Improved approximation algorithms for minimum-weight vertex separators

Uriel Feige* MohammadTaghi Hajiaghayi† James R. Lee‡

Abstract

We develop the algorithmic theory of vertex separators, and its relation to the embeddings of certain metric spaces. Unlike in the edge case, we show that embeddings into L_1 (and even Euclidean embeddings) are insufficient, but that the additional structure provided by many embedding theorems does suffice for our purposes.

We obtain an $O(\sqrt{\log n})$ approximation for min-ratio vertex cuts in general graphs, based on a new semidefinite relaxation of the problem, and a tight analysis of the integrality gap which is shown to be $\Theta(\sqrt{\log n})$. We also prove an optimal $O(\log k)$-approximate max-flow/min-vertex-cut theorems for arbitrary vertex-capacitated multi-commodity flow instances on k terminals. For uniform instances on any excluded-minor family of graphs, we improve this to $O(1)$, and this yields a constant-factor approximation for min-ratio vertex cuts in such graphs. Previously, this was known only for planar graphs, and for general excluded-minor families the best-known ratio was $O(\log n)$.

These results have a number of applications. We exhibit an $O(\sqrt{\log n})$ pseudo-approximation for finding balanced vertex separators in general graphs. In fact, we achieve an approximation ratio of $O(\sqrt{\log \mathsf{opt}})$ where opt is the size of an optimal separator, improving over the previous best bound of $O(\log \mathsf{opt})$. Likewise, we obtain improved approximation ratios for treewidth: in any graph of treewidth k, we show how to find a tree decomposition of width at most $O(k\sqrt{\log k})$, whereas previous algorithms yielded $O(k \log k)$. For graphs excluding a fixed graph as a minor (which includes, e.g. bounded genus graphs), we give a constant-factor approximation for the treewidth. This in turn can be used to obtain polynomial-time approximation schemes for several problems in such graphs.

*Microsoft Research, Redmond, Washington, and Department of Computer Science and Applied Mathematics, Weizmann Institute, Rehovot, Israel. urifeige@microsoft.com, uriel.feige@weizmann.ac.il.

†Computer Science and Artificial Intelligence Laboratory, Massachusetts Institute of Technology, hajiagha@theory.csail.mit.edu. This work was done while the author was an intern in the Microsoft Research Theory Group.

‡Computer Science and Engineering, University of Washington. jrl@cs.washington.edu. This work was done while the author was an intern in the Microsoft Research Theory Group.

1 Introduction

Given a graph $G = (V, E)$, one is often interested in finding a small "separator" whose removal from the graph leaves two sets of vertices of roughly equal size (say, of size at most $2|V|/3$), with no edges connecting these two sets. The separator itself may be a set of edges, in which case it is called a *balanced edge separator*, or a set of vertices, in which case it is called a *balanced vertex separator*. In the present work, we focus on vertex separators.

Balanced separators of small size are important in several contexts. They are often the bottlenecks in communication networks (when the graph represents such a network), and can be used in order to provide lower bounds on communication tasks (see e.g. [37, 35, 9]). Perhaps more importantly, finding balanced separators of small size is a major primitive for many graph algorithms, and in particular, for those that are based on the divide and conquer paradigm [39, 9, 36].

Certain families of graphs always have small vertex separators. For example, trees always have a vertex separator containing a single vertex. The well known planar separator theorem of Lipton and Tarjan [39] shows that every n-vertex planar graph has a balanced vertex separator of size $O(\sqrt{n})$, and moreover, that such a separator can be found in polynomial time. This was later extended to show that more general families of graphs (any family of graphs that excludes some minor) have small separators [25, 2]. However, there are families of graphs (for example, expander graphs) in which the smallest separator is of size $\Omega(n)$.

Finding the smallest separator is an NP-hard problem (see, e.g. [15]). In the current paper, we study approximation algorithms that find vertex separators whose size is not much larger than the optimal separator of the input graph. These algorithms can be useful in detecting small separators in graphs that happen to have small separators, as well as in demonstrating that an input graph does not have any small vertex separator (and hence, for example, does not have serious bottlenecks for routing).

Much of the previous work on approximating vertex separators piggy-backed on work for approximating edge separators. For graphs of bounded degree, the sizes of the minimum edge and vertex separators are the same up to a constant multiplicative factor, leading to a corresponding similarity in terms of approximation ratios. However, for general graphs (with no bound on the degree), the situation is different. For example, every edge separator for the star graph has $\Omega(n)$ edges, whereas the minimum vertex separator has just one vertex. One can show that approximating vertex separators is at least has hard as approximating edge separators (see [15]). As to the reverse direction, it is only known that approximating vertex separators is at least as easy as approximating edge separators in *directed graphs* (a notion that will not be discussed in this paper).

The previous best approximation ratio for vertex separators is based in the work of Leighton and Rao [36]. They presented an algorithm based on linear programming that approximates the minimum edge separator within a ratio of $O(\log n)$. They observed that their algorithm can be extended to work on directed graphs, and hence gives an approximation ratio of $O(\log n)$ also for vertex separators, using the algorithm for (directed) edge separators as a black box. More recently, Arora, Rao and Vazirani [7] presented an algorithm based on semidefinite programming that approximates the minimum edge separator within a ratio of $O(\sqrt{\log n})$. Their remarkable techniques, which are a principal component in our algorithm for vertex separators, are discussed more in the following section.

In the present work, we formulate new linear and semidefinite program relaxations for the vertex separator problem, and then develop rounding algorithms for these programs. The rounding algorithms are based on techniques that were developed in the context of edge separators, but we exploit new properties of these techniques and adapt and enhance them to the case of vertex separators. Using this approach, we improve the best approximation ratio for vertex separators to $O(\sqrt{\log n})$. In fact, we obtain an $O(\sqrt{\log \mathsf{opt}})$ approximation, where opt is the size of an optimal separator. (An $O(\log \mathsf{opt})$ approximation can be derived from the techniques of [36].) In addition,

we derive and extend some previously known results in a unified way, such as a constant factor approximation for vertex separators in planar graphs (a result originally proved in [5]), which we extend to any family of graphs excluding a fixed minor.

Before delving into more details, let us mention two aspects in which edge and vertex separators differ. One is the notion of a *minimum ratio cut*, which is an important notion used in our analysis. For edge cuts, all "natural" definitions of such a notion are essentially equivalent. For vertex separators, this is not the case. One consequence of this is that our algorithms provide a good approximation for *vertex expansion* in bounded degree graphs, but not in general graphs. This issue will be discussed in Section 2. Another aspect where there is a distinction between edge and vertex separators is that of the role of embeddings into L_1 (a term that will be discussed later). For edge separators, if the linear or semidefinite program relaxations happen to provide such an embedding (i.e. if the solution is an L_1 metric), then they in fact yield an optimal edge separator. For vertex separators, embeddings into L_1 seem to be insufficient, and we give a number of examples that demonstrate this deficiency. Our rounding techniques for the vertex separator case are based on embeddings with small average distortion into a line, a concept that was first systematically studied by Rabinovich [41].

As mentioned above, finding small vertex separators is a basic primitive that is used in many graph algorithms. Consequently, our improved approximation algorithm for minimum vertex separators can be plugged into many of these algorithms, improving either the quality of the solution that they produce, or their running time. Rather than attempting to provide in this paper a survey of all potential applications, we shall present one major application, that of improving the approximation ratio for *treewidth*, and discuss some of its consequences.

1.1 Some related work

An important concept that we shall use is the *ratio* of a vertex separator (A, B, S). Given a weight function $\pi : V \to \mathbb{R}_+$ on vertices, and a set $S \subseteq V$ which separates G into two disconnected pieces A and B, we can define the *sparsity* of the separator by

$$\frac{\pi(S)}{\min\{\pi(A), \pi(B)\} + \pi(S)}.$$

Indeed, most of our effort will focus on finding separators (A, B, S) for which the sparsity is close to minimal among all vertex separators in G.

In the case of edge separators, there are intimate connections between approximation algorithms for minimums ratio cuts and the theory of metric embeddings. In particular, Linial, London, and Rabinovich [38] and Aumann and Rabani [8] show that one can use L_1 embeddings to round the solution to a linear relaxation of the problem. For the case of vertex cuts, we will show that L_1 embeddings (and even Euclidean embeddings) are insufficient, but that the additional structure provided by many embedding theorems does suffice. This structure corresponds to the fact that many embeddings are of *Fréchet-type*, i.e. their basic component takes a metric space X and a subset $A \subseteq X$ and maps every point $x \in X$ to its distance to A. This includes, for instance, the classical theorem of Bourgain [14].

The seminal work of Leighton and Rao [36] showed that, in both the edge and vertex case, one can achieve an $O(\log n)$ approximation algorithm for minimum-ratio cuts, based on a linear relaxation of the problem. Their solution also yields the first approximate max-flow/min-cut theorems in a model with uniform demands. The papers [38, 8] extend their techniques for the *edge case* to non-uniform demands. Their main tool is Bourgain's theorem [14], which states that every n-point metric space embeds into L_1 with $O(\log n)$ distortion.

Recently, Arora, Rao, and Vazirani [7] exhibit an $O(\sqrt{\log n})$ approximation for finding minimum-ratio edge cuts, based on a known semi-definite relaxation of the problem, and a fundamentally new

technique for exploiting the global structure of the solution. This approach, combined with the embedding technique of Krauthgamer, Lee, Mendel, and Naor [32], has been extended further to obtain approximation algorithms for minimum-ratio edge cuts with non-uniform demands. In particular, using [7], [32], and the quantitative improvements of Lee [34], Chawla, Gupta, and Räcke [17] exhibit an $O(\log n)^{3/4}$ approximation. More recently, Arora, Lee, and Naor [6] have improved this bound almost to that of the uniform case, yielding an approximation ratio of $O(\sqrt{\log n}\log\log n)$.

On the other hand, progress on the vertex case has been significantly slower. In the sections that follow, we attempt to close this gap by providing new techniques for finding approximately optimal vertex separators.

Since the initial (conference) publication of this manuscript, we have learned of two other papers which contain some independently discovered, overlapping results. All three papers first appeared in STOC 2005. In particular, the work of Agarwal, et. al. [1] gives an $O(\sqrt{\log n})$-approximation for a *directed* version of the Sparsest Cut problem which implies a similar result for vertex cuts by a well-known reduction (see e.g. [36]). Their algorithm is also based on rounding an SDP (though they use a different relaxation). Secondly, the paper of Chekuri, Khanna, and Shepherd [18] shows the the max-multicommodity-flow/min-vertex-cut gap for product demands *in planar graphs* is bounded by a universal constant. As discussed later, we prove this theorem not only for planar graphs, but for any *excluded-minor family of graphs.*

1.2 Results and techniques

In Section 2, we introduce a new semi-definite relaxation for the problem of finding minimum-ratio vertex cuts in a general graph. In preparation for applying the techniques of [7], the relaxation includes so-called "triangle inequality" constraints on the variables. The program contains strictly more than one variable per vertex of the graph, but the SDP is constructed carefully to lead to a single metric of negative type[1] on the vertices which contains all the information necessary to perform the rounding.

In Section 3, we exhibit a general technique for rounding the solution to optimization problems involving "fractional" vertex cuts. These are based on the ability to find line embeddings with small *average* distortion, as defined by Rabinovich [41] (though we extend his definition to allow for arbitrary weights in the average). In [41], it is proved that one can obtain line embeddings with constant average distortion for metrics supported on planar graphs. This is observed only as an interesting structural fact, without additional algorithmic consequences over the known average distortion embeddings into all of L_1 [42, 31]. For the vertex case, we will see that this additional structure is crucial.

Using the seminal results of [7], which can be viewed as a line embedding (see Appendix A.2), we then show that the solution of the semi-definite relaxation can be rounded to a vertex separator whose ratio is within $O(\sqrt{\log n})$ of the optimal separator. For the SDP used in [7] for approximating minimum-ratio edge cuts, only a constant lower bound is known for the integrality gap. Recent work of Khot and Vishnoi [30] shows that in the non-uniform demand case, the integrality gap must tend to infinity with the size of the instance. In contrast, we show that our analysis is tight by exhibiting an $\Omega(\sqrt{\log n})$ integrality gap for the SDP in Section 5. Interestingly, this gap is achieved by an L_1 metric. This shows that L_1 metrics are not as intimately connected to vertex cuts as they are to edge cuts, and that the use of the structural theorems discussed in the previous paragraph is crucial to obtaining an improved bound.

We exhibit an $O(\log k)$-approximate max-flow/min-vertex-cut theorems for general instances with k commodities. The best previous bound of $O(\log^3 k)$ is due to [22] (they actually show this bound for directed instances with symmetric demands, but this implies the vertex case). The result

[1] A metric space (X, d) is said to be of *negative type* if $d(x, y) = ||f(x) - f(y)||^2$ for some map $f : X \to L_2$.

is proved in Section 4. A well-known reduction shows that this theorem implies the edge version of [38, 8] as a special case. Again, our rounding makes use of the general tools developed in Section 3 based on average-distortion line embeddings. In Sections 4.2 and 4.4, we show that any approach based on low-distortion L_1 embeddings and Euclidean embeddings, respectively, must fail since the integrality gap can be very large even for spaces admitting such embeddings. Using the improved line embeddings for metrics on graphs which exclude a fixed minor [41] (based on [31] and [42]), we also achieve a constant-factor approximation for finding minimum ratio vertex cuts in these families. This answers an open problem asked in [19].

By improving the approximation ratios for balanced vertex separators in general graphs and graphs excluding a fixed minor, we improve the approximation factors for a number of problems relating to graph-theoretic decompositions such as treewidth, branchwidth, and pathwidth. For instance, we show that in any graph of treewidth k, we can find a tree decomposition of width at most $O(k\sqrt{\log k})$. This improves over the $O(\log n)$-approximation of Bodlaender et al. [11] and the $O(\log k)$-approximation algorithm of Amir [4]. A result of Seymour and Thomas [44] shows that a decomposition of width $1.5k$ can be found efficiently in planar graphs. We offer a significant generalization by giving an algorithm that finds a decomposition of width $O(k)$ whenever the input graph excludes a fixed minor. See Section 6.3 and Corollaries 6.4 and 6.5 for a discussion of these problems, along with salient definitions, and a list of the problems to which our techniques apply.

Improving the approximation factor for treewidth in general graphs and graphs excluding a fixed minor to $O(\sqrt{\log n})$ and $O(1)$, respectively, answers an open problem of [19], and leads to an improvement in the running time of approximation schemes and sub-exponential fixed parameter algorithms for several NP-hard problems on graphs excluding a fixed minor. For instance, we obtain the first polynomial-time approximation schemes (PTAS) for problems like *minimum feedback vertex set* and *connected dominating set* in such graphs (see Theorem 6.6 for more details). Finally, our techniques yield an $O(g)$-approximation algorithm for the vertex separator problem in graphs of genus at most g. It is known that such graphs have balanced separators of size $O(\sqrt{gn})$ [25], and that these separators can be found efficiently [28] (earlier, [3] gave a more general algorithm which, in particular, finds separators of size $O(\sqrt{g^{3/2}n})$). Our approximation algorithms thus finds separators of size $O(\sqrt{g^3 n})$, but when the graph at hand has a smaller separator, our algorithms perform much better than the worst-case bounds of [25, 3, 28].

2 A vector program for minimum-ratio vertex cuts

Let $G = (V, E)$ be a graph with non-negative vertex weights $\pi : V \to [1, \infty)$. For a subset $U \subseteq V$, we write $\pi(U) = \sum_{u \in U} \pi(u)$. A vertex separator partitions the graph into three parts, S (the set of vertices in the separator), A and B (the two parts that are separated). We use the convention that $\pi(A) \geq \pi(B)$. We are interested in finding separators that minimize the ratio of the "cost" of the separator to its "benefit." Here, the cost of a separator is simply $\pi(S)$. As to the benefit of a separator, it turns out that there is more than one natural way in which one can define it. The distinction between the various definitions is relatively unimportant whenever $\pi(S) \leq \pi(B)$, but it becomes significant when $\pi(S) > \pi(B)$. We elaborate on this below.

In analogy to the case of edge separators, one may wish to take the benefit to be $\pi(B)$. Then we would like to find a separator that minimizes the ratio $\pi(S)/\pi(B)$. However, there is evidence that no polynomial time algorithm can achieve an approximation ratio of $O(|V|^{\delta})$ for this problem (for some $\delta > 0$). See Appendix A.1 for details.

For the use of separators in divide and conquer algorithms, the benefit is in the reduction in size of the parts that remain. This reduction is $\pi(B) + \pi(S)$ rather than just $\pi(B)$, and the quality

of a separator is defined to be

$$\frac{\pi(S)}{\pi(B) + \pi(S)}.$$

This definition is used in the introduction to our paper, and in some other earlier work (e.g. [5]).

As a matter of convenience, we use a slightly different definition. We shall define the *sparsity* of a separator to be

$$\alpha_\pi(A, B, S) = \frac{\pi(S)}{\pi(A \cup S) \cdot \pi(B \cup S)}.$$

Under our convention that $\pi(A) \geq \pi(B)$, we have that $\pi(V)/2 \leq \pi(A \cup S) \leq \pi(V)$, and the two definitions differ by a factor of $\Theta(\pi(V))$.

We define $\alpha_\pi(G)$ to be the minimum over all vertex separators (A, B, S) of $\alpha_\pi(A, B, S)$. The problem of computing $\alpha_\pi(G)$ is NP-hard (see [15]). Our goal is to give algorithms for finding separators (A, B, S) for which $\alpha_\pi(A, B, S) \leq O(\sqrt{\log k})\, \alpha_\pi(G)$, where $k = |\mathrm{supp}(\pi)|$ is the number of vertices with non-zero weight in G.

Let us pause for a moment to discuss an aspect of approximation algorithms for $\alpha_\pi(G)$ that is often overlooked. The optimal solution minimizing $\alpha_\pi(A, B, S)$ is indeed a nontrivial separator, in the sense that both A and B are nonempty (unless the underlying graph G is a clique). However, when $\pi(S)$ is large relative to $\pi(B)$ in the optimal separator, sets S', B' that only approximately minimize $\alpha_\pi(A', B', S')$ might correspond to trivial separators in the sense that B' is empty. Hence approximation algorithms for $\alpha_\pi(G)$ might return trivial separators rather than nontrivial ones. Whenever this happens, we assume as a convention that the algorithm instead returns a minimum weight vertex cut in G. These cuts are nontrivial, can be found in polynomial time (see Section 3 for example), and the corresponding value of $\alpha_\pi(A, B, S)$ is not larger than that for any trivial separator. (In fact, for trivial separators $\alpha_\pi(A, B, S) = 1/\pi(V)$, whereas for every nontrivial separator, whether optimal or not, one always has $\alpha_\pi(A, B, S) \leq 1/\pi(V)$.)

Before we move onto the main algorithm, let us define

$$\tilde{\alpha}_\pi(A, B, S) = \pi(S)/[\pi(A) \cdot \pi(B \cup S)].$$

Note that $\alpha_\pi(A, B, S)$ and $\tilde{\alpha}_\pi(A, B, S)$ are equivalent up to a factor of 2 whenever $\pi(A) \geq \pi(S)$. Hence in this case it will suffice to find a separator (A, B, S) with $\alpha_\pi(A, B, S) \leq O(\sqrt{\log k})\, \tilde{\alpha}_\pi(G)$. Allowing ourselves to compare $\alpha_\pi(A, B, S)$ to $\tilde{\alpha}_\pi(G)$ rather than $\alpha_\pi(G)$ eases the formulation of the semi-definite relaxations that follow. When $\pi(S) > \pi(A)$, $\tilde{\alpha}$ no longer provides a good approximation to α. However, in this case $\pi(S) > \pi(B)$, and returning a minimum weight vertex cut provides a constant factor approximation to $\alpha_\pi(G)$.

2.1 The quadratic program

We present a quadratic program for the problem of finding min-ratio vertex cuts. All constraints in this program involve only terms that are quadratic (products of two variables). Our goal is for the value of the quadratic program to be equal to $\tilde{\alpha}_\pi(G)$. Let $G = (V, E)$ be a vertex-weighted graph, and let (A^*, B^*, S^*) be an optimal separator according to $\tilde{\alpha}_\pi(\cdot)$, i.e. such that $\tilde{\alpha}_\pi(G) = \tilde{\alpha}_\pi(A^*, B^*, S^*)$.

With every vertex $i \in V$, we associate three indicator 0/1 variables, x_i, y_i and s_i. It is our intention that for every vertex exactly one indicator variable will have the value 1, and that the other two will have value 0. Specifically, $x_i = 1$ if $i \in A^*$, $y_i = 1$ if $i \in B^*$, and $s_i = 1$ if $i \in S^*$. To enforce this, we formulate the following two sets of constraints.

Exclusion constraints. These force at least two of the indicator variables to be 0.

$$x_i \cdot y_i = 0,\ x_i \cdot s_i = 0,\ y_i \cdot s_i = 0,\ \text{for every } i \in V.$$

Choice constraints. These force the non-zero indicator variable to have value 1.

$$x_i^2 + y_i^2 + s_i^2 = 1,\ \text{for all } i \in V.$$

The combination of exclusion and choice constraints imply the following *integrality constraints*, which we formulate here for completeness, even though they are not explicitly included as part of the quadratic program: $x_i^2 \in \{0,1\}$, $y_i^2 \in \{0,1\}$, $s_i^2 \in \{0,1\}$, for all $i \in V$.

Edge constraints. This set of $2\,|E|$ constraints express the fact that there are no edges connecting A and B.

$$x_i \cdot y_j = 0 \text{ and } x_j \cdot y_i = 0,\ \text{for all } (i,j) \in E.$$

Now we wish to express the fact that we are minimizing $\tilde{\alpha}_\pi(A,B,S)$ over all vertex separators (A,B,S). To simplify our presentation, it will be convenient to assume that we know the value $K = \pi(A^*) \cdot \pi(B^* \cup S^*)$. We can make such an assumption because the value of K can be guessed (since eventually we will only need to know K within a factor of 2, say, there are only $O(\log \pi(V))$ different values to try). Alternatively, the assumption can be dropped at the expense of a more complicated relaxation.

Spreading constraint. The following constraint expresses our guess for the value of K.

$$\tfrac{1}{2} \sum_{i,j \in V} \pi(i)\pi(j)(x_i - x_j)^2 \geq K.$$

Notice that $(x_i - x_j)^2 = 1$ if and only if $\{x_i, x_j\} = \{0,1\}$.

The objective function. Finally, we write down the objective we are trying to minimize:

$$\text{minimize} \qquad \frac{1}{K} \sum_{i \in V} \pi(i) s_i^2.$$

The above quadratic program computes exactly the value of $\tilde{\alpha}_\pi(G)$, and hence is NP-hard to solve.

2.2 The vector relaxation

We relax the quadratic program of Section 2.1 to a "vector" program that can be solved up to arbitrary precision in polynomial time. The relaxation involves two aspects.

Interpretation of variables. All variables are allowed to be arbitrary vectors in $\mathbb{R}^d$, rather than in $\mathbb{R}$. The dimension d is not constrained, and might be as large as the number of variables (i.e., $3n$).

Interpretation of products. The original quadratic program involved products over pairs of variables. Every such product is interpreted as an inner product between the respective vector variables. The exclusion constraints merely force vectors to be orthogonal (rather than forcing one of them to be 0), and the integrality constraints are no longer implied by the exclusion and choice constraints. The choice constraints imply (among other things) that no vector has norm greater than 1, and the edge constraints imply that whenever $(i,j) \in E$, the corresponding vectors x_i and y_j are orthogonal.

2.3 Adding valid constraints

We now strengthen the vector program by adding more valid constraints. This should be done in a way that will not violate feasibility (in cases where the original quadratic program was feasible), and moreover, that preserves polynomial time solvability (up to arbitrary precision) of the resulting vector program. It is known that this last condition is satisfied if we only add constraints that are linear over inner products of pairs of vectors, and this is indeed what we shall do. The reader is encouraged to check that every constraint that we add is indeed satisfied by feasible 0/1 solutions to the original quadratic program.

The 1-vector. We add additional variable v to the vector program. It is our intention that variables whose value is 1 in the quadratic program will have value equal to that of v in the vector program. Hence v is a unit vector, and we add the constraint $v^2 = 1$.

Sphere constraints. For every vector variable z we add the constraint $z^2 = v \cdot z$. Geometrically, this forces all vectors to lie on the surface of a sphere of radius $\frac{1}{2}$ centered at $\frac{v}{2}$ because the constraint is equivalent to $(z - \frac{v}{2})^2 = \frac{1}{4}$.

Triangle constraints. For every three variables z_1, z_2, z_3 we add the constraint

$$(z_1 - z_2)^2 + (z_2 - z_3)^2 \geq (z_1 - z_3)^2.$$

This implies that every three variables (which are points on the sphere $S(\frac{v}{2}, \frac{1}{2})$) form a triangle whose angles are all at most $\pi/2$. We remark that we shall eventually use only those triangle constraints in which all three variables are x variables.

Removing the s_i vectors. In the upcoming sections we shall describe and analyze a rounding procedure for our vector program. It turns out that our rounding procedure does not use the vectors s_i, only the values $s_i^2 = 1 - x_i^2 - y_i^2$. Hence we can modify the choice constraints to

$$x_i^2 + y_i^2 \leq 1$$

and remove all explicit mention of the s_i vectors, without affecting our analysis for the rounding procedure. The full vector program follows.

$$\begin{array}{rll}
\text{minimize} & \frac{1}{K} \sum_{i \in V} \pi(i)(1 - x_i^2 - y_i^2) & \\
\\
\text{subject to} & x_i, y_i, v \in \mathbb{R}^{2n}, & i \in V \\
& x_i^2 + y_i^2 \leq 1, & i \in V \\
& x_i \cdot y_i = 0, & i \in V \\
& x_i \cdot y_j = x_j \cdot y_i = 0, & (i,j) \in E \\
& v^2 = 1 & \\
& v \cdot x_i = x_i^2, \ v \cdot y_i = y_i^2, & i \in V \\
& \frac{1}{2} \sum_{i,j \in V} \pi(i)\pi(j)(x_i - x_j)^2 \geq K & \\
& (x_i - x_j)^2 \leq (x_i - x_h)^2 + (x_h - x_j)^2, & h, i, j \in V.
\end{array}$$

In the following section, we will show how to use this SDP to obtain a solution which is within an $O(\sqrt{\log k})$ factor of the best vertex separator. In Section 5, we show that this analysis is tight, even for a family of stronger (i.e. more constrained) vector programs.

3 Algorithmic framework for rounding

In this section, we develop a general algorithmic framework for rounding solutions to optimization problems on vertex cuts.

3.1 Capacities and demands

In the vector program of Section 2, vertices have weights π. These weights served two purposes. One was as a measure of cost for the separator (we are charged $\pi(S)$ in the numerator of α_π). The other as a measure of benefit of the separator (we get credit of $\pi(A \cup S)\pi(B \cup S)$ in the denominator). Here, we shall not insist on having one weight function serving both purposes. Instead, we allow the cost to be measured with respect to one weight function (say, π_1), and the benefit to be measured with respect to another weight function (say, π_2). It is customary to call these functions *capacity* and *demand*. Let us provide more details.

Vertices are assumed to have non-negative capacities $\{c_v\}_{v \in V} \subseteq \mathbb{N}$. For simplicity of presentation, we are assuming here that capacities are integer, but all results of this paper can also be extended to the case of arbitrary nonnegative capacities. For a subset $S \subseteq V$, we define $\mathrm{cap}(S) = \sum_{v \in S} c_v$.

In its most general form, we have a *demand function* $\omega : V \times V \to \mathbb{R}_+$ which is *symmetric*, i.e. $\omega(u, v) = \omega(v, u)$. In interesting special cases, this demand function is induced by weights $\pi_2 : V \to \mathbb{R}_+$ via the relation $\omega(u, v) = \pi_2(u)\pi_2(v)$ for all $u, v \in V$

Given a capacity function and a demand function, we define the *sparsity of* (A, B, S) by

$$\alpha^{\mathrm{cap},\omega}(A, B, S) = \frac{\mathrm{cap}(S)}{\sum_{u \in A \cup S} \sum_{v \in B \cup S} \omega(u, v)}.$$

We define the *sparsity of* G by $\alpha^{\mathrm{cap},\omega}(G) = \min\{\alpha^{\mathrm{cap},\omega}(A, B, S)\}$ where the minimum is taken over all vertex separators. Note that $\alpha_\pi(A, B, S) = \alpha^{\mathrm{cap},\omega}(A, B, S)$ when $c_v = \pi(v)$ and $\omega(u, v) = \pi(u)\pi(v)$ for all $u, v \in V$.

3.2 Line embeddings and distortion

A key feature of the vector program is that its solution associates is a a set of vectors in high dimensional Euclidean space $\mathbb{R}^{2n}$. Moreover, the triangle constraints imply that for the x_i vectors, also the square of their Euclidean distance forms a metric. Technically, such a metric is said to be of *negative type*. Our rounding framework is based on properties of metric spaces.

Let (X, d) be a metric space. A map $f : X \to \mathbb{R}$ is called 1-*Lipschitz* if, for all $x, y \in X$,

$$|f(x) - f(y)| \leq d(x, y).$$

Given a 1-Lipschitz map f and a demand function $\omega : X \times X \to \mathbb{R}_+$, we define its *average distortion under* ω by

$$avd_\omega(f) = \frac{\sum_{x,y \in X} \omega(x, y) \cdot d(x, y)}{\sum_{x,y \in X} \omega(x, y) \cdot |f(x) - f(y)|}.$$

We say that a weight function ω is a *product weight* if it can be written as $\omega(x, y) = \pi(x)\pi(y)$ for all $x, y \in X$, for some $\pi : X \to \mathbb{R}_+$. We now state three theorems which give line embeddings of small average distortion in various settings. The proofs of these theorems are sketched in Appendix A.2.

Theorem 3.1 (Bourgain, [14])**.** *If (X, d) is an n-point metric space, then for every weight function $\omega : X \times X \to \mathbb{R}_+$, there exists an efficiently computable 1-Lipschitz map $f : X \to \mathbb{R}$ with $avd_\omega(f) = O(\log n)$.*

Theorem 3.2 (Rabinovich, [41]). *If (X, d) is any metric space supported on a graph which excludes a K_r-minor, then for every* product weight $\omega_0 : X \times X \to \mathbb{R}_+$*, there exists an efficiently computable* 1*-Lipschitz map $f : X \to \mathbb{R}$ with $avd_{\omega_0}(f) = O(r^2)$.*

Theorem 3.3 (Arora, Rao, Vazirani, [7]). *If (X, d) is an n-point metric of negative type, then for every* product weight $\omega_0 : X \times X \to \mathbb{R}_+$*, there exists an efficiently computable* 1*-Lipschitz map $f : X \to \mathbb{R}$ with $avd_{\omega_0}(f) = O(\sqrt{\log n})$.*

We also recall the following classical result.

Lemma 3.4. *Let (Y, d) be any metric space and $X \subseteq Y$. Given a 1-Lipschitz map $f : X \to \mathbb{R}$, there exists a 1-Lipschitz extension $\tilde{f} : Y \to \mathbb{R}$, i.e. such that $\tilde{f}(x) = f(x)$ for all $x \in X$.*

Proof. One defines

$$\tilde{f}(y) = \sup_{x \in X} [f(x) - d(x, y)]$$

for all $y \in Y$. □

3.3 Menger's theorem

The following classical theorem is an important ingredient in our rounding framework.

Theorem 3.5 (Menger's theorem). *A graph $G = (V, E)$ contains at least k vertex-disjoint paths between two non-adjacent vertices $u, v \in V$ if and only if every vertex cut that separates u from v has size at least k.*

It is well-known that a smallest vertex cut separating u from v can be found in polynomial time (in the size of G) by deriving Menger's Theorem from the Max-Flow-Min-Cut Theorem (see e.g. [45]).

Suppose that, in addition to a graph $G = (V, E)$, we have a set of non-negative vertex capacities $\{c_v\}_{v \in V} \subseteq \mathbb{N}$. (For simplicity, we are assuming here that capacities are integer.) For a subset $S \subseteq V$, we define $\mathrm{cap}(S) = \sum_{v \in S} c_v$. We have the following immediate corollary.

Corollary 3.6. *Let $G = (V, E)$ be a graph with vertex capacities. Then for any two non-adjacent vertices $u, v \in V$, the following two statements are equivalent.*

1. *Every vertex cut $S \subseteq V$ that separates u from v has* $\mathrm{cap}(S) \geq k$.
2. *There exist u-v paths $p_1, p_2, \ldots, p_k \subseteq V$ such that for every $w \in V$,*

$$\#\{1 \leq i \leq k : w \in p_i\} \leq c_w.$$

Furthermore, a vertex cut S of minimal capacity can be found in polynomial time.

Proof. The proof is by a simple reduction. From $G = (V, E)$ and the capacities $\{c_v\}_{v \in V}$, we create a new uncapacitated instance $G' = (V', E')$ and then apply Menger's theorem to G'.

To arrive at G', we replace every vertex $v \in V$ with a collection of representatives $v_1, v_2, \ldots, v_{c_v}$ (if $c_v = 0$, then this corresponds to deleting v from the graph). Now for every edge $(u, v) \in E$, we add edges $\{(u_i, v_j) : 1 \leq i \leq c_u, 1 \leq j \leq c_v\}$. It is not hard to see that every minimal vertex cut either takes all representatives of a vertex or none, giving a one-to-one correspondence between minimal vertex cuts in G and G'. □

Furthermore, given such a capacitated instance $G = (V, E), \{c_v\}_{v \in V}$ along with $u, v \in V$, it is possible to find, in polynomial time, a vertex cut $S \subseteq V$ of minimal capacity which separates u from v.

3.4 Line embeddings and vertex separators

Having presented the tools that we shall be using (line embeddings, Menger's theorem), we present here an algorithmic framework based on an arbitrary line embedding $f : V \to \mathbb{R}$ for finding a vertex cut. Different instantiations of this algorithm may use different line embeddings f. The analysis of this algorithm will use among other things Menger's theorem. It will also involve a certain cost function $\mathsf{cost} : V \to \mathbb{R}_+$ that is left unspecified in this section. However, in later sections (e.g., Section 3.5) the cost of a vertex will be instantiated to be the contribution of the vertex to the objective function of a relaxation of the minimum vertex separator problem (e.g., $\pi(i)(1 - x_i^2 - y_i^2)$ in the vector program). The key technical property of the algorithm is summarized in Lemma 3.7, and it relates between the cost (which is the value of the relaxation) and the sparsity of the cut found by the algorithm. Hence Lemma 3.7 can be used in order to analyze the approximation ratio of algorithms that use this algorithmic framework.

Let $G = (V, E)$ be a graph with vertex capacities $\{c_v\}_{v \in V}$, and a demand function $\omega : V \times V \to \mathbb{R}_+$. Furthermore, suppose that we have a map $f : V \to \mathbb{R}$. We give the following algorithm which computes a vertex cut (A, B, S) in G.

Algorithm FindCut(G, f)

1. Sort the vertices $v \in V$ according to the value of $f(v)$: $\{y_1, y_2, \ldots, y_n\}$.
2. For each $1 \leq i \leq n$,
3. Create the augmented graph $G_i = (V \cup \{s, t\}, E_i)$ with
   $E_i = E \cup \{(s, y_j), (y_k, t) : 1 \leq j \leq i, i < k \leq n\}$.
4. Find the minimum capacity s-t separator S_i in G_i.
5. Let $A_i \cup \{s\}$ be the component of $G[V \cup \{s, t\} \setminus S_i]$ containing s, let $B_i = V \setminus (A_i \cup S_i)$.
6. Output the vertex separator (A_i, B_i, S_i) for which $\alpha^{\mathrm{cap},\omega}(A_i, B_i, S_i)$ is minimal.

The analysis. Suppose that we have a cost function $\mathsf{cost} : V \to \mathbb{R}_+$. We say that the map $f : V \to \mathbb{R}$ is *respects the cost function* cost if, for any $(u, v) \in E$, we have

$$|f(u) - f(v)| \leq \frac{\mathsf{cost}(u) + \mathsf{cost}(v)}{2}. \tag{1}$$

We now move onto the main lemma of this section.

Lemma 3.7 (Charging lemma)**.** *Let $G = (V, E)$ be any capacitated graph with demand function $\omega : V \times V \to \mathbb{R}_+$. Suppose additionally that we have a cost function $\mathsf{cost} : V \to \mathbb{R}_+$ and a map $f : V \to \mathbb{R}$ which respects cost. If α_0 is the sparsity of the minimum ratio vertex cut output by* FindCut(G, f)*, then*

$$\sum_{v \in V} c_v \cdot \mathsf{cost}(v) \geq \alpha_0 \sum_{u,v \in V} \omega(u, v)|f(u) - f(v)|.$$

Proof. Recall that we have sorted the vertices v according to the value of $f(v)$: $\{y_1, y_2, \ldots, y_n\}$. Let $C_i = \{y_1, \ldots, y_i\}$ and $\varepsilon_i = f(y_{i+1}) - f(y_i)$. First we have the following lemma which relates the size of the separators found to the average distance under f, according to ω.

Lemma 3.8.

$$\sum_{i=1}^{n-1} \varepsilon_i \operatorname{cap}(S_i) \geq \alpha_0 \sum_{u,v \in V} \omega(u, v)|f(u) - f(v)|.$$

Proof. Using the fact that α_0 is the minimum sparsity of all cuts found by FindCut(G, f),

$$\begin{aligned}\operatorname{cap}(S_i) &\geq \alpha_0 \sum_{u \in A_i \cup S_i} \sum_{v \in B_i \cup S_i} \omega(u,v) \\ &\geq \alpha_0 \sum_{u \in C_i} \sum_{v \in V \setminus C_i} \omega(u,v).\end{aligned}$$

Note that the second inequality follows from the fact in FindCut(G, f), since C_i contains A_i and $V \setminus C_i$ contains B_i, $A_i \cup S_i$ contains C_i and $B_i \cup S_i$ contains $V \setminus C_i$.

Multiplying both sides of the previous inequality by ε_i and summing over $i \in \{1, 2, \ldots, n-1\}$ proves the lemma. □

Now we come to the heart of the charging argument which relates the cost function to the capacity of the cuts occurring in the algorithm.

Lemma 3.9 (Charging against balls)**.**

$$\sum_{v \in V} c_v \cdot \mathsf{cost}(v) \geq \sum_{i=1}^{n-1} \varepsilon_i \operatorname{cap}(S_i).$$

Proof. We first present an interpretation of the quantity $\sum_{i=1}^{n-1} \varepsilon_i \operatorname{cap}(S_i)$. Consider a nonnegative function g defined on the line segment $[f(y_1), f(y_n)]$ whose value at point z is defined as $g(z) = \operatorname{cap}(S_i)$, where i is the unique value such that z is in the half open interval $[f(y_i), f(y_{i+1}))$. Then $\sum_{i=1}^{n-1} \varepsilon_i \operatorname{cap}(S_i)$ is precisely $\int_{\mathbb{R}} g$.

Now, for every v, we present an interpretation of $c_v \cdot \mathsf{cost}(v)$. Consider a nonnegative function g_v whose value is c_v on the interval $[f(v) - \frac{1}{2}\mathsf{cost}(v), f(v) + \frac{1}{2}\mathsf{cost}(v)]$ and 0 elsewhere. Then $c_v \cdot \mathsf{cost}(v)$ is precisely $\int_{\mathbb{R}} g_v$. We shall refer to the support of g_v as the *ball* of v (as it is a ball centered at $f(v)$ of radius $\frac{1}{2}\mathsf{cost}(v)$).

Lemma 3.9 can now be rephrased as

$$\int_{\mathbb{R}} g(z)\, dz \leq \sum_v \int_{\mathbb{R}} g_v(z)\, dz$$

We shall prove this inequality pointwise.

Consider an arbitrary point z, belonging to an arbitrary interval $[f(y_i), f(y_{i+1}))$. Since S_i is a minimum capacity s-t separator, applying Menger's theorem yields a family of s-t paths $p_1, \ldots, p_m$ (where $m = \operatorname{cap}(S_i)$) which use no vertex $v \in V$ more than c_v times. We view each of these paths as contributing 1 to the value of $g(z)$, and hence fully accounting for the value $g(z) = \operatorname{cap}(S_i)$. We now consider the contribution of these paths to the functions g_v.

Consider an arbitrary such path p_j. Since it crosses from C_i to $V \setminus C_i$, there must exist two consecutive vertices along the path (say u and v) such that $u \in C_i$ and $v \in V \setminus C_i$. The fact that f respects cost implies that the union of the balls of u and v covers the interval $[f(u), f(v)]$ that includes the interval $[f(y_i), f(y_{i+1}))$. Hence z is in at least one of these two balls (say, the ball of v), and then we have p_j contribute one unit to $g_v(z)$. Note that the total contribution of the m disjoint paths to $g_v(z)$ can be at most c_v, because v can occur in at most c_v of these paths.

In summary, based on the disjoint paths, we provided a charging mechanism that accounts for all of $g(z)$, and charges at least as much to $\sum_v g_v(z)$, without exceeding the respective values c_v. This completes the proof of Lemma 3.9. □

Combining Lemmas 3.8 and 3.9 finishes the proof of Lemma 3.7. □

3.5 Analysis of the vector program

We now continue our analysis of the vector program from Section 2.3. Recall that $\pi(i)(1-x_i^2-y_i^2)$ is the contribution of vertex i to the objective function. For every $i \in V$, define $\mathsf{cost}(i) = 4(1-x_i^2-y_i^2)$. We will consider the metric space (V,d) given by $d(i,j) = (x_i - x_j)^2$ (note that this is a metric space precisely because every valid solution to the SDP must satisfy the triangle inequality constraints). The following key proposition allows us to apply the techniques of Sections 3.4 and 3.2 to the solution of the vector program.

Proposition 3.10. *For every edge $(i,j) \in E$, $(x_i - x_j)^2 \le \frac{\mathsf{cost}(i)+\mathsf{cost}(j)}{2}$.*

Proof. Since $(i,j) \in E$, we have $x_i \cdot y_j = x_j \cdot y_i = 0$, and recall that $x_i \cdot y_i = x_j \cdot y_j = 0$. It follows that

$$(x_i - x_j)^2 \le 2[(x_i + y_i - v)^2 + (x_j + y_i - v)^2] \le 2[(1 - x_i^2 - y_i^2) + (1 - x_j^2 - y_i^2)].$$

Note that the first inequality above follows from the fact that $(x_i - x_j)^2 = ((x_i + y_i - v) - (x_j + y_i - v))^2$, and the inequality $(x-y)^2 \le 2(x^2+y^2)$. Substitute $x = x_i + y_i - v$ and $y = x_j + y_i - v$. Then the second inequality follows from the constraints $vx_i = x_i^2$ and $vy_i = y_i^2$.

Putting y_j instead of y_i in the above equation gives $(x_i - x_j)^2 \le 2[(1 - x_i^2 - y_j^2) + (1 - x_j^2 - y_j^2)]$. Summing these two inequalities yields

$$2(x_i - x_j)^2 \le 4[(1 - x_i^2 - y_i^2) + (1 - x_j^2 - y_j^2)] = \mathsf{cost}(i) + \mathsf{cost}(j). \tag{2}$$

□

Now, let $U = \mathrm{supp}(\pi) = \{i \in V : \pi(i) \neq 0\}$, and put $k = |U|$. Finally, let $f : (U,d) \to \mathbb{R}$ be any 1-Lipschitz map, and let $\tilde{f} : V \to \mathbb{R}$ be the 1-Lipschitz extension guaranteed by Lemma 3.4.

Then for any edge $(u,v) \in E$, we have

$$|\tilde{f}(u) - \tilde{f}(v)| \le d(u,v) = (x_u - x_v)^2 \le \frac{\mathsf{cost}(u) + \mathsf{cost}(v)}{2},$$

where the final inequality is from Proposition 3.10. We conclude that $\tilde{f}$ respects cost.

Defining a product demand by $\omega(i,j) = \pi(i)\pi(j)$ for every $i,j \in V$ and capacities $c_i = \pi(i)$, we now apply FindCut$(G, \tilde{f})$. If the best separator found has sparsity α_0, then by Lemma 3.7,

$$\begin{aligned}
\frac{1}{K}\sum_{i\in V}\pi(i)(1-x_i^2-y_i^2) = \frac{1}{4K}\sum_{i\in V} c_i \cdot \mathsf{cost}(i) &\ge \frac{\alpha_0}{4K}\sum_{i,j\in V}\omega(i,j)\cdot|\tilde{f}(i)-\tilde{f}(j)| \\
&= \frac{\alpha_0}{4K}\sum_{i,j\in U}\omega(i,j)\cdot|f(i)-f(j)| \\
&\ge \frac{\alpha_0}{2}\cdot\frac{\sum_{i,j\in U}\omega(i,j)\cdot|f(i)-f(j)|}{\sum_{i,j\in U}\omega(i,j)\cdot d(i,j)} \\
&= \frac{\alpha_0}{2\cdot avd_\omega(f)}.
\end{aligned}$$

It follows that $\tilde{\alpha}_\pi(G) \ge \alpha_0/(2 \cdot avd_\omega(f))$. Since the metric (V,d) is of negative type and $\omega(\cdot,\cdot)$ is a product weight, we can achieve $avd_\omega(f) = O(\sqrt{\log k})$ using Theorem 3.3. Using this f, it follows that FindCut$(G, \tilde{f})$ returns a separator (A,B,S) such that $\alpha_\pi(A,B,S) \le O(\sqrt{\log k})\,\tilde{\alpha}_\pi(G)$, completing the analysis.

Theorem 3.11. *Given a graph $G = (V, E)$ and vertex weights $\pi : V \to \mathbb{R}_+$, there exists a polynomial-time algorithm which computes a vertex separator (A, B, S) for which*

$$\alpha_\pi(A, B, S) \leq O(\sqrt{\log k})\, \alpha_\pi(G),$$

where $k = |\mathrm{supp}(\pi)|$.

In the next section, we extend this theorem to more general weights. This is necessary for some of the applications in Section 6.3.

3.6 More general weights

An important generalization of the min-ratio vertex cut introduced in Section 2 is when a pair of weight functions $\pi_1, \pi_2 : V \to \mathbb{R}_+$ is given and one wants to find the vertex separator (A, B, S) which minimizes

$$\alpha_{\pi_1,\pi_2}(A, B, S) = \frac{\pi_1(S)}{\pi_2(A \cup S) \cdot \pi_2(B \cup S)},$$

where as a convention, $\pi_2(B) \leq \pi_2(A)$. We let $\alpha_{\pi_1,\pi_2}(G)$ denote the minimum value of $\alpha_{\pi_1,\pi_2}(A, B, S)$ in graph G. Under a common interpretation, π_1 denotes vertex capacities, and π_2 induces a *demand* (one needs to route $\pi_2(u)\pi_2(v)$ units of flow between vertices u and v), and then the value of $\alpha_{\pi_1,\pi_2}(G)$ serves as an upper bound on the fraction of demand that can be routed subject to the capacity constraints on the vertices.

In analogy to the discussion in Section 2, call a separator *trivial* if $\pi_2(B) = 0$ (and in particular, when B is empty). Unlike the case in Section 2, when π_1 differs from π_2 it may happen that $\alpha_{\pi_1,\pi_2}(G)$ is obtained by a trivial separator. Hence in the current section, algorithms that minimize (or approximately minimize) $\alpha_{\pi_1,\pi_2}(A, B, S)$ are allowed to return a trivial separator.

We now explain how our approximation algorithm can be extended to give an $O(\sqrt{\log k})$ approximation for $\alpha_{\pi_1,\pi_2}(G)$, where here $k = |\mathrm{supp}(\pi_2)|$.

Let

$$\tilde{\alpha}_{\pi_1,\pi_2}(A, B, S) = \pi_1(S)/[\pi_2(A) \cdot \pi_2(B \cup S)],$$

where $\pi_2(A) \geq \pi_2(B)$. Also define $\alpha_{\pi_1,\pi_2}(G)$ and $\tilde{\alpha}_{\pi_1,\pi_2}(G)$ as before. By changing the vector program so that K is defined in terms of π_2 and the objective is to minimize $\frac{1}{K}\sum_{i \in V} \pi_1(i)(1 - x_i^2 - y_i^2)$, it becomes a relaxation for $\tilde{\alpha}_{\pi_1,\pi_2}(G)$. The rounding analysis goes through unchanged to yield a separator (A, B, S) with

$$\alpha_{\pi_1,\pi_2}(A, B, S) \leq O(\sqrt{\log k})\, \tilde{\alpha}_{\pi_1,\pi_2}(G).$$

One difficulty still remains. It may happen that for the optimal separator (A^*, B^*, S^*), $\pi_2(S^*) \geq \pi_2(A^*)$, and then the values $\alpha_{\pi_1,\pi_2}(A^*, B^*, S^*)$ and $\tilde{\alpha}_{\pi_1,\pi_2}(A^*, B^*, S^*)$, are not within a factor of 2 of each other. In this case we show how to output a (possibly trivial) separator that approximates $\alpha_{\pi_1,\pi_2}(G)$ within constant factors. Observe that in this case

$$\frac{\pi_1(S^*)}{\pi_2(S^*)^2} \leq 4\, \alpha_{\pi_1,\pi_2}(G).$$

Hence it suffices to find an approximation for a different problem, that of finding a subset $S \subseteq V$ which minimizes the ratio $\pi_1(S)/\pi_2(S)^2$. This problem can be solved in polynomial time; see Appendix A.3.

Theorem 3.12. *Given a graph $G = (V, E)$ and vertex weights $\pi_1, \pi_2 : V \to \mathbb{R}_+$, there exists a polynomial-time algorithm which computes a vertex separator (A, B, S) for which*

$$\alpha_{\pi_1,\pi_2}(A, B, S) \leq O(\sqrt{\log k})\, \alpha_{\pi_1,\pi_2}(G),$$

where $k = |\mathrm{supp}(\pi_2)|$.

4 Approximate max-flow/min-vertex-cut theorems

Let $G = (V, E)$ be a graph with capacities $\{c_v\}_{v \in V}$ on vertices and a demand function $\omega : V \times V \to \mathbb{R}_+$. For every pair of distinct vertices $u, v \in V$, let $\mathcal{P}_{uv}$ be the set of all simple u-v paths in G. For $s, t \in V$, an *s-t flow in G* is a mapping $F : \mathcal{P}_{st} \to \mathbb{R}_+$ where for $p \in \mathcal{P}_{st}$, $F(p)$ represents the amount of flow sent from s to t along path p.

For any simple path p in G, let p_0 and p_1 denote the initial and final nodes of p, respectively. By convention, we will assert that for such a flow F and for every $p \in \mathcal{P}_{st}$, the flow path p uses up $\frac{1}{2}F(p)$ of the capacity of p_0 and p_1 and uses up $F(p)$ of the capacity of all other nodes in p. Intuitively, one can think of the loss in capacity for flowing through a vertex to be charged half for entering the vertex and half for exiting, hence the initial and final vertices of a flow path are only charged half. This is made formal in the linear program that follows. We remark that this choice (as opposed to incurring a full loss of capacity in the initial and final nodes) is only for simplicity in the dual linear program; it is easily seen that all the results in this section hold for the other setting, with a possible loss of a factor of 2. To simplify notation, we also define, for any $p \in \mathcal{P}_{uv}$ and $w \in p$, the number $\kappa_p(w)$ to be 1 if w is an intermediate vertex of p and $\frac{1}{2}$ if w is the initial or final vertex of p.

The *maximum concurrent vertex flow* of the instance $(G, \{c_v\}_{v \in V}, \omega)$ is the maximum constant $\epsilon \in [0, 1]$ such that one can simultaneously route an ϵ fraction of each u-v demand $\omega(u, v)$ without violating the capacity constraints. For each $p \in \mathcal{P}_{uv}$, let p^{uv} denote the amount of the u-v commodity that is sent from u to v along p. We now write a linear program that computes the maximum concurrent vertex flow.

$$\begin{aligned}
\text{maximize} \quad & \epsilon \\
\text{subject to} \quad & \sum_{p \in \mathcal{P}_{uv}} p^{uv} \geq \epsilon \cdot \omega(u, v), && u, v \in V \\
& \sum_{u,v \in V} \sum_{p \in \mathcal{P}_{uv} : w \in p} \kappa_p(w) p^{uv} \leq c_w, && w \in V \\
& p^{uv} \geq 0 && u, v \in V, p \in \mathcal{P}_{uv}
\end{aligned}$$

We now write the dual of this LP with variables $\{s_v\}_{v \in V}$ and $\{\ell_{uv}\}_{u,v \in V}$.

$$\begin{aligned}
\text{minimize} \quad & \sum_{v \in V} c_v s_v \\
\text{subject to} \quad & \sum_{w \in p} \kappa_p(w) s_w \geq \ell_{uv} && p \in \mathcal{P}_{uv}, \forall u, v \in V \\
& \sum_{u,v \in V} \omega(u, v) \ell_{uv} \geq 1 \\
& \ell_{uv} \geq 0, s_v \geq 0 && u, v \in V.
\end{aligned}$$

Finally, define

$$\mathsf{dist}(u, v) = \min_{p \in \mathcal{P}_{uv}} \sum_{w \in p} \kappa_p(w) s_w.$$

By setting $\ell_{uv} = \mathsf{dist}(u, v)$, we see that the above dual LP is equivalent to the following.

$$\begin{aligned} \text{minimize} \quad & \sum_{v \in V} c_v s_v \\ \text{subject to} \quad & \sum_{u,v} \omega(u,v) \cdot \mathsf{dist}(u,v) \geq 1. \end{aligned}$$

Remark 4.1. *We remark that the distance function* $\mathsf{dist}(u, v)$ *is a metric which can be alternatively defined as follows: For any* $u, v \in V$, $\mathsf{dist}(u, v)$ *is precisely the (edge-weighted) shortest-path distance in* G *between* u *and* v *where the weight of the edge* $(u, v) \in E$ *is* $\frac{1}{2}(s_u + s_v)$.

4.1 Rounding to vertex separators

Any vertex separator (A, B, S) yields an upper bound on the maximum concurrent flow in G via the following the expression:

$$\frac{\mathrm{cap}(S)}{\sum_{u \in A, v \in B} \omega(u,v) + \sum_{u,v \in S} \omega(u,v) + \frac{1}{2} \sum_{u \in S} \sum_{v \in A \cup B} \omega(u,v)}. \tag{3}$$

The numerator is the capacity of the separator. Every unit of demand served between $u \in A$ and $v \in B$ must consume at least one unit of capacity from S. Likewise, every unit of demand served between $u \in S$ and $v \in S$ must consume at least one unit of capacity from S. Finally, every unit of demand served between $u \in S$ and $v \in A \cup B$ must consume at least half a unit of capacity from S. Hence the denominator is a lower bound on the amount of S's capacity burned by every unit of concurrent flow. We observe that the quantity (3) is bounded between $\alpha^{\mathrm{cap},\omega}(A, B, S)$ and $2 \cdot \alpha^{\mathrm{cap},\omega}(A, B, S)$.

We will write $\alpha = \alpha^{\mathrm{cap},\omega}$ if the capacity and demands are clear from context. For a graph G, we will write $\alpha(G)$ for the minimum of $\alpha(A, B, S)$, where this minimum is taken over all vertex separators in G. We wish to study how tight the upper bound of $2 \cdot \alpha(G)$ is. In order to do so, we take the dual of the max-concurrent-flow LP from the previous section and round it to a vertex separator. The increase in cost incurred by the rounding provides an upper bound on the worst possible ratio between $\alpha(G)$ and the maximum concurrent flow.

We note that the dual LP is a relaxation of the value $2 \cdot \alpha(G)$, since every vertex separator (A, B, S) gives a feasible solution, where $s_v = 1/\lambda$ if $v \in S$ and $s_v = 0$ otherwise. In this case $\mathsf{dist}(u, v) \geq 1/(2\lambda)$ if $u \in A \cup S$ and $v \in B \cup S$ or vice-versa, so that setting $\lambda = \sum_{u \in A \cup S, v \in B \cup S} \omega(u, v)$ yields a feasible solution.

4.2 The rounding

Before presenting our approach for rounding the LP, let us recall a typical rounding approach for the case of *edge-capacitated* flows. In the edge context [38, 8], one observes that the dual LP is essentially integral when $\mathsf{dist}(\cdot, \cdot)$ forms an L_1 metric. To round in the case when $\mathsf{dist}(\cdot, \cdot)$ does not form an L_1 metric, one uses Bourgain's theorem [14] to embed (V, dist) into L_1 (with $O(\log n)$ distortion, that translates to a similar loss in the approximation ratio), and then rounds the resulting L_1 metric (where rounding the L_1 metric does not incur a loss in the approximation ratio). This approach is not as effective in the case of vertex separators, because rounding an L_1 metric does incur a loss in the approximation ratio (as the example below shows), and hence there is not much point in embedding (V, dist) into L_1 and paying the distortion factor.

The discrete cube. Let $G = (V, E)$ be the d-dimensional discrete hypercube $\{0, 1\}^d$. We put $c_v = 1$ for every $v \in V$, and $\omega(u, v) = 1$ for every pair $u, v \in V$. It is well-known that $\alpha(G) =$

$\Theta(1/(2^d\sqrt{d}))$ [27]. On the other hand, consider the fractional separator (i.e. dual solution) given by $s_v = 10 \cdot \frac{4^{-d}}{d}$. Note that $\mathsf{dist}(u,v)$ is proportional to the shortest-path metric on the standard cube, hence $\sum_{u,v} \mathsf{dist}(u,v) \geq 1$, yielding a feasible solution which is a factor $\Theta(\sqrt{d})$ away from $\alpha(G)$. It follows that even when (V, dist) is an L_1 metric, the integrality gap of the dual LP might be as large as $\Omega(\sqrt{\log n})$.

Rounding with line embeddings. The rounding is done as follows. Let $\{s_v\}_{v \in V}$ be an optimal solution to the dual LP, and let $\mathsf{dist}(\cdot,\cdot)$ be the corresponding metric on V. Suppose that the demand function $\omega : V \times V \to \mathbb{R}_+$ is supported on a set S, i.e. $\omega(u,v) > 0$ only if $u, v \in S$, and that $|S| = k$. Let $f : (S, \mathsf{dist}) \to \mathbb{R}$ be the map guaranteed by Theorem 3.1 with $avd_\omega(f) = O(\log k)$, and let $\tilde{f} : (V, \mathsf{dist}) \to \mathbb{R}$ be the 1-Lipschitz extension from Lemma 3.4.

For $v \in V$, define $\mathsf{cost}(v) = s_v$. Then since $\tilde{f}$ is 1-Lipschitz, for any edge $(u,v) \in E$, we have

$$|\tilde{f}(u) - \tilde{f}(v)| \leq \mathsf{dist}(u,v) = \frac{s_u + s_v}{2} = \frac{\mathsf{cost}(u) + \mathsf{cost}(v)}{2}$$

hence $\tilde{f}$ respects cost.

We now apply FindCut$(G, \tilde{f})$. If the best separator found has sparsity α_0, then by Lemma 3.7,

$$\begin{aligned}
\sum_v c_v s_v = \sum_v c_v \cdot \mathsf{cost}(v) &\geq \alpha_0 \sum_{u,v \in V} \omega(u,v)\, |\tilde{f}(u) - \tilde{f}(v)| \\
&= \alpha_0 \sum_{u,v \in S} \omega(u,v)\, |f(u) - f(v)| \\
&\geq \Omega\left(\frac{\alpha_0}{\log k}\right) \sum_{u,v \in V} \omega(u,v)\, \mathsf{dist}(u,v) \geq \Omega\left(\frac{\alpha_0}{\log k}\right).
\end{aligned}$$

Theorem 4.1. *For an arbitrary vertex-capacitated flow instance, where the demand is supported on a set of size k, there is an $O(\log k)$-approximate max-flow/min-vertex-cut theorem. In particular, this holds if there are only k commodities.*

4.3 Excluded minor families

Recall that by Remark 4.1, we can view the metric dist arising from the LP dual as an edge-weighted metric on the graph G. A consequence of this is that if the graph G excludes some fixed graph H as a minor, then the metric dist is an H-excluded metric.

It follows that applying Theorem 3.2, yields a better result when G excludes a minor and the demand function $\omega(u,v)$ is uniform on a subset of the vertices. This special case will be needed later when we discuss treewidth, and follows from the following theorem (because product demands include as a special case demand functions that are uniform on a subset of the vertices).

Theorem 4.2. *When G is an H-minor-free graph, there is an $O(|V(H)|^2)$-approximate max-flow/min-vertex-cut theorem with product demands. Additionally, there exists an $O(|V(H)|^2)$ approximation algorithm for finding min-quotient vertex cuts in G.*

4.4 More integrality gaps for uniform demands

Expanders. Our analysis for the integrality gap of the dual LP is tight. Just as in the edge case, constant-degree expander graphs provide the bad example. If $G = (V, E)$ is such a graph, with

uniform vertex capacities and uniform demands, then $\alpha(G) = 1/\Theta(n)$, while the dual LP has a solution of value $1/\Omega(n \log n)$ (by setting $s_v = 1/\Omega(n^2 \log n)$ for every $v \in V$).

Euclidean metrics. Even if the vertex-weighted distance function returned by the LP is equivalent to a Euclidean metric, up to a universal constant, there may still be an integrality gap of $\Omega\left(\sqrt{\frac{\log n}{\log \log n}}\right)$. We sketch the argument here. The idea is to take a fine enough "mesh" on a high-dimensional sphere so that the shortest-path distance along the mesh approximates the Euclidean distance. Using standard isoperimetric considerations on high-dimensional spheres, we are able to determine the structure of the near-optimal vertex separators. Here we will only sketch the proof; one may refer to [40] for a more detailed argument along these lines.

Let S^d be the d-dimensional sphere, let $\epsilon = 1/\Theta(d)$, and let V be an ϵ-net on the sphere S^d. (An ϵ-net in a metric space X is a subset $N \subseteq X$ such that $x, y \in N \implies d(x, y) \geq \epsilon$, and $X \subseteq \bigcup_{x \in N} B(x, \epsilon)$.) Standard arguments show that $n = |V| \leq O(d)^d$. Define a graph G with vertex set V and an edge between $u, v \in V$ whenever $\|u - v\|_2 \leq 10\,\epsilon$. We claim the following facts without proof (see [40] for a similar argument).

Claim 4.3. *The following three assertions holds true.*

1. $\alpha(G) = 1/\Theta(n\sqrt{d})$.
2. *Setting* $s_v = 1/\Theta(n^2 d)$ *in the dual LP yields a feasible solution with value* $1/\Theta(nd)$.
3. *The (vertex-weighted) shortest path metric on* G *with weights given by* $\{s_v\}_{v \in V}$ *is equivalent (up to a universal constant) to a Euclidean metric* (V, d). *(Namely the metric given by* $d(u, v) = \|u - v\|_2/n^2$, *recalling that* $V \subseteq S^d$.*)*

It follows that the integrality gap is at least $\Theta(\sqrt{d}) = \Theta\left(\sqrt{\frac{\log n}{\log \log n}}\right)$.

5 An integrality gap for the vector program

Consider the hypercube graph. Namely, the n vertices of the graph (where n is a power of 2) can be viewed as all vectors in $\{\pm 1\}^{\log n}$, and edges connect two vertices that differ in exactly one coordinate. Every vertex separator (A, B, S) has $\alpha(A, B, S) \geq 1/O(n\sqrt{\log n})$. This follows from standard vertex isoperimetry on the cube [27]. We show a solution to the vector program with value of $O(n/\log n)$, proving an integrality ratio of $\Omega(\sqrt{\log n})$ for the vector program, and implying that our rounding technique achieves the best possible approximation ratio (relative to the vector program), up to constant multiplicative factors.

In the solution to the vector program, we describe for every vertex i the associated vectors x_i and y_i. The vectors s_i will not be described explicitly, but are implicit, using the relation $s_i = v - x_i - y_i$. Note that the exclusion constraints $s_i \cdot x_i = s_i \cdot y_i = 0$ are implied by the exclusion constraints $x_i \cdot y_i = 0$ and the sphere constraints. Each vector will be described as a vector in $1 + n \log n + 2(n - 1)$ dimensions (even though n dimensions certainly suffice). Our redundant representation in terms of number of dimensions helps clarify the structure of the solution.

To describe the vector solution, we introduce two parameters, a and b. Their exact value will be determined later, and will turn out to be $a = 1/2 - \Theta(1/\log n)$, $b = \Theta(1/\sqrt{n \log n})$. We partition the coordinates into three groups of coordinates.

G1. Group 1, containing one coordinate. This coordinate corresponds to the direction of vector v (which has value 1 in this coordinate and 0 elsewhere). All x_i and y_i vectors have value a on this coordinate.

G2. Group 2, containing n identical blocks of $\log n$ coordinates. The coordinates within a block exactly correspond to the structure of the hypercube. Within a block, each x_i is a vector in $\{\pm b\}^{\log n}$ derived by scaling the hypercube label of vertex i (which is a vector in $\{\pm 1\}^{\log n}$) by a factor of b. Vector y_i is the negation of vector x_i on the coordinates of Group 2.

G3. Group 3, containing 2 identical blocks of $n-1$ coordinates. The coordinates within a block arrange all the x_i vectors as vertices of a simplex. This is done in the following way. Let H_n be the n by n Hadamard matrix with entries ± 1, obtained by taking the $(\log n)$-fold tensor product [16] of the 2 by 2 the matrix H_2 that has rows $(1,1)$ and $(1,-1)$. The inner product of any two rows of H_n is 0, the first column is all 1, and the sum of entries in any other column is 0. Remove the first column to obtain the matrix H'_n. Within a block, let vector x_i be the ith row of H'_n, scaled by a factor of b. Hence within a block, $x_i x_i = b^2(n-1)$, and $x_i x_j = -b^2$ for $i \neq j$. Vector y_i is identical to x_i on the coordinates of Group 3.

We now show that the triangle constraints are satisfied by our vector solution. Recall (see Section 2) that there is some flexibility in the choice of which triangle constraints to include in the vector program (and likewise for many other constraints that are valid for 0/1 solutions but are not used in our analysis). We shall address here a subset of the triangle constraints that is larger than that actually used in the analysis of our rounding algorithm.

There are five sets of vectors from which we can take the three vectors that participate in a triangle constraint: X (the x_i vectors), Y (the y_i vectors), S (the s_i vectors), v and 0. In our analysis we used only triangle constraints over vectors from X. Here we show that all the triangle constraints that involve only vectors from $X \bigcup Y$ are satisfied. All vectors in $X \bigcup Y$ have the identical value a in their first coordinate, and in every other coordinate they take only values from $\pm b$. Hence *every* quadratic constraint that holds for all ± 1 vectors (including, but not limited to, the triangle constraints) is satisfied on every coordinate separately, which implies that it is satisfied for all x_i and y_i vectors.

We let $K = \sum_{i,j \in V} (x_i - x_j)^2 = \Theta(n^3 b^2 \log n)$. The value of the parameters a and b is governed by the following three constraints.

1. The exclusion constraints imply that
$$a^2 - nb^2 \log n + 2b^2(n-1) = 0$$

2. The edge constraints (and the fact that edges connect vertices of Hamming distance 1) imply that
$$a^2 - nb^2(\log n - 2) - 2b^2 = 0$$

3. The sphere constraints imply that
$$a = a^2 + nb^2 \log n + 2b^2(n-1)$$

Hence we have a system of three equalities in two unknowns (a and b). This system is consistent, because the first two equalities are in fact identical (due to our careful choice of number of blocks in each group). They both give:

$$a^2 + (-n \log n + 2n - 2)b^2 = 0$$

By setting $b = a/\sqrt{n \log n - 2n + 2}$ the first two equalities are satisfied. The third equality now reads $a = a^2(2 + \epsilon)$ for some $\epsilon = \Theta(1/\log n)$. This equality is satisfied by taking a roughly equal to $1/2 - \epsilon/4$, which is $1/2 - \Theta(1/\log n)$.

It follows that in the vector solution all $s_i^2 = 1 - x_i^2 - y_i^2$ is $O(1/\log n)$ for every $i \in V$. Hence our vector solution has value

$$\frac{1}{K}\sum_{i\in V} s_i^2 = \frac{1}{\Theta(n \log n)}.$$

Finally we note that rather than having only one coordinate in Group 1, we can have $(a/b)^2 = n\log n - 2n + 2$ coordinates, and give the x and y vectors values b in these coordinates. Then all x and y vectors become vertices of a $2n\log n$-dimensional hypercube (of side length b). We see that even in this special case, the integrality gap remains $\Omega(\sqrt{\log n})$.

6 Balanced separators and applications

6.1 Reduction from min-ratio cuts to balanced separators

In this section, we sketch a pseudo-approximation for finding balanced vertex separators in a graph $G = (V, E)$. Let $W \subseteq V$ be an arbitrary subset of V. For $\delta \in (0, 1)$, we say that a subset $X \subseteq V$ is a *δ-vertex-separator (with respect to W)* if every connected component C of $G[V \setminus X]$ has $|C \cap W| \leq \delta|W|$. Our goal in this section is to show that we can find a $\frac{3}{4}$-vertex-separator $X \subseteq V$ whose size is within an $O(\beta)$ factor of the optimal $\frac{2}{3}$-vertex-separator of G, whenever we can find approximate min-ratio cuts in G within factor β. This technique is standard (see [36]).

The algorithm. Let $m = |W|$, and for any subset $U \subseteq V$, define $|U|_W = |U \cap W|$. Let $\pi_1(v) = 1$ for every $v \in V$, and $\pi_2(v) = 1$ if $v \in W$ and $\pi_2(v) = 0$ otherwise. These are the weights for the numerator and denominator, respectively, i.e. we assume that we have a β-approximation for $\alpha_{\pi_1,\pi_2}(\cdot)$. We maintain a vertex separator $S \subseteq V$. Initially, $S = \emptyset$. As long as there exists some connected component $U \subseteq V$ in $G[V \setminus S]$ with $|U|_W \geq \frac{3}{4}|W|$, we use our β-approximation to find a minimum-ratio vertex cut S' in $G[U]$ which is within β of optimal. We then set $S \leftarrow S \cup S'$ and continue.

The analysis. Let S be the final vertex separator. By construction, it is a $\frac{3}{4}$-vertex separator since every connected component U of $G[V \setminus S]$ has $|U|_W < \frac{3}{4}|W|$. Let $T \subseteq V$ be an optimal $\frac{2}{3}$-vertex separator.

Claim 6.1. $|S| \leq O(\beta)|T|$.

Proof. The fact that T is a $\frac{2}{3}$-vertex separator with respect to W implies that the vertices in $V \setminus T$ can be partitioned into two disjoint sets $A_T, B_T \subseteq V$ such that $|A_T \cup T|_W, |B_T \cup T|_W \geq \frac{1}{3}|W|$, and with no edges between A_T and B_T. Suppose we are at a step where $|U|_W \geq \frac{3}{4}|W|$. Let (A', B', S') be the vertex separator in $G[U]$ that we find by running our min-quotient cut algorithm with ratio β, and suppose that $|A'|_W \geq |B'|_W$. We know that

$$\frac{|S'|}{|A' \cup S'|_W \cdot |B' \cup S'|_W} \leq \beta \frac{|T|}{|(A_T \cup T) \cap U|_W \cdot |(B_T \cup T) \cap U|_W} \leq \frac{18\beta|T|}{m^2},$$

where the final inequality follows because $|U|_W \geq \frac{3m}{4}$. It follows that

$$|S'| \leq \frac{18\beta|T|(|B'|_W + |S'|_W)}{m}.$$

To see that $|S| \leq O(\beta)|T|$, it suffices to see that when we sum $|B'|_W + |S'|_W$ over all iterations, the value is at most $O(m)$. But since we throw away the vertices of $B' \cup S'$ in every iteration (and recurse only on A'), the sum is clearly at most m. □

6.2 Getting an $O(\sqrt{\log \mathsf{opt}})$ approximation for vertex separators

In this section, we sketch a proof of how one can obtain an $O(\sqrt{\log \mathsf{opt}})$ pseudo-approximation for finding balanced vertex separators. In other words, given a graph G with a $\frac{2}{3}$-vertex-separator of size m, we find a $\frac{3}{4}$-vertex-separator whose size is at most $(m\sqrt{\log m})$. The method is based on the following enhancement of Theorem 3.3.

Theorem 6.2. *Let $C > 0$ be a universal constant. Let (X, d) be an n-point metric space of negative type, and let $\omega_0 : X \times X \to \mathbb{R}_+$ be any product weight. If*

$$\frac{\sum_{x,y} \omega_0(x,y)\, d(x,y)}{\sum_{x,y} \omega_0(x,y)} = 1,$$

and there exists an ε-net $N \subseteq X$ with $|N| \le m$ and $\varepsilon \le 1/(C\sqrt{\log m})$, then there exists an efficiently computable map $f : X \to \mathbb{R}$ with $avd_{\omega_0}(f) = O(\sqrt{\log m})$.

Proof. Assume that $\omega_0(x,y) = \pi(x)\pi(y)$ for all $x, y \in X$. As in the proof of Theorem 3.3 (see sketch in Section A.2), if there exists some point $x_0 \in X$ for which $\pi(B(x_0, \frac{1}{4n^2})) \ge \frac{1}{2}\pi(X)$, then we achieve a map $f : X \to \mathbb{R}$ with $avd_{\omega_0}(f) = O(1)$. If no such x_0 exists, then it must be the case (see the proof of [7, Lemma 14]) that there exists a set $S \subseteq X \times X$ of pairs for which $\sum_{(x,y)\in S} \pi(x)\pi(y) \ge \Omega(1) \sum_{x,y} \omega_0(x,y)$, and $d(x,y) \ge \frac{1}{100}$ for $(x,y) \in S$.

We construct a new weight function $\pi^* : N \to \mathbb{R}_+$ on N as follows. Since N is an ε-net, we have $X \subseteq \bigcup_{y\in N} B(y, \varepsilon)$. For every point $x \in X$, put x into a set S_y for some net point $y \in N$ with $d(x,y) \le \varepsilon$ (so that $\{S_y\}_{y\in N}$ is a partition of X). Define $\pi^*(y) = \sum_{x\in S_y} \pi(x)$ for every $y \in N$.

We now consider the quantity

$$\bar{d}_N = \frac{\sum_{x,y\in N} \pi^*(x)\pi^*(y)\, d(x,y)}{\sum_{x,y\in N} \pi^*(x)\pi^*(y)} = \frac{\sum_{x,y\in N} \sum_{u\in S_x, v\in S_y} \pi(u)\pi(v)\, d(x,y)}{\sum_{x,y} \omega_0(x,y)}.$$

We claim that $\bar{d}_N = \Omega(1)$. But this follows since

$$\begin{aligned}
\sum_{x,y\in N} \sum_{u\in S_x, v\in S_y} \pi(u)\pi(v)\, d(x,y) &\ge \sum_{x,y\in N} \sum_{u\in S_x, v\in S_y, (u,v)\in S} \pi(u)\pi(v)\, d(x,y) \\
&\ge \sum_{x,y\in N} \sum_{u\in S_x, v\in S_y, (u,v)\in S} \pi(u)\pi(v)\ (d(u,v) - 2\varepsilon) \\
&\ge \frac{1}{2} \sum_{(u,v)\in S} \pi(u)\pi(v)\, d(u,v) = \Omega(1) \sum_{x,y} \omega_0(x,y).
\end{aligned}$$

As discussed in Section A.2, the techniques of [7] now show that there exist two subsets $L, R \subseteq N$ for which $d(L,R) \ge 1/O(\sqrt{\log m})$ and $\pi^*(L), \pi^*(R) \ge \frac{1}{10}\pi^*(X)$. Construct sets

$$L' = \{x \in X : x \in S_y \text{ for some } y \in L\} \quad \text{and} \quad R' = \{x \in X : x \in S_y \text{ for some } y \in R\}.$$

Note that $\pi(L') = \pi^*(L)$ and $\pi(R') = \pi^*(R)$, hence $\pi(L'), \pi(R') \ge \frac{1}{10}\pi(X)$. Finally, for any points $x_L \in L'$, $x_R \in R'$, let y_L, y_R be such that $x_L \in S_{y_L}$ and $x_R \in S_{y_R}$, and notice that

$$d(x_L, x_R) \ge d(y_L, y_R) - d(x_L, y_L) - d(x_R, y_R) \ge \tfrac{1}{O(\sqrt{\log m})} - 2\varepsilon \ge \tfrac{1}{O(\sqrt{\log m})},$$

where the last inequality holds for $C > 0$ chosen sufficiently large (and hence ε chosen sufficiently small). Now one simply takes the map $f(x) = d(x, L')$, which has $avd_{\omega_0}(f) = O(\sqrt{\log m})$. □

Next, we make an observation about solutions to the SDP of Section 2.3.

Lemma 6.3. *If $\{x_i, y_i\}$ is a solution to the SDP with $W = \sum_{i \in V}(1 - x_i^2 - y_i^2)$, then in the metric space $(\{x_1, \ldots, x_n\}, d)$ where $d(i,j) = (x_i - x_j)^2$, there exists an ε-net $N \subseteq \{x_1, \ldots, x_n\}$ with $|N| \leq O(W/\varepsilon)$.*

Proof. For each $i \in V$, define $w(i) = 1 - x_i^2 - y_i^2$. For a subset $S \subseteq V$, define $w(S) = \sum_{x \in S} w(x)$. Let $G = (V, E)$ be the original graph, and let $d_G(i,j) = \min_{p \in \mathcal{P}_{ij}} w(p)$, where we recall that $\mathcal{P}_{ij}$ is the set of all simple i-j paths. We claim first that $d(i,j) \leq 4\, d_G(i,j)$. Indeed, let $i = i_1, i_2, \ldots, i_k = j$ be a minimum-weight path in G, then

$$\begin{aligned} d(i,j) = (x_i - x_j)^2 &\leq \sum_{h=1}^{k-1} (x_{i_h} - x_{i_{h+1}})^2 && (4) \\ &\leq 2 \sum_{h=1}^{k-1} \Big((1 - x_{i_h}^2 - y_{i_h}^2) + (1 - x_{i_{h+1}}^2 - y_{i_{h+1}}^2) \Big) && (5) \\ &= 2 \sum_{h=1}^{k-1} (w(i_h) + w(i_{h+1})) \\ &\leq 4\, d_G(i,j), \end{aligned}$$

where (4) follows from the squared triangle inequalities, and (5) follows from line (2) in Proposition 3.10.

Thus it will suffice to find an $\varepsilon/4$-net N in the metric d_G, and the rest of the proof refers to this metric on $X = \{x_1, \ldots, x_n\}$. Choose a maximal set $Y \subseteq \{x_1, \ldots, x_n\}$ among all points $x \in X$ for which $w(B_{d_G}(x, \varepsilon/8)) \geq \varepsilon/16$, subject to the constraint that $x, y \in Y \implies d(x,y) > \varepsilon/8$. By construction, the balls $B_{d_G}(x, \varepsilon/8)$ are disjoint for $x \in Y$, hence $|Y| \leq 16W/\varepsilon$, recalling that $W = w(X)$.

So we are done once we prove that Y is an $\varepsilon/4$-net in (X, d_G). For the sake of contradiction, suppose there is a point $x \in X$ with $d_G(x, Y) > \varepsilon/4$. Let $y \in Y$ be such that $d_G(x,y) = d_G(x, Y)$, and and consider any shortest-path $x = y_1, \ldots, y_k = y$ in G. Letting $P = \{y_1, \ldots, y_k\}$, we know that $w(P) = d_G(x,y) > \varepsilon/4$. If we set $P' = \{u \in P : d_G(y,u) > \varepsilon/8\}$, then $w(P') > w(P) - \varepsilon/8 \geq \varepsilon/8$ and for every $u \in P'$, we have $d_G(u, Y) > \varepsilon/8$. So if there exists any point $u \in P'$ with $w(u) \geq \varepsilon/16$ then we could add u to Y, contradicting its maximality. Thus we may assume that for every $u \in P'$, we have $w(u) < \varepsilon/16$. But now let $z \in P'$ be the point of P' which is closest to y. Then $d_G(z,x) = w(P') > \varepsilon/8$, hence we know that

$$w(B_{d_G}(z, \varepsilon/8)) \geq w(B_{d_G}(z, \varepsilon/8) \cap P') \geq \varepsilon/16,$$

because the first point along P' not included in $B_{d_G}(z, \varepsilon/8)$ (which must exist) must be further than $\varepsilon/8$ away from z, but also have weight at most $\varepsilon/16$. We again conclude that Y is not maximal, completing the proof. □

Combining Theorem 6.2 and Lemma 6.3, along with the analysis of Section 3 yields an $O(\sqrt{\log m})$-approximation to vertex sparsest cut where m is the number of vertices in an optimal $\frac{2}{3}$-vertex-separator. Now applying the transformation of Section 6.1 yields the desired $O(\sqrt{\log \mathsf{opt}})$ pseudo-approximation for finding balanced vertex separators.

6.3 Applications

The notion of treewidth was introduced by Robertson and Seymour [43] and plays an important role in their fundamental work on graph minors. In addition, it has numerous practical applications (see

e.g. [10]). A large amount of effort has been put into determining treewidth, which is NP-complete even when the input graph is severely restricted (see the discussion in [21] for a brief history).

From the approximation viewpoint, Bodlaender et al. [11] gave an $O(\log n)$-approximation algorithm for treewidth on general graphs. Amir [4] improved the approximation factor to $O(\log \mathsf{opt})$ where opt is the actual treewidth of the graph. Constant-factor approximations for treewidth were obtained on AT-free graphs [13, 12] and on planar graphs [44]. The approximation for planar graphs is a consequence of the polynomial-time algorithm given by [44] for computing the parameter *branchwidth*, whose value approximates treewidth within a factor of 1.5. Recently, [5] obtained a new approximation algorithm for treewidth in planar graphs with a constant factor slightly worse than 1.5, and the authors of [21] derived a polynomial-time algorithms for approximating treewidth within a factor of 1.5 for single-crossing-minor-free graphs, generalizations of planar graphs. A well-known open problem is whether treewidth can be approximated within a constant factor.

Using our new approximation algorithms for vertex separators, we improve the approximation ratio for treewidth, both in general graphs and in some special families of graphs. Our improvements and some of their implications will be presented after we formally define the notion of treewidth.

Treewidth. The notion of *treewidth* involves a representation of a graph as a tree, called a tree decomposition. More precisely, a *tree decomposition* of a graph $G = (V, E)$ is a pair (T, χ) in which $T = (I, F)$ is a tree and $\chi = \{\chi_i \mid i \in I\}$ is a family of subsets of $V(G)$ such that (1) $\bigcup_{i \in I} \chi_i = V$; (2) for each edge $e = \{u, v\} \in E$, there exists an $i \in I$ such that both u and v belong to χ_i; and (3) for all $v \in V$, the set of nodes $\{i \in I \mid v \in \chi_i\}$ forms a connected subtree of T. To distinguish between vertices of the original graph G and vertices of T in the tree decomposition, we call vertices of T *nodes* and their corresponding χ_i's *bags*. The maximum size of a bag in χ minus one is called the *width* of the tree decomposition. The *treewidth* of a graph G, which we denote by $\mathrm{tw}(G)$, is the minimum width over all possible tree decompositions of G. A tree decomposition is called a *path decomposition* if $T = (I, F)$ is a path. The *pathwidth* of a graph G is the minimum width over all possible path decompositions of G.

Now we are ready to state our approximation result for treewidth.

Theorem 6.4. *There exist polynomial time algorithms that find a tree decomposition of width at most $O(\sqrt{\log \mathrm{tw}(G)}\,\mathrm{tw}(G))$ for a general graph G and at most $O(|V(H)|^2\,\mathrm{tw}(G))$ for an H-minor-free graph G.*

Proof. The proof follows by plugging our improved approximation ratios for balanced vertex separators into the known approximation algorithms for treewidth. Specifically, the algorithm of [11] finds a tree decomposition by recursively using a balanced vertex separator algorithm. The vertex separator algorithm is applied to subgraphs of the original graph, in a product demand setting. It turns out that the approximation ratio obtained for treewidth is at most a constant factor worse than that of the underlying vertex separator algorithm. Using our bounds from Section 6.2 one obtains the first part of Theorem 6.4, and using Theorem 4.2 one obtains the second part of Theorem 6.4. □

Improving the approximation factor of treewidth improves the approximation factor for several other problems. We refer to [11] for a discussion of these implications and the relevant definitions.

Corollary 6.5. *There exist $O(\sqrt{\log \mathsf{opt}})$ (resp., $O(|V(H)|^2)$) approximation algorithms for branchwidth, minimum front size and minimum size of a clique in a chordal supergraph of a general (resp., H-minor-free) graph G. Additionally, there are $O(\sqrt{\log \mathsf{opt} \log n})$ (resp., $O(|V(H)|^2 \log n)$) approximation algorithms for pathwidth, minimum height elimination order tree, and search number in a general (resp., H-minor-free) graph G.*

We also note that Theorem 3.12 with general weights π_1, π_2 is useful for certain hypergraph partitioning problems [36]. Improving the approximation factor for treewidth has a direct improvement on the running time of approximation schemes and subexponential fixed parameter algorithms for several NP-hard problems on graphs families which exclude a fixed minor. In such algorithms finding the tree decomposition of almost minimum width, on which we can run dynamic programming, plays a very important role. More precisely, Demaine and Hajiaghayi [20, 19] introduced the concept of (contraction/minor) bidimensional parameters for planar graphs and more generally for excluded-minor families. Examples of bidimensional parameters include number of vertices, diameter, and the size of various structures, e.g., feedback vertex set, vertex cover, minimum maximal matching, face cover, a series of vertex-removal parameters, dominating set, edge dominating set, r-dominating set, connected dominating set, connected edge dominating set, connected r-dominating set, and unweighted TSP tour (a walk in the graph visiting all vertices).

They show how one can obtain PTASs for almost all bidimensional parameters on planar graphs, single-crossing-minor-free graphs and bounded genus graphs. In fact, as they mentioned their approach can be extended to work on apex-minor-free graphs for contraction-bidimensional parameters and on H-minor-free graphs, where H is a fixed graph, for minor-bidimensional parameters (see [20, 19] for appropriate definitions). However currently they obtained quasi-polynomial-time approximation schemes for these general settings. The only barrier to obtain PTASs for these general settings is obtaining a constant-factor polynomial-time approximation algorithm for treewidth of an H-minor-free graph for a fixed H (this is posed as an open problem in [20]). Using Theorem 6.4, we overcome this barrier and obtain PTASs for contraction-bidimensional parameters in apex-minor-free graphs and for minor-bidimensional parameters in H-minor-free graphs for a fixed H. As an immediate consequence, we obtain the following theorem (see [20, 19] for the exact definitions of the problems mentioned below).

Theorem 6.6. *There are PTASs for feedback vertex set, vertex cover, minimum maximal matching, and a series of vertex-removal problems in H-minor-free graphs for a fixed H. Also, there are PTASs for dominating set, edge dominating set, r-dominating set, connected dominating set, connected edge dominating set, connected r-dominating set, and clique-transversal set in apex-minor-free graphs.*

Among the problems mentioned above, PTASs for vertex cover and dominating set (but not its other variants) using a different approach were known before (see e.g. [26]).

Acknowledgements. We would like to thank the anonymous reviewers for their very useful comments on an earlier version of this manuscript. The second author would like to thank Erik D. Demaine and MohammadAli Safari for helpful comments and discussions.

References

[1] A. Agarwal, M. Charikar, K. Makarychev, and Y. Makarychev, *$O(\sqrt{\log n})$ approximation algorithms for Min UnCut, Min 2CNF Deletion, and directed cut problems*, in Proceedings of the 37th Annual ACM Symposium on Theory of Computing, 2005, pp. 573–581.

[2] N. Alon, P. Seymour, and R. Thomas, *A separator theorem for for graphs with excluded minor and its applications*, in Proceedings of the 22nd Annual ACM Symposium on Theory of Computing (Baltimore, MD, 1990), 1990, pp. 293–299.

[3] ——, *A separator theorem for nonplanar graphs*, J. Amer. Math. Soc., 3 (1990), pp. 801–808.

[4] E. Amir, *Efficient approximation for triangulation of minimum treewidth*, in Proceedings of the 17th Annual Conference on Uncertainty in Artificial Intelligence, Morgan Kaufmann, 2001, pp. 7–15. Journal version titled "Approximation Algorithms for Treewidth" is available at the author's homepage.

[5] E. Amir, R. Krauthgamer, and S. Rao, *Constant factor approximation of vertex-cuts in planar graphs*, in Proceedings of the thirty-fifth annual ACM symposium on Theory of computing, ACM Press, 2003, pp. 90–99.

[6] S. Arora, J. R. Lee, and A. Naor, *Euclidean distortion and the Sparsest Cut*, in 37th Annual Symposium on the Theory of Computing, 2005, pp. 553–562. To appear, *J. Amer. Math. Soc.*

[7] S. Arora, S. Rao, and U. Vazirani, *Expander flows, geometric embeddings, and graph partitionings*, in 36th Annual Symposium on the Theory of Computing, 2004, pp. 222–231.

[8] Y. Aumann and Y. Rabani, *An $O(\log k)$ approximate min-cut max-flow theorem and approximation algorithm*, SIAM J. Comput., 27 (1998), pp. 291–301.

[9] S. N. Bhatt and F. T. Leighton, *A framework for solving VLSI graph layout problems*, J. Comput. System Sci., 28 (1984), pp. 300–343.

[10] H. L. Bodlaender, *A partial k-arboretum of graphs with bounded treewidth*, Theoret. Comput. Sci., 209 (1998), pp. 1–45.

[11] H. L. Bodlaender, J. R. Gilbert, H. Hafsteinsson, and T. Kloks, *Approximating treewidth, pathwidth, frontsize, and shortest elimination tree*, J. Algorithms, 18 (1995), pp. 238–255.

[12] V. Bouchitté, D. Kratsch, H. Müller, and I. Todinca, *On treewidth approximations*, in Proceedings of the 1st Cologne-Twente Workshop on Graphs and Combinatorial Optimization, vol. 8 of Electronic Notes in Discrete Mathematics, 2001.

[13] V. Bouchitté and I. Todinca, *Treewidth and minimum fill-in: grouping the minimal separators*, SIAM J. Comput., 31 (2001), pp. 212–232.

[14] J. Bourgain, *On Lipschitz embedding of finite metric spaces in Hilbert space*, Israel J. Math., 52 (1985), pp. 46–52.

[15] T. N. Bui and C. Jones, *Finding good approximate vertex and edge partitions is NP-hard*, Inf. Process. Lett., 42 (1992), pp. 153–159.

[16] N. Burbaki, *Elements of Matematics. Algebra.*, Springer, New York, 2006.

[17] S. Chawla, A. Gupta, and H. Raecke, *Embeddings of negative-type metrics and improved approximations to sparsest cut*, in Proceedings of the 16th Annual ACM-SIAM Symposium on Discrete Algorithms, Vancouver, Canada, January 2005, pp. 102–111.

[18] C. Chekuri, S. Khanna, and B. Shepherd, *Multicommodity flow, well-linked terminals, and routing problems*, in Proceedings of the 37th Annual ACM Symposium on Theory of Computing, ACM, 2005, pp. 183–192.

[19] E. D. Demaine and M. Hajiaghayi, *Linearity of grid minors in treewidth with applications through bidimensionality*, Combinatorica. To appear. A preliminary version appeared in Proceedings of the 16th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA 2005).

[20] ——, *Bidimensionality: New connections between FPT algorithms and PTASs*, in Proceedings of the 16th Annual ACM-SIAM Symposium on Discrete Algorithms, Vancouver, Canada, January 2005, pp. 590–601.

[21] E. D. Demaine, M. Hajiaghayi, N. Nishimura, P. Ragde, and D. M. Thilikos, *Approximation algorithms for classes of graphs excluding single-crossing graphs as minors*, Journal of Computer and System Sciences, 69 (2004), pp. 166–195.

[22] G. Even, J. Naor, B. Schieber, and M. Sudan, *Approximating minimum feedback sets and multicuts in directed graphs*, Algorithmica, 20 (1998), pp. 151–174.

[23] U. Feige, *Relations between average case complexity and approximation complexity*, in Proceedings of the thiry-fourth annual ACM symposium on Theory of computing, ACM Press, 2002, pp. 534–543.

[24] U. Feige and S. Kogan, *Hardness of approximation of the balanced complete bipartite subgraph problem*, Technical report MCS04-04, Department of Computer Science and Applied Math., The Weizmann Institute of Science, (2004).

[25] J. R. Gilbert, J. P. Hutchinson, and R. E. Tarjan, *A separator theorem for graphs of bounded genus*, J. Algorithms, 5 (1984), pp. 391–407.

[26] M. Grohe, *Local tree-width, excluded minors, and approximation algorithms*, Combinatorica, 23 (2003), pp. 613–632.

[27] L. H. Harper, *Optimal numberings and isoperimetric problems on graphs*, J. Combinatorial Theory, 1 (1966), pp. 385–393.

[28] J. A. Kelner, *Spectral partitioning, eigenvalue bounds, and circle packings for graphs of bounded genus*, in Proceedings of the thirty-sixth annual ACM symposium on Theory of computing, ACM Press, 2004, pp. 455–464.

[29] S. Khot, *Ruling out PTAS for graph min-bisection, densest subgraph and bipartite clique*, in 45th Annual Symposium on Foundations of Computer Science, 2004, pp. 136–145.

[30] S. Khot and N. Vishnoi, *The unique games conjecture, integrality gap for cut problems and embeddability of negative type metrics into L_1*, in 46th Annual Symposium on Foundations of Computer Science, 2005, pp. 53–62.

[31] P. N. Klein, S. A. Plotkin, and S. Rao, *Excluded minors, network decomposition, and multicommodity flow*, in Proceedings of the 25th Annual ACM Symposium on Theory of Computing, 1993, pp. 682–690.

[32] R. Krauthgamer, J. R. Lee, M. Mendel, and A. Naor, *Measured descent: a new embedding method for finite metrics*, Geom. Funct. Anal., 15 (2005), pp. 839–858.

[33] E. L. Lawler, *Combinatorial optimization: networks and matroids*, Holt, Rinehart and Winston, New York, 1976.

[34] J. R. Lee, *On distance scales, embeddings, and efficient relaxations of the cut cone*, in Proceedings of the 16th Annual ACM-SIAM Symposium on Discrete Algorithms, Vancouver, 2005, pp. 92–101.

[35] F. T. Leighton, *Complexity Issues in VLSI: Optimal Layout for the Shuffle-Exchange Graph and Other Networks*, MIT Press, Cambridge, MA, 1983.

[36] T. Leighton and S. Rao, *Multicommodity max-flow min-cut theorems and their use in designing approximation algorithms*, J. ACM, 46 (1999), pp. 787–832.

[37] C. Leiserson, *Area-efficient graph layouts (for VLSI)*, in 21th Annual Symposium on Foundations of Computer Science, IEEE Computer Soc., Los Alamitos, CA, 1980, pp. 270–280.

[38] N. Linial, E. London, and Y. Rabinovich, *The geometry of graphs and some of its algorithmic applications*, Combinatorica, 15 (1995), pp. 215–245.

[39] R. J. Lipton and R. E. Tarjan, *Applications of a planar separator theorem*, SIAM J. Comput., 9 (1980), pp. 615–627.

[40] J. Matoušek and Y. Rabinovich, *On dominated l_1 metrics*, Israel J. Math., 123 (2001), pp. 285–301.

[41] Y. Rabinovich, *On average distortion of embedding metrics into the line and into L_1*, in 35th Annual ACM Symposium on Theory of Computing, 2003, pp. 456–462.

[42] S. Rao, *Small distortion and volume preserving embeddings for planar and Euclidean metrics*, in Proceedings of the 15th Annual Symposium on Computational Geometry, New York, 1999, ACM, pp. 300–306.

[43] N. Robertson and P. D. Seymour, *Graph minors. II. Algorithmic aspects of tree-width*, J. Algorithms, 7 (1986), pp. 309–322.

[44] P. D. Seymour and R. Thomas, *Call routing and the ratcatcher*, Combinatorica, 14 (1994), pp. 217–241.

[45] D. B. West, *Introduction to Graph Theory*, Prentice Hall Inc., Upper Saddle River, NJ, 1996.

A Appendix

A.1 A note about approximating vertex expansion

In the case of *edge cuts*, the value of the sparsest cut (under uniform weights) corresponds to *edge expansion* of the graph G. Thus it is perhaps more natural to consider finding the vertex separator (A, B, S) which minimizes the ratio $|S|/|B|$, where by convention, $|B| \leq |A|$.

We now show that having the $|S|$ term in the denominator, i.e. $|S|/(|B| + |S|)$, is crucial to obtaining polylogarithmic approximation ratios. We present here an argument (essentially due to Shimon Kogan) that demonstrates this fact.

Consider the problem of balanced bipartite independent set (BBIS). The input is a bipartite graph $G(U \bigcup V, E)$ with $|U| = |V| = n$, and the goal is to find the maximum value of t and sets $A \subset U, B \subset V$ with $|A| = |B| = t$ with no edges between A and B. It is known that when t is small compared to n, approximating this problem (the value of t) within a ratio of n^δ for some $\delta > 0$ will have some major algorithmic consequences [23, 24], including subexponential algorithms for all NP problems [29]. Now modify G by making U into a clique and V into a clique, obtaining a graph G'. The set S of vertices not in the maximum BBIS provides a vertex separator (A, B, S) for G'. The ratio $|S|/|B|$ for this separator is the minimum possible up to constant factors. (For every separator (A', B', S'), side U cannot contain vertices both from A and from B. Hence $|S'| = \Omega(n)$ unless both A' and B' are of size nearly n. When t is known to be small, this implies that $|S'| = \Theta(n)$ for all separators. Hence the ratio $|S'|/|B'|$ of any separator in G' is governed by $|B'|$ rather than by $|S|$. The value of $|B'|$ is maximized by taking the separator (A, B, S).) This implies that for the minimum balanced vertex separator the quantity $|S|/|B|$ cannot be approximated within a ratio of n^δ (unless NP has subexponential algorithms).

Remark. For a set B of vertices, let $N(B)$ denote the set of vertices not in B that are neighbors of vertices in B. Then the expansion of B is $|N(B)|/|B|$. The expansion of a graph is the minimum over all sets B up to a certain size of the ratio $|N(B)|/|B|$. The restriction on the size of B is necessary so as to avoid B being the whole graph, giving expansion 0. For bounded degree graph, one typically requires $|B| \leq n/2$. For graphs of unbounded degree, such a requirement is insufficient, as it always bounds the expansion by 1 (taking B to be half the graph), whereas one would like to allow for much higher expansions. A possible restriction on B in this case is to require it to be the smaller side of an (A, B, S) vertex separator. Under this definition of vertex expansion, the above argument shows that vertex expansion cannot be approximated within a factor of n^δ unless NP has subexponential algorithms.

A.2 Line embedding theorems

We now sketch how the following three theorems follow from their respective sources. We begin with Bourgain's theorem.

Theorem A.1 (Bourgain, [14])**.** *If (X, d) is an n-point metric space, then for every weight function $\omega : X \times X \to \mathbb{R}_+$, there exists an efficiently computable map $f : X \to \mathbb{R}$ with $avd_w(f) = O(\log n)$.*

In [14], it is shown that every n-point metric (X, d) space embeds into a Hilbert space with distortion $O(\log n)$, but Bourgain actually shows something stronger. He proves that there exists a probability space (Ω, μ) on random subsets $A_\tau \subseteq X$, $\tau \in \Omega$, satisfying the following property. For every $x, y \in X$,

$$\mathbb{E}_\Omega \left[|d(x, A_\tau) - d(y, A_\tau)|\right] \geq \frac{d(x, y)}{O(\log n)}.$$

To show how this implies the theorem, note that by linearity of expectation

$$\mathbb{E}_\Omega \left[\sum_{x,y \in X} \omega(x,y) \cdot |d(x,A_\tau) - d(y,A_\tau)| \right] \geq \frac{1}{O(\log n)} \sum_{x,y \in X} \omega(x,y) \cdot d(x,y).$$

Hence there must exist some subset $A_\tau \subseteq X$ for which the map $f : X \to \mathbb{R}$ given by $f(x) = d(x, A_\tau)$ has $avd_\omega(f) = O(\log n)$. An efficient randomized algorithm for sampling A_τ is given in [38].

Theorem A.2 (Rabinovich, [41]). *If (X,d) is any metric space supported on a graph which excludes a K_r-minor, then for every* product weight $\omega_0 : X \times X \to \mathbb{R}_+$*, there exists an efficiently computable map $f : X \to \mathbb{R}$ with $avd_{\omega_0}(f) = O(r^2)$.*

In [41], Rabinovich proves precisely this fact, although only for the uniform weight function $\omega_0(x,y) = 1$ for all $x, y \in X$. It is easy to see that we can assume arbitrary product form for ω_0 without loss of generality. Suppose that we have vertex weights $\pi : V \to \mathbb{R}_+$. We can replace X by the pseudo-metric where each copy of $x \in X$ occurs $\pi(x)$ times. Then applying the analysis of [41] immediately yields the desired result.

Theorem A.3 (Arora, Rao, Vazirani, [7]). *If (X,d) is an n-point metric of negative type, then for every* product weight $\omega_0 : X \times X \to \mathbb{R}_+$*, there exists an efficiently computable map $f : X \to \mathbb{R}$ with $avd_{\omega_0}(f) = O(\sqrt{\log n})$.*

Assume that $\omega_0(x,y) = \pi(x)\pi(y)$ for all $x, y \in X$. We will "mentally" replace every copy of x by $\pi(x)$ copies, but we will ensure that this increase in the number of points doesn't affect the quality of our map f. Also, suppose that (by scaling) $\left(\frac{1}{\sum_{x,y} \omega_0(x,y)}\right) \sum_{x,y \in X} \omega_0(x,y) \cdot d(x,y) = 1$.

Suppose there exists some point $x_0 \in X$ for which $\pi(B(x_0, \frac{1}{4})) \geq \frac{1}{2}\pi(X)$. In this case, the map $f(x) = d\left(x, B(x_0, \frac{1}{4})\right)$ has $avd_{\omega_0}(f) = O(1)$ (see, e.g., [7, Lemma 14]).

Otherwise, the techniques of [7] show that there exist two subsets $L, R \subseteq X$ for which $d(L,R) \geq 1/O(\sqrt{\log n})$ and $\pi(L), \pi(R) \geq \frac{1}{10}\pi(X)$. The fact that the number of copies of a point $x \in X$ doesn't affect the analysis is somewhat technical, and relies on the fact that an "(ϵ, δ)-cover" has size which is lower-bounded by the number of *distinct* points that it contains. In this latter case, one simply takes the map $f(x) = d(x, L)$, which has $avd_{\omega_0}(f) = O(\sqrt{\log n})$. A simpler algorithm for computing the map f (which consists of choosing a few random hyperplanes) is given in [34].

A.3 Approximating the "densest subgraph"

To orient the reader arriving at this section from Section 3.6, let us remark that π and ω below can correspond to π_1 and a product distribution $\pi_2 \times \pi_2$ in Section 3.6.

Given a set $V = \{v_1, \ldots, v_n\}$ with a positive rational weight function π on V, and a nonnegative rational weight function ω on $V \times V$, we need to find a set $S \subseteq V$ of maximum density, where the density of a set is defined as

$$\Delta(S) \equiv \frac{\sum_{i,j \in S} \omega(i,j)}{\pi(S)}. \tag{6}$$

This is a weighted version of the *densest subgraph* problem, and can be solved in polynomial time (see for example, Chapter 4 in [33]). For completeness, we sketch the algorithm.

Construct a bipartite graph with sides U and W, where U has n vertices labeled $\{u_1, \ldots u_n\}$, and W has n^2 vertices labeled w_{ij} for $1 \leq i, j \leq n$. For every i, connect vertex u_i to the vertices w_{ij} and w_{ji} (for all j). All these edges have infinite capacity. Add two special vertices, s and t to the graph. For every i, connect vertex u_i to s by an edge of capacity $k\pi(i)$, where k is a parameter

whose value will be optimized later. For every $1 \leq i, j \leq n$, connect vertex $w_{i,j}$ to t by an edge of capacity $\omega(i, j)$. Now compute the minimum capacity (s, t)-cut in the resulting capacitated graph (a problem that can be solved in polynomial time by using flow techniques).

We now analyze the above algorithm. Observe first that the minimum (s, t)-cut contains only edges that are connected to either t or s, as other edges have infinite capacity. Furthermore, observe that if the parameter k is sufficiently large, then the minimum (s, t)-cut contains exactly those edges connected to t. (Here we used our assumption that $\pi(i) > 0$ for all i, but we remark that this assumption can be made without loss of generality, because all v_i with $\pi(i) = 0$ can be placed in S.) How low should k be so that the cut also cuts edges connected to s? This may happen only when $k \leq \Delta$ (and will necessarily happen when $k < \Delta$), where $\Delta = \min_S \Delta(S)$. The reason is the following. Cutting a set $S \subset U$ from s costs $k\pi(S)$. This needs to be offset by a gain on the t side, resulting from the fact that edges between t and vertices of W labeled by $S \times S$ no longer need to be cut. This gives a saving of $\sum_{i,j \in S} \omega(i, j)$. The saving equals the cost precisely when $k = \Delta$.

Using the above analysis, it follows that by performing a search over the parameter k, one can find the value of Δ and the densest set S achieving this value.