

Fast Approximation Algorithms for Graph Partitioning Using Spectral and Semidefinite-Programming Techniques

by

Lorenzo Orecchia

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

in

Computer Science

in the

Graduate Division

of the

University of California, Berkeley

Committee in charge:

Professor Satish Rao, Chair
Professor David Aldous
Professor Umesh V. Vazirani

Spring 2011

**Fast Approximation Algorithms for Graph Partitioning Using Spectral and
Semidefinite-Programming Techniques**

Copyright 2011
by
Lorenzo Orecchia

Abstract

Fast Approximation Algorithms for Graph Partitioning Using Spectral and Semidefinite-Programming Techniques

by

Lorenzo Orecchia

Doctor of Philosophy in Computer Science

University of California, Berkeley

Professor Satish Rao, Chair

Graph-partitioning problems are a central topic of research in the study of approximation algorithms. They are of interest to both theoreticians, for their far-reaching connections to different areas of mathematics, and to practitioners, as algorithms for graph partitioning can be used as fundamental building blocks in many applications, such as image segmentation and clustering. While many theoretical approximation algorithms exist for graph partitioning, they often rely on multicommodity-flow computations that run in quadratic time in the worst case and are too time-consuming for the massive graphs that are prevalent in today's applications. In this dissertation, we study the design of approximation algorithms that yield strong approximation guarantees, while running in subquadratic time and relying on computational procedures that are often fast in practice. The results that we describe encompass two different approaches to the construction of such fast algorithms.

Our first result exploits the Cut-Matching game of Khandekar, Rao and Vazirani [41], an elegant framework for designing graph-partitioning algorithms that rely on single-commodity, rather than multicommodity, maximum flow. Within this framework, we give two novel algorithms that achieve an $O(\log n)$ -approximation for the problem of finding the cut of minimum expansion in the instance graph. The running time of these algorithms is $\tilde{O}(m + n^{3/2})$ and is dominated by a polylogarithmic number of single-commodity maximum-flow computations. Moreover, we give the first analysis of the limitations of the Cut-Matching game by showing that, for the minimum-expansion problem, no approximation better than $\Omega(\sqrt{\log n})$ can be obtained within this framework.

Our second result is a spectral method for the problem of finding the balanced cut of minimum conductance in a graph. In its design, we abandon the use of flow computations, in favor of spectral methods that give the algorithm a nearly-linear running time. At the same time, we can show that this algorithm achieves the asymptotically optimal approximation

ratio for spectral methods, settling an open question in the seminal work of Spielman and Teng [64] on spectral algorithms. Moreover, our algorithm has applications to the computation of graph decompositions, the solution of systems of linear equations and sparsification.

In both results, our approach to graph partitioning consists of a combination of spectral and semidefinite-programming techniques. A crucial ingredient in our algorithmic design is the use of random walks that, by hedging among many eigenvectors in the graph spectrum, capture the existence of low-conductance cuts better than single eigenvectors. The analysis of our methods is particularly simple, as it relies on a semidefinite programming formulation of the graph partitioning problem of choice. Indeed, we can describe our algorithms as primal-dual methods for solving a semidefinite program and show that certain random walks arise naturally from this approach. In this pursuit, we make use of the powerful Matrix Multiplicative Weight Update method of Arora and Kale [11], which helps us to formalize the connections between random walks, semidefinite programming and hedging, the common themes of our algorithms.

To my parents

Ai miei genitori

Contents

1	Introduction	1
1.0.1	Background	2
1.0.2	Fast Algorithms for Graph Partitioning	3
1.1	Summary of Results	4
1.1.1	Techniques	4
1.1.2	The Cut-Matching Game and Fast Algorithms for Graph Partitioning	5
1.1.3	Fast Spectral Algorithms for BALANCED SEPARATOR	7
1.2	Organization	8
1.3	Bibliographic Notes	9
2	Notation, Definitions and Basic Inequalities	10
2.1	Notation and Basic Facts	10
2.2	Basic Definitions	11
2.2.1	Graph-Partitioning Problems	11
2.2.2	Spectral Gap and Cheeger’s Inequality	12
2.2.3	Spectral and Flow Embeddings	13
2.2.4	Matrix Exponentiation	13
2.2.5	Heat-Kernel Random Walk	14
2.3	Basic Inequalities	14
2.3.1	Scalar Inequalities	14
2.3.2	Matrix Inequalities	15
3	The Matrix Multiplicative Weight Update Method	17
3.1	MWU Basics	18
3.2	The Vector MWU Algorithm	19
3.3	The Matrix MWU Algorithm	22
3.3.1	A More General Formulation of the Matrix MWU Algorithm	25
3.4	Solving SDPs by the Matrix MWU algorithm	27

4	The Cut-Matching Game and Fast Algorithms for Graph Partitioning	30
4.1	The Cut-Matching Game	30
4.2	Construction of Cut Strategies	34
4.2.1	Using Random Walks	34
4.2.2	Potential Analysis	37
4.3	The Random Walks	38
4.3.1	The Random Walk for \mathcal{C}_{KRV}	38
4.3.2	The Random Walk for \mathcal{C}_{EXP}	39
4.3.3	The Random Walk for \mathcal{C}_{NAT}	40
4.4	Analyzing the Potential Reduction	43
4.4.1	Algorithmic Template	44
4.5	Completing the Analysis	44
4.5.1	The \mathcal{C}_{KRV} Strategy	44
4.5.2	The \mathcal{C}_{EXP} Strategy	47
4.5.3	The \mathcal{C}_{NAT} Strategy	49
4.6	Matrix MWU Interpretation	51
4.7	Lower Bounds for the Cut-Matching Game	53
4.7.1	Proof Idea	53
4.7.2	Main Lemma	53
4.7.3	Preliminaries	54
4.7.4	Proof of Theorem 4.1.6	54
4.7.5	Proof of Lemma 4.7.4	56
4.8	Related Work	59
5	Fast Spectral Algorithms for Balanced Separator and Graph Decomposition	61
5.0.1	Our Result	62
5.0.2	Application to Graph Decomposition.	63
5.0.3	Our Techniques	63
5.1	Algorithm Statement and Main Theorems	65
5.1.1	Notation and Basic Facts	65
5.1.2	SDP Formulation	66
5.1.3	Primal-Dual Framework	68
5.2	ORACLE and Proof of the Main Theorem	71
5.3	Proof of Theorem on ORACLE	74
5.3.1	Preliminaries	74
5.3.2	Proof of Theorem 5.2.1	74
5.4	Random Walk Interpretation	76
5.5	Other Proofs	79
5.5.1	Proof of Basic Lemmata	79
5.5.2	Projection Rounding	79

Bibliography	85
A Omitted Proofs	91
A.1 Projection Lemma	91
A.2 Proof of Lemma 5.1.10	92

Acknowledgments

This dissertation owes a great deal to the mentorship I have received from three people during my time at Berkeley. First and foremost, I thank my advisor, Satish Rao, who has supported me through good and bad times, shaped my research in graph partitioning and given me confidence when I needed it. His remarkable intuition, perspective and commitment to our area of research inspire me, and scare me, every day. Secondly, Umesh Vazirani has been a source of invaluable advice at all times in my years as a graduate student. His guidance, delivered in modern day sutras, taught me a lot about research and about all things in life. Finally, Nisheeth Vishnoi has been a very close colleague and friend. He has taught me a lot, inspired me to work in approximation algorithms and given me great support in every way.

I also thank all my fellow graduate students in the Theory group for their friendship and for maintaining a serene and happy atmosphere in the group, even through the hardship and rigor of graduate school. I want to acknowledge in particular Alexandre Stauffer, who made me feel back at home by discussing soccer at every possible break, and Anindya De, for the fun and engaging conversations about life, India and, more importantly, cricket.

My friends, close and far, have been a continuous source of support through my graduate career. I thank all of them. In particular, Bruno Benedetti and Simone Gambini have always helped me in many ways, including keeping my confidence up and carrying boxes around in one of my many moves.

My parents Luisa and Carlo and my brother Giulio have always been there for me and have shaped my passion for mathematics and combinatorial reasoning. No words can express my gratefulness for their love and support. This dissertation is dedicated to my mother, who taught me what it meant to prove something mathematically, and to my father, who explained to me the applications of combinatorics to soccer leagues at the right time of my life.

In my time at Berkeley, I have also been lucky enough to acquire a new family who has shown incredibly generous support of me. Thank you, Elaine and Richard. And I want to give a special acknowledgment to Becca, who has been a friend and a sister to me, and has always made me feel very appreciated.

Finally, I thank my wife Sam. Without her, not only this dissertation, but all the beauty in my life today, would not have been possible.

Chapter 1

Introduction

Graph-partitioning problems are a central topic of research in the study of approximation algorithms. They are of interest to theoretical computer scientists for their far-reaching connections to spectral graph theory [21], metric embeddings [52] and the mixing of Markov chains [33]. Moreover, graph-partitioning problems have received particular attention in complexity theory, as settling their approximability is a fundamental open question in this field. But graph partitioning is also important for many practitioners, as algorithms for these problems are often fundamental primitives in other tasks, such as image segmentation [60], clustering [36] and social-network analysis [51]. In today’s applications, the input to these problems tend to consist of very large graphs, such as VLSI circuits or web-data graphs, that require algorithms running in time as close to linear as possible while preserving a good approximation ratio. The main object of this dissertation is the study of such fast approximation algorithms for graph partitioning.

Graph-partitioning problems can be generically defined as a family of NP-hard problems in which we are asked to partition the vertex set of a graph into two components such that few edges are going across the cut and the two components are both large. This is often achieved by optimizing a ratio of the number of edges cut and the “size” of the smaller side of the partition. Changing the notion of “size” yields different graph-partitioning problems.

In this dissertation, we will focus on the two main graph-partitioning objectives, *expansion* and *conductance*, but our techniques extend to other versions of graph partitioning. Given an undirected unweighted graph $G = (V, E)$, the expansion of a cut $S \subseteq V$, is defined as

$$\alpha(S) \stackrel{\text{def}}{=} \frac{|E(S, \bar{S})|}{\min\{|S|, |\bar{S}|\}}.$$

The conductance of S is

$$\phi(S) \stackrel{\text{def}}{=} \frac{|E(S, \bar{S})|}{\min\{\text{vol}(S), \text{vol}(\bar{S})\}},$$

where $\text{vol}(S)$ denotes the volume of S , i.e. the sum of the degrees of vertices in S . Then, the EXPANSION problem is that of finding the cut of G of minimum expansion $\alpha(G)$. In the

case of conductance, we will be interested in the BALANCED SEPARATOR problem, which asks us to find the cut of G of minimum conductance that contains at least some constant fraction of the volume of the graph. Besides being a theoretically rich problem, BALANCED SEPARATOR is of great practical importance, as it plays a crucial role in the design of divide-and-conquer algorithms [61], where the balance constraint is used to ensure that the depth of the recursion tree is at most logarithmic.

1.0.1 Background

Any approximation algorithm for a graph partitioning problem, e.g. EXPANSION, must exhibit a certificate that lower-bounds the expansion of all the exponentially-many cuts in the instance graph G . Different approaches to create such a succinct lower bound yield different approximation algorithms. Classically, two main techniques have been employed in this pursuit for graph partitioning: spectral and flow methods.

The spectral method is based on the realization that the graph cuts approximately determine the mixing of random walks over the graph, with sparse, low-expansion cuts causing slow mixing [21]. Alon and Milman [2] formalized this intuition showing that, in a d -regular graph

$$\sqrt{2 \cdot \text{gap}(G)} \geq \frac{\alpha(G)}{d} \geq \text{gap}(G), \quad (1.1)$$

where $\text{gap}(G)$ is the spectral gap [21] of G , defined in Chapter 2, a measure of the speed of mixing of random walks in the graph G . $\text{gap}(G)$ can be computed by finding the slowest mixing eigenvector of the graph Laplacian. Alon and Milman also show that a sweep cut of this eigenvector has expansion at most $d \cdot \sqrt{2 \cdot \text{gap}(G)}$, yielding a pseudo-approximation algorithm for EXPANSION. This algorithm achieves a good approximation ratio on graphs of large expansion, but can be very far from optimal in graphs of low expansion. Indeed, its approximation ratio can be as bad as $\Omega(\sqrt{m})^1$ [30].

The flow-based algorithm of Leighton and Rao [49] certifies expansion by routing a scaled copy of the complete graph K_V in G , i.e. by showing that a multicommodity flow with demands corresponding to the edges of K_V can be routed concurrently in G with minimum congestion c . This implies that

$$\alpha(G) \geq \frac{\alpha(K_V)}{c},$$

as the capacity of each cut in G must be able to support the capacity of the demands K_V . Conversely, if the algorithm is not able to route the flow with congestion c , it finds a cut S with

$$\alpha(S) \leq O(\log n) \cdot \frac{\alpha(K_V)}{c}.$$

¹We use the conventional notation $n \stackrel{\text{def}}{=} |V|$ and $m \stackrel{\text{def}}{=} |E|$ for an instance graph $G = (V, E)$.

This yields a $O(\log n)$ -approximation algorithm for EXPANSION. The algorithm of Leighton and Rao can also be stated as solving and rounding a natural linear-programming relaxation of the EXPANSION problem.

The spectral and flow paradigms were combined by Arora, Rao and Vazirani in their seminal paper [12], which gave a $O(\sqrt{\log n})$ -approximation algorithm for EXPANSION. The authors used a semidefinite program (SDP) to relax the EXPANSION problem and introduced the concept of *expander flows* to certify expansion. This consists of combining the spectral and flow lower bounds as follows: we first embed a graph H into G with minimum conductance c and then lower-bound the expansion of H using the spectral method. Note that this kind of certificate is more general than that of Leighton and Rao and hence yields tighter bounds on expansion. Arora et al. [12] followed the expander-flow paradigm by describing a $O(\sqrt{\log n})$ -approximation algorithm that routes a regular expander in G using multicommodity maximum-flow operations.

We now consider the running times of these algorithms. The spectral algorithm of Alon and Milman runs in nearly-linear-time² $O(m/\sqrt{\text{gap}(G)})$, if we use Lanczos method to compute the necessary eigenvector [29]. However, the more advanced algorithms of Leighton and Rao and Arora et al., with their superior approximation guarantees, were only known to run in quadratic time $\tilde{O}(n^2)$ [8], because of the use of multicommodity maximum-flow computations. This bottleneck has pushed many practitioners towards spectral methods and advanced heuristics, such as METIS [37], which perform well in many cases, but have no guarantee on their worst-case behavior.

1.0.2 Fast Algorithms for Graph Partitioning

Khandekar, Rao and Vazirani (KRV) [41] were the first to address this problem by using the expander-flow idea to route a simpler graph than a general regular expander. Their algorithm iteratively constructs an union of perfect matchings H that can be routed in G with small congestion and such that $\alpha(H)$ is large. Even though they do not directly use a spectral lower bound on the expansion of $\alpha(H)$, their methods to achieve a lower bound on $\alpha(H)$ are inherently spectral and in the spirit of the expander-flow paradigm. The main advantage of this algorithm is that, as it is routing a simpler, more restrictive kind of graph, it only needs to run a polylogarithmic number of *single-commodity maximum-flow* operations to achieve a $O(\log^2 n)$ -approximation. As single-commodity maxflows can be computed in time $\tilde{O}(m^{3/2})$ by the Goldberg-Rao algorithm [28], the algorithm of KRV [41] runs in time $\tilde{O}(m^{3/2})$, improving on the quadratic running time of the other algorithms achieving polylogarithmic approximation ratios.

In this paper, KRV also implicitly define the Cut-Matching game, a powerful framework to obtain fast approximation algorithms for EXPANSION and other graph-partitioning prob-

²Following Spielman and Teng [64], we denote by nearly-linear time a time that is almost linear in the number of edges, with a possible inverse polynomial dependence on the conductance of the graph.

lems using single-commodity maxflows. This framework will be the main object of our study in Chapter 4. We contribute in two ways to the study of the Cut-Matching game: first, we use it to construct improved algorithms for EXPANSION that achieve an approximation ratio of $O(\log n)$ in time $\tilde{O}(m^{3/2})$; secondly, we show a lower bound of $\Omega(\sqrt{\log n})$ on the approximation ratio achievable within the framework. This line of work was recently advanced by Sherman [59], who described a $O(\sqrt{\log n})$ -approximation algorithm for EXPANSION, outside of the Cut-Matching-game framework, that only uses $O(n^\epsilon)$ single-commodity-maxflow computations. This research direction has been very fruitful in obtaining strong approximation algorithms running in essentially single-commodity-maxflow time, but, for some applications, even a running time of $\tilde{O}(m^{3/2})$ can be excessive and a nearly-linear-time algorithm is highly desirable.

While the algorithm of Alon and Milman already achieves nearly-linear time to approximate conductance and expansion of regular graphs, no such algorithm was known for the important variant of BALANCED SEPARATOR until the fundamental work of Spielman and Teng [66, 67, 65, 64]. Their sequence of papers gives nearly-linear time algorithms to solve systems of linear equations involving Laplacian matrices and to construct high-quality spectral sparsifiers and makes use of a nearly-linear-time approximation algorithm for BALANCED SEPARATOR as an important primitive. However, their algorithm fails to achieve the asymptotic quadratic approximation ratio of Equation 1.0.1, which is optimal for spectral algorithms, and opens the question of whether such ratio can be obtained in nearly-linear time [62]. Our work in Chapter 5 resolves this standing open-question positively by constructing an algorithm BALCUT that meets these bounds. Our algorithm can be substituted for the original Spielman and Teng’s algorithm in the application to the solution of systems of linear equations and sparsification, yielding polylogarithmic improvements in running time.

1.1 Summary of Results

This dissertation will present our results in two areas: Chapter 4 contains our work on the Cut-Matching game, including our improved algorithms and lower bound argument. Chapter 5 deals with our novel algorithm for the BALANCED SEPARATOR problem. In the rest of this Introduction, we discuss our techniques and give a more precise sketch of our results and their significance.

1.1.1 Techniques

The two results in this dissertation share a common approach and a common set of techniques: in both cases, we construct algorithms that leverage the power and speed of random walks to find sparse cuts, yet we do not rely on classical theorems about the convergence of random walks, such as the lemmata of Lovasz and Simonovits [53]. Instead, our analysis is often a simple consequence of our SDP formulations of the problem, combined with other

SDP-based ideas. Indeed, this SDP angle seems to allow us to construct and handle advanced spectral algorithms that capture the behavior of sophisticated walks over the graph, without having to design such walks in an ad-hoc manner, as is done in local-random-walk methods.

A major role in this approach is played by Multiplicative Weight Update methods [11], a technique to which we dedicate Chapter 3. Multiplicative Weight Updates are a class of methods developed in online learning and game theory to analyze and exploit the effectiveness of hedging over multiple strategies in a repeated game. Despite their simplicity, these updates are capable of efficiently capturing the power of strong duality and have many applications in Theoretical Computer Science [10, 11], both as an algorithmic tool and as a proof strategy. In our work, Multiplicative Weight Updates come into play in solving SDP formulations of graph-partitioning problems, where they provide a framework for designing efficient and robust primal-dual schemes that rapidly converge to approximately optimal solutions. These methods allow us to formalize the connections among SDPs for graph partitioning problems, random walks and the concept of hedging, which is a common thread in our algorithms. We further discuss this link in Section 4.3, where we show how random walks arise naturally in the Cut-Matching game as a way of designing a robust potential function that captures the expansion of many cuts. We also describe in detail the relation between SDPs and random walks for our BALANCED SEPARATOR algorithm in Section 5.4.

A general novel theme of this dissertation is the use of semidefinite programming ideas to construct very efficient algorithms. Originally, the use of SDP programs was confined to the design of the most theoretical approximation algorithms, which served as proofs of polynomial-time approximability rather than procedures expected to run on real hardware. The application of SDP ideas to the construction of fast approximation algorithms was made possible by the discovery of efficient primal-dual schemes, based on Multiplicative Weight Updates, to approximately solve SDP programs, due to Arora and Kale [11]. This dissertation is a contribution to this exciting new area of research.

1.1.2 The Cut-Matching Game and Fast Algorithms for Graph Partitioning

The expander-flow formalism of Arora et al. [12] provides a new way of certifying expansion, by constructing a flow routing of a scaled expander in the instance graph. However, the first algorithms designed to use this idea [12, 8, 11] require multicommodity maximum-concurrent-flow operations that constitute a quadratic bottleneck in the running time. To obtain faster algorithms, researchers investigated the following question: is it possible to use single-commodity maximum-flow computations to route a sufficiently good expander or find a cut in the instance graph? More generally, can we design approximation algorithms for EXPANSION and other graph partitioning problems, while only using a small number of single-commodity maxflows?

KRV [41] were the first to answer this question positively by giving a $O(\log^2 n)$ ap-

proximation algorithm for EXPANSION that only uses polylogarithmic maxflow calls. More importantly, KRV not only gave an algorithm, but also a framework to design approximation algorithms for graph partitioning using single-commodity maxflows. This system is based on the elegant abstraction of the Cut-Matching game and allows us to separate the flow and spectral part of the algorithm, restricting the problem to a simple game of spectral flavor.

The Cut-Matching game is a multi-round game between a *cut player* and a *matching player*: at each interaction the cut player gives a bisection and the matching returns a perfect bipartite matching across the bisection. The goal of the cut player is to ensure that the union of the matchings quickly achieves as high expansion as possible, while the matching player tries to keep the expansion of the graph small for a large number of iterations. Following the reduction of KRV, a cut-player strategy achieving large expansion in a small number of iterations can be turned into an algorithm achieving a good approximation ratio using only a small number of maxflow computations.

In Chapter 4, we present a number of original results on the Cut-Matching game and its application to graph partitioning. Our main results are the following:

- We give two new cut-player strategies, yielding $O(\log n)$ -approximation algorithms for EXPANSION that run in time $\tilde{O}(m+n^{3/2})$. These strategies are presented in Section 4.2.
- We give the first lower bound argument for the Cut-Matching game, implying that no algorithm designed within this framework can achieve an approximation better than $\Omega(\sqrt{\log n})$ for EXPANSION. This appears in Section 4.7.

In the process of describing these results, we introduce two slightly different versions of the Cut-Matching game: in the first, as in KRV, the cut player is asked to lower-bound the expansion of the final graph. In the second, more restrictive version, the cut player must lower-bound the spectral gap of the graph instead. The spectral version of the game is of independent interest and helps us to better understand the power and limitations of the different strategies. For instance, while this version of the game is more challenging for the cut player than that based on expansion, we show that both our cut strategies and that of KRV yield comparable results under this stricter definition. Moreover, a tighter lower bound of $O(\log n / \log \log n)$ on the approximation achievable by any cut strategy was given by Sherman [59] for the spectral version of the game.

In the construction of our cut strategies, we emphasize the connection between strategies and random walks, formalizing why random walks arise naturally in the context of the Cut-Matching game as a means of “hedging” between many cuts at once. This is discussed in Section 4.3. Moreover, we pursue the relation between Cut-Matching game and hedging further, by showing that it is possible to apply the Multiplicative Weight Updates framework of Chapter 3 to give a very simple proof of the performance of one of our cut strategies.

From a lower-bound perspective, we expose the limitation of the original Cut-Matching-game framework by providing an $\Omega(\sqrt{\log n})$ lower bound on the approximation ratio achievable by any cut strategy. It is interesting that this is exactly the approximation ratio achieved

by Arora et al. [12]. By contrast, the best lower bound known for the approach of [12] is $\Omega(\log \log n)$, as proved in Devanur et al. [23] via an involved and technical argument. This suggests that the cut-matching game provides an attractive, concrete framework in which to study the complexity of finding sparse cuts. The proof of the lower bound relies on the following combinatorial statement that is related to the iso-perimetric inequality for the hypercube: given a bisection of the vertices of the d -dimensional hypercube, there is always a pairing of vertices between the two sides such that the average Hamming distance between paired vertices is at most $O(\sqrt{d})$.

1.1.3 Fast Spectral Algorithms for BALANCED SEPARATOR

The simplest algorithm for BALANCED SEPARATOR is the recursive spectral algorithm [36]. To test whether an instance graph G contains a balanced cut of conductance less than γ , this algorithm recursively applies the spectral algorithm of Alon and Milman: at each iteration, it test whether G has spectral gap larger than γ . If this is not the case, it finds a cut S of conductance $O(\sqrt{\gamma})$ and removes it from G together with all its adjacent edges. The algorithm then recurses on the residual graph. These recursive calls stop when the union of the cuts removed becomes balanced, in which case it forms a balanced cut of conductance $O(\sqrt{\gamma})$, or when the residual graph is found to have spectral gap at least γ , certifying that no balanced cut of the required conductance exists. As every iteration may only remove $O(1)$ volume and the eigenvector computation takes $\Omega(m)$ time, this algorithm may have quadratic running time.

The algorithm of Spielman and Teng [66] runs in time $\tilde{O}(m/\text{poly}(\gamma))$ and outputs a balanced cut of conductance $O(\sqrt{\gamma \cdot \text{polylog}(n)})$, if there exists a balanced cut of conductance less than γ . This improves the running time of the basic recursive algorithm for BALANCED SEPARATOR, while only losing a polylogarithmic factor in approximation. This algorithm is also spectral in nature and uses, as main subroutine, local random walks that run in time proportional to the volume of any sparse cut they find, so bypassing the obstacle encountered by the recursive algorithm. The idea of Spielman and Teng has been refined by Andersen, Chung and Lang [4] and Andersen and Peres [5], who gave nearly-linear time algorithms capable of outputting balanced cuts of conductance $O(\sqrt{\gamma \cdot \log(n)})$. These local methods are based on truncated random walks on the input graph and careful aggregation of the information obtained from these walks.

The question of whether the additional $\text{polylog}(n)$ -factor in the approximation ratio is necessary has been an object of study since the work of Spielman and Teng. Our research settles this question by giving a spectral algorithm BALCUT for BALANCED SEPARATOR that runs in time $\tilde{O}(m/\gamma)$ and outputs a balanced cut of conductance $O(\sqrt{\gamma})$. BALCUT is the first spectral algorithm to achieve this approximation ratio, which is asymptotically optimal for spectral methods [30], in nearly-linear time. Moreover, our algorithm shares some additional properties with that of Spielman and Teng [66], which make it applicable in the algorithms for the solution of systems of linear equations and sparsification [67, 65],

yielding polylogarithmic improvements in running time.

In the design of BALCUT, we depart from the local-random-walk paradigm of Spielman and Teng and consider instead a natural SDP-relaxation for the BALANCED SEPARATOR problem, which BALCUT solves approximately using the primal-dual method of Arora et al. [11] and a novel separation oracle. However, BALCUT also has an appealing interpretation based on random walks, which we describe in Section 5.4.

As we saw in our description of the recursive approach to BALANCED SEPARATOR, unbalanced cuts of low conductance are the main obstacles to finding a sparse balanced cut, as they are picked out by the spectral algorithm and do not yield much progress when removed. The problem with the recursive approach is that the $O(n)$ slowest-mixing eigenvectors of the graph may be all well-correlated with unbalanced cuts of low conductance, so that the algorithm may have to compute each of these $O(n)$ eigenvectors without finding a balanced cut. Intuitively, BALCUT overcomes this problem by considering a distribution over eigenvectors at every iteration, rather than a single eigenvector. This distribution is represented as a vector embedding of the vertices, and can also be seen as a candidate solution for an SDP formulation of BALANCED SEPARATOR. The sweep cut over the eigenvector, which is the rounding used by the spectral algorithm of Alon and Milman and also by Spielman and Teng, is replaced by a sweep cut over the radius of the vectors in the embedding (see Figure 1.1). This allows BALCUT to capture many unbalanced cuts of low conductance at once and allows us to bound the number of iterations by $O(\log n/\gamma)$.

Moreover, at any iteration, rather than removing the unbalanced cut found, BALCUT penalizes it by modifying the graph so that it is unlikely, but still possible, for a similar cut to turn up again in future iterations. Hence, in both its cut-finding and cut-eliminating procedures, BALCUT tends to “hedge its bets” more than the greedy recursive spectral method. This hedging, which ultimately allows BALCUT to achieve its faster running time, is implicit in the primal-dual framework of Arora and Kale[11].

1.2 Organization

Chapter 2 introduces basic notation and simple inequalities that will be useful in our proofs. We recommend using this chapter for reference and approaching this dissertation starting at Chapter 3, which presents the Multiplicative Weight Update methods and derives a simple extension of the primal-dual scheme of Arora and Kale [11] to approximately solve SDP programs. Our main original contributions are shown in Chapter 4, which deals with the Cut-Matching game, and Chapter 5, where we describe the algorithm BALCUT for the BALANCED SEPARATOR problem.

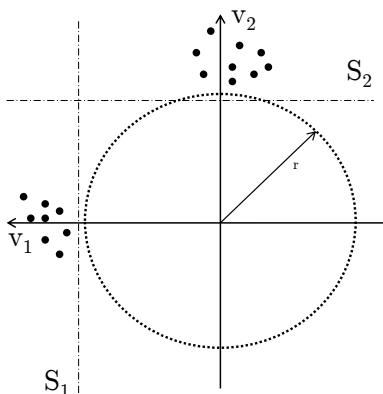


Figure 1.1: Schematic representation of the speed-up introduced by BALCUT when the instance graph contains many unbalanced cuts of low conductance. Let v_1 and v_2 be the two slowest-mixing eigenvectors of G . Assume that their minimum-conductance sweep cuts S_1 and S_2 are unbalanced cuts of conductance less than γ . If we use the recursive spectral algorithm, two iterations could be required to remove S_1 and S_2 . However, BALCUT considers a multidimensional embedding containing contributions from multiple eigenvectors and performs a radial sweep cut. This allows S_1 and S_2 to be removed in a single iteration.

1.3 Bibliographic Notes

The two results in Chapter 4 appeared as a single paper [56] in the Proceedings of the 40th ACM Symposium on Theory of Computing in 2008 and are joint work with Leonard J. Schulman, Umesh V. Vazirani and Nisheeth K. Vishnoi. The result of Chapter 5, which is joint work with Nisheeth K. Vishnoi, was published in the Proceedings of the 22nd Annual ACM-SIAM Symposium on Discrete Algorithms in 2011.

Chapter 2

Notation, Definitions and Basic Inequalities

In this chapter, we describe our notational conventions, formally introduce important definitions and state and prove some simple inequalities that are necessary for our proofs.

2.1 Notation and Basic Facts

Notation for special sets We let $[m]$ denote the set of integers $1, \dots, m$ and let $\Delta_n = \{x \in \mathbb{R}^n : x \geq 0 \text{ and } \sum x_i = 0\}$ be the n -dimensional simplex.

Graphs. All graphs in this dissertation are assumed to be undirected. An unweighted graph $G = (V, E)$ is defined by its vertex set V and edge set E , while the description of a weighted graph $G = (V, E, \omega)$ includes an additional weight vector $\omega \in \mathbb{R}^{V \times V}$, with support contained in E . The weight of edge $e \in E$ is ω_e . In this dissertation, all weighted graphs have non-negative weight vector. When not explicitly defined, $E(G)$ will denote the edge set of a graph G .

Graph matrices. For an undirected graph $G = (V, E)$, let $A(G)$ denote the adjacency matrix of G and $D(G)$ the diagonal matrix of degrees of H . The (combinatorial) Laplacian of G is defined as $L(G) \stackrel{\text{def}}{=} D(G) - A(G)$. Note that for all $x \in \mathbb{R}^V$,

$$x^T L(G)x = \sum_{\{i,j\} \in E_G} (x_i - x_j)^2.$$

Finally, $W(G)$ denotes the probability transition matrix of the natural random walk over G , which is defined as

$$W(G) \stackrel{\text{def}}{=} A(G)D(G)^{-1}.$$

This is the random walk that, given a starting vertex, picks one of its adjacent edges uniformly at random and transitions to the other end of that edge. When we are dealing with a single instance graph G , we will use the short forms D and L to denote $D(G)$ and $L(G)$ respectively.

Vector and matrix notation. We will be mostly working within a vector space \mathbb{R}^n . We will denote by I the identity matrix over this space. For a symmetric matrix M , we will use $M \succeq 0$ to indicate that M is positive semi-definite and $M \succ 0$ to denote that it is positive definite. The expression $A \succeq B$ is equivalent to $A - B \succeq 0$. For two matrices A, B of equal dimensions, let $A \bullet B \stackrel{\text{def}}{=} \text{Tr}(A^T B) = \sum_{ij} A_{ij} \cdot B_{ij}$. For a matrix $D \succeq 0$, define Δ_D as the set of matrices $X \succeq 0$ with $D \bullet X = 1$. If $X \in \Delta_D$, we call X a *density matrix*.

Real symmetric matrices will be play a major role in this dissertation. Hence, for a symmetric matrix $A \in \mathbb{R}^{n \times n}$, we use the following notation: the eigenvalues of A are denoted as $\lambda_1(A) \leq \lambda_2(A) \leq \dots \leq \lambda_n(A)$. $\lambda_{\min}(A)$ will be used as an alternative notation for $\lambda_1(A)$. We will also deal with generalized eigenvalues. For symmetric matrices $A, B \in \mathbb{R}^{n \times n}$ with $B \succ 0$, $\lambda_{i,B}(A)$ will stand for the i th smallest eigenvalue of A with respect to B , i.e.

$$\lambda_{i,B}(A) = \lambda_i(B^{-1/2} A B^{-1/2}).$$

Finally, for a matrix A , we also indicate by t_A the time necessary to compute the matrix-vector multiplications Au for any vector u .

2.2 Basic Definitions

2.2.1 Graph-Partitioning Problems

The main graph-partitioning objectives that we will be considering are expansion and conductance. For a graph $G = (V, E)$, the expansion of a cut $S \subseteq V$, is defined as

$$\alpha(S) \stackrel{\text{def}}{=} \frac{|E(S, \bar{S})|}{\min\{|S|, |\bar{S}|\}}.$$

Let vertex $i \in V$ have degree d_i in G and define the volume of a cut $S \subseteq V$, as

$$\text{vol}(S) = \sum_{i \in V} d_i.$$

Then, the conductance of $S \subseteq V$ is

$$\phi(S) \stackrel{\text{def}}{=} \frac{|E(S, \bar{S})|}{\min\{\text{vol}(S), \text{vol}(\bar{S})\}},$$

We also introduce notation for the minimum expansion and minimum conductance of any cut in G .

$$\alpha(G) = \min_{S \subseteq V} \alpha(S),$$

$$\phi(G) = \min_{S \subseteq V} \phi(S).$$

In some cases, we want to restrict our attention to cuts of large volume. A cut $S \subseteq V$ is b -balanced if

$$\text{vol}(S) \geq b \cdot \text{vol}(V).$$

We can now introduce the two main problems we address in this dissertation.

Definition 2.2.1. The EXPANSION problem on input $G = (V, E)$ is the problem of finding the cut $S \subseteq V$ of minimum expansion in G .

Definition 2.2.2. The BALANCED SEPARATOR problem on input $G = (V, E)$ and a constant parameter $b \in (0, 1/2]$ is the problem of finding the b -balanced cut $S \subseteq V$ of minimum conductance in G .

While all our instance graphs will be unweighted and undirected, for our analysis we will sometimes need to extend the concepts of conductance and expansion to weighted graphs. This is done by replacing the cardinality of the edges cut with their weight at the numerator. The denominator is unchanged for expansion, while for conductance the degree of each vertex is now the sum of the weight of the edges adjacent to it. Similarly, for a weighted graph $H = (V, E(H), \omega)$ the laplacian $L(H)$ is defined as the matrix for which, for all $x \in \mathbb{R}^V$

$$x^T L(H)x = \sum_{\{i,j\} \in E(H)} \omega_{ij} (x_i - x_j)^2.$$

2.2.2 Spectral Gap and Cheeger's Inequality

The spectral gap of a graph G is defined as

$$\text{gap}(G) \stackrel{\text{def}}{=} \min_{x^T D(G)x = 1} \frac{x^T L(G)x}{x^T D(G)x} = \lambda_{2,D}(L(G)).$$

Cheeger's Inequality [21] relates the conductance of G to its spectral gap.

Lemma 2.2.3 (Cheeger's Inequality).

$$\frac{\phi(G)^2}{2} \leq \text{gap}(G) \leq \phi(G).$$

We recommend the book [21] for an in-depth treatment of the spectral gap and its connection to other graph properties.

2.2.3 Spectral and Flow Embeddings

An embedding (or routing) of a graph $G = (V, E(G))$ into a graph $H = (V, E(H))$ enables us to compare the cut and spectral properties of G and H . An embedding of G into H is a solution to the concurrent-multicommodity-flow problem on H with demands equal to the edges of G , i.e. a way of routing the edges of G as flow paths into H . The embedding has congestion c if the congestion established by the concurrent multicommodity flow on every edge of H is at most c . Moreover, we say that the embedding has dilation ℓ if the edges of G are routed using flow paths in H of length at most ℓ . Denote by $\alpha_G(S)$ and $\alpha_H(S)$ the expansion of a cut $S \subseteq V$ in G and H respectively.

Lemma 2.2.4. [21] *If there exists an embedding of G into H with congestion c and dilation ℓ , then*

$$c \cdot \ell \cdot L(H) \succeq L(G)$$

and, for all cuts $S \subseteq V$,

$$c \cdot \alpha_H(S) \geq \alpha_G(S).$$

In particular, this means that $c \cdot \alpha(H) \geq \alpha(G)$.

2.2.4 Matrix Exponentiation

For a symmetric matrix $A \in \mathbb{R}^{n \times n}$, the matrix exponential is defined as

$$e^A \stackrel{\text{def}}{=} \sum_{i=0}^{\infty} \frac{A^i}{i!}.$$

Accordingly, for a scalar $b > 0$, we define

$$b^A \stackrel{\text{def}}{=} e^{\log(b) \cdot A}.$$

Notice that, by considering the eigenvector decomposition of A , we obtain the following relationship between the eigenvalues of A and that of its exponential. For all $i \in [n]$,

$$\lambda_i e^A = e^{\lambda_i(A)}.$$

The following special matrices, related to the exponential, play an important role in the algorithms of this dissertation.

Definition 2.2.5. Let $\epsilon \geq 0$. For matrices $A \in \mathbb{R}^{n \times n}$, $B \in \mathbb{R}^{n \times n}$ and for a projection matrix $\Pi \in \mathbb{R}^{n \times n}$, we let

$$E_\epsilon(A) \stackrel{\text{def}}{=} \frac{(1 - \epsilon)^A}{I \bullet A},$$

$$E_{\epsilon, B, \Pi}(A) \stackrel{\text{def}}{=} \frac{B^{-1/2} (1 - \epsilon)^{B^{-1/2} A B^{-1/2}} B^{-1/2}}{\Pi \bullet (1 - \epsilon)^{B^{-1/2} A B^{-1/2}}}$$

Notice that these special matrices are defined so that

$$\text{Tr}(E_\epsilon(A)) = 1$$

and

$$(B^{1/2}\Pi B^{1/2}) \bullet E_{\epsilon,B,\Pi}(A) = 1.$$

Because of this normalization, we have the following fact.

Fact 2.2.6. *Let $N = B^{1/2}\Pi B^{1/2}$. For $\alpha \in \mathbb{R}$,*

$$E_{\epsilon,B,\Pi}(A + \alpha N) = E_{\epsilon,B,\Pi}(A)$$

.

2.2.5 Heat-Kernel Random Walk

The heat-kernel random walk on a graph G with rate t is the random walk defined by the following probability-transition matrix

$$P(t) = e^{-t(I-W(G))} = e^{-tL(G)D(G)^{-1}}.$$

Notice that this can be seen as the random walk corresponding to applying the natural random walk for a number of steps which is Poisson-distributed with rate t as

$$e^{-t(I-W(G))} = e^{-t} \cdot \sum_{i=0}^{\infty} \frac{t^i}{i!} \cdot W(G)^i.$$

2.3 Basic Inequalities

2.3.1 Scalar Inequalities

Lemma 2.3.1. *For $\epsilon \in (0, 1)$ and $x \in [0, 1]$,*

$$(1 - \epsilon)^x \leq (1 - \epsilon x).$$

Proof. The inequality follows from the convexity of $(1 - \epsilon)^x$. □

Lemma 2.3.2. *For $x \in (0, 1/2)$, the following inequalities hold*

$$\log(1 - x) \leq -x, \quad \log(1 - x) \geq -x - x^2.$$

Proof. These are consequences of the Taylor expansion of $\log(1 - x)$. □

2.3.2 Matrix Inequalities

The following are standard matrix inequalities involving real symmetric matrices. Their proof, when not included, can be found in the textbooks [29] and [15].

Lemma 2.3.3. *For a symmetric matrix $A \in \mathbb{R}^{n \times n}$ such that $\rho I \succeq A \succeq 0$, we have*

$$e^{-A} \preceq \left(I - \frac{(1 - e^{-\rho})}{\rho} A \right).$$

Proof. Consider each eigenspace of A separately and notice that, for $x \in [0, \rho]$,

$$e^{-x} \leq \left(1 - (1 - e^{-\rho}) \frac{x}{\rho} \right)$$

as functions on the left-hand side and right-hand side have the same value at $x = 0$ and $x = \rho$, but the former is convex and the latter is linear. \square

Lemma 2.3.4. *For a symmetric matrix $M \in \mathbb{R}^{n \times n}$, such that $I \succeq M \succeq 0$ and a real number $\epsilon \in (0, 1)$, we have*

$$(1 - \epsilon)^M \preceq I - \epsilon M.$$

Proof. This is a straightforward consequence of Lemma 2.3.3 with $A = -\log(1 - \epsilon)M$. \square

The next two facts state basic properties of the trace function.

Fact 2.3.5 ([15]). *For matrices $A, B \in \mathbb{R}^{n \times n}$,*

$$\text{Tr}(AB) = \text{Tr}(BA).$$

Fact 2.3.6. *Given symmetric matrices $A, B, C \in \mathbb{R}^{n \times n}$ such that $A \succeq 0$ and $B \succeq C$,*

$$\text{Tr}(AB) \geq \text{Tr}(AC).$$

Proof. As $A \succeq 0$, we can write $A = A^{1/2}A^{1/2}$, where $A^{1/2} \succeq 0$. Then, by Fact 2.3.5,

$$\text{Tr}(AB) = \text{Tr}(A^{1/2}BA^{1/2}) = \sum_{i=1}^n (A^{1/2}e_i)^T B (A^{1/2}e_i) \geq \text{Tr}(A^{1/2}CA^{1/2}) = \text{Tr}(AC),$$

where the last inequality follows as $B \succeq C$. \square

The remaining inequalities are rearrangement inequalities for symmetric matrices under trace. They will play a fundamental role in Chapter 4.

Fact 2.3.7 ([15]). *Let $X \in \mathbb{R}^{n \times n}$. Then for any positive integer k ,*

$$\text{Tr}(X^{2k}) \leq \text{Tr}\left((XX^T)^{2k-1}\right).$$

Lemma 2.3.8 (Golden-Thompson Inequality [15]). *Let $X, Y \in \mathbb{R}^{n \times n}$ be symmetric matrices. Then,*

$$\mathrm{Tr}(e^{X+Y}) \leq \mathrm{Tr}(e^X e^Y).$$

Theorem 2.3.9. *Let $X, Y \in \mathbb{R}^{n \times n}$ be symmetric matrices. Then for any positive integer k ,*

$$\mathrm{Tr} \left[(XYX)^{2^k} \right] \leq \mathrm{Tr} \left[X^{2^k} Y^{2^k} X^{2^k} \right].$$

Proof. The proof is by induction on k . The base case is when $k = 0$, in which case the equality is trivial. Hence, we may assume by the induction hypothesis that the inequality is true for $k - 1$ (≥ 0), and we prove it for k . It follows from Fact 2.3.5 that $\mathrm{Tr} \left[(XYX)^{2^k} \right] = \mathrm{Tr} \left[(X^2 Y)^{2^k} \right]$. This, by Fact 2.3.7 is at most

$$\mathrm{Tr} \left[(X^2 Y (X^2 Y)^T)^{2^{k-1}} \right] = \mathrm{Tr} \left[(X^2 Y^2 X^2)^{2^{k-1}} \right].$$

The last equality follows from the fact that X, Y are symmetric. Hence, by the induction hypothesis on $X^2 Y^2 X^2$ we conclude that

$$\mathrm{Tr} \left[(X^2 Y^2 X^2)^{2^{k-1}} \right] \leq \mathrm{Tr} \left[(X^2)^{2^{k-1}} (Y^2)^{2^{k-1}} (X^2)^{2^{k-1}} \right] = \mathrm{Tr} \left[X^{2^k} Y^{2^k} X^{2^k} \right].$$

This completes the proof of the theorem. □

Chapter 3

The Matrix Multiplicative Weight Update Method

The *Multiplicative Weights Update* (MWU) method is a powerful, yet simple, technique that has been rediscovered and applied in many fields of Mathematics and Computer Science, including Game Theory, Machine Learning and Combinatorial Optimization. In a generic setting for this method, we consider an iterative process, in which at every round t an agent chooses a strategy p in a decision set D and obtains an associated penalty $\ell^{(t)}(p) \in [0, 1]$. The goal of the agent is to repeatedly pick strategies such that, over a large number of iterations, his total loss is not far from the loss incurred by the best fixed strategy in D . Notice that the loss functions $\ell^{(1)}, \dots, \ell^{(t)}, \dots$ are arbitrary and the agent may have no information about them in advance, except for the fact that they lie in the unit interval. In this scenario, the agent cannot achieve the goal of improving or matching the loss suffered by the best fixed strategy, but it can ensure its total loss is not much larger. This is just what the MWU method does.

The MWU method achieves its performance guarantee by maintaining weights on the strategies and picking the next strategies at random based on these weights. Once the loss function is revealed, the agent updates each weight by multiplying by a factor depending on the loss incurred by that strategy. This reweights the distribution and places more weight on the strategies that have received less loss so far. Intuitively, this allows the agent's loss to closely track the loss of these "good" strategies and do not perform much worse than the best of them.

The MWU idea is a cornerstone component in many important algorithms: in Machine Learning, it lies at the heart of many boosting procedures, such as AdaBoost [25], and is a major tool in online learning and optimization [46]; in Game Theory, MWU-style algorithms were proposed as early as the 1950s by Von Neumann and other researchers [17, 16], to compute equilibria in zero-sum games. Finally, in the study of algorithms, the MWU method has been applied to design fast approximate solvers for convex optimization programs [6, 11, 40], with particular success for flow [27, 24, 20] and cut problems [11], and has also

contributed to important developments in the study of Quantum Computation [32]. Today, the MWU method is recognized as a fundamental technique in the design of algorithms. We recommend the survey paper [10] and the theses of Rohit Khandekar [40] and Satyen Kale [34] for the reader interested in an unified development of the MWU method and an in-depth treatment of his many applications.

In this dissertation, we will use the MWU method to construct fast algorithms for graph partitioning problems and we will be mostly interested in the application of Matrix MWU [11] to solving certain SDPs associated with these problems. In this chapter, we provide the background for understanding the application of the MWU method in our work. The chapter is based on the treatment of MWU methods in the survey [10], in the textbook [18] and in the work of Arora and Kale [11]. Our presentation includes some minor novel extensions, found in Section 3.3.1 and Section 3.4, and is adapted to better integrate with the proofs of our results in the following chapters.

We start by describing the simplest MWU algorithm, the Vector MWU algorithm, and proceed to generalize it to the Matrix MWU. Finally, we demonstrate the application of Matrix MWU to approximately solving SDPs due to Arora and Kale [11] and extend it to the case of SDPs that are normalized by a possibly singular normalization matrix.

3.1 MWU Basics

We start by providing a more formal description of the generic setup for the MWU method and of its objective. In this setting, at every round t , the agent chooses a decision $p^{(t)}$ from a convex decision space D . Following this choice, the adversary reveals a convex loss function $\ell^{(t)} : D \rightarrow [0, 1]$.

In many regression problems, the loss functions $\{\ell^{(t)}\}$ are produced by some stationary stochastic process, so that a notion of risk for the algorithm can be derived. In this context, different algorithms can be compared in terms of risk minimization. However, here we are interested in the case where the sequence $\{\ell^{(t)}\}$ is completely arbitrary, and possibly adversarial to the algorithm. As the absence of an underlying stochastic process eliminates the possibility of using risk as a performance benchmark, we need to redefine our objective. For this purpose, we define a class of reference agents F , or “experts”. Each expert e proposes a strategy $f_e^{(t)} \in D$ at time t . The total loss of the forecasting algorithm is then compared with the total loss of the best fixed expert in F over the number T of rounds played. The difference in loss is known as the *regret* of the forecasting algorithm and is denoted $R^{(T)}$:

$$R^{(T)} \stackrel{\text{def}}{=} \sum_{t=1}^T \ell^{(t)}(p^{(t)}) - \min_{e \in F} \ell^{(t)}(f_e^{(t)}).$$

The MWU method allows us to construct agents achieving a small regret. In particular, in many cases of interest, algorithms based on the MWU method achieve a regret that grows

sublinearly with respect to T , i.e. $R^{(T)} = o(T)$. Algorithms based on the MWU method maintain a distribution over experts that they use to make their next prediction. Their key feature is the following: at every iteration the weights of the distribution are updated in a multiplicative fashion as a function of the experts' losses in the previous iteration. The update shifts the mass of the distribution towards the experts that are faring best and ensures that the loss of the algorithm will resemble their loss in the upcoming iterations.

3.2 The Vector MWU Algorithm

Our first example of MWU algorithm deals with the following simple scenario: we have n experts and every expert $i \in [n]$ has a fixed prediction $f_i^{(t)} = e_i \in \mathbb{R}^n$ for all t throughout the game. D is just the set of distributions over the predictions of the experts, i.e. the n -dimensional simplex Δ_n . To define the loss function at time t , we first fix the loss $\ell^{(t)}(e_i) \in [0, 1]$ incurred by each expert i at time t . Then, the loss of prediction $p^{(t)}$ at time t is just

$$\ell^{(t)}(p^{(t)}) \stackrel{\text{def}}{=} \sum_{i=1}^n p_i^t \ell^{(t)}(e_i). \quad (3.1)$$

In words, the loss of the algorithm is the expected loss of the distribution over experts that the algorithm chooses.

Finally, we can describe our first instance of a MWU algorithm, the Vector MWU algorithm. At every iteration t , the algorithm keeps a vector $w^{(t)} \in \mathbb{R}^n$ of positive weights over the experts. During the initialization, we set $w^{(1)} = \vec{1}$. At every iteration t , the vector $w^{(t)}$ is used to produce the current prediction by rescaling its entries by the normalization factor $W^{(t)} \stackrel{\text{def}}{=} \sum_{i=1}^n w_i^{(t)}$:

$$p^{(t)} \stackrel{\text{def}}{=} \frac{\sum_{i=1}^n w_i^{(t)} e_i}{W^{(t)}} = \frac{w^{(t)}}{W^{(t)}} \in \Delta_n. \quad (3.2)$$

Once the loss function $\ell^{(t)}$ is revealed, the weights are updated as follows for each $i \in [n]$:

$$w_i^{(t+1)} = w_i^{(t)} \cdot (1 - \epsilon)^{\ell^{(t)}(e_i)}, \quad (3.3)$$

where $\epsilon \in (0, 1)$ is a parameter of the algorithm, known as the *learning rate*. The rationale behind the update is clear: experts that incurred little loss in the last iteration do not have their weight decreased significantly, while experts that suffered a large loss have their weight greatly reduced and become less influential in the next choice of action. The learning rate regulates the speed at which the algorithm adjusts the distribution in response to the loss function. A larger learning rate yields a more sensitive agent, while a small learning rate produces a more conservative agent, one that is slower in shifting the distribution from one set of experts to another.

Unraveling the definition of the update, we find that

$$w_i^{(T+1)} = (1 - \epsilon)^{\sum_{t=1}^T \ell^{(t)}(e_i)}. \quad (3.4)$$

This expression highlights how the weight of each expert has an inverse exponential dependence on the current cumulative loss of each expert. This view will facilitate the generalization of the MWU method to the matrix case in Section 3.3.

Now, we can turn to the analysis of the Vector MWU algorithm. At every iteration t , we track the progress of the algorithm by the potential function $\sum_{i=1}^n w_i^{(t+1)}$. The next lemma shows how the potential function establishes a relation between the loss of the algorithm and that of the best expert. This analysis will be a blueprint for the analysis of bounds for more complex versions of the MWU algorithm.

Lemma 3.2.1. *For $t \in [T]$, $T \geq 1$, let $\ell^{(t)}$ be a loss function with $\ell^{(t)}(e_i) \in [0, 1]$ for all $i \in [n]$. For $p \in \Delta_n$, let $\ell^{(t)}(p)$ be defined as in Equation 3.1. Also, let $p^{(t)}$ and $w^{(t)}$ be defined as in Equations 3.2 and 3.3, with learning rate $\epsilon \in (0, 1)$. Then, the following inequality holds for any expert $i \in [n]$:*

$$(1 - \epsilon)^{\sum_{t=1}^T \ell^{(t)}(e_i)} \leq n \cdot \prod_{t=1}^T (1 - \epsilon \cdot \ell^{(t)}(p^{(t)})). \quad (3.5)$$

Proof. Let $W^{(t)} = \sum_{i=1}^n w_i^{(t)}$. We first relate this potential function to the performance of any expert i . For $i \in [n]$ and any $T \geq 1$, we have:

$$W^{T+1} \geq w_i^{(T+1)} = (1 - \epsilon)^{\sum_{t=1}^T \ell^{(t)}(e_i)}. \quad (3.6)$$

On the other hand, we can also relate the potential function to the performance of the algorithm:

$$W^{(T+1)} = \sum_{i=1}^n w_i^{(T+1)} = \sum_{i=1}^n w_i^{(T)} \cdot (1 - \epsilon)^{\ell^{(T)}(e_i)}.$$

By Lemma 2.3.1:

$$W^{(T+1)} \leq \sum_{i=1}^n w_i^{(T)} \cdot (1 - \epsilon \cdot \ell^{(T)}(e_i)) = W^{(T)} \cdot (1 - \epsilon \cdot \ell^{(T)}(p^{(T)})).$$

By iterating this argument, we obtain that the potential decreases multiplicatively with the loss of the algorithm:

$$W^{(T+1)} \leq W^{(1)} \cdot \prod_{t=1}^T (1 - \epsilon \cdot \ell^{(t)}(p^{(t)})). \quad (3.7)$$

As $W^{(1)} = n$, combining Equation 3.7 with Equation 3.6 yields the required inequality. \square

Lemma 3.2.1 already establishes a relation between the loss of the algorithm and that of the best expert. However, this relation is often too complex to work with. The next theorem gives a simplified, albeit looser, form of this bound, and one which is most useful in practice and that can be readily related to regret. Notice that this requires $\epsilon \in (0, 1/2)$.

Theorem 3.2.2. *For $t \in [T]$, $T \geq 1$, let $\ell^{(t)}$ be a loss function with $\ell^{(t)}(e_i) \in [0, 1]$ for all $i \in [n]$. For $p \in \Delta_n$, let $\ell^{(t)}(p)$ be defined as in Equation 3.1. Also, let $p^{(t)}$ and $w^{(t)}$ be defined as in Equations 3.2 and 3.3, with learning rate $\epsilon \in (0, 1/2)$. Then, the following inequality holds for any expert $i \in [n]$:*

$$\sum_{t=1}^T \ell^{(t)}(p^{(t)}) \leq \frac{\log n}{\epsilon} + (1 + \epsilon) \sum_{t=1}^T \ell^{(t)}(e_i)$$

Proof. Consider the inequality guaranteed by Lemma 3.2.1. Taking logarithms and rearranging, we have, for all i :

$$-\sum_{t=1}^T \log(1 - \epsilon \cdot \ell^{(t)}(p^{(t)})) \leq \log(1/1-\epsilon) \cdot \sum_{t=1}^T \ell^{(t)}(e_i) + \log n. \quad (3.8)$$

By Lemma 2.3.2,

$$-\log(1 - \epsilon \cdot \ell^{(t)}(p^{(t)})) \geq -\epsilon \cdot \ell^{(t)}(p^{(t)}), \quad (3.9)$$

$$\log(1/1-\epsilon) \leq \epsilon + \epsilon^2. \quad (3.10)$$

Substituting these bounds in Equation 3.8 and dividing by ϵ yields the desired result. \square

At this point, notice the dependence of Theorem 3.2.2 on the learning rate ϵ . For larger more aggressive settings of ϵ , the algorithm pays a smaller fixed penalty of $\log n/\epsilon$, but can suffer a larger relative penalty with respect to the best expert. Intuitively, this is the case as the algorithm quickly shifts to the best current experts, but is susceptible to incur larger losses if the loss functions penalize these experts the most. Contrarily, a smaller ϵ corresponds to an algorithm shifting its distribution more slowly and paying a larger fixed cost, while achieving a better bound in the second term.

As a consequence of Theorem 3.2.2, setting $\epsilon = \sqrt{\log n/T}$ yields a sublinear bound on the regret $R^{(T)}$:

$$R_T = \frac{\log n}{\epsilon} + \epsilon \cdot \sum_{t=1}^T \ell^{(t)}(e_i) \leq \frac{\log n}{\epsilon} + \epsilon T \leq 2\sqrt{T \log n}.$$

Lower bounds on regret have been given in different contexts and show that the regret achieved by the Vector MWU algorithm is essentially tight against arbitrary losses. These lower bounds are often based on a randomized setting of the losses. The intuition is that, if in every round the loss of each expert is distributed as an independent Bernoulli variable

with success probability $1/2$, the total loss of the algorithm in T iteration will be close to $T/2$, while, by lowerbounds on the tail of the binomial distribution, with non-zero probability there will be experts achieving $O(\sqrt{T \log n})$ loss less than the expected value. See the survey of Arora, Hazan and Kale [10] for a more formal development of this argument.

3.3 The Matrix MWU Algorithm

In this subsection, we generalize the MWU method to handle density matrices, rather than probability distributions. At the same time loss matrices replace the loss vectors defining an arbitrary loss for each expert. More precisely, we operate in the vector space \mathbb{R}^n . We let $\{e_1, \dots, e_n\}$ be the standard orthonormal basis for \mathbb{R}^n and associate an expert with each vector on the n -dimensional complex unit sphere \mathbb{S}^{n-1} . This is equivalent to generalizing from a set of experts corresponding to the standard basis $\{e_1, \dots, e_n\}$ to all combinations of the form $\sum_{i=1}^n x_i e_i$, where $x \in \mathbb{S}^{n-1}$. At every iteration t , a loss function is presented by defining losses

$$\ell^{(t)}(b_1^{(t)}), \dots, \ell^{(t)}(b_n^{(t)}) \in [0, 1]$$

for a basis of orthonormal vectors $\{b_1^{(t)}, \dots, b_n^{(t)}\}$ of \mathbb{R}^n . The loss of a general expert x is then derived from its representation in this basis:

$$\ell^{(t)}(x) \stackrel{\text{def}}{=} \sum_{i=1}^n \ell^{(t)}(b_i^{(t)}) (x^T b_i^{(t)})^2.$$

If we let $L^{(t)} \in \mathbb{R}^{n \times n}$ be the symmetric matrix with eigenvalues $\ell^{(t)}(b_1^{(t)}), \dots, \ell^{(t)}(b_n^{(t)})$ and eigenvectors $\{b_1^{(t)}, \dots, b_n^{(t)}\}$, we have:

$$\ell^{(t)}(x) = x^T L^{(t)} x.$$

Hence, the loss function is a quadratic function with values in the unit interval and is completely determined by the matrix $L^{(t)}$, which we call the *loss matrix* at time t .

The set D of actions available to the algorithm is once again the set of probability distributions over the set of experts F . As in the discrete case, the loss of an action is just the expectation of the loss under the corresponding distribution. Formally, if the distribution corresponding to the strategy played at time t is $D^{(t)}$, the algorithm suffers loss:

$$\mathbb{E}_{v \leftarrow D^{(t)}} [v^T L^{(t)} v] = L^{(t)} \bullet \mathbb{E}_{v \leftarrow D^{(t)}} [v v^T] = L^{(t)} \bullet \sigma^{(t)},$$

where $\sigma^{(t)}$ is the second moment matrix of the distribution $D^{(t)}$. Note that $\sigma^{(t)}$ is a density matrix, i.e. $\sigma^{(t)} \bullet I = 1$, and that every density matrix is the second moment matrix of a

distribution over the unit sphere. Because the loss incurred by $D^{(t)}$ is completely captured by the density matrix $\sigma^{(t)}$, we identify each strategy in D with such a density matrix and denote by $\sigma^{(t)}$ the density matrix chosen by the algorithm at time t . Henceforth, we then simply consider $D = \Delta_I$.

The goal of the algorithm is to minimize its total loss relative to the loss of the best fixed expert. But, in this context, the loss of the best fixed expert is just the minimum eigenvalue of $\sum_{t=1}^T L^{(t)}$ as

$$\lambda_{\min} \left(\sum_{t=1}^T L^{(t)} \right) = \min_{x \in \mathbb{S}^{n-1}} x^T \left(\sum_{t=1}^T L^{(t)} \right) x.$$

This eigenvalue interpretation will be particularly useful in the application of the Matrix MWU method to graph partitioning in Chapter 4.

Now we are ready to discuss how to update $\sigma^{(t)}$ at every round. As $\sigma^{(t)}$ is a density matrix, we just need to specify its eigenvectors and corresponding eigenvalues, which must form a probability distribution. The key idea in the update is again that of establishing an inverse exponential dependence between the weight of an expert and its current total loss, as in Equation 3.7. To do so, we consider the matrix $\sum_{s=1}^{t-1} L^{(s)}$, representing the current cumulative loss, and denote by $\{c_1^{(t)}, \dots, c_n^{(t)}\}$ its eigenvectors. We then associated to each of these vectors a weight $w_i^{(s)}$ as in Equation 3.7:

$$w_i^{(t)} \stackrel{\text{def}}{=} (1 - \epsilon)^{\sum_{s=1}^{t-1} \ell^{(s)}(c_i^{(t)})} = (1 - \epsilon)^{(c_i^{(t)})^T (\sum_{s=1}^{t-1} L^{(s)}) c_i^{(t)}}.$$

This setting of weights is described more succinctly by the weight matrix $W^{(t)}$, where

$$W^{(t)} \stackrel{\text{def}}{=} (1 - \epsilon)^{\sum_{s=1}^{t-1} L^{(s)}} \tag{3.11}$$

and $\{w_i^{(t)}\}$ is the set of eigenvalues of $W^{(t)}$. For a review of matrix exponentiation, see Section 2.2.4. Finally, $\sigma^{(t)}$ is taken to be the properly normalized copy of $W^{(t)}$,

$$\sigma^{(t)} = \frac{W^{(t)}}{\text{Tr}(W^{(t)})} = E_\epsilon \left(\sum_{s=1}^{t-1} L^{(s)} \right) \tag{3.12}$$

recalling the definition of E_ϵ in Section 2.2.4. $W^{(1)}$ is initialized to the identity matrix, so that $\sigma^{(1)} = E_\epsilon(0)$.

Notice that if the loss matrices are diagonal, then the matrix update of Equation 3.12, reduces to the Vector MWU update, as we always only consider the standard basis. The next theorem shows that the same bound achieved for the Vector MWU in Theorem 3.2.2 holds in the matrix setting. The proof follows the same line, starting with the analysis of a potential function equal to the sum of the weights, i.e. $\text{Tr}(W^{(t)})$. The only additional idea needed is the following: a shift in the eigenbasis of the loss matrix with respect to the current density matrix only helps the algorithm, as it drives the potential function even lower. This is captured by the Golden-Thompson Inequality in Lemma 2.3.8.

Lemma 3.3.1. For $t \in [T]$, $T \geq 1$, let $L^{(t)}$ be a loss matrix with $0 \preceq L^{(t)} \preceq I$. Let $\sigma^{(t)}$ and $W^{(t)}$ be defined as in Equations 3.7 and 3.11, with learning rate $\epsilon \in (0, 1)$. Then, the following inequality holds

$$(1 - \epsilon)^{\lambda_{\min}(\sum_{t=1}^T L^{(t)})} \leq n \cdot \prod_{t=1}^T (1 - \epsilon \cdot L^{(t)} \bullet \sigma^{(t)}).$$

Proof. As in the vector case, the right potential to consider is the sum of the weights, i.e. $\text{Tr}(W^{(t)})$. On one hand,

$$\text{Tr}(W^{(T+1)}) \geq (1 - \epsilon)^{\lambda_{\min}(\sum_{t=1}^T L^{(t)})}. \quad (3.13)$$

On the other hand, by the Golden-Thompson Inequality (Lemma 2.3.8):

$$\begin{aligned} \text{Tr}(W^{(T+1)}) &= \text{Tr} \left((1 - \epsilon)^{\sum_{t=1}^T L^{(t)}} \right) \leq \text{Tr} \left((1 - \epsilon)^{\sum_{t=1}^{T-1} L^{(t)}} (1 - \epsilon)^{L^{(T)}} \right) = \\ &= \text{Tr}(W^{(T)}) \text{Tr} \left((1 - \epsilon)^{L^{(T)}} \sigma^{(T)} \right). \end{aligned}$$

As $0 \preceq L^{(T)} \preceq I$, by Lemma 2.3.4 and Fact 2.3.6:

$$\begin{aligned} \text{Tr}(W^{(T+1)}) &\leq \text{Tr}(W^{(T)}) \text{Tr} \left((1 - \epsilon)^{L^{(T)}} \sigma^{(T)} \right) \leq \text{Tr}(W^{(T)}) \text{Tr} \left((I - \epsilon L^{(T)}) \sigma^{(T)} \right) = \\ &= \text{Tr}(W^{(T)}) (1 - \epsilon L^{(T)} \bullet \sigma^{(T)}). \end{aligned}$$

Applying the same argument recursively, we obtain:

$$\text{Tr}(W^{(T+1)}) \leq \text{Tr}(W^{(1)}) \prod_{t=1}^{T-1} (1 - \epsilon \cdot L^{(t)} \bullet \sigma^{(t)}). \quad (3.14)$$

Combining Equation 3.13 and Equation 3.14, and noticing that $\text{Tr}(W^{(1)}) = n$, completes the proof. \square

Assuming $\epsilon \in (0, 1/2)$, we can now obtain exactly the same bound as in Theorem 3.2.2.

Theorem 3.3.2. For $t \in [T]$, $T \geq 1$, let $L^{(t)} \in \mathbb{R}^{n \times n}$ be a loss matrix with $0 \preceq L^{(t)} \preceq I$. Let $\sigma^{(t)}$ and $W^{(t)}$ be defined as in Equations 3.7 and 3.11, with learning rate $\epsilon \in (0, 1/2)$. Then, the following inequality holds

$$\sum_{t=1}^T L^{(t)} \bullet \sigma^{(t)} \leq \frac{\log n}{\epsilon} + (1 + \epsilon) \cdot \lambda_{\min} \left(\sum_{t=1}^T L^{(t)} \right).$$

Proof. Taking logs and rearranging:

$$-\sum_{t=1}^T \log(1 - \epsilon \cdot L^{(t)} \bullet \sigma^{(t)}) \leq \log(1/1-\epsilon) \cdot \lambda_{\min} \left(\sum_{t=1}^T L^{(t)} \right) + \log n. \quad (3.15)$$

By Lemma 2.3.2,

$$-\log(1 - \epsilon \cdot \ell^{(t)}(p^{(t)})) \geq -\epsilon \cdot \ell^{(t)}(p^{(t)}), \quad (3.16)$$

$$\log(1/1-\epsilon) \leq \epsilon + \epsilon^2. \quad (3.17)$$

Substituting these bounds in Equation 3.15 and dividing by ϵ yields the desired result. \square

The Matrix MWU algorithm was introduced as a generalization of the Vector MWU algorithm by Tsuda, Rätsch and Warmuth [46], Warmuth and Kuzmin [71] and independently by Arora and Kale [11] in the context of Combinatorial Optimization. The Matrix MWU algorithm has been applied in many contexts in Theoretical Computer Science [11, 32]. In this dissertation, it will play a fundamental role in our construction of fast algorithms for graph partitioning.

3.3.1 A More General Formulation of the Matrix MWU Algorithm

In this subsection, we prove a generalized version of Theorem 3.3.2 that allows us to deal with different normalization conditions. This result is an extension of work of Steurer [68] and Arora et al. [11] to semidefinite positive matrices, including possibly singular ones. The purpose of the result is that of describing a variant of the Matrix MWU algorithm for which the decision space is the set Δ_N for a choice of $N \succeq 0$, different from the identity and possibly singular.

The following theorem shows that a modified form of the Matrix MWU update of Equation 3.12 achieves a regret bound similar to that of Theorem 3.3.2. The modified update is based on a decomposition of N as

$$N = D^{1/2} \Pi D^{1/2},$$

where Π is a projection and $D \succ 0$ is any full-rank matrix that satisfies the equality. It is easy to see that such a decomposition always exists by considering the eigenvector decomposition of N . However, different decompositions have different properties and some may yield a matrix D which is computationally easier. This is why we state the following theorem in this degree of generality. We will be applying in Chapter 5 to a specific decomposition different from the obvious one, but simpler for our algorithmic purposes.

Theorem 3.3.3. *Let $N \in \mathbb{R}^{m \times m}$ be a symmetric matrix of rank n with $N \succeq 0$. Assume that N can be decomposed as*

$$N = D^{1/2} \Pi D^{1/2},$$

where $\Pi \in \mathbb{R}^{m \times m}$ is a projection matrix of rank n and $D \in \mathbb{R}^{m \times m}$ with $D \succ 0$. Let $\epsilon \in (0, 1/2)$ and let $\{Y^{(t)} \in \mathbb{R}^{m \times m}\}$ be a sequence of loss matrices such that $-\ell N \preceq Y^{(t)} \preceq \rho N$, where $\rho \geq \ell \geq 0$, for all t . Define the update

$$X^{(t)} = E_{\epsilon, D, \Pi} \left(\frac{1}{2\rho} \sum_{s=1}^{t-1} Y^{(s)} \right) \in \Delta_N.$$

Then, for any $T \geq 1$, we have

$$\sum_{t=1}^T Y^{(t)} \bullet X^{(t)} \leq \frac{2\rho \log n}{\epsilon} + T\epsilon\ell + (1 + \epsilon) \cdot \lambda_{\min, N} \left(\sum_{t=1}^T Y^{(t)} \right)$$

Proof. Consider the loss matrices

$$L^{(t)} = \frac{D^{-1/2} Y^{(t)} D^{-1/2} + \ell \cdot \Pi}{2\rho} \in \mathbb{R}^{m \times m},$$

and notice that

$$0 \preceq L^{(t)} \preceq \Pi$$

We will apply Theorem 3.3.2 to the loss functions $\{L^{(t)}\}$ in the vector space corresponding to the n -dimensional subspace of \mathbb{R}^m described by Π . To do so, we let

$$\sigma^{(t)} = E_{\epsilon, I, \Pi} \left(\sum_{s=1}^{t-1} L^{(s)} \right)$$

and notice that, by Fact 2.2.6, $X^{(t)} = D^{-1/2} \sigma^{(t)} D^{-1/2}$. We can now verify that the update $X^{(t)}$ is in Δ_D , as required, as

$$N \bullet X^{(t)} = \text{Tr}(N D^{-1/2} \sigma^{(t)} D^{-1/2}) = \Pi \bullet \sigma^{(t)} = 1.$$

Consider the restriction of $\sigma^{(t)}$ to the n -dimensional subspace described by Π and notice that, on this subspace, it is exactly equal to the update defined by Equation 3.12. Hence, by Theorem 3.3.2, we have

$$\sum_{t=1}^T \left(\frac{D^{-1/2} Y^{(t)} D^{-1/2} + \ell \cdot \Pi}{2\rho} \right) \bullet \sigma^{(t)} \leq \frac{\log n}{\epsilon} + (1 + \epsilon) \cdot \lambda_{\min, \Pi} \left(\sum_{t=1}^T \frac{D^{-1/2} Y^{(t)} D^{-1/2} + \ell \cdot \Pi}{2\rho} \right).$$

Using the fact that $\Pi \bullet \sigma^{(t)} = 1$ and rearranging:

$$\sum_{t=1}^T (D^{-1/2} Y^{(t)} D^{-1/2}) \bullet \sigma^{(t)} \leq \frac{2\rho \log n}{\epsilon} + T\epsilon\ell + (1 + \epsilon) \cdot \lambda_{\min, \Pi} \left(\sum_{t=1}^T D^{-1/2} Y^{(t)} D^{-1/2} \right).$$

Finally, by Fact 2.3.5,

$$Y^t \bullet X^{(t)} = \text{Tr}(Y^{(t)} D^{-1/2} \sigma^{(t)} D^{-1/2}) = \text{Tr}(D^{-1/2} Y^{(t)} D^{-1/2} \sigma^{(t)}) = (D^{-1/2} Y^{(t)} D^{-1/2}) \bullet \sigma^{(t)},$$

and by the definition of generalized eigenvector, we obtain the required result. \square

3.4 Solving SDPs by the Matrix MWU algorithm

In this subsection, we briefly explore the application of the Matrix MWU algorithm to the approximate solution of SDP problems. This direction was pioneered by Arora et al. [11]. This result is a minor extension of their framework that can be applied to slightly more general SDP problems.

We consider a generic SDP optimization problem on the variable $X \in \mathbb{R}^{m \times m}$. We isolate a particular constraint, $N \bullet X = 1$, in the primal SDP to act as a normalization constraint.

$$\begin{array}{ll}
 \text{PRIMAL :} & \min C \bullet X \\
 & \forall i \in [m], A_i \bullet X \geq b_i \\
 & N \bullet X = 1 \\
 & X \succeq 0 \\
 \text{DUAL :} & \max \sum_{i=1}^m \beta_i b_i + \alpha \\
 & C - \sum_{i=1}^m \beta_i A_i - \alpha N \succeq 0 \\
 & \alpha \in \mathbb{R}, \beta \in \mathbb{R}^m, \beta \geq 0
 \end{array}$$

Note that most SDP problems of interest can be formulated in this way. In particular, the constraint $A \bullet X \geq b$ can be expressed as $(A - bN) \bullet X \geq 0$. Moreover, it is always possible to obtain a normalization constraint by properly scaling the desired feasible solutions.

We apply binary search to the objective value of the primal to reduce from optimization to feasibility and focus on solving the following feasibility problem $\text{psdp}(\gamma)$ and a dual program $\text{dsdp}(\gamma)$.

$$\begin{array}{ll}
 \text{psdp}(\gamma) : & C \bullet X < \gamma \\
 & \forall i \in [m], A_i \bullet X \geq b_i \\
 & N \bullet X = 1 \\
 & X \succeq 0 \\
 \text{dsdp}(\gamma) : & \sum_{i=1}^m \beta_i b_i + \alpha \geq \gamma \\
 & C - \sum_{i=1}^m \beta_i A_i - \alpha N \succeq 0 \\
 & \alpha \in \mathbb{R}, \beta \in \mathbb{R}^m, \beta \geq 0
 \end{array}$$

The following is a simple consequence of weak SDP duality [58].

Fact 3.4.1. *If $\text{dsdp}(\gamma)$ has a feasible solution, $\text{psdp}(\gamma)$ has no feasible solution.*

In the rest of this subsection, we are going to make frequent use of the constraints in the $\text{dsdp}(\gamma)$ program. Hence, we define the following short-hand notation:

$$\begin{aligned}
 V(\alpha, \beta) &\stackrel{\text{def}}{=} \sum_{i=1}^m \beta_i b_i + \alpha, \\
 M(\alpha, \beta) &\stackrel{\text{def}}{=} C - \sum_{i=1}^m \beta_i A_i - \alpha N.
 \end{aligned}$$

Definition 3.4.2. An (ℓ, ρ) -oracle for $\text{psdp}(\gamma)$ is an algorithm that on input $X \in \Delta_N$, either fails or outputs (α, β) with $\alpha \in \mathbb{R}, \beta \in \mathbb{R}^m, \beta \geq 0$ satisfying

$$V(\alpha, \beta) \geq \gamma, \quad (3.18)$$

$$M(\alpha, \beta) \bullet X \geq 0, \quad (3.19)$$

$$-\ell N \preceq M(\alpha, \beta) \preceq \rho N \quad (3.20)$$

Fact 3.4.3. If an (ℓ, ρ) -oracle for $\text{psdp}(\gamma)$ does not fail on input $X \in \Delta_N$, X is infeasible for $\text{psdp}(\gamma)$.

Proof. Suppose X were feasible and let (α, β) be the output of the oracle. Then,

$$M(\alpha, \beta) \bullet X = C \bullet X - \sum_{i=1}^m \beta_i A_i \bullet X - \alpha N \bullet X < \gamma - \sum_{i=1}^m \beta_i b_i - \alpha = \gamma - V(\alpha, \beta) < 0.$$

This contradicts the definition of an (ℓ, ρ) -oracle. Hence, X must be infeasible. \square

An oracle returns information about the way in which the input candidate solution is infeasible in the coefficients α, β . The Matrix MWU algorithm will exploit this feedback to iteratively produce new candidate solutions. More formally, we will consider the Matrix MWU setting in which, at every iteration t , the algorithm must produce a candidate solution $X^{(t)} \in \Delta_N$. Then, $X^{(t)}$ is fed to a (ℓ, ρ) -oracle for $\text{psdp}(\gamma)$. If the oracle does not fail, we let $(\alpha^{(t)}, \beta^{(t)})$ be its output and set the loss matrix at time t to $M(\alpha^{(t)}, \beta^{(t)})$ and continue the repeated game. At every iteration, the MWU algorithm will incorporate the oracle's feedback to try and produce a feasible primal solution. The following theorem shows that, if the oracle does not fail for a sufficiently large number of iterations, it is possible to read off a near-feasible dual solution in the form of a feasible solution to $\text{dsdp}(\gamma - \delta)$ for small δ . By Fact 3.4.1, this implies that $\text{psdp}(\gamma - \delta)$ is infeasible and that the optimization program has objective value at least $\gamma - \delta$. As in the previous section, we assume we have a decomposition of $N \succeq 0$, as $N = D^{1/2} \Pi D^{1/2}$, where $D \succ 0$ and Π is a projection matrix.

Theorem 3.4.4. Let ORACLE be a (ℓ, ρ) -oracle for $\text{psdp}(\gamma)$ and let $\delta > 0$. Assume that N can be decomposed as

$$N = D^{1/2} \Pi D^{1/2},$$

where $\Pi \in \mathbb{R}^{m \times m}$ is a projection matrix of rank n and $D \in \mathbb{R}^{m \times m}$ with $D \succ 0$. Let $\epsilon = \min\{1/2, \delta/2\ell\}$. For $t \geq 1$, let

$$X^{(t)} = E_{\epsilon, D, \Pi} \left(\frac{1}{2\rho} \sum_{s=1}^{t-1} Y^{(s)} \right),$$

where $Y^{(t)} = M(\alpha^{(t)}, \beta^{(t)})$ and $(\alpha^{(t)}, \beta^{(t)})$ is the output of ORACLE on input $X^{(t)}$. Suppose that such output exists, i.e. ORACLE does not fail, for T rounds where

$$T = O\left(\frac{\rho \log n}{\delta \epsilon}\right) \leq \max\left\{O\left(\frac{\rho \log n}{\delta}\right), O\left(\frac{\rho \ell \log n}{\delta^2}\right)\right\}$$

and define $\bar{\alpha} \stackrel{\text{def}}{=} 1/T \sum_{t=1}^T \alpha^{(t)}$, $\bar{\beta} \stackrel{\text{def}}{=} 1/T \sum_{t=1}^T \beta^{(t)}$. Then $(\bar{\alpha} - \delta, \bar{\beta})$ is a feasible solution for $\text{dsdp}(\gamma - \delta)$.

Proof. Apply Theorem 3.3.3 to obtain that after T rounds:

$$\sum_{t=1}^T M(\alpha^{(t)}, \beta^{(t)}) \bullet X^{(t)} \leq \frac{2\rho \log n}{\epsilon} + T\epsilon\ell + (1 + \epsilon) \cdot \lambda_{\min, N} \left(\sum_{t=1}^T M(\alpha^{(t)}, \beta^{(t)}) \right).$$

As ORACLE is a (ℓ, ρ) -oracle, for all t , we have $M(\alpha^{(t)}, \beta^{(t)}) \bullet X^{(t)} \geq 0$. Hence,

$$(1 + \epsilon) \cdot \lambda_{\min, N} \left(\sum_{t=1}^T M(\alpha^{(t)}, \beta^{(t)}) \right) \geq -\frac{2\rho \log n}{\epsilon} - T\epsilon\ell.$$

Dividing by $(1 + \epsilon)T$,

$$\lambda_{\min, N} (M(\bar{\alpha}, \bar{\beta})) \geq -\frac{2\rho \log n}{(1 + \epsilon)\epsilon T} - \frac{\epsilon}{1 + \epsilon}\ell \geq -\frac{2\rho \log n}{\epsilon T} - \epsilon\ell.$$

For $T = 4\rho \log n / \delta \epsilon$ and $\epsilon \leq \delta / 2\ell$, we have

$$\begin{aligned} \epsilon\ell &\leq \frac{\delta}{2}, \\ \frac{2\rho \log n}{\epsilon T} &\leq \frac{\delta}{2}. \end{aligned}$$

These bounds yield

$$\lambda_{\min, N} (M(\bar{\alpha}, \bar{\beta})) \geq -\delta,$$

which implies

$$M(\bar{\alpha} - \delta, \bar{\beta}) = C - \sum_{i=1}^m \bar{\beta}_i A_i - (\bar{\alpha} - \delta)N \succeq 0. \quad (3.21)$$

Moreover, by Definition 3.4.2, $V(\bar{\alpha}, \bar{\beta}) = 1/T \sum_{t=1}^T V(\alpha^{(t)}, \beta^{(t)}) \geq \gamma$. Hence,

$$V(\bar{\alpha} - \delta, \bar{\beta}) = -\delta + V(\bar{\alpha}, \bar{\beta}) \geq \gamma - \delta. \quad (3.22)$$

Equations 3.21 and 3.22 imply that $(\bar{\alpha} - \delta, \bar{\beta})$ is feasible for $\text{dsdp}(\gamma - \delta)$. \square

This theorem generalizes the result of Arora et al. [11] by allowing us to apply the primal-dual framework with a possibly singular normalization matrix N .

Chapter 4

The Cut-Matching Game and Fast Algorithms for Graph Partitioning

In the first part of this chapter, we review the definition of the Cut-Matching game, its connection to graph partitioning and the work of Khandekar, Rao and Vazirani (KRV) [41]. Then, we present our two new strategies and compare them to that of KRV, focusing on how each strategy arises from a random-walk process over the current graph and highlighting the connections between cut strategies, random walks and the Matrix MWU algorithm. In the second part, we prove our lower-bound result for the Cut-Matching game based on expansion and relate it to Sherman's result on the spectral version of the game.

4.1 The Cut-Matching Game

An instance of the Cut-Matching Game $(\mathcal{G}(n), f(n), g(n))$ is defined by a multiround game $\mathcal{G}(n)$ between players \mathcal{C} , the cut player, and \mathcal{M} , the matching player, and by positive functions $f(n)$ and $g(n)$. The cut player is identified with the strategy \mathcal{C} it uses and the matching player with its strategy \mathcal{M} . The game starts with an empty weighted graph G_1 on vertex set V , where $V = [n]$ for even $n \in \mathbb{Z}$. Let $G_t = (V, E_t, \omega_t)$ be the resulting weighted graph after $t - 1$ rounds of the game. In each round $t \geq 1$, first the cut player \mathcal{C} chooses a bisection (S_t, \bar{S}_t) of $[n]$. This choice may depend on the actions of the players in the previous rounds and, in particular, on G_t . The matching player picks then a perfect matching M_t across the bisection (S_t, \bar{S}_t) . The action of the matching player may also depend on the actions of the players in the previous rounds and also on (S_t, \bar{S}_t) . The graph M_t is then added to the graph G_t to obtain graph G_{t+1} . Thus $G_{t+1} \stackrel{\text{def}}{=} G_t + M_t$, where the sum denotes edgewise addition of the weights. The weights of M_t are assumed to be one on each matching edge, as M_t is unweighted. The game terminates after $T \stackrel{\text{def}}{=} g(n)$ rounds. There are 2 possible winning criteria: according to the *gap criterion*, \mathcal{C} wins if $\text{gap}(G_{T+1})$ is at least $f(n)$. According to the *expansion criterion*, \mathcal{C} wins if $\alpha(G_{T+1})$ is at least $f(n) \cdot g(n)$. Otherwise, the matching

Cut-Matching Game $(\mathcal{G}(n), f(n), g(n))$:

- $G_1 := (V, \emptyset, 0)$ be an empty weighted graph, where $V = [n]$ and n is an even integer.
- Fix a cut player \mathcal{C} and a matching player \mathcal{M} .
- For $t = 1, \dots, T = g(n)$,
 1. \mathcal{C} chooses a bisection (S_t, \bar{S}_t) of V .
 2. \mathcal{M} chooses a perfect matching $M_t = (V, E(M_t))$ across (S_t, \bar{S}_t) .
 3. $G_{t+1} \leftarrow G_t + M_t$.
- Winning criteria:
 - **Gap criterion:** \mathcal{C} wins if $\text{gap}(G_{T+1}) \geq f(n)$. Otherwise, \mathcal{M} wins.
 - **Expansion criterion:** \mathcal{C} wins if $\alpha(G_{T+1}) \geq f(n) \cdot g(n)$. Otherwise, \mathcal{M} wins.

Figure 4.1: The Cut-Matching Game with two different winning criteria.

player \mathcal{M} is the winner. A summary of the game definition is given in Figure 4.1.

From the definition of the game, we can immediately observe that the graph G_t will be $(t-1)$ -regular and that $|E(G_t)| \leq (t-1) \cdot O(n)$, by the bound on the size of the edge set of each perfect matching. In particular, we have the following fact, which will be useful when discussing the running times of the cut strategies.

Fact 4.1.1. For $t \in [T+1]$,

$$|E(G_t)| \leq T \cdot O(n).$$

It is very important to notice the relation between the two winning criteria. If a cut player wins according to the gap criterion than it must also win by the expansion criterion as

$$\alpha(G_{T+1}) \geq \text{gap}(G_{T+1}) \cdot g(n) \geq f(n) \cdot g(n). \quad (4.1)$$

because G_{T+1} is $g(n)$ -regular. KRV introduced the expansion criterion; the stronger gap criterion was introduced in our work. While the expansion criterion suffices for the application to EXPANSION, the gap criterion allows a cleaner formulation of the game in terms of spectral quantities alone. Moreover, we will see how different strategies perform differently according to the two criteria and how we have different lower bounds for the two criteria.

Intuitively, the cut player wants to improve the connectivity properties of G_t , measured either by $\text{gap}(G_t)$ or $\alpha(G_t)$, by picking bisections containing few edges. Notice that by picking

a bisection (S, \bar{S}) , the cut player can ensure that the expansion of S and of all similar cuts is large in the next graph. Similarly, it practically enforces that random walks will mix quickly across the chosen bisection and similar ones. On the other hand, the matching player will try to place his perfect matching in a way that preserves either low expansion or the slow mixing of random walks on the graph. We will further explore the usage of random walks in the design of cut-player strategies for the Cut-Matching game in Section 4.2.

The importance of the Cut-Matching Game stems from its connection to the design of fast algorithms for the EXPANSION problem, which is captured by the following lemma of KRV [41].

Lemma 4.1.2 ([41]). *Consider an instance graph $G = (V, E)$ with $|V| = n$, $|E| = m$. Assume that there exists a winning cut-player strategy \mathcal{C} for $(G(n), f(n), g(n))$ under the expansion criterion and that this strategy runs in time $T(n)$ per round. Let $T_{\text{flow}} \stackrel{\text{def}}{=} \tilde{O}(m + n^{3/2})$. Then, there is an $O(1/f(n))$ -approximation algorithm for the EXPANSION problem on G . Moreover, the algorithm runs in time $\tilde{O}(g(n) \cdot (T(n) + T_{\text{flow}}))$.*

Notice that a winning cut-player strategy for $(G(n), f(n), g(n))$ under the gap criterion also ensures the existence of an approximation algorithm with the same parameters by Equation 4.1.

Here we give a sketch of the proof of Lemma 4.1.2. Suppose we are trying to decide whether the instance graph G has expansion larger than γ . Then, the idea behind this reduction is to let the matching player \mathcal{M} perform a single-commodity flow operation at every round. The flow computation attempts to “improve” the bisection given by the cut player, i.e. to find a cut of expansion less than γ that is well-correlated to the bisection. The flow operation either finds such cut, in which case the algorithm is successful and the game stops, or displays a certificate that no low-expansion cut exists “near” (S_t, \bar{S}_t) . The certificate takes the form of a perfect matching between (S_t, \bar{S}_t) that is routed with congestion $1/\gamma$ in the instance graph. If no cut is found after $g(n)$ rounds, the cut strategy guarantees that the union of the certificate matchings G_{T+1} has expansion $f(n) \cdot g(n)$. As G_{T+1} can be routed in G with congestion $g(n)/\gamma$, it must be the case that G has expansion at least $\gamma \cdot f(n)$. Hence, we can distinguish between $\alpha(G) < \gamma$ and $\alpha(G) \geq \gamma \cdot f(n)$, which implies a $1/f(n)$ -approximation algorithm for EXPANSION. Moreover, we only require $g(n)$ maxflow operations to solve this decision problem. Guessing the optimal value for the expansion by binary search only increases the running time by a logarithmic factor, so that the total running time is $\tilde{O}(g(n) \cdot (T(n) + T_{\text{flow}}))$.

From this discussion, it should be clear that we are ideally looking for cut-player strategies that achieve large $f(n)$ and small $g(n)$, as they yield better approximations in lower running time. In particular, all the strategies that we will discuss achieve $g(n) = O(\text{polylog}(n))$ and $f(n) = \Omega(1/\text{polylog}(n))$. Also, all the strategies that we present will be randomized and provably successful with high probability. The main result of KRV [41] is the existence of a cut strategy yielding $f(n) = \Omega(1/\log^2 n)$ and $g(n) = O(\log^2 n)$ under the expansion criterion. We will show that their strategy also yields a weaker guarantee for the gap criterion.

Theorem 4.1.3 (Extension of result in [41]). *There is a cut-player \mathcal{C}_{KRV} that runs in time $\tilde{O}(n)$ and with high probability wins the Cut-Matching game $(\mathcal{G}(n), \Omega(\frac{1}{\log^2 n}), O(\log^2 n))$ under the expansion criterion and the game $(\mathcal{G}(n), \Omega(\frac{1}{\log^4 n}), O(\log^2 n))$ under the gap criterion.*

In our paper [56], we improve on this result by giving two different strategies that yield a $O(\log n)$ -approximation for EXPANSION. The first strategy \mathcal{C}_{EXP} is based on the same ideas that are behind the Matrix MWU algorithm of Chapter 3 and achieves the same performance for both the gap and expansion criteria. We discuss its connection with the Matrix MWU algorithm in Section 4.6.

Theorem 4.1.4. *There is a cut strategy \mathcal{C}_{EXP} that runs in time $\tilde{O}(n)$ and with high probability wins the Cut-Matching game $(\mathcal{G}(n), \Omega(\frac{1}{\log n}), O(\log^2 n))$ under both the expansion and gap criterion.*

The strategy \mathcal{C}_{NAT} , while using similar ideas as \mathcal{C}_{EXP} , is computationally and intuitively simpler and can be seen as a hybrid between \mathcal{C}_{EXP} and \mathcal{C}_{KRV} . While this strategy yields a $O(\log n)$ -approximation to EXPANSION through its performance in the game under the expansion criterion, as for \mathcal{C}_{KRV} , we can only prove weaker bounds on its success under the gap criterion.

Theorem 4.1.5. *There is a cut strategy \mathcal{C}_{NAT} that runs in time $\tilde{O}(n)$ and with high probability wins the Cut-Matching game $(\mathcal{G}(n), \Omega(\frac{1}{\log n}), O(\log^2 n))$ under both the expansion criterion and the Cut-Matching game $(\mathcal{G}(n), \Omega(\frac{1}{\log^3 n}), O(\log^2 n))$.*

Our second main result regarding the Cut-Matching game is the first lower bound on the performance of any cut player under the expansion criterion. It implies that no approximation algorithm for EXPANSION designed following the Cut-Matching framework can achieve an approximation ratio better than $\Omega(\sqrt{\log n})$. Curiously, this is also the best approximation ratio known for any polynomial-time algorithm for EXPANSION [12].

Theorem 4.1.6. *There is a matching player \mathcal{M}^* that is successful against any cut player on the game $(\mathcal{G}(n), O(1/\sqrt{\log n}), g(n))$ for all $g(n)$ under the expansion criterion.*

This lower bound is tight, as there is a cut strategy, albeit an inefficient one, that is successful on the under the expansion criterion. This strategy requires the use of multi-commodity flows and is mentioned in various papers [8, 11, 59]. Hence, our lower bound settles the question of the power of the Cut-Matching game. Sherman [59] has showed that a matching player similar to that of Theorem 4.1.6 is successful on the game the game $(\mathcal{G}(n), O(\log \log n / \log n), g(n))$ for all $g(n)$ under the gap criterion. This is significant as cut strategies for the gap criterion tend to be computationally fast because they only use nearly-linear-time spectral routines. Hence, this theorem suggests that no nearly-linear-time strategies based on spectral ideas can achieve a better approximation than $O(\log n / \log \log n)$. In particular, \mathcal{C}_{EXP} is almost optimal in this class of strategies.

In the next section, we discuss how random-walks arise naturally when trying to use potential-reduction arguments to analyze cut strategies. In Section 4.3, we describe the random walks of interest and some of their fundamental properties. In Section 4.4, we present the geometric argument at the core of the potential-reduction analysis. Finally, in Section 4.5, we complete the proofs of the performance of each cut strategy. The remaining sections are dedicated to the proof of the lower-bound result of Theorem 4.1.6 and a brief survey of related work.

4.2 Construction of Cut Strategies

4.2.1 Using Random Walks

We motivate the use of random walks from the perspective of a cut player trying to win under the gap criterion. Similar ideas apply under the expansion criterion. We start by recalling the definition of spectral gap for a regular graph. Let \mathcal{S} be the set of unit vectors in the subspace of \mathbb{R}^n perpendicular to the vector $\vec{1}$, i.e. $\mathcal{S} \stackrel{\text{def}}{=} \{x \in \mathbb{R}^n : x^T \vec{1} = 0, \|x\|_2 = 1\}$. Then, for a d -regular graph $H = (V, E(H))$,

$$\text{gap}(H) = \frac{1}{d} \min_{x \in \mathcal{S}} x^T L(H)x.$$

As G_{T+1} is $g(n)$ -regular, the goal of the cut player is to ensure that $\min_{x \in \mathcal{S}} x^T L(G_t)x$ grows above $f(n)$ as t goes to $g(n)$. For a vector $x \in \mathcal{S}$, we call $x^T L(G)x$ the *mixing* of x , as this quantity describes how quickly the vector x approaches the uniform distribution under the natural random walk. Hence, the goal of the cut player is to ensure that all vectors $x \in \mathcal{S}$ have mixing greater than $f(n)$.

The following lemma shows that the cut player can enforce that a given vector $x \in \mathcal{S}$ have a large mixing by presenting a bisection obtained from a sweep cut of the vector x . Then, for any choice M of the matching player, $x^T L(M)x$ will be at least 1.

Lemma 4.2.1. *Given $x \in \mathcal{S}$, with $x_{i_1} \geq x_{i_2} \geq \dots \geq x_{i_n}$, let $S \subseteq [n]$ be defined as $S \stackrel{\text{def}}{=} \{i_1, i_2, \dots, i_{\frac{n}{2}}\}$. Then, for any perfect matching $M = (V, E(M))$ across (S, \bar{S}) , we have:*

$$x^T L(M)x \geq 1$$

Proof. Let $x_{i_{n/2}} = a$. Then, for any $\{i, j\} \in E(M)$, we must have without loss of generality $x_i \geq a, x_j < a$, so that

$$(x_i - x_j)^2 \geq (x_i - a)^2 + (x_j - a)^2.$$

Hence,

$$x^T L(M)x = \sum_{\{i,j\} \in E(M)} (x_i - x_j)^2 \geq \sum_{\{i,j\} \in E(M)} (x_i - a)^2 + (x_j - a)^2 \geq \sum_{i \in V} (x_i - a)^2 \geq \|x\|_2^2 = 1,$$

where the second inequality follows as M is 1-regular and the last inequality is a consequence of the fact that $x^T \vec{1} = 0$ as $x \in \mathcal{S}$. \square

In particular, if at round t , the cut player chooses bisection S_t based on the bisecting sweep of x , the mixing of x will be larger than $1 \gg f(n)$ from then on in the game and x will not constitute an obstacle to the cut player's success.

As the cut player wants to lower-bound $\text{gap}(G_T)$ when $T = g(n)$, a naive approach would be to use $\text{gap}(G_t)$ as a measure of progress and, at time t , return the bisection corresponding to the eigenvector v associated to the spectral gap, in the hope of increasing $\text{gap}(G_t)$ sufficiently. After the addition of M_t , v will have large mixing by Lemma 4.2.1. However, another eigenvector with associated eigenvalue just larger than $\text{gap}(G_t)$ may not have had its mixing improved by M_t at all and still be present in G_{t+1} , causing the spectral gap to be almost unchanged. Indeed, it is possible to construct examples when the cut strategy of playing the bisection given by the least mixing eigenvector requires $\Omega(n)$ rounds to produce any non-trivial guarantee on the gap.

Hence, in our choice of bisection, we must go beyond the greedy approach that just considers the eigenvector associated to the spectral gap, as, while very large progress is made on a single eigenvector, many iterations may be necessary to “fix” all eigenvectors. Instead, we must settle for a cut player that induces less progress on the mixing of each single eigenvector, but is able to raise the mixing of multiple eigenvectors at the same time. In this way, many of the low-mixing eigenvectors of $L(G_t)$ will increase by a sufficiently large amount. To apply this idea, we must choose a measure of progress different from the mixing of the lowest-mixing vector in \mathcal{S} and, in particular, one that incorporates information from possibly all the eigenvectors of G_t , while focusing on the progress of the lowest-mixing ones. To achieve this effect, we will use a potential Φ_t based on a measure of the mixing behavior of random walks that is more robust than $\text{gap}(G_t)$.

An abstract random-walk process P (such as the natural random walk or the heat-kernel random walk [21]) defines a sequence of probability-transition matrices $\{P_t \in \mathbb{R}^n\}_{t \in [T]}$, where P_t is the matrix corresponding to process P on graph G_t . Notice that, as P_t depends on G_t , it also depends on the first $(t - 1)$ choices of the matching player \mathcal{M} . We require that P_t have stationary distribution that is uniform over the vertices, i.e. $P_t \vec{1} = \vec{1}$. Note that, as G_t is regular, the stationary distribution is also uniform over the edges. For the purpose of the following discussion, the reader may find it simpler to let P_t represent a specific random-walk process, such as the natural random walk.

The spectral gap and the expansion are often used to study the mixing of a random walk in a regular graph, as it is usually possible to upper-bound the ℓ_2 -distance between the random-walk operator P_t and the projection $\Pi \stackrel{\text{def}}{=} 1/n \cdot \vec{1} \vec{1}^T$ onto the uniform distribution as a function of these quantities. For instance, if P_t equals $W(G_t)^k$, the natural random walk

for k steps, we have

$$\|P_t - \Pi\|_2^2 = (1 - \mathbf{gap}(G_t))^{2k} \geq \left(1 - \frac{\alpha(G_t)}{t-1}\right)^{2k},$$

where the second inequality follows by Cheeger's Inequality in Lemma 2.2.3. Hence, an upper bound on the left-hand side yields a lower bound on the spectral gap and the expansion. To obtain a potential that takes into account multiple eigenvectors and still bounds the spectral gap and the expansion, we consider the Frobenius norm [29], instead of the ℓ_2 -norm. As the Frobenius norm is at least the spectral ℓ_2 norm, an upper bound on $\|P_t - \Pi\|_F$ yields a lower bound on $\mathbf{gap}(G_t)$ and $\alpha(G_t)$. At the same time, the Frobenius norm, with its dependence on multiple eigenvectors and eigenvalues, fulfills our requirement for an improved measure of progress. Hence, given a choice for P_t , our potential function will be

$$\Phi_t \stackrel{\text{def}}{=} \|P_t - \Pi\|_F^2.$$

The choice of P_t will be fundamental, and the next section will be dedicated to it. For example, if $P_t = W(G_t)^k$, as k grows, the potential will capture the spectral gap more tightly, but will also begin to suffer of the same problem as the greedy strategy, namely it will not capture the mixing along vectors different from the second eigenvector. This problem will then make it impossible to relate Φ_{t+1} to Φ_t while making meaningful progress. Hence, the random walk must be chosen to ensure a trade-off between two important properties:

- tight connection with spectral gap or expansion, and
- ease of relating Φ_{t+1} to Φ_t and guaranteeing that the potential decreases at every round.

The next subsection will address which conditions are required of a random walk more in detail.

The potential Φ_t can be given a simple interpretation by noticing that

$$\Phi_t = \|P_t - \Pi\|_F^2 = \sum_{i \in V} \|P_t e_i - \frac{1}{n} \vec{1}\|_2^2;$$

hence, Φ_t is just the sum, over all vertices, of the ℓ_2^2 -distance between the distribution given by the walk P_t when started at a single vertex and the uniform distribution. The connection with the spectral gap of G_t can then be stated in random-walk language: if all the $P_t e_i$ random walks mix, then the spectral gap must be large. Equivalently, if the spectral gap were small, some random walks from single vertices would mix slowly and the potential would be high. Finally, we remark that Φ_t can also be rewritten as

$$\Phi_t = \text{Tr}(P_t^T P_t) - 1,$$

by noticing that $P_t \vec{1} = \vec{1}$.

This use of random walks in the design of cut strategies was championed by KRV. While their random walk was designed ad-hoc for ease of analysis in the Cut-Matching game, in our work we use more general and powerful random walks, such as the heat kernel, defined in Chapter 2, and develop a more sophisticated understanding of which properties are desirable in a random-walk process for it to apply to the design of cut strategies. Alternatively, in Section 4.6, we will also show how the problem of designing cut strategies can be approached using the Matrix MWU algorithm. Unfortunately, we are still unable to show that the natural random walk, the most obvious choice for this process, can be used in the construction of cut strategies: this stems from the difficulty of relating P_{t+1} to P_t .

4.2.2 Potential Analysis

In this subsection, we formalize the conditions under which a random-walk process will successfully yield a cut strategy through the potential analysis outlined above. Our goal is to show that our choice of potential, i.e. $\Phi_t = \text{Tr}(P_t^T P_t) - 1$ significantly decreases at every round for any choice of \mathcal{M} . To follow this direction, we will need our choice of random walk to obey the following conditions:

1. **Connection to spectral gap and expansion:** A upper bound on Φ_{T+1} implies a lower bound on $\text{gap}(G_{T+1})$ and $\alpha(G_{T+1})$.
2. **Round-by-round decomposition:** For all $t \in [T]$,

$$\Phi_{t+1} \leq \Phi_t - \eta_t \cdot \text{Tr}(P_t^T L(M_t) P_t),$$

where η_t is some coefficient, usually a constant, depending on the choice of random walk. Hence, the expression $\text{Tr}(P_t^T L(M_t) P_t)$ will regulate the potential reduction at every step.

3. **Bisection choice:** Given P_t , it is possible to choose bisection (S, \bar{S}) such that, for any perfect matching M , which is bipartite across (S, \bar{S}) ,

$$\text{Tr}(P_t^T L(M_t) P_t) \geq O\left(\frac{\Phi_t}{\log n}\right).$$

Conditions 2 and 3 guarantee that the potential decreases at every round by a multiplicative factor of $1 - \Omega(1/\log n)$. After T steps, this reduction will yield an upper bound on Φ_{T+1} , which, combined with condition 1, will complete the proof of the performance of the cut player.

While the first two conditions will be proved separately for each different random walk, the procedure allowing us to fulfill condition 3 will be the same. We describe it in Section 4.4. In the next section, we define the random walks of interest and prove lemmata about their connection to spectral gap and expansion and about their round-to-round decomposition. Section 4.5 will be dedicated to completing the analysis for each cut strategy.

4.3 The Random Walks

In this subsection, we introduce the random-walk processes that we will be using in our construction of cut strategies. For each random walk, we will prove a decomposition lemma, which will allow us to relate Φ_{t+1} and Φ_t in our analysis, and a lemma connecting Φ_{T+1} to the expansion and the spectral gap of G_{T+1} . We will first present the sequential random walk of \mathcal{C}_{KRV} [41]. Then, we will define the random walks used by \mathcal{C}_{EXP} and by \mathcal{C}_{NAT} .

4.3.1 The Random Walk for \mathcal{C}_{KRV}

The random walk of KRV on G_t is just a sequential composition of the lazy random walks across each matching M_1, \dots, M_{t-1} . More formally, for $t \in [T]$, given G_t which is the sum of bipartite matchings M_1, \dots, M_{t-1} , we define:

$$P_{t+1}^{\text{KRV}} \stackrel{\text{def}}{=} \left(\frac{I + W(M_t)}{2} \right) \cdots \left(\frac{I + W(M_1)}{2} \right) = \prod_{i=t}^1 \left(\frac{I + W(M_i)}{2} \right),$$

and

$$P_1^{\text{KRV}} = I.$$

Notice that the definition of the walk depends strongly on the partition in perfect matchings of the edges of G_{t+1} and on the order in which these matchings arrived. In the rest of this subsection, let P_t denote P_t^{KRV} .

Connection with spectral gap and expansion

We start with proving the connection between the potential function Φ_t , based on P_t , and both $\text{gap}(G_t)$ and $\alpha(G_t)$. The main idea in the following lemma is the fact that we can embed the weighted graph with adjacency matrix P_{T+1} into G_{T+1} by using the paths defined by the random walk over matchings described by P_{T+1} . We will then apply the embedding results of Lemma 2.2.4 to yield the desired relations.

Lemma 4.3.1. *Consider the 1-regular weighted graph H with adjacency matrix P_{T+1} . The following two inequalities hold for $T \geq 1$:*

$$\begin{aligned} L(G_{T+1}) &\succeq \frac{1}{T} L(H) = \frac{1}{T} (I - P_{T+1}), \\ \alpha(G_{T+1}) &\geq \alpha(H) \geq \lambda_2(I - P_{T+1}) \end{aligned}$$

Proof. Consider the embedding of H given by routing each edge following the paths defined by the definition of P_{T+1} as

$$\prod_{i=T}^1 \left(\frac{I + W(M_i)}{2} \right).$$

The congestion of this embedding is 1 and its dilation is T . Hence, Lemma 2.2.4 together with Cheeger's Inequality, gives the required inequalities. \square

Decomposition lemma for P^{KRV}

The main advantage of the definition of P^{KRV} is the ease of relating P_{t+1} to P_t . Indeed, the following decomposition lemma regarding the potential function is a simple consequence of the walk definition.

Lemma 4.3.2.

$$\text{Tr}(P_{t+1}^T P_{t+1}) = \text{Tr}(P_t^T P_t) - 1/2 \cdot \text{Tr}(P_t^T L(M_t) P_t)$$

Proof.

$$\text{Tr}(P_{t+1}^T P_{t+1}) = \text{Tr}\left(P_t^T \left(\frac{I + W(M_t)}{2}\right)^2 P_t\right) = \text{Tr}\left(P_t^T \left(I - \frac{L(M_t)}{2}\right)^2 P_t\right).$$

We also have $L(M_t) \preceq 2I$, as M_t is 1-regular. Hence $L(M_t)^2 \preceq 2L(M_t)$, so that

$$\left(I - \frac{L(M_t)}{2}\right)^2 \preceq I - L(M_t) + \frac{L(M_t)}{2} = I - \frac{L(M_t)}{2}.$$

By Fact 2.3.6, we have the required result. \square

4.3.2 The Random Walk for \mathcal{C}_{EXP}

The cut player \mathcal{C}_{EXP} makes use of the heat kernel random walk defined in Chapter 2. The rate of the walk at time t is $t - 1$, meaning that this random walk can be seen as performing the natural random walk on G_t for a number of steps which is Poisson distributed with rate $t - 1$. Hence, we can write:

$$P_t^{\text{EXP}} = e^{-L(G_t)}.$$

Notice that $P_1^{\text{KRV}} = I$. This walk enjoys two useful properties; first, $\text{gap}(G_{T+1})$ will be lower bounded by definition by an explicit function of Φ_{T+1} , without recurring to any embedding. Secondly, Φ_{t+1} will be connected to Φ_t by using the Golden-Thompson Inequality of Lemma 2.3.8. For the rest of this subsection, we let P_t denote P_t^{EXP} .

Connection with spectral gap and expansion

Lemma 4.3.3. *For $T \geq 1$,*

$$\begin{aligned} \text{gap}(G_{T+1}) &\geq -\frac{\log(\Phi_{T+1})}{2T}, \\ \alpha(G_{T+1}) &\geq -\frac{\log(\Phi_{T+1})}{2}. \end{aligned}$$

Proof. It suffices to prove the first inequality, as it immediately implies the second. By definition, we have

$$P_{T+1} = e^{-L(G_{T+1})}.$$

By the properties of the matrix exponential in Chapter 2, we have

$$\lambda_{n+1-i}(e^{-L(G_{T+1})}) = e^{-\lambda_i(L(G_{T+1}))},$$

so that, as G_{T+1} is T -regular, we have

$$\text{gap}(G_{T+1}) = -\frac{1}{T} \cdot \log(\lambda_{n-1}(P_{T+1})).$$

As $\vec{1}$ is an eigenvector of P_{T+1} associated with the largest eigenvalue 1, it is clear that

$$(\lambda_{n-1}(P_{T+1}))^2 = \|P_{T+1} - \Pi\|_2^2 \leq \|P_{T+1} - \Pi\|_F^2 = \Phi_{T+1}.$$

Hence,

$$\text{gap}(G_{T+1}) = -\frac{\log(\lambda_{n-1}(P_{T+1}))}{T} \geq -\frac{\log(\sqrt{\Phi_{T+1}})}{T} = -\frac{\log(\Phi_{T+1})}{2T}.$$

□

Decomposition lemma for P_t^{EXP}

The main reason for the choice of the heat-kernel random walk is the Golden-Thompson Inequality of Lemma 2.3.8. Applied to P_{t+1} , it yields the following lemma.

Lemma 4.3.4.

$$\text{Tr}(P_{t+1}^T P_{t+1}) \leq \text{Tr}(P_t^T P_t) - \frac{(1 - e^{-4})}{4} \cdot \text{Tr}(P_t^T L(M_t) P_t).$$

Proof.

$$\text{Tr}(P_{t+1}^T P_{t+1}) = \text{Tr}(e^{-2L(G_{t+1})}) = \text{Tr}(e^{-2L(G_t)} e^{-2L(M_t)}) = \text{Tr}(P_t^T e^{-2L(M_t)} P_t).$$

We can now use Lemma 2.3.3 and Fact 2.3.6 to obtain the required inequality. □

4.3.3 The Random Walk for \mathcal{C}_{NAT}

Like P^{KRV} , the random walk P_{NAT} used by \mathcal{C}_{NAT} also depends on the decomposition of G_t into matchings, but it preserves properties of the heat kernel that make it more powerful in the context of the Cut-Matching game. It is a round-robin random walk that can be seen as a hybrid of the natural random walk and the heat kernel.

For the definition of P_{NAT} , we assume that the duration of the game $T = g(n)$ is a power of 2, i.e. $T = 2^k$, for some $k \geq 1$. Also, let $N_t \stackrel{\text{def}}{=} \frac{T-1}{T}I + \frac{1}{T}W(M_t)$. The probability-transition matrix P_t^{NAT} is defined recursively as follows:

- Let $Q_1 = I$ and $P_1^{\text{NAT}} := I$.
- For $1 < t \leq T$, define

$$Q_{t+1} \stackrel{\text{def}}{=} N_t Q_t N_t,$$

and

$$P_{t+1}^{\text{NAT}} \stackrel{\text{def}}{=} (Q_{T+1})^{T/4}$$

In other words, N_t is a lazy random walk across M_t with staying probability T^{-1}/T and

$$P_{t+1}^{\text{NAT}} = (N_t N_{t-1} \dots N_1 N_1 \dots N_{t-1} N_t)^{T/4}.$$

The main motivation behind the design of this random walk was to realize a construction closer to the natural random walk that still proved effective in the Cut-Matching game. P^{NAT} can be seen as an analogue of lazy natural random walk where, rather than sending a $1/T$ fraction of probability mass across every incident edge at once, the same fraction is sent across edges in M_{t-1} first, then M_{t-2} and so on, wrapping around in a round-robin fashion after reaching M_1 for the first time. In the rest of this subsection, we denote P_t^{NAT} by P_t .

Connection with spectral gap

The following analysis is different from both the analysis of \mathcal{C}_{KRV} and that of \mathcal{C}_{EXP} . For \mathcal{C}_{KRV} , we used an argument based on an embedding with bounded congestion and dilation, while for \mathcal{C}_{EXP} we relied on a stronger algebraic relation between the heat-kernel random walk and the spectral gap. In this case, we use a combination of these two methods. We will consider the 1-regular weighted graph R_{T+1} with adjacency matrix

$$A(R_{T+1}) \stackrel{\text{def}}{=} Q_{T+1} = (N_T N_{T-1} \dots N_1 N_1 \dots N_{T-1} N_T),$$

and embed R_{T+1} (and not the walk P_{T+1} itself) in G_{T+1} to show a connection between the two graphs. Then, we will then use the bound on $\text{gap}(R_{T+1})$ given by Φ_{T+1} to show a lower bound on the spectral gap and expansion of the graph G_{T+1} .

Lemma 4.3.5.

$$\begin{aligned} \text{gap}(G_{T+1}) &\geq \frac{1}{4T} \left(1 - (\Phi_{T+1})^{\frac{2}{T}}\right), \\ \alpha(G_{T+1}) &\geq \frac{T}{2} \left(1 - (\Phi_{T+1})^{\frac{2}{T}}\right), \end{aligned}$$

Proof. Notice that, by the definition of P_{T+1}

$$\Phi_{T+1} \geq \|P_{T+1} - \Pi\|_2^2 = \lambda_{n-1}(P_{T+1}^T P_{T+1}) = (1 - \text{gap}(R_{T+1}))^{T/2}.$$

Hence, we have

$$\text{gap}(R_{T+1}) \geq 1 - (\Phi_{T+1})^{\frac{2}{T}}. \quad (4.2)$$

Consider now the embedding of R_{T+1} in G_{T+1} that follows the paths defined by the product

$$N_T N_{T-1} \dots N_1 N_1 \dots N_{T-1} N_T.$$

This embedding has congestion $2/T$ by the definition of N_t and dilation $2 \cdot T$. Hence, by Lemma 2.2.4,

$$\begin{aligned} L(G_{d+1}) &\preceq \frac{1}{4} \cdot L(R_{d+1}), \\ \alpha(G_{d+1}) &\geq \frac{T}{2} \cdot \alpha(R_{d+1}). \end{aligned}$$

Combining this with Equation 4.2 and using the fact that G_{T+1} is T -regular yields the first required inequality. Moreover, as $\alpha(R_{T+1}) \geq \text{gap}(R_{T+1})$, the second part also follows. \square

Decomposition lemma for P^{NAT} .

The use of a round-robin ordering is motivated by the rearrangement inequality in Theorem 2.3.9, which allows us to prove the following lemma about decomposing the potential for the P^{NAT} walk.

Lemma 4.3.6.

$$\text{Tr}(P_{t+1}^T P_{t+1}) \leq \text{Tr}(P_t^T P_t) - \frac{(1 - e^{-2})}{2} \cdot \text{Tr}(P_t^T L(M_t) P_t). \quad (4.3)$$

Proof. Applying Theorem 2.3.9 and Fact 2.3.5:

$$\begin{aligned} \text{Tr}(P_{t+1}^T P_{t+1}) &= \text{Tr}\left((N_t Q_t N_t)^{T/2}\right) \leq \text{Tr}\left(N_t^{T/2} Q_t^{T/2} N_t^{T/2}\right) \leq \\ &\quad \text{Tr}\left(Q_t^{T/2} N_t^T\right). \end{aligned}$$

Now, notice that $N_t = I - 1/T \cdot L(M_t)$ as M_t is 1-regular. Hence:

$$N_t^T = \left(I - \frac{1}{T} \cdot L(M_t)\right)^T \preceq e^{-L(M_t)} \preceq \left(I - \frac{(1 - e^{-2})}{2} L(M_t)\right)$$

where the first inequality is a consequence of the fact that $(1 + a/x)^x \leq e^a$, for $a \in \mathbb{R}$, $x > 0$ and the second follows from Lemma 2.3.3. The lemma is a consequence of these two last inequalities together with Fact 2.3.6. \square

In Section 4.5, we will discuss how P^{NAT} can be cast as an approximation to the heat-kernel random walk, which partially justifies why the decomposition of Lemma 4.3.6 is possible.

4.4 Analyzing the Potential Reduction

In this subsection, we describe how the cut player achieves condition 3 of Section 4.2.2 by giving a subroutine that outputs a bisection (S, \bar{S}) given graph G_t and random walk process P_t . The same subroutine was used in the work of KRV.

The cut player must output a bisection (S, \bar{S}) such that, whichever response M_t the matching player chooses, we have

$$\text{Tr}(P_t^T L(M) P_t) = \Omega\left(\frac{\Phi_t}{\log n}\right).$$

Following [41], we take a geometric view of this problem by considering the vectors

$$v_i = P_t^T e_i - \frac{\vec{1}}{n} \in \mathbb{R}^n$$

for $i \in V$. Notice that

$$\sum_{i \in V} v_i = P_t^T \vec{1} - \vec{1} = 0.$$

Hence, the mean of the vectors $\{v_i\}$ is 0. Moreover, we have that

$$\Phi_t = \text{Tr}(P_t^T P_t) - 1 = \|P_t - \Pi\|_F^2 = \sum_{i \in V} \|v_i\|_2^2, \quad (4.4)$$

showing that Φ_t is the total variance of the embedding $\{v_i \in \mathbb{R}^n\}_{i \in V}$. At the same time, let us also consider a geometric interpretation of the potential reduction:

$$\text{Tr}(P_t^T L(M_t) P_t) = \sum_{i \in V} (P_t e_i)^T L(M_t) (P_t e_i) = \sum_{\{h,k\} \in E(M_t)} \|v_h - v_k\|_2^2.$$

Hence, it suffices for the cut player to find a bisection (S, \bar{S}) such that, for all matchings across (S, \bar{S}) , the distance between the vectors corresponding to matched vertices is a $\Omega(1/\log n)$ -fraction of the total variance. This can be achieved using random projections as follows. Noticing that all the vectors in $\{v_i\}_{i \in V}$ live in the subspace of \mathbb{R}^n perpendicular to the vector $\vec{1}$, let r be a random uniformly distributed unit vector in that $n - 1$ -dimensional subspace and consider the projection u of the embedding $\{v_i\}_{i \in V}$ onto r defined for all $i \in V$ as:

$$u_i \stackrel{\text{def}}{=} v_i^T r = e_i^T P_t r - 1/n r^T \mathbf{1} = e_i^T P_t r.$$

The second term disappears as $r^T \mathbf{1} = 0$, by assumption. Then, the cut player returns the bisecting sweep cut of vector u as the cut (S_t, \bar{S}_t) . This algorithm is based on the hope that vectors which are far apart when projected onto r may also be far apart before projection and hence contribute a large quantity to the potential reduction when matched. This intuition is formalized by the following lemma, which is based on the Gaussian nature of projections and appears in the same form in [41]. Our proof is given in Section A.1.

Lemma 4.4.1. *Let $\{v_i\}_{i=1}^n$ be vectors in \mathbb{R}^{n-1} such that $\sum_i v_i = 0$. Let $\Phi \stackrel{\text{def}}{=} \sum \|v_i\|^2$. Let r be a random uniform unit vector in \mathbb{R}^{n-1} and, for all i , set $u_i := v_i^T r$. Let S be the partition of $[n]$ such $|S| = n/2$ and, for all $i \in S$ and $j \in \bar{S}$, $u_i \geq u_j$. Consider any perfect matching M across (S, \bar{S}) . Then,*

$$\mathbb{E}_r \left[\sum_{\{i,j\} \in E(M)} \|v_i - v_j\|^2 \right] = \Omega \left(\frac{\Phi}{\log n} \right).$$

4.4.1 Algorithmic Template

We can now give an explicit algorithmic formulation of the cut strategies that we will be analyzing. At rounds t , given graph G_t which is the union of perfect matchings M_1, \dots, M_{t-1} , and a probability-transition matrix P_t on G_t , the cut player runs the following procedure to find which bisection to output:

1. Sample an uniformly distributed unit vector r_t in the subspace of perpendicular to $\vec{1}$.
2. Compute $y_t := P_t r_t$.
3. Sort the entries of $y_t = (y_1, \dots, y_n)$ as $y_{i_1} \leq \dots \leq y_{i_{n/2}} \leq y_{i_{n/2+1}} \leq \dots \leq y_{i_n}$.
4. Let $S_t := \{i_1, \dots, i_{n/2}\}$ and $\bar{S}_t := V \setminus S_t$.

From a combinatorial rather than geometric point of view, the reason this procedure finds a bisection containing a non-expanding cut is the following: Suppose (S, \bar{S}) is a cut across which there are very few edges in G_t , say none. Then the walk procedure P_t never transfers any charge across this cut. Also, the initial random vector will create a $\Theta(1/\sqrt{n})$ charge differential between S and \bar{S} , i.e $|\sum_{i \in S} (r_t)_i - \sum_{i \in \bar{S}} (r_t)_i| = \Theta(1/\sqrt{n})$ with high probability. Hence, in the bisection output after mixing and sorting based on y_t , the two sides of the bisection will have non-trivial correlations with S and \bar{S} respectively. Thus, the matching added in the next iteration will add some edges across the sparse cut (S, \bar{S}) . It is remarkable that the strategy $\mathcal{C}_{\text{KRV}}, \mathcal{C}_{\text{EXP}}$ and \mathcal{C}_{NAT} are able to ensure high expansion with high probability after only $O(\log^2 n)$ such simple steps.

4.5 Completing the Analysis

4.5.1 The \mathcal{C}_{KRV} Strategy

In this subsection, we complete the proof of Theorem 4.1.3. For the rest of this subsection, we denote P_t^{KRV} by P_t .

We are now ready to combine the connection with gap and expansion of Lemma 4.3.1, the decomposition result of Lemma 4.3.2 and the potential reduction guaranteed of Lemma 4.4.1 to prove Theorem 4.1.3.

Proof of Theorem 4.1.3. Recall that $\Phi_t = \text{Tr}(P_t^T P_t) - 1$ and that $T = g(n)$. Then, for all $t \in [T]$, by Lemma 4.3.2

$$\Phi_{t+1} = \Phi_t - \frac{1}{2} \text{Tr}(P_t^T L(M_t) P_t).$$

Define the vectors

$$v_i = P_t^T e_i - \frac{1}{n} \vec{1}.$$

Notice that, as $L(M_t) \vec{1} = 0$,

$$\text{Tr}(P_t^T L(M_t) P_t) = \sum_{\{i,j\} \in E(M_t)} \|v_i - v_j\|^2,$$

and, by Equation 4.4, $\Phi_t = \sum_{i \in V} \|v_i\|^2$. Applying Lemma 4.4.1, we obtain

$$\mathbb{E}_{r_t} [\text{Tr}(P_t^T L(M_t) P_t)] \geq \Omega\left(\frac{\Phi_t}{\log n}\right).$$

The last equation implies that, in expectation, the potential decreases by at least a $(1 - \Omega(1/\log n))$ -fraction at each iteration.

Hence, after all T rounds, we have

$$\mathbb{E}_{r_1, \dots, r_T} [\Phi_{T+1}] \leq (1 - \Omega(1/\log n)) \mathbb{E}_{r_1, \dots, r_{T-1}} [\Phi_T] \leq \dots \leq (1 - \Omega(1/\log n))^T \Phi_1.$$

But $\Phi_1 = \text{Tr}(P_1^T P_1) - 1 = \text{Tr}(I) - 1 = n - 1$, so that

$$\mathbb{E}_{r_1, \dots, r_T} [\Phi_{T+1}] \leq (n - 1) (1 - \Omega(1/\log n))^T.$$

Taking $T = O(\log^2 n)$, we can have

$$\mathbb{E}_{r_1, \dots, r_T} [\Phi_{T+1}] \leq \frac{1}{2n},$$

and, by Markov's Inequality, with high probability we obtain that

$$\|P_{T+1} - \Pi\|_F^2 = \Phi_{T+1} \leq \frac{1}{2}.$$

Finally, this means that

$$\|P_{T+1} - \Pi\|_2^2 \leq \|P_{T+1} - \Pi\|_F^2 \leq \frac{1}{2},$$

and

$$P_{T+1} \preceq \frac{1}{2}(I - \Pi) + \Pi = \frac{1}{2}(I + \Pi)$$

Combining with Lemma 4.3.1, we get

$$L(G_{T+1}) \succeq \frac{1}{T}(I - P_{T+1}) \succeq \frac{1}{2T}(I - \Pi),$$

and

$$\alpha(G_{T+1}) \geq \lambda_2(I - P_{T+1}) \geq \frac{1}{2}. \quad (4.5)$$

The first inequality immediately implies that

$$\text{gap}(G_{T+1}) \geq \frac{1}{2T^2}.$$

as G_{T+1} is T -regular. Hence, \mathcal{C}_{KRV} with high probability achieves a gap of $1/2T^2$ and an expansion of $1/2$ in $T = O(\log^2 n)$ rounds, implying that it is successful for the game $(\mathcal{G}(n), \Omega(1/\log^4 n), O(\log^2 n))$ under the gap criterion and for the $(\mathcal{G}(n), \Omega(1/\log^2 n), O(\log^2 n))$ -game under the expansion criterion. The running time required at each round by the cut strategy is $\tilde{O}(n)$ as it just needs to compute at most $T = O(\log^2 n)$ matrix-vector multiplications, where each matrix has at most $\tilde{O}(n)$ non-zero entries as it represents a perfect matching. \square

Discussion and comparison

In the above proof, \mathcal{C}_{KRV} can only guarantee constant edge expansion at termination, as seen in Equation 4.5. Here, we provide some intuition of why \mathcal{C}_{KRV} may fail to achieve better expansion. Suppose the union of matchings (M_1, \dots, M_{t-1}) , has no edges crossing some bisection (S, \bar{S}) . Now suppose that the walk P_t^{KRV} on matchings M_1, \dots, M_{t-1} mixes the probability charge on each side of (S, \bar{S}) perfectly (or very well). The next cut selected by \mathcal{C}_{KRV} is necessarily (S, \bar{S}) . Moreover, once any perfect matching M_t is added across (S, \bar{S}) , P_{t+1}^{KRV} mixes across that perfect matching in its last step and the random walk distribution becomes very close to stationary. At this point, we can have Φ_{t+1} arbitrarily small, yet the edge expansion across bisection (S, \bar{S}) is 1. This suggests considering a different walk, which is either lazier than P^{KRV} at every step or utilizes the matchings in a different order. We will see that P^{EXP} and P^{NAT} display both these characteristics.

As \mathcal{C}_{EXP} (and to some extent \mathcal{C}_{NAT}) makes use of the heat kernel random walk, it is worthwhile pointing out that \mathcal{C}_{KRV} can also be seen as using matrix exponentials. In particular, it is not difficult to show that $e^{-\eta L(M)}$ represents a slowed-down lazy random walk along matching M . As $\left(\frac{I+W(M)}{2}\right)^i = \frac{(I+W(M))^i}{2^i}$ for $i \geq 1$, the Taylor Series of the exponential yields: $e^{-\eta L(M)} = \frac{1+e^{-\eta}}{2}I + \frac{1-e^{-\eta}}{2}M$. Hence, the \mathcal{C}_{KRV} mixing procedure is close to the walk

$$e^{-\eta L(M_t)} e^{-\eta L(M_{t-1})} \dots e^{-\eta L(M_1)}$$

for small η . Our second cut-finding procedure \mathcal{C}_{EXP} uses the probability-transition matrix $e^{-\eta(tI - (M_1 + \dots + M_t))}$. If it were true that

$$e^{-\eta L(M_1 + \dots + M_t)} \approx e^{-\eta L(M_t)} e^{-\eta L(M_{t-1})} \dots e^{-\eta L(M_1)},$$

then it would establish that the two strategies are almost the same. In fact, even under the weaker condition that

$$\text{Tr} [e^{-\eta L(M_1 + \dots + M_t)}] \leq \text{Tr} [e^{-\eta L(M_t)} \dots e^{-\eta L(M_1)}]$$

we could extend the analysis of \mathcal{C}_{EXP} to \mathcal{C}_{KRV} and improve \mathcal{C}_{KRV} 's performance guarantee. Unfortunately, this is generally false, as the Golden-Thompson Inequality relies on the cyclical property of the trace function and only applies to two matrices.

4.5.2 The \mathcal{C}_{EXP} Strategy

We now complete the proof of our main theorem about the player \mathcal{C}_{EXP} . For the rest of this subsection, we denote P_t^{EXP} by P_t .

Proof of Theorem 4.1.4. Recall that $\Phi_t = \text{Tr}(P_t^T P_t) - 1$ and that $T = g(n)$. For all $t \in [T]$, by the decomposition result of Lemma 4.3.4, we have

$$\Phi_{t+1} = \Phi_t - \frac{(1 - e^{-4})}{4} \cdot \text{Tr} (P_t^T L(M_t) P_t).$$

As in the proof of Theorem 4.1.3, we apply Lemma 4.4.1 to vectors

$$v_i = P_t^T e_i - \frac{1}{n} \vec{1},$$

to obtain

$$\mathbb{E}_{r_t} [\text{Tr}(P_t^T L(M_t) P_t)] = \mathbb{E}_{r_t} \left[\sum_{\{i,j\} \in E(M_t)} \|v_i - v_j\|^2 \right] \geq \Omega \left(\frac{\Phi_t}{\log n} \right).$$

Hence, the potential decreases in expectation by a $(1 - \Omega(1/\log n))$ -fraction at every round. After T rounds, we have

$$\mathbb{E}_{r_1, \dots, r_T} [\Phi_{T+1}] \leq (1 - \Omega(1/\log n)) \mathbb{E}_{r_1, \dots, r_{T-1}} [\Phi_T] \leq \dots \leq (1 - \Omega(1/\log n))^T \Phi_1.$$

As $P_1 = I$, this implies that

$$\mathbb{E}_{r_1, \dots, r_T} [\Phi_{T+1}] \leq (n - 1) (1 - \Omega(1/\log n))^T.$$

We take $T = O(\log^2 n)$ to be large enough that

$$\mathbb{E}_{r_1, \dots, r_T} [\Phi_{T+1}] \leq \frac{1}{n^2}.$$

Hence, with high probability, by Markov's Inequality

$$\mathbb{E}_{r_1, \dots, r_T} [\Phi_{T+1}] \leq \frac{1}{n}.$$

Finally, by Lemma 4.3.3, we obtain

$$\text{gap}(G_{t+1}) \geq -\frac{\log(\Phi_{T+1})}{2T} \geq \Omega\left(\frac{1}{\log n}\right)$$

and

$$\alpha(G_{t+1}) \geq -\frac{\log(\Phi_{T+1})}{2} \geq \Omega(\log n).$$

Hence, \mathcal{C}_{EXP} is successful at the $(\mathcal{G}(n), \Omega(1/\log n), O(\log^2 n))$ -game for both the gap criterion and the expansion criterion. We complete the proof by analyzing the running time necessary to compute $y_t = P_t r_t$ at every round t , in the next subsection. \square

Running time

We approximate the exponential $e^{-L(G_t)}$ by truncating its Taylor series as is also done in Arora and Kale [11]. All we need to compute is $e^{-L(G_t)}u$ for some random unit vector u . We define the approximation v_k as

$$v_k := \sum_{j=0}^k \frac{1}{j!} (L(G_t))^j u.$$

For v_k to be a good approximation for the purposes of the algorithm it suffices to have

$$\|v_k - e^{-L(G_t)}u\|^2 \leq O\left(\frac{\Phi_t}{\log n}\right). \quad (4.6)$$

The error occurred by the truncation can be written as

$$\|v_k - e^{-L(G_t)}u\| \leq \sum_{j=k+1}^{\infty} \frac{1}{j!} \|L(G_t)\|^j.$$

By a standard approximation of the factorial, it is straightforward to show that, for

$$k \geq \max\{\Omega(t), \Omega(\log n)\},$$

Equation 4.6 is satisfied. Finally, to compute v_k we just need to perform k matrix-vector multiplications. Each of these takes time $O(tn)$ as G_t is $t - 1$ regular. As $t \leq O(\log^2 n)$ and by the bound on k , we get a running time of $O(n \log^4 n) = \tilde{O}(n)$ for running the strategy \mathcal{C}_{exp} for one round. The survey by Golub and Van Loan [29] describes other ways of computing the exponential efficiently that may be useful in practice. In Section 4.5, we will see a relation between one of these methods, that of “splitting”, and the \mathcal{C}_{NAT} random walk.

Discussion and comparison

The strong connection between Φ_{T+1} and $\text{gap}(G_{T+1})$, which arises from properties of the matrix exponential, allows \mathcal{C}_{EXP} to achieve the same performance under both criteria. In particular, \mathcal{C}_{EXP} does not need to recur to the embedding results of Lemma 2.2.4, which cause \mathcal{C}_{KRV} and \mathcal{C}_{NAT} to obtain only weaker spectral results because of the $\Omega(T)$ -dilation incurred by these embeddings.

In contrast to \mathcal{C}_{KRV} , \mathcal{C}_{EXP} is able to obtain a final expansion of $\alpha(G_{T+1}) \geq \Omega(\log n)$. The obstacle described above for \mathcal{C}_{KRV} does not present itself in this case as heat kernel walk does not mix fully across a newly-matched bisection. Hence, the laziness of the walk potentially allows multiple matchings to be added to each bisection. Intuitively, a lazier version of the \mathcal{C}_{KRV} strategy, where the random walk with staying probability $1/2$ is replaced by one with higher staying probability, may also achieve a better expansion. However, to date, we are still unable to prove or disprove this conjecture.

Finally, in Section 4.6, we discuss how the \mathcal{C}_{EXP} strategy is equivalent to a strategy that can be designed implicitly using the Matrix MWU algorithm of Chapter 3.

4.5.3 The \mathcal{C}_{NAT} Strategy

For the rest of this subsection, we denote P_t^{NAT} by P_t . Recall that

$$N_t = \frac{T-1}{T}I + \frac{1}{T}M_t.$$

We now complete the proof of the main theorem about the player \mathcal{C}_{NAT} .

Proof of Theorem 4.1.5. Recall that $\Phi_t = \text{Tr}(P_t^T P_t) - 1$ and that $T = g(n)$. For all $t \in [T]$, by the decomposition result of Lemma 4.3.6, we have

$$\Phi_{t+1} = \Phi_t - \frac{(1 - e^{-2})}{2} \cdot \text{Tr}(P_t^T L(M_t) P_t).$$

As in the proof of Theorem 4.1.3, we apply Lemma 4.4.1 to vectors

$$v_i = P_t^T e_i - \frac{1}{n} \vec{1},$$

to obtain

$$\mathbb{E}_{r_t} [\text{Tr}(P_t^T L(M_t) P_t)] = \mathbb{E}_{r_t} \left[\sum_{\{i,j\} \in E(M_t)} \|v_i - v_j\|^2 \right] \geq \Omega \left(\frac{\Phi_t}{\log n} \right).$$

Hence, the potential decreases in expectation by a $(1 - \Omega(1/\log n))$ -fraction at every round. After T rounds, we have

$$\mathbb{E}_{r_1, \dots, r_T} [\Phi_{T+1}] \leq (1 - \Omega(1/\log n)) \mathbb{E}_{r_1, \dots, r_{T-1}} [\Phi_T] \leq \dots \leq (1 - \Omega(1/\log n))^T \Phi_1.$$

As $P_1 = I$, this implies that

$$\mathbb{E}_{r_1, \dots, r_T} [\Phi_{T+1}] \leq (n-1) (1 - \Omega(1/\log n))^T.$$

We take $T = O(\log^2 n)$ to be large enough that

$$\mathbb{E}_{r_1, \dots, r_T} [\Phi_{T+1}] \leq \frac{1}{n^2}.$$

Hence, with high probability, by Markov's Inequality

$$\Phi_{T+1} \leq \frac{1}{n}.$$

Finally, by Lemma 4.3.5, we obtain

$$\text{gap}(G_{T+1}) = \text{gap}(G_{d+1}) \geq \frac{1}{4T} \left(1 - \left(\frac{1}{n} \right)^{\frac{2}{T}} \right)$$

and

$$\alpha(G_{T+1}) = \alpha(G_{d+1}) \geq \frac{T}{2} \left(1 - \left(\frac{1}{n} \right)^{\frac{2}{T}} \right).$$

But $T = O(\log^2 n)$, so that we have

$$\left(1 - \left(\frac{1}{n} \right)^{\frac{2}{T}} \right) = \left(1 - e^{-\Omega(\frac{1}{\log n})} \right) = (1 - (1 - \Omega(1/\log n))) \geq \Omega \left(\frac{1}{\log n} \right).$$

Hence,

$$\text{gap}(G_{T+1}) \geq \Omega \left(\frac{1}{\log^3 n} \right),$$

and

$$\alpha(G_{T+1}) \geq \Omega(\log n).$$

Therefore, \mathcal{C}_{NAT} is successful at the $(\mathcal{G}(n), \Omega(1/\log n), O(\log^2 n))$ -game under the expansion criterion and at the $(\mathcal{G}(n), \Omega(1/\log^3 n), O(\log^2 n))$ under the gap criterion.

Turning to the running time, note that we do not need to compute P_t explicitly, as we only need $P_t r_t$. Hence, at iteration t we only need to perform $O(2t \cdot T) = O(\log^4 n)$ matrix-vector multiplications. As each matrix is a step of a lazy random walk along a perfect matching, each of these operations takes time $O(n)$. Hence, the total running time is $\tilde{O}(n)$. \square

Discussion and comparison

In Section 4.3, we already described how \mathcal{C}_{NAT} can be seen as a proxy for the natural random walk, for which we do not have an adequate decomposition lemma. \mathcal{C}_{NAT} can also be seen as a hybrid between \mathcal{C}_{NAT} and \mathcal{C}_{EXP} . P^{NAT} takes distinct steps across each matching like \mathcal{C}_{KRV} ; however, these steps are lazier, a feature that might contribute to the better results obtained by \mathcal{C}_{NAT} . On the other hand, the powering of the sequential walk $T/4$ times makes P^{NAT} comparable to P^{EXP} . Indeed, the following theorem shows that P^{NAT} can be seen as an approximation to the matrix exponential. Such approximations have been studied before in the survey by Moler and Van Loan [55] and are known as “splitting methods” for computing the matrix exponential, as they spit the exponent, in this case $L(G_t)$ into a sum of addends, the matchings, for each of which it is easy to compute the exponential. The following theorem is a simple variation of a result in [55] and is stated without proof here.

Theorem 4.5.1. *For $q > 0$, let*

$$N_t = \left(\frac{q-1}{q} \cdot I + \frac{1}{q} \cdot W(L(M_t)) \right).$$

Then, for $t \in [T]$,

$$\lim_{q \rightarrow \infty} (N_t N_{t-1} \dots N_1 N_1 \dots N_{t-1} N_t)^{q/4} = e^{-1/2 \cdot L(G_{t+1})}.$$

Our result shows that it suffices to take $q = T$ to have a sufficiently good approximation for the purposes of the Cut-Matching game under the expansion criterion.

4.6 Matrix MWU Interpretation

Our discussion of why random walks arise in the construction of cut strategies in Section 4.3 was based on the following intuition: any successful cut strategy cannot only focus on the single lowest-mixing eigenvector, but must sufficiently hedge on all low-mixing eigenvectors to guarantee that progress is made at every round. This reasoning is completely analogous to that of Chapter 3, which motivated how the Matrix MWU algorithm arises and why it makes use of the matrix exponential. It is then not surprising that the same matrix exponential plays an important role in our discussion of the cut strategies, and in particular of \mathcal{C}_{EXP} .

We chose to present \mathcal{C}_{EXP} through the random-walk derivation of Section 4.3 because this how we constructed it first and because of the strong analogy with \mathcal{C}_{NAT} and \mathcal{C}_{KRV} . However, it is possible to give a simple proof of the performance of a variant of \mathcal{C}_{EXP} just by using the results of Chapter 3. We present a sketch of this proof in the rest of this section. Among other things, this connection highlights how \mathcal{C}_{NAT} and \mathcal{C}_{KRV} can be interpreted as variations of the Matrix MWU algorithm that are targeted for the Cut-Matching game, and in particular for bounding expansion rather than the spectral gap.

Let $N = I - 1/n \cdot \vec{1}\vec{1}^T$ and notice that N is a projection onto the subspace orthogonal to the vector $\vec{1}$. We consider a Matrix MWU setting in which we must choose action $X^{(t)}$ in the set Δ_N . The loss function at time t takes the form $L(M_t)$, where M_t is the matching output by \mathcal{M} at that round. We then have, by Theorem 3.3.3, for some choice of $\epsilon > 0$:

$$X^{(t+1)} = E_{\epsilon, I, N} \left(\sum_{s=1}^t L(M_s) \right) = \frac{(1 - \epsilon)^{\sum_{s=1}^t L(M_s)}}{N \bullet (1 - \epsilon)^{\sum_{s=1}^t L(M_s)}}.$$

This construction of $X^{(t)}$ is analogous to that of P_t^{EXP} , and the two are exactly the same for $\epsilon = (1 - e)$. However, for the following analysis, we assume $\epsilon < 1/2$. Now, we notice that Lemma 4.4.1 yields a lower bound on the loss of $X^{(t)}$ at every iteration. Indeed, if we pick the output bisection as in \mathcal{C}_{EXP} , we have that, for all $t \in [T]$,

$$L(M_t) \bullet X^{(t)} \geq \frac{1}{\log n}.$$

We have $0 \preceq L(M_t) \preceq 2N$, so that, by Theorem 3.3.3, :

$$\frac{T}{\log n} \leq \sum_{t=1}^T L(M_t) \bullet X^{(t)} \leq \frac{2\rho \log n}{\epsilon} + (1 + \epsilon)\lambda_{\min, N} \left(\sum_{t=1}^T L(M_t) \right).$$

Finally,

$$\begin{aligned} \text{gap}(G_{T+1}) = \lambda_{\min, T \cdot N} \left(\sum_{t=1}^T L(M_t) \right) &\geq \frac{1}{T} \left(\Omega \left(\frac{T}{\log n} \right) - O(\log n) \right) \geq \\ &\Omega \left(\frac{1}{\log n} \right) - O \left(\frac{\log n}{T} \right). \end{aligned}$$

Hence, taking $T = O(\log^2 n)$, the cut player \mathcal{C}_{EXP} achieves $\text{gap}(G_{T+1}) = \Omega(1/\log n)$.

Despite hiding the random-walk intuition, this simpler proof has many advantages. In particular, it can be used to generalize the results about \mathcal{C}_{EXP} to variant of the Cut-Matching games that capture conductance and other graph partitioning objectives.

4.7 Lower Bounds for the Cut-Matching Game

In this section we prove the lower bound of Theorem 4.1.6 on the performance of any cut player in the Cut-Matching game under the expansion criterion. This result establishes the existence of a matching strategy \mathcal{M}^* such that no cut players, including computationally unbounded ones, can achieve an expansion better than $\Omega(\sqrt{\log n}) \cdot g(n)$ against \mathcal{M}^* .

4.7.1 Proof Idea

A matching player \mathcal{M} wins the the game $(\mathcal{G}(n), O(1/\sqrt{\log n}), g(n))$ for all $g(n)$ if at each round t , \mathcal{M} is able to exhibit a cut with expansion less than $O(t/\sqrt{\log n})$ in the graph G_{t-1} formed by the union of the matchings thus far. A simple way for \mathcal{M} to do this would be to pick a fixed cut (D, \overline{D}) at the beginning of the game and keep this cut as sparse as possible round after round. However, if the cut player guesses one bisection containing or equal to D , any perfect matching that \mathcal{M} adds across this bisection will make the cut (D, \overline{D}) have constant expansion immediately.

To overcome this problem, the matching player \mathcal{M}^* first identifies the vertex set $[n]$ with the vertex set of a hypercube with d coordinates, $\{-1, 1\}^d$. (Assume $n = 2^d$.) Then, rather than trying to keep one bisection sparse, it tries to keep $d = \log n$ orthogonal bisections sparse on an average. The natural choice for such orthogonal bisections for the hypercube vertex set are those induced by the coordinate cuts. Formally, denote this set of bisections by $\mathcal{D} := \{(D_1, \overline{D}_1), \dots, (D_d, \overline{D}_d)\}$. Here, $D_i := \{(x_1, \dots, x_d) \in \{-1, 1\}^d \mid x_i = 1\}$, and $\overline{D}_i := \{-1, 1\}^d \setminus D_i$. The orthogonality makes it possible to add edges across one (D_i, \overline{D}_i) without increasing the expansion of other bisections in \mathcal{D} by too much. More formally, we will show that, after t rounds, the expected expansion is at most $O(t/\sqrt{\log n})$, which implies the existence of a cut D_i with the required expansion.

The Main Lemma, described in the next subsection, achieves this goal by proving that at any iteration t , and for any choice of a bisection (S_t, \overline{S}_t) (by any cut player), there exists a matching M_t across (S_t, \overline{S}_t) which increases the average expansion over \mathcal{D} by at most $O\left(\frac{1}{\sqrt{\log n}}\right)$.

4.7.2 Main Lemma

The main technical result of this section is the following claim about the hypercube. The lower bound is a simple consequence of this result.

Lemma 4.7.1 (Main Lemma). *Given a unit embedding of the d -dimensional hypercube $\left(\frac{\pm 1}{\sqrt{d}}, \frac{\pm 1}{\sqrt{d}}\right)^d$, for any bisection of its vertices, there exists a perfect matching of the vertices across the bisection such that the average ℓ_2^2 -distance of matched vertices is $O\left(\frac{1}{\sqrt{d}}\right)$.*

Notice that here a matching just indicates a pairing of the vertices and has no relation with the edges of the hypercube graph. The Main Lemma will be shown to be a consequence of the vertex iso-perimetry of the hypercube. Intuitively, if the vertex iso-perimetry is large, no two large sets of vertices can be a large distance apart. The proof shows how to apply the same reasoning to show the existence of a “short” perfect matching across any bisection. To establish this lemma, we first encode the task of finding a matching across the given bisection with minimum ℓ_2^2 length as a min-cost perfect-matching LP. Then, we show that the dual of this LP can be interpreted as a non-expanding embedding of the hypercube into ℓ_1 . This allows us to use the hypercube vertex iso-perimetry to upper bound its optimal value. The matching of interest can then be found by the matching player by solving the matching LP.

In the next subsection, we give some necessary preliminaries, before proceeding to the proof of Theorem 4.1.6 and Lemma 4.7.1.

4.7.3 Preliminaries

Cut vectors

For any cut (S, \bar{S}) of $[n]$, we define the cut vector $\vec{x}_S \in \mathbb{R}^n$ by:

$$(\vec{x}_S)_i = \begin{cases} +1 & \text{if } i \in S \\ -1 & \text{if } i \notin S \end{cases}$$

Hence, for any cut (S, \bar{S}) :

$$\vec{x}_S^T L(G) \vec{x}_S = 4|E(S, \bar{S})|.$$

Vertex iso-perimetry of the hypercube

For any graph $G = (V, E)$, let $\gamma(G)$ denote the vertex iso-perimetry number of G . $\gamma(G)$ is the minimum ratio among all cuts $S \subseteq V$, with $|S| \leq \frac{|V|}{2}$, of the number of neighbors of S outside of S to that of the size of S . That is

$$\gamma(G) := \min_{S \subseteq V, |S| \leq \frac{|V|}{2}} \frac{|\{i \in V \setminus S : \exists j \in S : \{i, j\} \in E\}|}{|S|}.$$

The following is a standard fact about the vertex iso-perimetry of the hypercube [21].

Fact 4.7.2. $\gamma(H_d) = \Theta\left(\frac{1}{\sqrt{d}}\right)$.

4.7.4 Proof of Theorem 4.1.6

Let $n := 2^d$ for a positive integer d . Let H_d denote the d -dimensional hypercube. This is the graph with $V(H_d) := \{-1, 1\}^d$ and $\{i, j\} \in E(H_d)$ if and only if i and j differ in exactly

one coordinate. At the start of the game, \mathcal{M}^* picks an arbitrary bijection $f : V \rightarrow H_d$. Let U_d be the unit embedding of H_d , i.e., $U_d := \frac{H_d}{\sqrt{d}}$ and, for all $v \in V$, denote by u_v the point $\frac{f(v)}{\sqrt{d}}$ of U_d . Each dimension cut in H_d corresponds to a cut in V through the mapping f . In particular, we denote by D_i the cut $\{v \in V : f(v)_i = +1\}$, and $\bar{D}_i := V \setminus D_i$. This defines a set $\mathcal{D} := \{D_1, \dots, D_d\}$ of bisections of V .

Fix an arbitrary cut player \mathcal{C} which at every round presents a bisection to the matching player \mathcal{M}^* to which \mathcal{M}^* must add a perfect matching. At every round, \mathcal{M}^* will output a perfect matching M_t . Let $G_t := (V, E_t)$ denote the graph formed by the union of matchings M_1, \dots, M_t , with $G_1 := (V, \emptyset)$. Define a potential function

$$\Phi_t \stackrel{\text{def}}{=} \mathbb{E}_{D_i \leftarrow \mathcal{D}} \left[\frac{|E_t(D_i, \bar{D}_i)|}{|D_i|} \right]$$

to be the expected expansion in G_t of a cut sampled uniformly at random from \mathcal{D} . The following fact shows that the value of the potential function Φ_t equals a scaling of the sum of the squared lengths of the edges of G_t in the hypercube embedding U_d .

Fact 4.7.3. $\Phi_t = \frac{1}{2n} \sum_{\{h,k\} \in E_t} \|u_h - u_k\|^2$

Proof.

$$\begin{aligned} \Phi_t &= \mathbb{E}_{D_i \leftarrow \mathcal{D}} \left[\frac{|E_t(D_i, \bar{D}_i)|}{|D_i|} \right] = \mathbb{E}_{D_i \leftarrow \mathcal{D}} \left[\frac{\vec{x}_{D_i}^T L(G_t) \vec{x}_{D_i}}{4|D_i|} \right] \\ &= \frac{1}{d} \sum_{i=1}^d \frac{\vec{x}_{D_i}^T L(G_t) \vec{x}_{D_i}}{2n} \\ &= \frac{1}{d} \sum_{i=1}^d \frac{\sum_{\{h,k\} \in E_t} ((\vec{x}_{D_i})_h - (\vec{x}_{D_i})_k)^2}{2n} \\ &= \frac{1}{2n} \sum_{\{h,k\} \in E_t} \sum_{i=1}^d \left(\frac{1}{\sqrt{d}} (\vec{x}_{D_i})_h - \frac{1}{\sqrt{d}} (\vec{x}_{D_i})_k \right)^2 \\ &= \frac{1}{2n} \sum_{\{h,k\} \in E_t} \|u_h - u_k\|^2. \end{aligned}$$

Notice that in the last inequality we used the definition of the cuts D_1, \dots, D_d as the coordinate cuts of H_d . \square

Hence, for any $t \geq 1$, we can rewrite the increase in potential at round t as:

$$\Phi_t - \Phi_{t-1} = \sum_{\{i,j\} \in E_t \setminus E_{t-1}} \|u_i - u_j\|^2 = \sum_{\{i,j\} \in E(M_t)} \|u_i - u_j\|^2$$

At every iteration t , given \mathcal{C} 's choice of (S_t, \bar{S}_t) , our player \mathcal{M}^* adds the matching M_t across (S_t, \bar{S}_t) which minimizes $\sum_{\{i,j\} \in E(M_t)} \|u_i - u_j\|^2$. This only requires a minimum cost matching computation on the complete bipartite graph induced by (S_t, \bar{S}_t) . Moreover, this choice of matching ensures the potential increases the least possible at every iteration.

The Main Lemma (Lemma 4.7.1) can now be restated as follows.

Lemma 4.7.4. *For all bisections (S, \bar{S}) of V , there exists a perfect matching M across (S, \bar{S}) such that $\sum_{\{i,j\} \in M} \|u_i - u_j\|^2 = O\left(\frac{n}{\sqrt{d}}\right)$.*

The proof of this Lemma will be given in Section 4.7.5. Here we see how the Main Lemma implies Theorem 4.1.6.

Proof of Theorem 4.1.6. By the Main Lemma and Fact 4.7.3, the potential increase at round t round is at most

$$\Phi_{t+1} - \Phi_t = O\left(\frac{1}{\sqrt{d}}\right).$$

Hence $\Phi_{t+1} \leq O\left(\frac{t}{\sqrt{d}}\right)$. This implies that $\mathbb{E}_{D_i \leftarrow \mathcal{D}} \left[\frac{|E_{t+1}(D_i, \bar{D}_i)|}{|D_i|} \right] \leq O\left(\frac{t}{\sqrt{d}}\right)$. Hence, there exists a cut D_i with $\frac{|E_{t+1}(D_i, \bar{D}_i)|}{|D_i|} \leq O\left(\frac{t}{\sqrt{d}}\right)$. This shows that $\alpha(G_{t+1}) \leq O\left(\frac{t}{\sqrt{d}}\right)$ for all integers $t \geq 1$. Hence for any choice of termination $T = g(n)$, we have

$$\alpha(G_{T+1}) \leq O\left(\frac{g(n)}{\sqrt{d}}\right) \leq O(1/\sqrt{\log n}) \cdot g(n)$$

as required. □

4.7.5 Proof of Lemma 4.7.4

We now proceed to prove the Main Lemma.

of Main Lemma 4.7.4. Let $c_{ij} := \|u_i - u_j\|^2$. Consider the LP relaxation of Figure 1 for computing the minimum cost perfect matching across the cut (S, \bar{S}) .

By the integrality of the bipartite perfect matching polytope (see [57]), the objective of this program is the minimum of $\sum_{\{i,j\} \in M} \|u_i - u_j\|^2$ over all perfect matchings M across (S, \bar{S}) . In Figure 2 we consider a formulation of the dual of this LP.

A feasible solution for this LP can be seen an embedding $\{y_i\}_{i \in [n]}$ of $[n]$ on the real line such that no pair i, j with $i \in S$ and $j \in \bar{S}$ and $y_i \geq y_j$ can be further away in ℓ_1 distance than its ℓ_2^2 distance in the hypercube embedding U_d . We now prove the following two properties of solutions to the dual LP:

1. If $\{y_i\}_{i \in [n]}$ is a feasible solution of value Y , then for any $c \in \mathbb{R}$, $\{y'_i = y_i + c\}_{i \in [n]}$ is a feasible solution of value $Y' = Y$.

$$\begin{array}{rcl}
& \text{Minimize} & \sum_{i \in S, j \in \bar{S}} c_{ij} x_{ij} \\
\text{Subject to} & & \\
& \forall i \in S, & \sum_{j \in \bar{S}} x_{ij} = 1 \\
& \forall j \in \bar{S}, & \sum_{i \in S} x_{ij} = 1 \\
& \forall i \in S, j \in \bar{S}, & x_{ij} \geq 0
\end{array}$$

Figure 4.2: LP for Bipartite Min-Cost Matching

$$\begin{array}{rcl}
& \text{Maximize} & \sum_{i \in S} y_i - \sum_{j \in \bar{S}} y_j \\
\text{Subject to} & & \\
& \forall i \in S, j \in \bar{S}, & y_i - y_j \leq c_{ij} \\
& \forall i \in V, & y_i \in \mathbb{R}
\end{array}$$

Figure 4.3: The dual of the LP for Bipartite Min-Cost Matching

2. In any optimal dual solution, we must have, for all pairs $i, j \in [n]$, $|y_i - y_j| \leq c_{ij} = \|u_i - u_j\|^2$.

Proof of Property 1: The shifted solution is feasible as for all $i \in S, j \in \bar{S}$:

$$y'_i - y'_j = y_i + c - y_j - c = y_i - y_j \leq c_{ij}$$

The value of this solution is:

$$\begin{aligned}
Y' &= \sum_{i \in S} y'_i - \sum_{j \in \bar{S}} y'_j = \sum_{i \in S} (y_i + c) - \sum_{j \in \bar{S}} (y_j + c) \\
&= \sum_{i \in S} y_i + \frac{cn}{2} - \sum_{j \in \bar{S}} y_j - \frac{cn}{2} = \sum_{i \in S} y_i - \sum_{j \in \bar{S}} y_j = Y
\end{aligned}$$

Proof of Property 2: Notice that the costs c_{ij} 's respect the triangle inequality as the ℓ_2^2 -distance on the hypercube is a metric. To prove the statement, we need to handle the three remaining cases:

1. $i \in S, j \in S$. Assume $y_i \geq y_j$ without loss of generality. As the solution is optimal, it is not possible to increase y_j to obtain a larger dual objective. This implies that there must exist $k \in \bar{S}$ such that $y_j - y_k = c_{jk}$. But we must have $c_{ik} \geq y_i - y_k = (y_i - y_j) + (y_j - y_k) = y_i - y_j + c_{jk}$. As $c_{ik} \leq c_{ij} + c_{jk}$, we have $y_i - y_j \leq c_{ij}$.
2. $i \in \bar{S}, j \in \bar{S}$. This is handled as in the previous case.

3. $i \in S, j \in \bar{S}$ such that $y_j \geq y_i$. Because the solution is optimal there must exist $j' \in S$ and $i' \in \bar{S}$ such that $y_{j'} - y_j = c_{j'j}$ and $y_i - y_{i'} = c_{ii'}$. But, by the dual constraint, we must have $c_{i'j'} \geq y_{j'} - y_{i'} = (y_{j'} - y_j) + (y_j - y_i) + (y_i - y_{i'}) = c_{j'j} + y_j - y_i + c_{ii'}$. By triangle inequality, $c_{i'j'} \leq c_{j'j} + c_{ji} + c_{ii'}$, so that $y_j - y_i \leq c_{ji}$ as required.

Application of vertex iso-perimetry of H_d : Now we use these properties of an optimal dual solution together with the vertex iso-perimetry of the hypercube to obtain an upper bound on the dual optimal. By Property 1, we can translate any optimal dual solution preserving optimality. Hence, we may consider an optimal solution $\{y_i\}_{i \in [n]}$ such that at most $\frac{n}{2}$ vertices are mapped to positive value and at most $\frac{n}{2}$ are mapped to negative values. Notice that, as $\max_{i,j} \|u_i - u_j\|^2 = 4$, we have $y_k \in [-4, 4]$ for all $k \in [n]$. Now define sets $R_1, \dots, R_{4d} \subseteq [n]$ as follows:

$$R_i := \left\{ k \in [n] : y_k \in \left(\frac{i-1}{d}, \frac{i}{d} \right] \right\}.$$

Similarly, for the negative side we can define L_1, \dots, L_{4d} :

$$L_i := \left\{ k \in [n] : y_k \in \left[-\frac{i}{d}, -\frac{i-1}{d} \right) \right\}.$$

We also define $A_i := \bigcup_{k=i}^{4d} R_k$ and $B_i := \bigcup_{k=i}^{4d} L_k$. By our assumption on $\{y_i\}_{i \in [n]}$ we know that, for all i , $|A_i|, |B_i| \leq \frac{n}{2}$.

Consider now any $k \in A_i$ for $i \geq 2$. Consider any $h \notin A_i$ such that $\|u_h - u_k\|^2 = \frac{1}{d}$, i.e., u_h is a neighbor of u_k in the hypercube graph. Notice that h must lie in R_{i-1} , as, by Property 2, $|y_k - y_h| \leq \frac{1}{d}$ and $h \notin A_i$. Hence, all vertices which are outside of A_i and adjacent to A_i in the hypercube must belong to R_{i-1} . Because $|A_i| \leq \frac{n}{2}$, by the vertex iso-perimetry of the hypercube, there are at least $\gamma(H_d)|A_i|$ such vertices and, for $i \geq 2$:

$$|R_{i-1}| \geq \Omega\left(\frac{1}{\sqrt{d}}\right) |A_i|.$$

This implies that for $i \geq 2$,

$$|A_{i-1}| \geq \left(1 + \Omega\left(\frac{1}{\sqrt{d}}\right)\right) |A_i|.$$

Since $|A_1| \leq \frac{n}{2}$,

$$|A_i| \leq \frac{n}{2} \left(1 + \Omega\left(\frac{1}{\sqrt{d}}\right)\right)^{-(i-1)}.$$

The same reasoning can be applied to B_i to deduce that

$$|B_i| \leq \frac{n}{2} \left(1 + \Omega\left(\frac{1}{\sqrt{d}}\right)\right)^{-(i-1)}.$$

Now notice that the cost of the dual solution $\{y_k\}_{k \in [n]}$ is upper bounded by

$$\begin{aligned} \frac{1}{d} \left(\sum_{i=1}^{4d} i|L_i| + \sum_{i=1}^{4d} i|R_i| \right) &\leq \frac{1}{d} \left(\sum_{i=1}^{4d} |A_i| + \sum_{i=1}^{4d} |B_i| \right) \\ &= \frac{n}{d} \sum_{i=1}^{4d} \left(1 + \Omega \left(\frac{1}{\sqrt{d}} \right) \right)^{-(i-1)} \\ &= \frac{n}{d} O(\sqrt{d}) = O\left(\frac{n}{\sqrt{d}}\right). \end{aligned}$$

But, by strong duality, the primal optimum equals the the dual optimum. Hence, there exists a matching M such that

$$\sum_{\{i,j\} \in M} \|u_i - u_j\|^2 = O\left(\frac{n}{\sqrt{d}}\right).$$

□

Finally, we consider a slight variation on our matching-player construction, in which the matching player does not solve the matching LP at ever round. Suppose that a matching player \mathcal{M}^* , given bisection (S, \bar{S}) outputs a perfect matching by greedily matching the two vertices i and j that are closest in U_d , removing them and iterating. Using our intuition based on iso-perimetry, Sherman [59] showed that this greedy player \mathcal{M}^* achieves asymptotically the same performance as that of Theorem 4.1.6. Moreover, Sherman also showed that the same iso-perimetry method can be extended to embeddings other than the hypercube. In particular, he applied this argument to the sphere embedding to obtain a stronger lower bound regarding the Cut-Matching game under the gap criterion. He shows that, for this version of the game, there is a matching player that is successful in the $(\mathcal{G}(n), \Omega(\log \log n / \log n), g(n))$ -game for all $g(n)$.

4.8 Related Work

We remark that Arora and Kale [11] described an algorithm that also achieves $O(\log n)$ -approximation using $\text{polylog}(n)$ single-commodity maximum flow computations. However, their algorithm works outside of the Cut-Matching game and its certificate of expansion seems different than the one produced by our algorithms.

The study of fast algorithms for graph partitioning using single-commodity flows saw three other developments after the publishing of our work. First, Sherman [59] gave a $O(\sqrt{\log n}/\epsilon)$ -approximation for EXPANSION using only $\tilde{O}(n^\epsilon)$ flow operations. His algorithm did not make use of the Cut-Matching game. Secondly, Madry [54] showed how to obtain

a trade-off between approximation and running time for many graph-partitioning problem. His result yields the first polylogarithmic approximation algorithms that run in time $o(m^{3/2})$.

Finally, Christiano et al. [20] gave an algorithm for approximate single-commodity maximum flow that runs in time $\tilde{O}(m^{4/3})$. As the reduction from Cut-Matching game to EXPANSION can be modified to use approximate maxflow, their result reduces the running time of all algorithms employing this framework to $\tilde{O}(n^{4/3} + m)$.

Chapter 5

Fast Spectral Algorithms for Balanced Separator and Graph Decomposition

Recalling the definition of BALANCED SEPARATOR in Chapter 2, we seek an approximation algorithm that, on input an unweighted undirected instance graph $G = (V, E)$ with $|V| = n$, $|E| = m$, a constant balance $b \in (0, 1/2]$ and a parameter $\gamma \in [0, 1]$, either outputs a cut of conductance at most $f(\gamma, n)$ and balance $\Omega_b(1)$ ¹ or a certificate that G has no b -balanced cut of conductance at most $\Omega(\gamma)$. In their seminal series of papers [66, 67, 65], Spielman and Teng use an approximation algorithm for BALANCED SEPARATOR as a fundamental primitive to decompose the instance graph into a collection of near-expanders. This decomposition is then used to construct spectral sparsifiers and solve systems of linear equations in nearly linear time. Their algorithm has two crucial features: first, it runs in nearly linear time; second, in the case that no balanced cut exists in the graph, it outputs a certificate of a special form. Such certificate consists of an unbalanced cut of small conductance which is well-correlated with all low-conductance cuts in the graph, i.e. contains at least half of the volume of any cut that has conductance less than $O(\gamma)$. This immediately implies that no large set of small conductance can exist.

Theorem 5.0.1. [66] *Given a graph G , a balance parameter $b \in (0, 1/2]$, $b = \Omega(1)$ and a conductance value $\gamma \in (0, 1)$, $\text{PARTITION}(G, b, \gamma)$ runs in time $T(\gamma, n)$ and outputs a cut $S \subseteq V$ such that $\text{vol}(S) \leq 7/8 \cdot \text{vol}(G)$, $\phi(S) \leq f(\gamma, n)$ or $S = \emptyset$, and with high probability, either*

1. S is $\Omega_b(1)$ -balanced, or
2. for all $C \subset V$ such that $\text{vol}(C) \leq 1/2 \cdot \text{vol}(G)$ and $\phi(C) \leq O(\gamma)$, $\frac{\text{vol}(S \cap C)}{\text{vol}(C)} \geq 1/2$.

¹We will use $O_b(\cdot)$ and $\Omega_b(\cdot)$ in our asymptotic notation when we want to emphasize the dependence of the hidden coefficient on b .

Originally, Spielman and Teng showed $f(\gamma, n) = O\left(\sqrt{\gamma \log^3 n}\right)$ and $T(\gamma, n) = \tilde{O}(m/\gamma^2)$. This was subsequently improved by Andersen, Chung and Lang [4] and then by Andersen and Peres [5] to the current best of $f(\gamma, n) = O\left(\sqrt{\gamma \log n}\right)$ and $T(\gamma, n) = \tilde{O}(m/\sqrt{\gamma})$. All these results made use of bounds on the convergence of random walk processes on the instance graph, such as the Lovasz-Simonovits bounds [53]. These bounds yield the $\log n$ factor in the approximation guarantee, which appears hard to remove while closely following this approach, as such an improvement would have consequences for important variations of the Unique Games Conjecture [43, 7], a fundamental open question in Inapproximability.

5.0.1 Our Result

In this chapter, we use a semidefinite programming approach to design a new spectral algorithm, called BALCUT, that improves on the result of Theorem 5.0.1. The following is our main result.

Theorem 5.0.2 (Main Theorem). *Given a graph $G = (V, E)$, a balance parameter $b \in (0, 1/2]$, $b = \Omega(1)$, and a conductance value $\gamma \in (0, 1)$, BALCUT(G, b, γ) runs in time $\tilde{O}(m/\gamma)$ and outputs a cut $S \subset V$ such that $\text{vol}(S) \leq 1/2 \cdot \text{vol}(G)$, if $S \neq \emptyset$ then $\phi(S) \leq O_b(\sqrt{\gamma})$, and with high probability, either*

1. S is $\Omega_b(1)$ -balanced, or
2. for all $C \subset V$ such that $\text{vol}(C) \leq 1/2 \cdot \text{vol}(G)$ and $\phi(C) \leq O(\gamma)$, $\frac{\text{vol}(S \cap C)}{\text{vol}(C)} \geq 1/2$.

Note that our result improves the parameters of previous algorithms by eliminating the $\log n$ factor in the quality of the cut output, making the approximation comparable to the best that can be hoped for using spectral methods [30]. Our result is also conceptually simple: we use the primal-dual framework of Arora and Kale [11], which we described in Section 3.4, to solve SDPs combinatorially, and we give a new separation oracle that yields Theorem 5.0.2. Moreover, our algorithm has a simple and intuitive interpretation in terms of random walks, which we discuss in Section 5.4. Finally, our result implies an approximation algorithm for BALANCED SEPARATOR, as the guarantee of Theorem 5.0.2 on the cut S output by BALCUT also implies a lower bound on the conductance of balanced cuts of G .

Corollary 5.0.3. *Given an instance graph G , a balance parameter $b \in (0, 1/2]$ and a target conductance $\gamma \in (0, 1]$, BALCUT(G, b, γ) either outputs an $\Omega_b(1)$ -balanced cut of conductance at most $O_b(\sqrt{\gamma})$ or a certificate that all $\Omega_b(1)$ -balanced cuts have conductance at least $\Omega(\gamma)$. The running time of the algorithm is $\tilde{O}(m/\gamma)$.*

This is the first nearly-linear-time spectral algorithm for BALANCED SEPARATOR that achieves the asymptotically optimal approximation guarantee for spectral methods.

5.0.2 Application to Graph Decomposition.

The main application of Theorem 5.0.1 is the construction of a particular kind of graph decomposition. In this decomposition, we wish to partition the vertex set of the instance graph V into components $V_1, \dots, V_i, \dots, V_k$ such that the graph induced by G on each V_i has conductance as large as possible, while at most a constant fraction of the edges have endpoints in different components. These decompositions are a useful algorithmic tool in several areas, such as clustering and preconditioning [69, 47, 67, 36].

Kannan, Vempala and Vetta [36] construct such decompositions achieving a conductance value of $\Omega(1/\log^2 n)$. However, their algorithm runs in time $\tilde{O}(n^2)$ on some instances. Spielman and Teng [67] relax this notion of decomposition by only requiring that each V_i be contained in a superset W_i in G , where W_i has large induced conductance in G . In the same work, they show that this relaxed notion of decomposition suffices for the purposes of sparsification by random sampling. The advantage of this relaxation is that it is now possible to compute this decomposition in nearly-linear time by recursively applying the algorithm of Theorem 5.0.1.

Theorem 5.0.4. [67] *Assume the existence of an algorithm achieving a nearly-linear running time $T(\gamma, n)$ and approximation $f(\gamma, n)$ in Theorem 5.0.1. Given $\gamma \in (0, 1)$, in time $\tilde{O}(T(\gamma, n))$, it is possible to construct a decompositions of the instance graph G into components V_1, \dots, V_k such that:*

1. *for each V_i , there exists $W_i \supseteq V_i$ such that the conductance of the graph induced by G on W_i is $\Omega(\gamma/\log n)$.*
2. *the fraction of edges with endpoints in different components is $O(f(\gamma, n) \cdot \log n)$.*

Using Theorem 5.0.4, Spielman and Teng showed the existence of a decomposition achieving conductance $\Omega(1/\log^6 n)$. Our improved results in Theorem 5.0.2 imply that we can obtain decompositions of the same kind with conductance bound $\Omega(1/\log^3 n)$. Our improvement also implies speed-ups in the sparsification procedure described by Spielman and Teng [67]. Our work leaves open the important question posed by Spielman [62] of whether stronger decompositions, of the kind proposed by Kannan, Vempala and Vetta [36], can be produced in nearly-linear time.

5.0.3 Our Techniques

We will use the SDP relaxation of Figure 5.1. We denote by $\mu : V \mapsto \mathbb{R}_{\geq 0}$ the distribution defined as $\mu_i \stackrel{\text{def}}{=} d_i/\text{vol}(G)$, and by d_i the degree of the i -th vertex. Also, $v_{\text{avg}} \stackrel{\text{def}}{=} \sum_i \mu_i v_i$. Even though our algorithm uses the SDP, at the core, it is spectral in nature, as it relies on the matrix-vector multiplication primitive. We will formalize this reasoning by discussing a random walk interpretation of our algorithm in Section 5.4.

$$\begin{aligned}
\text{psdp}(G, b, \gamma) : \quad & \frac{1}{4} \cdot \mathbb{E}_{\{i,j\} \in E} \|v_i - v_j\|_2^2 \leq \gamma \\
& \mathbb{E}_{j \sim \mu} \|v_j - v_{\text{avg}}\|_2^2 = 1 \\
\forall i \in V \quad & \|v_i - v_{\text{avg}}\|_2^2 \leq \frac{(1-b)}{b}
\end{aligned}$$

Figure 5.1: SDP for b -BALANCED SEPARATOR

For our SDP, the method of Arora and Kale can be understood as a game between two players: an embedding player and an oracle player. The embedding player, in every round of this game, gives a candidate vector embedding of the vertices of the instance graph to the oracle player. We show that if the embedding is close to feasible for the SDP, i.e. the first two constraints are satisfied and for a large set S , for every $i \in S$, $\|v_i - v_{\text{avg}}\|_2 \leq O((1-b)/b)$, then a projection of the vectors along a random direction followed by a sweep cut gives an $\Omega_b(1)$ -balanced cut of conductance at most $O(\sqrt{\gamma})$. We call such an embedding *roundable*. The difficult case takes place when the embedding given to the oracle player is not roundable. In this case, the oracle outputs a candidate dual solution along with a cut. The oracle obtains this cut by performing a radial sweep cut of the vectors given by the embedding player. We show that such a cut is of conductance at most $O_b(\sqrt{\gamma})$. If at any point in this game the union of cuts output by the oracle becomes balanced, we output this union and stop. If this union of cuts is not balanced, then the embedding player uses the dual solution output by the oracle to update the embedding. Finally, the matrix-exponential update rule ensures that this game cannot keep on going for more than $O(\log n/\gamma)$ rounds. Hence, if a balanced cut is not found after this many rounds, we certify that the graph does not contain any b -balanced cut of conductance less than γ . To achieve a nearly-linear running time, we maintain only a $\log n$ -dimensional sketch of the embedding. The guarantee on the running time then follows by noticing that, in each iteration, the most expensive computational step for each player is a logarithmic number of matrix-vector multiplications, which takes at most $\tilde{O}(m)$ time.

The reason why our approach yields the desired correlation condition in Theorem 5.0.2 is that, if no balanced cut is found, every unbalanced cut of conductance lower than γ will, at some iteration, have a lot of its vertices mapped to vectors of large radius. At that iteration, the cut output by the oracle player will have a large correlation with the target cut, which implies that the union of cuts output by the oracle player will also display such large correlation. This intuition is formalized in the proof of Theorem 5.0.2.

The implementation of the oracle player, specifically dealing with the case when the embedding is not roundable, is the main technical novelty of our work. Studying the problem in the SDP-framework is the main conceptual novelty. Before our work, all nearly-linear-time algorithms for this problem were based on the use of local random walks. The main

advantage of using SDPs to design a spectral algorithm seems to be that SDP solutions provide a simple representation for possibly complex random-walk objects. Furthermore, the benefits of using a carefully designed SDP formulation can often be reaped with little or no burden on the running time of the algorithm, thanks to the primal-dual framework of Arora and Kale [11].

5.1 Algorithm Statement and Main Theorems

In Section 5.1.1, we set some useful notation and state a few basic facts. In Section 5.1.2, we present our SDP, its dual and define the notion of a roundable embedding. In Section 5.1.3, we present the algorithm BALCUT and the separation oracle ORACLE, and reduce the task of proving Theorem 5.0.2 to proving statements about the ORACLE. Section 5.3 contains the proof of the main theorem about the ORACLE used in Section 5.1.3.

5.1.1 Notation and Basic Facts

Instance graph and edge volume. We denote by $G = (V, E)$ the unweighted instance graph, where $V = [n]$ and $|E| = m$. We let $d \in \mathbb{R}^V$, be the degree vector of G , i.e. d_i is the degree of vertex i . We mostly work with the edge measure μ over V , defined as $\mu_i \stackrel{\text{def}}{=} \mu(i) \stackrel{\text{def}}{=} d_i/2m$. For a subset $S \subseteq V$, we also define μ_S as the edge measure over S , i.e. $\mu_S(i) \stackrel{\text{def}}{=} \mu(i)/\mu(S)$. Notice that $\mu(S) = \text{vol}(S)/2m$, for the concept of volume defined in Chapter 2.

Special graphs For a subset $S \subseteq V$, we denote by K_S the complete graph over S such that edge $\{i, j\}$ has weight $\mu_i\mu_j$ for $i, j \in S$ and 0 otherwise. K_V is the complete graph with weight $\mu_i\mu_j$ between every pair $i, j \in V$. For $i \in V$, we denote by S_i the star graph rooted at i . S_i has an edge $\{i, j\}$ of weight μ_j for all $j \in V$.

Embedding notation. We will deal with vector embeddings of G , where each vertex $i \in V$ is mapped to a vector $v_i \in \mathbb{R}^h$. For such an embedding $\{v_i\}_{i \in V}$, we denote by v_{avg} the mean vector, i.e. $v_{\text{avg}} \stackrel{\text{def}}{=} \sum_{i \in V} \mu_i v_i$. Given a vector embedding of $\{v_i \in \mathbb{R}^h\}_{i \in V}$, recall that $X \succeq 0$, is the Gram matrix of the embedding if $X_{ij} = v_i^T v_j$. For any $X \in \mathbb{R}^{V \times V}$, $X \succeq 0$, we call $\{v_i\}_{i \in V}$ the *embedding corresponding to X* if X is the Gram matrix of $\{v_i\}_{i \in V}$. For $i \in V$, we denote by R_i the matrix such that $R_i \bullet X = \|v_i - v_{\text{avg}}\|_2^2$.

Basic facts. We will alternatively use vector and matrix notation to reason about the graph embeddings. The following are some simple conversions between vectors and matrix forms and some basic geometric facts which follow immediately from definitions. Here $X \succeq 0$ and $\{v_i\}$ is the corresponding embedding.

Fact 5.1.1. $\mathbb{E}_{i \sim \mu} \|v_i - v_{\text{avg}}\|_2^2 = 1/2 \cdot \mathbb{E}_{\{i, j\} \sim \mu \times \mu} \|v_i - v_j\|_2^2 = L(K_V) \bullet X$.

Fact 5.1.2. For $i \in V$, $L(S_i) = L(R_i) + L(K_V)$.

Fact 5.1.3. For a subset $S \subseteq V$, $\sum_{i \in \bar{S}} \mu_i R_i \succeq \mu(S)L(K_V) - L(K_S)$.

Fact 5.1.4. For a subset $S \subseteq V$, $\mathbb{E}_{\{i,j\} \sim \mu_S \times \mu_S} \|v_i - v_j\|_2^2 = 2 \cdot 1/\mu(S)^2 \cdot L(K_S) \bullet X$.

Modified matrix exponential update. We will apply the method of Section 3.4 with normalization matrix equal to $L(K_V)$. Letting $v = 1/2m \cdot D^{1/2} \vec{1}$, notice that

$$L(K_V) = \frac{1}{2m} \left(D - \frac{1}{2m} D \vec{1} \vec{1}^T D \right) = \left(\frac{1}{\sqrt{2m}} D^{1/2} \right) (I - vv^T) \left(\frac{1}{\sqrt{2m}} D^{1/2} \right). \quad (5.1)$$

As v is a unit vector, $I - vv^T$ is a projection matrix and Equation 5.1 gives us the decomposition of the normalization matrix required to apply Theorem 3.4.4. Then, let $\Pi = I - vv^T$. Our updates will take the following form, for a positive ϵ and a symmetric matrix $A \in \mathbb{R}^{V \times V}$,

$$U_\epsilon(A) \stackrel{\text{def}}{=} E_{\epsilon, 1/2m \cdot D, \Pi}(A) = 2m \cdot \frac{D^{-1/2} (1 - \epsilon)^{2m \cdot D^{-1/2} A D^{-1/2}} D^{-1/2}}{\Pi \bullet (1 - \epsilon)^{2m \cdot D^{-1/2} A D^{-1/2}}} = \frac{D^{-1/2} (1 - \epsilon)^{2m \cdot D^{-1/2} A D^{-1/2}} D^{-1/2}}{L(K_V) \bullet D^{-1/2} (1 - \epsilon)^{2m \cdot D^{-1/2} A D^{-1/2}} D^{-1/2}}$$

We will draw a connection between this update and the heat-kernel random walk in Section 5.4.

5.1.2 SDP Formulation

We consider an SDP relaxation to the decision problem of determining whether the instance graph G has a b -balanced cut of conductance at most γ . The SDP feasibility program $\text{psdp}(G, b, \gamma)$ appears in Figure 5.2, where we also rewrite the program in matrix notation, using Fact 5.1.1 and the definition of R_i . psdp can be seen as a scaled version of the balanced-

$$\begin{array}{ll} \text{psdp}(G, b, \gamma) : & \mathbb{E}_{\{i,j\} \in E} \|v_i - v_j\|_2^2 \leq 4\gamma & \text{psdp}(G, b, \gamma) : & \frac{1}{m} \cdot L \bullet X \leq 4\gamma \\ & \mathbb{E}_{j \sim \mu} \|v_j - v_{\text{avg}}\|_2^2 = 1 & & L(K_V) \bullet X = 1 \\ \forall i \in V & \|v_i - v_{\text{avg}}\|_2^2 \leq \frac{1-b}{b} & \forall i \in V & R_i \bullet X \leq \frac{1-b}{b} \\ & & & X \succeq 0 \end{array}$$

Figure 5.2: SDP for b -BALANCED SEPARATOR

cut SDP of [12], modified by replacing v_{avg} for the origin and removing the triangle-inequality constraints. The first change makes our **psdp** invariant under translation of the embeddings and makes the connection to spectral methods more explicit. Indeed, the first two constraints of **psdp** now exactly correspond to the standard eigenvector problem, with the addition of the R_i constraint ideally forcing all entries in the eigenvector not to be too far from the mean, just as it would be the case if the eigenvector exactly corresponded to a balanced cut. The removal of the triangle-inequality constraints causes **psdp** to only deal with the spectral structure of L and not to have a flow component. For the rest of the paper, denote by Δ the set $\{X \in \mathbb{R}^{V \times V}, X \succeq 0 : L(K_V) \bullet X = 1\}$.

The following simple lemma establishes that **psdp** is indeed a relaxation for the integral decision question and is proved in Section 5.5.

Lemma 5.1.5 (SDP is a Relaxation). *If there exists a b -balanced cut S with $\phi(S) \leq \gamma$, then $\text{psdp}(G, b, \gamma)$ has a feasible solution.*

BALCUT will use the primal-dual approach of [11] and Section 3.4 to determine the feasibility of $\text{psdp}(G, b, \gamma)$. When **psdp** is infeasible, BALCUT will output a solution to the dual $\text{dsdp}(G, b, \gamma)$, shown in Figure 5.3.

$$\begin{aligned} \text{dsdp}(G, b, \gamma) : \quad & \alpha - \frac{1-b}{b} \sum_{i \in V} \beta_i > 4\gamma \\ & \frac{1}{m} \cdot L + \sum_{i \in V} \beta_i R_i - \alpha L(K_V) \succeq 0 \\ & \alpha \in \mathbb{R}, \quad \beta \geq 0 \end{aligned}$$

Figure 5.3: $\text{dsdp}(G, b, \gamma)$ feasibility problem

Following the notation of Section 3.4, for the rest of this chapter we are going to use the following shorthands for the dual constraints

$$V(\alpha, \beta) \stackrel{\text{def}}{=} \alpha - \frac{1-b}{b} \sum_{i \in V} \beta_i, \quad M(\alpha, \beta) \stackrel{\text{def}}{=} \frac{L}{m} + \sum_{i \in V} \beta_i R_i - \alpha L(K_V).$$

Notice that $V(\alpha, \beta)$ is a scalar, while $M(\alpha, \beta)$ is a matrix in $\mathbb{R}^{V \times V}$. Given $X \succeq 0$, a choice of (α, β) such that $V(\alpha, \beta) > 4\gamma$ and $M(\alpha, \beta) \bullet X \geq 0$ corresponds to a hyperplane separating X from the feasible region of $\text{psdp}(G, b, \gamma)$ and constitutes a certificate that X is not feasible.

Ideally, BALCUT would produce a feasible solution to **psdp** and then round it to a balanced cut. However, as discussed in [11], it often suffices to find a solution “close” to feasible for the rounding procedure to apply. In the case of **psdp**, the concept of “closeness” is captured by the notion of *roundable* solution.

Definition 5.1.6 (Roundable Embedding). Given an embedding $\{v_i\}_{i \in V}$, let $R = \{i \in V : \|v_i - v_{\text{avg}}\|_2^2 \leq 32 \cdot (1-b)/b\}$. We say that $\{v_i\}_{i \in V}$ is a *roundable* solution to $\text{psdp}(G, b, \gamma)$ if:

- $\mathbb{E}_{\{i,j\} \in E} \|v_i - v_j\|_2^2 \leq 2\gamma$,
- $\mathbb{E}_{j \sim \mu} \|v_j - v_{\text{avg}}\|_2^2 = 1$,
- $\mathbb{E}_{\{i,j\} \sim \mu_R \times \mu_R} \|v_i - v_j\|_2^2 \geq 1/64$.

A roundable embedding can be converted into a balanced cut of the conductance required by Theorem 5.0.2 by using a standard projection rounding, which is a simple extension of an argument already appearing in [12] and [11]. The rounding procedure PROJROUND is described precisely in Section 5.5, where the following theorem is proved.

Theorem 5.1.7 (Rounding Roundable Embeddings). *If $\{v_i \in \mathbb{R}^h\}_{i \in V}$ is a roundable solution to $\text{psdp}(G, b, \gamma)$, then PROJROUND($\{v_i\}_{i \in V}, b$) produces a $\Omega_b(1)$ -balanced cut of conductance $O_b(\sqrt{\gamma})$ with high probability in time $\tilde{O}(nh + m)$.*

5.1.3 Primal-Dual Framework

In this subsection, we define the algorithm BALCUT and justify it as an instantiation of the SDP-solver of Section 3.4. This view is the most useful in the analysis, but does not convey a strong intuition behind the workings of the algorithm. In Section 5.4, we give a different interpretation of BALCUT, based on random walks.

Separation Oracle. By Theorem 3.4.4 in Section 3.4, the problem of checking the feasibility of an SDP can be reduced to that of, given a candidate solution X , to check whether it is close to feasible and, if not, provide a certificate of infeasibility in the form of a hyperplane separating X from the feasible set. The algorithm performing this computation is known as a separation oracle. Specific conditions under which a separation oracle yields an algorithm for approximately solving an SDP program were given in Definition 3.4.2. We introduce the concept of *good* separation oracle to capture these conditions for the program $\text{psdp}(G, \beta, 3/4 \cdot \gamma)$.

Definition 5.1.8 (Good Separation Oracle). An algorithm is a *good* separation oracle if, on input some representation of X , the algorithm either finds X to be a roundable solution to $\text{psdp}(G, b, \gamma)$ or outputs coefficients α, β such that $V(\alpha, \beta) \geq 3/4 \cdot \gamma$, $M(\alpha, \beta) \bullet X \geq 0 \cdot \gamma$ and $-\gamma L(K_V) \preceq M(\alpha, \beta) \preceq 5L(K_V)$.

Note that a good separation oracle is a $(\gamma, 5)$ -oracle for $\text{psdp}(G, \beta, 3/4 \cdot \gamma)$ by Definition 3.4.2.

Input: An instance graph $G = (V, E)$, a balance value $b \in (0, 1/2]$ such that $b = \Omega(1)$, a conductance value $\gamma \in (0, 1)$.

Let $\epsilon = 1/32$ and $\delta = \gamma/16$. For $t = 1, 2, \dots, T = O\left(\frac{\log n}{\gamma}\right)$:

- Compute the embedding $\{\tilde{v}_i^{(t)}\}_{i \in V}$ corresponding to

$$\tilde{X}^{(t)} = \tilde{U}_\epsilon \left(\frac{1}{10} \cdot \sum_{j=1}^{t-1} M(\alpha^{(j)}, \beta^{(j)}) \right).$$

If $t = 1$, $\tilde{X}^{(1)} = \tilde{U}_\epsilon(0) = 2m/n_{-1} \cdot D^{-1}$.

- Execute ORACLE $\left(G, b, \gamma, \{\tilde{v}_i^{(t)}\}_{i \in V}\right)$.
- If ORACLE finds that $\{\tilde{v}_i^{(t)}\}_{i \in V}$ is roundable, run PROJROUND $\left(G, b, \{\tilde{v}_i^{(t)}\}_{i \in V}\right)$, output the resulting cut and terminate.
- Otherwise, ORACLE outputs coefficients $(\alpha^{(t)}, \beta^{(t)})$ and cut $B^{(t)}$.
- Let $C^{(t)} \stackrel{\text{def}}{=} \bigcup_{i=1}^t B^{(i)}$. If $C^{(t)}$ is $b/4$ -balanced, output $C^{(t)}$ and terminate.
- Otherwise, proceed to the next iteration.

Output $S = \bigcup_{t=1}^T B^{(t)}$. Also output $\alpha' = 1/T \sum_{t=1}^T \alpha^{(t)} - \delta$ and $\bar{\beta} = 1/T \sum_{t=1}^T \beta^{(t)}$.

Figure 5.4: The BALCUT Algorithm

Algorithmic Scheme. The algorithmic strategy of Section 3.4 is to produce a sequence of candidate primal solutions $X^{(1)}, \dots, X^{(T)}$ iteratively, such that $X^{(t)} \in \Delta$ for all t . For the following discussion, let ϵ be a small constant parameter.

Our starting point $X^{(1)}$ will be the solution $U_\epsilon(0) = 2m/n_{-1} \cdot D^{-1}$. At every iteration, a *good* separation oracle ORACLE will take $X^{(t)}$ and either guarantee that $X^{(t)}$ is roundable or output coefficients $\alpha^{(t)}, \beta^{(t)}$ certifying the infeasibility of $X^{(t)}$. The algorithm makes use of the information contained in $\alpha^{(t)}, \beta^{(t)}$ by updating the next candidate solution as follows:

$$X^{(t+1)} \stackrel{\text{def}}{=} U_\epsilon \left(\frac{1}{10} \cdot \sum_{i=1}^t M(\alpha^{(i)}, \beta^{(i)}) \right) = E_{\epsilon, D, \Pi} \left(\frac{1}{10} \cdot \sum_{i=1}^t M(\alpha^{(i)}, \beta^{(i)}) \right). \quad (5.2)$$

The algorithm is presented in more detail in Figure 5.4. The good separation oracle ORACLE is given in Figure 5.5, while PROJROUND appears in Figure 5.6.

We want to emphasize at this point that ORACLE will not only be a good separation oracle, but will also have an additional property which will be crucial in proving the correlation condition in Theorem 5.0.2. Using Theorem 3.4.4, we prove that, after a small number of iterations this algorithm either yields a roundable embedding or a feasible solution to $\text{dsdp}(G, b, \Omega(\gamma))$.

Theorem 5.1.9 (Iterations of Oracle, [11]). *Let $\epsilon = 1/32$ and $\delta = \gamma/16$. Assume that the procedure ORACLE is a good separation oracle. Then, after $T = O(\log n/\gamma)$ iterations of the update of Equation 5.2, we either find a roundable solution to $\text{psdp}(G, b, \gamma)$ or a feasible solution $(1/T \sum_{t=1}^T \alpha^{(t)} - \delta, 1/T \sum_{t=1}^T \beta^{(t)})$ to $\text{dsdp}(G, b, 3/16 \cdot \gamma)$.*

Proof. By Theorem 3.4.4, we have that, if ORACLE does not find a roundable solution, after

$$T = O\left(\frac{\gamma \log n}{\delta^2}\right) = O\left(\frac{\log n}{\gamma}\right)$$

rounds, the assignment $(1/T \sum_{t=1}^T \alpha^{(t)} - \delta, 1/T \sum_{t=1}^T \beta^{(t)})$ to the dual variables constitutes a feasible solution for $\text{dsdp}(G, b, 3\gamma/4 - \delta) = \text{dsdp}(G, b, 3\gamma/16)$. \square

Approximate Computation. While we are seeking to construct a nearly-linear-time algorithm, we cannot hope to compute $X^{(t)}$ exactly and explicitly, as just maintaining the full $X^{(t)}$ matrix requires quadratic time in n . Instead, we settle for an approximation $\tilde{X}^{(t+1)}$ to $X^{(t+1)}$ which we define as

$$\tilde{X}^{(t+1)} = \tilde{U}_\epsilon \left(1/10 \cdot \sum_{i=1}^t M(\alpha^{(i)}, \beta^{(i)}) \right).$$

The function \tilde{U}_ϵ is a randomized approximation to U_ϵ obtained by applying the Johnson-Linderstrauss dimension reduction to the embedding corresponding to U_ϵ . \tilde{U}_ϵ is described in full in Section A.2, where we also prove the following lemma about the accuracy and sparsity of the approximation. It is essentially the same argument appearing in [35] applied to our context. We let t_M denote the running time necessary to perform a matrix-vector multiplication by matrix M .

Lemma 5.1.10. *Let $\epsilon = \Theta(1)$. For a matrix $M \in \mathbb{R}^{V \times V}$, $M \succeq 0$, let $\tilde{X} \stackrel{\text{def}}{=} \tilde{U}_\epsilon(M)$ and $X \stackrel{\text{def}}{=} U_\epsilon(M)$. Then, with high probability,*

1. $\tilde{X} \succeq 0$ and $\tilde{X} \in \Delta$.
2. The embedding $\{\tilde{v}_i\}_{i \in V}$ corresponding to \tilde{X} can be represented in $h = O(\log n)$ dimensions.
3. $\{\tilde{v}_i \in \mathbb{R}^h\}_{i \in V}$ can be computed in time $\tilde{O}(t_M + n)$.

4. for any graph $H = (V, E_H)$, with high probability

$$(1 - 1/64) \cdot L(H) \bullet X - \tau \leq L(H) \bullet \tilde{X} \leq (1 + 1/64) \cdot L(H) \bullet X + \tau,$$

and, for any vertex $i \in V$,

$$(1 - 1/64) \cdot R_i \bullet X - \tau \leq R_i \bullet \tilde{X} \leq (1 + 1/64) \cdot R_i \bullet X + \tau,$$

where $\tau \leq O(1/\text{poly}(n))$.

This lemma shows that $\tilde{X}^{(t)}$ is a close approximation to $X^{(t)}$. We will use this lemma to show that ORACLE can receive $\tilde{X}^{(t)}$ as input, rather than $X^{(t)}$, and still meet the conditions of Theorem 5.1.9. In the rest of the paper, we assume that $\tilde{X}^{(t)}$ is represented by its corresponding embedding $\{\tilde{v}_i^{(t)}\}_{i \in V}$.

5.2 Oracle and Proof of the Main Theorem

The Oracle. ORACLE is described in Figure 5.5. We show that ORACLE on input $\tilde{X}^{(t)}$ meets the condition of Theorem 5.1.9. Moreover, we show that ORACLE obeys an additional condition, which, combined with the dual guarantee of Theorem 5.1.9 will yield the correlation property of BALCUT. Under this additional condition, ORACLE not only finds vertices whose vectors in the embedding violate the R_i -constraint, but also finds a cut of conductance $O(\sqrt{\gamma})$ around such vertices.

Theorem 5.2.1 (Main Theorem on ORACLE). *On input $\tilde{X}^{(t)}$, ORACLE runs in time $\tilde{O}(m)$ and is a good separation oracle for $X^{(t)}$ with high probability. Moreover, the cut B in Step 4 is guaranteed to exist.*

Proof of Main Theorem. We are now ready to prove Theorem 5.0.2. To show the overlap condition, we consider the dual condition implied by Theorem 5.1.9 together with the cut $B^{(t)}$ and the values of the coefficients output by the ORACLE.

Proof of Theorem 5.0.2. With high probability ORACLE is a good separation oracle for $X^{(t)}$ at all iterations. Then, at any iteration t , if it finds that the embedding $\{\tilde{v}_i^{(t)}\}_{i \in V}$ corresponding to $\tilde{X}^{(t)}$ is roundable, so that the standard projection rounding PROJROUND produces a cut of balance $\Omega_b(1)$ and conductance $O_b(\sqrt{\gamma})$ with high probability by Theorem 5.1.7. Similarly, if for any t , $C^{(t)}$ is $b/4$ -balanced, BALCUT satisfies the balance condition in Theorem 5.0.2, as $\phi(C^{(t)}) \leq O(\sqrt{\gamma})$ because $C^{(t)}$ is the union of cuts of conductance at most $O(\sqrt{\gamma})$.

1. **Input:** The embedding $\{\tilde{v}_i\}_{i \in V}$, corresponding to $\tilde{X} \in \Delta$. Let $r_i = \|\tilde{v}_i - \tilde{v}_{\text{avg}}\|_2$ for all $i \in V$. Denote $R \stackrel{\text{def}}{=} \{i \in V : r_i^2 \leq 32 \cdot (1-b)/b\}$.
2. **CASE 1:** $\mathbb{E}_{\{i,j\} \in E} \|\tilde{v}_i - \tilde{v}_j\|_2^2 \geq 2\gamma$. Output $\alpha = \gamma$, $\beta = 0$ and $B = \emptyset$.
3. **CASE 2:** not CASE 1 and $\mathbb{E}_{\{i,j\} \sim \mu_R \times \mu_R} \|v_i - v_j\|_2^2 \geq \delta$. Then $\{\tilde{v}_i\}_{i \in V}$ is roundable, as $\tilde{X} \in \Delta$ implies $\mathbb{E}_{j \sim \mu} r_j^2 = 1$.
4. **CASE 3:** not CASE 1 or 2. Relabel the vertices of V such that $r_1 \geq r_2 \geq \dots \geq r_n$ and let $S_i = \{1, \dots, i\}$ be the i -th sweep cut of r . Let z the smallest index such that $\mu(S_z) \geq b/8$. Let B the most balanced sweep cut among $\{S_1, \dots, S_{z-1}\}$ such that $\phi(B) \leq 2048 \cdot \sqrt{\gamma}$. Output $\alpha = 7/8\gamma$, $\beta_i = \mu_i \cdot \gamma$ for $i \in B$ and $\beta_i = 0$ for $i \notin B$. Also output the cut B .

Figure 5.5: ORACLE

Otherwise, after $T = O(\log n/\gamma)$ iterations, by Theorem 5.1.9, we have that $(\alpha' \stackrel{\text{def}}{=} 1/T \sum_{t=1}^T \alpha^{(t)} - \delta, \bar{\beta} \stackrel{\text{def}}{=} 1/T \sum_{t=1}^T \beta^{(t)})$ constitutes a feasible solution $\text{dsdp}(G, b, 3/16 \cdot \gamma)$ with high probability. This implies that $M(\alpha', \bar{\beta}) \succeq 0$, i.e.

$$\frac{1}{m} \cdot L + \sum_{i \in V} \bar{\beta}_i R_i - \alpha' L(K_V) \succeq 0. \quad (5.3)$$

For any cut C such that $\mu(C) \leq 1/2$ and $\phi(C) \leq 3\gamma/64$, let the embedding $\{u_i \in \mathbb{R}\}_{i \in V}$ be defined as $u_i = \sqrt{\mu(\bar{C})/\mu(C)}$ for $i \in C$ and $u_i = -\sqrt{\mu(C)/\mu(\bar{C})}$ for $i \notin C$. Then $u_{\text{avg}} = 0$ and $\mathbb{E}_{i \sim \mu} \|u_i - u_{\text{avg}}\|_2^2 = 1$. Moreover,

$$\mathbb{E}_{\{i,j\} \in E} \|u_i - u_j\|_2^2 = 1/m \cdot |E(C, \bar{C})|/\mu(C)\mu(\bar{C}) \leq 4 \cdot \phi(C) \leq 3\gamma/16.$$

Let U be the Gram matrix of the embedding $\{u_i \in \mathbb{R}\}_{i \in V}$.

We apply the lower bound of Equation 5.3 to U . By Facts 5.1.1 and 5.1.3.

$$\begin{aligned} \mathbb{E}_{\{i,j\} \in E} \|u_i - u_j\|_2^2 + \sum_{i \in V} \bar{\beta}_i \|u_i - u_{\text{avg}}\|_2^2 - \alpha' \mathbb{E}_{i \sim \mu} \|u_i - u_{\text{avg}}\|_2^2 \\ = M(\alpha', \bar{\beta}) \bullet U \geq 0 \end{aligned}$$

Recall that, by the definition of ORACLE, for all $t \in [T]$, $\alpha^{(t)} \geq 7/8 \cdot \gamma$ and $\beta_i^{(t)} = \mu_i \cdot \gamma$ for $i \in B^{(t)}$ and $\beta_i^{(t)} = 0$ for $i \notin B^{(t)}$. Hence, we have $\alpha' = \frac{7}{8}\gamma - \delta = \frac{13}{16}\gamma$, and

$$3\gamma/16 + \gamma/T \cdot \sum_{t=1}^T (\mu(B^{(t)} \cap C) \cdot \mu(\bar{C})/\mu(C) + \mu(B^{(t)} \cap \bar{C}) \cdot \mu(C)/\mu(\bar{C})) - 13/16 \cdot \gamma \geq 0$$

Dividing by γ and using the fact that $\mu(C) \leq 1/2$ and $\mu(\bar{C}) \leq 1$, we obtain

$$1/T \cdot \sum_{t=1}^T \left(\frac{\mu(B^{(t)} \cap C)}{\mu(C)} + \frac{\mu(B^{(t)} \cap \bar{C})}{2 \cdot \mu(\bar{C})} \right) \geq (13/16 - 3/16) = 5/8.$$

Now, recalling that $S = \bigcup_{t=1}^T B^{(t)}$,

$$\frac{\mu(S \cap C)}{\mu(C)} + \frac{\mu(S \cap \bar{C})}{2 \cdot \mu(\bar{C})} \geq 1/T \cdot \sum_{t=1}^T \left(\frac{\mu(B^{(t)} \cap C)}{\mu(C)} + \frac{\mu(B^{(t)} \cap \bar{C})}{2 \cdot \mu(\bar{C})} \right),$$

so that we have

$$\frac{\mu(S \cap C)}{\mu(C)} + \frac{\mu(S \cap \bar{C})}{2 \cdot \mu(\bar{C})} \geq 5/8.$$

As $\mu(S) \leq b/4$, $\mu(S \cap \bar{C})/2 \cdot \mu(\bar{C}) \leq \mu(S) \leq \frac{b}{4} \leq 1/8$. This finally implies that

$$\frac{\mu(S \cap C)}{\mu(C)} \geq 1/2.$$

Moreover, being the union of cuts of conductance $O(\sqrt{\gamma})$, S also has $\phi(S) \leq O(\sqrt{\gamma})$.

Finally, both PROJROUND and ORACLE run in time $\tilde{O}(m)$ as the embedding is $O(\log n)$ dimensional. Notice that, by Fact 2.2.6, at time t , it suffices to compute $\tilde{U}_\epsilon(M)$ for

$$M = 1/10 \cdot \sum_{i=1}^{t-1} \left(\frac{1}{m} \cdot L + \sum_{j \in V} \beta_j^{(i)} R_j \right).$$

By Lemma 5.1.10, this can be done in time t_M . We show that $t_M = O(m)$, as follows. First, $t_L = O(m)$ as L has only $O(m)$ non-zero entries. Secondly, we let the algorithm maintain at each iteration an updated copy of the vector $\beta = \sum_{s=1}^t \beta^{(s)}$ and consider $M' = \sum_{j \in V} \beta_j R_j$. We have $M = L + M'$. To perform a matrix-vector multiplication $M'x$, we observe that, for all $j \in V$,

$$R_j x = (x_j - \sum_{i \in V} \mu_i x_i) = (x_j - x_{\text{avg}}).$$

Pre-computing x_{avg} takes time $O(n)$. After that, we can compute each $\beta_j R_j$ in constant time. Hence, $t_{M'} = O(n)$ and $t_M = O(m)$. Hence, each iteration runs in time $\tilde{O}(m)$, which shows that the total running time is $\tilde{O}(m/\gamma)$ as required. \square

5.3 Proof of Theorem on Oracle

5.3.1 Preliminaries

The following is a variant of the sweep cut argument of Cheeger's Inequality [21], tailored to ensure that a constant fraction of the variance of the embedding is contained inside the output cut. For a vector $x \in \mathbb{R}^V$, let $\text{supp}(x)$ be the set of vertices where x is not zero.

Lemma 5.3.1. *Let $x \in \mathbb{R}^V, x \geq 0$, such that $x^T L x \leq \lambda$ and $\mu(\text{supp}(x)) \leq 1/2$. Relabel the vertices so that $x_1 \geq x_2 \geq \dots \geq x_{z-1} > 0$ and $x_z = \dots = x_n = 0$. For $i \in [z-1]$, denote by $S_i \subseteq V$, the sweep cut $\{1, 2, \dots, i\}$. Further, assume that $\sum_{i=1}^n d_i x_i^2 \leq 1$, and, for some fixed $k \in [z-1]$, $\sum_{i=k}^n d_i x_i^2 \geq \sigma$. Then, there is a sweep cut S_h of x such that $z-1 \geq h \geq k$ and $\phi(S_h) \leq 1/\sigma \cdot \sqrt{2\lambda}$.*

We will also need the following simple fact.

Fact 5.3.2. *Given $v, u, t \in \mathbb{R}^h$, $(\|v - t\|_2 - \|u - t\|_2)^2 \leq \|v - u\|_2^2$.*

5.3.2 Proof of Theorem 5.2.1

The main novelty in ORACLE is found in the analysis of Case 3, where ORACLE finds a cut B of conductance $O(\sqrt{\gamma})$. Then, ORACLE is constrained to output β with $\text{supp}\beta \subseteq B$, while respecting the conditions necessary to be a good separation oracle. The proof of the following theorem shows that this is achieved by ensuring B contains a large constant fraction of the variance of the embedding.

Proof. Notice that, by Markov's Inequality, $\mu(\bar{R}) \leq b/(32 \cdot (1-b)) \leq b/16$. Recall that $\tau = O(1/\text{poly}(n))$.

- **CASE 1:** $\mathbb{E}_{\{i,j\} \in E} \|\tilde{v}_i - \tilde{v}_j\|_2^2 = \frac{1}{m} \cdot L \bullet \tilde{X} \geq 2\gamma$. We have $V(\alpha, \beta) \geq \gamma$ and, by Lemma 5.1.10,

$$M(\alpha, \beta) \bullet X \geq (1 - 1/64) \cdot 2\gamma - \gamma - \tau \geq 1/64 \cdot \gamma \geq 0.$$

- **CASE 2:** $\mathbb{E}_{\{i,j\} \sim \mu_R \times \mu_R} \|v_i - v_j\|_2^2 \geq 1/64$. Then $\{\tilde{v}_i\}_{i \in V}$ is *roundable* by Definition 5.1.6.
- **CASE 3:** $\mathbb{E}_{\{i,j\} \sim \mu_R \times \mu_R} \|v_i - v_j\|_2^2 < 1/64$. This means that, by Fact 5.1.4, $L(K_R) \bullet \tilde{X} < 1/2 \cdot \mu(R)^2 \cdot 1/64 < 1/128$. Hence, by Fact 5.1.3,

$$\begin{aligned} \sum_{i \in \bar{R}} \mu_i R_i \bullet \tilde{X} &= \sum_{i \in \bar{R}} \mu_i r_i \geq \mu(R) - 1/128 \geq 1 - 1/32 - 1/128 \\ &\geq 1 - 5/128. \end{aligned}$$

We then have $\bar{R} = S_g$ for some $g \in [n]$, with $g \leq z$ as $\mu(S_g) \leq \mu(S_z)$. Let $k \leq z$ be the the vertex in \bar{R} such that $\sum_{j=1}^k \mu_j r_j \geq (1 - 1/128) \cdot (1 - 5/128)$ and $\sum_{j=k}^g \mu_j r_j \geq$

$1/128 \cdot (1 - 5/128)$. By the definition of z , we have $k \leq g < z$ and $r_z^2 \leq 8/b \leq 16 \cdot (1-b)/b$. Hence, we have $r_z \leq 1/2 \cdot r_i$, for all $i \geq g$. Define the vector x as $x_i \stackrel{\text{def}}{=} 1/2m \cdot (r_i - r_z)$ for $i \in S_z$ and $r_i \stackrel{\text{def}}{=} 0$ for $i \notin S_z$. Notice that:

$$\begin{aligned} x^T L x &= \sum_{\{i,j\} \in E} (x_i - x_j)^2 \leq 1/2m \cdot \sum_{\{i,j\} \in E} (r_i - r_j)^2 \\ &\stackrel{\text{Fact 5.3.2}}{\leq} 1/2m \cdot \sum_{\{i,j\} \in E} \|\tilde{v}_i - \tilde{v}_j\|_2^2 \leq \gamma. \end{aligned}$$

Also, $x \geq 0$ and $\mu(\text{supp}(x)) \leq b/8 \leq 1/2$, by the definition of z . Moreover,

$$\sum_{i=1}^n d_i x_i^2 = 1/2m \cdot \sum_{i=1}^z d_i (r_i - r_z)^2 \leq 1/2m \cdot \sum_{i=1}^z d_i r_i^2 = 1,$$

and

$$\begin{aligned} \sum_{i=k}^n d_i x_i^2 &= 1/2m \cdot \sum_{i=k}^z d_i (r_i - r_z)^2 \\ &\geq 1/2m \cdot \sum_{i=k}^g d_i (r_i - 1/2 \cdot r_i)^2 \\ &= 1/2m \cdot 1/4 \cdot \sum_{i=k}^g d_i r_i^2 \\ &\geq 1/512 \cdot (1 - 5/128) \geq 1/1024 \end{aligned}$$

Hence, by Lemma 5.3.1, there exists a sweep cut S_h with $z > h \geq k$, such that $\phi(S_h) \leq 2048 \cdot \sqrt{\gamma}$. This shows that B , as defined in Figure 5.5 exists. Moreover, it must be the case that $S_k \subseteq S_h \subseteq B$. As $h \geq k$, we have

$$\sum_{i \in B} \mu_i r_i^2 \geq \sum_{i=1}^k \mu_i r_i^2 \geq (1 - 1/128) \cdot (1 - 5/128) \geq 1 - 3/64.$$

Recall also that, by the construction of z , $\mu(B) \leq b/8$. Hence, we have

$$\begin{aligned} V(\alpha, \beta) &= 7/8 \cdot \gamma - (1-b)/b \cdot \mu(B) \cdot \gamma \geq (7/8 - 1/8) \cdot \gamma \geq 3/4\gamma. \\ M(\alpha, \beta) \bullet X &\geq (1 - 1/64) \cdot (1 - 3/64)\gamma - 7/8\gamma - \tau \geq 1/64 \cdot \gamma \geq 0 \end{aligned}$$

This completes all the three cases. Notice that in every case we have:

$$1/m \cdot L - \gamma L(K_V) \preceq M(\alpha, \beta) \preceq 1/m \cdot L + \gamma L(K_V).$$

Hence,

$$-\gamma L(K_V) \preceq M(\alpha, \beta) \preceq 5L(K_V),$$

as $L \preceq 4m \cdot L(K_V)$. Finally, using the fact that $\{\tilde{v}_i\}_{i \in V}$ is embedded in $O(\log n)$ dimensions, we can compute $L \bullet \tilde{X}$ in time $\tilde{O}(m)$. $L(K_R) \bullet \tilde{X}$ can also be computed in time $\tilde{O}(n)$ by using the decomposition $\mathbb{E}_{\{i,j\} \sim \mu_R \times \mu_R} \|v_i - v_j\|_2^2 = 2 \cdot \mathbb{E}_{i \sim \mu_R} \|v_i - v_{\text{avg}_R}\|_2^2$, where v_{avg_R} is the mean of vectors representing vertices in R . The sweep cut over r takes time $\tilde{O}(m)$. Hence, the total running time is $\tilde{O}(m)$. \square

5.4 Random Walk Interpretation

In this section, we give a novel detailed interpretation of our algorithm in terms of certain random walks over the instance graph. This view provides an intuition of why BALCUT and its SDP approach are successful at finding sparse balanced cuts. We start by defining the concept of *accelerated heat kernel*: this is a heat-kernel random walk in which certain vertices have increased probability-mass leakage towards the rest of the graph. We show that the updates $X^{(t)}$ computed by BALCUT can be seen as representing the probability transition matrix of accelerated-heat-kernel random walks. Intuitively, the vertices are contained in some unbalanced sparse cut have their rate increased, allowing probability to mix more quickly across such cut, so that different cuts become the next obstacles to mixing. The analysis of Section 3.4 allows us to argue that after at most $\tilde{O}(\log n / \gamma)$ iterations, this procedure either finds an accelerated-heat-kernel random walk whose slow mixing is due to a sparse balanced cut - in which case we can recover such cut - or a certificate that all balanced cuts have sufficiently large conductance.

Accelerated Heat Kernel

Definition 5.4.1. An accelerated heat-kernel process with rate vector $\beta \geq 0$ on the instance graph G is the continuous-time Markov process defined by transition rate matrix

$$Q_\beta = - \left(L + \sum_{i \in V} \beta_i d_i L(S_i) \right) D^{-1}.$$

Hence, it has probability transition matrix $P_\beta(t)$ at time t :

$$P_\beta(t) = e^{-t(L + \sum_{i \in V} \beta_i d_i L(S_i))D^{-1}}.$$

It is easy to verify that Q_β respects the necessary conditions to be a transition rate matrix. In particular, the rates are balanced as $\bar{1}^T Q_\beta = 0$ and Q_β has positive off-diagonal entries and negative diagonal entries. Moreover, it is possible to verify that μ , the uniform distribution weighted by the edge measure, is the unique stationary distribution for $P_\beta(t)$.

Unfortunately, $P_\beta(t)$ is not equal to the heat kernel on the modified graph $L + \sum_{i \in V} d_i \beta_i L(S_i)$ as the normalization by D^{-1} does not match the degree of $L(S_i)$. However, $P_\beta(t)$ still has an interesting interpretation. The best way to understand the behavior of $P_\beta(t)$ is to compare it with the heat-kernel process by considering the differential equation characterizing the accelerated heat-kernel process. Let $p(t)$ be the probability distribution of an accelerated heat-kernel process at time t . Then, for all $j \in V$,

$$\begin{aligned} \left(\frac{\partial p(t)}{\partial t} \right)_j &= - \left(LD^{-1}p(t) + \sum_{i \in V} d_i \beta_i L(S_i) D^{-1}p(t) \right)_j = \\ &- \left(p(t)_j - \sum_{\{i,j\} \in E} \frac{p(t)_i}{d_i} \right) - d_j \beta_j \left(\frac{p(t)_j}{d_j} - \sum_{i \in V} \frac{p(t)_i}{2m} \right) = \\ &- \left(p(t)_j - \sum_{\{i,j\} \in E} \frac{p(t)_i}{d_i} \right) - \beta_j (p(t)_j - \mu_j). \end{aligned} \quad (5.4)$$

While the first term of this expression is the same as for the heat kernel, the second term shows us how the accelerated heat-kernel yields a larger out-rate at vertices where $p(t)_j$ is far from μ_j . The parameter β_j controls the magnitude of this increase in rate. From this discussion, it should be clear that the accelerated heat-kernel displays a faster converge to uniform than the heat kernel, particularly at vertices with large β values. This fact is exploited crucially by BALCUT.

Vector Embedding as Accelerated Heat Kernel We now show that the update $X^{(t)}$, which BALCUT approximates, is strictly related to the transition matrix of an accelerated heat kernel. This is an easy consequence of Definition 5.4.1. In the following, we let Z denote the factor used to normalize for $X^{(t)}$ such that $L(K_V) \bullet X^{(t)} = 1$. The embedding corresponding to $X^{(t)}$ can be recovered from the columns of $(X^{(t)})^{1/2}$. We have:

$$\begin{aligned} (X^{(t)})^{1/2} &= \frac{1}{\sqrt{Z}} \cdot D^{-1} (1 - \epsilon)^{m/10 \cdot \sum_{i=1}^{t-1} \left(\frac{1}{m} L + \sum_{j \in B^{(i)}} \beta_j^{(i)} R_j \right) D^{-1}} = \\ &\frac{1}{\sqrt{Z}} \cdot D^{-1} \cdot e^{-\frac{\log(1-\epsilon)}{10} t (LD^{-1} + \frac{1}{2t} \cdot \sum_{i \in V} d_i \beta_i R_i D^{-1})} \end{aligned}$$

where $\beta = D^{-1} \sum_{i=1}^{t-1} \beta^{(i)}$. Notice that our choice of $(X^{(t)})^{1/2}$ is symmetric. Hence, if we define the rate vector

$$q \stackrel{\text{def}}{=} 1/2t \cdot \beta \in \mathbb{R}^n,$$

and the time

$$t' = -\frac{\log(1-\epsilon)}{10} t,$$

we see that the embedding $\{v_i^{(t)}\}_{i \in V}$ corresponding to $X^{(t)}$ is given by

$$v_i^{(t)} = \frac{1}{\sqrt{Z}} \cdot D^{-1} P_q(t') e_i.$$

It is convenient to think of $v_i^{(t)}$ as a scaled version of the vector that at entry i contains the amount of probability mass on edges adjacent to vertex i under the distribution $P_q(t') e_i$. Following this view, the quantity

$$d_i \|v_i^{(t)} - \frac{1}{2m} \vec{1}\|^2,$$

which plays an important part below, is just the ℓ_2 -distance over the edges of the distribution from the stationary. Moreover, because $(X^{(t)})^{1/2}$ is symmetric, the i th-row of the embedding $(X^{(t)})^{1/2}$ also equals $v_i^{(t)}$. This will facilitate the expression of some quantities of interest in the next paragraph.

Algorithm Interpretation Given this interpretation of the embedding produced by BAL-CUT at every step, we can verify that the psdp is asking the algorithm to find a rate vector q such that:

- On average over all starting points in V , the random walks display low mixing, i.e.

$$\sum_{i \in V} \left(v_i^{(t)}\right)^T L v_i^{(t)} \leq 4\gamma m \cdot \sum_{i \in V} \left(v_i^{(t)}\right)^T L(K_V) v_i^{(t)} = 2\gamma \cdot \sum_{i \in V} d_i \|v_i^{(t)} - \frac{1}{2m} \vec{1}\|^2.$$

- No single vertex i contributes too much to the total ℓ_2 -distance of the vectors $\{v_i^{(t)}\}$ from the stationary distribution, i.e. for all $i \in V$,

$$\|v_i^{(t)} - \frac{1}{2m} \vec{1}\|^2 \leq \frac{1-b}{b} \cdot \sum_{i \in V} \mu_i \|v_i^{(t)} - \frac{1}{2m} \vec{1}\|^2.$$

If the current rate q does not satisfy the first requirement the time t' is increased so that the mixing of the random walk decreases. This corresponds to case 1 in the description of ORACLE. The more interesting case is when enough of the other constraints are violated so that the embedding is not roundable, i.e. it is not possible to recover a sparse balanced cut from it. The analysis of ORACLE shows that this is the case when a small fraction of vertices contribute a large constant fraction of the total distance from the stationary distribution.

Under our random-walk interpretation, this means that the for a small fraction of starting vertices $R = \{r_j\}$, the walk $P_q(t') r_j$ is very far from stationary. Using this fact, together with Cheeger's Inequality, we can find an unbalanced sparse cut $B \subseteq R$ that is also responsible for most of the distance from the stationary distribution. At this point, we want to modify

q such that the random walk $P_q(t)$ will be somehow able to bypass B and highlight some different, hopefully balanced, cuts in G . To do so, BALCUT increases the rate out of each vertex in B by increasing the entries q_{ji} for $i \in B$. By Equation 5.4, this ensures that probability mass is leaked towards the uniform distribution faster at vertices in B , which allows the new walk to better mix across cut B . Finally, by the primal-dual analysis, the fact that B contributed to a large fraction of the total distance from uniform allows BALCUT to make sufficient progress at every iteration to achieve the guarantee of Theorem 5.0.2 after $O(\log n/\gamma)$ rounds.

5.5 Other Proofs

5.5.1 Proof of Basic Lemmata

Proof of Lemma 5.1.5. For a b -balanced cut (S, \bar{S}) with $\phi(S) \leq \gamma$. Without loss of generality, assume $\mu(S) \leq 1/2$. Consider the one-dimensional solution assigning $v_i = \sqrt{\mu(\bar{S})/\mu(S)}$ to $i \in S$ and $v_i = -\sqrt{\mu(S)/\mu(\bar{S})}$ to $i \in \bar{S}$. Notice that $v_{\text{avg}} = 0$ and that $\|v_i - v_j\|_2^2 = 1/\mu(S)\mu(\bar{S})$ for $i \in S, j \notin S$. We then have:

-

$$\begin{aligned} \mathbb{E}_{\{i,j\} \in E} \|v_i - v_j\|_2^2 &= \frac{1}{m} \cdot \frac{|E(S, \bar{S})|}{\mu(S)\mu(\bar{S})} \leq 2 \cdot \frac{|E(S, \bar{S})|}{2m \cdot \mu(S)\mu(\bar{S})} \\ &\leq 4 \cdot \phi(S) \leq 4\gamma. \end{aligned}$$

-

$$\mathbb{E}_{i \sim \mu} \|v_i - v_{\text{avg}}\|_2^2 = \mu(S) \cdot \mu(\bar{S})/\mu(S) + \mu(\bar{S}) \cdot \mu(S)/\mu(\bar{S}) = 1.$$

- for all $i \in V$,

$$\|v_i - v_{\text{avg}}\|_2^2 \leq \frac{\mu(\bar{S})}{\mu(S)} \leq \frac{1-b}{b},$$

where the last inequality follows as S is b -balanced.

□

5.5.2 Projection Rounding

The description of the rounding algorithm PROJROUND is given in Figure 5.6. We remark that during the execution of BALCUT the embedding $\{v_i \in \mathbb{R}^h\}_{i \in V}$ will be represented by a projection over $h = O(\log n)$ random directions, so that it will suffice to take a balanced sweep cut of each coordinate vector. We now present the proof of Theorem 5.1.7. The constants in this argument were not optimized to preserve the simplicity of the proof.

1. **Input:** An embedding $\{v_i \in \mathbb{R}^h\}_{i \in V}$, $b \in (0, 1/2]$.
2. Let $c = \Omega_b(1)$ be a constant to be fixed in the proof.
3. For $t = 1, 2, \dots, O(\log n)$:
 - (a) Pick a unit vector u uniformly at random from \mathbb{S}^{h-1} and let $x \in \mathbb{R}^n$ with $x_i \stackrel{\text{def}}{=} \sqrt{h} \cdot u^T v_i$.
 - (b) Sort the vector x . Assume w.l.o.g. that $x_1 \geq x_2 \geq \dots \geq x_n$. Define $S_i \stackrel{\text{def}}{=} \{j \in [n] : x_j \geq x_i\}$.
 - (c) Let $S^{(t)} \stackrel{\text{def}}{=} (S_i, \bar{S}_i)$ which minimizes $\phi(S_i)$ among sweep-cuts for which $\text{vol}(S_i) \in [c \cdot 2m, (1 - c) \cdot 2m]$.
4. **Output:** The cut $S^{(t)}$ of least conductance over all choices of t .

Figure 5.6: PROJROUND

Preliminaries.

We will make use of the following simple facts. Recall that for $y \in \mathbb{R}^h$, $\text{sgn}(y) = 1$ if $y \geq 0$, and -1 otherwise.

Fact 5.5.1. For all $y, z \in \mathbb{R}$, $(y + z)^2 \leq 2(y^2 + z^2)$.

Fact 5.5.2. For all $y \geq z \in \mathbb{R}$, $|\text{sgn}(y) \cdot y^2 - \text{sgn}(z) \cdot z^2| \leq (y - z)(|y| + |z|)$.

Proof.

1. If $\text{sgn}(y) = \text{sgn}(z)$, then $|\text{sgn}(y) \cdot y^2 - \text{sgn}(z) \cdot z^2| = |y^2 - z^2| = (y - z) \cdot |y + z| = (y - z)(|y| + |z|)$ as $y \geq z$.
2. If $\text{sgn}(y) \neq \text{sgn}(z)$, then since $y \geq z$, $(y - z) = |y| + |z|$. Hence, $|\text{sgn}(y) \cdot y^2 - \text{sgn}(z) \cdot z^2| = y^2 + z^2 \leq (|y| + |z|)^2 = (y - z)(|y| + |z|)$.

□

Fact 5.5.3. For all $y \geq z \in \mathbb{R}$, $(y - z)^2 \leq 2(\text{sgn}(y) \cdot y^2 - \text{sgn}(z) \cdot z^2)$.

Proof.

1. If $\text{sgn}(y) = \text{sgn}(z)$, $(y - z)^2 = y^2 + z^2 - 2yz \leq y^2 + z^2 - 2z^2 = y^2 - z^2$ as $y \geq z$. Since $\text{sgn}(y) = \text{sgn}(z)$, $y^2 - z^2 \leq 2(\text{sgn}(y) \cdot y^2 - \text{sgn}(z) \cdot z^2)$.
2. If $\text{sgn}(y) \neq \text{sgn}(z)$, $(y - z)^2 = (|y| + |z|)^2 \leq 2(|y|^2 + |z|^2) = 2(\text{sgn}(y) \cdot y^2 - \text{sgn}(z) \cdot z^2)$. Here, we have used Fact 5.5.1.

□

We also need the following standard facts.

Fact 5.5.4. *Let $v \in \mathbb{R}^h$ be a vector of length ℓ and u a unit vector chosen uniformly at random in \mathbb{S}^{h-1} . Then,*

1. $\mathbb{E}_u (v^T u)^2 = \frac{\ell^2}{h}$, and
2. for $0 \leq \delta \leq 1$, $\mathbb{P}_u \left[\sqrt{h} \cdot |v^T u| \leq \delta \ell \right] \leq 3\delta$.

Fact 5.5.5. *Let Y be a non-negative random variable such that $\mathbb{P}[Y \leq K] = 1$ and $\mathbb{E}[Y] \geq \delta$. Then,*

$$\mathbb{P}[Y \geq \delta/2] \geq \frac{\delta}{2K}.$$

The following lemma about projections will be crucial in the proof of Theorem 5.1.7. It is a simple adaptation of an argument appearing in [12].

Lemma 5.5.6 (Projection). *Given a roundable embedding $\{v_i \in \mathbb{R}^h\}_{i \in V}$, consider the embedding $x \in \mathbb{R}^n$ such that $x_i \stackrel{\text{def}}{=} \sqrt{d} \cdot u^T v_i$, where $u \in \mathbb{S}^{h-1}$, and assume without loss of generality that $x_1 \geq \dots \geq x_n$. Then, there exists $c \in (0, b]$ such that with probability $\Omega_b(1)$ over the choice of $u \in \mathbb{S}^{h-1}$, the following conditions hold simultaneously:*

1. $\mathbb{E}_{\{i,j\} \in E} (x_i - x_j)^2 \leq O_b \left(\mathbb{E}_{\{i,j\} \in E} \|v_i - v_j\|^2 \right) = O_b(\gamma)$,
2. $\mathbb{E}_{i \sim \mu} (x_i - x_{\text{avg}})^2 = O_b(1)$, and
3. *there exists $1 \leq l \leq n$ with $\text{vol}(\{1, \dots, l\}) \geq c \cdot \text{vol}(G)$ and, there exists $l \leq r \leq n$ such that $\text{vol}(\{r, \dots, n\}) \geq c \cdot \text{vol}(G)$ such that $x_l - x_r \geq \Omega_b(1)$.*

Proof. We are going to lower bound the probability, over u , of each of (1), (2) and (3) in the lemma and then apply the union bound.

Part (1). By applying Fact 5.5.4 to $v = v_i - v_j$ and noticing $\sqrt{h} \cdot |v^T u| = |x_i - x_j|$, we have

$$\mathbb{E}_u \mathbb{E}_{\{i,j\} \in E} (x_i - x_j)^2 = \mathbb{E}_{\{i,j\} \in E} \|v_i - v_j\|^2.$$

Hence, by Markov's Inequality, for some p_1 to be fixed later

$$\mathbb{P}_u \left[\mathbb{E}_{\{i,j\} \in E} (x_i - x_j)^2 \geq 1/p_1 \cdot \mathbb{E}_{\{i,j\} \in E} \|v_i - v_j\|^2 \right] \leq p_1.$$

Part (2).

$$\mathbb{E}_u \mathbb{E}_{i \sim \mu} (x_i - x_{\text{avg}})^2 \stackrel{\text{Fact 5.5.4-(1)}}{=} \mathbb{E}_u \mathbb{E}_{i \sim \mu} \|v_i - v_{\text{avg}}\|^2 \stackrel{\text{roundability}}{=} 1.$$

Hence, for some p_2 be fixed later

$$\mathbb{P}_u \left[\mathbb{E}_{i \sim \mu} (x_i - x_{\text{avg}})^2 \geq 1/p_2 \cdot \mathbb{E}_{i \sim \mu} \|v_i - v_{\text{avg}}\|^2 \right] \leq p_2.$$

Part (3). Let $R \stackrel{\text{def}}{=} \{i \in V : \|v_i - v_{\text{avg}}\|^2 \leq 32 \cdot (1-b)/b\}$. Let $\sigma \stackrel{\text{def}}{=} 4 \cdot \sqrt{2} \sqrt{(1-b)/b}$. By Markov's Inequality, $\mu(\bar{R}) \leq 1/\sigma^2$. As $\{v_i\}_{i \in V}$ is roundable, for all $i, j \in R$, $\|v_i - v_j\| \leq 2\sigma$. Hence, $\|v_i - v_j\| \geq 1/2\sigma \cdot \|v_i - v_j\|^2$ for such $i, j \in R$. This, together with the roundability of $\{v_i\}_{i \in V}$, implies that

$$\mathbb{E}_{\{i,j\} \sim \mu_R \times \mu_R} \|v_i - v_j\| \geq 1/128\sigma.$$

For any $k \in R$, we can apply the triangle inequality for the Euclidean norm as follows

$$\begin{aligned} \mathbb{E}_{\{i,j\} \sim \mu_R \times \mu_R} \|v_i - v_j\| &\leq \mathbb{E}_{\{i,j\} \sim \mu_R \times \mu_R} (\|v_i - v_k\| + \|v_k - v_j\|) \\ &\leq 2 \cdot \mathbb{E}_{i \sim \mu_R} \|v_i - v_k\|. \end{aligned}$$

Hence, for all $k \in R$

$$\mathbb{E}_{i \sim \mu_R} \|v_i - v_k\| \geq 1/256\sigma.$$

Let R_k be the set $\{i \in R : \|v_i - v_k\| \geq 1/512\sigma\}$. Since $\|v_i - v_k\| \leq 2\sigma$, applying Fact 5.5.5 yields that, for all $k \in R$,

$$\mathbb{P}_{i \sim \mu_R} [i \in R_k] \geq 1/1024\sigma^2.$$

For all vertices $i \in R_k$, by Fact 5.5.4

$$\mathbb{P}_u [|x_k - x_i| \geq 1/9 \cdot 1/512\sigma = 1/4608\sigma] \geq \frac{2}{3}.$$

Let $\delta \stackrel{\text{def}}{=} 1/2 \cdot 1/4608\sigma = 1/9216\sigma$. Consider the event $\mathcal{E} \stackrel{\text{def}}{=} \{i \in R_k \wedge |x_i - x_k| \geq 2 \cdot \delta\}$. Then,

$$\begin{aligned} \mathbb{P}_{u, \{i,k\} \sim \mu_R \times \mu_R} [\mathcal{E}] &= \mathbb{P}_{\{i,k\} \sim \mu_R \times \mu_R} [i \in R_k] \cdot \mathbb{P}_u [|x_i - x_k| \geq 2 \cdot \delta \mid i \in R_k] \\ &\geq \frac{1}{1024\sigma^2} \cdot \frac{2}{3} = \frac{1}{1536\sigma^2} \stackrel{\text{def}}{=} \rho. \end{aligned}$$

Hence, from Fact 5.5.5, with probability at least $\rho/2$ over directions u , for a fraction $\rho/2$ of pairs $\{i, k\} \in R \times R$, $|x_k - x_i| \geq 2 \cdot \delta$. Let ν be the median value of $\{x_i\}_{i \in V}$. Let $L \stackrel{\text{def}}{=} \{i : x_i \leq \nu - \delta\}$ and $H \stackrel{\text{def}}{=} \{i : x_i \geq \nu + \delta\}$. Any pair $\{i, j\} \in R \times R$ with $|x_i - x_j| \geq 2 \cdot \delta$ has at least one vertex in $L \cup H$. Hence,

$$\mu(L \cup H) \geq 1/2 \cdot \rho/2 \cdot \mu(R)^2 \geq \rho/4 \cdot (\sigma^2 - 1/\sigma^2)^2 \geq \rho/16 = \Omega_b(1).$$

Assume $\mu(L) \geq \rho/32$, otherwise, apply the same argument to H . Let l be the largest index in L . For all $i \in L$ and j such that $x_j \geq \nu$, we have $|x_i - x_j| \geq \delta$. (Similarly, let r be the smallest index in H .) This implies that,

$$|x_l - x_{\lfloor n/2 \rfloor}| \geq \delta$$

with probability at least $\rho/2 = \Omega_b(1)$, satisfying the required condition. Let p_3 be the probability that this event does not take place. Then,

$$p_3 \leq 1 - \rho/2.$$

To conclude the proof, notice that the probability that all three conditions do not hold simultaneously is, by a union bound, at most $p_1 + p_2 + p_3$. Setting $p_1 = p_2 = \rho/5 = \Omega_b(1)$, we satisfy the first and third conditions and obtain

$$p_1 + p_2 + p_3 \leq 1 - \rho \cdot (1/2 - 1/5 - 1/5) \leq 1 - \rho/10.$$

Hence, all conditions are satisfied at the same time with probability at least $\rho/10 = \Omega_b(1)$. \square

From this proof, it is possible to see that the parameter c in our rounding scheme should be set to $\rho/32$. We are now ready to give a proof of Theorem 5.1.7. It is essentially a variation of the proof of Cheeger's Inequality, tailored to produce balanced cuts.

Proof of Theorem 5.1.7. For this proof, assume that x has been translated so that $x_{\text{avg}} = 0$. Notice that the guarantees of 5.5.6 still apply. Let x, l, r and c be as promised by Lemma 5.5.6. For $z \in \mathbb{R}$, let $\text{sgn}(z)$ be 1 if $z \geq 0$ and -1 otherwise. Let

$$y_i \stackrel{\text{def}}{=} \text{sgn}(x_i) \cdot x_i^2.$$

Hence,

$$\begin{aligned} \mathbb{E}_{\{i,j\} \in E} |y_i - y_j| &\stackrel{\text{Fact 5.5.2}}{\leq} \mathbb{E}_{\{i,j\} \in E} (|x_i - x_j|) \cdot (|x_i| + |x_j|) \\ &\leq \sqrt{\mathbb{E}_{\{i,j\} \in E} (x_i - x_j)^2 \cdot \mathbb{E}_{\{i,j\} \in E} (|x_i| + |x_j|)^2} \\ &\stackrel{\text{Fact 5.5.1}}{\leq} \sqrt{2 \cdot \mathbb{E}_{\{i,j\} \in E} (x_i - x_j)^2 \cdot \mathbb{E}_{\{i,j\} \in E} (x_i^2 + x_j^2)} \\ &= \sqrt{2 \cdot \mathbb{E}_{\{i,j\} \in E} (x_i - x_j)^2 \cdot \frac{2m}{m} \cdot \mathbb{E}_{i \sim \mu} x_i^2} \\ &= \sqrt{4 \cdot \mathbb{E}_{\{i,j\} \in E} (x_i - x_j)^2 \cdot \mathbb{E}_{i \sim \mu} x_i^2} \\ &\stackrel{\text{Lemma 5.5.6-(1),(2)}}{\leq} O_b(\sqrt{\gamma}). \end{aligned}$$

Now we lower bound $\mathbb{E}_{\{i,j\} \in E} |y_i - y_j|$. Notice that if $x_i \geq x_j$, then $y_i \geq y_j$ and vice-versa. Hence,

$$y_1 \geq \dots \geq y_n.$$

Let $S_i \stackrel{\text{def}}{=} \{1, \dots, i\}$ and let ϕ be the minimum conductance of S_i over all $l \leq i \leq r$

$$\begin{aligned} \mathbb{E}_{\{i,j\} \in E} |y_i - y_j| &= \frac{1}{|E|} \sum_{i=1}^{n-1} |E(S_i, \bar{S}_i)| \cdot (y_i - y_{i+1}) \\ &\geq \phi \cdot \sum_{l \leq i \leq r} \frac{\min\{\text{vol}(S_i), \text{vol}(\bar{S}_i)\}}{|E|} (y_i - y_{i+1}) \\ &\stackrel{\text{Lemma 5.5.6-(3)}}{=} \Omega_b(1) \cdot \phi \cdot \sum_{l \leq i \leq r} (y_i - y_{i+1}) \\ &\geq \Omega_b(1) \cdot \phi \cdot (y_l - y_r) \\ &\stackrel{\text{Fact 5.5.3}}{\geq} \Omega_b(1) \cdot \phi \cdot (x_l - x_r)^2 \\ &\stackrel{\text{Lemma 5.5.6}}{\geq} \Omega_b(1) \cdot \phi. \end{aligned}$$

Hence, $\phi \leq O_b(\sqrt{\gamma})$ with constant probability over the choice of projection vectors u . Repeating the projection $O(\log n)$ times and picking the best balanced cut found yields a high probability statement. Finally, as the embedding is in h dimensions, it takes $\tilde{O}(nh)$ time to compute the projection. After that, the one-dimensional embedding can be sorted in time $\tilde{O}(n)$ and the conductance of the relevant sweep cuts can be computed in time $O(m)$, so that the total running time is $\tilde{O}(nh + m)$. \square

Bibliography

- [1] Rudolf Ahlswede and Andreas Winter. Strong converse for identification via quantum channels. *IEEE Transactions on Information Theory*, 48(3):569–579, 2002.
- [2] Noga Alon and V. D. Milman. λ_1 , isoperimetric inequalities for graphs, and superconcentrators. *J. Comb. Theory, Ser. B*, 38(1):73–88, 1985.
- [3] Christoph Ambuhl, Monaldo Mastrolilli, and Ola Svensson. Inapproximability results for sparsest cut, optimal linear arrangement, and precedence constrained scheduling. In *FOCS'07: Proc. 48th Ann. IEEE Symp. Foundations of Computer Science*, pages 329–337, 2007.
- [4] Reid Andersen, Fan R. K. Chung, and Kevin J. Lang. Local graph partitioning using pagerank vectors. In *FOCS'06: Proc. 47th Ann. IEEE Symp. Foundations of Computer Science*, pages 475–486, 2006.
- [5] Reid Andersen and Yuval Peres. Finding sparse cuts locally using evolving sets. In *STOC '09: Proc. 41st Ann. ACM Symp. Theory of Computing*, pages 235–244, 2009.
- [6] Serge A. Plotkin, David B. Shmoys, and Éva Tardos. Fast approximation algorithms for fractional packing and covering problems. In *FOCS'91: Proc. 32nd Ann. IEEE Symp. Foundations of Computer Science*, pages 495–504, 1991.
- [7] Sanjeev Arora, Boaz Barak, and David Steurer. Subexponential algorithms for unique games and related problems. In *FOCS'11: Proc. 52nd Ann. IEEE Symp. Foundations of Computer Science*, 2010.
- [8] Sanjeev Arora, Elad Hazan, and Satyen Kale. $O(\sqrt{\log n})$ approximation to sparsest cut in $\tilde{O}(n^2)$ time. In *FOCS'04: Proc. 45th Ann. IEEE Symp. Foundations of Computer Science*, pages 238–247, 2004.
- [9] Sanjeev Arora, Elad Hazan, and Satyen Kale. $O(\sqrt{\log n})$ approximation to sparsest cut in $\tilde{O}(n^2)$ time. In *FOCS'04: Proc. 45th Ann. IEEE Symp. Foundations of Computer Science*, volume 00, pages 238–247, 2004.

- [10] Sanjeev Arora, Elad Hazan, and Satyen Kale. The multiplicative weights update method: a meta algorithm and applications, 2005. Manuscript.
- [11] Sanjeev Arora and Satyen Kale. A combinatorial, primal-dual approach to semidefinite programs. In *STOC '07: Proc. 39th Ann. ACM Symp. Theory of Computing*, pages 227–236, 2007.
- [12] Sanjeev Arora, Satish Rao, and Umesh Vazirani. Expander flows, geometric embeddings and graph partitioning. In *STOC '04: Proc. 36th Ann. ACM Symp. Theory of Computing*, pages 222–231, 2004.
- [13] Joshua D. Batson, Daniel A. Spielman, and Nikhil Srivastava. Twice-ramanujan sparsifiers. In *STOC '09: Proc. 41st Ann. ACM Symp. Theory of Computing*, pages 255–262, 2009.
- [14] András A. Benczúr and David R. Karger. Approximating s-t minimum cuts in $\tilde{O}(n^2)$ time. In *STOC '96: Proc. 28th Ann. ACM Symp. Theory of Computing*, pages 47–55, 1996.
- [15] Rajendra Bhatia. *Matrix Analysis (Graduate Texts in Mathematics)*. Springer, 1996.
- [16] G. W. Brown and J. von Neumann. Solutions of games by differential equations. *Annals of Mathematics Studies*, 24:73–79, 1950.
- [17] George W. Brown. Iterative solution of games by fictitious play. *Activity Analysis of Production and Allocation*, pages 374–376, 1951.
- [18] Nicolo Cesa-Bianchi and Gabor Lugosi. *Prediction, Learning, and Games*. Cambridge University Press, 2006.
- [19] Shuchi Chawla, Robert Krauthgamer, Ravi Kumar, Yuval Rabani, and D. Sivakumar. On the hardness of approximating multicut and sparsest-cut. *Comput. Complex.*, 15(2):94–114, 2006.
- [20] P. Christiano, J. A. Kelner, A. Madry, D. A. Spielman, and S.-H. Teng. Electrical flows, Laplacian systems, and faster approximation of maximum flow in undirected graphs. *ArXiv e-prints*, arXiv:1010.2921v2 [cs.DS], 2010.
- [21] Fan R.K. Chung. *Spectral Graph Theory (CBMS Regional Conference Series in Mathematics, No. 92)*. American Mathematical Society, 1997.
- [22] Samuel I. Daitch and Daniel A. Spielman. Faster approximate lossy generalized flow via interior point algorithms. pages 451–460, 2008.

- [23] Nikhil R. Devanur, Subhash A. Khot, Rishi Saket, and Nisheeth K. Vishnoi. Integrality gaps for sparsest cut and minimum linear arrangement problems. In *STOC '06: Proc. 38th Ann. ACM Symp. Theory of Computing*, pages 537–546, 2006.
- [24] Lisa K. Fleischer. Approximating fractional multicommodity flow independent of the number of commodities. *SIAM J. Discret. Math.*, 13(4):505–520, 2000.
- [25] Yoav Freund and Robert E. Schapire. A decision-theoretic generalization of on-line learning and an application to boosting. *Journal of Computer and System Sciences*, 55(1):119 – 139, 1997.
- [26] M. R. Garey and David S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness*. W. H. Freeman, 1979.
- [27] Naveen Garg and Jochen Könemann. Faster and simpler algorithms for multicommodity flow and other fractional packing problems. In *FOCS'98: Proc. 39th Ann. IEEE Symp. Foundations of Computer Science*, pages 300–309, 1998.
- [28] Andrew V. Goldberg and Satish Rao. Beating the flow decomposition barrier. *J. ACM*, 45:783–797, 1998.
- [29] Gene H. Golub and Charles F. van Loan. *Matrix computations (3. ed.)*. Johns Hopkins University Press, 1996.
- [30] Stephen Guattery and Gary L. Miller. On the performance of spectral graph partitioning methods. In *SODA'95: Proc. 6th Ann. ACM-SIAM Symp. Discrete Algorithms*, pages 233–242, 1995.
- [31] G. Iyengar, David J. Phillips, and Clifford Stein. Approximation algorithms for semidefinite packing problems with applications to maxcut and graph coloring. In *IPCO'05: Proc. 11th Conf. Integer Programming and Combinatorial Optimization*, pages 152–166, 2005.
- [32] Rahul Jain, Zhengfeng Ji, Sarvagya Upadhyay, and John Watrous. Qip = pspace. In *STOC '10: Proc. 42nd Ann. ACM Symp. Theory of Computing*, pages 573–582, 2010.
- [33] Mark Jerrum and Alistair Sinclair. Approximating the permanent. *SIAM J. Comput.*, 18(6):1149–1178, 1989.
- [34] Satyen Kale. Efficient algorithms using the multiplicative weights update method. Technical report, Princeton University, Department of Computer Science, 2007.
- [35] Satyen Kale. Efficient algorithms using the multiplicative weights update method. Technical report, Princeton University, Department of Computer Science, 2007.

- [36] R. Kannan, S. Vempala, and A. Vetta. On clusterings-good, bad and spectral. In *FOCS'00: Proc. 41st Ann. IEEE Symp. Foundations of Computer Science*, page 367, 2000.
- [37] George Karypis and Vipin Kumar. A fast and high quality multilevel scheme for partitioning irregular graphs. *SIAM Journal on Scientific Computing*, 20:359–392, 1999.
- [38] George Karypis and Vipin Kumar. A fast and high quality multilevel scheme for partitioning irregular graphs. *SIAM Journal on Scientific Computing*, 20:359–392, 1999.
- [39] Jonathan A. Kelner and Aleksander Madry. Faster generation of random spanning trees. In *FOCS'09: Proc. 50th Ann. IEEE Symp. Foundations of Computer Science*, pages 13–21, 2009.
- [40] Rohit Khandekar. Lagrangian relaxation based algorithms for convex programming problems. Technical report, Indian Institute of Technology Delhi, 2004.
- [41] Rohit Khandekar, Satish Rao, and Umesh Vazirani. Graph partitioning using single commodity flows. In *STOC '06: Proc. 38th Ann. ACM Symp. Theory of Computing*, pages 385–390, 2006.
- [42] Rohit M. Khandekar, Subhash Khot, Lorenzo Orecchia, and Nisheeth K. Vishnoi. On a cut-matching game for the sparsest cut problem. Technical Report EECS-2007-177, EECS Department, University of California, 2007.
- [43] Subhash Khot. On the power of unique 2-prover 1-round games. In *STOC '02: Proc. 34th Ann. ACM Symp. Theory of Computing*, pages 767–775, 2002.
- [44] Subhash Khot. On the power of unique 2-prover 1-round games. In *CCC '02: Proc. 17th Ann. IEEE Conference on Computational Complexity*, page 25, 2002.
- [45] Subhash A. Khot and Nisheeth K. Vishnoi. The unique games conjecture, integrality gap for cut problems and embeddability of negative type metrics into ℓ_1 . In *FOCS'05: Proc. 46th Ann. IEEE Symp. Foundations of Computer Science*, pages 53–62, 2005.
- [46] Jyrki Kivinen and Manfred K. Warmuth. Exponentiated gradient versus gradient descent for linear predictors. *Inf. Comput.*, 132:1–63, 1997.
- [47] Ioannis Koutis and Gary L. Miller. Graph partitioning into isolated, high conductance clusters: theory, computation and applications to preconditioning. In *SPAA'08: Proc. 20th ACM Symp. Parallelism in Algorithms and Architectures*, pages 137–145, 2008.
- [48] Ioannis Koutis, Gary L. Miller, and Richard Peng. Approaching optimality for solving sdd linear systems. In *FOCS'10: Proc. 51st Ann. IEEE Symp. Foundations of Computer Science*, pages 235–244, 2010.

- [49] Frank Thomson Leighton and Satish Rao. An approximate max-flow min-cut theorem for uniform multicommodity flow problems with applications to approximation algorithms. In *FOCS'88: Proc. 29th Ann. IEEE Symp. Foundations of Computer Science*, pages 422–431, 1988.
- [50] Tom Leighton and Satish Rao. Multicommodity max-flow min-cut theorems and their use in designing approximation algorithms. *J. ACM*, 46(6):787–832, 1999.
- [51] Jure Leskovec, Kevin J. Lang, Anirban Dasgupta, and Michael W. Mahoney. Community structure in large networks: Natural cluster sizes and the absence of large well-defined clusters. *CoRR*, abs/0810.1355, 2008.
- [52] Nathan Linial, Eran London, and Yuri Rabinovich. The geometry of graphs and some of its algorithmic applications. *Combinatorica*, 15:577–591, 1995.
- [53] László Lovász and Miklós Simonovits. Random walks in a convex body and an improved volume algorithm. *Random Struct. Algorithms*, 4(4):359–412, 1993.
- [54] Aleksander Madry. Fast approximation algorithms for cut-based problems in undirected graphs. In *FOCS'10: Proc. 51st Ann. IEEE Symp. Foundations of Computer Science*, pages 245–254, 2010.
- [55] Cleve Moler and Charles Van Loan. Nineteen dubious ways to compute the exponential of a matrix. *SIAM Review*, 20:801–836, 1978.
- [56] Lorenzo Orecchia, Leonard J. Schulman, Umesh V. Vazirani, and Nisheeth K. Vishnoi. On partitioning graphs via single commodity flows. In *STOC '08: Proc. 40th Ann. ACM Symp. Theory of Computing*, pages 461–470, 2008.
- [57] Christos H. Papadimitriou and Kenneth Steiglitz. *Combinatorial Optimization: Algorithms and Complexity*. Prentice-Hall, 1982.
- [58] Boyd S. and L. Vandenberghe. *Convex optimization*. Cambridge University Press, 2004.
- [59] Jonah Sherman. Breaking the multicommodity flow barrier for $O(\sqrt{\log n})$ -approximations to sparsest cut. In *FOCS'09: Proc. 50th Ann. IEEE Symp. Foundations of Computer Science*, 2009.
- [60] Jianbo Shi and Jitendra Malik. Normalized cuts and image segmentation. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 22:888–905, 2000.
- [61] David Shmoys. Cut problems and their application to divide and conquer. In Dorit Hochbaum, editor, *Approximation algorithms for NP-hard problems*, pages 192–235. PWS Publishing Co., 1996.

- [62] Daniel A. Spielman. Algorithms, graph theory, and linear equations in laplacian matrices. In *ICM'10: Proc. International Congress of Mathematicians*, 2010.
- [63] Daniel A. Spielman and Nikhil Srivastava. Graph sparsification by effective resistances. In *STOC '08: Proc. 40th Ann. ACM Symp. Theory of Computing*, pages 563–568, 2008.
- [64] Daniel A. Spielman and Shang-Hua Teng. Nearly-linear time algorithms for graph partitioning, graph sparsification, and solving linear systems. In *STOC '04: Proc. 36th Ann. ACM Symp. Theory of Computing*, pages 81–90, 2004.
- [65] Daniel A. Spielman and Shang-Hua Teng. Nearly-linear time algorithms for preconditioning and solving symmetric, diagonally dominant linear systems. *CoRR*, abs/cs/0607105, 2006.
- [66] Daniel A. Spielman and Shang-Hua Teng. A local clustering algorithm for massive graphs and its application to nearly-linear time graph partitioning. *CoRR*, abs/0809.3232, 2008.
- [67] Daniel A. Spielman and Shang-Hua Teng. Spectral sparsification of graphs. *CoRR*, abs/0808.4134, 2008.
- [68] David Steurer. Fast SDP algorithms for constraints satisfaction problems. In *SODA'10: Proc. 21st Ann. ACM-SIAM Symp. Discrete Algorithms*, 2010.
- [69] Luca Trevisan. Approximation algorithms for unique games. In *Proc. 46th Ann. IEEE Symp. Foundations of Computer Science*, pages 05–34, 2005.
- [70] Koji Tsuda, Gunnar Rätsch, and Manfred K. Warmuth. Matrix exponentiated gradient updates for on-line learning and bregman projection. *J. Mach. Learn. Res.*, 6:995–1018, 2005.
- [71] Manfred K. Warmuth and Dima Kuzmin. Online variance minimization. In *COLT*, pages 514–528, 2006.
- [72] Neal E. Young. Randomized rounding without solving the linear program. In *SODA'95: Proc. 6th Ann. ACM-SIAM Symp. Discrete Algorithms*, pages 170–178, 1995.

Appendix A

Omitted Proofs

A.1 Projection Lemma

The results in this section essentially appear in [41], albeit without a detailed proof. We include a formal proof here for completeness.

Fact A.1.1 (Gaussian behavior of projections). *If v is a vector of length l in \mathbb{R}^m and u is a random vector in \mathbb{S}^{m-1} . Then*

1. $\mathbb{E}_u [(v^T u)^2] = \frac{l^2}{m}$,
2. For $x \leq m/16$, $\mathbb{P}_u [(v^T u)^2 \geq xl^2/m] \leq e^{-x/4}$.

Lemma A.1.2. *Let $\{v_i\}_{i=1}^n$ be vectors in \mathbb{R}^{n-1} such that $\sum_i v_i = 0$. Let $\Phi \stackrel{\text{def}}{=} \sum \|v_i\|^2$. Let r be a random uniform unit vector in \mathbb{R}^{n-1} and for all i set $u_i := v_i^T r$. Let S be the partition of $[n]$ such $|S| = n/2$ and for all $i \in S$ and $j \in \bar{S}$ $u_i \geq u_j$. Consider any matching M of the indices $[n]$ across (S, \bar{S}) . Then,*

$$\mathbb{E}_r \left[\sum_{\{i,j\} \in E(M)} \|v_i - v_j\|^2 \right] = \Omega \left(\frac{\Phi}{\log n} \right).$$

Proof. Define the event

$$\mathcal{E}_{ij} := \left\{ (u_i - u_j)^2 \leq \frac{c \log n}{n-1} \|v_i - v_j\|^2 \right\}$$

for some constant $c > 0$. Let $\mathcal{E} := \bigcap_{i,j} \mathcal{E}_{ij}$. By the Fact A.1.1 we have that $\mathbb{P}[\bar{\mathcal{E}}_{ij}] \leq n^{-c/4}$. Hence, by a union bound, $\mathbb{P}[\bar{\mathcal{E}}] \leq n^{-c/4+2}$. Then,

$$\mathbb{E}_r \left[\sum_{\{i,j\} \in E(M)} \|v_i - v_j\|^2 \right] \geq \frac{n-1}{c \log n} \mathbb{E}_r \left[\sum_{\{i,j\} \in E(M)} (u_i - u_j)^2 \mid \mathcal{E} \right] \cdot \mathbb{P}[\mathcal{E}].$$

Let a be the real number such that $u_i \geq a \geq u_j$ for all $i \in S, j \in \bar{S}$. We have $\sum_{i=1}^n u_i = \sum_{i=1}^n v_i^T r = (\sum_{i=1}^n v_i)^T r = 0$. Hence, by the same argument as in Lemma 4.2.1,

$$\sum_{\{i,j\} \in E(M)} (u_i - u_j)^2 \geq \sum_{i=1}^n (u_i - a)^2 = \|u\|^2 - 2a \left(\sum_{i=1}^n u_i \right) + na^2 = \|u\|^2 + na^2 \geq \|u\|^2.$$

So, we have

$$\mathbb{E}_r \left[\sum_{\{i,j\} \in E(M)} \|v_i - v_j\|^2 \right] \geq \frac{n-1}{c \log n} \mathbb{E}_r [\|u\|^2 \mid \mathcal{E}] \cdot \mathbb{P}[\mathcal{E}].$$

To obtain a lower bound on the r.h.s., notice that

$$\mathbb{E}_r [\|u\|^2] = \sum_{i=1}^n \mathbb{E}_r [u_i^2] = \sum_{i=1}^n \frac{\|v_i\|^2}{n-1} = \frac{\Phi}{n-1}.$$

Moreover, as $\|u\|^2 \leq \Phi$,

$$\mathbb{E}_r [\|u\|^2 \mid \mathcal{E}] \cdot \mathbb{P}[\mathcal{E}] \geq \mathbb{E}_r [\|u\|^2] - \mathbb{E}_r [\|u\|^2 \mid \bar{\mathcal{E}}] \cdot \mathbb{P}[\bar{\mathcal{E}}] \geq \frac{\Phi}{n-1} - \Phi \cdot n^{-c/4+2}.$$

By picking c to be a large enough constant, one obtains

$$\mathbb{E}_r \left[\sum_{\{i,j\} \in M} \|v_i - v_j\|^2 \right] \geq \frac{\Phi}{2c \log n}.$$

□

A.2 Proof of Lemma 5.1.10

This argument follows almost exactly a similar analysis by Kale [34]. It is included for completeness.

Preliminaries

For the rest of this section the norm notation will mean the norm in the subspace described by Π . Hence $\|A\| = \|\Pi A \Pi\|$. Recall also that t_A is the running time necessary to perform a matrix-vector multiplication by matrix A . We will need the following lemmata.

Lemma A.2.1 (Johnson-Lindenstrauss). *Given an embedding $\{v_i \in \mathbb{R}^n\}_{i \in V}$, $V = [n]$, let u_1, u_2, \dots, u_k , be vectors sampled independently uniformly from the $n-1$ -dimensional sphere of radius $\sqrt{n/k}$. Let U be the $k \times t$ matrix having the vector u_i as i -th row and let $\tilde{v}_i \stackrel{\text{def}}{=} Uv_i$. Then, for $k_\delta \stackrel{\text{def}}{=} O(\log n/\delta^2)$, for all $i, j \in V$*

$$(1 - \delta) \cdot \|v_i - v_j\|^2 \leq \|\tilde{v}_i - \tilde{v}_j\|^2 \leq (1 + \delta) \cdot \|v_i - v_j\|^2$$

and

$$(1 - \delta) \cdot \|v_i\|^2 \leq \|\tilde{v}_i\|^2 \leq (1 + \delta) \cdot \|v_i\|^2.$$

Lemma A.2.2 ([35]). *There exists an algorithm EXPV which, on input of a matrix $A \in \mathbb{R}^{n \times n}$, a vector $u \in \mathbb{R}^n$ and a parameter η , computes a vector $v \in \mathbb{R}^n$, such that $\|v - e^{-A}u\| \leq \|e^{-A}\| \cdot \eta$ in time $O(t_A \log^3(1/\eta))$.*

The algorithm EXPV is described in [35] and [31].

Proof

We define the \tilde{U}_ϵ algorithm in Figure A.1 and proceed to prove Lemma 5.1.10.

- **Input:** A matrix $M \in \mathbb{R}^{n \times n}$.
- Let $\eta \stackrel{\text{def}}{=} O(1/\text{poly}(n))$. Let $\delta = \Theta(1)$ and $\epsilon = 1/32$.
- For k_δ as in Lemma A.2.1, sample k_δ vectors $u_1, \dots, u_{k_\delta} \in \mathbb{R}^n$ as in Lemma A.2.1.
- Let $A \stackrel{\text{def}}{=} \log(1 - \epsilon) \cdot 2m \cdot D^{-1/2} M D^{-1/2}$.
- For $1 \leq i \leq k_\delta$, compute vectors $b_i \in \mathbb{R}^n$, $b_i \stackrel{\text{def}}{=} \text{EXPV}(1/2 \cdot A, D^{-1/2}u_i, \eta)$.
- Let B be the matrix having b_i as i -th row, and let \tilde{v}_i be the i -th column of B . Compute $Z \stackrel{\text{def}}{=} \mathbb{E}_{\{i,j\} \in \mu \times \mu} \|\tilde{v}_i - \tilde{v}_j\|^2 = L(K_V) \bullet B^T B$.
- Return $\tilde{X} \stackrel{\text{def}}{=} 1/Z \cdot B^T B$, by giving its corresponding embedding, i.e., $\{1/\sqrt{Z} \cdot \tilde{v}_i\}_{i \in V}$.

Figure A.1: The \tilde{U}_ϵ algorithm

Proof. We verify that the conditions required hold.

- By construction, $\tilde{X} \succeq 0$, as $\tilde{X} = 1/Z \cdot B^T B$, and $L(K_V) \bullet \tilde{X} = 1$.

- $\tilde{X} = (1/\sqrt{Z} \cdot B)^T (1/\sqrt{Z} \cdot B)$ and B is a $k_\delta \times n$ matrix, with $k_\delta = O(\log n)$, by Lemma A.2.1.
- We perform $k_\delta = O(\log n)$ calls to the algorithm EXPV, each of which takes time $\tilde{O}(t_A) = \tilde{O}(t_M + n)$. Sampling the vectors requires $\tilde{O}(n)$ time $\{u_i\}_{1, \dots, k_\delta}$ and so does computing Z as we can exploit Fact 5.1.1. Hence, the total running time is $\tilde{O}(t_M + n)$.
- Let U be the $k_\delta \times n$ matrix having the sampled vectors u_1, \dots, u_{k_δ} as rows. Let $\{v_i\}_{i \in V}$ be the embedding corresponding to matrix $Y \stackrel{\text{def}}{=} D^{-1/2} e^{-A} D^{-1/2}$, i.e., v_i is the i -th column of $Y^{1/2}$. Notice that $X = Y/L(K_V) \bullet Y$. Define $\hat{v}_i \stackrel{\text{def}}{=} U v_i$ for all i and let \hat{Y} be the Gram matrix corresponding to this embedding, i.e., $\hat{Y} \stackrel{\text{def}}{=} (Y^{1/2})^T U^T U (Y^{1/2})$. Also, let \tilde{Y} be the Gram matrix corresponding to the embedding $\{\tilde{v}_i\}_{i \in V}$, i.e., $\tilde{Y} = B^T B$ and $\tilde{X} = \tilde{Y}/L(K_V) \bullet \tilde{Y}$. We will relate Y to \hat{Y} and \hat{Y} to \tilde{Y} to complete the proof.

First, by Lemma A.2.1, applied to $\{v_i\}_{i \in V}$, with high probability, for all H

$$(1 - \delta) \cdot L(H) \bullet Y \leq L(H) \bullet \hat{Y} \leq (1 + \delta) \cdot L(H) \bullet Y$$

and for all $i \in V$

$$(1 - \delta) \cdot R_i \bullet Y \leq R_i \bullet \hat{Y} \leq (1 + \delta) \cdot R_i \bullet Y.$$

In particular, this implies that $(1 - \delta) \cdot \Pi \bullet Y \leq \Pi \bullet \hat{Y} \leq (1 + \delta) \cdot \Pi \bullet Y$. Hence,

$$\frac{1 - \delta}{1 + \delta} \cdot L(H) \bullet X \leq L(H) \bullet \hat{X} \leq \frac{1 + \delta}{1 - \delta} \cdot L(H) \bullet X$$

and for all i

$$\frac{1 - \delta}{1 + \delta} \cdot R_i \bullet X \leq R_i \bullet \hat{X} \leq \frac{1 + \delta}{1 - \delta} \cdot R_i \bullet X.$$

Now we relate \hat{Y} and \tilde{Y} . Let $E \stackrel{\text{def}}{=} (\tilde{Y}^{1/2} - \hat{Y}^{1/2}) D^{1/2}$. By Lemma A.2.2

$$\begin{aligned} \|E\|^2 &\leq \|E\|_F^2 = \sum_i \|d_i \tilde{v}_i - \hat{v}_i\|^2 \leq 2m \cdot \|e^{\epsilon/2 \cdot A}\|^2 \cdot \eta^2 \\ &\leq 2m \cdot \|Y^{1/2} D^{1/2}\|^2 \cdot \eta^2 \leq (2m)^2 \cdot L(K_V) \bullet Y \cdot \eta^2. \end{aligned}$$

This also implies

$$\begin{aligned} \|E\| \cdot \|\hat{Y}^{1/2} D^{1/2}\| &\leq \|E\|_F \cdot \|\hat{Y}^{1/2} D^{1/2}\|_F \\ &\leq \left(\sqrt{2m} \cdot \|Y^{1/2} D^{1/2}\| \cdot \eta \right) \cdot \sqrt{\sum_i d_i \|\hat{v}_i - \hat{v}_{\text{avg}}\|^2} \\ &\leq 2m \cdot \eta \cdot (1 + \delta) \cdot L(K_V) \bullet Y. \end{aligned}$$

As $D^{1/2} (\tilde{Y} - \hat{Y}) D^{1/2} = E^T E + (\hat{Y}^{1/2})^T E + E^T \hat{Y}^{1/2}$, we have

$$\begin{aligned}
& \|D^{1/2} (\tilde{Y} - \hat{Y}) D^{1/2}\| \\
& \leq \|E^T E + (\hat{Y}^{1/2})^T E + E^T \hat{Y}^{1/2}\| \\
& \leq \sqrt{3} \left(\|E\|^2 + 2 \cdot \|E\| \|\hat{Y}^{1/2}\| \right) \\
& \leq 9 \cdot (2m)^2 \cdot (1 + \delta) \cdot L(K_V) \bullet Y \cdot \eta.
\end{aligned}$$

and

$$\begin{aligned}
& |L(K_V) \bullet (\tilde{Y} - \hat{Y})| \\
& \leq L(K_V) \bullet (E^T E) + 2 \cdot |L(K_V) \bullet (E^T \hat{Y}^{1/2})| \\
& \leq 1/2m \cdot \|E\|_F^2 + 2/2m \cdot \|E\|_F \|\hat{Y}^{1/2}\|_F \\
& \leq 3 \cdot 2m \cdot (1 + \delta) \cdot L(K_V) \bullet Y \cdot \eta.
\end{aligned}$$

Finally, combining these bounds we have

$$\begin{aligned}
\|\tilde{X} - \hat{X}\|_2 &= \left\| \frac{\tilde{Y}}{L(K_V) \bullet \tilde{Y}} - \frac{\hat{Y}}{L(K_V) \bullet \hat{Y}} \right\|_2 \\
&\leq \left\| \frac{\tilde{Y}}{L(K_V) \bullet \tilde{Y}} - \frac{\tilde{Y}}{L(K_V) \bullet \hat{Y}} \right\|_2 \\
&\quad + \left\| \frac{\tilde{Y}}{L(K_V) \bullet \hat{Y}} - \frac{\hat{Y}}{L(K_V) \bullet \hat{Y}} \right\|_2 \\
&\leq \frac{\|\tilde{Y}\| \cdot |L(K_V) \bullet \tilde{Y} - L(K_V) \bullet \hat{Y}|}{L(K_V) \bullet \tilde{Y} \cdot L(K_V) \bullet \hat{Y}} + \frac{\|\tilde{Y} - \hat{Y}\|}{L(K_V) \bullet \hat{Y}} \\
&\leq \frac{2m \cdot |L(K_V) \bullet \tilde{Y} - L(K_V) \bullet \hat{Y}| + \|\tilde{Y} - \hat{Y}\|}{L(K_V) \bullet \hat{Y}} \\
&\leq \frac{12 \cdot (2m)^2 \cdot (1 + \delta) \cdot L(K_V) \bullet Y \cdot \eta}{(1 - \delta) \cdot L(K_V) \bullet Y} \\
&\leq 12 \cdot (2m)^2 \cdot 1^{1+\delta}/1-\delta \cdot \eta \\
&\leq O(1/\text{poly}(n))
\end{aligned}$$

by taking η sufficiently small in $O(1/\text{poly}(n))$ and $\delta = \Theta(1)$.

Hence, as $\|L(H)\| \leq O(m)$ and $\|R_i\| \leq O(m)$

$$|L(H) \cdot \hat{X} - L(H) \cdot \tilde{X}| \leq O(1/\text{poly}(n))$$

and

$$|R_i \bullet \hat{X} - R_i \bullet \tilde{X}| \leq O(1/\text{poly}(n)).$$

This, together with the fact that we can pick $\delta = O(1)$ such that $1-\delta/1+\delta \geq 1-1/64$ and $1+\delta/1-\delta \leq 1+1/64$ completes the proof.

□