Finding almost-perfect graph bisections

Venkatesan Guruswami[1] Yury Makarychev[2] Prasad Raghavendra[3]
David Steurer[4] Yuan Zhou[5]

[1]Computer Science Department, Carnegie Mellon University, Pittsburgh, PA. Some of this work was done during a visit to Microsoft Research New England.
[2]Toyota Technological Institute at Chicago, Chicago, IL.
[3]College of Computing, Georgia Institute of Technology, Atlanta, GA.
[4]Microsoft Research New England, Cambridge, MA.
[5]Computer Science Department, Carnegie Mellon University, Pittsburgh, PA. Some of this work was done when the author was a student intern at TTI-C.
guruswami@cmu.edu yury@ttic.edu raghavendra@cc.gatech.edu dsteurer@cs.princeton.edu
yuanzhou@cs.cmu.edu

Abstract: We give a polynomial time algorithm that given a graph which admits a bisection cutting a fraction $(1-\varepsilon)$ of edges, finds a bisection cutting a $(1-g(\varepsilon))$ fraction of edges where $g(\varepsilon) \to 0$ as $\varepsilon \to 0$. One can take $g(\varepsilon) = O(\sqrt[3]{\varepsilon}\log(1/\varepsilon))$. Previously known algorithms for Max Bisection could only guarantee finding a bisection that cuts a fraction of edges bounded away from 1 (in fact less than 3/4) in such graphs. The known Unique-Games hardness results for Max Cut imply that one cannot achieve $g(\varepsilon) \leqslant C\sqrt{\varepsilon}$ for some absolute constant C.

Likewise, for the Min Bisection problem, if the graph has a bisection cutting *at most* ε-fraction of the edges, our methods enable finding a bisection cutting at most $O(\sqrt[3]{\varepsilon}\log(1/\varepsilon))$-fraction of the edges.

The algorithms can also find cuts that are nearly-balanced (say with imbalance of at most 0.01) with value at least $1 - O(\sqrt{\varepsilon})$ (for Max Bisection) and at most $O(\sqrt{\varepsilon})$ (for Min Bisection). These guarantees are optimal up to constant factors in the $O(\sqrt{\varepsilon})$ term under the Unique Games and related conjectures.

Our general approach is simple to describe: First, we show how to solve the problems if the graph is an expander or if the graph consists of many disjoint components, each of small measure. Then, we show that every graph can be decomposed into disjoint components that are either expanders or have small measure, and how the cuts on different pieces may be merged appropriately.

Keywords: Graph bisection; Max-Cut; Expander Decomposition; Semidefinite Programming; Approximation Algorithms

1 Introduction

In the Max Cut problem, we are given a graph and the goal is to partition the vertices into two parts so that a maximum number of edges cross the cut. As one of the most basic problems in the class of constraint satisfaction problems, the study of Max Cut has been highly influential in advancing the subject of approximability of optimization problems, from both the algorithmic and hardness sides. The celebrated Goemans-Williamson (GW) algorithm for Max Cut [GW95] was the starting point for the immense and highly successful body of work on semidefinite programming (SDP) based approximation algorithms. This algorithm guarantees finding a cut whose value (i.e., fraction of edges crossing it) is at least 0.878 times the value of the maximum cut. On graphs that are "almost-bipartite" and admit a partition such that most edges cross the cut, the algorithm performs much better — in particular, if there is a cut such that a fraction $(1-\varepsilon)$ of edges cross the cut, then the algorithm finds a partition cutting at least a fraction $1-O(\sqrt{\varepsilon})$

of edges. (All through the discussion in the paper, think of ε as a very small positive constant.)

On the hardness side, the best known NP-hardness result shows hardness of approximating Max Cut within a factor greater than 16/17 [Hås01, TSSW00]. Much stronger and in fact tight inapproximability results are now known conditioned on the Unique Games conjecture of Khot [Kho02]. In fact, one of the original motivations and applications for the formulation of the UGC in [Kho02] was to show that finding a cut of value larger than $1-o(\sqrt{\varepsilon})$ in a graph with Max-Cut value $(1-\varepsilon)$ (i.e., improving upon the above-mentioned performance guarantee of the GW algorithm substantially) is likely to be hard. This result was strengthened in [KKMO07] to the optimal $1-O(\sqrt{\varepsilon})$ bound, and this paper also showed that the 0.878 approximation factor is the best possible under the UGC. (These results relied, in addition to the UGC, on the Majority is Stablest conjecture, which was proved shortly afterwards in [MOO10].)

There are many other works on Max Cut, including algorithms that improve on the GW algorithm for certain ranges of the optimum value and integrality gap constructions showing limitations of the SDP based approach. This long line of work on Max Cut culminated in the paper [OW08] which obtained the precise integrality gap and approximation threshold curve as a function of the optimum cut value.

Maximum Bisection. Let us consider a closely related problem called Max Bisection, which is Max Cut with a global "balanced cut" condition. In the Max Bisection problem, given as input a graph with an even number of vertices, the goal is to partition the vertices into two equal parts while maximizing the fraction of cut edges. Despite the close relation to Max Cut, the global constraint in Max Bisection changes its character substantially, and the known algorithms for approximating Max Bisection have weaker guarantees. While Max Cut has a factor 0.878 approximation algorithm [GW95], the best known approximation factor for Max Bisection equals 0.7027 [FL06], improving on previous bounds of 0.6514 [FJ97], 0.699 [Ye01], and 0.7016 [HZ02].

In terms of inapproximability results, it is known that Max Bisection cannot be approximated to a factor larger than 15/16 unless NP $\subseteq \bigcap_{\gamma>0} \mathrm{TIME}(2^{n^\gamma})$ [HK04]. Note that this hardness factor is slightly better than the inapproximability factor of 16/17 known for Max Cut [Hås01, TSSW00]. A simple approximation preserving reduction from Max Cut shows that Max Bisection is no easier to approximate than Max Cut (the reduction is simply to take two disjoint copies of the Max Cut instance). Therefore, the factor 0.878 Unique-Games hardness for Max Cut [KKMO07] also applies for Max Bisection. Further, given a graph that has a bisection cutting $1-\varepsilon$ of the edges, it is Unique-Games hard to find a bisection (or even any partition in fact) cutting $1-O(\sqrt{\varepsilon})$ of the edges.

An intriguing question is whether Max Bisection is in fact harder to approximate than Max Cut (so the global condition really changes the complexity of the problem), or whether there are algorithms for Max Bisection that match (or at least approach) what is known for Max Cut.[1] None of the previously known algorithms for Max Bisection [FJ97, Ye01, HZ02, FL06] are guaranteed to find a bisection cutting most of the edges even when the graph has a near-perfect bisection cutting $(1-\varepsilon)$ of the edges (in particular, they may not even cut 75% of the edges). These algorithms are based on rounding a vector solution of a semidefinite programming relaxation into a cut (for example by a random hyperplane cut) and then balancing the cut by moving low-degree vertices from the larger side to the smaller side. After the first step, most of the edges are cut, but the latter rebalancing step results in a significant loss in the number of edges cut. In fact, as we will illustrate with a simple example in Section 2, the standard SDP for Max Bisection has a large integrality gap: the SDP optimum could be 1 whereas every bisection might only cut less than 0.95 fraction of the edges.

Thus an interesting "qualitative" question is one can one efficiently find an *almost-complete* bisection when promised that one exists. Formally, we ask the following question.

Question 1.1. Is there a polynomial time algorithm that given a graph $G = (V, E)$ with a Max Bisection solution of value $(1-\varepsilon)$, finds a bisection of value $(1-g(\varepsilon))$, where $g(\varepsilon) \to 0$ as $\varepsilon \to 0$?

[1]Note that for the problem of minimizing the number of edges cut, the global condition does make a big difference: Min Cut is polynomial-time solvable whereas Min Bisection is NP-hard.

Note that without the bisection constraint, we can achieve $g(\varepsilon) = O(\sqrt{\varepsilon})$, and when $\varepsilon = 0$, we can find a bisection cutting all the edges (see the first paragraph of Section 2 for details). Thus this question highlights the role of both the global constraint and the "noise" (i.e., ε fraction of edges need to be removed to make the input graph bipartite) on the complexity of the problem.

Our results. In this paper, we answer the above question in the affirmative, by proving the following theorem.

Theorem 1.2. *There is a randomized polynomial time algorithm that given an edge-weighted graph G with a* Max Bisection *solution of value*[2] $(1 - \varepsilon)$ *finds a* Max Bisection *of value* $(1 - O(\sqrt[3]{\varepsilon}\log(1/\varepsilon)))$.

We remark that for regular graphs any cut with most of the edges crossing it must be near-balanced, and hence we can solve Max Bisection by simply reducing to Max Cut. Thus the interesting instances for our algorithm are non-regular graphs.

Our algorithms are not restricted to finding exact bisections. If the graph has a β-*balanced* cut (which means the two sides have a fraction β and $1-\beta$ of the vertices) of value $(1-\varepsilon)$, then the algorithm can find a β-balanced cut of value $1 - O(\varepsilon^{1/3}\log(1/\varepsilon))$. The performance guarantee of a variant of the algorithm improves if some extra imbalance is tolerated in the solution — for any constant parameter δ, the algorithm can return a cut with balance in the range $\beta \pm \delta$ with value at least $1 - O(\varepsilon^{1/2})$. Formally, we prove the following theorem.

Theorem 1.3. *There is a randomized polynomial time algorithm that for any constant $\delta > 0$ given an edge-weighted graph G with a β-balanced cut of value $(1 - \varepsilon)$ finds a cut (A, B) such that $\left||A|/|V| - \beta\right| \leqslant O(\delta)$ of value $(1 - O(\sqrt{\varepsilon}))$.*

As mentioned earlier, as a function of ε, the above approximation is best possible assuming the Unique Games Conjecture.

Our results are not aimed at improving the general approximation ratio for Max Bisection which remains at ≈ 0.7027 [FL06]. It remains an interesting question how much this can be improved and whether one approach (or even match) the 0.878 factor possible for Max Cut. We hope that some of our techniques will also be helpful towards this goal.

More generally, our work highlights the challenge of understanding the complexity of solving constraint satisfaction problems with global constraints. Algorithmically, the challenge is to ensure that the global constraint is met without hurting the number of satisfied constraints. From the hardness side, the Unique-Games based reductions which have led to a complete understanding of the approximation threshold of CSPs [Rag08] are unable to exploit the global constraint to yield stronger hardness results.

Our methods apply just as well to the Min Bisection problem where the goal is to find a bisection that cuts fewest number of edges. The best approximation factor known for Min Bisection is poly-logarithmic in the number of vertices [FK02]. Here we show that we can get better guarantees in the case when the best bisection cuts a constant fraction of the edges. Specifically, if there is a bisection of value ε, our methods guarantee efficiently finding a bisection cutting at most $O(\varepsilon^{1/3}\log(1/\varepsilon))$ of the edges, and a nearly-balanced cut (say with bias at most 0.01) of value at most $O(\varepsilon^{1/2})$. The adaptation of our algorithm and analysis to Min Bisection is simple, see Section 8. (Similar to Max Bisection, the approximation guarantee $O(\sqrt{\varepsilon})$ is optimal as a function of ε for constant imbalance, assuming the so-called Small-Set Expansion conjecture [RST10] or a variant of the Unique Games Conjecture [AKK+08].)

2 Method overview

2.1 Integrality gap

We begin by describing why the standard SDP for Max Bisection has a large gap. Given a graph $G = (V, E)$, this SDP, which is the basis of all previous algorithms for Max Bisection starting with that of Frieze and Jerrum [FJ97], solves for unit vectors v_i for each vertex $i \in V$ subject to $\sum_i v_i = 0$, while maximizing the objective function $\mathbb{E}_{e=(i,j)\in E}\,\frac{1}{4}\|v_i - v_j\|^2$.

This SDP could have a value of 1 and yet the graph may not have any bisection of value more than 0.95 (in particular the optimum is bounded away from 1), as the following example shows.

[2]The value of a cut in an edge-weighted graph is defined as the weight of the edges crossing the cut divided by the total weight of all edges.

Take G to be the union of *three* disjoint copies of $K_{2m,m}$ (the complete $2m \times m$ bipartite graph) for some even m. It can be seen that every bisection fails to cut at least $m^2/2$ edges, and thus has value at most $11/12$. On the other hand, the SDP has a solution of value 1. Let $\omega = e^{2\pi i/3}$ be the primitive cube root of unity. In the two-dimensional complex plane, we assign the vector/complex number ω^{i-1} (resp. $-\omega^{i-1}$) to all vertices in the larger part (resp. smaller part) of the i^{th} copy of $K_{2m,m}$ for $i = 1, 2, 3$. These vectors sum up to 0 and for each edge, the vectors associated with its endpoints are antipodal.

For all CSPs, a tight connection between integrality gaps (for a certain "canonical" SDP) and inapproximability results is now established [Rag08]. The above gap instance suggests that the picture is more subtle for CSPs with global constraints — in this work we give an algorithm that does much better than the integrality gap for the "basic" SDP. Could a stronger SDP relaxation capture the complexity of approximating CSPs with global constraints such as Max Bisection? It is worth remarking that we do not know whether an integrality gap instance of the above form (i.e., $1-\varepsilon$ SDP optimum vs. say 0.9 Max Bisection value) exists even for the basic SDP augmented with triangle inequalities.

2.2 Notation

Suppose we are given a graph $G = (V, E)$. We use the following notation: $E(U) = \{(u, v) \in E : u, v \in U\}$ denotes the set of edges within a set of vertices U, $\mathsf{edges}(U_1, U_2) = \{(u, v) \in E : u \in U_1, v \in U_2\}$ denotes the set of edges between two sets of vertices U_1 and U_2, and $G[U]$ denotes the subgraph of G induced by the set U.

Definition 2.1 (Value and bias of cuts)**.** *For a cut $(S, V \setminus S)$ of a graph $G = (V, E)$, we define its* value *to be $\frac{|\mathsf{edges}(S,V\setminus S)|}{|E|}$ (i.e., the fraction of edges which cross the cut) if G is unweighted, and $\frac{w(\mathsf{edges}(S,V\setminus S))}{w(E)}$ if G is edge-weighted with weight function $w : E \to \mathbb{R}^{\geqslant 0}$ (where for $F \subseteq E$, $w(F) = \sum_{e\in F} w(e)$).*
We define the bias *$\beta \in [0, 1]$ of a cut $(S, V \setminus S)$ to be $\beta = \frac{1}{|V|} \cdot \Big||S| - |V \setminus S|\Big|$, and we say that the cut $(S, V \setminus S)$ is β-biased. (Note that a 0-biased cut is a bisection.)*

Recall that the normalized Laplacian of G is a matrix $\mathcal{L}_G$ whose rows and columns correspond to vertices of G that is defined as follows

$$\mathcal{L}_G(u, v) = \begin{cases} 1, & \text{if } u = v \text{ and } d_u \neq 0, \\ -1/\sqrt{d_u d_v}, & \text{if } (u, v) \in E, \\ 0, & \text{otherwise,} \end{cases}$$

where d_u is the degree of the vertex u. Let $\lambda_2(\mathcal{L}_G)$ be the second smallest eigenvalue of $\mathcal{L}_G$. We abuse the notation by letting $\lambda_2(G) = \lambda_2(\mathcal{L}_G)$. We define the volume of a set $U \subseteq V$ as $\mathrm{vol}(U) = \mathrm{vol}_G(U) = \sum_{u\in U} d_u$.

We will use the following version of Cheeger's inequality.

Theorem 2.2 (Cheeger's inequality for non-regular graphs [Chu96])**.** *For every graph $G = (V, E)$,*

$$\lambda_2(G)/2 \leqslant \phi(G) \leqslant \sqrt{2\lambda_2(G)},$$

where $\phi(G)$ is the expansion of G,

$$\phi(G) \equiv \min_{S\subseteq V} \frac{|\mathsf{edges}(S, V \setminus S)|}{\min(\mathrm{vol}(S), \mathrm{vol}(V \setminus S))}.$$

Moreover, we can efficiently find a set $A \subseteq V$ such that $\mathrm{vol}(A) \leqslant \mathrm{vol}(V)/2$ and $|\mathsf{edges}(A, V \setminus A)|/\mathrm{vol}(A) \leqslant \sqrt{2\lambda_2(G)}$.

For any two disjoint sets $X, Y \subseteq V$, let $\mathsf{uncut}(X, Y) = |E(X) + E(Y)|/|E(X \cup Y)|$ be the fraction of edges of $G[X \cup Y]$ that do not cross the cut (X, Y). We say that a cut (X, Y) of V is perfect if $\mathsf{uncut}(X, Y) = 0$.

2.3 Our approach

In this section, we give a brief overview of our algorithm. It is instructive to consider first the case when G has a perfect bisection cut. In this case, G is a bipartite graph. If G has only one connected component, each part of this component has the same number of vertices, so this is the desired bisection. Now assume that G has several connected components. Then each connected component C of G is a bipartite graph with two parts X_C and Y_C. Since all edges are cut in the optimal solution, X_C must lie on one side of the optimal cut and Y_C on the other. So in order to find a perfect bisection (X, Y), for every connected component C we need to either (i) add X_C to X and Y_C to Y or (ii) add X_C to Y and Y_C to X so that $|X| = |Y| = |V|/2$. We can do that using dynamic programming.

Our algorithm for almost satisfiable instances proceeds in a similar way. Assume that the optimal bisection cuts a $(1 - \varepsilon)$ fraction of edges.

1. In a preprocessing step, we use the algorithm of Goemans and Williamson [GW95] to find an approximate maximum cut in G. A fraction $1 - O(\sqrt{\varepsilon})$ of edges cross this cut. We remove all uncut edges and obtain a bipartite graph. We denote the parts of this graph by A and B. (Of course, in general $|A| \neq |B|$.)

2. Then we recursively partition G into pieces $W_1, \ldots, W_s$ using Cheeger's Inequality (see Lemma 3.1). Every piece is either a sufficiently small subgraph, which contains at most an ε fraction of all vertices, or is a spectral expander, with $\lambda_2 \geqslant \varepsilon^{2/3}$. There are very few edges between different pieces, so we can ignore them later. In this step, we obtain a collection of induced subgraphs $G[W_1], \ldots, G[W_s]$ with very few edges going between different subgraphs.

3. Now our goal is to find an "almost perfect" cut in every $G[W_i]$, then combine these cuts and get a bisection of G. Note that every $G[W_i]$ is bipartite and therefore has a perfect cut (since G is bipartite after the preprocessing step). However, we cannot restrict our attention only to this perfect cut since the optimal solution (S, T) can cut $G[W_i]$ in another proportion. Instead, we prepare a list $\mathcal{W}_i$ of "candidate cuts" for each $G[W_i]$ that cut W_i in different proportions. One of them is close to the cut $(W_i \cap S, W_i \cap T)$ (the restriction of the optimal cut to W_i).

4. If $G[W_i]$ is an expander, we find a candidate cut that cuts $G[W_i]$ in a given proportion by moving vertices from one side of the perfect cut $(W_i \cap A, W_i \cap B)$ to the other, greedily (see Lemma 4.1 and Lemma 4.2).

5. If $G[W_i]$ is small, we find a candidate cut that cuts $G[W_i]$ in a given proportion using semi-definite programming (see Lemma 4.3 and Corollary 4.4). We solve an SDP relaxation similar to the Goemans–Williamson relaxation [GW95] with "ℓ_2^2-triangle inequalities", and then find a cut by using hyperplane or threshold rounding.

   In fact, the cut that we find can be more unbalanced than $(W_i \cap S, W_i \cap T)$ but this is not a problem since the set W_i is small. Note however that if a cut of another piece W_j is very unbalanced than we might need to find a cut of W_i that is unbalanced in the other direction. So it is important that the candidate cut of W_i is at least as unbalanced as $(W_i \cap S, W_i \cap T)$.

6. Finally, we combine candidate cuts of subgraphs $G[W_i]$ into one balanced cut of the graph G, in the optimal way, using dynamic programming (see Lemma 5.1).

Our improved algorithm (Theorem 1.3) which delivers a cut of value $1 - O(\sqrt{\varepsilon})$ with a small constant bias has the same high level structure. Namely, we decompose the graph into expanding and small parts, find good cuts in these parts, and combine them together to get a global near-balanced cut. The small parts are handled in the same way as above. The difference is in the notion of expansion used for the large parts and how we find cuts in the expanding parts. For the former, a part is deemed to have sufficient expansion if the standard SDP relaxation for α-balanced separator (for some small $\alpha > 0$) has large $(\Omega(\varepsilon))$ objective value. (This in particular means there every α-balanced cut has value at least $\Omega(\varepsilon)$, but is potentially stronger as it ensures that there are no sparse "vector-valued" cuts either.)

We give a decomposition procedure to split a graph into such "SDP expanders" and small parts (Lemma 7.5). We find a good cut in the union of the expanding parts by rounding the Goemans-Williamson SDP. The key in our analysis is to use the "SDP expansion" properties to argue that the vector solution must essentially behave like an integral solution. This gives us much better control of the bias of the cut produced by random hyperplane rounding.

2.4 Organization

The rest of the paper is devoted to the full description and proof of the algorithm. In Section 3, we partition the graph into expanders and small pieces, after proper preprocessing. In Section 4, we produce a list of candidate cuts for each expander and small piece, by different methods. In Section 5, we show how to choose one candidate cut for each part. In Section 6, we put everything together to finish the proof of Theorem 1.2. In Section 7, we prove Theorem 1.3, giving a variant of the algorithm that only uncut $O(\sqrt{\varepsilon})$ fraction of edges when some constant imbalance is tolerated. In Section 8, we apply our techniques to Min Bisection problem.

3 Preprocessing and partitioning graph G

In this section, we present the preprocessing and partitioning steps of our algorithms. We will assume that we know the value of the optimal solution $OPT = 1 - \varepsilon_{OPT}$ (with a high precision). If we do not, we can run the algorithm for many different values of ε and output the best of the bisection cuts we find.

3.1 Preprocessing: Making G bipartite and unweighted

In this section, we show that we can assume that the graph G is bipartite, with parts A and B, unweighted, and that $|E| \leqslant O(|V|/\varepsilon_{OPT}^2)$.

First, we "sparsify" the edge-weighted graph $G = (V, E)$, and make the graph unweighted: we sample $O(\varepsilon_{OPT}^{-2}|V|)$ edges (according to the weight distribution) with replacement from G, then with high probability, every cut has the same cost in the original graph as in the new graph, up to an additive error ε_{OPT} (by Chernoff's bound). So we assume that $|E| \leqslant O(\varepsilon_{OPT}^{-2}|V|)$.

We apply the algorithm of Goemans and Williamson to G and find a partitioning of G into two pieces A and B so that only an $O(\sqrt{\varepsilon_{OPT}})$ fraction of edges lies within A or within B.

3.2 Partitioning

In this section, we describe how we partition G into pieces.

Lemma 3.1. *Given a graph $G = (V, E)$, and parameters $\delta \in (0, 1)$ and $\lambda \in (0, 1)$ such that $|E| = O(|V|/\delta^2)$, we can find a partitioning of V into disjoint sets $U_1, \ldots, U_p$ ("small sets"), and $V_1, \ldots, V_q$ ("expander graphs"):*

$$V = \bigcup_i U_i \cup \bigcup_j V_j,$$

in polynomial time, so that

1. *$|U_i| \leqslant \delta|V|$ for each $1 \leqslant i \leqslant p$;*
2. *$\lambda_2(G[V_i]) \geqslant \lambda$ for each $1 \leqslant i \leqslant q$;*
3. *$\sum_i |E(U_i)| + \sum_j |E(V_j)| \geqslant (1 - O(\sqrt{\lambda}\log(1/\delta)))|E|$.*

Proof. We start with a trivial partitioning $\{V\}$ of V and then iteratively refine it. Initially, all sets in the partitioning are "active"; once a set satisfies conditions 1 or 2 of the lemma, we permanently mark it as "passive" and stop subdividing it. We proceed until all sets are passive. Specifically, we mark a set S as passive in two cases. First, if $|S| \leqslant \delta|V|$ then we add S to the family of sets U_i. Second, if $\lambda_2(G[S]) \geqslant \lambda$ then we add S to the family of sets V_i.

We subdivide every active S into smaller pieces by applying the following easy corollary of Cheeger's inequality (Theorem 2.2) to $H = G[S]$.

Corollary 3.2. *Given a graph $H = (S, E(H))$ and a threshold $\lambda > 0$, we can find, in polynomial time, a partition $S_1, S_2, \cdots, S_t$ of S such that*

1. *$|E(S_i)| \leqslant |E(S)|/2$ or $\lambda_2(H[S_i]) \geqslant \lambda$, for each $1 \leqslant i \leqslant t$.*
2. *$\sum_{i<j} |\mathsf{edges}(S_i, S_j)| \leqslant \sqrt{8\lambda}|E(H)|$.*
3. *each graph $H[S_i]$ is connected.*

Proof. If $\lambda_2(H) \geqslant \lambda$ then we just output a trivial partition $\{S\}$. Otherwise, we apply Theorem 2.2 to $H_1 = H$, find a set S_1 s.t. $\mathrm{vol}_{H_1}(S_1) \leqslant \mathrm{vol}_{H_1}(S)/2$ and $|\mathsf{edges}(S_1, S \setminus S_1)|/\mathrm{vol}_{H_1}(S_1) \leqslant \sqrt{2\lambda_2(H_1)} \leqslant \sqrt{2\lambda}$. Then we remove S_1 from H_1, obtain a graph H_2 and iteratively apply this procedure to H_2. We stop when either $\lambda_2(H_i) \geqslant \lambda$ or $|E(H_i)| \leqslant |E(S)|/2$.

We verify that the obtained partitioning $S_1, \ldots, S_t$ of S satisfies the first condition. For each $i \in \{1, \ldots, t-1\}$, we have $|E(S_i)| \leqslant \mathrm{vol}_{H_i}(S_i)/2 \leqslant \mathrm{vol}_{H_i}(V(H_i))/4 = E(H_i)/2 \leqslant |E(H)|/2$. Our stopping criterion guarantees that S_t satisfies the first condition. We verify the second condition.

$$\sum_{i<j} |\mathsf{edges}(S_i, S_j)| = \sum_{i=1}^{t-1} |\mathsf{edges}(S_i, V(H_i) \setminus S_i)|$$
$$\leqslant \sum_{i=1}^{t-1} \sqrt{2\lambda}\,\mathrm{vol}_{H_i}(S_i) \leqslant \sqrt{2\lambda}\,\mathrm{vol}_H(S) = 2\sqrt{2\lambda}|E(H)|.$$

Finally, if for some i, $H[S_i]$ is not connected, we replace S_i in the partitioning with the connected components of $H[S_i]$. □

By the definition, sets U_i and V_j satisfy properties 1 and 2. It remains to verify that

$$\sum_{i=1}^{p} |E(U_i)| + \sum_{j=1}^{q} |E(V_j)| \geqslant (1 - O(\sqrt{\lambda}\log(1/\delta)))|E|.$$

We first prove that the number of iterations is $O(\log(1/\delta))$. Note that if S is an active set and T is its parent then $|E(S)| \leqslant |E(T)|/2$. Set V contains $O(|V|/\delta^2)$ edges. Every active set S contains at least $\delta|V|/2$ edges, since $|E(S)| \geqslant |S| - 1 \geqslant \delta|V|/2$ (we use that $G[S]$ is connected). Therefore, the number of iterations is $O(\log_2((|V|/\delta^2)\big/(\delta|V|/2))) = O(\log 1/\delta)$.

We finally observe that when we subdivide a set S, we cut $O(\sqrt{\lambda}|E(S)|)$ edges. At each iteration, since all active sets are disjoint, we cut at most $O(\sqrt{\lambda}|E|)$ edges. Therefore, the total number of edges cut in all iterations is $O(\sqrt{\lambda}\log(1/\delta))|E|$. □

4 Finding cuts in sets U_i and V_i

In the previous section, we showed how to partition the graph G into the union of "small graphs" $G[U_i]$ and expander graphs $G[V_i]$. We now show how to find good "candidate cuts" in each of these graphs.

4.1 Candidate cuts in V_i

In this section, first we prove that there is essentially only one almost perfect maximum cut in an expander graph (Lemma 4.1). That implies that every almost perfect cut in the graph $G[V_i]$ should be close to the perfect cut $(V_i \cap A, V_i \cap B)$. Using that we construct a list of good candidate cuts (Lemma 4.2). One of these cuts is close to the restriction of the optimal cut to subgraph $G[V_i]$.

Lemma 4.1. *Suppose we are given a graph* $H = (V, E)$ *and two cuts* (S_1, T_1) *and* (S_2, T_2) *of* G*, each of value at least* $(1 - \delta)$*. Then*

$$\min\{\mathrm{vol}_H(S_1 \vartriangle S_2), \mathrm{vol}_H(S_1 \vartriangle T_2)\} \leqslant 4\delta|E|/\lambda_2(H).$$

Proof. Let

$$X = S_1 \vartriangle S_2 = (S_1 \cap T_2) \cup (S_2 \cap T_1);$$
$$Y = S_1 \vartriangle T_2 = (S_1 \cap T_1) \cup (S_2 \cap T_2).$$

Note that $V = X \cup Y$. There are at most $2\delta|E|$ edges between X and Y, since

$$\mathsf{edges}(X, Y) \subset E(S_1) \cup E(S_2) \cup E(T_1) \cup (T_2),$$

$|E(S_1) \cup E(T_1)| \leqslant \delta|E|$ and $|E(S_2) \cup E(T_2)| \leqslant \delta|E|$.

On the other hand, by Cheeger's inequality (Theorem 2.2), we have

$$\frac{|\mathsf{edges}(X, Y)|}{\min(\mathrm{vol}_H(X), \mathrm{vol}_H(Y))} \geqslant \lambda_2(H)/2.$$

Therefore,

$$\min(\mathrm{vol}_H(X), \mathrm{vol}_H(Y)) \leqslant 2|\mathsf{edges}(X, Y)|/\lambda_2(H) \leqslant \frac{4\delta|E|}{\lambda_2(H)}.$$

□

Consider one of the sets V_i. Let $H = G[V_i]$. Denote $A_i = V_i \cap A$ and $B_i = V_i \cap B$. We sort all vertices in A_i and B_i w.r.t. their degrees in H. Now we are ready to define the family of candidates cuts $(X_0, Y_0), \dots, (X_{|V_i|}, Y_{|V_i|})$ for $G[V_i]$. For each j, we define (X_j, Y_j) as follows.

– If $j \leqslant |A_i|$ then X_j consists of j vertices of A_i with highest degrees, and Y_j consists of the remaining vertices of H (i.e. Y_j contains all vertices of B_i as well as $|A_i| - j$ lowest degree vertices of A_i).
– If $j \geqslant |A_i|$ then Y_j consists of $|V_i| - j$ vertices of B_i with highest degrees, and X_j consists of the remaining vertices of H.

Clearly, $|X_j| = j$ and $|Y_j| = |V_i| - j$. Let (S, T) be the restriction of the optimal bisection of G to H. We will show that one of the cuts (X_j, Y_j) is not much worse than (S, T). By Lemma 4.1 applied to cuts (A_i, B_i) and (S, T) (note that $\mathsf{uncut}(A_i, B_i) = 0$),

$$\min\{\mathrm{vol}_H(A_i \vartriangle S), \mathrm{vol}_H(A_i \vartriangle T)\} \leqslant \frac{4 \cdot \mathsf{uncut}(S, T)|E(H)|}{\lambda_2(H)}.$$

Assume without loss of generality that $\mathrm{vol}_H(A_i \vartriangle S) \leqslant 4E(H)/\lambda_2(H)$ (otherwise, rename sets X and Y). We show that $\mathrm{vol}_H(A_i \vartriangle X_{|S|}) \leqslant \mathrm{vol}_H(A_i \vartriangle S)$. Consider the case $|A_i| \geqslant |S|$. Note that by the definition of $X_{|S|}$, the set $X_{|S|}$ has the largest volume among all subsets of A_i of size at most $|S|$. Correspondingly, $A_i \setminus X_{|S|}$ has the smallest volume among all subsets of A_i of size at least $|A_i| - |S|$. Finally, note that $|A_i \setminus S| \geqslant |A_i| - |S|$. Therefore,

$$\mathrm{vol}_H(A_i \vartriangle X_{|S|}) = \mathrm{vol}_H(A_i \setminus X_{|S|}) \leqslant \mathrm{vol}_H(A_i \setminus S) \leqslant \mathrm{vol}_H(A_i \vartriangle S).$$

The case when $|A_i| \leqslant |S|$ is similar. We conclude that

$$\mathrm{vol}_H(A_i \vartriangle X_{|S|}) \leqslant 4 \cdot \mathsf{uncut}(S, T)|E_H|/\lambda_2(H).$$

Therefore, the size of the cut $(X_{|S|}, Y_{|S|})$ is at least

$$|E(H)| - \mathrm{vol}_H(A_i \vartriangle X_{|S|}) \geqslant \Big(1 - \frac{4 \cdot \mathsf{uncut}(S, T)}{\lambda_2(H)}\Big)|E(H)|.$$

We have thus proved the following lemma.

Lemma 4.2. *There is a polynomial time algorithm that given a graph $H = G([V_i])$ finds a family of cuts $\mathcal{V}_i = \{(X_1, Y_1), \dots, (X_{|V_i|}, Y_{|V_i|})\}$ such that for every cut (S, T) of H there exists a cut $(X, Y) \in \mathcal{V}_i$ with $|X| = \min(|S|, |T|)$ and*

$$\mathsf{uncut}(X, Y) \leqslant \frac{4 \cdot \mathsf{uncut}(S, T)}{\lambda_2(H)}.$$

4.2 Candidate cuts in U_i

In this section, we show how to find candidate cuts for the small parts, i.e., the induced subgraphs $G[U_i]$.

Lemma 4.3. *Suppose we are given a graph $H = (U, E)$ and two parameters $0 \leqslant \theta \leqslant 1/2$ and $0 < \Delta < 1$. Then in polynomial time we can find a cut (X, Y) such that for every cut (S, T) in H, with $|S| \leqslant \theta|U|$, we have*

1. $\mathsf{uncut}(X, Y) \leqslant O(\sqrt{\mathsf{uncut}(S, T)} + \mathsf{uncut}(S, T)/\Delta)$.
2. $|X| \leqslant (\theta + \Delta)|U|$.

Proof. Let (S, T) be the maximum cut among all cuts with $|S| \leqslant t|U|$ (of course, our algorithm does not know (S, T)). Let $\varepsilon_H = \mathsf{uncut}(S, T)$. We may assume that our algorithm knows the value of ε_H (with high precision) — as otherwise, we can run our algorithm on different values of ε and output the best of the cuts the algorithm finds.

We write the following SDP program. For every vertex $i \in U$, we introduce a unit vector v_i. Additionally, we introduce a special unit vector v_0.

$$\begin{aligned}
&\text{maximize } \frac{1}{|U|}\sum_{i\in U}\langle v_0, v_i\rangle \\
&\text{subject to } \frac{1}{4|E|}\sum_{(i,j)\in E}\|v_i + v_j\|^2 \leqslant \varepsilon_H \\
&\qquad \|v_i\|^2 = 1 \qquad \forall i \in V \cup \{0\} \\
&\qquad |\langle v_i + v_j, v_0\rangle| \leqslant \frac{\|v_i + v_j\|^2}{2} \qquad \forall i, j \in V.
\end{aligned}$$

The "intended solution" to this SDP is $v_i = v_0$ if $i \in T$ and $v_i = -v_0$ if $i \in S$ (vector v_0 is an arbitrary unit vector). Clearly, this solution satisfies all SDP constraints. In particular, it satisfies the last constraint ("an ℓ_2^2-triangle inequality") since the left hand side is positive only when $v_i = v_j$, then $|\langle v_i + v_j, v_0\rangle| = \frac{\|v_i+v_j\|^2}{2} = 2$. The value of this solution is $(|T| - |S|)/|U| \geqslant 1 - 2\theta$.

We solve the SDP and find the optimal SDP solution $\{v_i\}$. Note that $\sum_{i\in U}\langle v_0, v_i\rangle \geqslant (1 - 2\theta)|U|$.

Let $\Delta' = 2\Delta/3$. Choose $r \in [\Delta', 2\Delta']$ uniformly at random. Define a partition of U into sets Z_k, $0 \leqslant k < 1/\Delta'$, as follows: let $Z_k = \{i : k\Delta' + r < |\langle v_0, v_i\rangle| \leqslant (k + 1)\Delta' + r\}$ for $k \geqslant 1$ and $Z_0 = \{i : -\Delta' - r \leqslant \langle v_0, v_i\rangle \leqslant \Delta' + r\}$. We bound the probability that the endpoints of an edge (i, j) belong to different sets Z_k. Note that if no point from the set $\{\pm(k\Delta' + r) : k \geqslant 1\}$ lies between $|\langle v_i, v_0\rangle|$ and $|\langle v_j, v_0\rangle|$ then i and j belong to the same set Z_k. The distance between $|\langle v_i, v_0\rangle|$ and $|\langle v_j, v_0\rangle|$ is at most $|\langle v_i + v_j, v_0\rangle|$. Therefore, the probability (over r) that i and j belong to different sets Z_k is at most $|\langle v_i + v_j, v_0\rangle|/\Delta'$. So the expected number of cut edges is at most

$$\frac{1}{\Delta'}\sum_{(i,j)\in E}|\langle v_i+v_j, v_0\rangle| \leqslant \frac{1}{2\Delta'}\sum_{(i,j)\in E}\|v_i+v_j\|^2 \leqslant \frac{2|E|\varepsilon_H}{\Delta'}. \tag{4.1}$$

For each $k \geqslant 1$, let $Z_k^+ = \{i \in Z_k \mid \langle v_i, v_0\rangle > 0\}$ and $Z_k^- = \{i \in Z_k \mid \langle v_i, v_0\rangle < 0\}$. We use hyperplane rounding of Goemans and Williamson [GW95] to divide Z_0 into two sets Z_0^+ and Z_0^-. We are ready to define sets X and Y. For each k, we add vertices from the smaller of the two sets Z_k^+ and Z_k^- to X, and vertices from the larger of them to Y.

Now we bound $\mathsf{uncut}(X, Y)$. Note that

$$\begin{aligned}
|\mathsf{uncut}(X, Y)| \leqslant\ & \sum_{k<l}|\mathsf{edges}(Z_k, Z_l)| \\
&+ \sum_{k\geqslant 0}(|E(Z_k^+)| + |E(Z_k^-)|).
\end{aligned}$$

We have already shown that $\sum_{k<l}|\mathsf{edges}(Z_k, Z_l)|$ is less than $2\varepsilon_H|E|/\Delta'$ in expectation. If $(i, j) \in E(Z_k^+)$ or $(i, j) \in E(Z_k^-)$ for $k \geqslant 1$ then $|\langle v_i + v_j, v_0\rangle| \geqslant \Delta'$. Therefore,

$$\sum_{k\geqslant 1}(|E(Z_k^+)| + |E(Z_k^-)|) \leqslant \frac{2\varepsilon_H|E|}{\Delta'} = \frac{3\varepsilon_H|E|}{\Delta}.$$

Finally, note that when we divide Z_0, the fraction of edges of $E(Z_0)$ that do not cross the random hyperplane is $O(\sqrt{\varepsilon_0})$ (in expectation) where

$$\varepsilon_0 = \frac{1}{4|E(Z_0)|}\sum_{(i,j)\in E(Z_0)}\|v_i + v_j\|^2$$

$$\leqslant \frac{1}{4|E(Z_0)|}\sum_{(i,j)\in E}\|v_i+v_j\|^2 \leqslant \frac{\varepsilon_H\cdot|E|}{|E(Z_0)|}.$$

Thus,

$$\mathbb{E}[|E(Z_0^+)|+|E(Z_0^-)|\,|r]$$
$$\leqslant O\left(\sqrt{\varepsilon_H|E|/|E(Z_0)}\right)|E(Z_0)| \leqslant O(\sqrt{\varepsilon_H})|E|.$$

Combining the above upper bounds, we conclude that

$$\mathbb{E}[\mathsf{uncut}(X,Y)] \leqslant O\left(\frac{\varepsilon_H}{\Delta}+\sqrt{\varepsilon_H}\right).$$

Finally, we estimate the size of the set X. Note that if $v_i\in Z_k^+$ then $|\langle v_i,v_0\rangle - k\Delta'|\leqslant 3\Delta'$, if $v_i\in Z_k^-$ then $|\langle v_i,v_0\rangle + k\Delta'|\leqslant 3\Delta'$. Therefore, $\sum_{i\in Z_k}\langle v_i,v_0\rangle \leqslant k(|Z_k^+|-|Z_k^-|)\Delta' + 3\Delta'|Z_k|$, which implies

$$\begin{aligned}\sum_k k(|Z_k^+|-|Z_k^-|)\Delta' &\geqslant \sum_{i\in U}\langle v_i,v_0\rangle - 3\Delta'|U| \\ &\geqslant (1-2\theta-3\Delta')|U|.\end{aligned}$$

Therefore,

$$\begin{aligned}|Y|-|X| = \sum_k \left||Z_k^+|-|Z_k^-|\right| &\geqslant \sum_{k:|Z_k^+|-|Z_k^-|>0}(|Z_k^+|-|Z_k^-|) \\ &\geqslant \sum_{k:|Z_k^+|-|Z_k^-|>0}(k\Delta')(|Z_k^+|-|Z_k^-|) \\ &\geqslant \sum_k k(|Z_k^+|-|Z_k^-|)\Delta' \geqslant (1-2\theta-3\Delta')|U|,\end{aligned}$$

implying $|X|\leqslant(\theta+3\Delta'/2)|U| = (\theta+\Delta)|U|$. □

We apply this algorithm to every graph $G[U_i]$ and every $\theta = k/|U_i|$, $0<k\leqslant|U_i|/2$, and obtain a list of candidate cuts. We get the following corollary.

Corollary 4.4. *There is a polynomial time algorithm that given a graph $H = G([U_i])$ and a parameter $\Delta\in(0,1)$ finds a family of cuts $\mathcal{U}_i$ such that for every cut (S,T) of H there exists a cut $(X,Y)\in\mathcal{U}_i$ with $|X|\leqslant\min(|S|,|T|)+\Delta|U_i|$ and*

$$\mathsf{uncut}(X,Y)\leqslant O\left(\sqrt{\mathsf{uncut}(S,T)}+\frac{\mathsf{uncut}(S,T)}{\Delta}\right).$$

5 Combining candidate cuts

In this section, we show how to choose one candidate cut for each set U_i and V_j.

For brevity, we denote $W_i = U_i$ for $i\in\{1,\dots,p\}$ and $W_{p+j} = V_j$ for $j\in\{1,\dots,q\}$. Similarly, $\mathcal{W}_i = \mathcal{U}_i$ for $i\in\{1,\dots,p\}$ and $\mathcal{W}_{p+j} = \mathcal{V}_j$ for $j\in\{1,\dots,q\}$ Then $W_1,\dots,W_{p+q}$ is a partitioning of V, and $\mathcal{W}_i$ is a family of cuts of $G[W_i]$.

We say that a cut (X,Y) of G is a combination of candidate cuts from $\mathcal{W}_i$ if the restriction of (X,Y) to each W_i belongs to $\mathcal{W}_i$ (we identify cuts (S,T) and (T,S)).

Lemma 5.1. *There exists a polynomial time algorithm that given a graph $G=(V,E)$ and a threshold $\zeta\in[0,1/2]$, sets W_i and families of cuts $\mathcal{W}_i$, finds the maximum cut among all combination cuts (X,Y) with $|X|,|Y|\in[(1/2-\zeta)|V|,(1/2+\zeta)|V|]$.*

Proof. We solve the problem by dynamic programming. Denote $H_k = G[\bigcup_{i=1}^k W_i]$. For every $a\in\{1,\dots,p+q\}$ and $b\in\{1,\dots,|G[H_a]|\}$, let $Q[a,b]$ be the size of the maximum cut among all combination cuts (X,Y) on H_a with $|X|=b$ ($Q[a,b]$ equals $-\infty$ if there are no such cuts). We loop over all value of a from 1 to $p+q$ and fill out the table Q using the following formula

$$Q[a,b] = \max_{(X,Y)\in\mathcal{W}_a \text{ or } (Y,X)\in\mathcal{W}_a} (Q[a-1,b-|X|]+|\mathsf{edges}(X,Y)|),$$

where we assume that $Q[0,0]=0$, and $Q[a,b]=-\infty$ if $a\leqslant 0$ and $b\leqslant 0$ and $(a,b)\neq(0,0)$.

Finally the algorithm outputs maximum among $T[p+q,\lceil(1/2-\zeta)|V|\rceil],\dots,T[p+q,\lfloor(1/2+\zeta)|V|\rfloor]$, and the corresponding combination cut. □

Finally, we prove that there exists a good almost balanced combination cut.

Lemma 5.2. *Let $G=(V,E)$ be a graph. Let $V=\bigcup_i U_i\cup\bigcup_j V_j$ be a partitioning of V that satisfies conditions of Lemma 3.1, and $\mathcal{U}_i$ and $\mathcal{V}_j$ be families of candidate cuts that satisfy conditions of Corollary 4.4 and Lemma 4.2, respectively. Then there exists a composition cut (X,Y) such that*

$$\left|\frac{|X|}{|V|}-\frac{1}{2}\right|\leqslant\max(\Delta,\delta)$$

and

$$\begin{aligned}\mathsf{uncut}(X,Y)\leqslant O\Big(&\sqrt{\lambda}\log(1/\delta)\\ &+\sqrt{\mathsf{uncut}(S_{OPT},T_{OPT})}\\ &+\mathsf{uncut}(S_{OPT},T_{OPT})\Big(\frac{1}{\lambda}+\frac{1}{\Delta}\Big)\Big),\end{aligned}$$

where (S_{OPT},T_{OPT}) is the optimal bisection of G.

Proof. Consider the optimal bisection cut (S_{OPT}, T_{OPT}). We choose a candidate cut for every set V_i. By Lemma 4.2, for every V_i there exists a cut $(X_i, Y_i) \in \mathcal{V}_i$ such that

$$\begin{aligned} & \mathsf{uncut}(X_i, Y_i) \\ \leqslant\ & 4\mathsf{uncut}(S_{OPT} \cap V_i, T_{OPT} \cap V_i)/\lambda_2(G[V_i]) \\ \leqslant\ & 4\mathsf{uncut}(S_{OPT} \cap V_i, T_{OPT} \cap V_i)/\lambda, \end{aligned} \tag{5.1}$$

and $|X_i| = \min(|S_{OPT} \cap V_i|, |T_{OPT} \cap V_i|)$. We define sets X^V and Y^V as follows. For each i, we add X_i to X^V if $|X_i| = |S_{OPT} \cap V_i|$, and we add Y_i to X^V, otherwise (i.e. if $|Y_i| = |S_{OPT} \cap V_i|$). Similarly, we add Y_i to Y^V if $|Y_i| = |T_{OPT} \cap V_i|$, and we add X_i to Y^V, otherwise. Clearly, (X^V, Y^V) is a candidate cut of $\bigcup_i V_i$ and $|X^V| = |S_{OPT} \cap \bigcup_i V_i|$. Assume without loss of generality that $|X^V| \geqslant |Y^V|$.

Now we choose a candidate cut for every set U_i. By Corollary 4.4, for every U_i there exists a cut $(X'_i, Y'_i) \in \mathcal{U}_i$ such that

$$\begin{aligned} & \mathsf{uncut}(X'_i, Y'_i) \\ & \leqslant O\Big(\sqrt{\mathsf{uncut}(S_{OPT} \cap U_i, T_{OPT} \cap U_i)} \\ & \quad + \frac{\mathsf{uncut}(S_{OPT} \cap U_i, T_{OPT} \cap U_i)}{\Delta}\Big), \end{aligned} \tag{5.2}$$

and $|X'_i| \leqslant \min(|S_{OPT} \cap U_i|, |T \cap U_i|) + \Delta|U_i|$. We assume that X'_i is the smaller of the two sets X'_i and Y'_i.

We want to add one of the sets X'_i and Y'_i to X^V, and the other set to Y^V so that the resulting cut (X, Y) is almost balanced. We set $X = X^V$ and $Y = Y^V$. Then consequently for every i from 1 to p, we add X'_i to the larger of the sets X and Y, and add Y'_i to the smaller of the two sets (recall that X'_i is smaller than Y'_i). We obtain a candidate cut (X, Y) of G.

We show that $\big||X|/|V| - 1/2\big| \leqslant \max(\Delta, \delta)$. Initially, $|X| = |X^V| \geqslant |Y| = |Y^V|$. If at some point X becomes smaller than Y then after that $\big||X| - |Y|\big| \leqslant \delta|V|$ since at every step $\big||X| - |Y|\big|$ does not change by more than $|U_i| \leqslant \delta|V|$. So in this case $\big||X|/|V| - 1/2\big| \leqslant \delta$. So let us assume that the set X always remains larger than Y. Then we always add X'_i to X and Y'_i to Y. We have

$$\begin{aligned} |X| &= \Big|X^V \cup \bigcup_i X'_i\Big| \\ &\leqslant \sum_{j=1}^{q} |S_{OPT} \cap V_j| + \\ & \quad \sum_{i=1}^{p} (\min(|S_{OPT} \cap U_i|, |T_{OPT} \cap U_i|) + \Delta|U_i|) \\ &\leqslant \sum_{j=1}^{q} |S_{OPT} \cap V_j| + \sum_{i=1}^{p} |S_{OPT} \cap U_i| + \Delta|V| \\ &= |S_{OPT}| + \Delta|V| = (1/2 + \Delta)|V|. \end{aligned}$$

It remains to bound $\mathsf{uncut}(X, Y)$. We have,

$$\begin{aligned} \mathsf{uncut}(X, Y)|E| \leqslant & \sum_{1 \leqslant i < j \leqslant p} |\mathsf{edges}(U_i, U_j)| \\ & + \sum_{1 \leqslant i < j \leqslant q} |\mathsf{edges}(V_i, V_j)| + \sum_{\substack{1 \leqslant i \leqslant p \\ 1 \leqslant j \leqslant q}} |\mathsf{edges}(U_i, V_j)| \\ & + \sum_{1 \leqslant i \leqslant p} \mathsf{uncut}(X'_i, Y'_i)|E(U_i)| + \sum_{1 \leqslant j \leqslant q} \mathsf{uncut}(X_j, Y_j)|E(V_j)|\,. \end{aligned}$$

By Lemma 3.1, the sum of the first three terms is at most $O(\sqrt{\lambda}\log(1/\delta))|E|$. From (5.2), we get

$$\begin{aligned} & \sum_{1 \leqslant i \leqslant p} \mathsf{uncut}(X'_i, Y'_i)|E(U_i)| \\ & \leqslant O(1) \sum_{i=1}^{p} \Big(\sqrt{\mathsf{uncut}(S_{OPT} \cap U_i, T_{OPT} \cap U_i)} \\ & \quad + \mathsf{uncut}(S_{OPT} \cap U_i, T_{OPT} \cap U_i)/\Delta\Big)|E(U_i)| \\ & = O(1) \sum_{i=1}^{p} \sqrt{(|E(S_{OPT} \cap U_i)| + |E(T_{OPT} \cap U_i)|)} \\ & \quad \cdot \sqrt{|E(U_i)|} \\ & \quad + O(1) \sum_{i=1}^{p} \frac{|E(S_{OPT} \cap U_i)| + |E(T_{OPT} \cap U_i)|}{\Delta} \\ & \leqslant O(1) \sqrt{\sum_{i=1}^{p} (|E(S_{OPT} \cap U_i)| + |E(T_{OPT} \cap U_i)|)} \\ & \quad \cdot \sqrt{\sum_{i=1}^{p} |E(U_i)|} + O\left(\frac{\mathsf{uncut}(S_{OPT}, T_{OPT}) \cdot |E|}{\Delta}\right) \\ & \leqslant O\Big(\sqrt{\mathsf{uncut}(S_{OPT}, T_{OPT})} \\ & \quad + \frac{\mathsf{uncut}(S_{OPT}, T_{OPT})}{\Delta}\Big) \cdot |E|. \end{aligned}$$

From (5.1), we get

$$\begin{aligned} & \sum_j \mathsf{uncut}(X_j, Y_j)|E(V_j)| \\ & \leqslant \sum_j \frac{4 \cdot \mathsf{uncut}(S_{OPT} \cap V_j, T_{OPT} \cap V_j)|E(V_j)|}{\lambda} \end{aligned}$$

$$\leqslant \frac{4 \cdot \mathsf{uncut}(S_{OPT}, T_{OPT})}{\lambda} \cdot |E| .$$

□

6 The bisection algorithm – proof of Theorem 1.2

First, we run the preprocessing step described in Section 3.1. Then we use the algorithm from Lemma 3.1 with $\lambda = \varepsilon_{OPT}^{2/3}$ and $\delta = \varepsilon_{OPT}$ to find a partition of V into sets $U_1, \dots, U_p, V_1, \dots, V_q$. We apply Corollary 4.4 with $\Delta = \sqrt{\varepsilon_{OPT}}$ to all sets U_i, and obtain a list $\mathcal{U}_i$ of candidate cuts for each set U_i. Then we apply Lemma 4.2 and obtain a list $\mathcal{V}_j$ of candidate cuts for each set V_j. Finally, we find the optimal combination of candidate cuts using Lemma 5.1. Denote it by (X, Y). By Lemma 5.2, we get that $\mathsf{uncut}(X, Y)$ is at most

$$O\left(\sqrt{\lambda}\log(1/\delta) + \sqrt{\varepsilon_{OPT}} + \frac{\varepsilon_{OPT}}{\lambda} + \frac{\varepsilon_{OPT}}{\Delta}\right)$$
$$\leqslant O(\sqrt[3]{\varepsilon_{OPT}}\log(1/\varepsilon_{OPT})),$$

and

$$\left|\frac{|X|}{|V|} - \frac{1}{2}\right| \leqslant \max(\Delta, \delta) = O(\sqrt{\varepsilon_{OPT}}).$$

By moving at most $O(\sqrt{\varepsilon_{OPT}})|V|$ vertices of the smallest degree from the larger size of the cut to smaller part of the cut, we obtain a balanced cut. By doing so, we increase the number of uncut edges by at most $O(\sqrt{\varepsilon_{OPT}}|E|)$. The obtained bisection cut cuts a $1 - O(\sqrt[3]{\varepsilon_{OPT}}\log(1/\varepsilon_{OPT}))$ fraction of all edges.

It is easy to see that a slight modification of the algorithm leads to the following extension of Theorem 1.2.

Theorem 6.1. *There is a randomized polynomial time algorithm that given an edge-weighted graph G with a β-biased cut of value $(1 - \varepsilon)$ finds a β-biased cut of value $(1 - O(\sqrt[3]{\varepsilon}\log(1/\varepsilon) + \sqrt{\varepsilon}/(1 - \beta)))$.*

Proof. We use the algorithm above, while changing DP algorithm used by Lemma 5.1 to find the best combination with bias $\beta \pm t$ (where $t = O(\sqrt{\varepsilon})$). We modify the proof of Lemma 5.2 to show that there exists a $\beta \pm t$ cut of value $1 - O(\sqrt{\varepsilon})$. As previously, we first find sets $X = X^V$ and $Y = Y^V$ with $|X^V| \geqslant |Y^V|$. Now, however, if $|X| - |Y| > \beta|V|$ then we add X_i' to X and Y_i' and Y; otherwise, we add X_i' to Y and Y_i' and X. We argue again that if at some point the difference $\big||X| - |Y|\big|$ becomes less than $O(\sqrt{\varepsilon}|V|)$, then after that $\big||X| - |Y|\big| = O(\sqrt{\varepsilon}|V|)$, and therefore, we find a cut with bias $\beta + O(\sqrt{\varepsilon})$. Otherwise, there are two possible cases: either we always have $|X| - |Y| > \beta|V|$, and then we always add X_i' to X and Y_i' to Y, or we always have $|X| - |Y| > \beta|V|$, and then we always add X_i' to Y and Y_i' to X. Note, however, that in both cases $\big||X| - |Y| - \beta|V|\big|$ decreases by $|Y_i'| - |X_i'| \geqslant \big||S_{OPT} \cap U_i| - |T_{OPT} \cap U_i|\big| - 2\Delta|U_i|$ after each iteration. Thus after all iterations, the value of $\big||X| - |Y| - \beta|V|\big|$ decreases by at least

$$\sum_{i=1}^{p} \big||S_{OPT} \cap U_i| - |T_{OPT} \cap U_i|\big| - 2\Delta|U_i|$$
$$\geqslant \Big||S_{OPT} \cap \bigcup_i U_i| - |T_{OPT} \cap \bigcup_i U_i|\Big|$$
$$- 2\Delta\Big|\bigcup_i U_i\Big|.$$

Taking into the account that $|X^V| - |Y^V| = |S_{OPT} \cap \bigcup_i V_i| - |T_{OPT} \cap \bigcup_i V_i|$, we get the following bound for the bias of the final combination cut (X, Y),

$$\big||X| - |Y| - \beta|V|\big|$$
$$\leqslant \Big||S_{OPT} \cap \bigcup_i V_i| - |T_{OPT} \cap \bigcup_i V_i| - \beta|V|\Big|$$
$$- \Big||S_{OPT} \cap \bigcup_i U_i| + |T_{OPT} \cap \bigcup_i U_i|\Big| + 2\Delta\Big|\bigcup_i U_i\Big|$$
$$\leqslant \big||S_{OPT}| - |T_{OPT}| - \beta|V|\big| + 2\Delta|V| = 2\Delta|V|.$$

We get the exact β-biased cut by moving at most $O(\sqrt{\varepsilon})|V|$ vertices of the smallest degree from the larger size of the cut to smaller part of the cut. By doing so, we lose at most $O(\sqrt{\varepsilon})|E|/(1 - \beta)$ cut edges. Therefore the theorem follows. □

7 Improving the performance when small (constant) imbalance is tolerated

In this section we prove Theorem 1.3. By applying the sparsification procedure described in Section 3.1, we may assume that the graph G is unweighted.

We begin by describing a difference graph decomposition procedure that will be used to prove

Theorem 1.3. This will ensure a slightly different expansion property for the expanding parts, which we define first.

7.1 (α, γ)-expansion

For a subset $W \subseteq V$, we denote by $\mu(W) = \frac{|W|}{|V|}$ the *measure* of W. Note that this measure is with respect to the uniform distribution and *not* the stationary distribution on vertices (which is used by definition of vol W).

Definition 7.1. *Let $0 < \alpha, \gamma < 1$. A graph $G = (V, E)$ is said to be an (α, γ)-expander if for every subset $S \subset V$ with $\alpha \leqslant \frac{|S|}{|V|} \leqslant 1/2$, one has $|\mathsf{edges}(S, V \setminus S)| \geqslant \gamma \cdot |E|$.*

An induced subgraph $G[W]$ of is said to be an (α, γ)-expander (relative to G) if for every $S \subset W$ with measure $\alpha \leqslant \frac{|S|}{|W|} \leqslant 1/2$, one has $|\mathsf{edges}(S, W \setminus S)| \geqslant \gamma \cdot \mu(W)|E|$.

Similar to spectral expanders, we can also argue that if a graph is an (α, γ)-expander, then any two large cuts in the graph must differ in very few vertices. This is the analog of Lemma 4.1 with the difference being that we measure distance between cuts with respect to the uniform measure instead of the stationary distribution.

Lemma 7.2. *Let $G = (V, E)$ be an (α, γ)-expander. For $\zeta < \gamma/2$, let $(A, \overline{A})$ and $(B, \overline{B})$ are two cuts in G such that at least a fraction $(1 - \zeta)$ of the edges of G cross each of them. Then*

$$\min\{\mu(A \cap \overline{B}) + \mu(\overline{A} \cap B), \mu(A \cap B) + \mu(\overline{A} \cap \overline{B})\} < \alpha.$$

Proof. Assume for definiteness that $|A \cap \overline{B}| + |\overline{A} \cap B| \leqslant |A \cap B| + |\overline{A} \cap \overline{B}|$. Let $X = (A \cap \overline{B}) \cup (\overline{A} \cap B)$; we have $\mu(A \cap \overline{B}) + \mu(\overline{A} \cap B) = \mu(X) \leqslant 1/2$. Any edge leaving X must fail to be in at least one of the two cuts $(A, \overline{A})$ and $(B, \overline{B})$. Thus $|\mathsf{edges}(X, \overline{X})| \leqslant 2\zeta|E| < \gamma|E|$. Since G is an (α, γ)-expander, this implies $\mu(X) < \alpha$. □

Now we prove a similar partition lemma as Lemma 3.1, but partition the graph into (α, γ)-expanders and small parts.

7.2 SDP relaxation for (α, γ)-expansion

Consider the following SDP for a graph $G = (V, E)$ (we identify $V = \{1, 2, \ldots, n\}$), parametrized by α. The SDP is the standard one for the α-balanced separator problem. (Below $\|x\|$ denotes the ℓ_2-norm of a vector x.)

Semidefinite program $\mathsf{SDP}^{(\alpha)}(G)$:

$$\text{Minimize } \frac{1}{4|E|} \sum_{e=(i,j)\in E} \|v_i - v_j\|^2$$

subject to

$$\begin{aligned} \|v_i\| &= 1 \quad \text{for } i = 1, 2, \ldots, n \\ \mathop{\mathbb{E}}_{i,j}[\|v_i - v_j\|^2] &\geqslant 8\alpha(1 - \alpha)\,. \end{aligned} \qquad (7.1)$$

Lemma 7.3. *If G is* not *an (α, γ)-expander, then the above SDP has a feasible solution of cost less than γ.*

Proof. Suppose $S \subset V$ is a subset of density $\mu(S) \in [\alpha, 1/2]$ such that $|\mathsf{edges}(S, \overline{S})| < \gamma \cdot |E|$. Taking $v_i = 1$ for $i \in S$ and $v_i = -1$ for $i \in \overline{S}$ gives a feasible solution to $\mathsf{SDP}^{(\alpha)}(G)$ of value $|\mathsf{edges}(S, \overline{S})|/|E| < \gamma$. □

We now show that a good SDP solution can be rounded into a sparse, somewhat balanced cut $(S, \overline{S})$ via random hyperplane rounding. The method is standard, but the subtlety lies in the fact that the measure we use for the size of S is not related to the edge weights or degrees but just the number of vertices in S. The proof is deferred to Appendix A.

Lemma 7.4. *Let $\alpha \leqslant 1/2$. There is a polynomial time randomized algorithm that given a solution to the above SDP with objective value γ, finds a set $S \subset V$ with $\alpha/2 \leqslant \mu(S) \leqslant 1/2$ satisfying $|\mathsf{edges}(S, \overline{S})| < \frac{6\sqrt{\gamma}}{\alpha}|E|$ with probability at least 0.9.*

In particular, G is not *an $(\alpha/2, 6\sqrt{\gamma}/\alpha)$-expander if the SDP has a solution with value at most γ.*

7.3 The partition lemma for (α, γ)-expanders

We now present the partition algorithm that given an arbitrary graph $G = (V, E)$, partitions the graph into induced subgraphs each of which is either an (α, γ)-expander or is of small measure (at most η), while cutting a small fraction of the edges. Formally, we prove the following.

Lemma 7.5. *For any choice of parameters $0 < \alpha, \gamma, \eta < 1$, there is a polynomial time algorithm that on input a graph $G = (V, E)$, outputs a partition $V = V_1 \cup V_2 \cup \cdots \cup V_r$ such that for i, $1 \leqslant i \leqslant r$, either $\mu(V_i) \leqslant \eta$ or the induced subgraph $G[V_i]$ is an (α, γ)-expander, and the number of edges crossing* *between different parts V_i and V_j for $i \neq j$ is at most $O\Big(\frac{\sqrt{\gamma}}{\alpha^2}\log(1/\eta)\Big)|E|$.*

Proof. Let $n = |V|$. The idea is to recursively decompose the graph using the method of Lemma 7.4 till each part either has at most ηn vertices, or is an (α, γ)-expander as certified by the optimum SDP objective value being larger than γ. For a subset $T \subseteq V$, define the subroutine $\mathsf{Decompose}(G[T])$ on the subgraph induced by T as follows:

1. If $|T| \leqslant \eta n$, return T.
   (In this case, the subgraph is already small enough.)

2. Compute the optimum value of the semidefinite program $\mathsf{SDP}^{(\alpha)}(G[T])$. If this value is at least
   $$\gamma_T \overset{\text{def}}{=} \frac{\gamma \cdot \mu(T)|E|}{|E(T)|},$$
   where $E(T)$ denotes the set of edges in the induced subgraph $G[T]$, return T.
   (In this case, the induced subgraph $G[T]$ is already an (α, γ)-expander by Lemma 7.3.)

3. Otherwise, run the algorithm of Lemma 7.4 (with input an SDP solution of value less than γ_T) to find a partition $T = T_1 \cup T_2$ with
   $$\max\{\mu(T_1), \mu(T_2)\} \leqslant (1-\alpha/2)\mu(T) \leqslant e^{-\alpha/2}\mu(T) \qquad (7.2)$$
   and
   $$|\mathsf{edges}(T_1, T_2)| \leqslant 6\frac{\sqrt{\gamma_T}}{\alpha}|E(T)| = \frac{6\sqrt{\gamma \cdot |E|}}{\alpha} \cdot \sqrt{\mu(T)|E(T)|}\,. \qquad (7.3)$$
   Now recursively call $\mathsf{Decompose}(G[T_1])$ and $\mathsf{Decompose}(G[T_2])$ on the subgraphs induced by T_1 and T_2, and return the union of the parts so produced.

Remark 7.6. For our application, it is crucial that the decomposition returns components that are all either small or *rigid* in the sense that any two good cuts are close to each other. A key idea in the above decomposition procedure is that the threshold for expansion γ_T is not fixed to be a constant, but varies with the graph $G[T]$. In particular, if the graph $G[T]$ is sparser than G, then the threshold for expansion is higher. This is because in a sparse component $G[T]$, two cuts could differ in a large fraction of vertices yet yield roughly the same value. Hence, the threshold for expansion γ_T beyond which $G[T]$ is not further decomposed is higher for a sparse component $G[T]$.

The partition claimed in the lemma is obtained by running $\mathsf{Decompose}(G)$. The algorithm clearly terminates (by virtue of (7.2) and the fact that we stop when the parts have at most ηn vertices). Clearly each part in the final decomposition is either an (α, γ)-expander or has at most ηn vertices. It only remains to establish the bound on the number of edges cut. By (7.3), the total number of edges cut by all the recursive calls at a particular depth j is

$$\frac{6\sqrt{\gamma \cdot |E|}}{\alpha}\sum_{i=1}^{r_j}\sqrt{\mu(T_i)|E(T_i)|}$$

where $G[T_i]$, $i = 1, 2, \ldots, r_j$ are the disjoint induced subgraphs occurring in the decomposition tree at depth j. By Cauchy-Schwartz and the fact that $\sum_{i=1}^{r_j}|E(T_i)| \leqslant |E|$, the above quantity is at most $\frac{6\sqrt{\gamma}}{\alpha} \cdot |E|$. Since the maximum depth of recursive calls is at most $O(\log(1/\eta)/\alpha)$ by (7.2), the total number of edges cut over all levels is at most $O(\log(1/\eta)/\alpha) \cdot \frac{6\sqrt{\gamma}}{\alpha} \cdot |E|$. □

7.4 The algorithm

First, we run the preprocessing step described in Section 3.1. Then we use the algorithm from Lemma 7.5 to find a partition of V into sets $V_1, \ldots, V_q$.

In Section 7.5, we prove the following lemma, where (α, γ)-SDP-expander is defined by Definition 7.10. Notice that by construction, the (α, γ)-expanders produced by Lemma 7.5 are (α, γ)-SDP-expanders.

Lemma 7.7. *Suppose G is the vertex-disjoint union of $G[V_1], \ldots, G[V_r]$ for a partition $V_1, \ldots, V_r$ of V (i.e., G does not contain edges joining V_i and V_j for $i \neq j$). Furthermore, suppose that for every $k \in [r]$, $\mu(V_k) \geqslant \eta$ and the induced subgraph $G[V_k]$ is an (α, γ)-SDP-expander (relative to G). If there exists a β-biased bisection of G that cuts $1-\varepsilon$ of the edges, we can compute $x \in \{\pm 1\}^n$ such that*

$$\Big|\mathop{\mathbb{E}}_i x_i - \beta\Big| \leqslant O(\sqrt{\alpha} + (\varepsilon/\gamma)^{1/5}), \qquad (7.4)$$

$$\mathop{\mathbb{E}}_{(i,j)\in E} \tfrac{1}{4}(x_i - x_j)^2 \geqslant 1 - O(\sqrt{\varepsilon}). \qquad (7.5)$$

The running time is polynomial in n and exponential in r (which is bounded by $1/\eta$).

Let W be the union of expanders, and $U_1, \ldots, U_p$ be the small sets. We enumerate all possible β, and use Lemma 7.7 to find a list of candidate cuts $\mathcal{W}$ for $G[W]$ with parameter β. We use Corollary 4.4 to find candidate cuts $\mathcal{U}_j$ for each $G[U_i]$.

We use Lemma 5.1 to find the best combination of the candidate cuts, with parameter $t = c\delta$ for some constant c. Thus, we only need to prove the following lemma.

Lemma 7.8. *Let $\mathcal{W}$ and $\mathcal{U}_j$ be families of candidate cuts we get by the procedure above. Then there exists a composition cut (X, Y) such that*

- $\mathsf{uncut}(X, Y) \leqslant O\Big(\sqrt{\gamma}\log(1/\eta)/\alpha^2 + \sqrt{\mathsf{uncut}(S_{OPT}, T_{OPT})} + \frac{\mathsf{uncut}(S_{OPT}, T_{OPT})}{\Delta}\Big)$ *and*
- $\Big|\frac{|X|}{|V|} - \frac{1}{2}\Big| \leqslant O\Big(\sqrt{\alpha} + \sqrt{\mathsf{uncut}(S_{OPT}, T_{OPT})} + \Big(\frac{\mathsf{uncut}(S_{OPT}, T_{OPT})}{\gamma}\Big)^{1/5}\Big),$

where (S_{OPT}, T_{OPT}) is the optimal bisection of G.

Proof. The proof goes along the lines of the proof for Lemma 5.2. Instead of choosing candidate cuts in $\mathcal{V}_i$, we need to choose a cut in $\mathcal{W}$ — we just choose the one generated from β^*, which is the bias of optimal solution in $G[W]$. □

Fix $\alpha = \delta^2$, $\gamma = \varepsilon_{OPT}/\delta^5$, $\eta = \delta^2$ and $\delta = \varepsilon_{OPT}$, we prove the following theorem.

Theorem 7.9. *There is a randomized polynomial time algorithm that for any constant $\delta > 0$ given an edge-weighted graph G with a* Max Bisection *of value $(1 - \varepsilon)$ finds a cut (A, B) of value at least $1 - O(\sqrt{\varepsilon})$ satisfying $\big||A|/|V| - 1/2\big| \leqslant O(\delta)$.*

It is easy to generalize Theorem 7.9 to its β-biased cut version, which is Theorem 1.3.

7.5 Improved Approximation with SDP-based Expansion

In this section, we prove Lemma 7.7 by showing how to approximate optimal solution's behavior on expanders when the graph is partitioned by Lemma 7.5. The rough idea is to use Lemma 7.2 to argue that finding max-cuts on the expanders is a good approximation. But in order to get better performance, we need to do some more work.

Observe that the decomposition procedure in Section 7.3 has a stronger guarantee than stated Lemma 7.5. Specifically, each of the components produced by the decomposition is an (α, γ)-expander as certified by the SDP, which is a stricter requirement.

To make this formal, define an (α, γ)-SDP expander as follows:

Definition 7.10. *For vertex set $A \subseteq V$, we say the induced subgraph $G[A]$ is a (α, γ)-SDP-expander (relative to G) if the optimal value of the relaxation $\mathrm{SDP}^{(\alpha)}(G[A])$ is at least $\gamma \cdot \mu(W) \cdot |E|/|E(A)|$. In other words, every embedding $u_1, \ldots, u_n \in \mathbb{R}^n$ with $\mathbb{E}_{i,j\in A}\|u_i - u_j\|^2 \geqslant 8\alpha(1-\alpha)$ satisfies*

$$\sum_{e=(i,j)\in E(A)} \|u_i - u_j\|^2 \geqslant 4\gamma \cdot \mu(A) \cdot |E|\,.$$

Since the semidefinite program $\mathrm{SDP}^{(\alpha)}$ is a relaxation, (α, γ)-SDP-expansion implies (α, γ)-expansion. In other words (α, γ)-SDP expansion is a strictly stronger notion than just (α, γ)-expansion.

Suppose $V_1, \ldots, V_r$ is the partition computed by the decomposition procedure in Section 7.3. Then, the subgraphs $G[V_i]$ with $\mu(V_i) > \eta$ are, in fact, guaranteed to be (α, γ)-SDP-expanders (instead of (α, γ)-expanders).

Now present an algorithm that exploits this property of the decomposition, and therefore prove Lemma 7.7.

Lemma 7.7. (restated) *Suppose G is the vertex-disjoint union of $G[V_1], \ldots, G[V_r]$ for a partition $V_1, \ldots, V_r$ of V (i.e., G does not contain edges joining V_i and V_j for $i \neq j$). Furthermore, suppose that for every $k \in [r]$, $\mu(V_k) \geqslant \eta$ and the induced subgraph $G[V_k]$ is an (α, γ)-SDP-expander (relative to G). If there exists a β-biased bisection of G that cuts $1 - \varepsilon$ of the edges, we can compute $x \in \{\pm 1\}^n$ such that*

$$\Big|\mathop{\mathbb{E}}_{i} x_i - \beta\Big| \leqslant O(\sqrt{\alpha} + (\varepsilon/\gamma)^{1/5}),$$
$$\mathop{\mathbb{E}}_{(i,j)\in E} \tfrac{1}{4}(x_i - x_j)^2 \geqslant 1 - O(\sqrt{\varepsilon}).$$

The running time is polynomial in n and exponential in r (which is bounded by $1/\eta$).

Proof. Let x' be a β-biased bisection of G that cuts $1-\varepsilon$ of the edges. Suppose x' has bias β_k in part V_k, so that $\mathbb{E}_{i\in V_k} x'_i = \beta_k$ and therefore $\mathbb{E}_{i,j\in V_k}(x'_i - x'_j)^2 = 2(1-\beta_k)(1+\beta_k) = 2(1-\beta_k^2)$.

Since there are only r parts, we can enumerate all choices for $\beta_1, \dots, \beta_r$ in time polynomial in n and exponential in r. Hence, we can assume that the biases $\beta_1, \dots, \beta_r$ are known to the algorithm. Let $v_1, \dots, v_n$ be a feasible solution to the following semidefinite program.

$$\sum_{e=(i,j)\in E} \tfrac{1}{4}\|v_i - v_j\|^2 \geqslant (1-\varepsilon)|E| \tag{7.6}$$

$$\mathop{\mathbb{E}}_{i,j\in V_k} \|v_i - v_j\|^2 = 2(1-\beta_k^2). \tag{7.7}$$

Note that the semidefinite program is feasible since the integral cut x' yields a solution. Furthermore, given the values of $\beta_1, \dots, \beta_r$ a feasible solution to the SDP can be found efficiently. We will show that the expansion properties of the graphs $G[V_1], \dots, G[V_r]$ imply that the embedding $v_1, \dots, v_n$ form essentially an *integral solution*. Recall that in the intended integral solution to the semidefinite program, all vertices are assigned either $+1$ or -1, and hence are clustered in two antipodal directions. We will argue that for most parts $G[V_i]$, nearly all of its corresponding vectors are clustered along two anti-podal directions.

Consider the vectors $u_1, \dots, u_n$ defined as $u_i = v_i^{\otimes t}$, where t is the even integer (we determine the best choice for t later). We can upper bound the average length of an edge of G in the embedding $u_1, \dots, u_n$ as follows

$$\mathop{\mathbb{E}}_{(i,j)\in E} \tfrac{1}{4}\|u_i - u_j\|^2 \leqslant O(t\varepsilon)\,.$$

We say a part V_k is *good* if $\mathbb{E}_{i,j\in V_k}\|u_i - u_j\|^2 \leqslant 8\alpha(1-\alpha)$. Otherwise, we say that V_k is *bad*. Let V_{bad} be the union of the bad parts. Consider a bad part $V_k \subseteq V_{bad}$ (i.e., $\mathbb{E}_{i,j\in V_k}\|u_i - u_j\|^2 > 8\alpha(1-\alpha)$). Since $G[V_k]$ is an (α,γ)-SDP-expander, it holds that $\sum_{e=(i,j)\in E(A)} \frac{1}{4}\|u_i - u_j\|^2 \geqslant \gamma\mu(V_k)|E|$. Therefore, we can lower bound the average length of an edge of G in the embedding $u_1, \dots, u_n$ in terms of $\mu(V_{bad})$,

$$\mathop{\mathbb{E}}_{(i,j)\in E} \tfrac{1}{4}\|u_i - u_j\|^2 \geqslant \gamma\mu(V_{bad})\,.$$

It follows that $\mu(V_{bad}) \leqslant O(t\varepsilon/\gamma)$ (which means that the vertices in V_{bad} have negligible influence on the bias of a bisection).

Now we will show that for every *good* part $V_k \subseteq V \setminus V_{bad}$, the vectors are somewhat integral in that they are all clustered around two antipodal points. Consider a good part $V_k \subseteq V \setminus V_{bad}$. The vectors associated with V_k are close to each other on average, i.e., $\mathbb{E}_{i,j\in V_k}\|u_i - u_j\|^2 \leqslant 8\alpha(1-\alpha)$. Since $u_i = v_i^{\otimes t}$, if the vectors u_i are correlated, then the original vectors v_i are highly correlated (or anti-correlated) on average. Formally, suppose two vertices $i, j \in V_k$ satisfy $|\langle v_i, v_j\rangle| \leqslant 1-\delta$, where we choose $\delta = 1/t$. Then, $\langle u_i, u_j\rangle \leqslant (1-\delta)^t \leqslant e^{-t\delta} = 1/e$, which means $\|u_i - u_j\|^2 \geqslant \Omega(1)$.Since $\mathbb{E}_{i,j\in V_k}\|u_i - u_j\|^2 \leqslant 8\alpha(1-\alpha)$, it follows that

$$\mathop{\mathbb{P}}_{i,j\in V_k}\left\{|\langle v_i, v_j\rangle| \leqslant 1-\delta\right\} \leqslant O(\alpha)\,. \tag{7.8}$$

In other words, within a good component $G[V_k]$, at most α-fraction of the pairs of vectors v_i, v_j are less than $1-\delta$ correlated.

The improved bound on the bias stems from the fact that the Goemans-Williamson rounding procedure has a better balance guarantee on near-integral solution. Specifically, consider the following simple rounding procedure: Let g be an independent standard Gaussian vector in $\mathbb{R}^n$. For $i \in V$, define x_i as the sign of $\langle g, v_i\rangle$. The Goemans–Williamson analysis shows that x cuts $1 - O(\sqrt{\varepsilon})$ of the edges in expectation. Another (related) property of this rounding is that for any two vertices i, j with $|\langle v_i, v_j\rangle| \geqslant 1-\delta$,

$$\mathop{\mathbb{E}}_{x}\left|\tfrac{1}{4}(x_i - x_j)^2 - \tfrac{1}{4}\|v_i - v_j\|^2\right| \leqslant O(\sqrt{\delta}) = O(\sqrt{1/t})\,. \tag{7.9}$$

Consider a good part $V_k \subseteq V \setminus V_{bad}$. We will estimate $\mathbb{E}_x\left|\mathbb{E}_{i,j\in V_k} \frac{1}{4}(x_i - x_j)^2 - \frac{1}{2}(1-\beta_k^2)\right|$. The second property (7.9) of the Goemans–Williamson rounding implies that

$$\mathop{\mathbb{E}}_{x}\Bigg|\mathop{\mathbb{E}}_{i,j\in V_k}\left[\tfrac{1}{4}(x_i - x_j)^2 \;\middle|\; |\langle v_i, v_j\rangle| \geqslant 1-\delta\right] - \mathop{\mathbb{E}}_{i,j\in V_k}\left[\tfrac{1}{4}\|v_i - v_j\|^2 \;\middle|\; |\langle v_i, v_j\rangle| \geqslant 1-\delta\right]\Bigg| = O(\sqrt{1/t})\,.$$

On the other hand, we can estimate the effect of conditioning on the event $|\langle v_i, v_j\rangle| \geqslant 1-\delta$ on the expectation over $i, j \in V_k$ (using the fact that V_k is a good part and the observation (7.8)).

$$\Bigg|\mathop{\mathbb{E}}_{i,j\in V_k}\left[\tfrac{1}{4}(x_i - x_j)^2\right] -$$

$$\left| \mathop{\mathbb{E}}_{i,j\in V_k} \left[\tfrac{1}{4}(x_i - x_j)^2 \;\middle|\; |\langle v_i, v_j\rangle| \geqslant 1-\delta \right] \right| \leqslant O(\alpha) \tag{7.10}$$

$$\left| \mathop{\mathbb{E}}_{i,j\in V_k} \left[\tfrac{1}{4}\|v_i - v_j\|^2 \right] - \mathop{\mathbb{E}}_{i,j\in V_k} \left[\tfrac{1}{4}\|v_i - v_j\|^2 \;\middle|\; |\langle v_i, v_j\rangle| \geqslant 1-\delta \right] \right| \leqslant O(\alpha) \tag{7.11}$$

Combining these bounds and using the fact that $\mathbb{E}_{i,j\in V_k} \frac{1}{4}\|v_i - v_j\|^2 = \frac{1}{2}(1-\beta_k^2)$ implies that

$$\mathop{\mathbb{E}}_{x} \left| \mathop{\mathbb{E}}_{i,j\in V_k} \tfrac{1}{4}(x_i - x_j)^2 - \tfrac{1}{2}(1-\beta_k^2) \right| \leqslant O\left(\sqrt{1/t} + \alpha\right) \tag{7.12}$$

Observe that the bias is given by,

$$|\mathop{\mathbb{E}}_{i\in V_k} x_i| = \sqrt{\mathop{\mathbb{E}}_{i,j\in V_k} x_i x_j}$$
$$= \sqrt{1 - 2 \mathop{\mathbb{E}}_{i,j\in V_k} \left(\frac{1 - x_i x_j}{2}\right)} = \sqrt{1 - 2 \mathop{\mathbb{E}}_{i,j\in V_k} \left(\frac{(x_i - x_j)^2}{4}\right)}\,.$$

If f denote the function $f(u) = \sqrt{1-2u}$, then by the above equality the bias $\mathbb{E}_{i\in V_k} x_i = f(\mathbb{E}_{i,j\in V_k} \frac{1}{4}(x_i - x_j)^2)$. Since f satisfies $|f(u) - f(u')| \leqslant O(\sqrt{|u-u'|})$, we can estimate the expected deviation of the bias $\mathbb{E}_x|\mathbb{E}_{i\in V_k} x_i - \beta_k|$ as follows:

$$\mathop{\mathbb{E}}_{x} \left| \mathop{\mathbb{E}}_{i\in V_k} x_i \; - \; \beta_k \right| \tag{7.13}$$
$$= \mathop{\mathbb{E}}_{x} \left| f\left(\mathbb{E}_{i,j\in V_k} \tfrac{1}{4}(x_i - x_j)^2\right) \; - \; f\left(\tfrac{1}{2}(1+\beta_k)(1-\beta_k)\right) \right|$$
$$\leqslant O(1) \mathop{\mathbb{E}}_{x} \left| \mathbb{E}_{i,j\in V_k} \tfrac{1}{4}(x_i - x_j)^2 \; - \; \tfrac{1}{2}(1-\beta_k^2) \right|^{1/2}$$
$$\leqslant O(1) \left(\mathop{\mathbb{E}}_{x} \left| \mathbb{E}_{i,j\in V_k} \tfrac{1}{4}(x_i - x_j)^2 \; - \; \tfrac{1}{2}(1-\beta_k^2) \right| \right)^{1/2}$$
$$\text{(using Cauchy–Schwarz)}$$
$$\leqslant O(\sqrt{1/t} + \alpha)^{1/2} \quad \text{(by (7.12))}$$
$$\leqslant O(t^{-1/4} + \alpha^{1/2})\,. \tag{7.14}$$

Finally, we obtain the following bound on the expected deviation of the bias

$$\mathop{\mathbb{E}}_{x} \left| \mathop{\mathbb{E}}_{i\in V} x_i - \beta \right|$$
$$\leqslant \sum_{k=1}^{r} \mu(V_k) \mathop{\mathbb{E}}_{x} \left| \mathop{\mathbb{E}}_{i\in V_k} x_i - \beta_k \right|$$
$$\text{(using triangle inequality)}$$
$$\leqslant \sum_{\substack{k\in[r] \\ V_k \text{good}}} \mu(V_k) \mathop{\mathbb{E}}_{x} \left| \mathop{\mathbb{E}}_{i\in V_k} x_i - \beta_k \right| + O(t\varepsilon/\gamma)$$
$$\text{(using } \mu(V_{bad}) \leqslant O(t\varepsilon/\gamma))$$
$$\leqslant O\left(t^{-1/4} + \alpha^{1/2} + t\varepsilon/\gamma\right) \quad \text{(by (7.14))}$$
$$\leqslant O(\sqrt{\alpha} + (\varepsilon/\gamma)^{1/5})$$
$$\text{(choosing } t \text{ such that } t^5 = (\gamma/\varepsilon)^4)\,.$$

□

8 Minimum Bisection

The techniques of this work almost immediately imply the following approximation algorithm for the minimum bisection problem.

Theorem 8.1. *Suppose G is a graph with a β-biased bisection cutting ε-fraction of edges then:*

- *For every constant $\delta > 0$ it is possible to efficiently find a bisection with bias $\beta \pm O(\delta)$ cutting at most $O(\sqrt{\varepsilon})$-fraction of edges.*
- *It is possible to efficiently find a β-biased bisection cutting at most $O(\sqrt[3]{\varepsilon}\log(1/\varepsilon) + \sqrt{\varepsilon}/(1-\beta))$-fraction of edges.*

Proof. The result follows essentially along the lines of the corresponding results for Max-Bisection (Theorem 1.2 and Theorem 1.3). We simply need to change the $v_i - v_j$ terms in the SDP formulation and analysis to $v_i + v_j$, and likewise for $x_i - x_j$ in the ± 1 vector x (the cut) found by the algorithm. The details are therefore omitted in this version. □

A Proof of Lemma 7.4

Proof. We will round the vectors into a cut using a random hyperplane. Specifically, pick a random vector $r \in (N(0,1))^n$ at random, and let $P = \{i \mid \langle v_i, r\rangle \geqslant 0\}$. We will let S be the smaller of the two sets $P, V \setminus P$. Clearly $\mu(S) \leqslant 1/2$. We will now bound the probability that $\mu(S) < \alpha/2$ or the cut $(S, \overline{S})$ has too many edges crossing it.

The expected number of edges crossing the cut is

$$\mathbb{E}\,|\mathsf{edges}(S, \overline{S}))| = \sum_{(i,j)\in E} \frac{\arccos\langle v_i, v_j\rangle}{\pi}$$
$$= |E| \mathop{\mathbb{E}}_{(i,j)\in E} \left[\frac{\arccos\langle v_i, v_j\rangle}{\pi} \right] \tag{A.1}$$

As the SDP objective value is γ, we have $\mathbb{E}_{(i,j)\in E}[\|v_i - v_j\|^2] \leqslant 4\gamma$, or equivalently

$\mathbb{E}_{(i,j)\in E}[\langle v_i, v_j\rangle] \geqslant 1-2\gamma$. By an averaging argument,

$$\mathop{\mathbb{P}}_{(i,j)\in E}[\langle v_i, v_j\rangle \leqslant 0] \leqslant 2\gamma\,. \tag{A.2}$$

Now

$$\begin{aligned}
&\mathop{\mathbb{E}}_{(i,j)\in E}\left[\frac{\arccos\langle v_i, v_j\rangle}{\pi}\right]\\
\leqslant\ & \mathop{\mathbb{P}}_{(i,j)\in E}[\langle v_i, v_j\rangle \leqslant 0]\\
&\quad + \mathop{\mathbb{E}}_{(i,j)\in E}\left[\frac{\arccos\langle v_i, v_j\rangle}{\pi}\,\middle|\,\langle v_i, v_j\rangle > 0\right]\\
\leqslant\ & 2\gamma + \frac{1}{\pi}\arccos\Big(\mathop{\mathbb{E}}_{(i,j)\in E}[\langle v_i, v_j\rangle \mid \langle v_i, v_j\rangle > 0]\Big)\\
&\qquad\text{(using (A.2) and Jensen)}\\
\leqslant\ & 2\gamma + \frac{\arccos(1-2\gamma)}{\pi}\\
\leqslant\ & 2\gamma + \frac{2}{\pi}\sqrt{\gamma} < 3\sqrt{\gamma}\,.
\end{aligned}$$

By Markov inequality, the probability that $|\mathsf{edges}(S,\overline{S})| \geqslant \frac{6\sqrt{\gamma}}{\alpha}\cdot|E|$ is at most $\alpha/2$.

Let us now analyze the size of S. Clearly $\mathbb{E}[|P|] = |V|/2$. Also

$$\begin{aligned}
\mathbb{E}[|P||V\setminus P|] &= \frac{1}{2}\sum_{i,j}\frac{\arccos\langle v_i, v_j\rangle}{\pi}\\
&\geqslant 0.878\cdot\sum_{i,j}\frac{1-\langle v_i, v_j\rangle}{2}\\
&\qquad\text{(by Goemans-Williamson analysis)}\\
&\geqslant 0.878\cdot 2\alpha(1-\alpha)|V|^2 \quad \text{(using (7.1))}
\end{aligned}$$

Thus $\mathbb{E}[\mu(P)] = 1/2$ and $\mathbb{E}[\mu(P)(1-\mu(P))] \geqslant 0.878\cdot 2\alpha(1-\alpha) \geqslant 3\alpha/4$ since $\alpha \leqslant 1/2$. Note that $\mu(S) = 1/2 - |\mu(P) - 1/2|$. Hence

$$\begin{aligned}
\mathbb{P}[\mu(S) < \alpha/2] &= \mathbb{P}\left[\left|\mu(P) - \frac{1}{2}\right| > \frac{1}{2} - \frac{\alpha}{2}\right]\\
&< \frac{\mathbb{E}[(\mu(P)-1/2)^2]}{1/4 - \frac{\alpha}{2}(1-\alpha/2)}\\
&\leqslant \frac{1/4 - 3\alpha/4}{1/4 - \alpha/2} \leqslant 1-\alpha\,.
\end{aligned}$$

We can thus conclude that with probability at least $\alpha/2$, $\mu(S) \geqslant \alpha/2$ *and* $|\mathsf{edges}(S,\overline{S})| < \frac{6\sqrt{\gamma}}{\alpha}\cdot|E|$. Repeating the rounding procedure $O(1/\alpha)$ times boosts the success probability to 90%. □

References

[AKK+08] Sanjeev Arora, Subhash Khot, Alexandra Kolla, David Steurer, Madhur Tulsiani, and Nisheeth K. Vishnoi, *Unique games on expanding constraint graphs are easy*, Proceedings of the 40th Annual ACM Symposium on Theory of Computing, 2008, pp. 21–28. 3

[Chu96] F. R. K. Chung, *Laplacians of graphs and Cheeger's inequalities*, Combinatorics, Paul Erdős is Eighty **2** (1996), 157–172. 4

[FJ97] Alan M. Frieze and Mark Jerrum, *Improved approximation algorithms for max k-cut and max bisection*, Algorithmica **18** (1997), no. 1, 67–81. 2, 3

[FK02] Uriel Feige and Robert Krauthgamer, *A polylogarithmic approximation of the minimum bisection*, SIAM J. Comput. **31** (2002), no. 4, 1090–1118. 3

[FL06] Uriel Feige and Michael Langberg, *The RPR^2 rounding technique for semidefinite programs*, J. Algorithms **60** (2006), no. 1, 1–23. 2, 3

[GW95] Michel X. Goemans and David P. Williamson, *Improved approximation algorithms for maximum cut and satisfiability problems using semidefinite programming*, Journal of the ACM **42** (1995), no. 6, 1115–1145. 1, 2, 5, 8

[Hås01] Johan Håstad, *Some optimal inapproximability results*, Journal of the ACM **48** (2001), no. 4, 798–859. 2

[HK04] Jonas Holmerin and Subhash Khot, *A new PCP outer verifier with applications to homogeneous linear equations and max-bisection*, Proceedings of the 36th Annual ACM Symposium on Theory of Computing, 2004, pp. 11–20. 2

[HZ02] Eran Halperin and Uri Zwick, *A unified framework for obtaining improved approximation algorithms for maximum graph bisection problems*, Random Struct. Algorithms **20** (2002), no. 3, 382–402. 2

[Kho02] Subhash Khot, *On the power of unique 2-prover 1-round games*, Proceedings of the 34th Annual ACM Symposium on Theory of Computing, 2002, pp. 767–775. 2

[KKMO07] Subhash Khot, Guy Kindler, Elchanan Mossel, and Ryan O'Donnell, *Optimal inapproximability results for MAX-CUT and other 2-variable CSPs?*, SIAM J. Comput. **37** (2007), no. 1, 319–357. 2

[MOO10] Elchanan Mossel, Ryan O'Donnell, and

Krzysztof Oleszkiewicz, *Noise stability of functions with low influences: invariance and optimality*, Annals of Mathematics **171** (2010), no. 1, 295–341. 2

[OW08] Ryan O'Donnell and Yi Wu, *An optimal sdp algorithm for max-cut, and equally optimal long code tests*, Proceedings of the 40th Annual ACM Symposium on Theory of Computing, 2008, pp. 335–344. 2

[Rag08] Prasad Raghavendra, *Optimal algorithms and inapproximability results for every CSP?*, Proceedings of the 40th ACM Symposium on Theory of Computing, 2008, pp. 245–254. 3, 4

[RST10] Prasad Raghavendra, David Steurer, and Madhur Tulsiani, *Reductions between expansion problems*, manuscript, 2010. 3

[TSSW00] Luca Trevisan, Gregory B. Sorkin, Madhu Sudan, and David P. Williamson, *Gadgets, approximation, and linear programming*, SIAM J. Comput. **29** (2000), no. 6, 2074–2097. 2

[Ye01] Yinyu Ye, *A .699-approximation algorithm for Max-Bisection*, Mathematical Programming **90** (2001), 101–111. 2