough to teach in an undergraduate class and so difficult to solve that most proof techniques have been proven not to work, it is not clear that it captures the true nature of "when good algorithms exist".

Objections are plentiful. Suppose a problem, like estimating the volume of a convex body within a factor of $(1+\epsilon)$ is possible in polynomial time, but the polynomial grows like n^{23}. Is it really fair to consider this a good algorithm? Or, what if the big-$\mathcal{O}$ notation hides a constant so large that even a linear-time algorithm cannot be run? Worst-case complexity theory can be overly optimistic.

On the other hand, it is possible that an algorithm for a particular problem takes

exponential time on some instances, like the simplex algorithm for linear programming, but these instances are so infrequent that, in practice, the algorithm always succeeds with an acceptably short running time. Worst-case complexity theory can be overly pessimistic as well.

This thesis considers an alternative approach, which does not seek to replace worst-case complexity theory, but to offer another perspective, one which should work in conjunction with worst-case complexity theory and empirical study of algorithmic performance to predict the difficulty of computational problems. The approach has come to be known as average-case analysis of algorithms (or, alternatively, probabilistic analysis of algorithms). Average-case analysis of algorithms is largely a way to avoid some of the pessimistic predictions of complexity theory.

1.2 Central hypothesis of average-case analysis

The central hypothesis of average-case analysis of algorithms could be summarized as

"In the real world, problem instances incorporate elements of chance, so an algorithm need not be good for all problem instances, as long as it is likely to work on the instances which show up."

The key difficulty in developing algorithms based on this hypothesis is imprecise nature of *how* problem instances incorporate randomness in the real world. Historically, this hasn't stopped researchers from picking some particular distribution and performing an average-case analysis of some algorithm with that distribution. Much of the work on average-case analysis developed from random graph theory, which, in turn, emerged from the probabilistic method. The earliest applications of the probabilistic method were purely mathematical questions and so there was no reason to be concerned with "the real world" as referred to above; to prove that there is a graph which simultaneously has large girth and large chromatic number, you may choose *any* random distribution over graphs and show that the probability these properties hold is nonzero with respect to *that* distribution.

However, it is tempting to make predictions about algorithmic performance on real-world networks based on the theory of random graphs. Some of the first research to do so explicitly appeared in the 1970s. For example, algorithms for several problems on unweighted graphs were analyzed on graphs drawn from the Erdős-Rényi distributions (defined in Section 2.1.1) by Grimmett and McDiarmid [148], Karp [164], Angluin and Valiant [20], and DeWitt and Krieger [97, 98].

1.3 Current directions

In the past 10 years, the search for appropriate distributions has been influenced by several developments.

1.3.1 Small worlds and power laws

The growth of the Internet and the prevalence of large networks therein has resulted in a large collection of naturally occurring networks which are easy for researchers to obtain and compare with graphs generated by various random distributions. This has led to a wide variety of alternative models. Some of the most studied distributions were designed to have small dense subgraphs or power-law degree distributions (described in Section 2.1.2). Both of these properties have been observed in the real world, but are not likely to appear under the Erdős-Rényi distributions. These distributions have been developed largely in isolation from average-case analysis of algorithms, and it is remains to be determined if they will yield useful predictions about designing heuristics. Some steps in this direction are the analysis of algorithms web-graph crawling [89] and trawling [180], locating secret societies in a power-law graph [88], and algorithms for finding short paths with only local information [168, 7].

1.3.2 Smoothed analysis and semi-random instances

Smoothed analysis and semi-random instances provide an alternative approach to the distribution quandary. Instead of trying to tune distributions to more accurately reflect real instances, this approach provides a way to study the performance of an algorithm on a large collection of distributions simultaneously. Smoothed analysis and semi-random instances can be described in terms of an adversary generating instances in the presence of noise, which makes the similarity and difference between smoothed analysis and semi-random instances clear; smoothed analysis corresponds to an oblivious adversary, while in semi-random instances, the adversary may use an adaptive strategy [39, 213, 121].

1.3.3 Searching for hard instances and planted instances

A completely different motivation drives the search for difficult distributions, which are especially useful if you can generate hard problems to which you know the answer. Such a distribution serves as a one-way function, which is a basic cryptographic primitive [152]. In the case of random instances of 3-SAT, just knowing that there is no efficient algorithm to refute the satisfiability of a randomly generated problem would have powerful applications to worst-case complexity theory [111].

1.4 Outline of what follows

The remainder of this thesis divides roughly into 3 parts. Chapter 2 is an extensive catalog of the distributions which have been used in average-case analysis of algorithms. Many of the distributions come from the theory of random graphs, which is also the area where most of the proof techniques used in average-case analysis developed. Some of the other distribtions were developed in response to observations of real data, in an attempt to capture aspects of the data in a model simple enough to analyze theoretically. There is an element of artistry to selecting a distribution for an average case analysis. Choose right and the

calculations will be a breeze, but make just a small change and you will be buried under a pile of binomial coefficients (or derive theoretical conclusions which have no bearing on reality).

There are many surveys and reference works already available about these distributions, but none of the existing work concisely subsumes the material in Chapter 2. Some of these alternative references are the books on random graphs [46, 216, 156, 220, 196] and the surveys [132, 137, 175, 65], as well as work focusing on modelling real-world graphs [225, 149, 24, 54, 202, 103, 56].

Chapter 3 contains a smattering of useful tools for proving theoretical results about these distributions. These tools all appear in many other places and in many different levels of detail. They are collected here for easy reference.

The remaining chapters provide a number of case studies in average-case analysis of algorithms, including: Subset sum with n random integers each of $(\log n)^2$ bits, Planted 3-SAT, smoothed connectivity, facility location, 2-stage spanning tree. Additional details about the algorithms and the combinatorial search and optimization problems that they address will be reserved for the appropriate chapters, so that each chapter can be read (nearly) in isolation.

All these case studies have appeared previously as conference and/or journal papers. They provide some examples of the sort of "theoretical experimental" evidence that average-case analysis can provide towards a theory of "when good algorithms exist".

1.5 Reference of mathematical notation

This section collects the mathematical notation and conventions that will be used repeatedly throughout this thesis.

1.5.1 Asymptotic analysis

Throughout this document, the variable n will be the central parameter for instance size and running time. The frequently used asymptotic notation will always be in terms of n, so

$f(n) = \mathcal{O}(g(n))$ means that there exists n_0 and C with $f(n) \leq Cg(n)$ for all $n \geq n_0$,
$f(n) = o(g(n))$ means that $\lim_{n\to\infty} f(n)/g(n) = 0$,

$f(n) = \Omega(g(n))$ means that there exists n_0 and C with $f(n) \geq Cg(n)$ for all $n \geq n_0$,
$f(n) = \omega(g(n))$ means that $\lim_{n\to\infty} f(n)/g(n) = \infty$.

As another useful short-hand notation, a sequence of events $\mathcal{E}_1, \mathcal{E}_2, \ldots$ holds *with high probability* (which is abbreviated **whp**) if $\mathbb{P}[\mathcal{E}_n] = o(1)$. The sequence of events holds *quite surely* (abbreviated **qs**) if $\mathbb{P}[\mathcal{E}_n] = o(n^{-k})$ for any integer k.

1.5.2 Discrete probability

For random variables and distributions, it will be convenient to use "blackboard" notation as follows: if $\mathbb{D}$ is a distribution then write $Z \sim \mathbb{D}$ to mean that Z is a random variable distributed according to $\mathbb{D}$.

Some common distributions of random variables are the following:

- The Bernoulli distribution, $\mathrm{Be}(p)$, corresponds the outcome of a single toss of a biased coin which is heads with probability p and tails otherwise. $Z \sim \mathrm{Be}(p)$ means that $\mathbb{P}[Z = 1] = p$ and $\mathbb{P}[Z = 0] = 1 - p$.

- The binomial distribution, $\mathrm{Bi}(n, p)$, corresponds to the number of heads the occurs in n tosses of a biased coin that comes up heads with probability p. Formally, $Z \sim \mathrm{Bi}(n, p)$ means that
$$\mathbb{P}[Z = k] = \binom{n}{k} p^k (1 - p)^{n-k}$$

- The uniform distribution $\mathcal{U}[0, 1]$ intuitively should have $Z \sim \mathcal{U}[0, 1]$ mean that Z is equally likely to be any real between 0 and 1. But that is hopelessly non-rigorous, so formally (without getting any messy measure theory involved), $Z \sim \mathcal{U}[0, 1]$ means that for any $0 \leq a \leq b \leq 1$,
$$\mathbb{P}[Z \in [a, b]] = b - a$$

- The exponential distribution, $\mathrm{Exp}(\lambda)$, is the unique distribution that is continuous and memoryless. Formally, $Z \sim \mathrm{Exp}(\lambda)$ means that for any $t \geq 0$,
$$\mathbb{P}[Z \geq t] = e^{-\lambda t}.$$
More important that the formal definition is the fact that an exponentially distributed random variable is "memoryless",
$$\mathbb{P}[Z \geq t + s \mid Z \geq t] = \mathbb{P}[Z \geq s]$$

- The Normal distribution, $\mathcal{N}(\mu, \sigma^2)$, is a distribution the appears as a limit of the binomial distribution for many parameters, and also as the limiting sum of independent random variables that satisfy certain requirements. Formally, $Z \sim \mathcal{N}(\mu, \sigma^2)$ means that
$$\mathbb{P}[Z \leq t] = \frac{1}{\sqrt{2\pi\sigma^2}} \int_{-\infty}^{t} e^{-(\tau-\mu)^2/(2\sigma^2)} d\tau.$$

1.5.3 Combinatorics and graph theory

- For a finite set S, let $|S|$ denote the number of elements in S.

- The "square-bracket" set notation $[k] = \{1, 2, \ldots, k\}$ is a convenient shorthand that is common in combinatorics but not as common in other areas of mathematics.

- A *graph*, usually denoted by G, consists of a set of *vertices*, usually denoted V, and a set of *edges*, usually denoted E, where $e \in E$ is a set of 2 vertices, while a directed graph (or *digraph*) will usually be denoted D, and will consist of a set of *nodes*, usually denoted N, and a set of *arcs*, usually denoted A. Sometimes, when the edge set of a graph G has not been given a specific name, it will be convenient to denote it by $E(G)$.
- As mentioned above, all asymptotic notation is in terms of n. When dealing with graph problems, as often as is possible, n will denote the number of vertices in the graph, and m will be the number of edges.

There are many good textbooks and reference works on graph theory, and a few of these are by West [227], Bollobás [45] and Diestel [102].

Chapter 2

Distributions for average-case analysis of algorithms

This chapter contains descriptions of some distributions which have been employed in average-case analysis of combinatorial search and optimization problems. It is not meant to be exhaustive catalog. It is meant to collect in one place some interesting distributions, using somewhat uniform terminology and a level of rigor that is enough, but not too much.

2.1 Unweighted graphs and the like

2.1.1 The Erdős-Rényi graphs

The binomial random graph, $\mathbb{G}_{n,p}$

For historical reasons, the most extensively studied distribution for random graphs is $\mathbb{G}_{n,p}$, which is called *the binomial random graph* or the *Erdős-Rényi graph*. Informally, if $G \sim \mathbb{G}_{n,p}$, then G is a random graph with n vertices, where each pair of vertices appears as an edge independently with probability p. This can be defined formally by the following: $G \sim \mathbb{G}_{n,p}$ means that for any n-graph $G^\star = ([n], E)$,

$$\mathbb{P}[G = G^\star] = p^{|E|}(1-p)^{\binom{n}{2}-|E|}.$$

However, it is often more convenient to think of $G \sim \mathbb{G}_{n,p}$ as an n-graph where each edge appears independently with probability p. Because of the independence of candidate edges, this distribution is particularly amenable to analysis by the method of deferred decisions.

According to Bollobás, the study of these graph was initiated by Erdős and Rényi in 1959 [106], and while other research [126, 140, 21] is similar in some ways, to attribute the theoretical interest in this distribution to all of these researchers is to "misconceive the nature of the subject. Only Erdős and Rényi introduced the methods which underlie the probabilistic treatment of random graphs. The other authors were all concerned with enumeration problems and their techniques were essentially deterministic." [46, p. xii]

This distribution is the central to both of the excellent random graphs textbooks [46, 156]

The random graph with m edges, $\mathbb{G}_{n,m}$

The distribution $\mathbb{G}_{n,m}$ was defined and studies by Erdős and Rényi at the same time as $\mathbb{G}_{n,p}$. $\mathbb{G}_{n,m}$ is the uniform distribution over graphs with n vertices and m edges. So, if $G \sim G_{n,m}$, then for $G^\star = ([n], E)$ with $|E| = m$,

$$\mathbb{P}[G = G^\star] = \binom{\binom{n}{2}}{m}^{-1}.$$

Sometimes a calculation will be easiest using the $\mathbb{G}_{n,m}$ distribution, and sometimes it is more convenient to work with $\mathbb{G}_{n,p}$ with p chosen appropriately to make the expected number of edges equal to m. A rule of thumb is that, for reasonable values of p, everything interesting about $\mathbb{G}_{n,m}$ is the same in $G_{n,p}$ for $p = 2m/n^2$. (For a formal version of this rule of thumb, see Bollobás [46, Theorem 2.2, p. 37–38]).

Some comments on $\mathbb{G}_{n,p}$ and $\mathbb{G}_{n,m}$

Since average-case analysis of algorithms focuses on asymptotic algorithm behavior, it is usually appropriate to think of n growing large while p and m grow as carefully chosen functions of n.

An important phenomenon in $\mathbb{G}_{n,p}$ and $\mathbb{G}_{n,m}$ is the existence of sharp thresholds for graph properties. As a concrete example, consider the property of being connected. For $G \sim \mathbb{G}_{n,p}$, Erdős and Rényi showed that for any $\epsilon > 0$,

$$\mathbb{P}[G \text{ is connected}] = \begin{cases} o(1), & \text{if } np \leq (1-\epsilon)\log n; \\ 1 - o(1), & \text{if } np \geq (1+\epsilon)\log n. \end{cases}$$

The criteria for the existence of sharp threshold functions due to Friedgut [127, 128] informally states

All monotone graph properties which do not have a sharp threshold may be approximated by a local property
(for example, containing a triangle as a subgraph.)

It remains an intriguing open problem to show that this threshold function converges to a threshold value.

It is possible that there is some mysterious relationship between thresholds and computation time, about which there will be more to say later in this section.

The random multigraphs with m edges, $\tilde{\mathbb{G}}_{n,m}$

It can sometimes simplify computation to consider the following distribution over random multigraph. $G \sim \tilde{\mathbb{G}}_{n,m}$ is the random graph formed by choosing edges m times, where each edge is chosen uniformly at random from the $\binom{n}{2}$ vertex pairs. For many reasonable values of m, results for this distribution can be translated to $\mathbb{G}_{n,m}$ simply by showing that the

probability of having no repeated edges is at least a constant. One of the first papers to explicitly define this distribution is [50].

The random regular graph, $\mathbb{G}_{n,r}$

The random r-regular graph $\mathbb{G}_{n,r}$ is a distribution that is similar in spirit to $\mathbb{G}_{n,m}$. To formally define the distribution $\mathbb{G}_{n,r}$, first let $\mathcal{G}_{n,r}$ denote the collection of r-regular n-graphs. Then, for $G \sim \mathbb{G}_{n,r}$ and $G^\star \in \mathcal{G}_{n,r}$

$$\mathbb{P}[G = G^\star] = |\mathcal{G}_{n,r}|^{-1}.$$

However, this is not a particularly convenient formula for studying properties of random regular graphs, and it is almost always easier to work with the configuration model of Bollobás [43] (which was originally devised as a method to bound $|\mathcal{G}_{n,r}|$). In the configuration model, every vertex is associated with r "clones" and the rn clones are paired up by a uniformly random matching. Then the r clones associated with each vertex are contracted into a single vertex, which yields a multigraph that may include self-loops. If this multigraph happens to have no repeated edges and also no self-loops, then it is identically distributed with $\mathbb{G}_{n,r}$. When r is a constant, the probability that the configuration model yields a simple graph exceeds a constant value for all n. For $\mathbb{G}_{n,r}$ with r large enough, it is possible to couple this graph closely with $\mathbb{G}_{n,p}$ for appropriate values of p [166].

Non-regular graphs with a given degree sequence can also be generated easily with the configuration model, and they have been the area of some research recently, including a result of Molloy and Reed giving a criteria for the emergence of the giant component [195]. This relates to the recent interest in studying random graphs which are likely to exhibit a power-law degree sequence. (There is more about distributions for power-law graphs in section 2.1.2.)

Random planar graphs, etc.

In a manner analogous to the formal definition of random regular graphs above, it is possible to define random graphs from any other family of special graphs. However, without the configuration model, these distributions are very difficult to work with. For example, a distribution over random planar graphs on n vertices can be defined by letting $\mathcal{P}$ denote the set of all planar graphs n vertices, and taking $\mathbb{P}[G = G^\star] = |\mathcal{P}|^{-1}$ for any $G^\star \in \mathcal{P}$.

Directed graphs

All these distributions have analogous directed versions, and the directed Erdős-Rényi graph, $\mathbb{D}_{n,p}$ is prominent enough to warrant specific mention. As you can guess, the formal definition is as follows: for $D \sim \mathbb{D}_{n,p}$, any digraph $D^\star = ([n], A)$ has

$$\mathbb{P}[D = D^\star] = p^{|A|}(1-p)^{n(n-1)-|A|}.$$

Similarly to the case with $\mathbb{G}_{n,p}$, it is often more convenient to view $D \sim \mathbb{D}_{n,p}$ as a digraph where each pair of nodes appears as an arc independently with probability p. Many results for random graphs carry over directly to random digraphs. An important technique developed by Karp describes the component structure of a random graph by comparing a breadth-first search of the graph to a branching process, and this technique was first applied to study the strongly connected component of a random digraph [165].

Related random graphs

Some other distributions that are related to these random graphs and worth mentioning are the random k-out, and the random ℓ-in-k-out. In the random k-out, every vertex chooses k neighbors and adds directed arcs to them (usually with replacement). In the ℓ-in-k-out, every vertex chooses k neighbors to point at, and ℓ neighbors to be point at by. These directed random graphs, and the undirected graphs formed by ignoring the directions of the arcs are often useful in studying the minimum obstructions to graph properties.

For example, the minimum conditions necessary for a random graph to contain a Hamiltonian cycle have driven algorithmic analysis on such graphs, most recently leading to the proof that a 3-out is Hamiltonian **whp** [42].

Random subgraph of a given graph

One direction of generalization of the Erdős-Rényi graph distributions is to consider a random subgraph of a given graph. The distribution $\mathbb{G}_{n,p}$ can be interpreted as starting with the complete graph and allowing each edge to remain independently with probability p. A natural generalization of this interpretation is to start with some other graph besides the complete graph. A complete bipartite graph K_{n_1,n_2}, the edge-adjacency graph of a d-dimensional hypercube, and an (n, d, λ)-expander graph all make good choices. Bipartite random graph appears frequently in average-case analysis, such as the analysis of matching algorithm [199, 29]. In the case of pseudo-random graphs, it is sometimes not necessary to know the graph exactly and just knowing that it is an expander is enough [131].

Random lifts of graphs

Another way to generate a random graph based on a given graph is the random lift. Here a base graph G is lifted to a random graph $\tilde{G}$ by replacing every vertex v of G with n vertices $v_1, \ldots, v_n$, and turning each edge $uv \in G$ into a random matching between the u_i's and the v_i's. The random lift of a given graph has been proposed both as a new method of constructing exotic graphs for the probabilistic method, and as a technique for adapting algorithms that work on Erdős-Rényi graphs to work on worst-case instances [19, 18, 183, 64].

Hypergraphs

It is easy to imagine distributions on hypergraphs analogous to all the graph distributions above. The most common distribution is $\mathbb{H}^k_{n,p}$, the random k-uniform hypergraphs,

where every k-set appears as a hyperedge independently with probability $p = p(n)$. Another important hyper-multigraph is $\tilde{\mathbb{H}}^k_{n,m}$, where m hyperedges are selected randomly with replacement from $\binom{[n]}{k}$.

The $\mathbb{H}^k_{n,p}$ distribution has been used to analyze the performance of hypergraph coloring algorithms by a number of researchers [208, 207, 3], and the sharp threshold for 2-colorability is known quite precisely for sufficiently large k [4]. There is also a survey paper which focuses on properties of random hypergraphs, [163].

k-SAT and other constraint satisfaction problems

From $\tilde{\mathbb{H}}^k_{n,m}$, it is a small step to an interesting distribution over Boolean formulas, which is called $\mathbb{I}^k_{n,m}$ in this thesis. This is a distribution over boolean formulas that can be expressed as k-CNF formulas (which is to say, the formula can be written as an "AND of ORs", where each OR clause contains k literals). A realization of $\tilde{\mathbb{I}}^k_{n,m}$ is generated by taking $\mathbb{H}^k_{n,m}$ and associating every vertex of the hypergraph with a variable, and then choosing to negate or not negate each variable independently with probability 1/2.

Here is where the mysterious relationship between threshold phenomenon and computational difficulty first observed by Cheeseman, Kanefsky, Taylor [73] has been explored in great detail. As mentioned above, it has been known since the pioneering work of Erdős and Rényi that certain properties of random graphs exhibit threshold phenomena. What Cheeseman, Kanefsky, and Taylor observed is that the running time of specific algorithms for testing these properties seem to take the longest when faced with structures at the threshold. For example, computer experiments suggest that $\mathbb{I}_{n,m}$ has a satisfiability threshold at about $m = 4.26n$, meaning for $m = (4.26 - \epsilon)n$ the random instance is likely to be satisfiable, while for $m = (4.26 + \epsilon)n$ the random instance is likely to be unsatisfiable [95, 210]. (It is worth mentioning here that the number 4.26 is not likely to be exact, and even the existence of a threshold constant is currently unproven. However, Friedgut's work, as mentioned above, has shown that there is a threshold *function* which may depend on n for which taking $m = (1 - \epsilon)c(n)n$ results in a satisfiable formula **whp** while taking $m = (1 + \epsilon)c(n)n$ results in an unsatisfiable instance **whp** [127].) These experiments also found that the computation time required to determine if the random instances were satisfiable followed an "easy-hard-easy" pattern, with the longest running time coinciding very closely with the satisfiability threshold. Subsequent experiments indicate that this correlation does not occur for some algorithms other than the DPLL-based ones used in the original experiments [83].

Planted distributions

Planted distributions are a common modification of the distributions above which are often considered in average-case analysis of algorithms for combinatorial search problems. Planted distributions generate problem instances together with a solution. One motivation for studying these distributions is that if a planted distribution has no good algorithms, then the instance-solution pairs constitute a one-way function, which is a basic cryptographic primitive.

For example random graph with planted clique can be constructed by taking $G \sim \mathbb{G}_{n,1/2}$ and then adding every edge between the vertices in a vertex set of size k. If $k = \Omega\left(n^{1/2}\right)$ then a polynomial-time algorithms are known which will recover the planted clique **whp** [16, 113] (the paper of Feige and Krauthgamer shows that an algorithm is likely to succeed even in the "semi-random" model, which is described in Section 2.3.) On the other hand, if $k = (1+\epsilon)\log_2 n$ for ϵ sufficiently small, then the planted clique is smaller than the largest clique **whp**, and [160] shows that if a polynomial-time algorithm succeeds in finding the planted clique, then it will also succeed in finding a clique of size $(1+\epsilon)\log_2 n$ under the non-planted distribution, $\mathbb{G}_{n,1/2}$.

In the same spirit, it possible to construct random 3-colorable graphs [15], 2-colorable hypergraphs [74], and satisfiable formulas [173, 117] and more complicated satisfiable constraint satisfaction problems [228]. Chapter 5 constitutes a detailed example of the average-case analysis of a heuristic for 3-SAT on a planted random instance, which originally appeared in [117].

One of the most popular planted distributions for average-case analysis is the planted cut. A planted cut is formed by partitioning the vertices into 2 or more parts $V_1, V_2, \ldots$ and then including each candidate edge between V_i and V_j independently with probability p_{ij} [57, 158, 86, 191, 67, 55, 84].

2.1.2 Alternatives to the Erdős-Rényi graphs

Geometric random graphs

The distribution $\mathbb{G}(\mathcal{X}; r)$ of geometric random graph has a history in percolation theory, but is of increasing importance in the study of sensor networks. To describe the distribution of $G \sim \mathbb{G}(\mathcal{X}; r)$, first take n random points $X_i = (a_i, b_i)$, $i = 1, 2, \ldots, n$ uniformly in the unit square $[0,1]^2$. Then let $G = (V, E)$ where $V = \{X_1, \ldots, X_n\}$ and $E = \{(X_i, X_j) \big| \, \|X_i - X_j\| \leq r\}$.

Dealing with $\mathbb{G}(\mathcal{X}; r)$ rigorously requires more care than the graphs in the previous section because of the continuous random variables involved in the definition.

It is discussed in [147] (which details the connection between this finite graph and "continuum percolation"), and it is the central object of study in [101, 204]. Some useful tools for analyzing the behavior of geometric random graphs are the bin covering technique of Muthukrishnan and Pandurangan [200] and the application of minimum bottleneck matching developed in [141].

Of course, it is easy to extend the definition to random points chosen in other metric spaces, including using a more complicated subset of the plane than the unit square or sprinkling the points in a subset of $\mathbb{R}^3$ instead of $\mathbb{R}^2$.

A very closely related random structure is the weighted graph where instead of adding edges between vertices that are sufficiently close together, every edge is included but each edge ij has a weight that depends on the distance between X_i and X_j. This weighted random graph will the subject of Section 2.2.2 discusses this weighted random graph further. Also, instead of including edges between points within a critical radius, the edges can be included randomly with probability given by some function of the interpoint distance (this a the

finite version of "long-range percolation").

If the randomly sprinkled points in geometric random graph are meant to model the position of elements in some ad-hoc network, then it is reasonable to consider the possibility that the position of the elements will change over time. Geometric random graphs with moving points have been considered by Diaz, Pérez, Serna, and Wormald [100].

Small worlds graphs

The small-world phenomenon, that all vertices in a connected network are connected by short paths has been observed experimentally in real world networks for years, including the seminal work of Milgram [193]. Many of the distributions in this chapter exhibit this property in the sense that **whp** the diameter grows as a logarithmic function of the number of vertices (or slower). In fact, it is more difficult to design a distribution which does not have low diameter (the geometric random graph above is one).

Nonetheless, distributions for graphs specifically designed to model the small-world phenomenon have been proposed. These graphs might be simple models which capture the important aspects of social networks, a link which has been described in detail by Watts [225].

The following version of the small-worlds graph is from work by Jon Kleinberg [168], and is slightly different from the original formulation of Watts and Strogatz [226] (which is also analyzed theoretically in [27]). In the spirit of the variations on the Erdős-Rényi graphs above, it is not hard to imagine a number of minor changes to this model, each of which could be more or less convenient depending on the situation.

A small-world graph G is formed by starting with some base graph H, which is typically a path of length n or an $n \times n$ grid graph, and then choosing m long-range random edges out of each vertex, independently according to the distribution

$$\mathbb{P}[i \text{ connects to } j] = \frac{d_H(i,j)^{-\alpha}}{\sum_{k \neq i} d_H(i,k)^{-\alpha}},$$

where α is a non-negative constant and $d_H(i,j)$ is the distance between i and j in H (here distance is number of edges on the shortest path, so, for example, if H is a path, then $d_H(i,j) = |i-j|$.)

When $\alpha = 0$, this is equivalent forming G by adding a random m-out to the base graph H, a model very similar to the smoothed random graph which will be described in Section 2.3 and used in as the model for the case study of connectivity algorithms in Chapter 7. Another extension of this approach appears in the hybrid power law graphs proposed by Chung and Lu [77].

Power law graphs

In the late 1990s, a selection of empirical measurements of networks in the real-world, such as those by Albert, Barabási, and Jeong [10], Broder et al [62], and Faloutsos, Faloutsos, and Faloutsos [108], led researchers to believe that the random graphs generated by Erdős-Rényi distributions fail to capture fundamental properties of "typical" graphs.

Several observers have found that, in the World Wide Web and the Internet, the proportion of vertices of a given degree follows an approximate inverse power law, which is to say that the proportion of vertices of degree k is approximately $Ck^{-\alpha}$ for some constants C, α. Since none of the Erdős-Rényi distributions above yield graphs with power law degree sequences, these observations have driven the development of several alternative models for random graphs.

There are currently several references on these distributions available [54, 202, 149, 225, 103, 56]. The survey by Bollobás and Riordan [54] particularly emphasizes mathematically rigorous results.

Random graph with a given degree sequence or given expected degree sequence

One approach to generating random power-law graphs is to study graphs with a given degree sequence (or given expected degree sequence), where the given sequence follows a power law. This was proposed as a model for the web graph by Aiello, Chung, and Lu in [9]. Mihail and Papadimitriou also use this model [192] in their study of large eigenvalues, as do Chung, Lu, and Vu in [78]. In this setting the configuration model of Bollobás can be used with a degree distribution that obeys a power law, or, alternatively, a model analogous to $\mathbb{G}_{n,p}$ but where each vertex has an expected degree d_i and the each pair ij appears as an edge independently with probability

$$\mathbb{P}[ij \in G] = \frac{d_i d_j}{\sum_{k=1}^{n} d_k}.$$

(For this to make sense, the expected degree sequence $(d_1, d_2, \ldots, d_n)$ must have $d_i^2 \leq \sum_{k=1}^{n} d_k$ for all i.)

This model for a random graph with a power-law degree distribution is the approach most similar to the Erdős-Rényi distributions, and it was studied more generally before the search for power laws began, for example by Molloy and Reed [195].

Preferential attachment

A popular alternative to the given-degree-sequence approach to power law graph generation is to sample graphs according to some generative procedure which happens to yield a power law distribution. There is a long history of this sort of generative procedure, which is outlined in the survey by Mitzenmacher [194]. It was proposed as a model for the web graph by Barabási and Albert [25], and their description was elaborated by Bollobás and Riordan in [52]. It was used by Bollobás, Riordan, Spencer, and Tusnády [53] who proved that the degree sequence does follow a power law distribution.

The distribution is defined conditionally, in the form of a procedure for generating a graph one edge at a time. Let G_t denote the random graph after the t-th edge has been added.

- At time step t, add a vertex v_t, and add an edge from v_t to some other vertex u, where

u is chosen at random according to the distribution:

$$\mathbb{P}\left[u = v_i \mid G_t\right] = \begin{cases} \frac{d_t(v_i)}{2t-1}, & \text{if } v_i \neq v_t; \\ \frac{1}{2t-1}, & \text{if } v_i = v_t; \end{cases}$$

where $d_t(v)$ denotes the degree of vertex v at time t. This means that each vertex receives an additional edge with probability proportional to its current degree (sometime summarized as "the rich get richer").

- For some constant m, every m steps contract the most recently added m vertices to form a supervertex.

This basic model has been extended in a number of directions. A generalization of the preferential attachment model is described by Drinea, Enachescu, and Mitzenmacher in [104], and degree sequence results analogous to [53] are proved for that model by Buckley and Osthus in [63]. Directed graphs generated with a preferential attachment model appear in [47]. Models which allow deletion of edges and vertices appear in [76, 92, 123].

An important tool in the analysis of the Preferential Attachment graph and its variants is a coupling designed by Bollobás and Riordan [51] which provides a (relatively) simple way to compare properties of a preferential attachment graph with an Erdős-Rényi graph.

The copying model

A completely different generative model, based on the idea that new webpages are often consciously or unconsciously copies of existing pages, is developed by Kleinberg et al and Kumar et al in [169, 180, 179, 178]. The copying model generates a directed graph.

As in the case of the preferential attachment model, the copying model is defined conditionally, and can be thought of as a dynamically growing digraph, where one node is added every time step. The copying model has two parameters, the out-degree d and the "copy factor" $\alpha \in [0, 1]$. If the digraph is D_t at time t, then D_{t+1} is obtained by adding node v_{t+1} and directing d arcs out of v_{t+1}, in the following way:

- Choose a prototype node w uniformly at random from the nodes in D_t.
- For the i-th out-arc of v_{t+1}, with probability α choose the destination of the arc uniformly at random from the nodes in D_t, and with the remaining probability choose the destination to be the same as the i-th out-arc of w.

Cooper and Frieze analyze a model combining preferential attachment and the copying model approaches in [90].

HOTs models

An alternative approach to generating random power law graphs is based on the observation that trade-offs between optimizing 2 different objectives can result in a power-law distribution. This was first proposed as a model for random graphs by Carlson and Doyle [66]. The

idea describes a very general class of distributions, which, like the preceding models is most conveniently defined by conditional probabilities, in terms of a procedure for generating the graph one vertex at a time.

Given graph G_{t-1}, the new vertex v_t attaches to vertices in the existing network so as to minimize a cost function which is a linear combination of two competing costs, which can be though of as a startup cost and an operating cost. The HOTs models which have been rigorously analyzed have added only one edge with each vertex, yielding a random tree.

One concrete example is the following: let each vertex v_i be associated with a point X_i, chosen uniformly at random from the unit square. Then when vertex v_t is added, it is connected to a single neighbor v_j which is the vertex which minimizes the sum

$$\alpha d_{G_{t-1}}(v_j, v_1) + |X_t - X_j|,$$

where α is a parameter of the model, $d_{G_{t-1}}(\cdot, \cdots)$ is the hop-count in graph G_{t-1}, and $|X_t - X_j|$ is the euclidean distance in the unit square. This model is studied in [107, 34], and an approach with alternative competing distances appears in [35, 36].

2.2 Problems with weights/distances:

Many problems in combinatorial optimization arise from discrete structures like graphs that have real valued weights on the edges or vertices. For example, an instance of the traveling salesman problem can be described by specifying the distance between each pair of cities.

2.2.1 Independent random weights

One natural distribution for problems of this type is to make all the weights independent random variables, uniformly distributed on the interval $[0, 1]$. Or, instead of uniform random variables, it is sometimes more convenient or elegant to consider independent exponentially distributed random variables.

In some graph problems, having a uniformly random weight on all vertices does not yield interesting instances, and it is better to start with a sparse random graph and then assign random weights to the edges which appear in the random graph.

2.2.2 Geometric distances as edge weights

Another common distribution for studying optimization algorithm on weighted graphs is similar to the geometric random graph distribution in Section 2.1.2, where n points are placed uniformly at random in the unit square, and the weight of an edge in the graph is given by the distance between the corresponding points in the square.

As above, any other metric space can be used for such a distribution.

2.3 Smoothed analysis and semi-random instances

Smoothed Analysis was introduced by Spielman and Teng in [213] to help explain why the

simplex algorithm for linear programming works well in practice but not in (worst-case) theory. They considered instances formed by taking an arbitrary constraint matrix and perturbing it by adding independent Gaussian noise with variance ϵ to each entry. They showed that, in this case, the shadow-vertex pivot rule succeeds in expected polynomial time.

The XOR perturbation is a method of smoothing an instance particularly suited for combinatorial problems. Here a random instance is formed by starting with an arbitrary graph $\bar{G}$ and then perturbing it by "XOR"ing it with a sparse random graph $R \sim \mathbb{G}_{n,\epsilon/n}$, resulting in a random graph $G = \bar{G} + \oplus R$, where an edge $ij \in G$ if and only if it is in exactly one of $\bar{G}$ and R.

Other models of smoothed instances have been considered by Banderier, Mehlhorn, and Beier [23]. One model used is that of randomly flipping low order bits of integer data. This model is also used in [30] to study the smoothed competitive ratio of a scheduling algorithm.

In the work of Beier and Vöcking [32], the authors analyze an algorithm for optimizing any integer linear program where the variables range over $\{0,1\}^n$ provided that one constraint (or the objective) is smoothed by adding a small amount of noise. They find that the smoothed problem has a polynomial-time algorithm if and only if the worst-case problem has a "pseudo-polynomial-time" algorithm, meaning an algorithm which runs in time polynomial in the size of the input when the input is represented in *unary*.

The semi-random model was introduced by Santha and Vazirani in [206]. In this model an adversary adaptively chooses a sequence of bits and each is corrupted independently with probability δ. They present this as a model for real-world random bits, such as the output of a Geiger counter or noisy diode, and consider the possibility of using such random bits in computation on worst-case instances. Blum and Spencer considered the performance of a graph coloring heuristic on random and semi-random instances in [39]. Subsequent work has uncovered an interesting difference between the random and semi-random instances in graph coloring. The work of Alon and Kahale [15] developed a heuristic which succeeds **whp** on random instances with constant expected degree, while work by Feige and Kilian [112] showed no heuristic can succeed on semi-random instances with expected degree $(1-\epsilon)\ln n$ (they also developed a heuristic for semi-random instances with expected degree $(1+\epsilon)\ln n$).

In the original semi-random model of Santha and Vazirani, an instance is formed by an adaptive adversary, who looks at all the bits generated so far, asks for a particular value for the next bit, and gets the opposite of what was asked for with probability δ. Several modifications are proposed in Blum and Spencer [39] and also in Subramanian, Fürer, and Veni Madhavan [218] and Feige and Krauthgamer [113]. However, all these variations maintain the adaptive aspect of the adversary's strategy, which at low density allows too much power; if the error probability $p = (1-\epsilon)\ln n/n$ then there will be roughly n^ϵ isolated vertices in $G \sim \mathbb{G}_{n,p}$ and the adversary will be able to encode a polynomial sized instance containing no randomness.

The XOR perturbation is equivalent to a natural weakening of the semi-random model: making the adversary oblivious.

Chapter 3

Tools

This chapter collects together a number of mathematical techniques that are indispensable for average-case analysis of algorithms. It is intended to serve primarily as a quick reference for the precise formulation of a number of useful inequalities. For this reason, instead of striving for the greatest generality, I have attempted to include the formulations that strike a balance between generality and ease of application. Where appropriate, I have included references which contain full proofs.

A greatly extended version of this chapter is currently planned, and will will include example applications for all the techniques in [125].

3.1 Moment Inequalities

3.1.1 First Moment Method

The *First Moment Method* is a term often used to refer to the following inequality which holds when X is any non-negative integer valued random variable:

$$\mathbb{P}[X \neq 0] \leq \mathbb{E}[X]. \tag{3.1}$$

Applications of the first moment method often take X to be the sum of indicator random variables, so this technique is sometimes called Boole's Inequality or the union bound.

3.1.2 Second Moment Method

The *Second Moment Method* often refers to the application of the following inequality, which holds when X is any real-valued random variable (and does not need X to non-negative nor integral as the First Moment Method does),

$$\mathbb{P}[X \neq 0] \geq \frac{\mathbb{E}[X]^2}{\mathbb{E}\left[X^2\right]}. \tag{3.2}$$

3.2 Tight Concentration Inequalities

3.2.1 Chernoff's Bound

Chernoff's bound refers to an inequality the bounds the probability of deviation from mean in terms of the mean only and not the number of terms in the sum. However it only applies to sums of independent random variables.

A versatile but easy-to-apply version of Chernoff's inequality states that for a sum of independent 0-1 Bernoulli random variables with parameters $p_1, \ldots, p_n$ and expectation $\mu = \sum_{i=1}^{n} p_i$,

$$\mathbb{P}[X \geq \mu + t] \leq \exp\left\{-\frac{t^2}{2\mu + 2t/3}\right\}, \tag{3.3}$$

$$\mathbb{P}[X \leq \mu - t] \leq \exp\left\{-\frac{t^2}{2\mu}\right\}. \tag{3.4}$$

Sometimes it is more convenient to use the weaker bound that for $\epsilon \leq 1$,

$$\mathbb{P}[|X - \mu| \geq \epsilon\mu] \leq 2\exp\left\{-\frac{\epsilon^2\mu}{3}\right\}. \tag{3.5}$$

The proof of these inequalities appears, for example as [156, Theorems 2.1 and 2.8].

3.2.2 Hoeffding-Bernstein Inequality

The previous concentration bound only applies to a random variable that is the sum of Bernoulli random variables. The Hoeffding-Bernstein Inequality applies when X_i, $1 \leq i \leq n$ are independent random variables with each $\mathbb{E}[X_i] = 0$ and no two values of any X_i ever more than one apart. Let $S = X_1 + \cdots + X_n$. Then

$$\mathbb{P}[S > a] < \exp\left\{-2a^2\right\}. \tag{3.6}$$

The proof of this inequality appears, for example as [17, Theorem A.1.18].

3.2.3 Azuma-Hoeffding Inequality

When a random variable is a function of independent random variables it is possible to obtain exponential concentration bounds in a similar spirit to those above. In fact, the bounds above can be considered a special case, where the random variable of interest is a function of independent random variables, and the function is to sum the inputs. However, when the function is more complicated than the sum, in addition to the mean, the number of inputs to the function also must appear in the inequality.

The following is a version of the Azuma-Hoeffding inequality in a form due to McDiarmid [190] (see also Bollobás [44]): Let $X_1, \ldots, X_n$ be independent random variables, with X_k taking values in a set A_k for each k. Suppose that a measurable function $f\colon \prod A_k \to \mathbb{R}$

satisfies $|f(x) - f(x')| \leq c_k$ whenever the vectors x and x' differ only in the k'th coordinate. Let Z be the random variable $Z = f(X_1, \ldots, X_n)$. Then for any $t > 0$,

$$\mathbb{P}\left[Z \geq \mathbb{E}[Z] + t\right] \leq \exp\left\{-2t^2 / \sum c_k^2\right\}. \tag{3.7}$$

As for many inequalities of this sort, it follows from symmetry (replacing f by $-f$) that

$$\mathbb{P}\left[Z \leq \mathbb{E}[Z] - t\right] \leq \exp\left\{-2t^2 / \sum c_k^2\right\}. \tag{3.8}$$

3.2.4 Talagrand's Inequality

An alternative inequality which is capable of providing concentration bounds for a random variable that is a function of independent random variables is Talagrand's Inequality. The setting is as follows: let $A \subseteq \Omega^n$. Any $\beta \in \mathbf{R}^n$ defines a sort of weighted Hamming distance, so that for any $\mathbf{x}, \mathbf{y} \in \Omega^n$, $d_\beta(\mathbf{x}, \mathbf{y}) = \sum_{i=1}^n \beta_i \mathbf{1}(\mathbf{x}_i \neq \mathbf{y}_i)$. This induces a distance between a set and a point, $d_\beta(A, \mathbf{x}) = \inf_{\mathbf{y} \in A} d_\beta(\mathbf{x}, \mathbf{y})$. The Talagrand convex distance function is then $d(A, \mathbf{x}) = \sup_{\beta : \|\beta\|_2 = 1} d_\beta(A, \mathbf{x})$.

Talagrand's Inequality states that for any measure space $(\Omega, \mathcal{E}, \mathbb{P})$, and value $t > 0$, for any $A \in \Omega^n$ such that A and $\{\mathbf{x} : d(A, \mathbf{x}) \geq t\}$ are measurable,

$$\mathbb{P}[\mathbf{x} \in A]\mathbb{P}[d(A, \mathbf{x}) \geq t] \leq \exp\left\{-t^2/4\right\}. \tag{3.9}$$

In combinatorial applications, the measurability is generally automatic; in particular this is so when Ω is a finite space and n is finite. Talagrand's inequality is often applied by choosing some explicit weighting $\beta = \beta(\mathbf{x})$ with $\|\beta(\mathbf{x})\| = 1$, so that by definition $\mathbb{P}\left[d_{\beta(\mathbf{x})}(A, \mathbf{x}) \geq t\right] \leq \mathbb{P}[d(A, \mathbf{x}) \geq t]$, and thus $\mathbb{P}[\mathbf{x} \in A]\mathbb{P}\left[d_\beta(A, \mathbf{x}) \geq t\right] \leq \exp\left\{-t^2/4\right\}$. By choosing A so that either $\mathbb{P}[A] \geq 1/2$ or $\mathbb{P}\left[d_\beta(A, \mathbf{x}) \geq t\right] \geq 1/2$, we get a bound on the other.

A proof of Talagrand's Inequality, as well as example applications and several related concentration inequalities can be found in [221].

3.2.5 Symmetric Logarithmic Sobolev Inequality

The Symmetric Logarithmic Sobolev Inequality is yet another technique for showing that a random variable is tightly concentrated around its mean. It is equivalent to Talagrand's Inequality, in the sense that each can be used to quickly derive the other. However, they seem philosophically different in approach. To describe the Symmetric Log-Sob Inequality, let $\tau(x) = x(e^x - 1)$. Let $X_1, X_1', \ldots, X_n, X_n'$ be independent identically distributed random variables, let $Z = g(X_1, \ldots, X_n)$, and let $Z_i' = g(X_1, \ldots, X_{i-1}, X_i', X_{i+1}, \ldots, X_n)$. Then it follows that

$$s\mathbb{E}[Ze^{sZ}] - \mathbb{E}[e^{sZ}] \log \mathbb{E}[e^{sZ}] \leq \sum_{i=1}^n \mathbb{E}[e^{sZ} \tau(-s(Z - Z_i'))\mathbf{1}(Z > Z_i')]$$

and

$$s\mathbb{E}[Ze^{sZ}] - \mathbb{E}[e^{sZ}]\log\mathbb{E}[e^{sZ}] \leq \sum_{i=1}^{n}\mathbb{E}[e^{sZ}\tau(-s(Z_i'-Z))\mathbf{1}(Z<Z_i')].$$

Often, this imposing inequality can be used in a more digestible form. Using the same definitions as in the previous theorem, if

$$\mathbb{E}\Big[\sum_{i=1}^{n}(Z-Z_i')^2\mathbf{1}(Z>Z_i') \;\Big|\; X_1,\ldots,X_n\Big] \leq C$$

then

$$\mathbb{P}[Z > \mathbb{E}[Z]+t] \leq e^{-t^2/4C},$$

and if

$$\mathbb{E}\Big[\sum_{i=1}^{n}(Z-Z_i')^2\mathbf{1}(Z<Z_i') \;\Big|\; X_1,\ldots,X_n\Big] \leq C$$

then

$$\mathbb{P}[Z < \mathbb{E}[Z]-t] \leq e^{-t^2/4C}.$$

The proof of this inequality, as well as many example applications appears in [60].

3.3 Principle of deferred decisions

The principle of deferred decisions, which appears in [170], is a powerful method of simplifying probabilistic reasoning, that is often summarized as "Don't do today what you can put off till tomorrow."

It seems that this principle is better described by example than any formal definition. So as an example consider the probability that vertices v and w are within distance 2 in $G \sim \mathbb{G}_{n,p}$. Employing the principle of deferred decisions, first reveal candidate edge vw. If it is in G (which is the case with probability p), then v and w are adjacent. If vw is not an edge of G, then reveal all the candidate edges incident to v (and defer the decisions for the other edges in the graph). Letting S denote the neighborhood of v (so $|S| \sim \mathrm{Bi}(n-2,p)$), to determine if v and w are distance 2 apart reveal all the candidate edges from S to w. The probability that at least one of these edges appears in G is $1-(1-p)^{|S|}$. Putting these 3 phases of decision making together, and calculating that $\mathbb{E}[(1-p)^{|S|}] = \left(1-p^2\right)^{n-2}$ shows that

$$\mathbb{P}[d_G(v,w)\leq 2] = p+(1-p)\mathbb{E}[1-(1-p)^S] = p+(1-p)(1-\left(1-p^2\right)^{n-2}).$$

In [171], the principle of deferred decisions is called "late binding".

Chapter 4

Sharp thresholds for a boolean k-CSP

This chapter originally appeared as [118]. It applies some of the techniques from Chapter 3 to study the following constraint satisfaction problem (CSP):

GIVEN: a set of boolean variables $V = \{x_1, \ldots, x_n\}$;
a set of clauses, $C = \{C_1, \ldots, C_m\}$, where $C_i = (s_{i_1} x_{i_1}, \ldots, s_{i_k} x_{i_k})$, for $s_{i_j} \in \{-1, 1\}$;
a set of "bad" clause assignments $Q \subseteq \{-1, 1\}^k$ with $|Q| = q$.

FIND: an assignment ψ: $V \to \{-1, 1\}$ such that for all C_i,

$$(s_{i_1}\psi(x_{i_1}), \ldots, s_{i_k}\psi(x_{i_k})) \notin Q$$

An instance $I = (V, C, Q)$ is called *satisfiable* if such an assignment exists. If no such assignment exists, I is called *unsatisfiable*.

This chapter focuses on instances generated by including every k-tuple of literals independently at random with probability $p = p(n)$, while allowing arbitrary sets Q of bad clause assignments. By considering particular sets of bad clause assignments, CSP specializes to two well known problems, k-SAT and not-all-equal k-SAT.

- *k-SAT is a special case of CSP:* take $Q = \{-1^k\}$, i.e. there is one way for a clause to go bad, the setting which makes every literal in the clause false. Random k-SAT has been well studied, and a sharp threshold is known for $k = 2$ [48, 81, 116, 133, 138, 142, 222] and $k - \log n \to \infty$ [134]. For other values of k, in particular $k = 3$, a sharp threshold function is known to exist [127], but it is unknown what the function is. Upper and lower bounds are given in [1, 6, 61, 70, 71, 105, 138, 157, 167]. Recent work using a subtle modification of the second moment method has determined that the function tends to a constant for k a sufficiently large constant [5].

- *not-all-equal k-SAT is a special case of CSP:* take $Q = \{-1^k, 1^k\}$. The satisfiability threshold for random not-all-equal-SAT is studied for $k = 3$ in [2] and a sharp threshold constant is known when k is a sufficiently large constant [4].

Let the clause size $k = k(n)$ a function satisfying $k \geq D_{\epsilon,q} \log_2 n$, where $D_{\epsilon,q}$ is sufficiently large (for $\epsilon \leq \frac{1}{9}, D \geq 5\frac{1}{\epsilon} \ln \frac{q}{\epsilon}$ is enough). Then for any p and for a family of bad clause assignments $\{Q_i\}$ with $|Q_n| = q$, define $I = I_{n,p}$ to be $(\{x_1, \ldots, x_n\}, C_{n,p}, Q_n)$, where $C_{n,p}$ is generated by including each k-tuple of literals independently at random with probability p.

Theorem 1 *For any natural number q and any $\epsilon > 0$ there exists $D_{\epsilon,q}$ such that for $k \geq D_{\epsilon,q} \log n$ and any family of bad clause assignments $\{Q_i\}$ with $|Q_n| = q$,*

$$\lim_{n\to\infty} \mathbb{P}[I_{n,p} \text{ is satisfiable}] = \begin{cases} 1, & \text{if } p \leq (1-\epsilon)\frac{\ln 2}{qn^{k-1}}; \\ 0, & \text{if } p \geq (1+\epsilon)\frac{\ln 2}{qn^{k-1}}. \end{cases}$$

The consideration of "moderately growing clauses" is inspired by the work of Frieze and Wormald [134]. It appears that threshold results which require great labor for constant clause size become much easier when clause size is a sufficiently large function of n. In the following, the minimum necessary clause size $D_{\epsilon,q} \log n$ will be larger than $\log n$, so Theorem 1 holds for a smaller range of k than the threshold of [134] (which in turn holds for a smaller range of k that the threshold of [5]). However, Theorem 1 does not require as delicate a calculation as [134], and proves thresholds for other interesting specializations in one go.

CSP is equivalent to a problem studied by Creignou and Daudé in [96]. They apply Friedgut's theory of sharp threshold *functions* to CSP and determine that there are essentially 2 types of bad clauses that lead to coarse thresholds. The results of this paper compliment their results by proving that some of the sharp threshold functions are *constant*, and by determining exactly what the constant is.

Xu and Li obtained similar results using similar techniques for a different type of constraint satisfaction problem in [229]. They consider instances which have clauses of a fixed size k, allow variables to take values from a domain with $d = n^\alpha$ values, and have a different bad set for each clause chosen randomly, to prohibit $\Theta(d^k)$ candidate assignments. (In contrast, here clauses have size $k = \Omega(\log n)$, a boolean domain of size $d = 2$, and a bad set prohibiting a small number candidate assignments, which is the same set for each clause, and chosen non-randomly.)

The remainder of this note will prove Theorem 1. Section 4.1 shows unsatisfiability above the threshold by the first moment method. Section 4.2 shows satisfiability below the threshold by the second moment method.

In this chapter $\log x$ means $\log_2 x$, while $\ln x$ refers to the base e logarithm, and $\log_\alpha x$ for the base-α logarithm.

4.1 Upper bound via First Moment Method

This section shows that $I = I_{n,p}$ is unsatisfiable above the threshold **whp**. The proof is by the first moment method.

Claim 1 *Let $p_0 = \frac{\ln 2}{qn^{k-1}}$. Then for any $p \geq (1+\epsilon)p_0$, for any Q with $|Q| = q$,*

$$\lim_{n\to\infty} \mathbb{P}[I_{n,p} \text{ is satisfiable}] = 0.$$

Proof For a particular assignment ϕ, there are qn^k clauses which violate some constraint of Q with respect to ϕ. So the probability that ϕ satisfies I is the probability that none of these clauses occur,

$$\mathbb{P}[\phi \text{ satisfies } I] = (1-p)^{qn^k}.$$

Let X denote the expected number of assignments satisfying I, so $\mathbb{E}[X] = 2^n(1-p)^{qn^k}$.

For $p \geq \frac{\ln 2}{qn^{k-1}}(1+\epsilon)$,

$$\mathbb{E}[X] \leq 2^n \exp\{-n(1+\epsilon)\ln 2\} = 2^{-\epsilon n}.$$

Therefore

$$\mathbb{P}[X \neq 0] \leq \mathbb{E}[X] \leq 2^{-\epsilon n}.$$

□

4.2 Lower bound via Second Moment Method

This section shows $I = I_{n,p}$ is satisfiable below the threshold **whp**. The proof is by the second moment method.

Claim 2 *Let $p_0 = \frac{\ln 2}{qn^{k-1}}$. Then for any $p \leq (1-\epsilon)p_0$, for any Q with $|Q| = q$,*

$$\lim_{n\to\infty} \mathbb{P}[I_{n,p} \text{ is satisfiable}] = 1.$$

Proof As above, let X denote the number of assignments satisfying I. Begin by calculating the second moment of X. Let $Q_i = \{\{b, b'\} \in Q \times Q\colon dist(b, b') = i\}$, where $dist(b, b')$ is the Hamming distance between b and b' (in other words, Q_i is the set of pairs of bad assignments which differ in i places). Let $q_i = |Q_i|$. Note that $q_0 = q$ and $q_k \leq q/2$.

$$\begin{aligned}
\mathbb{E}\left[X^2\right] &= \sum_{\phi} \mathbb{P}[\phi \text{ satisfies } I] \sum_{\phi'} \mathbb{P}[\phi' \text{ satisfies } I \,\big|\, \phi \text{ satisfies } I] \\
&= \sum_{\phi} \mathbb{P}[\phi \text{ satisfies } I] \sum_{s=0}^{n} \binom{n}{s} \mathbb{P}\left[\phi' \text{ satisfies } I \,\big|\, \substack{\phi \text{ satisfies } I \\ dist(\phi,\phi')=n-s}\right] \\
&= \sum_{\phi} (1-p)^{qn^k} \sum_{s=0}^{n} \binom{n}{s} (1-p)^{qn^k - \sum_{i=0}^{k} q_i s^{k-i}(n-s)^i} \\
&= 2^n (1-p)^{qn^k} \sum_{s=0}^{n} \binom{n}{s} (1-p)^{qn^k - \sum_{i=0}^{k} q_i s^{k-i}(n-s)^i}.
\end{aligned}$$

where the probabilities in the second to last line follow since there are qn^k candidate clauses which are bad for assignment ϕ, qn^k which are bad for assignment ϕ', and $\sum_{i=0}^{k} q_i s^{k-i}(n-s)^i$ which are bad for both ϕ and ϕ'.

The ratio $\mathbb{E}\left[X^2\right]/\mathbb{E}[X]^2$ is the expected value of a different random variable:

$$\begin{aligned}\frac{\mathbb{E}\left[X^2\right]}{\mathbb{E}[X]^2} &= \sum_{s=0}^{n} \binom{n}{s} 2^{-n} (1-p)^{-\sum_{i=0}^{k} q_i s^{k-i}(n-s)^i} \\ &= E\left[(1-p)^{-\sum_{i=0}^{k} q_i S^{k-i}(n-S)^i}\right] \\ &= E\left[\left(1+\frac{p}{1-p}\right)^{\sum_{i=0}^{k} q_i S^{k-i}(n-S)^i}\right],\end{aligned}$$

where $S \sim \mathrm{Bi}(n, 1/2)$.

Now, let

$$Y = \left(1+\frac{p}{1-p}\right)^{\sum_{i=0}^{k} q_i S^{k-i}(n-S)^i},$$

and bound $\mathbb{E}[Y]$ in 3 parts:

$$\mathbb{E}[Y] \le \sum_{i=1}^{3} \mathbb{E}\left[Y \;\middle|\; \eta_{i-1} \le |n/2 - S| \le \eta_i\right] \mathbb{P}\left[\eta_{i-1} \le |n/2 - S| \le \eta_i\right],$$

where

$$\eta_0 = 0, \quad \eta_1 = \epsilon\frac{n}{2}, \quad \eta_2 = \frac{n}{2}\left(1 - \frac{\epsilon}{\log n}\right), \quad \eta_3 = \frac{n}{2}.$$

The following calculations rely on the fact that $\sum_{i=0}^{k} q_i = q(q+1)/2 < q^2$.

First Term: Provided $k \ge 2\log_\alpha n$ where $\alpha = \frac{2}{1+\epsilon}$,

$$\begin{aligned}\mathbb{E}\left[Y \;\middle|\; \eta_0 \le |n/2 - S| \le \eta_1\right] &\mathbb{P}\left[\eta_0 \le |n/2 - S| \le \eta_1\right] \\ &\le \left(1+\frac{p}{1-p}\right)^{q^2\left(\frac{1}{2}n(1+\epsilon)\right)^k} \\ &\le \exp\left\{n\frac{q\ln 2(1-\epsilon)}{1-p}\left(\frac{1+\epsilon}{2}\right)^k\right\} \\ &= 1 + o(1).\end{aligned}$$

Second Term: By the version of Chernoff's bound from Inequality (3.5),

$$\mathbb{P}\left[\eta_1 \le |n/2 - S| \le \eta_2\right] \le 2e^{-n\epsilon^2/3}.$$

So, since $k \geq \left(\frac{2}{\epsilon}\ln\frac{3q}{\epsilon^2}\right)\log n$,

$$\begin{aligned}\mathbb{E}\left[Y \;\middle|\; \eta_1 \leq |n/2 - S| \leq \eta_2\right]\mathbb{P}\left[\eta_1 \leq |n/2 - S| \leq \eta_2\right] \\ &\leq \left(1+\frac{p}{1-p}\right)^{q^2\left(n(1-\frac{\epsilon}{2\log n})\right)^k} e^{-n\epsilon^2/3} \\ &\leq \exp\left\{n\frac{q\ln 2(1-\epsilon)}{1-p}\left(1-\frac{\epsilon}{2\log n}\right)^k - n\epsilon^2/3\right\} \\ &= o(1).\end{aligned}$$

Third Term: Note that $q_k \leq q$. So, since $\eta_2 \leq |n/2 - S| \leq \eta_3$,

$$\sum_{i=0}^{k} q_i S^{k-i}(n-S)^i \leq qn^k + q\left(\frac{n}{\log n}\right)^k + \sum_{i=1}^{k-1} q_i S^{k-i}(n-S)^i \leq (q + q^2/\log n)n^k,$$

and

$$\begin{aligned}\mathbb{E}\left[Y \;\middle|\; \eta_2 \leq |n/2 - S| \leq \eta_3\right]\mathbb{P}\left[\eta_2 \leq |n/2 - S| \leq \eta_3\right] \\ &\leq \left(1+\frac{p}{1-p}\right)^{n^k\left(q+q^2/\log n\right)} 2\binom{n}{n/2+\eta_2}2^{-(n/2+\eta_2)} \\ &\leq 2e^{n\frac{\ln 2(1-\epsilon)}{1-p}(1+q/\log n)} n^{n\frac{\epsilon}{2\log n}} 2^{-n(1-\frac{\epsilon}{2\log n})} \\ &= 2^{1+n(1-\epsilon)(1+o(1))+n\frac{\epsilon}{2}-n(1-\frac{\epsilon}{2\log n})} \\ &= 2^{-\frac{\epsilon}{2}n(1-o(1))} \\ &= o(1).\end{aligned}$$

Putting the parts together and using the second moment inequality, we have

$$\mathbb{P}[X \neq 0] \geq \frac{\mathbb{E}[X]^2}{\mathbb{E}[X]^2} \geq 1 - o(1).$$

□

Chapter 5

3-SAT in planted random instances

This chapter originally appeared as [117]. It studies the performance of a heuristic for solving 3-SAT on random instances that a drawn from a distribution with a planted satisfying assignment.

5.1 Introduction

A 3CNF formula over variables $x_1, \ldots, x_n$ consists of clauses $C_1, \ldots, C_m$, where each clause is the disjunction of 3 literals, $C_i = \ell_{i_1} \vee \ell_{i_2} \vee \ell_{i_3}$, and each literal is a variable or the negation of a variable. A 3CNF formula is satisfiable if there is an assignment of variables to truth values so that every clause contains at least one true literal. Finding a satisfying assignment for a given 3CNF formula is **NP**-hard, so it is unlikely there is an efficient algorithm (meaning an algorithm with running time bounded by a polynomial function of the input size) which succeeds on all 3CNF formulas [87, 182]. In the spirit of average-case analysis of algorithms, it is interesting to investigate the existence of efficient algorithms testing for and generating satisfying assignments which work with high probability (meaning with probability tending to 1 as n goes to infinity and abbreviated **whp**) over some reasonable distribution of formulas.

Uniformly random 3CNF formulas have been the focus of extensive research. It is known that there is a sharp threshold in the ratio of clauses to variables; a random 3CNF with clause-to-variable ratio below the threshold is satisfiable **whp** and one with ratio above is not satisfiable **whp** [127]. This threshold is not known exactly, (and not even known to tend to a constant) but it is known to be at least 3.4 (see [161]) and no more than 4.5 (see [162]). Experimental results predict the higher end of this interval [95]. (Much more is known for k-CNF formulas where k is a large constant or slowly growing function [5, 134, 118]).

It is conjectured that proving the non-existence of satisfying assignments slightly above the threshold is computationally difficult, which yields nice results in hardness of approximations [111]. One piece of evidence supporting this conjecture is the exponential length of resolution-type proofs refuting such instances [82, 33]. Spectral techniques are effective in efficiently proving the unsatisfiability of formulas an $n^{1/2+\epsilon}$ factor above the threshold [129, 143, 144, 85].

An alternative approach is to investigate exponential-time algorithms which find a satisfying assignment or prove that none exists for all instances. Then the challenge is to make the base of the exponent as small as possible (see, for example, [209, 151]).

5.1.1 The distribution

In this chapter we will consider random 3CNF formulas with a "planted" satisfying assignment. That is, formulas which have a clause-to-variable ratio above the threshold, but which are generated in a way to ensure that they are satisfiable. To be exact, to form instance $I = I_{n,p_1,p_2,p_3}$ we choose a truth assignment ϕ on n variables uniformly at random and include in I each clause with exactly i literals satisfied by ϕ independently with probability p_i. By setting $p_1 = p_2 = p_3$ we obtain the model studied by Motoki and Uehara in [198], which shows a threshold of $p = \Theta(\log n/n^2)$ for 3CNF formulas to have exactly 1 satisfying assignment. An algorithm for $p_1 = p_2 = p_3$ was analyzed by Koutsoupias and Papadimitriou in [173]. They show that a greedy variable assignment rule successfully discovers a satisfying assignment **whp** for instances with $p_1 = p_2 = p_3 = \Omega(\log n/n^2))$. The authors conjecture that some modification of the greedy algorithm will work when $p_1 = p_2 = p_3 = O(1/n^2)$. The results of this chapter show that their conjecture is correct; in the case where $p_1 = p_2 = p_3$, the spectral phase of the algorithm presented below can be replaced by the greedy assignment rule.

By setting $p_1 = p_2$ and $p_3 = 0$, we obtain a natural distribution on 3CNFs with a planted not-all-equal assignment, a situation where the greedy variable assignment rule generates a random assignment. By setting $p_2 = p_3 = 0$, we obtain 3CNFs with a planted exactly-one-true assignment (which succumb to the greedy algorithm followed by the non-spectral steps below). Also, by correctly adjusting the ratios of p_1, p_2, and p_3, we obtain a variety of (slightly less natural) instance distributions which thwart the greedy algorithm. Carefully selected values of p_1, p_2, and p_3 are considered in [28], where it is conjectured that no algorithm running in polynomial time can solve I_{n,p_1,p_2,p_3} **whp** when $p_i = c_i\alpha/n^2$ and

$$0.077 < c_3 < 0.25 \qquad c_2 = (1 - 4c_3)/6$$
$$c_1 = (1 + 2c_3)/6 \qquad \alpha > \frac{4.25}{7}.$$

An implication of the results in this chapter is that this conjecture fails when α is a sufficiently large constant.

This chapter originally appeared as an extended abstract [117], and subsequent extensions have shown that a similar styled analysis is capable of showing that alternative algorithms succeed in *expected* polynomial time [177] and on *semi-random* instances [115].

In this chapter we allow clauses with repeated variables, and formulas with the same clause with the literals in a different order, so there are $8n^3$ possible clauses, $7n^3$ which are consistent with our planted assignment ϕ. The results and proofs hold for other similar models, such as prohibiting clauses with repeated literals or clauses where the same set of literals appear in a different order.

5.1.2 The algorithm

The main result of this chapter is a polynomial time algorithm which returns a satisfying assignment to I_{n,p_1,p_2,p_3} **whp** when $p_1 = d/n^2$, $p_2 = \eta_2 d/n^2$ and $p_3 = \eta_3 d/n^2$, for $0 \leq \eta_2, \eta_3 \leq 1$, and $d \geq d_{min}$, where d_{min} is a function of η_2, η_3. These restrictions on η_2 and η_3 are more for convenience than necessity, and in Section 5.2, we will see the matrix equation which dictates the allowable range of η_i's. The unsymmetrical feature of this parameterization, that there is no η_1 is not entirely for convenience, however. If we desire any $\eta_1 > \max\{\eta_2, \eta_3\}$, we can renormalize by changing the value of d. Unfortunately, taking $\eta_1 = 0$ is more complicated. The proof of correctness of Step 4 of the algorithm as described below relies on there being some positive fraction of clauses with 1 true literal, and it is not clear how to remove this requirement.

The algorithm below is an extension of the 3-coloring algorithm of Alon and Kahale [15]. They describe an algorithm which, in an analogous model of a random 3-colorable graph, finds a proper 3-coloring in polynomial time **whp**. A previous extension of their algorithm by Chen and Frieze [74] adapted the technique to 2-color random 3-uniform bipartite hypergraphs. We follow the same approach, which is outlined in Figure 5.1

---

procedure Spectral3Sat(I)

1. Construct a graph G from the 3CNF.
2. Find the most negative eigenvalue of a matrix related to the adjacency matrix of G.
3. Assign a value to each variable based on the signs of the eigenvector corresponding to the most negative eigenvalue.
4. Iteratively improve the assignment.
5. Perfect the assignment by exhaustive search over a small set containing all the incorrect variables.

Figure 5.1: Outline of algorithm for solving random satisfiable instances of 3SAT

Each step in Figure 5.1 requires some elaboration.

Step (1): Given 3CNF $I = I_{n,p_1,p_2,p_3}$, where $p_1 = \frac{d}{n^2}$, $p_2 = \eta_2 \frac{d}{n^2}$, and $p_3 = \eta_3 \frac{d}{n^2}$, the graph in step (1) $G = (V, E)$ has $2n$ vertices, corresponding to the literals in I and labeled $\{x_1, \overline{x}_1, \ldots x_n, \overline{x}_n\}$. G has an edge between vertices ℓ_i and ℓ_j if I includes a clause with both ℓ_i and ℓ_j (do not add multiple edges).

Step (2): We consider $G' = (V, E')$ formed by deleting all the edges incident to vertices with degree greater than $180d$. Let A be the adjacency matrix of G'. Let λ be the most negative eigenvalue of A and $\mathbf{v}$ be the corresponding eigenvector.

Step (3): There are two assignments to consider, π_+, which is defined by

$$\pi_+(x_i) = \begin{cases} T, & \text{if } \mathbf{v}_i \geq 0; \\ F, & \text{otherwise;} \end{cases}$$

and π_-, which is defined by

$$\pi_-(x) = \neg\pi_+(x).$$

Let π_0 be the better of π_+ and π_- (that is, the assignment which satisfies more clauses). We will argue in the next section that π_0 agrees with ϕ on at least $(1 - C/d)n$ variables for some absolute constant C.

Step (4): For $i = 1, \ldots, \log n$ do the following: for each variable x, if x appears in $5\epsilon d$ clauses unsatisfied by π_{i-1}, then set $\pi_i(x) = \neg\pi_{i-1}(x)$, where ϵ is an appropriately chosen constant (taking $\epsilon = 0.1$ works); otherwise set $\pi_i(x) = \pi_{i-1}(x)$.

Step (5): Let $\pi'_0 = \pi_{\log n}$ denote the final assignment generated in step (4). Let $\mathcal{A}_4^{\pi'_0}$ be the set of variables which do not appear in $(3 \pm 4\epsilon)d$ clauses as the only true literal with respect to assignment π'_0, and let $\mathcal{B}$ be the set of variables which do not appear in $(\mu_D \pm \epsilon)d$ clauses, where $\mu_D d = (3+6)d + (6+3)\eta_2 d + 3\eta_3 d + \mathcal{O}(1/n)$ is the expected number of clauses containing variable x. Form partial assignment π'_1 by unassigning all variables in $\mathcal{A}_4^{\pi'_0}$ and $\mathcal{B}$. Now, for $i \geq 1$, if there is a variable x_i which appears in less than $(\mu_D - 2\epsilon)d$ clauses consisting of variables that are all assigned by π'_i, let π'_{i+1} be the partial assignment formed by unassigning x_i in π'_i. Let π' be the partial assignment when this process terminates. Consider the graph Γ with a vertex for each variable that is unassigned in π' and an edge between two variables if they appear in a clause together. If any connected component in Γ is larger than $\log n$ fail. Otherwise, find a satisfying assignment for I by performing an exhaustive search on the variables in each connected component of Γ.

Theorem 2 *For any constants $0 \leq \eta_2, \eta_3 \leq 1$, except $(\eta_2, \eta_3) = (0, 1)$, there exists a constant d_{min} such that for any $d \geq d_{min}$, if $p_1 = d/n^2$, $p_2 = \eta_2 d/n^2$, and $p_3 = \eta_3 d/n^2$ then this polynomial-time algorithm produces a satisfying assignment for random instances drawn from I_{n,p_1,p_2,p_3}* **whp**.

The exception in this theorem is easy to circumvent superficially, since the case where $\eta_2 = 0$ is solvable in worst-case polynomial time by Gaussian elimination. However, the parameterization above obscures the fact that there are instances which appear to be hard in general when $p_2 = \sqrt{d}n$. This is the case sometimes called Gaussian elimination with noise.

5.1.3 Outline of what follows

We will prove Theorem 2 in the next 2 sections. Section 5.2 shows that **whp** the eigenvector corresponding to the most negative eigenvalue is close to a satisfying assignment, by showing that π_0 agrees with ϕ on at least $(1 - C/d)n$ clauses, where C is an absolute constant. Section 5.3 shows that **whp** the iterative reassignment in step (4) correctly assigns a larger

fraction of variables, so large that the connected components left after unassignment are size $\mathcal{O}(\log n)$ and the exhaustive search in step (5) can perfect the assignment in polynomial time.

5.2 Spectral Arguments

The goal of this section is to show the assignment constructed from the eigenvector corresponding to the most negative eigenvalue of A is correct on a $1 - C/d$ fraction of variables, where C is a constant independent of d. The intuition behind this result is as follows. Suppose $\phi(x) = T$. Then

- literal x appears in about $3d$ clauses with 2 false literals, $6\eta_2 d$ clauses with 1 false and 1 true literal, and $3\eta_3 d$ clauses with 2 true literals
- literal $\bar{x}$ appears in about $6d$ clauses with 1 false and 1 true literal and $3\eta_2 d$ clauses with 2 true literals.

We use these estimates to describe roughly the adjacency matrix A. For each row corresponding to a true literal, we have nonzero columns for about $6\eta_2 d + 6\eta_3 d$ true literals and $6d + 6\eta_2 d$ false literals. Similarly, for each row corresponding to a false literal, we have nonzero columns for about $6d + 6\eta_2 d$ true literals and $6d$ false literals. Let $\mathbf{v}_T$ be the vector with $\mathbf{v}_T(\ell) = 1$ if ℓ is a true literal, and 0 if ℓ is a false literal, and let $\mathbf{v}_F = \mathbf{1} - \mathbf{v}_T$. Then we have

$$A\mathbf{v}_T \approx (6\eta_2 d + 6\eta_3 d)\mathbf{v}_T + (6d + 6\eta_2 d)\mathbf{v}_F$$
$$A\mathbf{v}_F \approx (6d + 6\eta_2 d)\mathbf{v}_T + 6d\mathbf{v}_F.$$

So, heuristically, we expect 2 eigenvectors of A to look something like $\beta\mathbf{v}_T + \gamma\mathbf{v}_F$ where β, γ solve the 2-dimensional system

$$\begin{bmatrix} \eta_2 + \eta_3 & 1 + \eta_2 \\ 1 + \eta_2 & 1 \end{bmatrix} \begin{bmatrix} \beta \\ \gamma \end{bmatrix} = \alpha \begin{bmatrix} \beta \\ \gamma \end{bmatrix}.$$

When $0 \leq \eta_2, \eta_3 \leq 1$ and $(\eta_2, \eta_3) \neq (0, 1)$, this yields a positive eigenvalue α_+ and a negative eigenvalue α_-. To see this, note that the trace of the matrix equals the sum of the eigenvalues, and then calculate that one of the eigenvalues take the form $(1 + \eta_2 + \eta_3 + \sqrt{(1 + \eta_2 + \eta_3)^2 - 4(\eta_2 + \eta_3 - (1 + \eta_2)^2)})/2$. The trace of the matrix is $1 + \eta_2 + \eta_3$ and this eigenvalue is strictly larger provided $\eta_2 + \eta_3 - (1 + \eta_2)^2$ is negative. Simplifying terms shows that is equivalent to having $1 + \eta_2 + \eta_2^2 - \eta_3 > 0$, which is the case for all $0 \leq \eta_2, \eta_3 \leq 1$ besides $(\eta_2, \eta_3) = (0, 1)$.

Let $\begin{bmatrix} \beta_+ \\ \gamma_+ \end{bmatrix}$ be the eigenvector corresponding to α_+ and $\begin{bmatrix} \beta_- \\ \gamma_- \end{bmatrix}$ be the eigenvector corresponding to α_-. Then additional calculation shows that if we normalize so $\beta_-^2 + \gamma_-^2 = 1$, then β_- and γ_- have opposite signs, and $|\beta_-|, |\gamma_-| \geq \sqrt{\frac{1}{17}}$.

Let $\mathbf{v}_+ = \beta_+ \mathbf{v}_T + \gamma_+ \mathbf{v}_F$ and $\mathbf{v}_- = \beta_- \mathbf{v}_T + \gamma_- \mathbf{v}_F$ denote our heuristic approximation of the eigenvectors.

We will argue that **whp** A has large positive and negative eigenvalues roughly equal to $6d$ times the $\alpha_\pm$ and all the other eigenvalues of A are smaller than $C\sqrt{d}$ in absolute value, so the assignment based on the signs of the most negative eigenvector is close to correct.

To make this precise, let $I = I_{n,p_1,p_2,p_3}$ be a random 3CNF as described above and let $G = (V, E)$ be the graph on the literals of I with edges connecting every pair of literals appearing in a common clause. Let $G' = (V, E')$ be obtained by deleting all edges of G adjacent to a vertex of degree greater than $180d$ and let A be the adjacency matrix of G'. Denote by $\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_{2n}$ the eigenvalues of A and by $\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_{2n}$ a corresponding collection of orthonormal eigenvectors.

Lemma 1 *There is an absolute constant C such that the following hold* **whp***:*

1. $\lambda_1 \geq (6d)\alpha_+ - 2^{-d/C}$
2. $\lambda_{2n} \leq (6d)\alpha_- + 2^{-d/C}$
3. $|\lambda_i| \leq C\sqrt{d}$ *for* $i = 2, \ldots, 2n-1$

Proof The proof is very similar to Lemma 5 of [74] and Proposition 2.1 of [15], which use the techniques of Kahn and Szemerédi from [130]. For a self-contained treatment, see also [114]

Our main tool is Rayleigh's Principle,

$$\lambda_i = \min_L \max_{\mathbf{v} \in L, \mathbf{v} \neq \mathbf{0}} \frac{\mathbf{v}^T A \mathbf{v}}{\mathbf{v}^T \mathbf{v}} \tag{5.1}$$

where L ranges over all dimension $2n - i + 1$ subspaces of $\mathbb{R}^{2n}$. (See, for example, [217]).

We partition the matrix A into 4 blocks, $A_{i,j}$, $i, j \in \{T, F\}$, where $A_{T,T}$ corresponds to the literals l with $\phi(l) = T$, and the other $A_{i,j}$ are defined similarly. The edge-set of G has corresponding partition into $E_{i,j}$ for $i, j \in \{T, F\}$.

Proposition 1 *For the edge sets defined above, the following hold* **whp***: $|E_{T,T}| = (3\eta_3 d + 3\eta_2 d \pm o(1))n$, $|E_{T,F}| = (6\eta_2 d + 6d \pm o(1))n$, $|E_{F,F}| = (3d \pm o(1))n$, and $|E \setminus E'| \leq 2^{-2d/C} n$.*

These all follow from standard calculations which are omitted here.

We are now ready to prove Lemma 1. To prove part (1) we apply Rayleigh's Principle with $\mathbf{v}_+ = \beta_+ \mathbf{v}_T + \gamma_+ \mathbf{v}_F$.

$$\begin{aligned}
\mathbf{v}_+^T A \mathbf{v}_+ &= \beta_+^2 2|E'_{T,T}| + 2\beta_+\gamma_+|E'_{T,F}| + \gamma_+^2 2|E'_{F,F}| \\
&\geq \beta_+^2 2(|E_{T,T}| - 2^{-2d/C}n) \\
&\quad + 2\beta_+\gamma_+(|E_{T,F}| - 2^{-2d/C}n) \\
&\quad + \gamma_+^2 2(|E_{F,F}| - 2^{-2d/C}n) \\
&\geq \beta_+^2(6\eta_2 d + 6\eta_3 d \pm o(1))n \\
&\quad + \beta_+\gamma_+(12d + 12\eta_2 d \pm o(1))n \\
&\quad + \gamma_+^2(6d \pm o(1))n \\
&\quad - 2(|\beta_+| + |\gamma_+|)^2 2^{-2d/C}n \\
&\geq \begin{bmatrix}\beta_+ & \gamma_+\end{bmatrix} \begin{bmatrix}6d(\eta_2+\eta_3) & 6d(1+\eta_2) \\ 6d(1+\eta_2) & 6d\end{bmatrix} \begin{bmatrix}\beta_+ \\ \gamma_+\end{bmatrix} n \\
&\quad - 3(|\beta_+| + |\gamma_+|)^2 2^{-2d/C}n \\
&\geq (6d)\alpha_+ n - 2^{-d/C}n
\end{aligned}$$

Since $\mathbf{v}_+^T \mathbf{v}_+ = n$ we conclude that $\lambda_1 \geq (6d)\alpha_+ - 2^{-d/C}$.

To prove part (2) of the lemma, we apply Rayleigh's Principle with $L = \{t\mathbf{v}_+ \colon t \in \mathbb{R}\}$. The calculation is very similar to the one above, and is omitted.

Proving (3) takes more work. Fortunately we can use a reduction very similar to Lemma 5(iii) in [74]. Recall that $\mathbf{v}_T$ is the vector with $\mathbf{v}_T(\ell) = 1$ if literal ℓ is true and 0 otherwise. Also, recall that $\mathbf{v}_F = \mathbf{1} - \mathbf{v}_T$. We begin by showing

Proposition 2 *For any* $\mathbf{v}$ *with* $\mathbf{v}^T\mathbf{v} = 1$, $\mathbf{v}^T\mathbf{v}_T = 0$, *and* $\mathbf{v}^T\mathbf{v}_F = 0$, *we have* $|\mathbf{v}^T A \mathbf{v}| \leq C\sqrt{d}$.

Proof Observe that the entries of $A_{F,F}$ are 1 independently with probability $1 - (1 - p_1)^{3n} \sim 3d_1/n$. Unfortunately, all the other $A_{i,j}$ have dependencies because the edges are added a triangle at a time. To work around this, for each clause, we randomly color the edges of the triangle corresponding to the clause, one edge red, one green, and one blue. We add each to the appropriately colored graph G^r, G^g, or G^b (but add an edge to at most 1 graph). Note that the edges of a particular G^c appear independently, as each edge is contributed by a different clause. Let A^c be the adjacency matrix of G^c for $c \in \{r, g, b\}$. Note that $A = A^r + A^g + A^b$. Let $A^c_{i,j}$ for $i, j \in \{T, F\}$ be the submatricies of A^c corresponding to the submatricies of A defined above. Then we have

$$|\mathbf{v}^T A \mathbf{v}| \leq \sum_{c\in\{r,g,b\}} |\mathbf{v}^T A^c \mathbf{v}| \leq \sum_{c\in\{r,g,b\}} \sum_{i,j\in\{T,F\}} |\mathbf{v}_i^T A^c_{i,j} \mathbf{v}_j|.$$

The edges of $A^c_{i,j}$ occur with different probabilities for different combinations of i, j, but, provided we have made $d_{min}(\eta_2, \eta_3)$ large enough, all submatricies with non-zero edge

probabilities have edge probabilities exceeding D/n and we can use an argument identical to Lemma 2.4 of [15] (except with $5d$ changed to $180d$) to show that any unit vectors $\mathbf{v}$ with $\mathbf{v}^T\mathbf{v}_T = 0$ and $\mathbf{v}^T\mathbf{v}_F = 0$ has $|\mathbf{v}_i^T A^c_{i,j}\mathbf{v}_j| \leq (C/12)\sqrt{d}$, which implies $|\mathbf{v}^T A\mathbf{v}| \leq C\sqrt{d}$. □

We also need

Proposition 3 *The following hold* **whp**

$$\|(A - (6d)\alpha_+ I)\mathbf{v}_+\|^2 \leq d\|\mathbf{v}_+\|^2,$$
$$\|(A - (6d)\alpha_- I)\mathbf{v}_-\|^2 \leq d\|\mathbf{v}_-\|^2.$$

Proof We work with $\mathbf{v}_+$, as the bound on $\mathbf{v}_-$ is calculated analogously.

Setting $\mathbf{y} = (A - (6d)\alpha_+ I)\,\mathbf{v}_+$, we see that

$$\begin{aligned}\mathbf{y}^T\mathbf{y} &= \mathbf{v}_+^T(A - (6d)\alpha_+ I)^T(A - (6d)\alpha_+ I)\mathbf{v}_+ \\ &= \mathbf{v}_+^T A^2\mathbf{v}_+ - 2(6d)\alpha_+\mathbf{v}_+^T A\mathbf{v}_+ + ((6d)\alpha_+)^2\mathbf{v}_+^T\mathbf{v}_+.\end{aligned}$$

We know that, **whp**, $\mathbf{v}_+^T A\mathbf{v}_+ \geq (6d)\alpha_+ n - 2^{-d/C}n$ from the proof of Lemma 1 (1). By a similar calculation, we have $\mathbf{v}_+^T A\mathbf{v}_+ \leq (6d)\alpha_+ n + 2^{-d/C}n$. To complete the proposition, we calculate that, **whp**, $\mathbf{v}_+^T A^2\mathbf{v}_+ = ((6d)\alpha_+)^2 n \pm 2^{-d/C}n$. (To see this, write $\mathbf{v}_+$ as a linear combination of the eigenvectors of A, and note that the coefficient of the eigenvector corresponding to eigenvalue $(6d)\alpha_+$ must have most of the weight in the sum in order for $\mathbf{v}_+^T A\mathbf{v}_+$ to be close to $(6d)\alpha_+$). Summing up, we see that **whp**

$$\begin{aligned}\mathbf{y}^T\mathbf{y} &= ((6d)\alpha_+)^2 n \pm 2^{-d/C}n - 2(6d)\alpha_+((6d)\alpha_+ n \pm 2^{-d/C}n) + ((6d)\alpha_+)^2 n \\ &= \pm 3\cdot 2^{-d/C}n.\end{aligned}$$

□

We can now complete the lemma. To show $\lambda_2 \leq (C+2)\sqrt{d}$, we apply Rayleigh's Principle with $L = \{\mathbf{v}\colon \mathbf{v}^T\mathbf{v}_+ = 0\}$. Then we write $\mathbf{v} \in L$ as $t\mathbf{v}_- + \mathbf{w}$, where $\mathbf{w}^T\mathbf{v}_+ = 0$ and $\mathbf{w}^T\mathbf{v}_- = 0$. So

$$\begin{aligned}\mathbf{v}^T A\mathbf{v} &= t^2\mathbf{v}_-^T A\mathbf{v}_- + 2t\mathbf{w}^T A\mathbf{v}_- + \mathbf{w}^T A\mathbf{w} \\ &= t^2\mathbf{v}_-^T A\mathbf{v}_- + 2t\mathbf{w}^T(A - (6d)\alpha_- I)\mathbf{v}_- + \mathbf{w}^T A\mathbf{w} \\ &\leq t^2((6d)\alpha_- + 2^{-d/C})\|\mathbf{v}_-\|^2 + 2t\sqrt{d}\|\mathbf{w}\|\|\mathbf{v}_-\| \\ &\qquad + C\sqrt{d}\|\mathbf{w}\|^2 \\ &\leq (C+2)\sqrt{d}\|\mathbf{v}\|^2,\end{aligned}$$

where the final inequality follows from $\alpha_- < 0$, $\|\mathbf{w}\| \leq \|\mathbf{v}\|$, and $t\|\mathbf{v}_-\| \leq \|\mathbf{v}\|$.

To show $\lambda_{2n-1} \geq -C\sqrt{d}$, we let L be any 2 dimensional subspace of $\mathbb{R}^{2n}$. Since L is 2 dimensional, it must contain a unit vector $\mathbf{v}'$ which is orthogonal to $\mathbf{v}_-$. We may express $\mathbf{v}'$ as $c_1\mathbf{v} + c_2\mathbf{v}_+/\sqrt{n}$, where $\mathbf{v}$ is a norm 1 vector that is orthogonal to $\mathbf{v}_+$ and $c_1^2 + c_2^2 = 1$. Since $\mathbf{v}$ is orthogonal to $\mathbf{v}_+$ and $\mathbf{v}_-$, Proposition 2 shows that $\mathbf{v}^T A\mathbf{v}/(\mathbf{v}^T\mathbf{v}) \geq -C\sqrt{d}$.

Then, by using $c_1\mathbf{v} + c_2\mathbf{v}_+/\sqrt{n}$ as a vector to bound Rayleigh's Principle, we have

$$\max_{\mathbf{v}\in L,\mathbf{v}\neq\mathbf{0}} \frac{(c_1\mathbf{v} + c_2\mathbf{v}_+/\sqrt{n})^T A(c_1\mathbf{v} + c_2\mathbf{v}_+/\sqrt{n})}{(c_1\mathbf{v} + c_2\mathbf{v}_+/\sqrt{n})^T(c_1\mathbf{v} + c_2\mathbf{v}_+/\sqrt{n})}$$
$$\geq c_1^2(-C\sqrt{d}) + c_2^2\left((6d)\alpha_+ - 2^{d/C}\right) \geq -(C\sqrt{d}).$$

□

We conclude the section by proving

Lemma 2 *Let λ_{2n} be the most negative eigenvalue of A and let $\mathbf{v}_{2n}$ be the corresponding eigenvector. Let $\mathbf{v}_- = \beta_-\mathbf{v}_T + \gamma_-\mathbf{v}_F$ as above. Then the sign of $\mathbf{v}_{2n}$ or $-\mathbf{v}_{2n}$ disagrees with the sign of $\mathbf{v}_-$ on at most $(C/d)n$ coordinates.*

Proof Expand $\mathbf{v}_-$ as a linear combination of orthonormal eigenvectors of A, so that we have $\mathbf{v}_- = \sum_{i=1}^{2n} c_i\mathbf{v}_i$. Then

$$((6d)\alpha_- I - A)\mathbf{v}_- = \sum_{i=1}^{2n}((6d)\alpha_- - \lambda_i)c_i\mathbf{v}_i$$

and

$$\begin{aligned}\|((6d)\alpha_- I - A)\mathbf{v}_-\|^2 &= \sum_{i=1}^{2n} c_i^2((6d)\alpha_- - \lambda_i)^2 \\ &\geq c_1^2((6d)\alpha_- - \lambda_1)^2 + \sum_{i=2}^{2n-1} c_i^2((6d)\alpha_- - C\sqrt{d})^2 \\ &\geq ((3d)\alpha_-)^2 \sum_{i=1}^{2n-1} c_i^2,\end{aligned}$$

since $\alpha_- < 0$, $\lambda_1 > 0$, and $\lambda_i < C\sqrt{d}$ for $i = 2, \ldots, 2n-1$.

We know from Proposition 3 above that

$$d\|\mathbf{v}_-\|^2 \geq \|((6d)\alpha_- I - A)\mathbf{v}_-\|^2,$$

so

$$\sum_{i=1}^{2n-1} c_i^2 \leq \frac{d}{((3d)\alpha_-)^2}\|\mathbf{v}_-\|^2 \leq \frac{1}{(9d)\alpha_-^2}\|\mathbf{v}_-\|^2 = \frac{1}{(9d)\alpha_-^2}n.$$

Let $\tilde{\mathbf{v}} = \sum_{i=1}^{2n-1} c_i\mathbf{v}_i$, and we have $c_{2n}\mathbf{v}_{2n} = \mathbf{v}_- - \tilde{\mathbf{v}}$. Each entry of $\mathbf{v}_-$ is at least $\sqrt{\frac{1}{17}}$ in absolute value, so $c_{2n}\mathbf{v}_{2n}(\ell)$ may have sign opposite of $\mathbf{v}_-(\ell)$ for at most $\frac{17}{(9d)\alpha_-^2}n$ coordinates.
□

Corollary 1 *After step (3) at least $(1 - C/d)n$ variables are set correctly.*

5.3 Non-spectral Arguments

This section completes the main theorem by analyzing steps (4) and (5) of the algorithm. We choose d_{min} large enough so the truth assignment π produced in step (3) is correct on all but δn variables, where δ is a sufficiently small constant (like 0.001).

To simplify the following discussion we make a few definitions; in the following ψ is a partial truth assignment. Recall that ϕ is the satisfying assignment we used to generate the instance I.

- We say variable x *supports clause* C *with respect to assignment* ψ if x is the only true literal in C with respect to ψ or $\overline{x}$ is the only true literal in C with respect to ψ.
- Let $\mathcal{A}_k$ be the set of variables x such that there are $(3 \pm k\epsilon)d$ clauses which x supports with respect to ϕ. Here ϵ is a sufficiently small constant (eg. $\epsilon = 0.1$).
- Let $\mathcal{B}$ be the set of variables x such that x appears in $(\mu_D \pm \epsilon)d$ clauses, where $\mu_D d$ is the expected number of clauses containing x, which is $(3+6)d+(6+3)\eta_2 d+3\eta_3 d+\mathcal{O}(1/n)$.

We will be concerned with $\mathcal{A}_1$ and $\mathcal{A}_4$, so we may think of $\mathcal{A}_1$ as the variables that support "about the right number of clauses" and $\mathcal{A}_4$ as the variables that support "almost about the right number of clauses" (with respect to ϕ).

We now list some useful properties as in [74] which hold for I **qs**[*]:

Useful Property 1 $|\mathcal{A}_1 \cap \mathcal{B}| \geq n(1 - e^{-d/C})$.

Useful Property 2 There is no subset of variables U such that $|U| \leq 2\delta n$ and at least $\frac{1}{9}\epsilon d|U|$ clauses contain two variables from U.

These follows from standards calculations which are omitted here.

Now we show step (4) improves the assignment found in step (3).

Lemma 3 *After step (4) at least* $(1 - 2^{-d/C})n$ *variables are set correctly* **whp**.

Proof Define the set of variables H as follows:

1. Let $H_1 = \mathcal{A}_1 \cap \mathcal{B}$. Let B be the remaining variables.
2. While there is a variable $a_i \in H_i$ which is in less than $(\mu_D - 2\epsilon)d$ clauses with only variables in H_i, define H_{i+1} to be $H_i \setminus \{a_i\}$.
3. Let a_m be the last variable removed in step (2) and let $H = H_m$.

Proposition 4 $|H| \geq (1 - 2^{-d/C})n$ **qs**.

[*]We say a sequence of events $\mathcal{E}_n$ holds *quite surely* (**qs**) if the probability $\mathbb{P}[\mathcal{E}_n] = o(n^{-C})$ for any constant C.

Proof Useful Property 1 shows that $|H_1| \geq (1 - e^{-d/C})n$ **qs**. Suppose that $m \geq m_0 = e^{-d/C}n$. Let $U = \{a_1, \ldots, a_{m_0}\} \cup B$. Each a_i appears at least $(\mu_D - \epsilon)d$ clauses but at most $(\mu_D - 2\epsilon)d$ clauses with only variables in H, so for each a_i there must be at least ϵd clauses containing a_i and some other variable of U. But each clause can account for at most 3 of the $\epsilon m_0 d$ pairs, so the total number of clauses containing two variable from U is at least $\frac{1}{3}\epsilon|U|/2$, contradicting Useful Property 2.

Therefore, $|H| \geq (1 - 2e^{-d/C})n \geq (1 - 2^{-d/C})n$ **qs**. □

Proposition 5 *Let B_i be the incorrectly assigned variables in H at the i-th iteration of step (4). Then $|B_i| \leq |B_{i-1}|/2$* **qs**.

Proof We will assume not and use Useful Property 2 to derive a contradiction. We know $|B_0| \leq \delta n$ **qs** because step (3) works. If $x \in B_i$, there are 2 cases to consider.

If $x \in B_{i-1}$ then $\pi_{i-1}(x) = \pi_i(x)$ so x appears in at most $5\epsilon d$ clauses unsatisfied by π_{i-1}. But $x \in H$, so there are at least $(3 - \epsilon)d$ clauses which x supports with respect to ϕ, so at least $(3 - 6\epsilon)d$ of these clauses contain some other variable y with $\pi_{i-1}(y) \neq \phi(y)$. Also because $x \in H$, x appears in at most $(\mu_D + \epsilon)d$ clauses, at least $(\mu_D - 2\epsilon)d$ of which include only variables also in H. So x appears in at most $3\epsilon d$ clauses with variables not in H. This means of the $(3 - 6\epsilon)d$ clauses containing another variable which is assigned incorrectly by π_{i-1}, at least $(3 - 9\epsilon)d$ of them contain a variable in B_{i-1}.

If $x \notin B_{i-1}$ then, since it is in B_i, it must be in $5\epsilon d$ clauses which are unsatisfied with respect to π_{i-1}. Since all these clauses are satisfied with respect to ϕ, they must each contain some variable y which has $\pi_{i-1}(y) \neq \phi(y)$. But x is in at most $3\epsilon d$ clauses with variables outside of H, so x is in at least $2\epsilon d$ clauses with some variable in B_{i-1}.

In either case, every variable in B_i appears in at least $2\epsilon d$ clauses with some variable of B_{i-1}. Setting $U = B_i \cup B_{i-1}$, we have at least $\frac{1}{3}2\epsilon d|B_i|$ clauses containing two variables from U. If $|B_i| \geq |B_{i-1}|/2$ then the bound on number of clauses above exceeds $\frac{4}{9}\epsilon d|U|$, which contradicts Useful Property 2. □

This shows that **whp** all literals in H are assigned correctly in $\log n$ iterations, which completes the proof of Lemma 3. □

Lemma 4 *After unassignment in step (5) all variables in H remain assigned and no variable which remains assigned is assigned incorrectly* **whp**.

Proof *All variables in H remains assigned:* all $x \in H$ are assigned correctly at the end of step (4), and there are at most $3\epsilon d$ clauses containing x and variables outside of H, so there are at least $(3 - 4\epsilon)d$ clauses (the ones in H) which x supports and no more than $(3 + 4\epsilon)d$ clauses which x supports. In addition, we know $H \subseteq \mathcal{A}_4$, so all $x \in H$ remain assigned in π'_1. To see that no x is unassigned in later π'_i, note that for $x \in H$, x is in at least $(\mu_D - 2\epsilon)d$ clauses consisting only of other variables in H.

Any variable still assigned after unassignment is assigned correctly: Let U be the set of variables that are assigned incorrectly after unassignment. Suppose $x \in U$. Then x appears in at most $(\mu_D + \epsilon)d$ clauses, of which at most $3\epsilon d$ contain an unassigned variable. Also, x supports at least $(3 - 4\epsilon)d$ clauses, so x supports at least $(3 - 7\epsilon)d$ clauses containing

no unassigned variables. In the correct assignment, x is opposite its current value and all the clauses are satisfied, so each of these $(3-7\epsilon)d$ assigned clauses has some other assigned variable set incorrectly. Thus x appears in $(3-7\epsilon)d$ clauses with some other variable from U. Since each clause can account for at most 3 such pairs, we have at least $\frac{1}{3}(3-7\epsilon)d|U|$ clauses containing two variables of U. $|U| \leq 2^{-d/C}n$ so this contradicts Useful Property 2. □

For the final piece of the argument, consider the graph Γ with a vertex for each variable and an edge between two unassigned variables if they appear in a common clause. We will show Γ has connected components of size at most $\log n$ **whp**. This is proved similarly to Proposition 4 of [74]. The argument is based on a calculation of the expected number of $(\log n)$-sized trees covered by the clauses of I that are disjoint from H.

Lemma 5 *No connected component of* Γ *has size larger than* $\log n$ **whp**.

Proof Let T' be a fixed tree on $\log n$ vertices, and let T be a fixed collection of clauses such that each edge of T' appears in some clause of T. We call T minimal if deleting any clause results in a set which does not cover T'. Let $V(T)$ denote the set of variables appearing in some clause of T and $V(T')$ denote the set of variables appearing in T'. We wish to show that $\mathbb{P}[T \subseteq I \text{ and } V(T') \cap H = \emptyset]$ is small. Let J be the subset of variables of $V(T')$ which appear in at most 6 clauses of T.

Proposition 6 $|J| \geq |V(T')|/2$

Proof Suppose $|J| < |V(T')|/2$. Then at least $|V(T')|/2$ variables appear in more than 6 clauses of T. So $|T| \geq \frac{1}{3} \cdot 6 \cdot |V(T')|/2 = |V(T')|$. But since T is minimal, each clause of T covers at least 1 unique edge of T', so $|T| \leq |V(T')| - 1$. Contradiction. □

We define the set of variables H' by the following iterative procedure (which is similar to the procedure we used to generate H, but depends on $V(T) \setminus J$):

1. Set H'_1 to be the set of variables x such that x supports at least $(3-\epsilon)d$ clauses and at most $(3+\epsilon)d-6$ clauses with respect to ϕ, x appears in at least $(\mu_D-\epsilon)d$ clauses and at most $(\mu_D+\epsilon)d-6$ clauses, and x is not in $V(T) \setminus J$
2. While there exists x_i appearing in less than $(\mu_D-2\epsilon)d$ clauses with only variables from H'_i, set $H'_{i+1} = H'_i \setminus \{x_i\}$.
3. Set H' to H'_m, the final result of the previous step.

Proposition 7 *Let* F *be a set of clauses and let* $H(F \cup T)$ *be the value of* H *if* $I = F \cup T$ *and let* $H'(F)$ *be the value of* H' *if* $I = F$. *Then* $H'(F) \subseteq H(F \cup T)$.

Proof First, we argue that $H'_1(F) \subseteq H_1(F \cup T)$. For $x \notin H_1(F \cup T)$ consider the following cases:

1. If x appears in more than $(\mu_D+\epsilon)d$ clauses of $F \cup T$ or supports more than $(3+\epsilon)d$ clauses of $F \cup T$ with respect to ϕ then it is not included in $H_1(F \cup T)$; we argue x is also not in $H'_1(F)$ by examining 2 cases:

(a) $x \in V(T) \setminus J$. Then x is not included in $H_1'(F)$.

(b) $x \notin V(T) \setminus J$. Then x appears in most 6 clauses of T, so it appears in more than $(\mu_D + \epsilon)d - 6$ clauses of F or x supports more than $(3+\epsilon)d - 6$ clauses of F with respect to ϕ and hence is not included in $H_1'(F)$.

2. If x appears in less than $(\mu_D - \epsilon)d$ clauses of $F \cup T$ or supports less than $(3-\epsilon)d$ clauses of $F \cup T$ with respect to ϕ then, since it appears in no more clauses of F and supports no more clauses of F with respect to ϕ, it is not included in $H_1(F \cup T)$ or $H_1'(F)$.

We proceed by showing that if $H_i'(F) \subseteq H_i(F \cup T)$ then $H_{i+1}'(F) \subseteq H_{i+1}(F \cup T)$: if x_i appears in less than $(\mu_D - 2\epsilon)d$ clauses of $F \cup T$ with only variables of $H_i(F \cup T)$ then it also appears in less than $(\mu_D - 2\epsilon)d$ clauses of F with only variables of $H_i'(F)$.

Thus, we conclude that $H'(F) \subseteq H(F \cup T)$. □

Proposition 8 $\mathbb{P}[T \subseteq I \text{ and } V(T') \cap H = \emptyset] \leq \mathbb{P}[T \subseteq I]\mathbb{P}[J \cap H' = \emptyset]$

Proof It is sufficient to show that

$$\mathbb{P}[J \cap H = \emptyset \mid T \subseteq I] \leq \mathbb{P}[J \cap H' = \emptyset].$$

We do this now:

$$\begin{aligned}\mathbb{P}[J \cap H' = \emptyset] &= \sum_{F:\ J \cap H'(F) = \emptyset} \mathbb{P}[I = F] \\ &\geq \sum_{F:\ J \cap H(F \cup T) = \emptyset} \mathbb{P}[I = F],\end{aligned}$$

where the inequality follows from $H'(F) \subseteq H(F \cup T)$. Now, we break each set of clauses F

into $F' = F \setminus T$ and $F'' = F \cap T$. We rewrite the value above as

$$\begin{aligned}
&\sum_{F:\ J\cap H(F\cup T)=\emptyset} \mathbb{P}[I = F] \\
&= \sum_{\substack{F':\ F'\cap T=\emptyset,\\ J\cap H(F'\cup T)=\emptyset}} \ \sum_{F'':\ F''\subseteq T} \mathbb{P}\left[I \setminus T = F' \wedge I \cap T = F''\right] \\
&= \left(\sum_{\substack{F':\ F'\cap T=\emptyset,\\ J\cap H(F'\cup T)=\emptyset}} \mathbb{P}[I \setminus T = F']\right)\left(\sum_{F'':\ F''\subseteq T} \mathbb{P}[I \cap T = F'']\right) \\
&= \sum_{\substack{F':\ F'\cap T=\emptyset,\\ J\cap H(F'\cup T)=\emptyset}} \mathbb{P}[I \setminus T = F'] \\
&= \sum_{\substack{F':\ F'\cap T=\emptyset,\\ J\cap H(F'\cup T)=\emptyset}} \mathbb{P}[I \setminus T = F' \mid T \subseteq I] \\
&= \mathbb{P}[J \cap H = \emptyset \mid T \subseteq I].
\end{aligned}$$

□

Proposition 9 $\mathbb{P}[J \cap H' = \emptyset] \leq 2n^{-d/2C}$

Proof Although H' is formed by a complicated iterative procedure, this procedure does not depend in any way on J, and so the probability that J does not intersect H' is the same as the probability that any set of size $j = |J|$ does not intersect H'. Conditioned on $|H'|$, this is given by

$$\mathbb{P}\big[J \cap H' = \emptyset \;\big|\; |H'| = h\big] = \binom{n-j}{h} \Big/ \binom{n}{h} \leq (n-h)^j.$$

It follows from the same arguments as in Proposition 4 that $|H'| > (1 - 2^{-d/C})n$ **qs**.

Therefore the unconditional probability that $J \cap H' = \emptyset$ is at most $2^{-jd/C} + n^{-d/2C}$. Since $j = |J| \geq |V(T')|/2 = (\log n)/2$, the desired bound on the probability holds. □

Let $k = \log n$, and let $N_{T',s}$ denote the number of ways to pair $2s$ edges of T' to form s clauses which each cover 2 edges. Since there are 6 ways to permute the order of the variables in each clause, 8 ways to set the negations, and $k - 1 - s$ clauses total, it follows that there are at most $N_{T',s}(48n)^{k-1-2s}$ ways to cover tree T' with a minimal set of clauses such that s clauses cover 2 edges and $k - 1 - 2s$ clauses cover 1 edge. Let T be such a set of clauses. Then we have $\mathbb{P}[T \subseteq I] = (d/n^2)^{k-1-s}$. Above we showed that $\mathbb{P}[J \cap H'] \leq e^{-k(d/2C)}$. Thus, the probability that the random instance I contains a set of clauses which cover a k-tree

that is disjoint from H is at most

$$\sum_{k\text{-trees } T'} \sum_{s=0}^{k/2} N_{T',s}(48n)^{k-1-2s}\left(d/n^2\right)^{k-1-s} e^{-k(d/2C)}$$
$$\leq \sum_{k\text{-trees } T'} \left(\sum_{s=0}^{k/2} N_{T',s}\right)(48d)^k n^{1-k} e^{-k(d/2C)}$$

In order to obtain a useful upper bounds on the sum $\left(\sum_s N_{T',s}\right)$, we fix a degree sequence $(d_1,\ldots,d_k)$ for T', and consider the following procedure for pairing edges so that triangles can cover the edge pairs. For each vertex, we specify a permutation of the edges incident to that vertex. Then we iterate through the vertices, and for each vertex, we iterate through the edges and pair up each unpaired edge with the edge given by the permutation associated with the current vertex (and leave the edge unpaired if the permutation sends the edge to itself). Any pairing of edges which can be covered by clauses can be generated this way by choosing the permutations to transpose each pair of edges to be covered by a single clause and to leave fixed all the other edges. Since there are $d_i!$ different permutations for vertex i, we have

$$\sum_{s=0}^{k/2} N_{T',s} \leq \prod_{i=1}^{k} (d_i!)\,.$$

Prüfer codes give a bijection between the set $[k]^{k-2}$ and labeled trees on k vertices. They have the additional nice property that the degree of vertex i in the tree corresponding to code $c \in [k]^{k-2}$ is exactly 1 less than the number of times i appears in c. It follows that the number of k-trees with degree sequence $(d_1,\ldots,d_k)$ equals $\binom{k-2}{d_1-1,\ldots,d_k-1}$ (see, for example, [185, Section 4.1, p. 33]). There are $\binom{n}{k}$ ways to choose the k vertices of the tree. So the probability above is at most

$$\sum_{d_1+\ldots+d_k=2(k-1)} \binom{k-2}{d_1-1,\ldots,d_k-1}\binom{n}{k}\left(\prod_{i=1}^{k}(d_i!)\right)(48d)^k n^{1-k} e^{-k(d/2C)}$$
$$\leq \sum_{d_1+\ldots+d_k=2(k-1)} k^2 e^k \left(\prod_{i=1}^{k} d_i\right)(48d)^k n e^{-k(d/2C)}.$$

For $(d_1,\ldots,d_k)$ with $d_1+\ldots+d_k = 2(k-1)$, the product $\prod_{i=1}^k d_i$ is maximized when $d_1 = \ldots = d_k$ and so $\prod_{i=1}^k d_i < 2^k$. The number of ways to choose positive integers $(d_1,\ldots,d_k)$ so that $d_1+\ldots+d_k = 2(k-1)$ is less than $\binom{2k-1}{k-1}$, which is less than 2^{2k}. So, provided we have chosen the constant d sufficiently large, we find that the probability that I contains a set of clauses which covers a $(\log n)$-tree disjoint from H is at most

$$2^{2\log n}(\log n)^2 e^{\log n} 2^{\log n}(48d)^{\log n} n e^{-\log n(d/2C)} = o(1).$$

Chapter 6

Subset Sum on Medium-Dense Random Instances

This chapter originally appeared as [124]. It studies the performance of a heuristic for solving the subset sum problem, which has been proposed as a cryptographic primative both in theory and in practice.

6.1 Introduction

The subset sum problem (SSP), one of the classical NP-hard problems, is defined as follows: given n numbers and a target bound B, find a subset of the numbers whose sum equals B.

This chapter considers a case arising commonly in cryptographic applications where the numbers are represented by m-bit integers, and the sums are computed modulo M, where M is another m-bit integer. In other words, the addition is performed in $\mathbb{Z}_M$. More formally, the subset sum problem of dimensions n and m is:

GIVEN: n numbers $a_1, \ldots, a_n$, with $a_i \in \mathbb{Z}_M$, and a target $B \in \mathbb{Z}_M$, where M is an m-bit integer

FIND: a subset $S \subset \{1, \ldots, n\}$, such that

$$\sum_{i \in S} a_i \equiv B \mod M \ .$$

For the purposes of average case analysis, consider random instances of the problem, where both the input numbers and the bound are picked uniformly at random from $\mathbb{Z}_M$, where M is a function of n. Similar random instances (with a different dependence of M on n) were shown by Chvátal [80] to be hard instances for a class of knapsack algorithms.

The hardness of random SSP instances varies significantly with the choice of parameters, in particular the magnitude of m (the number of bits in M) as a function of n (this is described in detail in [152] and bears some relation to the phase transition phenomenon studied for Integer Partitioning problems in [59, 58]):

$m > n$**:** such instances are "almost 1-1" (each subset has a different sum), and are efficiently solvable by a reduction to a short vector in a lattice when $m \geq c \cdot n^2$, for some constant c [181, 136, 94].

$m < n$**:** such instances are "almost onto" (with multiple solutions for most targets), and are efficiently solvable by various techniques in high-density case, i.e., for $m = \mathcal{O}(\log n)$ (by dynamic programming, or, when $M = \mathcal{O}(n^2/\log n)$ by using methods of analytical number theory [69, 139, 68]).

Despite various efficient approaches to dense instances, prior to the initial publication of the results in this chapter [124], all algorithms take at least $\Omega(M)$ time, and so none of them works in polynomial time when $m = \omega(\log n)$. Independently from the work in [124], Lyubashevsky gave an alternative approach for random instances of subset sum, which runs in expected polynomial-time on instances with $m = c(\log n)^2$ for any constant c, and also runs in subexponential time on instances with $m = o(n)$ **whp** [188].

6.1.1 Related Work

The algorithm below bears some similarity to an approach developed by Blum *et al.* [38] in the context of computational learning theory. By employing a recursive approach much like that below, they provide an algorithm for learning an XOR function in the presence of noise. Their work began in a similar average-case setting, but was subsequently extended to work on arbitrary (worst-case) instances.

Beier and Vöcking [31] presented an expected polynomial time algorithm for solving random knapsack instances. Knapsack and subset sum have some compelling similarities, but the random instances considered there are quite different from those considered here, and this leads to the development of quite a different approach, bearing more similarity to the study of heuristics for real-values knapsack and partitioning problems [187, 186].

6.1.2 Notation and Conventions

A tuple $(a_1, \ldots, a_n; B, M)$ denotes an instance of SSP with input numbers a_i and target B to be solved over $\mathbb{Z}_M$.

For the clarity of presentation the discrete nature of some terms in summations are ignored to avoid the use of rounding operations (floors and ceilings). However, this simplification does not compromise the validity of the results. All asymptotic notation is with respect to n, and all logarithms are base 2.

6.2 The New Algorithm

We begin with a special case, an algorithm applicable when M is a power of 2. Then we present another special case, an algorithm applicable when M is odd. In general, we apply a combination of the two special cases. Given any modulus M we write $M = \bar{M} \cdot M'$, with $\bar{M} = 2^{\bar{m}}$ and M' odd. We use the first algorithm to reduce the original problem

$(a_1, \ldots, a_n; B, M)$ to a problem $(a'_1, \ldots, a'_{n'}; B', M')$, and then use the second algorithm to solve the reduced problem.

In the algorithms below ℓ is a parameter whose value will later be set to $(\log n)/2$. For simplicity, the description presented below focuses on the core part of the algorithms, which can fail on some inputs. Later we show that the failures have sufficiently low probability so that upon failure we can run a dynamic programming algorithm (which takes exponential time) and obtain an *expected* polynomial time algorithm.

6.2.1 Subset Sum Modulo Power of 2

Given an instance $(a_1, \ldots, a_n; B, M)$, with $M = 2^m$ and $B \neq 0$, we transform it to an equivalent instance with target zero, i.e., $(a_1, \ldots, a_n, a_{n+1}; 0, M)$, where $a_{n+1} = M - B$ and we require that a valid solution contain this additional element a_{n+1}. To solve the target-zero instance we proceed as follows: we find among the input numbers a maximum matching containing a_{n+1}, where two numbers a_i, a_j can be matched if the sum $(a_i + a_j)$ has its ℓ least significant bits equal to zero, (in other words, if $(a_i + a_j) \equiv 0 \mod 2^\ell$.) From the matching we generate a "smaller" instance of SSP, which we solve recursively: given a matching of size s, $((a_{i_1}, a_{j_1}), \ldots, (a_{i_s}, a_{j_s}))$, where wlog. $a_{i_s} = a_{n+1}$, we generate an instance $((a_{i_1} + a_{j_1})/2^\ell, \ldots, (a_{i_s} + a_{j_s})/2^\ell; 0, 2^{m-\ell})$, and we require that a valid solution of this instance must contain the last element. Note that the instance to be solved recursively is indeed smaller. It has at most $(n + 1)/2$ input numbers, and both the modulus and the input numbers are shorter by ℓ bits. When the recursion reaches the bottom, we extract a solution of the original problem in a straightforward way. Figure 6.1 presents the algorithm in pseudocode. Note that the algorithm returns a set $\mathcal{S}$ of disjoint subsets, where the the last subset is a solution to the input problem, and all remaining subsets sum up to zero modulo 2^m. These extra subsets are used in the combined algorithm in Sect. 6.2.3.

We remark that the above method can be used to solve instances of SSP with some other moduli, for example when M is a power of small primes, or when M is "smooth" (meaning the product of small primes). However, the method does not generalize easily to arbitrary moduli, and in particular gives no obvious way to handle a large prime modulus. In the next section we describe a different algorithm, which works with high probability for arbitrary *odd* moduli.

6.2.2 Subset Sum With An Odd Modulus

The algorithm for SSP with an odd modulus has on a high level the same strategy as the algorithm from the previous section, i.e., it successively reduces the size of the numbers by matching them in pairs. However, it differs in one significant detail. Instead of working on least significant bits, it zeros out the most significant bits at each step of the recursion.

Given an instance $(a_1, \ldots, a_n; B, M)$, with M odd and $B \neq 0$, we begin, as in the previous case, by transforming it to an equivalent instance with target 0. However, this time we use a different transformation. To each input number we add the value $\Delta := (-B/2^t) \mod M$, where $t = \lceil \log_2 M/\ell \rceil$, so the modified instance is $(a'_1, \ldots, a'_n; 0, M)$, where $a'_i = a_i + \Delta$.

procedure SSPmod2$(a_1, \ldots, a_n, B, m, \ell)$
  $a_{n+1} := -B$
  $\mathcal{S} :=$ SSPmod2rec$(a_1, \ldots, a_{n+1}, m, \ell)$
  /** *wlog.* $\mathcal{S} = (S_1, \ldots, S_s)$ *and* $(n+1) \in S_s$ **/
  return $(S_1, \ldots, S_{s-1}, S_s \setminus \{n+1\})$

procedure SSPmod2rec$(a_1, \ldots, a_{n+1}; m, \ell)$
  $\mathcal{S} := ()$
  $V := \{1, \ldots, n, n+1\}$
  $E := \{(i,j) : (a_i + a_j) \equiv 0 \mod 2^\ell\}$
  $E' :=$ maximum matching in $G = (V, E)$ containing vertex $(n+1)$
  /** *wlog.* $E' = (e_1, \ldots, e_s)$, *with* e_s *containing* $(n+1)$ **/
  if E' is non-empty **then**
    if $\ell < m$ **then**
      $\forall e_k \in E'$, $e_k = (i_k, j_k)$, **let** $a'_k := (a_{i_k} + a_{j_k})/2^\ell$
      $\mathcal{S}' :=$ SSPmod2rec$(a'_1, \ldots, a'_s; m - \ell, \ell)$
      if $\mathcal{S}'$ is not empty **then**
        /** *wlog.* $\mathcal{S}' = (S'_1, \ldots, S'_t)$, *with each* $S'_i \subseteq \{1 \ldots s\}$, *and* $s \in S'_t$ **/
        $\forall S'_i \in \mathcal{S}'$ **let** $S_i := \bigcup_{e_k : k \in S'_i, e_k = (i_k, j_k)} \{i_k, j_k\}$
        $\mathcal{S} := (S_1, \ldots S_t)$
    else
      $\forall e_k \in E'$, $e_k = (i_k, j_k)$, **let** $S_k := \{i_k, j_k\}$
      $\mathcal{S} := (S_1, \ldots S_s)$
  return $\mathcal{S}$

Figure 6.1: Algorithm for solving dense SSP instances modulo a power of 2

Our motivation for making this transformation becomes clear when we reveal our plan to make sure that any solution returned by our algorithm contains exactly 2^t elements. Since the sum of the solution of the modified instance is zero modulo M, the sum of the corresponding numbers in the original instance is B, as each number of the solution contributes an extra Δ to the sum and

$$2^t \cdot \Delta \equiv -B \mod M.$$

The fact that M is odd is required to ensure that such a Δ exists.

Now it is convenient to view elements from $\mathbb{Z}_M$ as numbers from the interval $I = \{-(M-1)/2, \ldots, (M-1)/2\}$, following the transformation

$$a \to \begin{cases} a, & \text{if } a \le (M-1)/2; \\ a - M, & \text{otherwise.} \end{cases} \tag{6.1}$$

Given a target-zero instance $(a'_1, \ldots, a'_n; 0, M)$ with M odd, we find a solution of cardinality 2^t as follows: we find a maximum matching among the input numbers, where two numbers a'_i, a'_j can be matched iff there exists an integer k so that when viewed as elements of the interval I, as in (6.1), $a'_i \in [kM/2^{\ell+1}, (k+1)M/2^{\ell+1}]$ and $a'_j \in [-(k+1)M/2^{\ell+1}, -kM/2^{\ell+1}]$. Again, from the matching we generate a "smaller" instance of SSP, which we solve recursively. Given a matching of size s,

$$((a'_{i_1}, a'_{j_1}), \ldots, (a'_{i_s}, a'_{j_s})),$$

we generate an instance $((a'_{i_1} + a'_{j_1}), \ldots, (a'_{i_s} + a'_{j_s}); 0, M)$. By the property of the matched numbers, the input numbers of the new instance are smaller in the sense that they are closer to 0 when viewed as elements of interval I. Figure 6.2 presents in pseudocode the algorithm for odd moduli.

---

procedure SSPmodOdd$(a_1, \ldots, a_n; B, M, \ell)$
  $t := \lceil \log_2 M/\ell \rceil$
  $\Delta := (-B/2^t) \mod M$
  return SSPmodOddRec$(a_1 + \Delta, \ldots, a_n + \Delta; M, \ell, 1)$

procedure SSPmodOddRec$(a_1, \ldots, a_n; M, \ell, d)$
  /** *we view a_i's as numbers from $I = \{-(M-1)/2, \ldots, (M-1)/2\}$* **/
  $S := \{\}$
  $V := \{1, \ldots, n\}$
  $E := \{(i,j) : \exists k \in \mathbb{Z}, a_i \in [kM/2^{d\ell+1}, (k+1)M/2^{d\ell+1}],$
    $a_j \in [-(k+1)M/2^{d\ell+1}, -kM/2^{d\ell+1}]\}$
  $E' :=$ maximum matching in $G = (V, E)$
  /** *wlog.* $E' = (e_1, \ldots, e_s)$ **/
  if E' is non-empty **then**
    if $d \cdot \ell < \lceil \log_2 M \rceil$ **then**
      $\forall e_k \in E'$, $e_k = (i_k, j_k)$, **let** $a'_k := (a_{i_k} + a_{j_k})$
      $S' :=$ SSPmodOddRec$(a'_1, \ldots, a'_s; M, \ell, d+1)$
      if S' is not empty **then**
        /** $S' \subseteq \{1 \ldots s\}$ **/
        $S := \bigcup_{e_k : k \in S', e_k = (i_k, j_k)} \{i_k, j_k\}$
    else
      $S := \{i_1, j_1\}$, where $e_1 \in E'$, $e_1 = (i_1, j_1)$.
  return S

Figure 6.2: Algorithm for solving dense SSP instances with an odd modulus

6.2.3 Combined Algorithm

As mentioned above, given an instance $(a_1, \ldots, a_n; B, M)$, for any (m-bit) modulus M, we write $M = \bar{M} \cdot M'$, with $\bar{M} = 2^{\bar{m}}$ and M' odd, and apply both algorithms described above, one for $\bar{M}$ and one for M'.

First, we solve the instance $(a_1, \ldots, a_n; B, \bar{M})$ using procedure SSPmod2. As a solution, we obtain a sequence $\mathcal{S} = (S_1, \ldots, S_s)$ of disjoint subsets of $\{1, \ldots, n\}$, where for each $i = 1..(s-1)$ we have $\sum_{j \in S_i} a_j \equiv 0 \mod \bar{M}$, and $\sum_{j \in S_s} a_j \equiv B \mod \bar{M}$(i.e., the last subset is a solution for target B). From this solution we generate an instance for the second algorithm, $(a'_1, \ldots, a'_{n'}; B', M')$, where $n' = s - 1$, $a'_i = (\sum_{j \in S_i} a_j) \mod M'$ for $i = 1..n'$, and $B' = B - \sum_{j \in S_s} a_j$. The second algorithm returns a solution $S' \subseteq \{1, \ldots, n'\}$, from which we derive our answer

$$S_s \cup \left(\bigcup_{j \in S'} S_j \right)$$

Figure 6.3 presents the combined algorithm in pseudocode.

---

procedure DenseSSP$(a_1, \ldots, a_n; B, M, \ell)$
  $S := \{\}$
  find M' and $\bar{M} = 2^{\bar{m}}$ such that M' is odd and $M = 2^{\bar{m}} \cdot M'$
  $\mathcal{S} :=$ SSPmod2$(a_1, \ldots, a_n, B, \bar{m}, \ell)$ /** *here* $\mathcal{S} = (S_1, \ldots, S_s)$, *with* **/
    /** $\sum_{j \in S_i} a_j \equiv 0 \mod \bar{M}$ *for* $i = 1..(s-1)$, *and* **/
    /** $\sum_{j \in S_s} a_j \equiv B \mod \bar{M}$ **/
  if $\mathcal{S}$ is not empty **then**
    $\forall i = 1..(s-1)$ **let** $a'_i := (\sum_{j \in S_i} a_j) \mod M'$
    $B' := B - \sum_{j \in S_s} a_j$
    $S' :=$ SSPmodOdd$(a'_1, \ldots, a'_{s-1}; B', M', \ell)$ /** *here* $S' \subseteq \{1, \ldots, n'\}$ **/
    if S' is not empty **then**
      $S := S_s \cup \left(\bigcup_{j \in S'} S_j \right)$
  return S

Figure 6.3: Algorithm for solving dense SSP instances

6.3 Analysis

6.3.1 Correctness

We need to argue that any non-empty subset returned by the algorithm is a valid solution.

First consider computation modulo $\bar{M}$ which is a power of 2. At each level of recursion we match pairs of input numbers so that ℓ least significant bits are zeroed, while respecting the constraint, that the last input number is matched. Therefore, in recursive call, we have zeroed the least significant bits of the resulting numbers, so it follows by induction that all

the subsets returned by SSPmod2rec sum up to $0 \mod \bar{M}$.

Moreover, we need to argue that the last subset returned by SSPmod2rec determines a solution for the given target B. Indeed, if S_s is the last subset returned by SSPmod2rec, then $(n+1) \in S_s$ and $\sum_{i \in S_s} a_i \equiv 0 \mod \bar{M}$. Since $a_{n+1} = -B$, this implies that $\sum_{i \in S \setminus \{n+1\}} a_i \equiv B \mod \bar{M}$, as desired.

To prove the correctness of the computation modulo an odd number M', note that the transformation to a target-zero instance gives the desired result: any solution is created bottom-up, by first matching two input numbers, than matching two pairs matched previously, and so on, i.e., at each recursion level the number of the numbers in a solution is doubled, so the size of the solution subset is equal 2^t, where t is the depth of recursion, which, in turn, equals $\lceil \log_2 M'/\ell \rceil$. Therefore, any solution with target zero will have exactly $2^t \cdot \Delta \equiv -B \mod M'$ of "extra" sum, i.e., the corresponding original numbers sum up to $B \mod M'$.

Further, since the algorithm matches the numbers which have opposite signs but "close" magnitudes, at each level of recursion a portion of ℓ *most* significant bits is zeroed, while avoiding the problems of overflows and wrap-arounds when adding the numbers. Hence, by induction, the subset returned by SSPmodOddRec sums up to $0 \mod M'$

The correctness of the combined argument follows from the above arguments and from the Chinese Remainder Theorem, since $M = \bar{M} \cdot M'$, where $\bar{M}$ and M' are relatively prime.

6.3.2 Success Probability

We consider the cases with magnitudes m up to $(\log n)^2/16$ and we set $\ell = (\log n)/2$. At a given level in the recursion, in both cases (power of 2 and odd), the success of the algorithm at that level depends on the number of numbers in the recursion being "enough". And, the number of numbers in the recursion at a given level is equal to the number of edges in the matching at the previous level. We will argue by induction. Let t_k denote the number of numbers available at the beginning of level k of the recursion, and let s_k denote the number of edges in the matching at level k.

Lemma 6 *For s_k and t_k defined as above, Let $\mathcal{A}_k$ denote the event that $s_k \geq t_k/4$. Then*

$$\mathbb{P}[\mathcal{A}_k \mid \mathcal{A}_1, \ldots, \mathcal{A}_{k-1}] \leq \exp\left(-n^{3/4}/32\right).$$

Proof If $\mathcal{A}_1, \ldots, \mathcal{A}_{k-1}$ occur (meaning, in every previous level of the recursion, we have managed to keep at least 1/4 of the numbers), then we begin level k with at least $n(1/4)^{(\log n)/8} = n^{3/4}$ numbers (since there are at most $m/\ell \leq (\log n)/8$ levels of recursion total).

Lemma 6 is easier to argue when the modulus is a power of 2. Then the subinstances are formed by zeroing least significant bits, and so the reduced numbers are independent and uniformly random. When the modulus is an odd, the reduced numbers are independent but not uniformly random. Fortunately, they are distributed symmetrically, in the sense that $\mathbb{P}[a'_i = a] = \mathbb{P}[a'_i = -a]$. We argue this by induction: Suppose $\mathbb{P}[a_i = a] = \mathbb{P}[a_i = -a]$

for all i. Then, since each edge $(i,j) \in E$ yields an $a'_k = a_i + a_j$, we have

$$\begin{aligned}\mathbb{P}[a'_k = a] &= \sum_b \mathbb{P}[a_i = b]\mathbb{P}[a_j = a - b] \\ &= \sum_b \mathbb{P}[a_i = -b]\mathbb{P}[a_j = -(a-b)] \\ &= \mathbb{P}[a'_k = -a].\end{aligned}$$

This symmetry property is all that we need to show s_k is very likely to exceed $t_k/4$. We can pretend the t_k input numbers are generated by a two-step process: first, we pick the absolute value of the numbers constituting the instance, and then we pick the sign of each number. Since the distribution is symmetric, in the second step each number in the instance becomes negative with probability 1/2.

Let T_i denote the number of numbers picked in the first step with absolute value in the interval $[(i-1)M/L^d, iM/L^d]$, where $L = 2^\ell$ and $i = 1 \dots L$. Then the number of *negative* numbers in interval i is a random variable $X_i \sim \mathrm{Bi}(T_i, 1/2)$, and we can match all but $Y_i := |X_i - (T_i - X_i)|$ numbers in interval i. Further,

$$\mathbb{E}[Y_i] = \sum_{k=1}^{T_i} \mathbb{P}[Y_i \geq k] = \sum_{k=1}^{T_i} \mathbb{P}[|X_i - T_i/2| \geq k/2],$$

and by Azuma's inequality, this is at most

$$\sum_{k=1}^{T_i} 2e^{-k^2/(2T_i)} \leq \int_{x=0}^{\infty} 2e^{-x^2}\sqrt{2T_i}dx = \sqrt{2\pi T_i}.$$

Let Y denote the total discrepancy of all the bins,

$$Y = \sum_{i=1}^{L} Y_i.$$

By linearity of expectation, we have that we expect to match all but

$$\mathbb{E}[Y] = \mathcal{O}(\sqrt{T_1} + \cdots + \sqrt{T_L})$$

numbers. This sum is maximized when $T_1 = \cdots = T_L$, and minimized when $T_i = t$ for some i (and all other T_j's are zero), hence

$$\mathcal{O}(\sqrt{t}) \quad \leq \mathbb{E}[Y] \leq \quad \mathcal{O}(\sqrt{tL})\ . \tag{6.2}$$

Changing a single number in the instance can change the discrepancy by at most 2, so we use Azuma's inequality in a convenient formulation given by McDiarmid [190] (see also Bollobás [44]) and the fact that $L = 2^\ell = \sqrt{n}$ and $t \geq n^{3/4}$.

$$\begin{aligned}\mathbb{P}[s \le t/4] &= \mathbb{P}[Y \ge t/2] \\ &\le \mathbb{P}[Y \ge \mathbb{E}[Y] + t/4] \\ &\le e^{-t/32} \\ &\le \exp\left(-n^{3/4}/32\right).\end{aligned}$$

□

Then, in the case of an odd modulus, the failure probability is bounded by

$$\mathbb{P}[\text{failure}] \le \sum_{k=1}^{m/\ell} \mathbb{P}[\mathcal{A}_k \mid \mathcal{A}_1, \ldots, \mathcal{A}_k] \le (\log n) \exp -n^{3/4}/32 = \mathcal{O}(e^{-\sqrt{n}}).$$

If the modulus is a power of 2, we must also account for the possibility of failure due to not matching the special number a_{n+1} at some stage. Let $\mathcal{B}_k$ denote the event that the special number is not matched at stage k. This type of failure only occurs if all $t_k - 1$ other numbers are different from the special number. Since the mod 2 reductions keep the numbers at stage k uniformly distributed among $m - k\ell$ possibilities, the probability of $\mathcal{B}_k$ given t_k is $\left(1 - \frac{1}{m-k\ell}\right)^{t_k - 1}$ and if $\mathcal{A}_1, \ldots, \mathcal{A}_{k-1}$ hold, this is at most $\exp\left(-(\log n)^{-2} n^{3/4}\right)$. So again, the probability of failure is $\mathcal{O}(e^{-\sqrt{n}})$.

6.3.3 Running Time

The running time of the algorithm above is dominated by the time required to solve all the subinstances, which is bounded by $(n-1)/(\ell-1) \cdot \mathcal{O}(2^\ell) = \mathcal{O}(n^{3/2})$.

In the case of failure, we can solve the instance by dynamic programming in time $O(2^{(\log n)^2})$. Since (when n is sufficiently large) the failure probability is much less than $2^{-(\log n)^2}$, combining the algorithm above with a dynamic programming backup for failures yields a complete algorithm that runs in expected polynomial time.

6.3.4 Choice of Parameters

The parameters above are not optimized, but there is a curious feature in the proof of Lemma 6 that puts a restriction on the range of coefficients c that would work for $m = c(\log n)^2$. Similarly, the range of constants c' that would work for $\ell = c' \log n$ is restricted in a way that does not seem natural. For $\ell = (\log n)/2$ and $m = (\log n)^2/16$, the number of stages of recursion is small enough that each stage has sufficiently many numbers to succeed with high probability. But for $\ell = (\log n)/2$ and $m = (\log n)^2/8$, McDiarmid's version of Azuma's inequality will not work in the way we have used it.

This restriction has been removed by the subsequent work of Lyubashevsky [188], which is an extension of an alternative approach to problems of this type due to Wagner [224].

6.4 Conclusions and Open Problems

We presented an expected polynomial time algorithm for solving uniformly random subset sum problems of medium density over $\mathbb{Z}_M$, with m bounded by $\mathcal{O}((\log n)^2)$, where n is the number of the input numbers. As far as we are aware, this is the first algorithm for dense instances that works efficiently beyond the magnitude bound of $\mathcal{O}(\log n)$, thus narrowing the interval with hard-to-solve SSP instances. A natural open question is whether the bound on the magnitude can be further extended, e.g. up to $(\log n)^z$ for some $z > 2$. Extending the bound in this manner would yield improved results in quantum algorithms for solving the shortest vector problem [22, 197].

Finally, recall that DenseSSP is a deterministic algorithm which can fail with non-zero probability. Since this probability is very low, upon failure we can run a dynamic programming algorithm and still obtain expected polynomial time in total. A different way of handling failures might be to run DenseSSP again on randomly permuted input. Note however that such multiple trials are not fully independent, thus complicating the analysis. It is an interesting problem to compare this alternative approach with the one we have analyzed.

Chapter 7

The diameter of randomly perturbed digraphs

This chapter originally appeared as [121]. It studies the effects of perturbing a graph by XORing it with a vary sparse random graph distributed according to $\mathbb{G}_{n,\epsilon/n}$. Changing ϵn random edges does not change a graph much, but if the original graph is connected, then the resulting graph will have logarithmic diameter, if it is also connected.

7.1 Introduction

The diameter of a graph G is the length of the longest shortest path in G. In other words, if $d(u, v)$ is the length of the shortest path from u to v in G, then the diameter of G is $\max_{u,v} d(u, v)$. A graph is connected (and a directed graph is strongly connected) if it has finite diameter. The central observation of this chapter is that if ϵn random edges are added to any n-node connected graph with degree not-too-large then the diameter becomes $\mathcal{O}(\ln n)$ with high probability (where "with high probability" means with probability tending to 1 as $n \to \infty$ and is abbreviated **whp**). This is also true for strongly connected directed graphs and digraphs with not-too-large in-degree and out-degree and for several ways of generating the random edges. For ease of exposition, we state this as a theorem only for a strongly connected digraph $\bar{D}$ with degree $\mathcal{O}(\ln n)$ that is perturbed by adding a random digraph $R \sim \mathbb{D}_{n,\epsilon/n}$. Here $R \sim \mathbb{D}$ means R is distributed according to distribution $\mathbb{D}$, and $\mathbb{D}_{n,p}$ is the distribution over of digraphs on vertex set $[n]$ in which each possible arc appears independently with probability p (so each digraph with m arcs is realized with probability $\binom{n(n-1)}{m} p^m (1-p)^{n(n-1)-m}$). Below, we use the notation $D = \bar{D} + R$ to mean that D is the graph formed by taking the union of the arcs of $\bar{D}$ and R.

Theorem 3 *Let ϵ be a positive constant with $\epsilon \leq 1$ and let $\Delta \leq n^{\epsilon/100}$. Let $\bar{D}$ be a strongly connected n-node digraph with in-degree and out-degree at most Δ. Let $D = \bar{D} + R$ where $R \sim \mathbb{D}_{n,\epsilon/n}$. Then* **whp** *the diameter of D is at most $100\epsilon^{-1} \ln n$.*

Similar results hold for perturbations formed by adding ϵn arcs selected at random with or without replacement, or by adding a random assignment with ϵn arcs.

Theorem 3 is related to a class of problems regarding the possible change of diameter in a graph where edges are added or deleted, for example, the results of Alon, Gryárfás, and Ruszinkó in [14] on the minimum number of edges that must be added to a graph to transform it into a graph of diameter at most d. The study of these extremal diameter alteration questions was initiated by Chung and Garey in [79]. It is also related to the theorem of Bollobás and Chung on the diameter of a cycle plus a random matching [49].

7.1.1 Application: Smoothed Analysis

A digraph is strongly connected if, for every vertex pair (s, t), there is a directed path from s to t. Recognizing strongly connected digraphs is a basic computational task, and the set of strongly connected digraphs is **NL**-complete [159]. Thus, if $\mathbf{NL} \not\subseteq \mathbf{L}$ then there is no log-space algorithm which recognizes strongly connected digraphs.

Perhaps this conclusion of worst-case complexity theory is too pessimistic. We will consider the performance of a simple heuristic which runs in randomized log-space. We will show that the heuristic succeeds on random instances **whp**. However, the "meaning" of this result depends on the probability space from which we draw the random instances. It seems reasonable to assume that most real-world digraphs will contain some amount of randomness, so it is tempting to believe this result shows that in the real-world strong connectivity only requires log-space. Unfortunately, this is not valid if we use the "wrong" model for randomness. For example, the distribution $\mathbb{D}_{n,p}$ (which is generated by taking n nodes and including each ordered pair as an arc independently with probability p) is pleasant for analysis, but basic statistics like the degree sequence seem to differ from several observed instances of real-world graphs [108].

We will use a model of randomness that is more flexible. We will start with an arbitrary digraph $\bar{D}$ and perturb it by XORing it with a very sparse random graph $R \sim \mathbb{D}_{n,\epsilon/n}$. This produces a random instance which is "less random" than $\mathbb{D}_{n,p}$. The study of worst case instances with small random perturbations is called Smoothed Analysis.

Smoothed Analysis was introduced by Spielman and Teng in [213] (the journal version is also available [215]) and they discuss a perturbation model for discrete problems in [214]. They consider perturbing graphs by XORing the adjacency matrix with a random adjacency matrix, where each edge is flipped with some constant probability. Since the probability of an edge flip is constant, the perturbed instances are all dense graphs (i.e. a constant fraction of all possible edges appear). Independently, Bohman, Frieze and Martin [41] studied the issue of Hamiltonicity in a dense graph when random edges are *added*, and other graph properties were analyzed in this model by Bohman, Frieze, Krivelevich and Martin [40] and Krivelevich, Sudakov, and Tetali [176].

We will also use an XOR perturbation, but we will make the probability of corruption much lower than [214]. Since we will have a linear number of arcs present, it is appropriate for the perturbation to change about ϵn arcs, which is the expected number of arcs in $\mathbb{D}_{n,\epsilon/n}$.

Randomness and strong connectivity

Recognizing strongly connected digraphs is closely related to recognizing (s,t)-connectivity in digraphs, which is the canonical **NL**-complete problem. It is possible to recognize connectivity in undirected graphs with a randomized log-space algorithm using random walks [13] (also, a deterministic log-space algorithm was recently discovered [205]). Since the cover time of an arbitrary connected graph is bounded by $\mathcal{O}(n^3)$ (see [110, 109] for a sharp bound), a random walk will visit every vertex in polynomial time **whp**. This approach will not work for arbitrary digraphs, however, since there the cover time can be exponential.

The diameter and connectivity of random graphs has been well-studied, see for example the books of Bollobás [46] and Janson, Łuczak, and Ruciński [156]. Perhaps closest in spirit to our investigation is the paper of Bollobás and Chung on the diameter of a Hamilton cycle plus a random matching [49] and the paper of Chung and Garey on the diameter of altered graphs [79]. Also, the component structure of random digraphs was studied by Karp in [165] and more recently by Cooper and Frieze [91].

A heuristic for recognizing strong connectivity

Figure 7.1 describes a simple heuristic to decide if a digraph is strongly connected.

In words, the heuristic is as follows. For each ordered pair of vertices (s,t) repeat the following procedure N_1 times: Starting from s, take N_2 steps in a random walk on the digraph. Here a random walk is the sequence of vertices $X_0, X_1, \ldots, X_t, \ldots$ visited by a particle which moves as follows: If $X_t = v$ then X_{t+1} is chosen uniformly at random from the out-neighbors of X_t. If any of the random walks ever reaches t, then the digraph contains an (s,t)-path, and we continue to the next pair of vertices.

---

procedure StrongConnHeuristic(D)
  for each ordered pair of vertices (s,t) **do**
    for $i = 1, \ldots, N_1$ **do**
      Starting from s, take N_2 steps in a random walk on the digraph D.
      /** *A "random walk" means the sequence of vertices* $X_0, X_1, \ldots, X_{N_2}$ *visited* **/
      /** *by a particle which moves as follows: If* $X_t = v$ *then* X_{t+1} *is chosen* **/
      /** *uniformly at random from the out-neighbors of* X_t. **/
    if none of the random walks reaches t **then**
      return not strongly connected.
  return strongly connected.

Figure 7.1: Algorithm $\mathcal{A}$, a heuristic for recognizing strong connectivity

If N_1 and N_2 are large enough, then algorithm in Figure 7.1 is correct **whp**. For example, if there is a path from s to t, then if the random walk has followed it correctly so far, it has probability more than $1/n$ of following it correctly for one more step. Since the distance from s to t is less than n, taking $N_2 = n$ we have that the success probability for a single walk exceeds n^{-n}. So taking $N_1 = n^{2n}$ we will discover the path **whp**. We have just given

a superexponential time algorithm for a problem in **NL** but the values of N_1 and N_2 can be significantly improved for smoothed random instances.

The main theorem of this section is that when N_1 and N_2 are suitable polynomials in n, this heuristic, which we will call Algorithm $\mathcal{A}$, is successful on perturbations of bounded out-degree instances **whp**. To prove this, we first show it is successful when the initial instance is a strongly connected digraph and the perturbation only adds arcs. Then we extend this to show success when the initial instance is not necessarily strongly connected and the perturbation only adds arcs. After this, it is simple to translate our results to the original perturbation model where arcs are added and removed, since we can generate the perturbation in 2 rounds, by first deleting each existing arc with some probability, and then adding random arcs to the resulting digraph.

Recall that $\mathbb{D}_{n,\epsilon/n}$ is the distribution of digraphs on vertex set $[n]$ in which each possible arc appears independently with probability ϵ/n, and we write $R \sim \mathbb{D}_{n,\epsilon/n}$ to mean R is selected randomly according to distribution $\mathbb{D}_{n,\epsilon/n}$. We write $G_1 \oplus G_2$ to mean the XOR of digraphs G_1 and G_2, (which is to say $e \in G_1 \oplus G_2$ if and only if $e \in G_1$ and $e \notin G_2$ or vice versa.)

Theorem 4 *Let ϵ and Δ be positive constants with ϵ sufficiently small. For any n-node digraph $\bar{D}$ with maximum in-degree and out-degree Δ, let $D = \bar{D} \oplus R$ where $R \sim \mathbb{D}_{n,\epsilon/n}$. Then there exist absolute constants A_1, B_1 such that* **whp** *Algorithm $\mathcal{A}$ is correct on D when $N_1 = n^{A_1 \epsilon^{-1} \ln(10\Delta)}$ and $N_2 = B_1 \epsilon^{-1} \ln n$.*

We find that $A_1 = 400$ and $A_2 = 200$ suffice in this theorem, but we do not attempt to optimize these values.

If a strongly connected digraph has bounded out-degree and has diameter $\mathcal{O}(\ln n)$ then a random walk of length $\mathcal{O}(\ln n)$ has a $1/\operatorname{poly}(n)$ chance of going from s to t, and Algorithm $\mathcal{A}$ will succeed **whp** using values of N_1 and N_2 that can be realized in log-space. Unfortunately, even though our initial instances have bounded out-degree and (as indicated by Theorem 3) our perturbed instances have logarithmic diameter, the perturbation increases the maximum degree to $\Omega(\ln n / \ln \ln n)$, so we must work a little harder to show that the random walk has a non-negligible probability of witnessing the path from s to t. (As an additional reward for this work, we find that Algorithm $\mathcal{A}$ can be derandomized by checking all paths from s of length $\mathcal{O}(\epsilon^{-1} \ln n)$ and still only use log-space.)

The analysis of Algorithm $\mathcal{A}$ is further complicated by the possibility of an instance $\bar{D}$ which is not strongly connected combining with R to produce a smoothed instance which is strongly connected. We handle this situation by a three-step argument. First, for the smoothed instance to become strongly connected there cannot be too many small strongly connected components of $\bar{D}$. Then, the large components merge to form a strongly connected component with low diameter. Finally, the small components, if they are connected to the large component, are "close" to it.

Why study instances with bounded degree?

It would be nice to extend our results to hold for perturbed copies of any digraph, instead of only digraphs with bounded degree. However, such a result is not possible for our heuristic.

We show that our assumption that $\bar{D}$ has bounded degree cannot be weakened too much by constructing a family of instances with maximum out-degree and maximum in-degree growing with n for which Algorithm $\mathcal{A}$ does not succeed **whp**.

Theorem 5 *Let ϵ be a sufficiently small positive constant, and let $R \sim \mathbb{D}_{n,\epsilon/n}$. Then for every sufficiently large n, there exists an n-node digraph $\bar{D}$ with maximum out-degree $\mathcal{O}(\ln n)$ and maximum in-degree $\mathcal{O}(n^{1/3}(\ln n)^2)$ such that for $\bar{D} \oplus R$ the probability that Algorithm $\mathcal{A}$ fails exceeds $1 - e^{-\epsilon} - o(1)$.*

Strong connectivity versus (s,t)-connectivity

Although strong connectivity is an **NL**-complete problem, (s,t)-connectivity is "the" **NL**-complete problem. In Sipser's undergraduate text [212], while the completeness of (s,t)-connectivity is proved in detail, the completeness of strong connectivity is left as an exercise (Section 7.7 includes a simple solution to this exercise which shows completeness still holds for strong connectivity in graphs with bounded out-degree.)

In light of this, it is natural to investigate the success of heuristics on smoothed instances of (s,t)-connectivity. Here we find that there are instances on which Algorithm $\mathcal{A}$ fails **whp**. What is more, no log-space heuristic exists, provided a conjecture of complexity theory holds.

Theorem 6 *If* **NL** $\not\subseteq$ *almost-***L** *then no log-space heuristic succeeds* **whp** *on smoothed instances of bounded out-degree (s,t)-connectivity.*

The proof consists of building a machine which simulates any nondeterministic log-space machine using the log-space heuristic for (s,t)-connectivity, were such a heuristic to exist. Before the proof, we will also recall the definition of almost-**L** and comment on why it appears instead of **BPL**.

Smoothed model versus semi-random model

The semi-random model was introduced by Santha and Vazirani in [206]. In this model an adversary adaptively chooses a sequence of bits and each is corrupted independently with probability δ. They present this as a model for real-world random bits, such as the output of a Geiger counter or noisy diode, and consider the possibility of using such random bits in computation on worst-case instances. Blum and Spencer considered the performance of a graph coloring heuristic on random and semi-random instances in [39]. Subsequent work has uncovered an interesting difference between the random and semi-random instances in graph coloring. The work of Alon and Kahale [15] developed a heuristic which succeeds **whp** on random instances with constant expected degree, while work by Feige and Kilian [112] showed no heuristic can succeed on semi-random instances with expected degree $(1-\epsilon)\ln n$ (they also developed a heuristic for semi-random instances with expected degree $(1+\epsilon)\ln n$).

In the original semi-random model of Santha and Vazirani, an instance is formed by an adaptive adversary, who looks at all the bits generated so far, asks for a particular value for the next bit, and gets the opposite of what was asked for with probability δ.

Several modifications are proposed in Blum and Spencer [39] and also in Subramanian, Fürer, and Veni Madhavan [218] and Feige and Krauthgamer [113]. However, all these variations maintain the adaptive aspect of the adversary's strategy, which at low density allows too much power; if the error probability $p = (1-\epsilon)\ln n/n$ then there will be roughly n^ϵ isolated vertices in $\mathbb{D}_{n,p}$ and the adversary will be able to encode a polynomial sized instance containing no randomness. Since we wish to consider extremely sparse perturbations, where the error probability $p = \epsilon/n$, we cannot allow an adversary as powerful as in the semi-random model. The XOR perturbation considered in this chapter is equivalent to a natural weakening of the semi-random model: making the adversary oblivious.

7.1.2 Application: Property Testing

Property testing provides an alternative weakening of worst-case analysis of decision problems. It was formalized by Goldreich, Goldwasser, and Ron in [145]. The goal in property testing is to design an algorithm which decides whether an instance has a property or differs significantly from all instances which have that property (usually without looking at more than a vanishing fraction of the input bits). For example, a property tester for strong connectivity in bounded degree digraphs should accept all strongly connected instances and reject all instances that are ϵn arcs away from being strongly connected. Note that Algorithm $\mathcal{A}$ (which is designed to work on smoothed random instances) can be converted into a property tester: given an instance $\bar{D}$ and a gap parameter ϵ, we can randomly perturb $\bar{D}$ ourselves by adding $\frac{\epsilon}{2}n$ random arcs and then run Algorithm $\mathcal{A}$ on the perturbed version. This does not yield anything impressive for testing strong connectivity, since the undirected connectivity testing results of Goldreich and Ron in [146] can be applied to the directed case to produce a constant time tester. However, our perturbation approach also yields a property tester for a more difficult connectivity problem, that of being k-linked.

A digraph is said to be k-linked if for every choice of $2k$ distinct vertices $s_1, \ldots, s_k$, $t_1, \ldots, t_k$, the graph contains k vertex disjoint paths joining s_1 to t_1, $\ldots$, s_k to t_k. Recognizing whether or not a digraph is k-linked is **NP**-complete for $k \geq 2$. In the bounded degree property testing version of being k-linked, we are given a constant ϵ and a digraph $\bar{D}$ with maximum in-degree and out-degree Δ and our goal is to accept if $\bar{D}$ is k-linked and reject if $\bar{D}$ is more than ϵn arcs away from being k-linked, and we can do anything if $\bar{D}$ is not k-linked, but is close to being so.

A heuristic for testing k-linkedness

For this section k is assumed to be fixed, independent of the input. Figure 7.2 is a simple heuristic for testing k-linkedness. In words, the heuristic goes is this. Given $\bar{D}$ and ϵ, we perturb $\bar{D}$ by generating a graph $R \sim \Gamma_{n,\epsilon/2n}$ ourselves and adding that to D. Let $D = \bar{D} + R$ denote this perturbed instance. Then, for each choice of $2k$ distinct vertices, repeat the following procedure N_1 times: For $i = 1, \ldots, k$, starting at s_i take N_2 steps in a random walk on the graph. If all k random walks ever reach the correct k terminals via vertex disjoint paths, we continue to the next choice of $2k$ vertices. Otherwise reject.

procedure k-LinkednessHeuristic($\bar{D}, \epsilon$)
  Generate $R \sim \mathbb{D}_{n,\epsilon/2n}$
  $D := \bar{D} + R$
  for each set of $2k$ distinct vertices **do**
    for $i = 1, \ldots, N_1$ **do**
      for $j = 1, \ldots, k$ **do**
        Starting from s_i, take N_2 steps in a random walk on the digraph D.
      if all k random walks reach the correct k terminals via vertex disjoint paths **then**
        continue to the next choice of $2k$ vertices
      else
        if $i = N_1$ **then**
          return not k-linked
  return k-linked

Figure 7.2: A heuristic for testing k-linkedness

Theorem 7 *Let k be a positive integer, and let ϵ and Δ be positive constants with ϵ sufficiently small. For any k-linked n-node graph $\bar{D}$ with maximum degree Δ, the algorithm above accepts in polynomial time* **whp**.

A few comments regarding the difference between this theorem and Theorems 3 and 4: The fact that k vertex disjoint paths exist **whp** follows from a calculation analogous to the proof of Theorem 3, but now we must explore disjoint neighborhoods around all $2k$ terminals simultaneously. Also, the analog of the most difficult part of Theorem 4, showing that Algorithm $\mathcal{A}$ is correct in the case where a disconnected $\bar{D}$ leads to a strongly connected D, is no longer necessary. In the property testing setting, we are not required to correctly recognize instances that lead to this situation. It seems as though it might be possible to carry out this most difficult part and obtain a heuristic for testing k-linkedness that works on smoothed instances, but the details remain elusive.

Often in property testing, the goal is to minimize the sample complexity (meaning the number of times the algorithm accesses a bit of the input), and here we diverge from the norm, since the algorithm above likely looks at every arc of the graph. So it may be more accurate to call this an "algorithm for a promise problem version of k-linkedness". Also, we will not make use of the full power of ϵ-far-ness, and could succeed even on instances for which adding ϵn random arcs has probability less than $1/2$ of linking the unlinked.

7.1.3 Outline of what follows

We first prove Theorem 3 in Section 7.2. In Section 7.3 we prove Theorem 4, showing that Algorithm $\mathcal{A}$ is successful **whp**. Section 7.4 is devoted to the proof of Theorem 5, by constructing an instance with growing out-degree where Algorithm $\mathcal{A}$ fails with constant probability. In Section 7.5, we prove Theorem 6 by showing how to use a log-space heuristic for (s,t)-connectivity to build an almost-**L** simulator for any **NL** machine. Finally, in

Section 7.6 we will prove Theorem 7, which is a reprise of the proof of Theorem 3. Section 7.8 is a brief conclusion.

7.1.4 Some facts and notation

We will use the following Chernoff bounds from [156, Theorem 2.1] on the Binomial random variable $\mathrm{Bi}(n,p)$

$$\mathbb{P}\left[\mathrm{Bi}(n,p) \geq np+t\right] \leq \exp\left\{-\frac{t^2}{2(np+t/3)}\right\} \tag{7.1}$$

$$\mathbb{P}\left[\mathrm{Bi}(n,p) \leq np-t\right] \leq \exp\left\{-\frac{t^2}{2np}\right\}. \tag{7.2}$$

$\mathbb{D}_{n,\epsilon/n}$ is the distribution of digraphs on vertex set $[n]$ in which each possible arc appears independently with probability ϵ/n, and we write $R \sim \mathbb{D}_{n,\epsilon/n}$ to mean R is selected randomly according to distribution $\mathbb{D}_{n,\epsilon/n}$.

We write $G_1 \oplus G_2$ to denote the digraph formed by XOR-ing digraphs G_1 and G_2, (which is to say $e \in G_1 \oplus G_2$ if and only if $e \in G_1$ and $e \notin G_2$ or vice versa.)

We write $G_1 + G_2$ to denote the digraph formed by taking the union of the arcs of G_1 and G_2.

7.2 Proof that diameter of D is $\mathcal{O}(\epsilon^{-1}\ln n)$ whp

We now show that if $\bar{D}$ is strongly connected and has in-degree and out-degree bounded by $\Delta = \mathcal{O}(\ln n)$ then for $R \sim \mathbb{D}_{n,\epsilon/n}$ the diameter of $D = \bar{D} + R$ is $\mathcal{O}(\epsilon^{-1}\ln n)$ **whp**.

We will show that **whp** D contains short paths of a special form, alternating between some arcs from $\bar{D}$ and random arcs from R. This is similar to the approach of Bollobás and Chung [49].

To proceed, we now fix 2 vertices s and t and look for a short path between them. Let P be a shortest path from s to t in $\bar{D}$. If P has length less than $100\epsilon^{-1}\ln n$ then this vertex pair is already close, so suppose P has length at least $100\epsilon^{-1}\ln n$.

Let $S_0 = T_0 = \emptyset$, and let S_0' be the first $32\epsilon^{-1}\ln n$ nodes of P and let T_0' be last $32\epsilon^{-1}\ln n$ nodes of P.

We call a node *useful* if it is not within distance $d = 5\epsilon^{-1}$ of any node which we have previously placed in an S or T set, where distance is the length of the shortest path in the undirected graph underlying $\bar{D}$.

To build S_i, we check for each node $s' \in S_{i-1}$ if R contains an arc from s' to some useful node s''. If it does, we include s'' in S_i and also all nodes reachable from s'' by taking d steps in $\bar{D}$.

T_j is defined analogously, but the paths lead towards t instead of away from s. So, for each $t' \in T_{i-1}$ if R contains an arc from some useful node t'' to t', we include t'' in T_j and also all nodes from which t'' is reachable by taking d steps in $\bar{D}$. To make this definition completely precise, we include a description of the procedure `GenerateSets` which produces S_i and T_j in Figure 7.3. We use U to denote the set of useful nodes. Also, the notation

$N_d^+(S)$ denotes the set of nodes reachable in $\bar{D}$ in at most d steps starting from some node of S, the notation $N_d^-(S)$ denotes the set of nodes from which some node of S is reachable in at most d steps in $\bar{D}$, and $N_d(S) = N_d^+(S) \cup N_d^-(S)$. Finally, let $\ell = \lceil \log_2 n \rceil$.

---

procedure `GenerateSets` [(s,t)-path P]

$S_0 :=$ first $32\epsilon^{-1} \ln n$ nodes of P.
$T_0 :=$ last $32\epsilon^{-1} \ln n$ nodes of P.
$U := V \setminus N_d(S_0 \cup T_0)$

$i := 0$
$j := 0$

while ($|S_i| \le n^{2/3}$ and $i \le \ell$) or ($|T_j| \le n^{2/3}$ and $j \le \ell$) **do**
  if $|S_i| \le n^{2/3}$ and $i \le \ell$ **then**
    $S_{i+1} := \emptyset$
    for all $s' \in S_i$ **do**
      if $|S_{i+1}| \le n^{2/3}$ and there exists $s'' \in U$ such that $(s', s'') \in R$ **then**
        $S_{i+1} := S_{i+1} \cup N_d^+(\{s''\})$
        $U := U \setminus N_d(N_d^+(\{s''\}))$
    $i := i + 1$

  if $|T_j| \le n^{2/3}$ and $j \le \ell$ **then**
    $T_{j+1} := \emptyset$
    for all $t' \in T_j$ **do**
      if $|T_{j+1}| \le n^{2/3}$ and there exists $t'' \in U$ such that $(t'', t') \in R$ **then**
        $T_{j+1} := T_{j+1} \cup N_d^-(\{t''\})$
        $U := U \setminus N_d(N_d^-(\{t''\}))$
    $j := j + 1$

Figure 7.3: Procedure to generate S_i and T_j

This procedure is convenient for analysis because no arc of R is examined more than once, due to the way the useful set U is maintained. Therefore, we can employ the *principle of deferred decisions* find a simple expression for the conditional probability that, for example, $(s', s'') \in R$ at any step of `GenerateSets`.

We will now show that when `GenerateSets` halts

$$\mathbb{P}[|S_i| \le n^{2/3} \text{ or } |T_j| \le n^{2/3}] = o(n^{-2}). \tag{7.3}$$

To see this, we first note that at any step of `GenerateSets`, $|U| \ge n - 2\Delta^{2d}(\ell n^{2/3} + 32\epsilon^{-1} \ln n) = (1 - o(1))n$. This is because at most $\Delta^{2d} \le n^{1/10}$ nodes are removed from U in any step where U is changed, and it is changed at most $n^{2/3}$ times in each inner loop, and the inner loops are executed at most ℓ times each. And, by similar considerations, the initialization of U has size at least $n - 2\Delta^{2d}(32\epsilon^{-1} \ln n)$.

Now we consider the event $\mathcal{E}_{s'}$ given by "$s' \in S_{i'}$ and there exists $s'' \in U$ with $(s', s'') \in R$." Since each arc appears in R independently with probability ϵ'/n, we can apply the

principle of deferred decisions. We condition on the entire history of `GenerateSets`, which can be described by $H = \langle U, S_1, T_1, \ldots, S_{i'}, T_{j'}, \tilde{S}_{i'+1}\rangle$, where $\tilde{S}_{i'+1}$ denotes the intermediate state of the set $S_{i'+1}$, and for $s' \in S_{i'}$, we have that the probability of $\mathcal{E}_{s'}$ depends only on the size of U, which is always $(1-o(1))n$. So

$$\mathbb{P}[\mathcal{E}_{s'} \mid H] = 1 - (1-p)^{|U|} = (1-o(1))\epsilon.$$

Every time $\mathcal{E}_{s'}$ occurs, at least d vertices are added to $S_{i'+1}$, so conditioned on $|S_{i'}|$, the random variable $|S_{i'+1}|/d$ stochastically dominates $Z_{i'+1} \sim \mathrm{Bi}(|S_{i'}|, (1-o(1))\epsilon)$. Thus, letting $\mathcal{B}_{i'+1}$ denote the event "$|S_{i'+1}| \le 2|S_{i'}|$" we have

$$\mathbb{P}\left[\mathcal{B}_{i'+1} \;\middle|\; S_{i'}\right] \le \mathbb{P}\left[Z_{i'+1} \le \mathbb{E}[Z_{i'+1}] - \frac{3}{5}\epsilon|S_{i'}| \;\middle|\; S_{i'}\right] \le e^{-\frac{9}{50}\epsilon|S_{i'}|},$$

where the final inequality is an application of the Chernoff bound (7.2).

Note that in order for `GenerateSets` to halt with $|S_{i'}| \le n^{2/3}$ it must be that some $\mathcal{B}_{i'}$ occurs for $i' \le i$. Since $|S_0| = 32\epsilon^{-1}\ln n$, we have that

$$\mathbb{P}[|S_i| \le n^{2/3}] \le \mathbb{P}\left[\bigcup_{i'=1}^{i} \mathcal{B}_{i'}\right] \le \sum_{i'=1}^{i} \mathbb{P}\left[\mathcal{B}_{i'} \;\middle|\; |S_{i'-1}| \ge 32\epsilon^{-1}\ln n\right] \le \ell \cdot e^{-5\ln n} = o(n^{-2}).$$

A similar argument shows that when `GenerateSets` halts we also have $\mathbb{P}[|T_j| \le n^{2/3}] = o(n^{-2})$.

Now, to finish the short path from s to t, we generate the random arcs of R between S_i and T_j

$$\mathbb{P}\left[R \cap S_i \times T_j = \emptyset \;\middle|\; |S_i| \ge n^{2/3} \wedge |T_j| \ge n^{2/3}\right] \le (1-p)^{n^{4/3}} \le e^{-\epsilon n^{1/3}} = o(n^{-2}).$$

Putting all the pieces together, we have an (s,t)-path consisting of a path of length at most $32\epsilon^{-1}\ln n$, followed by at most 2ℓ paths of length $d+1$ from $\bar{D}$ joined by edges from R, and finishing with a path of length at most $32\epsilon^{-1}\ln n$, for total length which numerical calculation shows is less than $100\epsilon^{-1}\ln n$.

Since there are only $n(n-1)$ choices for (s,t), the theorem follows by the union bound. □

7.3 Proof that log-space algorithm recognizes strong connectivity whp

7.3.1 When $\bar{D}$ is strongly connected

By Theorem 3 we know that the diameter of D is $\mathcal{O}(\ln n)$ **whp**. Unfortunately we cannot yet conclude that Algorithm $\mathcal{A}$ is successful **whp**. We must still argue that the probability of a random walk traversing the short path is not too small. In the a graph with out-degree less than some constant, having a diameter of $\mathcal{O}(\ln n)$ would imply an efficient algorithm.

Our random perturbation has likely created some vertices with out-degree $\Omega(\ln n/\ln\ln n)$, so we will have to work a little more. We use the notation $\deg^+_D(v)$ to denote the out-degree of a vertex v in digraph D.

Lemma 7 *Let $D = \bar{D} + R$, where $\bar{D}$ is an arbitrary digraph with maximum out-degree Δ and $R \sim \mathbb{D}_{n,\epsilon/n}$. Then* **whp** *$D$ contains no path P of length $\ell \le \ell_1 = 100\epsilon^{-1}\ln n$ with $\prod_{x\in P}\deg^+_D(x) \ge n^{100\epsilon^{-1}\ln(10\Delta)}$.*

Proof We prove the claim by the first moment method. First, we bound the number of paths with ℓ vertices that use $\ell - a$ arcs of $\bar{D}$. There are n places to start such a path, and there are $\binom{\ell}{a}$ different ways to decide when to take an arc not in $\bar{D}$. For each arc in $\bar{D}$, since the out-degree is bounded, there are at most Δ choices, and there are at most n^a choices for where the non-$\bar{D}$ arcs can go. So there are at most

$$n\Delta^{\ell-a}\binom{\ell}{a}n^a$$

potential paths of length ℓ that use a arcs from R. The probability such a potential path appears as a path in D is $\left(\frac{\epsilon}{n}\right)^a$.

Now we bound the probability that the sum of the logarithms of the out-degrees of the vertices along a path P of length ℓ exceeds $\ell\ln(\Delta+1)+t$. To do so, we first bound a similar quantity for the graph $R' = R \setminus P$. Note that

$$\mathbb{P}\left[\sum_{v\in P}\ln\{1+\deg^+_{R'}(v)\} \ge t\right] \le e^{-t}\mathbb{E}\left[\prod_{v\in P}(1+\deg^+_{R'}(v))\right]$$
$$= e^{-t}\prod_{v\in P}\mathbb{E}[1+\deg^+_{R'}(v)] \le (1+\epsilon)^{|P|}e^{-t}.$$

Then, since P is a path, it contains at most one arc incident with v, so

$$\ln\{\deg^+_D(v)\} \le \ln\{\deg^+_{\bar{D}}(v)+\deg^+_R(v)\} \le \ln\{\Delta+1+\deg^+_{R'}(v)\} \le \ln\{1+\Delta\}+\ln\{1+\deg^+_{R'}(v)\},$$

and we have

$$\mathbb{P}\left[\sum_{v\in P}\ln\{\deg^+_D(v)\} \ge \ell\ln(1+\Delta)+t\right] \le (1+\epsilon)^\ell e^{-t}.$$

So the expected number of paths of length ℓ with a arcs from R and product of degrees exceeding $\ell\ln(\Delta+1)+t$ is at most

$$n\Delta^{\ell-a}\binom{\ell}{a}n^a\left(\frac{\epsilon}{n}\right)^a(1+\epsilon)^\ell e^{-t} \le n((1+\epsilon)\Delta)^\ell e^{-t}\binom{\ell}{a}.$$

Let $\ell_1 = 100\epsilon^{-1}\ln n$ and let

$$t = 2\ln n + \ln\ell_1 + \ell_1\ln(2(1+\epsilon)\Delta) \le 100\epsilon^{-1}\ln(10\Delta)\ln n.$$

Then an upper bound on the probability that D contains such a path of length at most ℓ_1 is

$$\sum_{\ell=1}^{\ell_1}\sum_{a=0}^{\ell} ne^{-t}((1+\epsilon)\Delta)^{\ell}\binom{\ell}{a} = \sum_{\ell=1}^{\ell_1} ne^{-t}((1+\epsilon)\Delta)^{\ell}2^{\ell}$$
$$= ne^{-t}\sum_{\ell=1}^{\ell_1}(2(1+\epsilon)\Delta)^{\ell} \leq ne^{-t}\ell_1(2(1+\epsilon)\Delta)^{\ell_1}\} = o(1).$$

So **whp** there is no path P of length at most $100\epsilon^{-1}\ln n$ which has

$$\prod_{x\in P}\deg_D^+(x) \geq n^{100\epsilon^{-1}\ln(10\Delta)}.$$

□

The correctness of Algorithm $\mathcal{A}$ in the case when $\bar{D}$ is strongly connected now follows from that fact that the probability a random walk follows a path P from s to t is precisely $\left(\prod_{x\in P}\deg_D^+(x)\right)^{-1}$. □

7.3.2 When $\bar{D}$ not strongly connected

The previous section shows that Algorithm $\mathcal{A}$ is correct **whp** for strongly connected digraphs $\bar{D}$. To prove that Algorithm $\mathcal{A}$ is correct **whp** when $\bar{D}$ is not strongly connected, we must do some more work.

Outline of approach

Consider the strong components of $\bar{D}$. If there are many components of size less than $\frac{1}{4}\epsilon^{-1}\ln n$, then we show that **whp** one of them will be incident to no arcs of R and so D will not be strongly connected and Algorithm $\mathcal{A}$ will be correct. In the case where $\bar{D}$ consists mostly of larger strong components, we expose the random arcs R in two rounds. We argue that **whp** the strong components of $\bar{D}$ merge into a unique giant strong component S_g containing at least $n - n^{16/17}$ vertices after the first round. Then we invoke Lemma 3 from the previous section to show that the random arcs from the second round give the giant component a low diameter. Then we deal with the vertices that belong to small strong components after the first round of random arcs have been added. These vertices might be connected to S_g in both directions and they might not, and there is not necessarily a sharp threshold for strong connectivity. However, we show that for some constant A_1 **whp** no vertex which is not in S_g is connected in either direction to S_g only by paths of length more than $A_1\epsilon^{-1}\ln n$ i.e. such a vertex is close to the giant component in some direction, or cannot be reached at all in this direction. Finally, by Lemma 7 we know that all the paths of length at most $A_1\epsilon^{-1}\ln n$ have a non-negligible probability of being traversed by Algorithm $\mathcal{A}$ (take the bound in Lemma 7 and raise it to the power $A_1/100$). So we conclude that **whp** the graph is not strongly connected, in which case Algorithm $\mathcal{A}$ is correct, or the graph is strongly connected in such a way that Algorithm $\mathcal{A}$ is still correct.

The calculations required for this plan follow.

Lemma 8 *If $\bar{D}$ has more than $n^{1/2}\ln n$ strong components containing less than $\frac{1}{4}\epsilon^{-1}\ln n$ vertices, then* **whp** *one of these components is not incident to any arcs of R.*

Proof We use the second moment method (see, for example, [156, Page 54]). For each small strong component C of $\bar{D}$, let X_C be an indicator random variable for the event that C is not incident to any arc of R. Then $\mathbb{E}[X_C] = (1-\epsilon/n)^{2c(n-c)} \geq n^{-1/2}(1-o(1))$, where $c = |C| \leq \frac{1}{4}\epsilon^{-1}\ln n$, and

$$\begin{aligned}\mathbb{E}[X_{C_1}X_{C_2}] &= (1-\epsilon/n)^{2c_1(n-c_1-c_2)+2c_2(n-c_1-c_2)+2c_1c_2}\\ &= (1-\epsilon/n)^{2c_1(n-c_1)}(1-\epsilon/n)^{2c_2(n-c_2)}(1+\mathcal{O}((\ln n)^2/n))\\ &= \mathbb{E}[X_{C_1}]\mathbb{E}[X_{C_2}](1+o(1)),\end{aligned}$$

where $c_1 = |C_1|$ and $c_2 = |C_2|$. Let $Z = \sum_C X_C$ be the number of strong components that are incident to no arc of R. Then, since there are at least $n^{1/2}\ln n$ terms in this sum, $\mathbb{E}[Z] \geq \frac{1}{2}\ln n$. We also have

$$\begin{aligned}\frac{\mathbb{E}[Z^2]}{\mathbb{E}[Z]^2} &= \frac{\sum_C \mathbb{E}[X_C^2] + \sum_{C_1\neq C_2}\mathbb{E}[X_{C_1}X_{C_2}]}{\left(\sum_C \mathbb{E}[X_C]\right)^2}\\ &= \frac{\sum_C \mathbb{E}[X_C]}{\left(\sum_C \mathbb{E}[X_C]\right)^2} + (1+o(1))\frac{\sum_{C_1\neq C_2}\mathbb{E}[X_{C_1}]\mathbb{E}[X_{C_2}]}{\left(\sum_C \mathbb{E}[X_C]\right)^2} \leq \frac{2}{\ln n} + 1 + o(1).\end{aligned}$$

Now, by the second moment method, (see, for example, [156, Inequality 3.3, Page 54]) we have $\mathbb{P}[Z \neq 0] \geq \mathbb{E}[Z]^2/\mathbb{E}[Z^2] = 1 - o(1)$. □

It follows from this lemma that Algorithm $\mathcal{A}$ works in the case where the number of small strong components of $\bar{D}$ is large.

We now consider the case where $\bar{D}$ has at most $n^{1/2}\ln n$ strong components of size less than $\sigma = \frac{1}{4}\epsilon^{-1}\ln n$. As in the previous section, we consider adding the random arcs of R in two rounds, by introducing R' and R''. We take $R' \sim \mathbb{D}_{n,p'}$ with $p' = \frac{\epsilon}{2n}$ and $R'' \sim \mathbb{D}_{n,p''}$ with $p'' = \frac{\epsilon}{2n-\epsilon} = (1+o(1))\frac{\epsilon}{2n}$. Then the probability that an arc appears in $R' + R''$ is exactly $\frac{\epsilon}{n}$, and $R' + R''$ is identically distributed with R.

Let the strong components in $\bar{D}$ with size exceeding σ be $C_1, C_2, \ldots, C_a$ and let their sizes be $n_1, n_2, \ldots, n_a$. For $K \subseteq [a]$ let $C_K = \bigcup_{i\in K} C_i$, let $n_K = \sum_{i\in K} n_i$, and let A_K^+ denote the number of arcs of R' that go from C_K to $\overline{C_K}$ and let A_K^- denote the number of arcs of R' that go from $\overline{C_K}$ to C_K. Then

$$\mathbb{P}(A_K^+ = 0 \text{ or } A_K^- = 0) \leq 2\left(1-\frac{\epsilon}{2n}\right)^{n_K(n-n^{1/2}\log n - n_K)}.$$

To obtain an upper bound on the number of choices for K with a given value of n_K, first note that since each large strong component is of size at least σ we have that a, the number

of large strong components is at most n/σ. Also as a consequence of the strong components being large, for a given value of n_K we have $|K| \le n_K/\sigma$. So the number of choices for K with a given value of n_K is at most $\sum_{\ell=1}^{n_K/\sigma}\binom{a}{\ell}$ and for $n_K \le n/2$ this is at most $n_K/\sigma\binom{n/\sigma}{n_K/\sigma}$. Thus

$$\begin{aligned}
&\mathbb{P}(\exists K \subseteq [a]:\ n^{16/17} \le n_K \le n/2 \text{ and } A_K^+ = 0 \text{ or } A_K^- = 0)\\
&\quad\le \sum_{n_K=n^{16/17}}^{n/2} n_K/\sigma\binom{n/\sigma}{n_K/\sigma}2\left(1-\frac{\epsilon}{2n}\right)^{n_K(n(1-o(1))-n_K)}\\
&\quad\le \sum_{n_K=n^{16/17}}^{n/2} 2n_K/\sigma\left((ne/n_K)^{1/\sigma}e^{-\epsilon/4(1-o(1))}\right)^{n_K}\\
&\quad\le \sum_{n_K=n^{16/17}}^{n/2} e^{-(1-o(1))\epsilon n_K/68} = o(1).
\end{aligned}$$

It follows that **whp**

$$\bar{D}+R' \text{ contains a } \textit{giant} \text{ strong component } S_g \text{ with } |S_g| \ge n-(1+o(1))n^{16/17}. \tag{7.4}$$

We apply the results of the previous section to S_g. We have a strongly connected digraph S_g and we add $R'' \sim \mathbb{D}_{n,p''}$ (recall that $p'' = \epsilon/(2n-\epsilon)$), producing a digraph D_g with diameter at most $201\epsilon^{-1}\ln n$ for which all shortest paths P satisfy $\prod_{x\in P}\deg^+_{D_g}(x) \le n^{201\epsilon^{-1}\ln(10\Delta)}$.

The only detail remaining is how to deal with the at most $n^{16/17}+\sigma n^{1/2}\ln n$ vertices of $\bar{D}+R'$ that are not in S_g. Let x be such a vertex. We will show that **whp** if there is a path from x to any vertex in S_g then it is a short path. An identical argument shows the same property holds for paths from S_g to x.

We consider two cases. Let V_x denote the set of vertices reachable by following $5\epsilon^{-1}\ln n$ arcs of $\bar{D}+R'$, starting from x. If $|V_x| \ge 5\epsilon^{-1}\ln n$ we say x is *medium* and if $|V_x| < 5\epsilon^{-1}\ln n$ we say x is *small*. If x is a medium vertex, then the probability R'' does not add an arc from a vertex in V_x to the S_g is at most $(1-p'')^{5\epsilon^{-1}\ln n(n-n^{16/17})} \le n^{-2}$. Thus **whp** all medium x are close to S_g in D. Let S_m denote the set of medium vertices.

Consider now a shortest path in D from a small vertex x to a vertex z in $S_g \cup S_m$. Removing all of the arcs of R'' from this path decomposes it into subpaths $P_1, P_2, \dots, P_r$. Let x_i and y_i denote the starting and ending vertices of subpath P_i (it is possible that $x_i = y_i$ if a subpath has length 0). We will show that **whp** r is small by considering the probability that the subpaths has a sequence $x_1, y_1, \dots, x_r, y_r$ with $r \ge 18$.

For any $\bar{D}+R'$ for which (7.4) holds, the number of choices for our sequence is at most $(n^{16/17}\times 5\epsilon^{-1}\ln n)^r n$ and the probability that the R''-arcs exist is $(p'')^r$. Thus the probability there exists a small vertex x which requires $r \ge 18$ is at most

$$\sum_{r\ge 18} n((1+o(1))n^{16/17}\times 5\epsilon^{-1}\ln n \times p'')^r = o(1).$$

So **whp**, if D is strongly connected then its diameter is at most $201\epsilon^{-1}\ln n+2\times 18(5\epsilon^{-1}\ln n+1)$ which, for n sufficiently large is at most $400\epsilon^{-1}\ln n$. (The worst case here comes from a path of length at most $18(5\epsilon^{-1}\ln n+1)$ from s to S_g, followed by a path of length at most $201\epsilon^{-1}\ln n$ to some other vertex of S_g and then by a path of length at most $18(5\epsilon^{-1}\ln n+1)$ from there to t.)

Applying Lemma 7 we see that **whp** these paths are traversed by Algorithm $\mathcal{A}$ with probability $n^{-400\epsilon^{-1}\ln(10\Delta)}$ and so Algorithm $\mathcal{A}$ is correct **whp**. □

7.4 An example with growing degrees

We now consider the possibility of applying Algorithm $\mathcal{A}$ to the case where $\bar{D}$ has maximum in-degree and out-degree Δ that grows with n. The following example shows that in this case Algorithm $\mathcal{A}$ does not necessarily succeed when using logarithmic space.

Let $d=n^{1/3}\ln n$. To form an instance, start with the directed cycle $C=(v_1,v_2\ldots,v_n)$. Then, for each v_i with $i\geq n/d$, we let $i_0=\lfloor i/d\rfloor$, and we add arcs (v_i,v_j) to $\bar{D}$, for each $j\in\{i_0,i_0+1,\ldots,i_0+\ln n-1\}$. We call these arcs the "backwards arcs". Note that each v_i with $i\geq n/d$ has $\ln n$ backwards arcs going out, and every v_j with $j<n/d$ has $d\ln n$ backwards arcs coming in. Let $V_1=\{v_1,\ldots,v_{n/d+\ln n}\}$, and note that all backwards arcs lead to V_1.

When we perturb this $\bar{D}$ to obtain D, the probability that we do not delete any arc of cycle C is $(1-\epsilon/n)^n=e^{-\epsilon}-o(1)$, and so D is strongly connected with probability at least $e^{-\epsilon}-o(1)$.

We now derive bounds showing 2 properties that hold **whp** and will allow us to reason about the probability of Algorithm $\mathcal{A}$ succeeding on D.

Lemma 9 *Let $L=(3\epsilon)^{-1}\ln n$ and V_1 be as above. Then the following holds* **whp**

(a) *For $\ell\leq L$, there is no path from V_1 to $\{v_{n-L},\ldots,v_n\}$ of length ℓ which does not use any of the backwards arcs.*

(b) *For $\ell>L$, there are at most $e^{2\epsilon\ell}$ paths of length ℓ from V_1 to v_n which do not use any of the backwards arcs.*

Proof Let P be a path from v_i to v_n of length at most ℓ. Removing all of the arcs of R from this path decomposes it into subpaths $P_1,\ldots,P_m$, where each P_i is a path of $\bar{D}$ (or possibly a degenerate path containing no arcs). Let the lengths of subpath P_i be denoted by $\ell_i\geq 0$ and let $\lambda=\ell_1+\ldots+\ell_m$. Since the length of P is at most ℓ, we have $m\leq\ell$, and for a given m, we have $\lambda\leq\ell-m$. Also, P_1 starts at v_i and P_m ends at v_n so it must start at $v_{n-\ell_m}$. There are less than n possibilities for where the subpaths P_k begin for $k=2,\ldots,m-1$. So, since there are $m-1$ arcs of R in P, each of which appears with

probability ϵ/n, the expected number of path from v_i to v_n with length at most ℓ is at most

$$\begin{aligned}\mathbb{E}[\#\text{ paths from } v_i \text{ to } v_n \text{ with length} \le \ell] &\le \sum_{m=0}^{\ell}\sum_{\lambda=0}^{\ell-m}\sum_{\ell_1+\cdots+\ell_m=\lambda} n^{m-2}\left(\frac{\epsilon}{n}\right)^{m-1}\\ &= n^{-1}\sum_{m=0}^{\ell}\epsilon^{m-1}\sum_{\lambda=0}^{\ell-m}\binom{\lambda+m-1}{m-1}\\ &= n^{-1}\sum_{m=0}^{\ell}\epsilon^{m-1}\binom{\ell}{m}\\ &\le (\epsilon n)^{-1}\sum_{m=0}^{\ell}\frac{(\epsilon\ell)^m}{m!}\\ &\le (\epsilon n)^{-1}e^{\epsilon\ell}.\end{aligned}$$

(a) Since there are $(1+o(1))n/d$ vertices v_i in V_1, the probability that D contains a path of length at most L from some vertex of V_1 to v_n is at most

$$(1+o(1))(n/d)(\epsilon n)^{-1}e^{\epsilon L} = (1+o(1))d^{-1}\epsilon^{-1}n^{1/3} = o(1).$$

(b) Since there are $(1+o(1))n/d$ vertices v_i in V_1, the expected number of paths of length at most ℓ from some vertex of V_1 to v_n is at most $(1+o(1))(d\epsilon)^{-1}e^{\epsilon\ell}$ and then for $\ell \ge L$, the Markov inequality and the union bound show that

$$\mathbb{P}(\exists \ell \ge L:\ \text{number paths } \ge e^{2\epsilon\ell}) \le (1+o(1))(d\epsilon)^{-1}\sum_{\ell\ge L} e^{-\epsilon\ell} = o(1).$$

□

Now consider a random walk W from v_1 to v_n. Let the parts of W between the use of backwards arcs of $\bar{D}$ be called *attempts* and let W' denote the last attempt of W (denoted *successful*), all other attempts will be termed *failures*. Note that W' must start in V_1.

At some point Algorithm $\mathcal{A}$ checks for (v_1, v_n)-connectivity by executing T steps of a random walk from v_1 and declaring connectivity if v_n is reached. Also, this is to be repeated N times. Now we must have $T = n^{\mathcal{O}(1)}, N = n^{\mathcal{O}(1)}$ in order that we can do the counting in log-space. The probability that any walk reaches v_n can be bounded by $T\sum_{\ell=L}^{n}(\ln n-1)^{-\ell}e^{2\epsilon\ell} \le (\ln n)^{-L/2}$. The factor T accounts for the $\le T$ points at which the successful attempt begins, $e^{2\epsilon\ell}$ bounds the number of possible paths making up this attempt and $(\ln n-2)^{-\ell}$ bounds the probability we follow this path. This is because every vertex after on the successful attempt has out-degree at least $\ln n-2$ **whp**. (Note that we may have deleted some of the backwards arcs when we XORed with R, but the probability that there exists some vertex for which 2 backwards arcs are deleted is at most $n\binom{\ln n}{2}\left(\frac{\epsilon}{n}\right)^2 = o(1)$.)

Thus the probability that we will declare v_1 connected to v_n is at most $N(\ln n)^{-L/2} = o(1)$ and the algorithm fails with probability at least $e^{-\epsilon} - o(1)$.

(This constructions works with any slowly growing function as the out-degree, instead

of $\ln n$. But to reduce the in-degree, it seems that some additional work is necessary.)

7.5 Proof of "impossibility" of log-space recognizer for (s,t)-connectivity

Suppose a heuristic exists which uses log-space and is successful **whp** on smooth instances of (s,t)-connectivity. Then, using a log-space transducer, we convert a worst-case instance of (s,t)-connectivity on n nodes into a smoothed instance of (s,t)-connectivity. Unfortunately, this reduction is not sufficient to show the existence of a heuristic implies $\mathbf{NL} \subseteq \mathbf{BPL}$, since a nondeterministic log-space simulator does not have the space to store the output of the log-space transducer. The traditional technique for simulating a log-space machine on the output of a log-space transducer is to, each time a bit of the input is requested, restart the transducer and simulate it until it produces the bit in question. This is an inefficient use of time, but in exchange for taking longer we use only logarithmically bounded space.

In our case, since the reduction is randomized, we somehow need the log-space transducer to produce the *same* random instance each time. This seems to require "multiple access randomness" (also known as the "wrong" definition of **BPL** and denoted $\mathbf{BP}^\star\mathbf{L}$). Nisan provides some evidence that multiple access randomness is more powerful than read-once randomness in [203]. He also shows that $\mathbf{BP}^\star\mathbf{L} = \text{almost-}\mathbf{L} \subseteq \mathbf{L/poly}$ (where almost-**L** is the set of languages L for which $\mu(L \in \mathbf{L}^A) = 1$, where μ is the standard measure for the set of oracles. For more details on almost classes, see the survey of Vollmer and Wagner [223].)

Given an instance D_0 of (s,t)-connectivity on n nodes, we construct an instance $\bar{D}$ on n^3 nodes by adding $n^3 - n$ isolated vertices. Call the original n vertices A and the additional vertices B. We smooth the instance by XORing it with $R \sim \mathbb{D}_{n^3,\epsilon/n^3}$ to form $D = \bar{D} \oplus R$. (This is where our log-space machine uses multiple access randomness— since there is not room to write out this whole graph, we must generate the i-th bit from scratch every time the heuristic asks for it. The values of D differ from $\bar{D}$ with some small probability and they should differ the same way every time the heuristic asks for the i-th bit.)

The probability R contains an arc between any pair of vertices of A is bounded by $n^2(\epsilon/n^3) = o(1)$. So if D_0 is (s,t)-connected, then D is (s,t)-connected **whp**. Now, since the vertices of B are isolated in $\bar{D}$, they form a sparse random graph in D, so **whp** no component has size exceeding $\mathcal{O}(\ln n)$ (see, for example, [156, Theorem 5.4]). Thus, the probability that D contains a component of B with arcs to and from vertices of A is less than $\mathcal{O}((n^3 \ln n)n^2(\epsilon/n^3)^2) = o(1)$. (Explanation: There are $n^3 - n < n^3$ choices for one endpoint in B, and $\mathcal{O}(\ln n)$ choices for the other, since it must be in the same component. There are n choices for each endpoint in A, and the probability that each random arc appears is ϵ/n^3.)

Since **whp** there are no arcs added to A and no components of B that serve as a shortcut between vertices of A, if D_0 is not (s,t)-connected, then D is not (s,t)-connected **whp**. This is sufficient to conclude that if a heuristic exists then $\mathbf{NL} \subseteq \text{almost-}\mathbf{L}$, since given an D_0 we could form D and use the heuristic to solve it **whp** which would give the correct answer to the original problem about D_0 **whp**.

7.6 Testing k-linkedness

We will show that if $\bar{D}$ is k-linked then **whp** D contains disjoint paths of length at most $100k\epsilon^{-1}\ln n$ which witness this.

Fix $s_1,\dots,s_k,t_1,\dots,t_k$, and let $P_1,P_2,\dots,P_k$ be vertex disjoint paths in $\bar{D}$ such that P_r goes from s_r to t_r.

We order the paths from longest to shortest and define $r^\star$ so that for $r\le r^\star$ each P_r has length at least $100k\epsilon^{-1}\ln n$. If $r^\star=0$ then there is nothing to prove, so suppose $r^\star\ge 1$.

We use the same type of argument as in the proof of Theorem 3 to show the existence of short paths between s_r and t_r, but we work with all $r\le r^\star$ simultaneously to ensure that the paths we find are vertex disjoint. To this end, we define a sequence of collections of subsets of vertices $S_{i,r}$ and $T_{i,r}$ for $i\ge 0$ and $1\le r\le r^\star$, (we also define $S_{i,r}=P_i$ for $i\ge 0$ and $r>\ell$). Let $S_{0,r}$ be the first $32k\epsilon^{-1}\ln n$ vertices of P_r and $T_{0,r}$ be the last $32k\epsilon^{-1}\ln n$ vertices of P_r, for $1\le r\le \ell$.

We will call a node *useful* if it is not within distance $d=5\epsilon^{-1}$ of any node which we have previously placed in any S or T set, where "distance" is the length of the shortest path in the undirected graph corresponding to $\bar{D}$.

To define $S_{i,r}$, we check, for each node s' in $S_{i-1,r}$, if R contains an arc from s' to some useful node s''. If it does, we add s'' and all nodes reachable from s'' by d steps in $\bar{D}$ to $S_{i,r}$. Note that if s'' is useful, this will add at least d nodes to $S_{i,r}$.

$T_{j,r}$ is defined analogously, but the paths lead towards t_r instead of away from s_r. For a node t' in $T_{j-1,r}$, we look for useful nodes t'' where an arc of R is directed from t'' to t', and add all nodes from which t' is reachable by d steps in $\bar{D}$.

To make this definition completely precise, we include a description of the procedure `GenerateSets2` for forming $S_{i,r}$ and $T_{j,r}$ in Figure 7.4. We use U to denote the set of useful nodes. Also, the notation $N_d^+(S)$ denotes the set of nodes reachable in $\bar{D}$ in at most d steps starting from some node of S, the notation $N_d^-(S)$ denotes the set of nodes from which some node of S is reachable in at most d steps in $\bar{D}$, and $N_d(S)=N_d^+(S)\cup N_d^-(S)$. Finally, let $\ell=\lceil\log_2 n\rceil$.

The proof is largely the same at Theorem 4, but must argue that when `GenerateSets2` halts, $\mathbb{P}[S_{i_r,r}\le n^{2/3}\vee T_{j_r,r}\le n^{2/3}]=o(n^{-2})$.

As with `GenerateSets`, the procedure `GenerateSets2` is convenient for analysis because no arc of R is examined more than once, due to the way the useful set U is maintained. Therefore, we can employ the *principle of deferred decisions* find a simple expression for the conditional probability that, for example, $(s',s'')\in R$ at any step of the procedure.

We first note that at any step of `GenerateSets2`, $|U|\ge n-2k\Delta^{2d}(\ell n^{2/3}+32\epsilon^{-1}\ln n)=(1-o(1))n$. This is because at most Δ^{2d} nodes are removed from U in any step where U is changed, and it is changed at most $n^{2/3}$ times in each inner loop, and the 2 inner loops are executed at most ℓk times each. And, by similar considerations, the initialization of U has size at least $n-2k\Delta^{2d}(32\epsilon^{-1}\ln n)$.

Now we consider the event $\mathcal{E}_{s'}$ given by "$s'\in S_{i',r}$ and there exists $s''\in U$ with $(s',s'')\in R$." Since each arc appears in R independently with probability ϵ'/n, we can apply the principle of deferred decisions. We condition on the entire history of the procedure, and

procedure `GenerateSets` 2(Disjoint paths $P_1, P_2, \ldots, P_k$)
  $U := V$

  for $r = 1, \ldots, r^\star$ **do**
    $S_{0,r} :=$ first $32\epsilon^{-1} \ln n$ nodes of P_r.
    $T_{0,r} :=$ last $32\epsilon^{-1} \ln n$ nodes of P_r.
    $U := U \setminus N_d(S_{0,r} \cup T_{0,r})$

    $i_r := 0$
    $j_r := 0$
  for $r = r^\star + 1, \ldots, k$ **do**
    $U := U \setminus \{\text{nodes of } P_r\}$

  while $\exists r\colon (|S_{i,r}| \le n^{2/3}$ and $i_r \le \ell)$ or $(|T_{j,r}| \le n^{2/3}$ and $j_r \le \ell)$ **do**
    if $|S_{i_r}| \le n^{2/3}$ and $i_r \le \ell$ **then**
      $S_{i+1,r} := \emptyset$
      for all $s' \in S_{i,r}$ **do**
        if $|S_{i+1,r}| \le n^{2/3}$ and there exists $s'' \in U$ such that $(s', s'') \in R$ **then**
          $S_{i+1,r} := S_{i+1,r} \cup N_d^+(\{s''\})$
          $U := U \setminus N_d(N_d^+(\{s''\}))$
      $i_r := i_r + 1$

    if $|T_{j,r}| \le n^{2/3}$ and $j_r \le \ell$ **then**
      $T_{j+1,r} := \emptyset$
      for all $t' \in T_{j,r}$ **do**
        if $|T_{j+1,r}| \le n^{2/3}$ and there exists $t'' \in U$ such that $(t'', t') \in R$ **then**
          $T_{j+1,r} := T_{j+1,r} \cup N_d^-(\{t''\})$
          $U := U \setminus N_d(N_d^-(\{t''\}))$
      $j_r := j_r + 1$

Figure 7.4: Procedure to generate $S_{i,r}$ and $T_{j,r}$ for $r = 1, \ldots, r^\star$

for $s' \in S_{i',r}$, we have that the probability of $\mathcal{E}_{s'}$ depends only on the size of U, which is always $(1 - o(1))n$. So

$$\mathbb{P}[\mathcal{E}_{s'} \mid H] = 1 - (1-p)^{|U|} = (1 - o(1))\epsilon.$$

Every time $\mathcal{E}_{s'}$ occurs, at least d vertices are added to $S_{i'+1,r}$, so conditioned on $|S_{i',r}|$, the random variable $|S_{i'+1,r}|/d$ stochastically dominates $Z_{i'+1} \sim \mathrm{Bi}(|S_{i',r}|, (1 - o(1))\epsilon)$. Thus, letting $\mathcal{B}_{i'+1}$ denote the event "$|S_{i'+1,r}| \le 2|S_{i',r}|$" we have

$$\mathbb{P}\left[\mathcal{B}_{i'+1} \;\middle|\; S_{i',r}\right] \le \mathbb{P}\left[Z_{i'+1} \le \mathbb{E}[Z_{i'+1}] - \frac{3}{5}\epsilon|S_{i',r}| \;\middle|\; S_{i',r}\right] \le e^{-\frac{9}{50}\epsilon|S_{i',r}|},$$

where the final inequality is an application of the Chernoff bound (7.2).

Note that in order for the procedure to halt with $|S_{i_r,r}| \le n^{2/3}$ it must be that some $\mathcal{B}_{i'}$

occurs for $i' \leq i$. Since $|S_{0,r}| = 32\epsilon^{-1} \ln n$, we have that

$$\mathbb{P}[|S_{i_r,r}| \leq n^{2/3}] \leq \mathbb{P}\left[\bigcup_{i'=1}^{i_r} \mathcal{B}_{i'}\right] \leq \sum_{i'=1}^{i_r} \mathbb{P}\left[\mathcal{B}_{i'} \mid |S_{i'-1,r}| \geq 32k\epsilon^{-1} \ln n\right] \leq \ell \cdot e^{-5k \ln n} = o(n^{-2k}).$$

A similar argument shows that when the procedure halts we also have $\mathbb{P}[|T_{j_r,r}| \leq n^{2/3}] = o(n^{-2k})$.

Now, to finish the short path from s to t, we generate the random arcs of R between S_{i_r} and T_{j_r}

$$\mathbb{P}\left[R \cap (S_{i_r} \times T_{j_r}) = \emptyset \mid |S_{i_r,r}| \geq n^{2/3} \wedge |T_{j_r,r}| \geq n^{2/3}\right] \leq (1-p)^{n^{4/3}} \leq e^{-\epsilon n^{1/3}} = o(n^{-2k}).$$

Putting all the pieces together, we have k disjoint paths, each consisting of a path of length at most $32\epsilon^{-1} \ln n$, followed by at most 2ℓ paths of length $d+1$ from $\bar{D}$ joined by edges from R, and finishing with a path of length at most $32\epsilon^{-1} \ln n$, for total length which numerical calculation shows is less than $100k\epsilon^{-1} \ln n$.

Since there are less than n^{2k} choices for the terminal pairs, the union bound shows that all choices of $2k$ nodes have short vertex disjoint paths linking them **whp**.

To conclude, we apply Lemma 7, which shows that these short paths will be discovered **whp** in polynomial time.

□

7.7 NL-completeness

The standard proof of the **NL**-completeness of (s,t)-connectivity makes nodes correspond to machine configurations, and includes out-edges for each pair of machine configurations which can follow directly one after the other. Since Turing machines have a finite set of symbols, there is a finite set of configurations which can follow a given configuration. So the reduction produces a graph with bounded in-degree and out-degree, and (s,t)-connectivity of bounded out-degree graphs is **NL**-complete.

Now, given a bounded degree instance of (s,t)-connectivity, we reduce it to an instance of strong connectivity by, for each node $i \neq s,t$, adding an arc from i to s and an arc from t to i. This does not add any path from s to t, so not-connected instances stay not-connected. If the original instance contained an (s,t)-path, the new instance is strongly connected, since there is an arc from any vertex to s, a path from s to t, and an arc from t to any vertex.

Unfortunately, this does not have bounded degree, since we increased the out-degree of t and in-degree of s both to $n-2$. To avoid this, we add 2 complete binary trees with depth $\lceil \log(n-2) \rceil$, one with all edges are directed away from the root (call this the out-tree) and the other with all edges directed towards the root (call this the in-tree). Then we connect t to the root of the out-tree, and as many leaves as necessary of the out-tree to 2 vertices of the original graph. This has the same effect as adding the arcs directly from t to everything, but increases the out-degree of t by 1 and adds $\mathcal{O}(n)$ vertices with in-degree

1 and out-degree 2. Similarly, we connect the root of the in-tree to s and each vertex of the original graph to a leaf of the in-tree, at most 2 vertices to each leaf. This has the same effect as adding the arcs directly from everything to s, but increases the in-degree of s by 1 and adds $\mathcal{O}(n)$ vertices with in-degree 2 and out-degree 1. This transformation can be implemented by a log-space transducer, since it only requires a little bit-shifting to produce the binary tree.

7.8 Conclusion

Spielman and Teng introduced smoothed analysis to help explain the success of the simplex algorithm. We have used smoothed analysis to examine the complexity of strong connectivity and (s,t)-connectivity. In the analysis of **NP**-hard optimization problems, one can judge the degree of difficulty based on approximability. Here we provide another measure of difficulty, based on the existence of heuristics for smoothed instances. We find that, according to this measure, strong connectivity seems easier than (s,t)-connectivity. This claim is somewhat surprising, since we determine if a graph is strongly connected by repeatedly checking if pairs of vertices are (s,t)-connected. However, strong connectivity is a more global property, so much so that in an instance like that of Section 7.5, even though there exists *some* pair which Algorithm $\mathcal{A}$ incorrectly concludes is not connected, there will *almost always* be another pair which Algorithm $\mathcal{A}$ *correctly* concludes is not connected, so the net result will be correct.

There are several directions for future research. First, it is easy to come up with a measure of difficulty; it is not easy to come up with a good measure. This chapter represents a piece of "experimental data". Are there other problems which appear solvable or insolvable by heuristics on smoothed instances? Does this property seem to relate to the difficulty of these problems in practice? This is especially interesting in the case of **NP**-complete problems. (Questions of this nature are investigated by Beier and Vöcking in [31].)

Second, it would be nice if Theorem 6 was about **BPL** instead of almost-**L**. It is not clear how to achieve this, however. This complication seems related to the limitations of log-space computation. An analogous result holds (by the same proof, even) for the possibility of heuristics for recognizing if a digraph contains edge disjoint paths connecting 2 terminal pairs. In that case, the worst-case problem is **NP**-complete, and since the reduction has room to store the perturbed copy, we can show that if $\mathbf{NP} \not\subseteq \mathbf{BPP}$ then no heuristic is successful on smoothed instances. A similar question addresses the growing out-degree case, as in Section 3. We know Algorithm $\mathcal{A}$ does not work, but does some other heuristic? It would be nice to have a result of a form similar to Theorem 6 suggesting no heuristic works on graphs with unbounded out-degree.

Finally, it would be natural to extend these results to computing k-strong-connectivity and being k-linked to work for smoothed instances. Here we face some unresolved technical difficulties.

Chapter 8

2-stage spanning tree

This chapter originally appeared as [119] as an extended abstract, and a complete version appeared as [120]. It studies the performance of a class of heuristics for a 2-stage stochastic version of the minimum spanning tree problem and the minimum spanning arborescence problem, when the edge weights are all selected independently and uniformly from the interval $[0, 1]$.

8.1 Introduction

Stochastic Programming refers to the general class of optimization problems where uncertainty is modeled by a probability distribution on the input variables. *Two stage optimization with recourse* is a widely used framework for stochastic optimization (see, e.g., the recent text by Birge and Louveaux [37]). This chaper considers a particular example of this approach in the context of a basic combinatorial optimization problem.

The 2-stage spanning tree problem is defined as follows: We wish to find a low cost spanning tree of the complete graph K_n. On Monday, say, we are given edge costs, $c_M : E \to \mathbb{R}$. We also know that on Tuesday we will be given alternative costs for each edge, $c_T : E \to \mathbb{R}$. We do not know what the costs c_T will be, but they are random and we know their joint distribution $\pi(\omega)$, $\omega \in \Omega$, the set of possibilities. On Monday we must choose a set of edges X_M and pay for them at Monday's prices. On Tuesday, Monday's prices will no longer be available. Some edges will be cheaper and some will be more expensive. We must now choose a set of edges X_T, *at Tuesday's prices* to complete a spanning tree. Our total cost will be $c_M(X_M) + c_T(X_T)$, and our goal is to choose the set of edges X_M which minimizes the expected total cost of the tree we create. Formally, we wish to compute

$$OPT = \min_{X_M} \; c_M(X_M) + \mathbb{E}\left[\min_{X_T}\{c_T(X_T) : \; X_M \cup X_T \text{ is a spanning tree}\}\right].$$

Anupam Gupta has pointed out that in the worst case, we can encode set cover as such a problem and so it probably cannot be efficiently approximated beyond a ratio of $O(\log n)$. A version of his proof is included in Section 8.5.

The inapproximability result requires a worst-case set of costs c_M and a worst-case distribution for c_T. This chapter will carry out a probabilistic analysis for instances where $c_M(e)$ and $c_T(e)$, $e \in E(K_n)$ are selected independently and uniformly from the interval $[0, 1]$.

It is well-known that if only Monday's costs are available then we can find a minimum spanning tree in polynomial time and that the expected cost Z_1 of the optimum solution is asymptotically equal to $\zeta(3) \sim 1.20205\ldots$, Frieze [135]. Here $\zeta(3) = \sum_{n=1}^{\infty} n^{-3}$. Furthermore, if we could accurately predict the future and could find a minimum spanning tree using costs $c_2(e) = \min\{c_M(e), c_T(e)\}$ then [135] shows that we could pick edges so that our optimal cost Z_2 is asymptotically $\zeta(3)/2 \sim .601028\ldots$.

---

procedure ThresholdHeu($\alpha; c_M$)
  $G := (V, E)$ where $e \in E$ if and only if $c_M(e) \leq \alpha$
  return minimum spanning forest of G with respect to costs c_M

Figure 8.1: Algorithm $\mathcal{A}_\alpha$: A threshold heuristic for 2-stage MST

We first examine the performance of a simple threshold heuristic, described formally in Figure 8.1. In words, $\mathcal{A}_\alpha$ is the algorithm that finds the minimum spanning forest of K_n that only uses edges of cost less than α on Monday and then completes the tree as cheaply as possible with new edges paid for at Tuesday's prices. Let A_α be the (random) value of the cost of the solution returned by $\mathcal{A}_\alpha$.

Theorem 8 *The best choice for α is $1/n$ in the sense that*

$$\mathbb{E}\left[A_{1/n}\right] = \zeta(3) - \frac{1}{2} + o(1) \leq \mathbb{E}\left[A_\alpha\right] + o(1)$$

for any choice of α.

Furthermore, for all α, the value A_α is concentrated around its mean.

The proof of Theorem 8 also gives a lower bound on the *value of stochastic solution (VSS)*, which is defined as the difference between the *expected result of using the expected value solution (EEV)* and the value of the optimal 2-stage solution OPT. To find the EEV, we observe that when the distribution of each Tuesday edge costs is replaced by its expected value, then the optimal solution **whp** ignores the Tuesday edges (which now have cost 0.5) and buys the whole tree on Monday. Thus, the EEV is asymptotically equal to the cost of buying the whole tree on Monday, which is asymptotically $\zeta(3)$. So we have $VSS = OPT - EEV \geq \frac{1}{2} - o(1)$.

Note that $\frac{\zeta(3)-1/2}{\zeta(3)/2} \sim 1.168\ldots$ and so **whp** $A_{1/n}$ is within 17% of optimal.

Recently, Dhamdhere, Ravi, and Singh showed that $\mathcal{A}_{\zeta(3)/n}$ is a constant factor approximation algorithm for instances where Monday's costs are arbitrary and Tuesday's costs are selected independently and uniformly between 0 and 1 [99].

A threshold algorithm is the best we can do if we do not take account of the structure of the costs for Monday's edges. Can we improve on this if we do? The answer is yes. We show that we can reduce the expected cost by at least a (very) small amount.

Theorem 9 *There is a polynomial time algorithm $\mathcal{A}^\star$ for selecting X_M whose (random) cost $A^\star$ satisfies*

$$\mathbb{E}[A^\star] \leq \zeta(3) - \frac{1}{2} - 10^{-256}.$$

We see therefore that the algorithm $\mathcal{A}_{1/n}$ is not optimal. Is it possible to asymptotically achieve $\zeta(3)/2$? Let OPT denote the minimum expected cost achievable by any 2-stage algorithm.

Theorem 10

$$OPT \geq \zeta(3)/2 + 10^{-5}$$

Theorem 10 is equivalent to a lower bound on the *expected value of perfect information (EVPI)*, which is defined between the difference between the value of the optimal 2-stage solution OPT and the expected value of the optimal solution when Tuesday's costs are known (the *wait-and-see value (WS)*). Theorem 10 shows that $EVPI = OPT - WS \geq 10^{-5}$.

Finding the the optimal choice of X_M and determining what can be done in polynomial time remain challenging open problems.

We continue with a directed version of this problem. Here we are given Monday and Tuesday costs for all the arcs of the complete digraph D_n. (The vertices of D_n are $\{1, \ldots, n\}$, and each ordered pair (i, j), $1 \leq i \neq j \leq n$, forms an arc in D_n). We now wish to find a low cost spanning arborescence rooted at vertex 1 i.e. a tree with arcs directed away from vertex 1. We first consider the threshold algorithm $\vec{\mathcal{A}}_\alpha$ which finds a minimum cost rooted forest using arcs from Monday of cost less than α and then completes it to a rooted arborescence after Tuesday's costs are revealed. Here $\alpha = 1/n$ is also the best choice: Let $\vec{A}_\alpha$ be the cost of the output from $\vec{\mathcal{A}}_\alpha$.

Theorem 11

$$\mathbb{E}\left[\vec{A}_{1/n}\right] = 1 - e^{-1} + o(1) \leq \mathbb{E}\left[\vec{A}_\alpha\right] + o(1)$$

for any choice of α.

Furthermore, for all α, the value $\vec{A}_\alpha$ is concentrated around its mean.

This turns out to be asymptotically optimal. Let $\overrightarrow{OPT}$ denote the minimum expected cost achievable by any 2-stage algorithm.

Theorem 12

$$\overrightarrow{OPT} \geq 1 - e^{-1} - o(1).$$

We prove Theorem 8 in Sections 8.2.1, 8.2.2, Theorem 9 in Section 8.2.3, Theorem 10 in Section 8.2.4, Theorem 11 in Section 8.3 and Theorem 12 in Section 8.3.2.

8.2 Undirected case

8.2.1 Threshold heuristic

Proof of Theorem 8 Fix some $0 < \alpha \leq 1$ and let T be the spanning tree produced by the threshold heuristic $\mathcal{A}_\alpha$, and let T_m and T_t be the edges bought on Monday and Tuesday

respectively. Then

$$\begin{aligned} c(T) &= \sum_{e\in T_m} c_m(e) + \sum_{e\in T_t} c_t(e) \\ &= \sum_{e\in T_m} \int_{p=0}^{1} \mathbf{1}_{\{c_m(e)\geq p\}} dp + \sum_{e\in T_t} \int_{p=0}^{1} \mathbf{1}_{\{c_t(e)\geq p\}} dp \\ &= \int_{p=0}^{1} \sum_{e\in T_m} \mathbf{1}_{\{c_m(e)\geq p\}} dp + \int_{p=0}^{1} \sum_{e\in T_t} \mathbf{1}_{\{c_t(e)\geq p\}} dp \end{aligned}$$

For any graph G, let $\kappa(G)$ denote the number of connected components in G.

Now, let G_p be the graph containing only edges with Monday cost less than p. Since T_m is the minimum spanning forest on edges with Monday cost less than α, for $p \leq \alpha$ we have by the greedy algorithm of Kruskal:

$$\sum_{e\in T_m} \mathbf{1}_{\{c_m(e)\geq p\}} = \kappa(G_p) - \kappa(G_\alpha).$$

Let H_p be the graph containing edges with Monday cost less than α or Tuesday cost less than p. Then, since T_t is a minimum spanning tree on the graph formed by contracting each component of T_m to a single vertex, we have

$$\sum_{e\in T_t} \mathbf{1}_{\{c_t(e)\geq p\}} = \kappa(H_p) - 1.$$

Linearity of expectations gives

$$\begin{aligned} \mathbb{E}\left[c(T)\right] &= \int_{p=0}^{\alpha} \mathbb{E}\left[\kappa(G_p) - \kappa(G_\alpha)\right] dp + \int_{p=0}^{1} \mathbb{E}\left[\kappa(H_p) - 1\right] dp \\ &= \int_{p=0}^{\alpha} \mathbb{E}\left[\kappa(G_p)\right] dp - \alpha\mathbb{E}\left[\kappa(G_\alpha)\right] + \int_{p=0}^{1} \mathbb{E}\left[\kappa(H_p)\right] dp - 1. \end{aligned} \tag{8.1}$$

We point out here that this implies

$$\int_{p=0}^{1} \mathbb{E}\left[\kappa(G_p)\right] dp = 1 + \zeta(3) + o(1), \tag{8.2}$$

since putting $\alpha = 1$ into (8.1) we have $\mathbb{E}\left[c(T)\right]$ is the expected value of the minimum spanning tree using Monday costs. As already mentioned, this is $\zeta(3) + o(1)$. But we have $\kappa(G_\alpha) = \kappa(H_p) = 1$ for all $p \leq 1$.

Now, G_p is identically distributed with the Erdős-Rényi random graph $G_{n,p}$ (in which each pair of n vertices appears as an edge independently with probability p) and H_p is identically distributed with the Erdős-Rényi random graph $G_{n,p'}$ for $p' = \alpha + p - \alpha p$. So

we have

$$\int_{p=0}^{1} \mathbb{E}\left[\kappa(H_p)\right] dp = (1-\alpha)^{-1} \int_{p'=\alpha}^{1} \mathbb{E}\left[\kappa(G_{p'})\right] dp'.$$

We may assume $\alpha \leq 2\log n/n$. If $\alpha > 2\log n/n$ then **whp** G_α is connected (see, for example, Bollobás [46, Thm. 7.3, p. 164]) which means that **whp** all the edges are purchased on Monday, and thus the expected cost $E[\mathcal{A}]_\alpha$ will be $\zeta(3) + o(1)$.

The integral $\int_{p'=\alpha}^{1} \mathbb{E}\left[\kappa(G_{p'})\right] dp'$ is bounded by $\int_{p'=0}^{1} \mathbb{E}\left[\kappa(G_{p'})\right] dp' = 1 + \zeta(3) + o(1)$, so we have (recalling that $\alpha = o(1)$)

$$\begin{aligned}\int_{p=0}^{\alpha} \mathbb{E}\left[\kappa(G_p)\right] dp + \int_{p=0}^{1} \mathbb{E}\left[\kappa(H_p)\right] dp &= \int_{p=0}^{1} E\left[\kappa(G_p)\right] dp - \frac{\alpha}{1-\alpha} \int_{p=\alpha}^{1} E[\kappa(G_p)] dp \\ &= \int_{p=0}^{1} \mathbb{E}\left[\kappa(G_p)\right] dp + o(1).\end{aligned}$$

So, altogether we have

$$\mathbb{E}\left[c(T)\right] = \zeta(3) + o(1) - \alpha\mathbb{E}\left[\kappa(G_\alpha)\right].$$

Set β so that $\alpha = \beta/n$ and put κ_T equal to the number of tree components. There are at most $n^{2/3}$ components of size at least $n^{1/3}$ and so we see that

$$\alpha\mathbb{E}\left[\kappa_T(G_\alpha)\right] = \frac{\beta}{n} \sum_{k=1}^{n^{1/3}} \binom{n}{k} k^{k-2} \left(\frac{\beta}{n}\right)^{k-1} \left(1-\frac{\beta}{n}\right)^{k(n-k)+(k^2-3k+2)/2} + o(1) \tag{8.3}$$

$$= \sum_{k=1}^{\infty} \frac{k^{k-2}}{k!} \left(\beta e^{-\beta}\right)^k + o(1). \tag{8.4}$$

Note that the sum in (8.4) is convergent, even for $\beta = 1$.

Let κ_N denote the number of non-tree components. Then we have

$$\alpha\mathbb{E}\left[\kappa_N(G_\alpha)\right] \leq \frac{\beta}{n} \sum_{k=1}^{n^{1/3}} \binom{n}{k} k^k \left(\frac{\beta}{n}\right)^k \left(1-\frac{\beta}{n}\right)^{k(n-k)} + o(1) \leq \frac{\beta}{n} \sum_{k=1}^{n^{1/3}} (\beta e^{1-\beta})^k + o(1) = o(1),$$

and so

$$\alpha\mathbb{E}\left[\kappa(G_\alpha)\right] = \sum_{k=1}^{\infty} \frac{k^{k-2}}{k!} \left(\beta e^{-\beta}\right)^k + o(1). \tag{8.5}$$

Now $\beta e^{-\beta}$ has a unique maximum at $\beta = 1$, which shows that the threshold $\alpha = 1/n$ is asymptotically best for the threshold heuristic.

Finally, we note that for $\beta = 1$,

$$\alpha\mathbb{E}\left[\kappa(G_\alpha)\right] = \sum_{k=1}^{\infty} \frac{k^{k-2}}{k!} e^{-k} + o(1) = \frac{1}{2} + o(1), \tag{8.6}$$

and so the threshold heuristic attains a value of $\zeta(3) - \frac{1}{2} + o(1)$.

(The last equation in (8.6) can be justified as follows: Consider the exponential generating function $U(x) = \sum_{k=1}^{\infty} \frac{k^{k-2}}{k!} x^k$ for the number of labeled trees with k vertices and the exponential generating function $T(x) = \sum_{k=1}^{\infty} \frac{k^{k-1}}{k!} x^k$ for the number of labeled rooted trees with k vertices. These satisfy $U(x) = T(x) - \frac{T(x)^2}{2}$ (equation (3.3) of [155]). Now $T(e^{-1}) = 1$ can be seen from the fact that $nT(e^{-1})$ is asymptotically equal to the number of vertices on trees in the random graph $G_{n,1/n}$. The sum in (8.6) is $U(e^{-1})$.)

8.2.2 Concentration

The goal of this section is to prove that for any constant $\lambda > 0$, there exists $\delta = \delta(\lambda) > 0$ such that for sufficiently large n,

$$\mathbb{P}[|A_\alpha - \mathbb{E}[A_\alpha]| \geq \lambda] \leq e^{-\delta n}.$$

We need only show this for λ sufficiently small, and it is convenient to define ϵ so that $\epsilon + 4(1-\epsilon)^{-1}(2\epsilon + H(\epsilon)) = \lambda$, where $H(x) = -x \ln x - (1-x)\ln(1-x)$ is the entropy function.

In our analysis we consider separately the contribution of long and short edges. Let $C = 2\epsilon^{-1}$, and let Z denote the total cost of the edges used by $\mathcal{A}_\alpha$ with edge cost at most C/n. Let $N = 2\binom{n}{2}$ and note that Z is a function of N i.i.d. random variables $X_1, \ldots, X_N$ (one for each edge for each day). Also, each X_i is uniformly distributed on $[0,1]$.

We will show Z is concentrated using a variant of the Symmetric Logarithmic Sobolev Inequality from [60]. Let Z_i' denote the same quantity as Z, but with the variable X_i replaced by an independent copy X_i'. Then a simplified form of the Symmetric Logarithmic Sobolev Inequality [60, Corollary 3] says that if

$$\mathbb{E}\left[\sum_{i=1}^{N} (Z - Z_i')^2 \mathbf{1}_{Z > Z_i'} \,\middle|\, X_1, \ldots, X_N\right] \leq c$$

then for all $t > 0$,

$$\mathbb{P}[Z > \mathbb{E}[Z] + t] \leq e^{-t^2/4c},$$

and if

$$\mathbb{E}\left[\sum_{i=1}^{N} (Z_i' - Z)^2 \mathbf{1}_{Z_i' > Z} \,\middle|\, X_1, \ldots, X_N\right] \leq c$$

then for all $t > 0$,

$$\mathbb{P}[Z < \mathbb{E}[Z] - t] \leq e^{-t^2/4c}.$$

Changing the value of one edge can change the value of Z by at most C/n, so $(Z - Z_i')^2 < (C/n)^2$. Let I denote the indices of the edges which contribute to Z. If $i \notin I$ then $Z_i' < Z$

implies $X_i' \le C/n$. So

$$\sum_{i=1}^{N}(Z - Z_i')^2 \mathbf{1}_{Z>Z_i'} \le \sum_{i\in I}(C/n)^2 + \sum_{i\notin I}(C/n)^2 \mathbf{1}_{X_i'<C/n}.$$

Since there are less than n terms in the first sum and less than n^2 terms in the second sum, we have

$$\mathbb{E}\left[\sum_{i=1}^{N}(Z - Z_i')^2 \mathbf{1}_{Z>Z_i'} \;\middle|\; X_1, \ldots, X_N\right] \le C^2/n + C^3/n \le 2C^3/n.$$

If $i \notin I$ then we also have that $Z_i' > Z$ implies $X_i' \le C/n$. So we also have

$$\mathbb{E}\left[\sum_{i=1}^{N}(Z_i' - Z)^2 \mathbf{1}_{Z_i'>Z} \;\middle|\; X_1, \ldots, X_N\right] \le C^2/n + C^3/n \le 2C^3/n.$$

Therefore,

$$\mathbb{P}\left[|Z - \mathbb{E}[Z]| \ge \epsilon\right] \le 2e^{-\epsilon^2 n/8C^3} = 2e^{-\epsilon^5 n/64}.$$

Let Z' denote the total cost of the edges used by $\mathcal{A}_\alpha$ with edge cost at least C/n. We will show that $Z' \ge \lambda - \epsilon$ with exponentially small probability.

Let G be the graph containing edges with Monday or Tuesday cost less than C/n. Then G is identically distributed with $G_{n,p}$ for $p = 2C/n - (C/n)^2$. Let S denote the set of vertices that are not in the giant (more precisely, largest) component of G. We will obtain a exponential bound on the probability that $|S| \ge \epsilon n$. To do so, we let $\mathcal{B}_1$ denote the event "there exists a set T such that $\epsilon n \le |T| \le n/2$ and no edge of G crosses the cut between T and $\bar{T}$." Note that in order for $|S| \ge \epsilon n$, it is necessary that event $\mathcal{B}_1$ holds: if $|\bar{S}| \ge \epsilon n$, then (since $|S|$ also exceeds ϵn) either $T = S$ or $T = \bar{S}$ shows that $\mathcal{B}_1$ occurs; if $|\bar{S}| \le \epsilon n$, then all connected components of the graph have size at most ϵn and we can choose T to be the union of an appropriate collection of connected components.

Since $C = 2\epsilon^{-1}$, we have

$$\mathbb{P}[|S| \ge \epsilon n] \le \mathbb{P}[\mathcal{B}_1] \le \sum_{k=\epsilon n}^{n/2} \binom{n}{k} \left(1 - \frac{C}{n}\right)^{2k(n-k)} \le \sum_{k=\epsilon n}^{n/2} e^{n - 2Ck(1-k/n)} \le ne^{-n}. \tag{8.7}$$

Z' can be bounded by the sum of (i) the edges of length $> C/n$ in the minimum spanning tree using Monday costs and (ii) the sum of the edges of length $> C/n$ in a minimum spanning tree of the graph obtained by shrinking the components of the Tuesday forest. (ii) is stochastically less than by (i). The sum in (i) can be bounded by the sum over the vertices $s \in S$ of the length of the cheapest edge from s to the giant component (more precisely largest component) of the graph spanned by the edges of length $< C/n$.

We finish by calculating an upper bound on the probability that any subset of size ϵn has the sum of the minimum cost edges exceeding $(\lambda - \epsilon)/2$. Let V_1 denote the minimum of $n' := (1 - \epsilon)n$ independent random variables each uniformly distributed in $[0, 1]$. Then

$\mathbb{E}[V_1] = \frac{1}{n'+1}$, and

$$\mathbb{E}\left[e^{tV_1}\right] = \int_{x=0}^{1} e^{tx} n'(1-x)^{n'-1} dx = 1 + \sum_{k\geq 1} \frac{t^k}{n'(n'+1)\cdots(n'+k-1)} \leq \left(1 + \frac{2t}{n'}\right).$$

(The second equality follows from integration by parts, inequality holds for $t \leq n'/2$).

Then, for any set T with $|T| = k$,

$$\mathbb{P}\left[\sum_{v\in T} V_1(v) \geq \lambda\right] = \mathbb{P}\left[e^{\frac{n'}{2}\sum_{v\in T} V_1(v)} \geq e^{\lambda n'/2}\right] \leq e^{-\lambda n'/2}\mathbb{E}\left[e^{n'V_1/2}\right]^k \leq e^{-\lambda n'/2+k}.$$

Let $\mathcal{B}_2$ denote the event "there exists a set T with $|T| \leq \epsilon n$ and $\sum_{v\in T} V_1(v) \geq (\lambda-\epsilon)/2 = 2(1-\epsilon)^{-1}(2\epsilon + H(\epsilon))$". Then we have

$$\mathbb{P}[\mathcal{B}_2] \leq \sum_{1\leq k\leq \epsilon n} \binom{n}{k} e^{-\epsilon n - H(\epsilon)n} \leq \epsilon n e^{-\epsilon n}. \tag{8.8}$$

We combine (8.7) and (8.8) to show that the probability Z' exceeds $\lambda - \epsilon$ is small.

$$\mathbb{P}\left[Z' \geq \lambda - \epsilon\right] \leq \mathbb{P}[|S| \geq \epsilon n] + 2\mathbb{P}[\mathcal{B}_2] \leq ne^{-n} + 2\epsilon n e^{-\epsilon n}.$$

Finally,

$$\begin{aligned}\mathbb{P}[|\mathcal{A}_\alpha - \mathbb{E}[\mathcal{A}_\alpha]| \geq \lambda] &\leq \mathbb{P}[|Z - \mathbb{E}[Z]| \geq \epsilon] + \mathbb{P}[Z' \geq \lambda - \epsilon] \\ &\leq 2e^{-\epsilon^5 n/64} + ne^{-n} + 2\epsilon n e^{-\epsilon n}.\end{aligned}$$

8.2.3 Beyond the threshold heuristic

We can achieve a slightly better expected value than the threshold heuristic $\mathcal{A}_{1/n}$ by being more careful about edges with cost near the threshold.

Let ℓ be a positive integer and let $\epsilon > 0$ be a small positive constant and let F be the minimum spanning forest on the edges with Monday cost less than $(1-\epsilon)/n$. Let an edge $e = \{u, v\}$ be *bad* if it has Monday cost $c_M(e) \in [(1-\epsilon)/n, 1/n]$, and for $x = u, v$ there are:

(A) Exactly ℓ vertices w for which $c_M(x, w) < (1-2\epsilon)/n$. Denote this set of vertices by C_x.

(B) No vertices w for which $c_M(x, w) \in [(1-2\epsilon)/n, 1/n]$.

(C) No vertices $w \in C_x$ and $y \notin \{x\} \cup C_x$ for which $c_M(y, w) < 1/n$.

If e is bad then e will be part of an isolated tree of $G_{1/n}$ containing $2\ell + 1$ edges and e will be chosen by $\mathcal{A}_{1/n}$.

Let T_1 be the tree constructed by $\mathcal{A}_{1/n}$ and let T_2 be obtained by taking the minimum spanning forest which uses edges e with $c_M(e) < 1/n$ which are not bad, and then completing

this forest to a tree as cheaply as possible with edges at Tuesday's costs. We will show that

$$\mathbb{E}\left[T_1 - T_2\right] \geq 10^{-256} \tag{8.9}$$

and so completing the proof of Theorem 9.

We must estimate the expected savings if we leave out the bad edges and only the bad edges from the threshold solution. In this case, $\{x\} \cup C_x$, $x = u, v$ are trees of the forest of the edges chosen on Monday.

We consider the contribution from the removal of a single bad edge $e = \{u, v\}$. We expose the costs of the edges carefully to avoid unpleasant conditioning. First we expose the Monday cost of e. The probability $c_M(e)$ is in the correct range is ϵ/n. If $c_M(e)$ is in this range, we expose the Monday costs of the other edges incident to u and v. The probability that the costs of the other edges are in the correct range is

$$\left(\binom{n-1}{\ell}\left(\frac{1-2\epsilon}{n}\right)^{\ell}\left(1-\frac{1}{n}\right)^{n-2-\ell}\right)^2 \geq \frac{(1-2\epsilon)^{2\ell}}{e^2(\ell!)^2}(1-o(1)).$$

Now, we expose the Monday costs of the neighbors of $C_u \cup C_v$. The probability that (C) holds is $(1-1/n)^{2\ell(n-2-2\ell)} = e^{-2\ell}(1+o(1))$.

Thus the expectation of the number of bad edges b is given by

$$\mathbb{E}[b] = (1+o(1))\frac{\epsilon(1-2\epsilon)^{2\ell}e^{-2\ell-2}}{2(\ell!)^2}n. \tag{8.10}$$

We now expose all the Monday and Tuesday costs between the $n-2-2\ell$ vertices that are not part of C_u and C_v. Let H be the graph containing all edges just exposed with Monday or Tuesday cost at most $(1-2\epsilon)/n$. Note that H is identically distributed with $G_{n',p}$ for $n' = n-2-2\ell$ and $p = (1+o(1))(2-4\epsilon)/n$. If $\epsilon < 1/4$ then H has a giant component K_H **qs** *. We expose the remaining edge costs and let X_u (resp. X_v) be the minimum cost of a Tuesday edge from C_u (C_v) to K_H, assuming that it exists. The size of K_H is at least $\beta n(1-o(1))$ **qs**, where β is the root of $\beta + e^{-2(1-2\epsilon)\beta} = 1$ in the interval $(0,1)$. We take $\epsilon = 0.1$ and then $\beta > 0.7$. So, for $\ell = 100$ we have $E[X_u] = E[X_v] = \frac{1+o(1)}{\ell\beta n} \leq 0.02n^{-1}$. For each bad edge $e = \{u, v\}$ we then have expected cost savings of at least

$$\frac{1-\epsilon}{n} - \max\left\{\frac{1-2\epsilon}{n}, X_u + X_v\right\} \geq \frac{0.1}{n}. \tag{8.11}$$

We can prove (8.11) as follows: Let $e = \{u, v\}$ be bad. $e \notin T_2$ and there is a path from u to v which goes to a vertex of C_u, goes to H via an edge of length X_u, traverses H and then goes via an edge of length X_v to a vertex of C_v and then to v. If A, B are the components

*Recall that a sequence of events $\mathcal{E}_n$ occurs *quite surely* (**qs**) if $\mathbb{P}(\mathcal{E}_n) = 1 - O(n^{-K})$ for any $K > 0$.

of $T_1 - e$ then at least one edge $f \notin T_1$ of P will join A to B. We observe that

$$\min\{c_M(f), c_T(f)\} \le \max\left\{\frac{1-2\epsilon}{n}, X_u, X_v\right\} \le \max\left\{\frac{1-2\epsilon}{n}, X_u + X_v\right\}.$$

So, if we replace e by f in T_1 we will, by (8.11), save at least $\frac{0.1}{n}$. If we repeat this for all bad edges, then we will have a tree containing all of the Monday purchased edges and it will, in expectation, be at least $\frac{0.1\mathbb{E}[b]}{n}$ cheaper. We obtain (8.9) by using this together with (8.10) with $\ell = 100$.

8.2.4 A lower bound on OPT

If we could see all the Monday and Tuesday costs before selecting any edge then we could find a spanning tree with cost $\sim \zeta(3)/2$. Since we have to make some decisions before we see the Tuesday costs, it seems likely that our solution should, in expectation, cost at least $\zeta(3)/2 + \epsilon$, for some small ϵ. This is the content of Theorem 10.

Let C be a positive constant, (which we will eventually take to be 3, to obtain a concrete bound). Consider the edges we buy on Monday with cost exceeding $\frac{C}{n}$. Let

$$\epsilon = \beta_C e^{-(2C+3)}/2$$

where β_C is the solution to $\beta + e^{-(C-1)\beta} = 1$ in the interval $(0, 1)$.

We will see that if we buy more than ϵn of these edges, then we will regret our purchase on Tuesday. We also argue that if we buy less than ϵn, then we will regret it too.

Case 1: Suppose X_M contains at least ϵn edges with $c_M(e) \ge \frac{C}{n}$, and let $\{e_1, e_2, \ldots, e_m\}$ be these edges (where $m \ge \epsilon n$). Let H be the graph consisting of all the edges e' with $c_T(e') < \frac{C-1}{n}$. Then (for any $C > 2$), H contains a giant component K_H with size $\beta_C n(1 - o(1)$ **whp**. For $i = 1, \ldots, m$, if e_i has both end vertices in K_H, then we can find a cheaper spanning tree T_i by removing e_i from T_{i-1} and adding an edge from H on Tuesday. This will decrease the cost of the solution by at least $1/n$. Since each edge e_i has both vertices in K_H with probability $\sim \binom{\beta_C n}{2}/\binom{n}{2} \sim \beta_C^2$, the 2-stage solution exceeds the optimal solution by at least $\beta_C^2\epsilon - o(1)$ in expectation.

Case 2: Suppose X_M contains less than ϵn edges with $c_M(e) \ge \frac{C}{n}$. For a vertex v, let $\mathcal{E}_v$ be the event "the cheapest Monday edge incident to v has cost between $\frac{C}{n}$ and $\frac{C+1}{n}$ and the other endpoint is in K_H". Then $\mathbb{P}\big[\mathcal{E}_v \;\big|\; |K_H|\big] = |K_H|\frac{1}{n}\left(1 - \frac{C+1}{n}\right)^{n-2}$, and so $\mathbb{P}[\mathcal{E}_v] \sim \beta_C e^{-(C+1)}$.

Let $\mathcal{E}'_v$ be the event "there is no edge incident to v with Tuesday cost less than $\frac{C+2}{n}$". Then $\mathbb{P}[\mathcal{E}'_v] = \left(1 - \frac{C+2}{n}\right)^{n-1} \sim e^{-(C+2)}$. If $\mathcal{E}_v$ and $\mathcal{E}'_v$ occur then not buying the edge from v to K_H with cost less than $(C+1)/n$ on Monday results in paying at least $\frac{1}{n}$ more than optimal to connect v on Tuesday. But we only take ϵn edges on Monday with $c_M(e) \ge \frac{C}{n}$, so we expect to pay this penalty on at least $n\beta_C e^{-(2C+3)} - \epsilon n$ vertices, and so our 2-stage solution exceeds optimal by at least $\beta_C e^{-(2C+3)}/4$ in expectation, after accounting for the fact that one edge has 2 endpoints.

Taking $C = 3$, numerical computation shows that the 2-stage solution exceeds optimal by at least 10^{-5}.

8.3 Spanning arborescence problem

The directed version of this problem is to build a cheap spanning out-arborescence rooted at a fixed vertex r. Given a random cost for each directed edge on Monday and a distribution for the random cost for each directed edge on Tuesday, find directed edges to buy on Monday to minimize the expected total cost when you buy the missing edges on Tuesday. In other words, compute

$$OPT = \min_{X_M} \; c_M(X_M) \\ + \mathbb{E}\left[\min_{X_T}\{c_T(X_T)\colon\; X_M \cup X_T \text{ is a spanning arborescence rooted at } r\}\right].$$

In this case there is a lower bound that matches the threshold heuristic.

8.3.1 Threshold heuristic

This comprises two phases:

Phase 1: For each vertex, if the cheapest in-edge on Monday has cost at most α we will buy it, and otherwise we will wait till Tuesday and buy the cheapest in-edge available. This does not define an arborescence, it defines a functional digraph, with all in-degrees equal to 1. This consists of a collection of vertex disjoint cycles $C_1, C_2, \ldots, C_m$, and for each vertex v in $C_1 \cup C_2 \cup \cdots \cup C_m$ there is an arborescence directed from v.

Phase 2: We delete the arc directed into r. We then delete one (arbitrary) arc from each cycle that remains. At this point we have m' vertex disjoint directed rooted trees $T_1, T_2, \ldots, T_{m'}$, say, where $m \le m' \le m+1$. Assume that r is the root of T_1. Now we make a spanning arborescence as follows:
For $i = m', m'-1, \ldots, 2$ we do the following:
Find the cheapest arc, at Tuesday's prices, into T_i from a vertex in $T_1 \cap T_2 \cap \ldots \cap T_{i-1}$.
If this arc came from T_j then this creates a rooted tree T_j' from the vertices of T_j, T_i.
T_j' replaces T_j and T_i disappears.

Note that since the arc removed in Phase 2 is chosen arbitrarily, this procedure can be implemented in the 2-Stage framework: on Monday, we leave some edge out of any cycle that Phase 1 wants to buy. This does not require any knowledge of the Tuesday costs.

Analysis of Phase 1
We find that the expected cost of the arcs chosen is given by

$$n\left(\int_{x=0}^{\alpha}(n-1)x(1-x)^{n-2}dx + (1-\alpha)^{n-1}\int_{x=0}^{1}(n-1)x(1-x)^{n-2}dx\right) \\ = n\left(\frac{1}{n} - \frac{(1-\alpha)^{n-1}}{n}(1+\alpha n-\alpha) + \frac{(1-\alpha)^{n-1}}{n}\right) = 1-(n-1)\alpha(1-\alpha)^{n-1}.$$

This is minimized at $\alpha = 1/n$ giving a value which is asymptotically equal to $1 - e^{-1}$.

Analysis of Phase 2
It remains to show that the cost added in this phase is $o(1)$ **whp**. First of all, it is known (see, e.g., [46], Ch. 14.5), that for some $K > 0$, $m \leq K \log n$ with probability at least $1 - O(n^{-2})$. An easy calculation shows that with probability $1 - o(n^{-2})$ over Tuesdays' prices, for every ordered partition (V_1, V_2) of V the cheapest Tuesday's arc from V_1 to V_2 has cost at most $\frac{4 \log n}{n}$. Indeed, the probability that this is not so can be bounded from above by

$$\begin{aligned}\sum_{k=1}^{n-1} \binom{n}{k} \left(1 - \frac{4 \log n}{n}\right)^{k(n-k)} &\leq 2 \sum_{k=1}^{n/2} \binom{n}{k} \left(1 - \frac{4 \log n}{n}\right)^{k(n-k)} \leq 2 \sum_{k=1}^{n/2} \left(\frac{en}{k}\right)^k e^{-\frac{4 \log n}{n} \cdot \frac{kn}{2}} \\ &= o(n^{-2}) \ .\end{aligned}$$

Assuming the above conditions hold the arcs added at Phase 2 increase the total weight of the obtained solution by at most $O(\frac{\log^2 n}{n})$.

Concentration
The proof is analogous to the proof in section 8.2.2. Given a $\lambda > 0$, we pick the appropriate constant C, and use Azuma's inequality to show the total cost of the arcs with cost less than C/n is concentrated around its mean. Then we show that the probability the total cost of the remaining arcs is anything significant is exponentially small, by showing that (with probability exponentially close to 1) there are not too many vertices left unconnected, and for any small set of vertices, there is a set of edges connecting them to the remaining vertices which doesn't cost anything significant.

8.3.2 Matching lower bound on $\overrightarrow{OPT}$

In any feasible solution each vertex v besides the root r has to have a unique edge directed to it. So we can obtain a lower bound on what is achievable by looking at each vertex individually. For any realization of c_M, taking expectations over c_T we have

$$\begin{aligned}&\min_{X_M} \left\{ c_M(X_M) + \mathbb{E}\left[\min_{X_T} c_T(X_T) \colon \ X_M \cup X_T \text{ is an arborescence} \right] \right\} \\ &\quad \geq \sum_{v \neq r} \min \left\{ \min_{w \neq v} \{c_M(w,v)\}, \mathbb{E}\left[\min_{w \neq v} \{c_T(w,v)\} \right] \right\} = \sum_{v \neq r} \min \left\{ \min_{w \neq v} \{c_M(w,v)\}, 1/n \right\} .\end{aligned}$$

And so taking expectations over c_M, we obtain, where X_i are independent, uniformly distributed between 0 and 1,

$$\begin{aligned}\overrightarrow{OPT} &\geq (n-1)\mathbb{E}\left[\min\{1/n, X_1, X_2, \ldots, X_{n-1}\}\right] \\ &= (n-1)\left(\int_{x=0}^{1/n}(n-1)x(1-x)^{n-2}dx + \left(1-\frac{1}{n}\right)^{n-1}\int_{x=1/n}^{1}\frac{1}{n}dx\right) \\ &\sim 1-e^{-1}.\end{aligned}$$

8.4 Open questions

As far as this piece of work is concerned, the main open question is how to close the gap between the results of Theorems 9 and 10.

Another natural question might be to consider a 2-stage version of the random assignment problem. See Aldous [11], [12], Linusson and Wästlund [184] and Nair, Prabhakar and Sharma [201] for recent work on the standard *one-stage* analysis. In principal, one could try to carry out a similar 2-stage probabilistic analysis for any combinatorial optimization problem.

8.5 Hardness of approximation in worst case

We describe a gap preserving reduction from set cover. Let $S_1, S_2, \ldots, S_m \subseteq [n]$ be a set cover instance. We construct an MST instance with $n+m+1$ vertices by defining the function c_M and the random function c_T. Denote the vertices by $\{r, v_1, \ldots, v_m, 1, \ldots, n\}$. Set the Monday edge cost of $\{r, v_i\}$ to 1 and set all the other Monday edge costs to ∞.

$$c_M(\{u,v\}) = \begin{cases}1, & \text{if } \{u,v\} = \{r, v_i\};\\ \infty, & \text{otherwise.}\end{cases}$$

Make the Tuesday edge costs uniformly distributed over n functions, where the j-th function sets to ∞ the cost of edges in the cut separating $T_j = \{j\} \cup \{v_i \colon S_i \ni j\}$ from the rest of the graph, and sets the other edges costs to 0.

$$c_T^{(j)}(\{u,v\}) = \begin{cases}\infty, & \text{if } \{u,v\} \in (T_j, \overline{T_j});\\ 0, & \text{otherwise.}\end{cases}$$

If $S_{i_1} \cup S_{i_2} \cup \cdots \cup S_{i_k} = [n]$ then by buying Monday edges $\{r, v_{i_j}\}$ where $j = 1, \ldots, k$, we can complete the spanning tree on Tuesday with 0-cost edges for any future.

On the other hand, consider any set X_M of Monday edges such that the expected total cost of the spanning tree is finite. Then each $\{u,v\} \in X_M$ must have the form $\{r, v_{i_j}\}$. Consider set of sets corresponding to these edges, $\{S_{i_1}, \ldots, S_{i_k}\}$. For any $\ell \in [n]$, we must have $\ell \in S_{i_j}$ for some i_j; otherwise with probability $1/n$, we realize future ℓ, and have to buy an infinite cost edge across cut $(T_\ell, \overline{T_\ell})$.

Chapter 9

Facility Location

This chapter originally appeared as [122]. It studies the performance of some approximation algorithms designed for the metric facility location problem when these algorithms are applied instances generated by a geometric random distribution.

9.1 Introduction

Many optimization problems are NP-hard. This is an unfortunate fact of life. As discussed in Chapter 1, there are a variety of approaches to dealing with this fact. One popular approach is to find approximation algorithms with provably good worst-case performance guarantees. Another approach (the primary focus of this thesis) is to design heuristics which work well "on average". In this chapter we will combine the approaches, by analyzing an approximation algorithm in a probabilistic setting. The aim is to investigate the notion that such algorithms will "typically" do better than their worst-case guarantees.

In the uncapacitated facility location problem (UFLP) we are given a set of facilities $\mathcal{F}$ and a set of cities $\mathcal{C}$. For every facility $i \in \mathcal{F}$ there is a cost f_i for opening that facility, and for every facility-city pair $(i, j) \in \mathcal{F} \times \mathcal{C}$ there is a cost $c_{i,j}$ for connecting facility i to city j. There are no bounds on the number of cities that can be connected to a facility. Thus, if we open the set of facilities $F \subseteq \mathcal{F}$ then each city j will connect to the open facility with cheapest connection cost, and the total cost will be

$$c(F) = \sum_{i \in F} f_i + \sum_{j \in \mathcal{C}} \min_{i \in F} c_{i,j}.$$

The goal is to find a set of facilities F that will minimize the total cost $c(F)$.

Unfortunately, the problem is NP-hard, as it contains set-cover as a special case. It has been the focus of a great deal of attention from many perspectives. In the 1980's, the Operations Research community focused on branch and bound algorithms for solving it, which led to some considerable success, see for example [174]. From that period, there is also some worst-case analysis of the performance of greedy heuristics [93] and a probabilistic analysis of the related k-median problem [8]. More recently, the Theoretical Computer Sci-

ence community has placed a significant emphasis on finding approximation algorithms for NP-hard problems and one of its most notable successes has been in finding constant factor approximations for this problem when the connection costs obey the triangle inequality. The first algorithm to obtain a constant factor approximation was based on LP rounding [211] and subsequent approaches based on LP rounding improved the constant to $1 + 2/e$ [75] and then to 1.58 [219]. Alternative approaches to approximating the solution are based on local search techniques [172], primal-dual schema [154] and combinations of these [72]. At the present time the best approximation guarantee that is obtainable in polynomial time is 1.52, due to Mahdian, Ye and Zhang [189]. This is a greedy augmentation algorithm, and in this chapter, we will focus our attention on it and on 2 related greedy algorithms [153].

It is likely that approximation algorithms will find solutions closer to optimal than their guarantees guarantee. How much closer? One way to provide some answer to this question is via an experimental study, which is exactly the approach of [26, 150] and is also considered in Section 7 of [153]. Another way, which we will follow in this chapter, is to consider theoretically the result of applying the algorithms to an appropriate random instance. Since the constant factor approximation algorithms are only supposed to work on metric instances, we rule out one common random model, in which all distances are chosen independently and uniformly from $[0, 1]$. Another random model we do not study comes from choosing all distances from a discrete distribution that takes only the values 1 and 2. The random model we use will be geometric in nature, formed by placing points uniformly at random in the unit square. For additional reference on combinatorial optimization over instances derived from random points, see [204, 216, 230]. Although it is possible to design algorithms to take advantage of the special structure of these instances, that is not the focus of the current investigation. Instead of first choosing a distribution over instances and then designing an algorithm to work **whp** over this distribution, we begin by choosing the algorithms to study and then choose an interesting (but tractable) distribution of instances on which to run them.

9.1.1 Random model

We will study random instances formed by choosing n points $\mathcal{X} = \{X_1, X_2, \ldots, X_n\}$ uniformly at random in the unit square $[0, 1]^2$. We assume that each point represents a city and also the possible location of a facility. For simplicity we will use the ℓ_∞ distance between each facility-city pair as the connection cost (the techniques presented below would also work for the ℓ_1 norm, but for the ℓ_2 norm, additional effort would be needed to replace the results from Section 9.2).

Let m be a positive integer satisfying $m = o((n/\log n)^{1/2})$. Then let $\alpha = m^{-1}$ and define $\omega = m^{-1}(n/\log n)^{1/2}$, so that $\omega \to \infty$ with n.

We will give every facility the same opening cost,

$$f = \frac{1}{6}\alpha^3 n.$$

We have selected these values for later convenience in notation, and summarize it in the following table. It is really the facility cost f that controls the structure of the optimal

solution. As f tends to ∞, the optimal solution will open 1 facility in the center of the square and connect everything to it. As f tends to 0, the optimal solution will open a facility at every city. Section 9.2 will show that the transition between these extreme behaviors is described by f as parameterized above. For $f = \frac{1}{6}m^{-3}n$, the optimal solution will open about m^2 facilities.

$$\omega \to \infty \qquad m = \omega^{-1}\sqrt{\frac{n}{\log n}} \qquad \alpha = \omega\sqrt{\frac{\log n}{n}}$$
$$f = \frac{1}{6}\omega^3\frac{(\log n)^{3/2}}{\sqrt{n}}$$

We denote the ℓ_∞ distance between two points X_i and X_j by $d(X_i, X_j)$. All logarithms are base e.

We initially expected to prove that the algorithm of [189], which has worst-case approximation ratio 1.52, was asymptotically optimal i.e. that **whp**, as $n \to \infty$, the ratio of the cost of the solution found by the approximation algorithm and the optimum tends to 1. Instead we give a proof of the following: Let OPT denote the value of a minimum cost solution. The algorithm of [189] is similar in spirit to the 2 algorithms given in [153], which have worst-case approximation ratios of at most 1.861 and 1.61. We denote these approximation algorithms by H_1, H_2, H_3, and recall their descriptions in detail in Section 9.2. We let Z_i denote the value of the solution found by H_i.

Theorem 13 *There exists a positive constant $\epsilon > 0$ such that for $i = 1, 2, 3$,* **whp**

$$\frac{Z_i}{OPT} \geq 1 + \epsilon.$$

On the other hand it is not difficult to describe a "trivial heuristic" which *is* asymptotically optimal and so it is disappointing that these sophisticated approximation algorithms are in fact beaten by triviality **whp**.

9.1.2 Outline

In the next section we describe the greedy approximation algorithms and the trivial heuristic in detail, and give a non-rigorous explanation of "what goes wrong" to prevent the approximation algorithms from finding an asymptotically optimal solution.

Since our non-rigorous explanation will rely heavily on the asymptotic optimality of the trivial heuristic, we prove that the heuristic is asymptotically optimal in Section 9.3. The proof has 2 parts. First we obtain an upper bound that holds **whp** on the value of the solution found by the heuristic. Since the heuristic is so simple, this only requires us to consider basic probabilistic arguments. Some of these recur frequently enough to merit little lemmas, which are stated and proved in Section 9.3.1. Then we obtain an asymptotically matching lower bound that holds **whp** on the value any solution. We do this by constructing a solution to the dual of the LP-relaxation which is feasible **whp**.

The remainder of the chapter proves Theorem 13. To do so, in Section 9.4.1, we state and prove some lemmas which show that the structure of *any* near optimal solution must take a certain form; it must choose facilities to open so that, for most open facilities, the region of the plane which is closer to that facility than any other is approximately a square of a certain size and is approximately centered on the facility. Lemma 14 from Section 9.4.1 is a quantitative version of this. Roughly, it says that if there are ϵn facilities opened which violate these conditions then the solution will be a $1 + \delta$ factor away from optimal.

To complete the proof of Theorem 13, in Section 9.4.2 we show that the approximation algorithms from Section 9.2 open too many facilities which do not meet the requirements for a close to optimal solution.

9.2 Approximation Algorithms

The approximation algorithms we consider are all similar. We first recall Algorithm 1 of [153] (which is most convenient for us in its restated form).

Approximation Algorithm 1

(a) The algorithm starts at time 0. Initially, each city is defined to be *unconnected*. The set of unconnected cities is denoted by U. All facilities are considered to be *unopened* and $\delta_i = 0$ for $i \in C$, the set of cities.

(b) While $U \neq \emptyset$, increase the time and simultaneously for every city $j \in U$ increase the parameter δ_i at the same rate, until one of the following events occurs:

1. For some unconnected city i, and some open facility j, $\delta_i = d(i, j)$. In this case, connect city i to facility j and remove j from U.
2. For some unopened facility j, $\sum_{i \in U} \max\{0, \delta_i - d(i, j)\} = f_j$. In this case open this facility and for every unconnected city with $\delta_i \geq d(i, j)$, connect i to j and remove it from U.

Now we recall Algorithm 2 of [153], which is very similar to Algorithm 1, but allows connected cities to contribute funds towards opening additional facilities.

Approximation Algorithm 2

(a) The algorithm starts at time 0. Initially, each city is defined to be *unconnected*. The set of unconnected cities is denoted by U. All facilities are considered to be *unopened* and $\delta_i = 0$ for $i \in C$, the set of cities. We denote by π the mapping from connected cities to open facilities.

(b) While $U \neq \emptyset$, increase the time and simultaneously for every city $j \in U$ increase the parameter δ_i at the same rate, until one of the following events occurs:

1. For some unconnected city i, and some open facility j, $\delta_i = d(i, j)$. In this case, connect city i to facility j and remove j from U.

2. For some unopened facility j, we have

$$\sum_{i \in U} \max\{0, \delta_i - d(i,j)\} + \sum_{i \notin U} \max\{0, c_{i,j} - c_{i,\pi(i)}\} = f_j.$$

In this case open this facility and for every unconnected city with $\delta_i \geq d(i,j)$, connect i to j and remove it from U, and for every connected city with $c_{i,j} < c_{i,\pi(i)}$ change the facility to which i connects from $\pi(i)$ to j.

Now, we recall Algorithm 3, which appears in [189] and currently has the best proven bound on worst-case approximation ratio.

Approximation Algorithm 3

(a) In the first phase, the algorithm scales up the opening costs of all facilities by a constant $\delta = 1.504$, and uses Algorithm 2 to find a solution to the problem with these new costs.

(b) In the second phase, the algorithm considers the unmodified costs and performs a greedy augmentation to the solution found in phase 1. Let C denote the total connection cost in the phase 1 solution. For each unopened facility j, let C_j denote the total connection cost when j is also opened. If the maximum over unopened facilities of the ratio $(C - C_j - f_j)/f_j$ is positive, then open the facility that maximizes this ratio.

Finally, we describe the plane partitioning heuristic, which is *not* guaranteed to produce a solution within any constant factor. Figure 9.1 provides a visual reference.

Trivial Heuristic

(a) We partition the square into an $m \times m$ grid Γ of subsquares $S_{p,q}$, $1 \leq p, q \leq m$ of side length α, and then open the facility $F_{p,q}$ closest to the center of each subsquare, assuming that there is one within distance $\alpha/\omega = \left(\frac{\log n}{n}\right)^{1/2}$ of its center.

(b) If any subsquare $S_{p,q}$ has no facility within distance α/ω of its center, then open each X_i in $S_{p,q}$ as a facility.

The Trivial Heuristic pays little attention to the structure of the instance, but, as we will prove in Section 9.3, it produces a solution which is asymptotically optimal **whp**. In fact, in some sense, it is *because* it does not pay attention to the instance that it out-performs the approximation algorithms. All of the greedy algorithms are distracted by local deviations in city density, and (at least at first) they will open facilities at what amount to random points in the plane. This results in non-uniform coverage and requires some unlucky cities to suffer excessive connection costs.

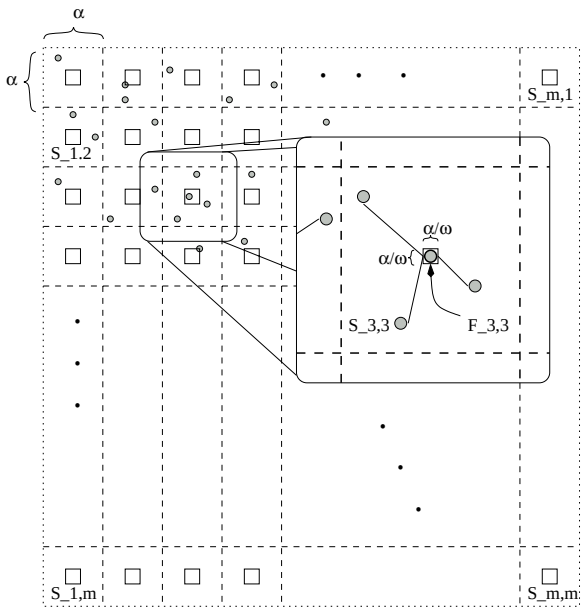

Figure 9.1: A schematic representation of the asymptotically optimal solution.

9.3 An asymptotically optimal solution

In this section, we prove that the solution found by the Trivial Heuristic is asymptotically optimal. To do so, we obtain an upper bound on the cost of this solution and a matching lower bound on the dual of the LP-relaxation.

Let HEU denote the total cost of the solution found by the Trivial Heuristic.

An intuition which explains the near optimality of this solution is that the cities and facilities are roughly uniformly distributed in the square, so the advantage of using the special structure of the instance is negligible.

To make this intuition rigorous, in the following 2 subsections, we obtain an upper bound on HEU which holds **whp**, and a lower bound on OPT which also holds **whp** and asymptotically matches the upper bound on HEU. But first we state and prove 2 lemmas that will aid in our analysis.

9.3.1 Some simple lemmas

The following 2 lemmas will help us in analyzing the heuristic and the dual lower bound.

Lemma 10 *Let $A_1, \ldots, A_k$ be subsets of $[0,1]^2$ each of area a, let $\mathcal{X}$ be a set of n random points distributed uniformly and independently in $[0,1]^2$, and let λ be a positive real with*

$\lambda \le 1/3$. *Then*

$$\mathbb{P}[\exists i\colon A_i \cap \mathcal{X} = \emptyset] \le k \cdot e^{-an} \tag{9.1}$$

$$\mathbb{P}[\exists i\colon |A_i \cap \mathcal{X}| \notin (1 \pm \lambda)an] \le k \cdot 2e^{-\lambda^2 an/3} \tag{9.2}$$

Proof (9.1) follows because the probability that a single point avoids A_i is $1 - a$ and $1 - x \le e^{-x}$ and the union bound.

(9.2) follows from Chernoff's bound and the union bound. □

Lemma 11 *Let t be a positive real, let $F_1, \ldots, F_k$ be points in $[t, 1-t]^2$, let $\mathcal{X}$ be a set of n random points distributed uniformly and independently in $[0,1]^2$, and let λ be a positive real with $\lambda \le 1/6$. For $i = 1, \ldots, k$, let $Z_i = \sum_{\substack{X \in \mathcal{X} \\ d(X,F_i) \le t}} d(X, F_i)$. Then*

$$\mathbb{E}[Z_i] = \frac{n(2t)^3}{3} \tag{9.3}$$

$$\mathbb{P}\left[\exists i\colon Z_i \notin (1 \pm \lambda)\frac{n(2t)^3}{3}\right] \le k \cdot 4e^{-\lambda^2 (2t)^2 n/12}. \tag{9.4}$$

Proof We begin by considering the contribution of a particular point X to Z_i. Conditioning on $d(X, F_i) \le t$, the expected distance is

$$\mathbb{E}[d(X, F_i) \mid d(X, F_i) \le t] = t^{-2} \int_{u=0}^{t} u \cdot 2u \, du = \frac{2t}{3}.$$

We define N_i to be the number of points within distance t of F_i,

$$N_i = |\{X \in \mathcal{X}\colon d(X, F_i) \le t\}|.$$

It follows from the linearity of expectations that

$$\mathbb{E}[Z_i \mid N_i] = N_i \frac{2t}{3}.$$

And, since $\mathbb{E}[N_i] = (2t)^2 n$, we have established (9.3),

$$\mathbb{E}[Z_i] = \left((2t)^2 n\right) \frac{2t}{3}.$$

Conditioning on N_i, Z_i is a sum of N_i independent random variables in the range $[0, t]$. So Hoeffding's inequality gives

$$\begin{aligned} \mathbb{P}\left[Z_i \notin (1 \pm \lambda) N_i \frac{2t}{3} \;\middle|\; N_i\right] &\le 2e^{-2(\lambda N_i 2t/3)^2/(N_i t^2)} \\ &= 2e^{-8\lambda^2 N_i/9}. \end{aligned}$$

Now, we apply Lemma 10 with $A_i = \{X\colon d(X, F_i) \le t\}$ and (9.2) shows that the probability that some N_i does not contain $(1 \pm \lambda)(2t)^2 n$ points is at most $k \cdot 2e^{-\lambda^2(2t)^2 n/3}$. Combining this with the conditional upper bound on the large deviation probability of Z_i and the union bound gives

$$\mathbb{P}\left[\exists i\colon Z_i \notin (1 \pm \lambda)\left((1 \pm \lambda)(2t)^2 n\right)\frac{2t}{3}\right] \le k \cdot 2e^{-\lambda^2(2t)^2 n/3} + k \cdot 2e^{-8\lambda^2(1-\lambda)(2t)^2 n/9}.$$

Since $\lambda \le 1/3$, this simplifies to

$$\mathbb{P}\left[\exists i\colon Z_i \notin (1 \pm \lambda)(2t)^3 n/3\right] \le 4ke^{-\lambda^2(2t)^2 n/3}.$$

□

9.3.2 An upper bound on HEU

To achieve this goal, we define several events and random variables and bound probabilities related to them.

Let $\hat{F}_{p,q}$ be the point in the center of subsquare $S_{p,q}$.

We begin by showing that in each subsquare, there is likely to be a facility within distance α/ω of $\hat{F}_{p,q}$ that we will open. To do this, we apply Lemma 10 with $k = m^2$ and A_{pm+q} equal to the square within distance α/ω of $\hat{F}_{p,q}$. Then, since $\text{area}(A_{pm+q}) = (2\alpha/\omega)^2 = \frac{4\log n}{n}$, (9.1) shows that

$$\mathbb{P}[\exists p, q :\ A_{pm+q} \cap \mathcal{X} = \emptyset] \le m^2 \cdot e^{-4\log n} = o(n^{-3}). \tag{9.5}$$

Now we bound the transportation costs. We define a mapping π so that for each X_i with $X_i \in S_{p,q}$ and $F_{p,q} = X_j$ we have $\pi(i) = j$ to indicate that facility j services city i. In the unlikely event that A_{pm+q} is empty, we open all the facilities in $S_{p,q}$ and set $\pi(i) = i$ for each of them, which results in transportation cost 0.

Note that, since $F_{p,q}$ is within α/ω of $\hat{F}_{p,q}$, we have

$$\sum_{X_i \in \mathcal{X}} d(X_i, X_{\pi(i)}) \le \sum_{X_i \in \mathcal{X}} d(X_i, F_{p,q}) + n\alpha/\omega. \tag{9.6}$$

We apply Lemma 11 with $t = \alpha/2$, $k = m^2$, $F_{pm+q} = \hat{F}_{p,q}$, and $\lambda = \omega^{-1}$. Then (9.3) and (9.6) together imply that

$$\mathbb{E}\left[\sum_{X_i \in \mathcal{X}} d(X_i, X_\pi(i))\right] \le m^2 \frac{n\alpha^3}{3} + n\alpha/\omega$$

and (9.4) and (9.6) imply that

$$\begin{aligned}\mathbb{P}\Big[\sum_{X_i\in\mathcal{X}} d(X_i, X_\pi(i)) \geq m^2\cdot(1+4\omega^{-1})\frac{n\alpha^3}{3} + \frac{n\alpha}{\omega}\Big] \\ \leq m^2\cdot 4e^{-16\omega^{-2}\alpha^2 n/12} \\ = 4m^2 e^{-4\log n/3}.\end{aligned}$$

Since there are m^2 facilities opened with probability at least $1-n^{-3}$, and there are at most n facilities opened in even the most pathological point set, we may the bound expected total cost of the solution by

$$\mathbb{E}[HEU] = \frac{n\alpha}{3} + n\alpha/\omega + m^2 f + nfn^{-3} = \frac{1}{2}\alpha n(1+o(1)).$$

Finally, we observe that the probability that HEU exceeds this bound tends to 0; the transportation cost is at most $\frac{n\alpha}{3}(1+O(\omega^{-1}))$ with probability $1-o(1)$ and the probability that more than m^2 facilities open is $o(1)$. So we conclude that

$$HEU \leq \frac{n\alpha}{2}(1+o(1)) \qquad \textbf{whp}. \tag{9.7}$$

9.3.3 Lower bound on OPT

To show this solution is asymptotically optimal, we will construct a solution to the dual of the strong LP relaxation:

(**LP-RELAX**)

$$\begin{aligned}\min \quad & \sum_{j=1}^n f y_j + \sum_{i=1}^n\sum_{j=1}^n d(X_i,X_j)x_{i,j} \\ \text{subj. to} \quad & \sum_{j=1}^n x_{i,j} = 1 \qquad 1\leq i\leq n \\ & 0 \leq x_{i,j}\leq y_j \quad 1\leq i,j\leq n.\end{aligned}$$

(**DUAL**)

$$\begin{aligned}\max \quad & \sum_{i=1}^n u_i \\ \text{subj. to} \quad & \sum_{i=1}^n v_{i,j} \leq f \qquad 1\leq j\leq n \\ & -v_{i,j}+u_i \leq d(X_i,X_j) \quad 1\leq i,j\leq n \\ & v_{i,j} \geq 0 \qquad 1\leq i,j\leq n.\end{aligned}$$

We get a good solution to **DUAL** as follows:

$$u_i = \begin{cases} \frac{\alpha}{2}(1-3\omega^{-1}) & X_i \in [\alpha, 1-\alpha]^2. \\ 0 & \text{otherwise.} \end{cases}$$
$$v_{i,j} = \max\{u_i - d(X_i, X_j), 0\}.$$

The fact that this solution is feasible **whp** follows from Lemma 10 and Lemma 11. We take $t = \frac{\alpha}{2}(1-3\omega^{-1})$, $k = n$, $F_i = X_i$, and $\lambda = 4\omega^{-1}$. Then (9.4) shows that

$$\begin{aligned} \mathbb{P}\left[\exists i\colon Z_i \le (1-4\omega^{-1})n(\alpha(1-3\omega^{-1}))^3/3\right] & \\ &\le n \cdot 4e^{-16\omega^{-2}(\alpha(1-3\omega^{-1}))^2 n/12} \\ &= 4ne^{-16(1-3\omega^{-1})^2 \log n/12} \\ &= o(1). \end{aligned}$$

Taking A_i to be the $\alpha\left(1-3\omega^{-1}\right) \times \alpha\left(1-3\omega^{-1}\right)$ square centered at X_i, (9.2) shows that

$$\begin{aligned} \mathbb{P}[\exists i\colon |A_i \cap \mathcal{X}| \ge (1+4\omega^{-1})(1-3\omega^{-1})^2\alpha^2 n] & \\ &\le n \cdot 2e^{-16\omega^{-2}\left(1-3\omega^{-1}\right)^2\alpha^2 n/3} \\ &= 2ne^{-16\left(1-3\omega^{-1}\right)^2 \log n/3} \\ &= o(1). \end{aligned}$$

So **whp** for all j we have

$$\begin{aligned} \sum_{i=1}^{n} v_{i,j} &= \sum_{X_i \in \mathcal{X}} \max\left\{\frac{\alpha}{2}\left(1-3\omega^{-1}\right) - d(X_i, X_j), 0\right\} \\ &< \frac{n\alpha^3}{6} = f. \end{aligned}$$

Since the objective value of this solution asymptotically matches that of (9.7), we conclude that our "heuristic" is asymptotically optimal.

9.4 Proof of Main Theorem

To prove Theorem 13, in Section 9.4.1 we state and prove some lemmas which show that the structure of *any* near optimal solution must take a certain form. In particular, the solution must choose facilities to open so that, for most open facilities, the region of the plane which is closer to that facility than any other (the Voronoi cell) is approximately a square of a certain size and is approximately centered on the facility. Lemma 14 from Section 9.4.1 gives a quantitative version of this fact: it says roughly that if there are ϵn facilities opened which violate the conditions then the solution will be a $1+\delta$ factor away from optimal.

To complete the proof of Theorem 13, in Section 9.4.2 we show that the approximation

algorithms from Section 9.2 open too many facilities which do not meet the requirements given in Lemma 14 for a close to optimal solution **whp**.

9.4.1 Properties of close-to-optimal solutions

Refining Γ to super-grid Γ_1

Now let $m_1 = \lfloor \omega^{1/2} \rfloor m$ and let Γ_1 be the $m_1 \times m_1$ super-grid of Γ where each subsquare has side $\alpha_1 = m_1^{-1}$. If we fix a subsquare S of Γ_1 then the number of points ν_S of $\mathcal{X}$ which fall in S is distributed as $\mathrm{Bi}(n, \alpha_1^2)$. Thus $\mathbb{E}(\nu_S) = \alpha_1^2 n = \omega \log n(1 + o(1))$. It follows from Lemma 10, part (2) that

$$\begin{aligned}\mathbb{P}\left[\exists S \in \Gamma_1 :\ \nu_S \notin (1 \pm \omega^{-1/3})\alpha_1^2 n\right] &\\ &\leq m_1^2 \cdot 2e^{-\omega^{-2/3}\alpha_1^2 n/3}\\ &< n \cdot 2e^{-\omega^{1/3}\log n/3}\end{aligned}$$

We use the term quite surely (**qs**) to describe a sequence of events which occurs with probability exceeding $1 - O(n^{-k})$ for any constant k. In this notation, we may say that

$$|\nu_S - \alpha_1^2 n| \leq \omega^{2/3} \log n,\ \forall S \in \Gamma_1, \qquad \textbf{qs}. \tag{9.8}$$

An assignment which respects super-grid Γ_1

For a set of facilities $\mathcal{F}$ and an assignment of cities to facilities $\phi\colon \mathcal{X} \to \mathcal{F}$ we let

$$\kappa(\mathcal{F}, \phi) = f|\mathcal{F}| + \sum_{X \in \mathcal{X}} d(X, \phi(X)).$$

The assignment which maps points to their closest facility in $\mathcal{F}$ will be denoted $\phi^\star_{\mathcal{F}}$ so that

$$c(\mathcal{F}) = \kappa(\mathcal{F}, \phi^\star_{\mathcal{F}}).$$

Consider a particular facility set $\mathcal{F} = \{F_1, F_2, \ldots, F_k\} \subseteq \mathcal{X}$. For each F_i let V_i be the Voronoi cell associated with F_i, which is to say V_i is the set of points in $[0,1]^2$ which are at least as close (in ℓ_∞ norm) to F_i as to any other member of $\mathcal{F}$.

We say an assignment ϕ respects Γ_1 if all the cities in a common subsquare of Γ_1 are assigned to the same facility by ϕ.

The next lemma says that there is an assignment which respects Γ_1 and is not much worse than $\phi^\star_{\mathcal{F}}$.

Lemma 12 *There exists an assignment $\tilde{\phi}_{\mathcal{F}}$ that respects Γ_1 and has*

$$|\kappa(\mathcal{F}, \tilde{\phi}_{\mathcal{F}})| - \kappa(\mathcal{F}, \phi^\star_{\mathcal{F}})| \leq 2\alpha_1 n.$$

Proof The proof of the lemma is a shifting argument. For any assignment ϕ, if there exists some $S \in \Gamma_1$ and $i \in [k]$ such that $V_i \cap S \cap \mathcal{X} \neq \emptyset$ and $S \setminus V_i \neq \emptyset$ then we make a slightly

different assignment $\tilde{\phi}$ which assigns all cities in S to the same facility. Let i be the smallest index in $[k]$ such that cities in S are assigned to F_i. Then we re-assign all $X_j \in S \setminus V_i$ to facility F_i. We claim that this adds at most $2\alpha_1$ in transportation cost for each city. Indeed, suppose that $X_j \in S \cap V_{i'}$ for $i' \neq i$. Then $d(X_j, F_i) \leq d(X_j, X) + d(X, F_i)$. If $X \in V_i \cap S$, then we also have that $d(X, F_i) \leq d(X, X_j) + d(X_j, F_{i'})$, since X is in V_i and not V_j. So $d(X_j, F_i) \leq d(X_j, F_{i'}) + 2d(X, X_j)$. Since X and X_j are both in S, $d(X, X_j) \leq \alpha_1$.

By starting with $\phi^\star_{\mathcal{F}}$ and repeating this shifting we eventually arrive with an assignment $\tilde{\phi}_{\mathcal{F}}$ (since assignments to cities in each cell are adjusted at most once). This assignment respects Γ_1 by construction, and (again because each city is reassigned at most once) we have

$$\kappa(\mathcal{F}, \tilde{\phi}_{\mathcal{F}}) \leq \kappa(\mathcal{F}, \phi^\star_{\mathcal{F}}) + 2\alpha_1 n. \tag{9.9}$$

□

The likely cost per facility under $\tilde{\phi}_{\mathcal{F}}$

For $F_i \in \mathcal{F}$, let the $\tilde{V}_i$ be the union of the subsquares in Γ_1 which contain cities which are mapped to F_i by $\tilde{\phi}_{\mathcal{F}}$ (we think of $\tilde{V}_i$ as the "quantized Voronoi cell" of F_i). Let η_i denote the number of subsquares in $\tilde{V}_i$. Let $\mathcal{X}_i = \mathcal{X} \cap \tilde{V}_i$ and let

$$c_i = \sum_{X \in \mathcal{X}_i} d(X, F_i).$$

Note that, because of the way $\tilde{\phi}_{\mathcal{F}}$ was constructed, for any Γ_1-subsquare S, if $S \subseteq V_i$ then $S \subseteq \tilde{V}_i$.

We say that $\tilde{V}_i$ is an *ϵ-quasi-square* if there exists a square S centered at F_i such that $\max\{\text{area}(S \setminus \tilde{V}_i), \text{area}(\tilde{V}_i \setminus S)\} \leq \epsilon \,\text{area}(\tilde{V}_i)$.

Lemma 13 *Assume that* (9.8) *holds. Assume that $\epsilon \gg \alpha_1$. Then* **whp** *the following hold for all i*

(i) $c_i \geq \frac{1}{3} n(1 - \omega^{-1/3}) \,\text{area}(\tilde{V}_i)^{3/2}$.

(ii) *If $\tilde{V}_i$ is not an ϵ-quasi-square then* $c_i \geq \frac{1+\epsilon^2/4}{3} n(1 - \omega^{-1/3}) \,\text{area}(\tilde{V}_i)^{3/2}$.

Proof In light of (9.8), this lemma reduces to a pair of geometric facts about collections of squares. However, it is convenient for us to prove the facts via linear programming.

We begin by establishing part (i) of the lemma. Fix i. For every j define $U_j = \{S \in \Gamma_1 : S \subseteq \tilde{V}_i \text{ and } j\alpha_1 \leq d(S, F_i) < (j+1)\alpha_1\}$. We have $|U_j| \leq 8j + 4$. Let k be such that $U_j = \emptyset$ for every $j > k$. Such k exists because $\tilde{V}_i$ is compact. By counting the number of Γ_1-squares in $\tilde{V}_i$ we get

$$\sum_{j=0}^{k} |U_j| = \eta_i = \text{area}(\tilde{V}_i)/\alpha_1^2.$$

Now,

$$\begin{aligned}
c_i &= \sum_{X\in\mathcal{X}_i} d(X,F_i)\\
&= \sum_{S\in\Gamma_1:S\subseteq\tilde{V}_i}\ \sum_{X\in S} d(X,F_i)\\
&\geq \sum_{S\subseteq\tilde{V}_i} \nu_S d(F_i,S)\\
&\geq (\alpha_1^2 n-\omega^{2/3}\log n)\sum_{S\subseteq\tilde{V}_i} d(F_i,S)\\
&= (\alpha_1^2 n-\omega^{2/3}\log n)\sum_{j=0}^{k}\ \sum_{S\in U_j} d(F_i,S)\\
&\geq (\alpha_1^2 n-\omega^{2/3}\log n)\alpha_1\sum_{j=0}^{k} j|U_j|
\end{aligned}$$

As we want a lower bound for c_i we consider the primal-dual pair

(**P.i**)

$$\begin{array}{lrcll}
\min & \sum_{j=0}^{k} jx_j & & & \\
\text{subj to} & x_j & \leq & 8j+4 & j=0,1,\ldots,k\\
& \sum_{j=0}^{k} x_j & = & \eta_i & \\
& x_j & \geq & 0 & j=0,1,\ldots,k
\end{array}$$

(**D.i**)

$$\begin{array}{lrcll}
\max & \eta_i z-\sum_{j=0}^{k}(8j+4)y_j & & & \\
\text{subj. to} & z-y_j & \leq & j & j=0,1,\ldots,k\\
& y_j & \geq & 0 & j=0,1,\ldots,k
\end{array}$$

A feasible solution for **D.i** is to take $z=\eta_i^{1/2}/2$ and $y_j=\max(\eta_i^{1/2}/2-j,0)$, $j=0,\ldots,k$ with dual value $\geq \eta_i^{3/2}/3$, and then $\sum_{j=0}^{k} j|U_j|\geq \eta_i^{3/2}/3=\text{area}(V_i)^{3/2}/3\alpha_1^3$.

(The expression $\sum_{j=0}^{\ell}(8j+4)(A-j)=4A(\ell+1)^2-\left(\frac{8}{3}\ell^3+6\ell^2+\frac{4}{3}\ell\right)$ will no doubt help the reader to verify the above claim.)

Now we show that part (ii) of the lemma holds. We introduce extra constraints in the linear program above in order to enforce the condition that $\tilde{V}_i$ is not an ϵ-quasi-square. For this, assume that $\tilde{V}_i$ is not an ϵ-quasi-square, let $\ell=\lfloor\eta_i^{1/2}/2\rfloor$ and let S be the square of side $2\ell\alpha_1$ centered a F_i. Then $\text{area}(\tilde{V}_i)\geq\text{area}(S)\geq(1-\epsilon)\,\text{area}(\tilde{V}_i)$ and therefore $\text{area}(S\cap\tilde{V}_i)<(1-\epsilon)\,\text{area}(\tilde{V}_i)$, otherwise $\text{area}(\tilde{V}_i\setminus S)=\text{area}(\tilde{V}_i)-\text{area}(\tilde{V}_i\cap S)\leq\epsilon\,\text{area}(\tilde{V}_i)$

and $\text{area}(S \setminus \tilde{V}_i) = \text{area}(S) - \text{area}(S \cap \tilde{V}_i) \leq \epsilon\,\text{area}(\tilde{V}_i)$. Then $\sum_{j=0}^{\ell} |U_j| = \text{area}(S \cap \tilde{V}_1)/\alpha_1^2 \leq (1-\epsilon)\,\text{area}(\tilde{V}_i)/\alpha_1^2 = (1-\epsilon)\eta_i$, so we consider the primal-dual pair

(**P.ii**)

$$\begin{array}{lrcll} \min & \sum_{j=0}^{k} j x_j & & & \\ \text{subj to} & x_j & \leq & 8j+4 & j = 0, 1, \ldots, k \\ & \sum_{j=0}^{k} x_j & = & \eta_i & \\ & \sum_{j=0}^{\ell} x_j & \leq & (1-\epsilon)\eta_i & \\ & x_j & \geq & 0 & j = 0, 1, \ldots, k \end{array}$$

(**D.ii**)

$$\begin{array}{lrcll} \max & \eta_i z - (1-\epsilon)\eta_i z_1 - \sum_{j=0}^{k} (8j+4) y_j & & & \\ \text{subj to} & z - z_1 - y_j & \leq & j & j = 0, 1, \ldots, \ell \\ & z - y_j & \leq & j & j = \ell+1, \ldots, k \\ & z_1 & \geq & 0 & \\ & y_j & \geq & 0 & j = 0, 1, \ldots, k \end{array}$$

A feasible solution for **D.ii** is $z = (1+\epsilon)\eta_i^{1/2}/2$, $z_1 = \epsilon\eta_i^{1/2}/2$, $y_j = (1-\epsilon/2)\eta_i^{1/2}/2 - j$, $j = 0, \ldots, \ell$ and $y_j = \max((1+\epsilon/2)\eta_i^{1/2}/2 - j, 0)$, $j = \ell+1, \ldots, k$ with dual value $\geq (1+\epsilon^2/4)\eta_i^{3/2}/3$, and then $\sum_{j=0}^{k} j|U_j| \geq (1+\epsilon^2/4)\eta_i^{3/2}/3 \geq (1+\epsilon^2/4)\,\text{area}(V_i)^{3/2}/3\alpha_1^3$. □

The structure of any near optimal solution

We continue by proving a property of any near optimal solution to the UFLP.

Lemma 14 *Assume that* (9.8) *holds. Let ϵ be a sufficiently small constant, and let $\mathcal{F} \subseteq \mathcal{X}$ with $\kappa(\mathcal{F}, \phi_{\mathcal{F}}^{\star}) \leq (1+\epsilon)\alpha n/2$. Then for $\epsilon_1 = 5\epsilon^{1/2}$,*

(a) $|\mathcal{F}| \in [(1-\epsilon_1)m^2, (1+\epsilon_1)m^2]$.

(b) *Suppose that $\theta_1 = 2\epsilon^{1/3}$ and $\theta_2 = 4\epsilon^{1/3}$ and $\epsilon_0 = 3\epsilon^{1/3}$. Then at least $(1-2\theta_2)m^2$ of the points $F_i \in \mathcal{F}$ are such that $\tilde{V}_i$ is an ϵ_0-quasi-square of area in the range $[(1-\theta_1)\alpha^2, (1+\theta_1)\alpha^2]$.*

Proof Let $\mathcal{F} = \{F_1, F_2, \ldots, F_k\}$ and let $a_i = |\tilde{V}_i|$ for $1 \leq i \leq k$. Let $J = \{j : \tilde{V}_j \text{ is not a } \epsilon_0\text{-quasi-square}\}$. Applying Lemma 13 and equation (9.9) we see that

$$\kappa(\mathcal{F}, \phi_{\mathcal{F}}^{\star}) \geq kf + \frac{1-\omega^{-1/3}}{3} n \left(\sum_{i=1}^{k} a_i^{3/2} + \frac{\epsilon_0^2}{12} \sum_{j \in J} a_j^{3/2} \right) - 2\alpha_1 n. \tag{9.10}$$

Now let $a_j = \frac{1+x_j}{k}$, where $-1 \leq x_j$ and $\sum_{j=1}^{k} x_j = 0$.

By examining the power series for $(1+x)^{3/2}$ when $|x| \leq 1$ and using elementary calculus for $x > 1$ we see that

$$(1+x)^{3/2} \geq 1 + \frac{3}{2}x + \min\left\{1, \frac{1}{4}x^2\right\} \qquad x \geq -1. \tag{9.11}$$

It follows from (9.10) that

$$\kappa(\mathcal{F}, \phi^\star_{\mathcal{F}}) \geq$$
$$kf + \left(\frac{1-\omega^{-1/3}}{3}n\right) \times \left(k^{-1/2} + k^{-3/2}\sum_{i=1}^{k} \min\left\{1, \frac{1}{4}x_i^2\right\} + \frac{\epsilon_0^2}{12}\sum_{j\in J} a_j^{3/2}\right) - 2\alpha_1 n \tag{9.12}$$

Now, let $k = (1+\theta)\alpha^{-2}$ for some $\theta \geq -1$ and assume wlog that $|\theta| \gg \omega^{-1/6}$. Notice that from (9.10) that we can assume $\theta < 3$, otherwise $kf \geq \frac{4}{6}\alpha n$. If $\theta \in [-1, 3]$ then $\frac{1}{(1+\theta)^{1/2}} \geq 1 - \frac{1}{2}\theta + \frac{1}{16}\theta^2$, and we get

$$kf + \frac{1-\omega^{-1/3}}{3}nk^{-1/2} - 2\alpha_1 n \geq$$
$$\frac{(1+\theta)}{6}\alpha n + \frac{(1-\omega^{-1/3})}{3}\alpha n\left(1 - \frac{1}{2}\theta + \frac{1}{16}\theta^2\right) - 2\alpha_1 n \geq$$
$$\frac{\alpha n}{2}\left(1 + \frac{\theta^2}{25}\right). \tag{9.13}$$

And using (9.12) we get

$$\kappa(\mathcal{F}, \phi^\star_{\mathcal{F}}) \geq \frac{\alpha n}{2}\left(1 + \frac{\theta^2}{25}\right) + \frac{n}{4}\left(k^{-3/2}\sum_{i=1}^{k} \min\left\{1, \frac{1}{4}x_i^2\right\} + \frac{\epsilon_0^2}{12}\sum_{j\in J} a_j^{3/2}\right). \tag{9.14}$$

Part (a) follows from (9.14): $(1+\epsilon)\alpha n/2 \geq \kappa(\mathcal{F}, \phi^\star_{\mathcal{F}}) \geq \frac{\alpha n}{2}\left(1 + \frac{\theta^2}{25}\right)$ and so $|\theta| \leq \epsilon^{1/2}/5$.

Using (9.14) again we get

$$\kappa(\mathcal{F}, \phi^\star_{\mathcal{F}}) \geq \frac{1}{2}\alpha n + \frac{n}{4k^{3/2}}\sum_{j=1}^{k} \min\left\{1, \frac{1}{4}x_j^2\right\},$$

So if $B = \{j : |x_j| \geq \theta_1\}$ and $|B| \geq \beta k$ for $\theta_1, \beta \leq 1$, we have $\kappa(\mathcal{F}, \phi^\star_{\mathcal{F}}) \geq \frac{1}{2}\alpha n + \frac{\theta_1^2 \beta}{16(1+\epsilon_1)^{1/2}}\alpha n$. Setting $\theta_1 = 2\epsilon^{1/3}$ we get $\beta \leq \theta_2 = 4\epsilon^{1/3}$. Returning once again to (9.14) we write

$$\begin{aligned}\kappa(\mathcal{F}, \phi^\star_{\mathcal{F}}) &\geq \frac{1}{2}\alpha n + \frac{\epsilon_0^2}{48} n \sum_{j \in J} a_j^{3/2} \\ &\geq \frac{1}{2}\alpha n + \frac{3\epsilon^{2/3}}{16} n(|J| - \theta_2 k) \left(\frac{1-\theta_1}{k}\right)^{3/2}.\end{aligned}$$

Thus, if $|J| \geq 2\theta_2 m^2$ then

$$\begin{aligned}\kappa(\mathcal{F}, \phi^\star_{\mathcal{F}}) &\geq \frac{1}{2}\alpha n + \frac{3\epsilon^{2/3}}{16} n\theta_2(2m^2 - k) \left(\frac{1-\theta_1}{k}\right)^{3/2} \\ &\geq \frac{1}{2}\alpha n + \frac{12}{16}\epsilon n(1-\epsilon_1)m^2 \left(\frac{1-2\epsilon^{1/3}}{(1+\epsilon_1)m^2}\right)^{3/2} \\ &\geq \frac{1}{2}\alpha n + \frac{11}{16}\epsilon\alpha n.\end{aligned}$$

□

9.4.2 Properties of Solutions Found by Greedy Approximation Algorithms

The goal of this section is to use the characterization of close-to-optimal solutions obtained in Section 9.4.1 to show that the greedy approximation algorithms described in Section 9.2 find solutions which are not asymptotically optimal. This is achieved by considering the behavior of the algorithm on a $14\alpha \times 7\alpha$ rectangular subregion of the unit square, and showing that, with constant probability, this region contains a facility for which the Vornoi region is not ϵ-quasi-square.

The intuition which motivates this approach is this: the candidate facilities which open in the subregion will do so at random locations, thus there is no reason to expect these random locations to result in nice Vornoi cells. Making this intuitive explaination rigorous requires some work because of the complicated dependencies between which facilities are opened. For example, all the approximation algorithms track a level of "funding" available for opening a candidate facility (in Approximation Algorithm 1, the funds for city j are $\sum_{i \in U} \max\{0, \delta_i - d(i,j)\}$, and in Approximation Algorithm 2 and 3, the funds are at least this much.) The funds available to a facility at time t is a difficult random variable to deal with, and we must work around this difficulty.

Let $pf(X,t)$ denote the *potential funds* at point X at time t, given by

$$pf(X,t) = \sum_{i \in C} \max\{0, t - d(X_i, X)\}.$$

This is the level of funding available to open a facility at X *if* no other facilities have already

opened within distance $2t$ of X.

Let $T(X) = \min\{\min\{t : pf(X,t) = f\}, \alpha\}$ be the *earliest opening time* of point X (which is truncated at time α, because we want $T(X)$ to only depend on the position of nearby points).

We note that $\mathbb{E}(pf(X,\alpha)) = f$ and $pf(X,\alpha)$ is the sum of n independent bounded random variables and so the Central Limit Theorem implies that

$$\mathbb{P}(pf(X,\alpha) \geq f) = \frac{1}{2} - o(1). \tag{9.15}$$

Consider concentric squares, $S_1, S_2, \ldots$, where S_i is an $i\alpha \times i\alpha$ square (see Figure 9.2 for visual reference). Some facility $X^\star$ in S_5 has the minimum value of $T(X)$ among all facilities in S_5, and which one it is only depends on the configuration of points in S_7.

Note that if $X^\star$ is in S_1, (and $T(X^\star) < \alpha$) then (in all 3 of the greedy approximation algorithms) $X^\star$ *actually* opens at time $T(X^\star)$, because no cities within distance α of S_1 are connected (because no facilities within 2α of S_1 are open; in other words, no facilities besides $X^\star$ are open in S_5.) Since nothing within α of $X^\star$ is connected,

$$\sum_{i \in U} \max\{0, \delta_i - d(X_i, X^\star)\} = \sum_{i \in C} \max\{0, T(X^\star) - d(X_i, X^\star)\}.$$

We will partition S_7 into subsquares of size $\alpha/4$, and obtain a constant lower bound on the probability $X^\star$ appears in one of these subsquares that is contained in S_1. For $(p,q) \in [4]^2$, let $Q_{p,q}$, denote such a subsquare (with side length $\alpha/4$ that is contained in S_1). Figure 9.2 provides a visual reference.

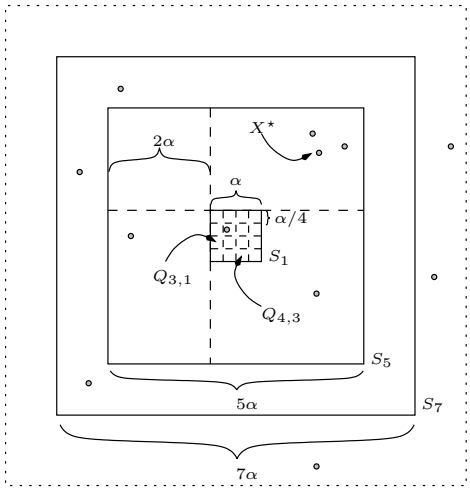

Figure 9.2: Concentric squares S_1, S_5, and S_7.

Lemma 15 *Let $X^\star$ be the facility in S_5 which minimizes $T(X)$ over $X \in \mathcal{X} \cap S_5$. There*

exists an absolute constant γ_0 such that for any $(p,q) \in [4]^2$,

$$\mathbb{P}[X^\star \in Q_{p,q} \text{ and } pf(X^\star, \alpha) \geq f] \geq \gamma_0.$$

Proof We write $\mathcal{A}^\star_{p,q} = \{X^\star \in Q_{p,q}\}$ and $\mathcal{B}^\star = \{pf(X^\star, \alpha) \geq f\}$. We then consider the analogous question for $\mathcal{X}' = \mathcal{X} \cap S_7$, where the edges of S_7 have been identified to "wrap-around", making the distance function

$$d_W((x_1,y_1),(x_2,y_2)) = \max\{\min\{|x_1-x_2|, 7\alpha - |x_1-x_2|\}, \min\{|y_1-y_2|, 7\alpha-|y_1-y_2|\}\}.$$

(This makes the space a torus topologically.) In this case, some $\alpha/4 \times \alpha/4$ subsquare of S_7 contains the facility $X^\star$ which minimizes $T(X)$ and, by symmetry, each subsquare is equally likely to contain it. So the probability that $X^\star$ is in $Q_{p,q}$ is the same as the probability that it is in any of the $(7\cdot 4)^2$ subsquares, which is exactly $1/(7\cdot 4)^2$. Using $\mathbb{P}'$ for this model and double $\star$'s to distinguish the S_7 case from the S_5 case, we have

$$\mathbb{P}'(\mathcal{B}^{\star\star}) = \sum_{(a,b)\in[7\cdot 4]^2} \mathbb{P}'(\mathcal{B}^\star \wedge \mathcal{A}^{\star\star}_{a,b}) = (7\cdot 4)^2 \mathbb{P}'(\mathcal{B}^{\star\star} \wedge \mathcal{A}^{\star\star}_{p,q}),$$

giving

$$\mathbb{P}'(\mathcal{B}^{\star\star} \wedge \mathcal{A}^{\star\star}_{p,q}) \geq \frac{1}{3(7\cdot 4)^2}.$$

Now, we remove the wrap-around on S_7 but continue to ignore all the points of $\mathcal{X}$ that lie outside S_7 (i.e. consider the potential funds for point set $\mathcal{X}' = \mathcal{X} \cap S_7$ under the ℓ_∞ distance). This change can only affect $T(X)$ for a point X which lies within distance α of the boundary of S_7, and for such an X, the change can only makes $T(X)$ larger than it was in the case with wrap-around. So every $\mathcal{X}$ which yielded $\mathcal{B}^{\star\star}$ and $X^{\star\star}$ in $Q_{p,q}$ with the wrap-around distance will have $\mathcal{B}^\star$ and the same $X^\star$ in this case. So the probability that $\mathcal{B}^\star$ occurs and $Q_{p,q}$ contains the point which opens first when $\mathcal{X}' = \mathcal{X} \cap S_7$ is at least $(7\cdot 4)^{-2}/3$.

Finally, we return to the original set $\mathcal{X}$, and note that considering the contributions of points outside of S_7 to the potential funds does not affect $T(X)$ for any X in S_5. So the probability that $Q_{p,q}$ contains the point which opens first in S_5 with respect to $\mathcal{X}$ is at least the probability that $Q_{p,q}$ contains the point which opens first in S_7 with respect to $\mathcal{X}' = \mathcal{X} \cap S_7$. The previous paragraph showed that this is at least $(7\cdot 4)^{-2}/3$. □

Now consider 2 side-by-side copies of S_7, as shown in Figure 9.3. Let $\mathcal{B}_1$ be the event that, in the left copy of S_7, the facility X^L which minimizes $T(X)$ in S_5^L is in $Q^L_{q,1}$ for some q. Let $\mathcal{B}_2$ be the event that, in the right copy of S_7, the facility X^R which minimizes $T(X)$ in S_5^R is in $Q^R_{q',3}$ for some q'. Because $Q^L_{q,1}$ and $Q^R_{q',3}$ are sufficiently far apart, $\mathbb{P}[\mathcal{B}_1 \mid \mathcal{B}_2] = \mathbb{P}[\mathcal{B}_1]$, and so $\mathbb{P}[\mathcal{B}_1\mathcal{B}_2] \geq \gamma_0^2$.

Suppose now that $\mathcal{B}_1$ and $\mathcal{B}_2$ occur. Let Σ be the $(1+\epsilon)\alpha \times 8\alpha$ strip containing S_1^L and S_1^R (so that S_1^L, S_1^R are located symmetrically at distance $\epsilon\alpha/2$ from the horizontal borders of Σ, as depicted in Figure 9.3.) Here ϵ is some sufficiently small positive constant. Let I be the index set of those open facilities whose quantized Voronoi cells $\tilde{V}_i$ meet the strip Σ.

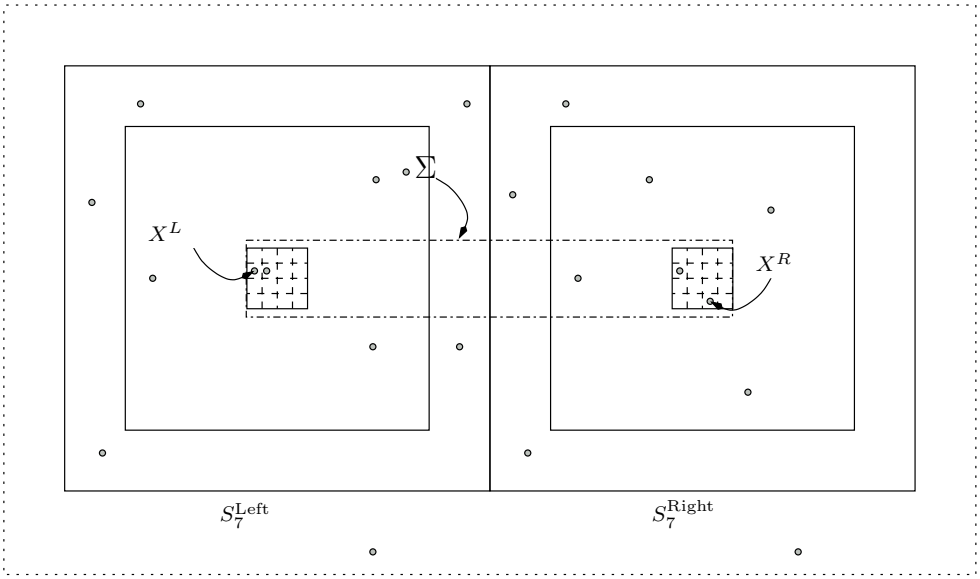

Figure 9.3: Two side-by-side copies of S_7

Lemma 16 Whp *there must be some facility $i \in I$ for which $\tilde{V}_i$ is not ϵ^3-quasi-square with area in $(1 \pm \epsilon^3)\alpha^2$.*

Proof Assume for the sake of contradiction that this is not the case. Each such Voronoi region $\tilde{V}_i$ can therefore be associated with a square W_i of side in the range $(1 \pm \epsilon^3)\alpha$. Furthermore, any two such squares have a common area of at most $\epsilon^3\alpha^2$. **Whp** there is no open facility j at distance 2α or more from Σ for which the quantized Voronoi region $\tilde{V}_j$ intersects Σ (every point in $\mathcal{X} \cap \Sigma$ is connected to a closer open facility). Thus $|I| \leq \frac{8(5+\epsilon)}{1-\epsilon/2} < 50$. It follows that all but an area of at most $50\epsilon^3\alpha^2$ of Σ is covered by the W_i, for $i \in I$. Now let Σ_1 denote a strip of length 8α and thickness $\epsilon\alpha/4$ running across the middle of Σ. Any sub-strip of Σ_1 which is of length $\epsilon\alpha$ is of area $\epsilon^2\alpha^2/4$ and so will contain members of $\mathcal{X}$ which are covered by some W_i, $i \in I$.

If the center of this W_i is outside Σ then W_i has side at least $(1+\epsilon/4)\alpha$, which contradicts our assumption. So let J be the set of facilities j with center in Σ for which there is a member of Σ_1 contained in W_j. If any of these facilities is not ϵ^3-quasi-square then we are done, so we may assume that they all are. There is an open facility in $S^L_{q,1}$ and in $S^R_{q',3}$, and these facilities cover squares of side at least α. Thus the other members of J appear in a substrip with length between 7.25α and 7.75α. If there are 6 or fewer open facilities in this the strip bounding the 2 copies, then some pair of facilities are at least 1.04α apart. Therefore, one of them, call it F_i has a W_i with side at least $(1.04 - 100\epsilon^3)\alpha$, contradiction. On the other hand, if there are 7 or more facilities in the strip, then some pair are at most $.96\alpha$ apart, and so some F_i has a W_i with side at most $(.96 + \frac{2\epsilon^3}{.48})\alpha$, contradiction. □

Since the event $\mathcal{B}_1\mathcal{B}_2$ occurs independently in sufficiently separated disjoint regions of

the square (modulo there being enough points in the cell), **whp** we will have $\Omega(m^2)$ facilities for which $\tilde{V}_i$ is not an ϵ^3-quasi-square with area $(1 \pm \epsilon^3)\alpha$. So Lemma 14 finishes the proof of the theorem. □

Bibliography

[1] Achlioptas, D. Setting 2 variables at a time yields a new lower bound for random 3-SAT (extended abstract). In *Proceedings of the Thiry-Second Annual ACM Symposium on Theory of Computing* (New York, 2000), ACM, pp. 28–37 (electronic).

[2] Achlioptas, D., Chtcherba, A. D., Istrate, G., and Moore, C. The phase transition in 1-in-k SAT and NAE 3-SAT. In *Proceedings of the 12th Annual ACM-SIAM Symposium on Discrete Algorithms* (2001), pp. 721–722.

[3] Achlioptas, D., Kim, J. H., Krivelevich, M., and Tetali, P. Two-coloring random hypergraphs. *Random Structures Algorithms 20*, 2 (2002), 249–259.

[4] Achlioptas, D., and Moore, C. On the 2-colorability of random hypergraphs. In *Randomization and approximation techniques in computer science*, vol. 2483 of *Lecture Notes in Comput. Sci.* Springer, Berlin, 2002, pp. 78–90.

[5] Achlioptas, D., and Peres, Y. The threshold for random k-SAT is $2^k \log 2 - O(k)$. *J. Amer. Math. Soc. 17*, 4 (2004), 947–973 (electronic).

[6] Achlioptas, D., and Sorkin, G. B. Optimal myopic algorithms for random 3-SAT. In *41st Annual Symposium on Foundations of Computer Science* (2000), IEEE Comput. Soc. Press, Los Alamitos, CA, pp. 590–600.

[7] Adamic, L. A., Lukose, R. M., and Huberman, B. A. Local search in unstructured networks. In *Handbook of graphs and networks*. Wiley-VCH, Weinheim, 2003, pp. 295–317.

[8] Ahn, S., Cooper, C., Cornuéjols, G., and Frieze, A. M. Probabilistic analysis of a relaxation for the k-median problem. *Math. Oper. Res. 13*, 1 (1988), 1–31.

[9] Aiello, W., Chung, F., and Lu, L. A random graph model for massive graphs. In *Proceedings of the Thiry-Second Annual ACM Symposium on Theory of Computing* (New York, 2000), ACM, pp. 171–180 (electronic).

[10] Albert, R., Jeong, H., and Barabási, A.-L. Diameter of the world wide web. *Nature 401* (1999), 103–131.

[11] Aldous, D. J. Asymptotics in the random assignment problem. *Probab. Theory Related Fields 93*, 4 (1992), 507–534.

[12] Aldous, D. J. The $\zeta(2)$ limit in the random assignment problem. *Random Structures Algorithms 18*, 4 (2001), 381–418.

[13] Aleliunas, R., Karp, R. M., Lipton, R. J., Lovász, L., and Rackoff, C. Random walks, universal traversal sequences, and the complexity of maze problems. In *20th Annual Symposium on Foundations of Computer Science (San Juan, Puerto Rico, 1979)*. IEEE, New York, 1979, pp. 218–223.

[14] Alon, N., Gyárfás, A., and Ruszinkó, M. Decreasing the diameter of bounded degree graphs. *J. Graph Theory 35*, 3 (2000), 161–172.

[15] Alon, N., and Kahale, N. A spectral technique for coloring random 3-colorable graphs. *SIAM J. Comput. 26*, 6 (1997), 1733–1748.

[16] Alon, N., Krivelevich, M., and Sudakov, B. Finding a large hidden clique in a random graph. *Random Structures Algorithms 13*, 3-4 (1998), 457–466.

[17] Alon, N., and Spencer, J. H. *The probabilistic method*, second ed. Wiley-Interscience Series in Discrete Mathematics and Optimization. Wiley-Interscience [John Wiley & Sons], New York, 2000. With an appendix on the life and work of Paul Erdős.

[18] Amit, A., Linial, N., and Matoušek, J. Random lifts of graphs: independence and chromatic number. *Random Structures Algorithms 20*, 1 (2002), 1–22.

[19] Amit, A., Linial, N., Matoušek, J., and Rozenman, E. Random lifts of graphs. In *Proceedings of the Twelfth Annual ACM-SIAM Symposium on Discrete Algorithms (Washington, DC, 2001)* (Philadelphia, PA, 2001), SIAM, pp. 883–894.

[20] Angluin, D., and Valiant, L. G. Fast probabilistic algorithms for hamiltonian circuits and matchings. In *STOC '77: Proceedings of the ninth annual ACM symposium on Theory of computing* (New York, NY, USA, 1977), ACM Press, pp. 30–41.

[21] Austin, T. L., Fagen, R. E., Penney, W. F., and Riordan, J. The number of components in random linear graphs. *Ann. Math. Statist 30* (1959), 747–754.

[22] Bacon, D., Childs, A. M., and van Dam, W. From optimal measurement to efficient quantum algorithms for the hidden subgroup problem over semidirect product groups. In *FOCS '05: Proceedings of the 46th Annual IEEE Symposium on Foundations of Computer Science* (Washington, DC, USA, 2005), IEEE Computer Society, pp. 469–478.

[23] Banderier, C., Beier, R., and Mehlhorn, K. Smoothed analysis of three combinatorial problems. In *Mathematical foundations of computer science 2003*, vol. 2747 of *Lecture Notes in Comput. Sci.* Springer, Berlin, 2003, pp. 198–207.

[24] Barabási, A.-L. *Linked: The New Science of Networks.* Cambridge, MA: Perseus Publishing, 2002.

[25] Barabási, A.-L., and Albert, R. Emergence of scaling in random networks. *Science 286*, 5439 (1999), 509–512.

[26] Barahona, F., and Chudak, F. A. Solving large scale uncapacitated facility location problems. In *Approximation and complexity in numerical optimization (Gainesville, FL, 1999)*, vol. 42 of *Nonconvex Optim. Appl.* Kluwer Acad. Publ., Dordrecht, 2000, pp. 48–62.

[27] Barbour, A. D., and Reinert, G. Small worlds. *Random Structures Algorithms 19*, 1 (2001), 54–74.

[28] Barthel, W., Hartmann, A. K., Leone, M., Ricci-Tersenghi, F., Weigt, M., and Zecchina, R. Hiding solutions in random satisfiability problems: A statistical mechanics approach. *Phys. Rev. Lett. 88* (2002), 188701.

[29] Bast, H., Mehlhorn, K., Schäfer, G., and Tamaki, H. Matching algorithms are fast in sparse random graphs. In *STACS 2004*, vol. 2996 of *Lecture Notes in Comput. Sci.* Springer, Berlin, 2004, pp. 81–92.

[30] Becchetti, L., Leonardi, S., Marchetti-Spaccamela, A., Schäfer, G., and Vredeveld, T. Average case and smoothed competitive analysis of the multi-level feedback algorithm. In *Proceedings of the 44th Annual IEEE Symposium on Foundations of Computer Science (FOCS 2003)* (Washington, DC, USA, 2003), IEEE Computer Society, p. 462.

[31] Beier, R., and Vöcking, B. Random knapsack in expected polynomial time. *J. Comput. System Sci. 69*, 3 (2004), 306–329.

[32] Beier, R., and Vöcking, B. Typical properties of winners and losers in discrete optimization. In *Proceedings of the 36th Annual ACM Symposium on Theory of Computing* (New York, 2004), ACM, pp. 343–352 (electronic).

[33] Ben-Sasson, E., and Wigderson, A. Short proofs are narrow — resolution made simple. *J. ACM 48*, 2 (2001), 149–169.

[34] Berger, N., Bollobás, B., Borgs, C., Chayes, J., and Riordan, O. Degree distribution of the FKP network model. In *Automata, languages and programming*, vol. 2719 of *Lecture Notes in Comput. Sci.* Springer, Berlin, 2003, pp. 725–738.

[35] Berger, N., Borgs, C., Chayes, J. T., D'Souza, R. M., and Kleinberg, R. D. Competition-induced preferential attachment. In *Automata, languages and programming*, vol. 3142 of *Lecture Notes in Comput. Sci.* Springer, Berlin, 2004, pp. 208–221.

[36] Berger, N., Borgs, C., Chayes, J. T., D'Souza, R. M., and Kleinberg, R. D. Degree distribution of competition-induced preferential attachment graphs. *Combin. Probab. Comput. 14*, 5-6 (2005), 697–721.

[37] Birge, J. R., and Louveaux, F. *Introduction to stochastic programming*. Springer Series in Operations Research. Springer-Verlag, New York, 1997.

[38] Blum, A., Kalai, A., and Wasserman, H. Noise-tolerant learning, the parity problem, and the statistical query model. *J. ACM 50*, 4 (2003), 506–519 (electronic).

[39] Blum, A., and Spencer, J. Coloring random and semi-random k-colorable graphs. *J. Algorithms 19*, 2 (1995), 204–234.

[40] Bohman, T., Frieze, A., Krivelevich, M., and Martin, R. Adding random edges to dense graphs. *Random Structures Algorithms 24*, 2 (2004), 105–117.

[41] Bohman, T., Frieze, A., and Martin, R. How many random edges make a dense graph Hamiltonian? *Random Structures Algorithms 22*, 1 (2003), 33–42.

[42] Bohman, T., and Frieze, A. M. The random 2-out is Hamlitonian. Manuscript in preparation.

[43] Bollobás, B. A probabilistic proof of an asymptotic formula for the number of labelled regular graphs. *European J. Combin. 1*, 4 (1980), 311–316.

[44] Bollobás, B. Martingales, isoperimetric inequalities and random graphs. In *Combinatorics*, A. Hajnal, L. Lovász, and V. T. Sós, Eds., no. 52 in Colloq. Math. Soc. János Bolyai. North Holland, 1988, pp. 113–139.

[45] Bollobás, B. *Modern graph theory*, vol. 184 of *Graduate Texts in Mathematics*. Springer-Verlag, New York, 1998.

[46] Bollobás, B. *Random graphs*, vol. 73 of *Cambridge Studies in Advanced Mathematics*. Cambridge University Press, Cambridge, 2001.

[47] Bollobás, B., Borgs, C., Chayes, J., and Riordan, O. Directed scale-free graphs. In *Proceedings of the Fourteenth Annual ACM-SIAM Symposium on Discrete Algorithms (Baltimore, MD, 2003)* (New York, 2003), ACM, pp. 132–139.

[48] Bollobás, B., Borgs, C., Chayes, J. T., Kim, J. H., and Wilson, D. B. The scaling window of the 2-SAT transition. *Random Structures Algorithms 18*, 3 (2001), 201–256.

[49] Bollobás, B., and Chung, F. R. K. The diameter of a cycle plus a random matching. *SIAM J. Discrete Math. 1*, 3 (1988), 328–333.

[50] Bollobás, B., and Frieze, A. M. On matchings and Hamiltonian cycles in random graphs. In *Random graphs '83 (Poznań, 1983)*, vol. 118 of *North-Holland Math. Stud.* North-Holland, Amsterdam, 1985, pp. 23–46.

[51] Bollobás, B., and Riordan, O. Coupling scale-free and classical random graphs. *Internet Math. 1*, 2 (2004), 215–225.

[52] Bollobás, B., and Riordan, O. The diameter of a scale-free random graph. *Combinatorica 24*, 1 (2004), 5–34.

[53] Bollobás, B., Riordan, O., Spencer, J., and Tusnády, G. The degree sequence of a scale-free random graph process. *Random Structures Algorithms 18*, 3 (2001), 279–290.

[54] Bollobás, B., and Riordan, O. M. Mathematical results on scale-free random graphs. In *Handbook of graphs and networks*. Wiley-VCH, Weinheim, 2003, pp. 1–34.

[55] Bollobás, B., and Scott, A. D. Max Cut for random graphs with a planted partition. *Combin. Probab. Comput. 13*, 4-5 (2004), 451–474.

[56] Bonato, A. A survey of models of the web graph. In *Combinatorial and Algorithmic Aspects of Networking, First Workshop on Combinatorial and Algorithmic Aspects of Networking, CAAN 2004, Banff, Alberta, Canada, August 5-7, 2004, Revised Selected Papers* (2004), pp. 159–172.

[57] Boppana, R. B. Eigenvalues and graph bisection: an average-case analysis. In *Proceedings of the 28th Annual IEEE Symposium on Foundations of Computer Science* (1987), pp. 280–285.

[58] Borgs, C., Chayes, J., and Pittel, B. Sharp threshold and scaling window for the integer partitioning problem. In *Proceedings of the Thirty-Third Annual ACM Symposium on Theory of Computing* (New York, 2001), ACM, pp. 330–336 (electronic).

[59] Borgs, C., Chayes, J. T., Mertens, S., and Pittel, B. Constrained integer partitions. In *LATIN 2004: Theoretical informatics*, vol. 2976 of *Lecture Notes in Comput. Sci.* Springer, Berlin, 2004, pp. 59–68.

[60] Boucheron, S., Lugosi, G., and Massart, P. Concentration inequalities using the entropy method. *Ann. Probab. 31*, 3 (2003), 1583–1614.

[61] Broder, A., Frieze, A., and Upfal, E. On the satisfiability and maximum satisfiability of random 3-cnf formulas. In *Proceedings of the 4th Annual ACM-SIAM Symposium on Discrete Algorithms* (1993), pp. 322–330.

[62] Broder, A., Kumar, R., Maghoul, F., Raghavan, P., Rajagopalan, S., Stata, R., Tomkins, A., and Wiener, J. Graph structure in the Web. *Comput. Networks 33*, 1-6 (2000), 309–320.

[63] Buckley, P. G., and Osthus, D. Popularity based random graph models leading to a scale-free degree sequence. *Discrete Math. 282*, 1-3 (2004), 53–68.

[64] Burgin, K., Chebolu, P., Cooper, C., and Frieze, A. M. Hamiltoniam cycles in random lifts of graphs. manuscript submitted for publication, 2006.

[65] Cannings, C., and Penman, D. B. Models of random graphs and their applications. In *Stochastic processes: modelling and simulation*, vol. 21 of *Handbook of Statist.* North-Holland, Amsterdam, 2003, pp. 51–91.

[66] Carlson, J. M., and Doyle, J. C. Highly optimized tolerance: Robustness and design in complex systems. *Physics Review Letters 84*, 11 (2000), 2529–2532.

[67] Carson, T., and Impagliazzo, R. Hill-climbing finds random planted bisections. In *Proceedings of the Twelfth Annual ACM-SIAM Symposium on Discrete Algorithms (Washington, DC, 2001)* (Philadelphia, PA, 2001), SIAM, pp. 903–909.

[68] Chaimovich, M. New algorithm for dense subset-sum problem. *Astérisque*, 258 (1999), xvi, 363–373. Structure theory of set addition.

[69] Chaimovich, M., Freiman, G., and Galil, Z. Solving dense subset-sum problems by using analytical number theory. *J. Complexity 5*, 3 (1989), 271–282.

[70] Chao, M., and Franco, J. Probabilistic analysis of 2 heuristics for the 3-satisfiability problem. *SIAM Journal of Computing 15* (1986), 1106–1118.

[71] Chao, M., and Franco, J. Probabilistic analysis of a generalization of the unit-clause literal selection heuristic for the k satisfiability problem. *Information Science* (1990), 289–314.

[72] Charikar, M., and Guha, S. Improved combinatorial algorithms for facility location problems. *SIAM J. Comput. 34*, 4 (2005), 803–824 (electronic).

[73] Cheeseman, P., Kanefsky, B., and Taylor, W. M. Where the Really Hard Problems Are. In *Proceedings of the Twelfth International Joint Conference on Artificial Intelligence, IJCAI-91, Sidney, Australia* (1991), pp. 331–337.

[74] Chen, H., and Frieze, A. M. Coloring bipartite hypergraphs. In *Proc. 5th IPCO* (1996), pp. 345–358.

[75] Chudak, F. A., and Shmoys, D. B. Improved approximation algorithms for the uncapacitated facility location problem. *SIAM J. Comput. 33*, 1 (2003), 1–25 (electronic).

[76] Chung, F., and Lu, L. Coupling online and offline analyses for random power law graphs. *Internet Math. 1*, 4 (2004), 409–461.

[77] Chung, F., and Lu, L. The small world phenomenon in hybrid power law graphs. In *Complex networks*, vol. 650 of *Lecture Notes in Phys.* Springer, Berlin, 2004, pp. 89–104.

[78] Chung, F., Lu, L., and Vu, V. The spectra of random graphs with given expected degrees. *Internet Math. 1*, 3 (2004), 257–275.

[79] Chung, F. R. K., and Garey, M. R. Diameter bounds for altered graphs. *J. Graph Theory 8*, 4 (1984), 511–534.

[80] Chvátal, V. Hard knapsack problems. *Oper. Res. 28*, 6 (1980), 1402–1411.

[81] Chvátal, V., and Reed, B. Mick gets some (the odds are on his side). In *33th Annual Symposium on Foundations of Computer Science (Pittsburgh, PA, 1992)*. IEEE Comput. Soc. Press, Los Alamitos, CA, 1992, pp. 620–627.

[82] Chvátal, V., and Szemerédi, E. Many hard examples for resolution. *J. Assoc. Comput. Mach. 35*, 4 (1988), 759–768.

[83] Coarfa, C., Demopoulos, D. D., San Miguel Aguirre, A., Subramanian, D., and Vardi, M. Y. Random 3-SAT: the plot thickens. *Constraints 8*, 3 (2003), 243–261. Special issue of the Sixth International Conference on Principles and Practice of Constraint Programming (Singapore, 2000).

[84] Coja-Oghlan, A. A spectral heuristic for bisecting random graphs. In *SODA '05: Proceedings of the sixteenth annual ACM-SIAM symposium on Discrete algorithms* (Philadelphia, PA, USA, 2005), Society for Industrial and Applied Mathematics, pp. 850–859.

[85] Coja-Oghlan, A., Goerdt, A., Lanka, A., and Schädlich, F. Techniques from combinatorial approximation algorithms yield efficient algorithms for random $2k$-SAT. *Theoret. Comput. Sci. 329*, 1-3 (2004), 1–45.

[86] Condon, A., and Karp, R. M. Algorithms for graph partitioning on the planted partition model. *Random Structures Algorithms 18*, 2 (2001), 116–140.

[87] Cook, S. The complexity of theorem-proving procedures. In *Proc. 3rd FOCS* (1971), pp. 151–158.

[88] Cooper, C. Classifying special interest groups in web graphs. In *Randomization and approximation techniques in computer science*, vol. 2483 of *Lecture Notes in Comput. Sci.* Springer, Berlin, 2002, pp. 263–275.

[89] Cooper, C., and Frieze, A. Crawling on simple models of web graphs. *Internet Math. 1*, 1 (2003), 57–90.

[90] Cooper, C., and Frieze, A. A general model of web graphs. *Random Structures Algorithms 22*, 3 (2003), 311–335.

[91] Cooper, C., and Frieze, A. The size of the largest strongly connected component of a random digraph with a given degree sequence. *Combin. Probab. Comput. 13*, 3 (2004), 319–337.

[92] Cooper, C., Frieze, A., and Vera, J. Random deletion in a scale-free random graph process. *Internet Math. 1*, 4 (2004), 463–483.

[93] Cornuéjols, G., Nemhauser, G. L., and Wolsey, L. A. The uncapacitated facility location problem. In *Discrete location theory*, Wiley-Intersci. Ser. Discrete Math. Optim. Wiley, New York, 1990, pp. 119–171.

[94] Coster, M. J., Joux, A., LaMacchia, B. A., Odlyzko, A. M., Schnorr, C.-P., and Stern, J. Improved low-density subset sum algorithms. *Comput. Complexity 2*, 2 (1992), 111–128.

[95] Crawford, J. M., and Auton, L. D. Experimental results on the crossover point in random 3-SAT. *Artificial Intelligence 81*, 1-2 (1996), 31–57. Frontiers in problem solving: phase transitions and complexity.

[96] Creignou, N., and Daudé, H. Generalized satisfiability problems: minimal elements and phase transitions. *Theoretical Computer Science 302* (2003), 417–430.

[97] DeWitt, H. K. Applications of the theory of random graphs to average algorithm performance analysis. In *ACM 79: Proceedings of the 1979 annual conference* (New York, NY, USA, 1979), ACM Press, pp. 251–258.

[98] DeWitt, H. K., and Krieger, M. M. Expected length of shortest paths and algorithm behavior. In *Proceedings of the Tenth Southeastern Conference on Combinatorics, Graph Theory and Computing (Florida Atlantic Univ., Boca Raton, Fla., 1979)* (Winnipeg, Man., 1979), Congress. Numer., XXIII–XXIV, Utilitas Math., pp. 367–380.

[99] Dhamdhere, K., Ravi, R., and Singh, M. On two-stage stochastic minimum spanning trees. In *Integer Programming and Combinatorial Optimization: 11th International IPCO Conference, Berlin, Germany, June 8-10, 2005. Proceedings*, vol. 3509 / 2005 of *Lecture Notes in Comput. Sci.* Springer, Berlin, 2005, pp. 321–334.

[100] Díaz, J., Pérez, X., Serna, M. J., and Wormald, N. C. Connectivity for wireless agents moving on a cycle or grid. In *STACS 2005*, vol. 3404 of *Lecture Notes in Comput. Sci.* Springer, Berlin, 2005, pp. 353–364.

[101] Díaz, J., Petit, J., and Serna, M. Random geometric problems on $[0,1]^2$. In *Randomization and approximation techniques in computer science (Barcelona, 1998)*, vol. 1518 of *Lecture Notes in Comput. Sci.* Springer, Berlin, 1998, pp. 294–306.

[102] Diestel, R. *Graph theory*, third ed., vol. 173 of *Graduate Texts in Mathematics*. Springer-Verlag, Berlin, 2005.

[103] Dorogovtsev, S. N., and Mendes, J. F. F. *Evolution of networks*. Oxford University Press, Oxford, 2003. From biological nets to the Internet and WWW.

[104] Drinea, E., Enachescu, M., and Mitzenmacher, M. Variations on random graph models for the web. Tech. rep., Harvard University, 2001.

[105] Dubois, O., Boufkhad, Y., and Mandler, J. Typical random 3-SAT formulae and the satisfiability threshold. In *Proceedings of the 11th Annual ACM-SIAM Symposium on Discrete Algorithms* (2000), pp. 126–127.

[106] Erdős, P., and Rényi, A. On random graphs. I. *Publ. Math. Debrecen 6* (1959), 290–297.

[107] Fabrikant, A., Koutsoupias, E., and Papadimitriou, C. H. Heuristically optimized trade-offs: a new paradigm for power laws in the internet. In *Automata, languages and programming*, vol. 2380 of *Lecture Notes in Comput. Sci.* Springer, Berlin, 2002, pp. 110–122.

[108] Faloutsos, M., Faloutsos, P., and Faloutsos, C. On power-law relationships of the internet topology. In *SIGCOMM '99: Proceedings of the conference on Applications, technologies, architectures, and protocols for computer communication* (New York, NY, USA, 1999), ACM Press, pp. 251–262.

[109] Feige, U. A tight lower bound on the cover time for random walks on graphs. *Random Structures Algorithms 6*, 4 (1995), 433–438.

[110] Feige, U. A tight upper bound on the cover time for random walks on graphs. *Random Structures Algorithms 6*, 1 (1995), 51–54.

[111] Feige, U. Relations between average case complexity and approximation complexity. In *Proc. 34th ACM STOC* (2002).

[112] Feige, U., and Kilian, J. Heuristics for semirandom graph problems. *J. Comput. System Sci. 63*, 4 (2001), 639–671. Special issue on FOCS 98 (Palo Alto, CA).

[113] Feige, U., and Krauthgamer, R. Finding and certifying a large hidden clique in a semirandom graph. *Random Structures Algorithms 16*, 2 (2000), 195–208.

[114] Feige, U., and Ofek, E. Spectral techniques applied to sparse random graphs. *Random Structures and Algorithms 27*, 2 (2005), 251–275.

[115] Feige, U., and Vilenchik, D. A local search algorithm for 3-SAT. Tech. rep., The Weizmann Institute, 2004.

[116] Fernandez de la Vega, W. On random 2-SAT. Manuscript, 1992.

[117] Flaxman, A. D. A spectral technique for random satisfiable 3CNF formulas. In *Proceedings of the Fourteenth Annual ACM-SIAM Symposium on Discrete Algorithms (Baltimore, MD, 2003)* (New York, 2003), ACM, pp. 357–363.

[118] Flaxman, A. D. A sharp threshold for a random constraint satisfaction problem. *Discrete Math. 285*, 1-3 (2004), 301–305.

[119] Flaxman, A. D., Frieze, A., and Krivelevich, M. On the random 2-stage minimum spanning tree. In *SODA '05: Proceedings of the sixteenth annual ACM-SIAM symposium on Discrete algorithms* (Philadelphia, PA, USA, 2005), Society for Industrial and Applied Mathematics, pp. 919–926.

[120] Flaxman, A. D., Frieze, A., and Krivelevich, M. On the random 2-stage minimum spanning tree. *Random Structures Algorithms 28*, 1 (2006), 24–36.

[121] Flaxman, A. D., and Frieze, A. M. The diameter of randomly perturbed digraphs and some applications. In *Proc. of the 8th Int. Workshop on Randomization and Computation (RANDOM)* (2004), K. Jansen, S. Khanna, J. D. P. Rolim, and D. Ron, Eds., vol. 3122 of *Lecture Notes in Computer Science*, Springer, pp. 345–356.

[122] Flaxman, A. D., Frieze, A. M., and Vera, J. Adversarial deletion in a scale free random graph process. In *SODA '05: Proceedings of the sixteenth annual ACM-SIAM symposium on Discrete algorithms* (Philadelphia, PA, USA, 2005), Society for Industrial and Applied Mathematics, pp. 287–292.

[123] Flaxman, A. D., Frieze, A. M., and Vera, J. C. On the average case performance of some greedy approximation algorithms for the uncapacitated facility location problem. In *STOC'05: Proceedings of the 37th Annual ACM Symposium on Theory of Computing* (New York, 2005), ACM, pp. 441–449.

[124] Flaxman, A. D., and Przydatek, B. Solving medium-density subset sum problems in expected polynomial time. In *STACS 2005*, vol. 3404 of *Lecture Notes in Comput. Sci.* Springer, Berlin, 2005, pp. 305–314.

[125] Flaxman, A. D., and Sorkin, G. B. Tools for average-case analysis of optimization heuristics. In *Handbooks of Operations Research and Management Science: Approximation and Heuristics*. In preparation.

[126] Ford, G. W., and Uhlenbeck, G. E. Combinatorial problems in the theory of graphs. I. *Proc. Nat. Acad. Sci. U. S. A. 42* (1956), 122–128.

[127] Friedgut, E. Sharp thresholds of graph properties, and the k-sat problem. *J. Amer. Math. Soc. 12*, 4 (1999), 1017–1054. With an appendix by Jean Bourgain.

[128] Friedgut, E. Hunting for sharp thresholds. *Random Structures Algorithms 26*, 1-2 (2005), 37–51.

[129] Friedman, J., and Goerdt, A. Recognizing more unsatisfiable random 3-SAT instances efficiently. In *Automata, languages and programming*, vol. 2076 of *Lecture Notes in Comput. Sci.* Springer, Berlin, 2001, pp. 310–321.

[130] Friedman, J., Kahn, J., and Szemerédi, E. On the second eigenvalue of random regular graphs. In *Proceedings of the 21st annual ACM symposium on Theory of computing* (New York, NY, USA, 1989), ACM Press, pp. 587–598.

[131] Frieze, A., and Krivelevich, M. Hamilton cycles in random subgraphs of pseudo-random graphs. *Discrete Math. 256*, 1-2 (2002), 137–150.

[132] Frieze, A., and McDiarmid, C. Algorithmic theory of random graphs. *Random Structures Algorithms 10*, 1-2 (1997), 5–42. Average-case analysis of algorithms (Dagstuhl, 1995).

[133] Frieze, A., and Sorkin, G. A note on random 2-sat with prescribed literal degrees. In *Proceedings of the 13th Annual ACM-SIAM Symposium on Discrete Algorithms* (2002).

[134] Frieze, A., and Wormald, N. C. Random k-SAT: a tight threshold for moderately growing k. *Combinatorica 25*, 3 (2005), 297–305.

[135] Frieze, A. M. On the value of a random minimum spanning tree problem. *Discrete Appl. Math. 10*, 1 (1985), 47–56.

[136] Frieze, A. M. On the Lagarias-Odlyzko algorithm for the subset sum problem. *SIAM J. Comput. 15*, 2 (1986), 536–539.

[137] Frieze, A. M., and Reed, B. Probabilistic analysis of algorithms. In *Probabilistic methods for algorithmic discrete mathematics*, vol. 16 of *Algorithms Combin.* Springer, Berlin, 1998, pp. 36–92.

[138] Frieze, A. M., and Suen, S. Analysis of two simple heuristics on a random instance of k-SAT. *J. Algorithms 20*, 2 (1996), 312–355.

[139] Galil, Z., and Margalit, O. An almost linear-time algorithm for the dense subset-sum problem. *SIAM J. Comput. 20*, 6 (1991), 1157–1189.

[140] Gilbert, E. N. Enumeration of labelled graphs. *Canad. J. Math. 8* (1956), 405–411.

[141] Goel, A., Rai, S., and Krishnamachari, B. Monotone properties of random geometric graphs have sharp thresholds. *Ann. Appl. Probab. 15*, 4 (2005), 2535–2552.

[142] Goerdt, A. A threshold for unsatisfiability. *J. Comput. System Sci. 53*, 3 (1996), 469–486.

[143] Goerdt, A., and Jurdziński, T. Some results on random unsatisfiable k-Sat instances and approximation algorithms applied to random structures. *Combin. Probab. Comput. 12*, 3 (2003), 245–267. Combinatorics, probability and computing (Oberwolfach, 2001).

[144] Goerdt, A., and Lanka, A. Recognizing more random unsatisfiable 3-SAT instances efficiently. In *Typical case complexity and phase transitions*, vol. 16 of *Electron. Notes Discrete Math.* Elsevier, Amsterdam, 2003, p. 26 pp. (electronic).

[145] Goldreich, O., Goldwasser, S., and Ron, D. Property testing and its connection to learning and approximation. *J. ACM 45*, 4 (1998), 653–750.

[146] Goldreich, O., and Ron, D. Property testing in bounded degree graphs. *Algorithmica 32*, 2 (2002), 302–343.

[147] Grimmett, G. *Percolation*, second ed., vol. 321 of *Grundlehren der Mathematischen Wissenschaften [Fundamental Principles of Mathematical Sciences]*. Springer-Verlag, Berlin, 1999.

[148] Grimmett, G. R., and McDiarmid, C. J. H. On colouring random graphs. *Math. Proc. Cambridge Philos. Soc. 77* (1975), 313–324.

[149] Hayes, B. Graph theory in practice: Part II. *American Scientist 88* (2000), 104–109.

[150] Hoefer, M. Experimental comparison of heuristic and approximation algorithms for uncapacitated facility location. In *Experimental and efficient algorithms*, vol. 2647 of *Lecture Notes in Comput. Sci.* Springer, Berlin, 2003, pp. 165–178.

[151] Hofmeister, T., Schöning, U., Schuler, R., and Watanabe, O. A probabilistic 3-SAT algorithm further improved. In *STACS 2002*, vol. 2285 of *Lecture Notes in Comput. Sci.* Springer, Berlin, 2002, pp. 192–202.

[152] Impagliazzo, R., and Naor, M. Efficient cryptographic schemes provably as secure as subset sum. *J. Cryptology 9*, 4 (1996), 199–216.

[153] Jain, K., Mahdian, M., Markakis, E., Saberi, A., and Vazirani, V. V. Greedy facility location algorithms analyzed using dual fitting with factor-revealing LP. *J. ACM 50*, 6 (2003), 795–824 (electronic).

[154] Jain, K., and Vazirani, V. V. Approximation algorithms for metric facility location and k-median problems using the primal-dual schema and Lagrangian relaxation. *J. ACM 48*, 2 (2001), 274–296.

[155] Janson, S., Knuth, D. E., Łuczak, T., and Pittel, B. The birth of the giant component. *Random Structures Algorithms 4*, 3 (1993), 231–358. With an introduction by the editors.

[156] Janson, S., Łuczak, T., and Rucinski, A. *Random graphs*. Wiley-Interscience Series in Discrete Mathematics and Optimization. Wiley-Interscience, New York, 2000.

[157] Janson, S., Stamatiou, Y. C., and Vamvakari, M. Bounding the unsatisfiability threshold of random 3-SAT. *Random Structures Algorithms 17*, 2 (2000), 103–116.

[158] Jerrum, M., and Sorkin, G. B. The Metropolis algorithm for graph bisection. *Discrete Appl. Math. 82*, 1-3 (1998), 155–175.

[159] Jones, N. D. Space-bounded reducibility among combinatorial problems. *J. Comput. System Sci. 11*, 1 (1975), 68–85.

[160] Juels, A., and Peinado, M. Hiding cliques for cryptographic security. *Des. Codes Cryptogr. 20*, 3 (2000), 269–280.

[161] Kaporis, A. C., Kirousis, L. M., and Lalas, E. G. The probabilistic analysis of a greedy satisfiability algorithm. In *Algorithms—ESA 2002*, vol. 2461 of *Lecture Notes in Comput. Sci.* Springer, Berlin, 2002, pp. 574–585.

[162] Kaporis, A. C., Kirousis, L. M., Stamatiou, Y. C., Vamvakari, M., and Zito, M. Coupon collectors, q-binomial coefficients and the unsatisfiability threshold. In *Theoretical computer science (Torino, 2001)*, vol. 2202 of *Lecture Notes in Comput. Sci.* Springer, Berlin, 2001, pp. 328–338.

[163] Karoński, M., and Łuczak, T. Random hypergraphs. In *Combinatorics, Paul Erdős is eighty, Vol. 2 (Keszthely, 1993)*, vol. 2 of *Bolyai Soc. Math. Stud.* János Bolyai Math. Soc., Budapest, 1996, pp. 283–293.

[164] Karp, R. M. The probabilistic analysis of some combinatorial search algorithms. In *Algorithms and complexity (Proc. Sympos., Carnegie-Mellon Univ., Pittsburgh, Pa., 1976)*. Academic Press, New York, 1976, pp. 1–19.

[165] Karp, R. M. The transitive closure of a random digraph. *Random Structures Algorithms 1*, 1 (1990), 73–93.

[166] Kim, J. H., and Vu, V. H. Sandwiching random graphs: universality between random graph models. *Adv. Math. 188*, 2 (2004), 444–469.

[167] Kirousis, L., Kranakis, E., Krizanc, D., and Stamatiou, Y. Approximating the unsatisfiability threshold of random formulas. *Random Structures and Algorithms 17* (2000), 103–116.

[168] Kleinberg, J. The small-world phenomenon: an algorithm perspective. In *Proceedings of the Thiry-Second Annual ACM Symposium on Theory of Computing* (New York, 2000), ACM, pp. 163–170 (electronic).

[169] Kleinberg, J. M., Kumar, R., Raghavan, P., Rajagopalan, S., and Tomkins, A. S. The web as a graph: measurements, models, and methods. In *Computing and combinatorics (Tokyo, 1999)*, vol. 1627 of *Lecture Notes in Comput. Sci.* Springer, Berlin, 1999, pp. 1–17.

[170] Knuth, D. E. *Stable marriage and its relation to other combinatorial problems*, vol. 10 of *CRM Proceedings & Lecture Notes*. American Mathematical Society, Providence, RI, 1997. An introduction to the mathematical analysis of algorithms, Translated from the French by Martin Goldstein and revised by the author.

[171] Knuth, D. E., Motwani, R., and Pittel, B. Stable husbands. *Random Structures Algorithms 1*, 1 (1990), 1–14.

[172] Korupolu, M. R., Plaxton, C. G., and Rajaraman, R. Analysis of a local search heuristic for facility location problems. *J. Algorithms 37*, 1 (2000), 146–188. Ninth Annual ACM-SIAM Symposium on Discrete Algorithms (San Francisco, CA, 1998).

[173] Koutsoupias, E., and Papadimitriou, C. H. On the greedy algorithm for satisfiability. *Inform. Process. Lett. 43*, 1 (1992), 53–55.

[174] Krarup, J., and Pruzan, P. M. The simple plant location problem: survey and synthesis. *European J. Oper. Res. 12*, 1 (1983), 36–81.

[175] Krivelevich, M. Coloring random graphs—an algorithmic perspective. In *Mathematics and computer science, II (Versailles, 2002)*, Trends Math. Birkhäuser, Basel, 2002, pp. 175–195.

[176] Krivelevich, M., Sudakov, B., and Tetali, P. On smoothed analysis in dense graphs and formulas. To appear in *Random Structures Algorithms*.

[177] Krivelevich, M., and Vilenchik, D. Solving random satisfiable 3cnf formulas in expected polynomial time. In *SODA* (2006).

[178] Kumar, R., Raghavan, P., Rajagopalan, S., Sivakumar, D., Tomkins, A., and Upfal, E. Stochastic models for the web graph. In *41st Annual Symposium on Foundations of Computer Science (Redondo Beach, CA, 2000)*. IEEE Comput. Soc. Press, Los Alamitos, CA, 2000, pp. 57–65.

[179] Kumar, R., Raghavan, P., Rajagopalan, S., Sivakumar, D., Tomkins, A., and Upfal, E. The Web as a graph. In *Proc. 19th ACM SIGACT-SIGMOD-AIGART Symp. Principles of Database Systems, PODS* (15–17 2000), ACM Press, pp. 1–10.

[180] Kumar, R., Raghavan, P., Rajagopalan, S., and Tomkins, A. Trawling the Web for emerging cyber-communities. *Computer Networks (Amsterdam, Netherlands: 1999) 31*, 11–16 (1999), 1481–1493.

[181] Lagarias, J. C., and Odlyzko, A. M. Solving low-density subset sum problems. *J. Assoc. Comput. Mach. 32*, 1 (1985), 229–246.

[182] Levin, L. A. Universal enumeration problems. *Problemy Peredači Informacii 9*, 3 (1973), 115–116.

[183] Linial, N., and Rozenman, E. Random lifts of graphs: perfect matchings. *Combinatorica 25*, 4 (2005), 407–424.

[184] Linusson, S., and Wästlund, J. A proof of Parisi's conjecture on the random assignment problem. *Probab. Theory Related Fields 128*, 3 (2004), 419–440.

[185] Lovász, L. *Combinatorial problems and exercises*, second ed. North-Holland Publishing Co., Amsterdam, 1993.

[186] Lueker, G. S. Average-case analysis of off-line and on-line knapsack problems. *J. Algorithms 29*, 2 (1998), 277–305. SODA '95 (San Francisco, CA).

[187] Lueker, G. S. Exponentially small bounds on the expected optimum of the partition and subset sum problems. *Random Structures Algorithms 12*, 1 (1998), 51–62.

[188] Lyubashevsky, V. On random high density subset sums. Tech. rep., Electronic Colloquium on Computational Complexity, 2005.

[189] Mahdian, M., Ye, Y., and Zhang, J. A 2-approximation algorithm for the soft-capacitated facility location problem. In *Approximation, randomization, and combinatorial optimization*, vol. 2764 of *Lecture Notes in Comput. Sci.* Springer, Berlin, 2003, pp. 129–140.

[190] McDiarmid, C. On the method of bounded differences. In *London Mathematical Society Lecture Note Series*, vol. 141. Cambridge University Press, 1989, pp. 148–188.

[191] McSherry, F. Spectral partitioning of random graphs. In *42nd IEEE Symposium on Foundations of Computer Science (Las Vegas, NV, 2001)*. IEEE Computer Soc., Los Alamitos, CA, 2001, pp. 529–537.

[192] Mihail, M., and Papadimitriou, C. On the eigenvalue power law. In *Randomization and approximation techniques in computer science*, vol. 2483 of *Lecture Notes in Comput. Sci.* Springer, Berlin, 2002, pp. 254–262.

[193] Milgram, S. The small world problem. *Psychology Today 61*, 1 (1967), 60–67.

[194] Mitzenmacher, M. A brief history of generative models for power law and lognormal distributions. *Internet Math. 1*, 2 (2004), 226–251.

[195] Molloy, M., and Reed, B. A critical point for random graphs with a given degree sequence. *Random Structures Algorithms 6*, 2-3 (1995), 161–179.

[196] Molloy, M., and Reed, B. *Graph colouring and the probabilistic method*, vol. 23 of *Algorithms and Combinatorics*. Springer-Verlag, Berlin, 2002.

[197] Moore, C., and Russell, A. Explicit multiregister measurements for hidden subgroup problems; or, fourier sampling strikes back. Tech. Rep. TR-CS-2005-20, University of New Mexico, 2005.

[198] Motoki, M., and Uehara, R. Unique solution instance generation for the 3-satisfiability (3SAT) problem. Tech. rep., Tokyo Institute of Technology, 1998.

[199] Motwani, R. Average-case analysis of algorithms for matchings and related problems. *J. Assoc. Comput. Mach. 41*, 6 (1994), 1329–1356.

[200] Muthukrishnan, S., and Pandurangan, G. The bin-covering technique for thresholding random geometric graph properties. In *SODA '05: Proceedings of the sixteenth annual ACM-SIAM symposium on Discrete algorithms* (Philadelphia, PA, USA, 2005), Society for Industrial and Applied Mathematics, pp. 989–998.

[201] Nair, C., Prabhakar, B., and Sharma, M. Proofs of the Parisi and Coppersmith-Sorkin random assignment conjectures. *Random Structures Algorithms 27*, 4 (2005), 413–444.

[202] Newman, M. E. J. Random graphs as models of networks. In *Handbook of graphs and networks*. Wiley-VCH, Weinheim, 2003, pp. 35–68.

[203] Nisan, N. On read-once vs. multiple access to randomness in logspace. *Theoret. Comput. Sci. 107*, 1 (1993), 135–144. Structure in complexity theory (Barcelona, 1990).

[204] Penrose, M. *Random geometric graphs*, vol. 5 of *Oxford Studies in Probability*. Oxford University Press, Oxford, 2003.

[205] Reingold, O. Undirected ST-connectivity in log-space. In *Proceedings of the 37th annual ACM symposium on Theory of computing* (New York, NY, USA, 2005), ACM Press, pp. 376–385.

[206] Sántha, M., and Vazirani, U. V. Generating quasirandom sequences from semi-random sources. *J. Comput. System Sci. 33*, 1 (1986), 75–87. Twenty-fifth annual symposium on foundations of computer science (Singer Island, Fla., 1984).

[207] Schmidt, J. P. Probabilistic analysis of strong hypergraph coloring algorithms and the strong chromatic number. *Discrete Math. 66*, 3 (1987), 259–277.

[208] Schmidt-Pruzan, J., Shamir, E., and Upfal, E. Random hypergraph coloring algorithms and the weak chromatic number. *J. Graph Theory 9*, 3 (1985), 347–362.

[209] Schöning, U. A probabilistic algorithm for k-SAT and constraint satisfaction problems. In *40th Annual Symposium on Foundations of Computer Science (New York, 1999)*. IEEE Computer Soc., Los Alamitos, CA, 1999, pp. 410–414.

[210] Selman, B., Mitchell, D. G., and Levesque, H. J. Generating hard satisfiability problems. *Artificial Intelligence 81*, 1-2 (1996), 17–29. Frontiers in problem solving: phase transitions and complexity.

[211] Shmoys, D. B., Tardos, E., and Aardal, K. Approximation algorithms for facility location problems (extended abstract). In *STOC '97: Proceedings of the twenty-ninth annual ACM symposium on Theory of computing* (New York, NY, USA, 1997), ACM Press, pp. 265–274.

[212] Sipser, M. *Introduction to the Theory of Computation*. PWS Publishing Company, 1996.

[213] Spielman, D., and Teng, S.-H. Smoothed analysis of algorithms: why the simplex algorithm usually takes polynomial time. In *Proceedings of the Thirty-Third Annual ACM Symposium on Theory of Computing* (New York, 2001), ACM, pp. 296–305 (electronic).

[214] Spielman, D. A., and Teng, S.-H. Smoothed analysis: motivation and discrete models. In *Algorithms and data structures*, vol. 2748 of *Lecture Notes in Comput. Sci.* Springer, Berlin, 2003, pp. 256–270.

[215] Spielman, D. A., and Teng, S.-H. Smoothed analysis of algorithms: why the simplex algorithm usually takes polynomial time. *J. ACM 51*, 3 (2004), 385–463 (electronic).

[216] Steele, J. M. *Probability theory and combinatorial optimization*, vol. 69 of *CBMS-NSF Regional Conference Series in Applied Mathematics*. Society for Industrial and Applied Mathematics (SIAM), Philadelphia, PA, 1997.

[217] Strang, G. *Linear algebra and its applications*, second ed. Academic Press [Harcourt Brace Jovanovich Publishers], New York, 1980.

[218] Subramanian, C. R., Fürer, M., and Veni Madhavan, C. E. Algorithms for coloring semi-random graphs. *Random Structures Algorithms 13*, 2 (1998), 125–158.

[219] Sviridenko, M. An improved approximation algorithm for the metric uncapacitated facility location problem. In *Proceedings of the 9th International IPCO Conference on Integer Programming and Combinatorial Optimization* (London, UK, 2002), Springer-Verlag, pp. 240–257.

[220] Szpankowski, W. *Average case analysis of algorithms on sequences*. Wiley-Interscience Series in Discrete Mathematics and Optimization. Wiley-Interscience, New York, 2001. With a foreword by Philippe Flajolet.

[221] Talagrand, M. Concentration of measure and isoperimetric inequalities in product spaces. *Inst. Hautes Études Sci. Publ. Math.*, 81 (1995), 73–205.

[222] Verhoeven, Y. Random 2-SAT and unsatisfiability. *Inform. Process. Lett. 72*, 3-4 (1999), 119–123.

[223] Vollmer, H., and Wagner, K. W. Measure one results in computational complexity theory. In *Advances in algorithms, languages, and complexity*. Kluwer Acad. Publ., Dordrecht, 1997, pp. 285–312.

[224] Wagner, D. A generalized birthday problem (extended abstract). In *Advances in cryptology—CRYPTO 2002*, vol. 2442 of *Lecture Notes in Comput. Sci.* Springer, Berlin, 2002, pp. 288–303.

[225] Watts, D. J. *Small worlds*. Princeton Studies in Complexity. Princeton University Press, Princeton, NJ, 1999. The dynamics of networks between order and randomness.

[226] Watts, D. J., and Strogatz, S. H. Collective dynamics of "small-world" networks. *Nature 292* (1998), 440–442.

[227] West, D. B. *Introduction to graph theory*. Prentice Hall Inc., Upper Saddle River, NJ, 1996.

[228] Xu, K., Boussemart, F., Hemery, F., and Lecoutre, C. A simple model to generate hard satisfiable instances. In *Proc. of 19th International Joint Conference on Artificial Intelligence (IJCAI)* (Edinburgh, Scotland, 2005), pp. 337–342.

[229] Xu, K., and Li, W. Exact phase transitions in random constraint satisfaction problems. *Journal of Artificial Intelligence Research 12* (2000), 93–103.

[230] Yukich, J. E. *Probability theory of classical Euclidean optimization problems*, vol. 1675 of *Lecture Notes in Mathematics*. Springer-Verlag, Berlin, 1998.