Concept and Existence of Good Vertex Sparsifiers

Radu Vintan

Technische Universität München

July 26, 2021

1 Introduction

Consider an undirected, capacitated graph $G := (V, E, c)$, with $n := |V|$ nodes, $m := |E|$ edges, capacities $c : E \to \mathbb{R}_{\geq 0}$, and a non-empty subset $K \subseteq V$ of *terminals*, where usually n is a large number and $k := |K|$ is much smaller than n.

Graph sparsification is the notion of representing or approximately preserving certain properties of G on a simpler graph G'. There exist many variants of this technique, depending on which property of G we want to preserve. For example, if G' is constructed to approximately preserve the distances between vertices, we use *spanners* (see e.g. [Pau89]), or if we want that the Laplacians of G' and G have similar eigenvalues, we use *spectral sparsifiers* (see e.g. [SS08]). For this paper, however, we are interested in the scenario where G' is constructed to approximate the *cuts* of G, in which case we will call G' a *cut-sparsifier*. Different variations of cut sparsification exist: *edge (cut-)sparsification* replaces G with a graph $G' := (V, E', c')$, using the same vertices, but only a subset $E' \subset E$ of edges, which have new capacities c'. This technique has been studied by Karger [Kar99], who proved that it can be efficiently used to approximately solve flow and cut problems. Karger and Benczúr showed [BK02] that there exists an edge sparsifier $G' := (V, E', c')$, such that $|E'| = \mathcal{O}(n \log n)$ and which approximates *all cuts* in G within a $(1 + \varepsilon)$-factor. Moreover, G' can be constructed in $\mathcal{O}(m \log^3 n)$ time, which is very efficient.

Another way to do cut sparsification is *vertex (cut-)sparsification*, which is the variant we will look at in this paper. Here, G is replaced by a graph $G' := (K, E', c')$ defined only on the terminals (with new edges E' and capacities c'). Of course, G' cannot preserve all the cuts of G in any meaningful way, because it has (potentially much) fewer vertices. Here, we will require that G' preserves the cut structure of the terminals. More precisely, given a subset $U \subset K$ of terminals, let $h_K(U)$ be the value of the minimal cut separating U from $K \setminus U$ in G. We will require that the cuts of G' approximate h_K. The multiplicative factor by which the cut function of G' approximates h_K is called the *quality* of G'. The closer the quality is to 1, the better. This variant has been studied by Moitra [Moi13], who showed that there exist vertex sparsifiers of quality $\mathcal{O}\left(\frac{\log k}{\log \log k}\right)$. Moreover, sparsifiers of quality polylog(k) can be constructed in polynomial time in the size of G [Moi13]. Moitra's results have been generalized and improved, see for example [Eng+10] or [Chu12]. In this paper, we will focus on just explaining Moitra's proof [Moi13] that a sparsifier of quality $\mathcal{O}\left(\frac{\log k}{\log \log k}\right)$ exists, which only provides an exponential-time construction algorithm.

Sparsification is a useful technique, because it allows us to solve a problem on the sparsifier G' instead of solving it directly on G, and this is usually easier and faster, because G' is smaller than

G. Of course, for this to work, we need that G' preserves the structure of G which is relevant for the problem we are trying to solve; if this is the case, then we can hope that the solution on G' is a reasonable approximation to the solution on G. Concretely, for an application of vertex cut-sparsifiers, consider the informal scenario of (statically) routing demands between different pairs of terminals on the graph G. The routes we select to route the demands could have the problem that the total amount of flow $f(e)$ we send on some edges is much bigger than their capacities $c(e)$. Intuitively, the bottlenecks in G that prevent us from sending too much flow between different pairs of terminals can be read from the values of the function h_K. The idea is, therefore, to replace G with a smaller graph G', using only the terminals K as its vertices, s.t. the cut function h' of G' approximates h_K nicely. This means that routing G' is similar to routing G. But, because the vertex sparsifier G' has (potentially much) fewer vertices, it will be easier to efficiently find an almost optimal routing for it.

The informal scenario described in the previous paragraph can have different variants. For example, if the demands between the terminals are known, and sending $f(e) > c(e)$ flow on any edge e is not allowed, the problem can be interpreted as an instance of a *multicommodity flow problem*. If the demands are not given to us (*obliviousness*), and we are trying to keep the maximum *congestion* $f(e)/c(e)$ across all edges e as close as possible to what a non-oblivious algorithm can achieve, we get the *oblivious routing problem* (see e.g. [Gha20] for a precise definition). While both these problems have been studied, and approximation algorithms exist for solving them (see e.g. [LR99] for multicommodity flow, and [Räc02; Aza+03] for oblivious routing), the beauty of using the sparsifier G' relies in the fact that this will provide an approximation of the optimal solution for any problem whose solution depends on the terminal cut structure of G. Therefore, vertex sparsifiers are a general approach, and can be especially useful if the number of terminals is much smaller than the number of vertices.

For the rest of the paper, we mostly go along Moitra's article [Moi13]. Section 2 introduces the relevant formal definitions. Section 3 introduces the 0-extension problem. Section 4 builds on top of the previous sections to prove the main result, i.e. that vertex sparsifiers of quality $\mathcal{O}\left(\frac{\log k}{\log \log k}\right)$ always exist.

2 Definitions

In the following, we formally define the concepts we require to describe what a *good sparsifier* is:

Definition 1. The *cut function* $h : 2^V \to \mathbb{R}_{\geq 0}$ of G is defined as:

$$h(A) := \sum_{u \in A, v \notin A} c(u, v), \ A \subseteq V \tag{1}$$

The *terminal cut function* $h_K : 2^K \to \mathbb{R}_{\geq 0}$ of G is defined as:

$$h_K(U) := \min_{A \subseteq V, A \cap K = U} h(A), \ U \subseteq K \tag{2}$$

i.e. $h_K(U)$ is the value of the smallest cut A of G which contains exactly the terminals from U. We call a cut A which minimizes the quantity on the right hand side a *min-terminal-cut*.

Computing $h(A)$ for some $A \subseteq V$ can be done in $\mathcal{O}(n+m)$ time, because we only need to scan all nodes and edges. Computing $h_K(U)$ for some $U \subseteq K$ seems somewhat harder, but there is an

elegant way to do this: we add two *new* nodes s and t. We connect s to the nodes in U and the node t to the nodes in $K \setminus U$. All these new edges should have infinite capacities. It is easy to see that a min-cut $B \subseteq V \uplus \{s,t\}$ of this larger graph isolates the terminals in U from the terminals in $K \setminus U$ with minimal cost. Therefore, $A := B \setminus \{s\}$ is a min-terminal-cut, and $h_K(U) = h(A)$. We can now define what a vertex sparsifier is:

Definition 2. A graph $G' := (K, E', c')$ defined on the terminals of G is called a *(cut-)sparsifier* of G iff:

$$h_K(U) \leq h'(U), \text{ for all } U \subseteq K \tag{3}$$

Definition 2 requires that h' is an upper bound for h_K. Hence, our plan is to find a sparsifier G', s.t. h' is not much bigger than h_K. To measure how closely a sparsifier G' preserves the terminal cut structure of G, we therefore have to look at how big the gap between h' and h_K is. This motivates the following:

Definition 3. For a sparsifier $G' := (K, E', c')$ of G, the following quantity:

$$q := \max_{U \subseteq K} \frac{h'(U)}{h_K(U)} \tag{4}$$

is called the *quality* of G'. Note that always $q \geq 1$.

To avoid the cases where the denominator is 0, i.e. where U is disconnected from $K \setminus U$, or $U = \emptyset$ or $U = K$, we make the notation abuse $\frac{0}{0} := 1$. We will generally ignore such annoying cases and will consistently make the convention that all our denominators are non-zero. The terminology from Definition 3 is a bit unfortunate, because it allows us to say that we want to *minimize the quality* of our sparsifiers, but we decided to keep using this seemingly standard (and funny) word.

3 0-extensions

The next subsections introduce and discuss properties of three related problems, which will later, in Section 4, help us show that *good sparsifiers* (i.e. sparsifiers with quality close to 1) always exist.

3.1 Multiway cut and 0-extensions

The *multiway cut* problem is a natural generalization of the min-cut problem. Instead of having only two special nodes s and t, we are (again) given an undirected, capacitated graph $G := (V, E, c)$ with *terminals* $K := \{v_1, \ldots, v_k\} \subseteq V$. The objective is to partition V into k subsets $V_1, \ldots, V_k$, s.t. the subset V_i contains no other terminal than v_i and s.t. the sum of the costs of the edges which connect nodes from different subsets is minimized. (Alternatively, we want to find a subset of edges $E' \subseteq E$, whose removal would disconnect all terminals and which has minimal cost.) Even though this definition is clear enough, we will now provide an algebraic formulation which will prove useful:

Definition 4. For $\delta : K^2 \to \{0,1\}$, given by $\delta(u,v) := \begin{cases} 1 & \text{if } u = v \\ 0 & \text{if } u \neq v \end{cases}$

the multiway cut problem is equivalent to the following optimization problem:

$$\text{Find } \arg\min_f \sum_{\{u,v\}\in E} c(u,v)\delta(f(u),f(v)) \text{ under all } f : V \to K, \text{ s.t. } f(t) = t \text{ for all } t \in K$$

The function $f : V \to K$ is an encoding of our partitioning. It tells for any node $v \in V$ the terminal $f(v)$ to which it was assigned. In particular, all terminals are in their own subset, i.e. assigned to themselves, meaning that $f(t) = t$ is required for all $t \in K$. Definition 4 also uses a simple auxiliary δ-function. It is not hard to see that δ is a *metric*, i.e. it fulfills the following axioms: $\delta(x, y) \geq 0$ for all $x, y \in V$ with equality iff $x = y$ (*identity of indiscernibles*), $\delta(x, y) = \delta(y, x)$ for all $x, y \in V$ (*symmetry*), and $\delta(x, y) \leq \delta(x, z) + \delta(z, y)$ for all $x, y, z \in V$ (*triangle inequality*).

However, for the rest of this paper, we are going to use a weaker notion of *metric*, where we require all the axioms, except the condition $\delta(x, y) = 0 \implies x = y$. This will allow us to *glue* different nodes together, by making the distance between them equal to 0. The following is a generalization of Definition 4:

Definition 5. For $\delta : K^2 \to \mathbb{R}_{\geq 0}$ a fixed *metric*, the *0-extension problem* (instantiated for δ) is given by:

$$\text{Find } \arg\min_f \sum_{\{u,v\}\in E} c(u,v)\delta(f(u),f(v)) \text{ under all } f : V \to K, \text{ s.t. } f(t) = t \text{ for all } t \in K$$

In this context, we call f a *0-extension*. We use this name because f allows δ to indirectly extend its domain from K^2 to V^2 by identifying a non-terminal $v \in V \setminus K$ with its corresponding terminal $f(v) \in K$. Geometrically, this identification can be seen as making the distance between v and $f(v)$ equal to 0. The function $(u, v) \to \delta(f(u), f(v))$ is a metric on V^2, which agrees with δ on K^2. The 0-extension problem has been introduced by Karzanov in [Kar98], and there is much literature on this particular generalization of multiway cut. A result by Fakcharoenphol *et al.* [Fak+03] will be very useful later. To introduce this result, we need to define an even more general version of the 0-extension problem. For this purpose, we introduce metrics $\beta : V^2 \to \mathbb{R}_{\geq 0}$, such that $\delta = \beta$ on K^2, i.e. β is an extension of the metric δ:

Definition 6. For $\delta : K^2 \to \mathbb{R}_{\geq 0}$ a fixed metric on K, the *relaxed 0-extension problem* (instantiated for δ) is given by:

$$\text{Find } \arg\min_\beta \sum_{\{u,v\}\in E} c(u,v)\beta(u,v) \text{ under all metrics } \beta : V^2 \to \mathbb{R}_{\geq 0}, \text{ s.t. } \delta = \beta \text{ on } K^2$$

This relaxed version of the 0-extension problem was also introduced by Karzanov in [Kar98]. It intuitively relaxes the 0-extension problem because β is not forced to glue all non-terminals to terminals, unlike δ in Definition 5, which does this because we use f to preprocess its arguments. Moreover, unlike the 0-extension problem, the relaxed 0-extension problem is just a linear program, and can therefore be solved in polynomial time.

Figure 1 helps visualize the relaxed 0-extension problem. The black nodes are the terminals, and the number of edges we draw between two nodes $u, v \in V$ equals the capacity $c(u, v)$. The length of any edge connecting u and v equals the distance $\beta(u, v)$ between these two nodes. With this convention, the sum $\sum_{\{u,v\}\in E} c(u, v)\beta(u, v)$ to minimize is just the sum of all edges we draw. According to Definition 6, we are given the positions of the black nodes, i.e. β is fixed on the terminals, and we want to draw the (blue) non-terminals in the plane, i.e. to extend β to V^2, s.t. the sum of all edges we draw is minimized. The (non-relaxed) 0-extension problem is the same, except that we are forced to draw all the blue nodes "on top" of black nodes.

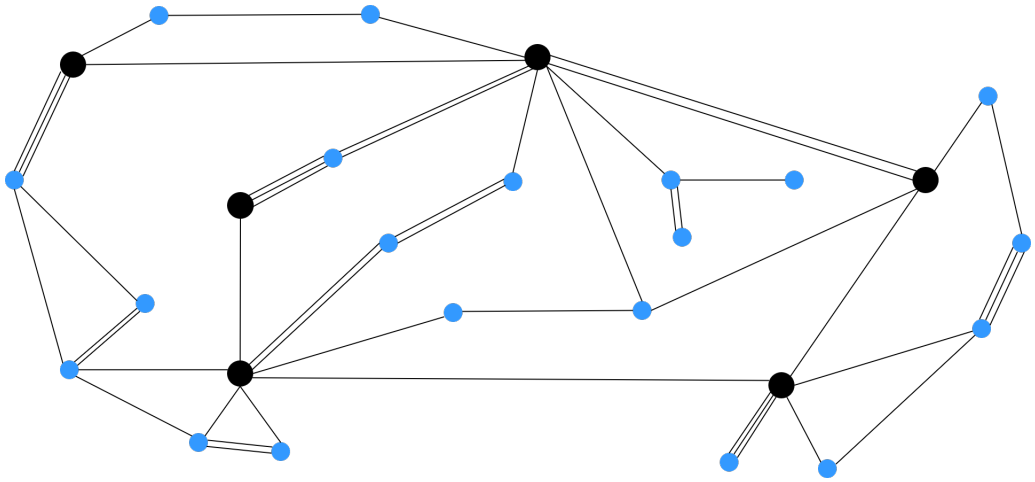

Figure 1: Visualizing the relaxed 0-extension problem

Let δ be any metric on K. Let β be the optimal metric for the relaxed 0-extension problem instantiated for δ, and the corresponding minimum be $OPT^* := \sum_{\{u,v\}\in E} c(u, v)\beta(u, v)$. Let f be the optimal 0-extension for the 0-extension problem instantiated for δ, and $OPT := \sum_{\{u,v\}\in E} c(u, v)\delta(f(u), f(v))$ the corresponding minimum. We first confirm formally that the problem given in Definition 6 is indeed a relaxation of the problem given in Definition 5:

Proposition. We have $OPT^* \leq OPT$.

Proof. The function $\alpha : V^2 \to \mathbb{R}_{\geq 0}$ defined as $\alpha(u, v) := \delta(f(u), f(v))$ is a metric on V^2, which agrees with δ on K^2. Because β is optimal for the relaxed 0-extension problem instantiated for δ, we obtain:

$$OPT^* = \sum_{\{u,v\}\in E} c(u, v)\beta(u, v) \leq \sum_{\{u,v\}\in E} c(u, v)\alpha(u, v) = \sum_{\{u,v\}\in E} c(u, v)\delta(f(u), f(v)) = OPT$$

□

It was proven by Fakcharoenphol *et al.* in [Fak+03] that the following also holds:

$$OPT \leq \mathcal{O}\left(\frac{\log k}{\log \log k}\right) OPT^* \tag{5}$$

In particular, this implies the following fact, which will be very useful later, because it provides a strategy to bound the objective function of the 0-extension problem:

Lemma 1. If $\beta : V^2 \to \mathbb{R}_{\geq 0}$ is a metric, s.t. $\sum_{\{u,v\}\in E} c(u,v)\beta(u,v) = 1$, then there exists a 0-extension f, s.t. $\sum_{\{u,v\}\in E} c(u,v)\beta(f(u),f(v)) = \mathcal{O}\left(\frac{\log k}{\log\log k}\right)$.

Proof. Let δ be the metric on K induced by β, i.e. $\delta(u,v) := \beta(u,v)$ for all $u,v \in K$. Let OPT^* be the optimal value of the relaxed 0-extension problem (instantiated for δ) and γ the minimizing -metric. Because $\sum_{\{u,v\}\in E} c(u,v)\beta(u,v) = 1$, and γ is optimal, we obtain $OPT^* = \sum_{\{u,v\}\in E} c(u,v)\gamma(u,v) \leq 1$.

Let f be the optimal 0-extension for the 0-extension problem instantiated for δ and OPT the corresponding minimum. Because β agrees with δ on K^2 (by definition), we get that $OPT = \sum_{\{u,v\}\in E} c(u,v)\delta(f(u),f(v)) = \sum_{\{u,v\}\in E} c(u,v)\beta(f(u),f(v))$. Using equation (5) and recalling that $OPT^* \leq 1$, we get that $\sum_{\{u,v\}\in E} c(u,v)\beta(f(u),f(v)) = \mathcal{O}\left(\frac{\log k}{\log\log k}\right)$, which is what we needed to prove. □

3.2 0-extensions are a tool for designing sparsifiers

The somewhat technical part leading to Lemma 1 is over. We now return to our initial problem of designing a (good) sparsifier for a capacitated graph $G := (V,E,c)$ with terminals $K \subseteq V$. 0-extensions can be used in the following way:

Definition 7. Let f be a 0-extension of G, i.e. a function $f : V \to K$, such that $f(t) = t$ for all $t \in K$. We define the cost function $c_f : K^2 \to \mathbb{R}_{\geq 0}$ as:

$$c_f(a,b) := \sum_{\substack{u,v\in V \\ f(u)=a, f(v)=b}} c(u,v) \tag{6}$$

The graph $G_f := (K, E_f, c_f)$, where $E_f := \{\{a,b\} : c_f(a,b) > 0\}$ is called the *graph induced by the 0-extension f.*

The following Theorem shows why 0-extensions are the right tool for designing a sparsifier:

Theorem 1. Let f be a 0-extension of G and G_f be the induced graph. Then G_f is a sparsifier of G, i.e.

$$h_K(U) \leq h_f(U) \text{ for all } U \subseteq K$$

Proof. Let $U \subseteq K$ and $A := f^{-1}(U) \subseteq V$ be the nodes mapped to terminals in U. We have:

$$h_f(U) = \sum_{a\in U, b\notin U} c_f(a,b) = \sum_{a\in U, b\notin U} \sum_{u\in f^{-1}(a), v\in f^{-1}(b)} c(u,v)$$

$$= \sum_{u\in f^{-1}(U), v\notin f^{-1}(U)} c(u,v) = h(f^{-1}(U)) = h(A)$$

Because A is a cut of V which contains exactly the terminals in U, by the definition of the terminal cut function (see Definition 1, equation (2)), we get that $h(A) \geq h_K(U)$. It follows that $h_f(U) = h(A) \geq h_K(U)$. □

Theorem 1, which shows that every 0-extension f gives us a sparsifier G_f "for free", is not surprising. When constructing G_f, we contract all edges which connect nodes that are assigned to the same terminal. This is equivalent to assuming that all these edges have infinite capacity, which means that the cuts of the contracted graph are an upper bound for the terminal cuts of the initial graph.

Figure 2 visualizes how these contractions act. On the left side we have G, and on the right side we have G_f. The terminals are $K := \{v_1, v_2, v_3\}$, the colors indicate the 0-extension f we use ($f(x) = v_1$, $f(y) = v_2$, $f(z) = v_3$), and the numbers next to the edges indicate the capacities. The bold edges in G connect nodes of different colors, so they will "become" edges connecting the corresponding terminals in G_f. For example, the two bold edges of capacities 3 and 5 connecting blue nodes to green nodes in G are added together to form an edge of capacity 8 connecting v_1 to v_2 in G_f. The non-bold edges are the ones that get contracted to obtain G_f from G in this way.

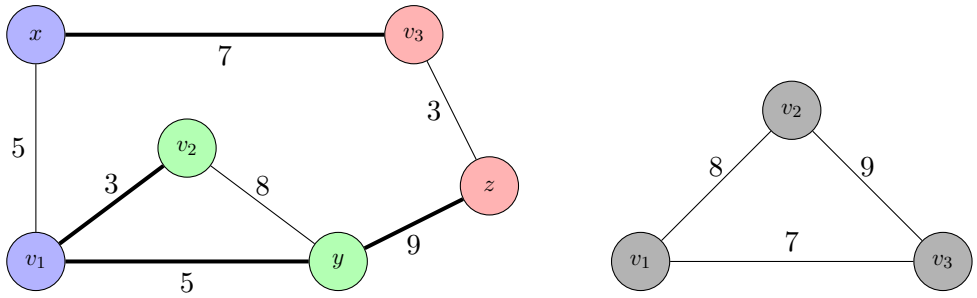

Figure 2: Visualizing G and G_f

4 Good sparsifiers exist

In this section we will prove that there always exist sparsifiers of quality $\mathcal{O}\left(\frac{\log k}{\log \log k}\right)$, which is the main result of Moitra's paper [Moi13]. The idea of the proof is to consider the following zero-sum game between two players P_1 and P_2:

- P_1 plays any 0-extension f / sparsifier G_f
- P_2 plays any cut $U \subseteq K$
- P_2 wins $N(f, U)$ points

We recall that the sparsifier G_f should have quality h_f/h_K as small as possible. Therefore, it makes sense to reward P_2 with points whenever he* manages to find a cut U for which this quantity

*We use "she" for P_1 and "he" for P_2

is big. We define:

$$N(f,U) := \frac{h_f(U)}{h_K(U)} = \sum_{\substack{\{u,v\}\in E \\ f(u),f(v) \text{ separated by } U}} \frac{c(u,v)}{h_K(U)} \tag{7}$$

The second equality can be justified as follows:

$$h_f(U) = \sum_{a\in U, b\notin U} c_f(a,b) = \sum_{a\in U, b\notin U} \sum_{u\in f^{-1}(a), v\in f^{-1}(b)} c(u,v) = \sum_{\substack{\{u,v\}\in E \\ f(u),f(v) \text{ separated by } U}} c(u,v) \tag{8}$$

4.1 Exploiting the theory of zero-sum games

There are at least two types of strategies we can consider for the two players:

- *Pure strategies*: in this model, P_1-s strategy is to choose a fixed 0-extension f and P_2-s strategy is to choose a fixed cut U.
- *Mixed strategies*: in this model, P_1-s strategy consists of a probabilistic distribution γ on 0-extensions. Whenever it is P_1-s turn, she chooses some 0-extension f by sampling from all possible 0-extensions using the distribution γ. Similarly, P_2-s strategy consists of a distribution μ on cuts of K. Whenever it is P_2-s turn, he chooses some cut U by sampling from all possible cuts using the distribution μ.

Considering pure strategies would be a big mistake: for some graphs, no single 0-extension f provides a good quality sparsifier G_f, so for such graphs we cannot find a good strategy for P_1. Taking mixed strategies will turn out to be crucial, because it will allow us to combine different 0-extensions to build our sparsifier, which will turn out to be a sufficiently powerful approach. Moreover, this choice allows us to use Neumann's Minimax Theorem, which provides us the following facts:

There exist optimal strategies (i.e. distributions) γ for P_1 and μ for P_2, which create a *saddle point*, i.e. the following holds:

- P_2 expectedly wins $\rho := E_{(f\leftarrow\gamma, U\leftarrow\mu)} N(f,U)$ points. ρ is called the *game value*.
- Any other distribution γ' on 0-extensions for P_1 leads to an expected win of at least ρ points for P_2, i.e. P_1 cannot improve by switching her strategy.
- Any other distribution μ' on cuts for P_2 leads to an expected win of at most ρ points for P_2, i.e. P_2 cannot improve by switching his strategy.

Using this, we obtain a lower and an upper bound for the game value ρ:

- Assume P_2 plays the optimal distribution μ on cuts of K. Then, for any 0-extension f that P_1 plays, P_2-s expected win is at least ρ (because he plays optimally), i.e.

$$\rho \le E_{(U\leftarrow\mu)} N(f,U) \tag{9}$$

- Assume P_1 plays the optimal distribution γ on 0-extensions. Then, for any cut U that P_2 plays, P_2-s expected win is at most ρ (because P_1 plays optimally), i.e.

$$E_{(f\leftarrow\gamma)} N(f,U) \le \rho \tag{10}$$

4.2 Lower bound for the game value

What happens if P_1 is great at this game? Let γ be the optimal distribution for P_1. Let $G' := \sum_{f \text{ is 0-extension}} \gamma(f) G_f$, where summing capacitated graphs (defined over the same set of nodes) is done by simply summing the capacities of the edges, and including all edges with strictly positive capacity. G' is a convex combination of sparsifiers induced by 0-extensions. It is easy to check that:

$$h'(U) = \sum_f \gamma(f) h_f(U) \geq \sum_f \gamma(f) h_K(U) = h_K(U) \text{ for any cut } U \subseteq K \tag{11}$$

where for the inequality we used Theorem 1 and where h' is the cut function of G'. We have, according to inequality (10):

$$\begin{aligned} \rho \geq E_{(f \leftarrow \gamma)} N(f, U) &= \sum_{f \text{ is 0-extension}} \frac{\gamma(f) h_f(U)}{h_K(U)} \\ &= \frac{1}{h_K(U)} \sum_f \gamma(f) h_f(U) = \frac{h'(U)}{h_K(U)} \end{aligned}$$

where for the last equality we used (11). This chain of equalities holds for any cut $U \subseteq K$. In particular, we obtain that the quality q of the sparsifier G' is upper bounded by ρ. We conclude:

Lemma 2. There exists a sparsifier G' of G, such that $1 \leq \max_{U \subseteq K} \frac{h'(U)}{h_K(U)} \leq \rho$.

4.3 Upper bound for the game value

If we find a good upper bound for ρ, then we are done, thanks to Lemma 2. Let μ be the optimal distribution for P_2. Using inequality (9), we have for any 0-extension f:

$$\rho \leq E_{(U \leftarrow \mu)} N(f, U) = \sum_{\{u,v\} \in E} \sum_{\substack{U \subseteq K \\ f(u), f(v) \text{ separated by } U}} \frac{\mu(U) c(u,v)}{h_K(U)} = \sum_{\{u,v\} \in E} c(u,v) D_\mu(f(u), f(v)) \tag{12}$$

where we defined:

$$D_\mu : K^2 \to \mathbb{R}_{\geq 0}, \; D_\mu(t, t') := \sum_{\substack{U \subseteq K \\ t, t' \text{ separated by } U}} \frac{\mu(U)}{h_K(U)} \tag{13}$$

Note that D_μ is a convex combination of (cut-)metrics, and therefore a metric too. Our strategy is to find an f, s.t. the quantity $\sum_{\{u,v\} \in E} c(u,v) D_\mu(f(u), f(v))$ is kept small. This quantity is precisely the objective function of the 0-extension problem instantiated for D_μ. Lemma 1 gives us the following hint for showing that a good f exists: find a metric β, defined on V^2, and not smaller than D_μ on K^2, s.t. $\sum_{\{u,v\} \in E} c(u,v) \beta(u,v)$ is small.

We will design an algorithm that runs on the graph G, and constructs a distance function $d : E \to \mathbb{R}_{\geq 0}$, which we will then use to define a β with the properties mentioned in the previous paragraph. The idea behind the algorithm is to define distances on G that are roughly consistent with D_μ. More precisely, because D_μ increases the distances between all pairs of terminals which are separated by some cut $U \subseteq K$, we will increase the distances on all edges which cross the min-terminal-cut A corresponding to U. We obtain:

Algorithm: Constructing d

Initially $d \equiv 0$.
for *cuts* $U \subseteq K$ **do**
  let $A \subseteq V$ be a min-terminal-cut of U (recall Definition 1)
  for *edges* $\{u, v\}$, *s.t.* u, v *are on different sides of* A **do**
    increase $d(u, v)$ by $\frac{\mu(U)}{h_K(U)}$
  end
end

Now define $\beta : V^2 \to \mathbb{R}_{\geq 0}$ as $\beta(u, v) :=$ length of the shortest u-v-path w.r.t. the distance function d. It can be shown that β is a metric on V^2. Moreover, β has the properties we wanted it to have:

Lemma 3. For all $t, t' \in K$, we have $D_\mu(t, t') \leq \beta(t, t')$.

Proof. Let $U \subseteq K$ be a cut which contributes to D_μ, i.e. (recall equation (13)) a cut such that t and t' are on different sides of U. Let A be the min-terminal-cut of U which the algorithm selected at Line 3, and consider a shortest t-t'-path P in G w.r.t. d. Because t, t' are on different sides of U, they are also on different sides of A, and therefore some edge e on the path P crosses the min-terminal-cut A. The algorithm increased $d(e)$ by $\frac{\mu(U)}{h_K(U)}$, and therefore $\beta(t, t') \geq d(e) \geq D_\mu(t, t')$. We obtain that $\beta \geq D_\mu$. □

Lemma 4. We have $\sum_{\{u,v\} \in E} c(u, v)\beta(u, v) = 1$.

Proof. β is a metric, so the *triangle inequality* implies $\beta(u, v) = d(u, v)$ for all $\{u, v\} \in E$. It suffices to show that $\sum_{\{u,v\} \in E} c(u, v)d(u, v) = 1$. Denote by S_U and S'_U the values of the left hand side of this expression before and after the algorithm considers the cut $U \subseteq K$. Let $\Delta_U := S'_U - S_U$.

Let A be the min-terminal-cut of U which the algorithm selected at Line 3. If the edge $\{u, v\} \in E$ crosses A, then d gets increased by $\frac{\mu(U)}{h_K(U)}$. If not, then d stays constant for this edge. It follows that:

$$\Delta_U = \sum_{\substack{\{u,v\} \in E \\ u,v \text{ separated by } A}} c(u, v)\frac{\mu(U)}{h_K(U)} = \frac{\mu(U)}{h_K(U)} \sum_{\substack{\{u,v\} \in E \\ u,v \text{ separated by } A}} c(u, v) = \frac{\mu(U)}{h_K(U)} h_K(U) = \mu(U)$$

Summing up over all cuts U finishes the proof:

$$\sum_{\{u,v\} \in E} c(u, v)d(u, v) = \sum_{U \subseteq K} \Delta_U = \sum_{U \subseteq K} \mu(U) = 1$$

□

Lemma 4 allows us to apply Lemma 1 to get:

Corollary 5. There exists a 0-extension f, s.t. $\sum_{\{u,v\} \in E} c(u, v)\beta(f(u), f(v)) \leq \mathcal{O}\left(\frac{\log k}{\log \log k}\right)$

We obtain using this particular f:

$$\rho \leq \sum_{\{u,v\}\in E} c(u,v) D_\mu(f(u), f(v)) \leq \sum_{\{u,v\}\in E} c(u,v)\beta(f(u), f(v)) \leq \mathcal{O}\left(\frac{\log k}{\log\log k}\right)$$

where we successively applied inequality (12), Lemma 3 and Corollary 5. We obtained an upper bound for the game value ρ:

Lemma 6. We have $\rho \leq \mathcal{O}\left(\frac{\log k}{\log\log k}\right)$

4.4 Combining the two inequalities

Combining Lemmas 2 and 6, we obtain our main result:

Theorem 2 ([Moi13])**.** For any $G := (V, E, c)$ with terminals $K \subseteq V$, there exists a sparsifier $G' := (K, E', c')$, s.t.

$$1 \leq \max_{U\subseteq K} \frac{h'(U)}{h_K(U)} \leq \mathcal{O}\left(\frac{\log k}{\log\log k}\right)$$

where $k := |K|$.

There are some important observations to make. Firstly, note that the bound $\mathcal{O}\left(\frac{\log k}{\log\log k}\right)$ has not been produced by our own calculations, but comes from inequality (5). In fact, this inequality can be strengthened under the assumption that G does *not* have the minor $K_{r,r}$ (the bipartite graph with r nodes on both sides). In this case, we have (see [Moi13; CKR01; FT03]):

$$OPT \leq \mathcal{O}(r^2) OPT^* \tag{14}$$

which then implies that Theorem 2 also holds for the upper bound $\mathcal{O}(r^2)$.

Secondly, the proof of Theorem 2 also provides an algorithm to construct the sparsifier G'. To do this, we compute the optimal distribution γ on 0-extensions for P_1, which can be done by solving the linear program corresponding to the zero-sum game, and then computing $G' := \sum_f \gamma(f) G_f$. Unfortunately, because the number of 0-extensions f is equal to the number of functions $f : V \to K$, s.t. $f(t) = t$ for all $t \in K$, writing all the equations of the game can require $\mathcal{O}(k^n)$ time. The problem of constructing a sparsifier efficiently is indeed quite different, and is beyond the scope of this paper. It is worth noting that our main source [Moi13] tackles this problem too, by constructing a linear program in polynomial time, which is feasible thanks to Theorem 2. The solution of the linear program encodes a good sparsifier. More general and more efficient constructions exist, see for example [Eng+10; Chu12].

5 Conclusion

We introduced vertex (cut-)sparsifiers as graphs G' defined on the terminals K of another graph G, such that the cut function of G' is an upper bound of the terminal cut function h_K of G. We introduced the notion of quality and then proved that there exist sparsifiers whose quality is at least as good as $\mathcal{O}\left(\frac{\log k}{\log\log k}\right)$. The proof used game theory and, in particular, Neumann's Minimax Theorem. It also exploited some properties of the 0-extension problem, like the fact that bounding its objective function can be done by solving the relaxed 0-extension problem instead, which is

easier. Furthermore, it was essential that 0-extensions induce sparsifiers, and the good sparsifier we found turned out to be a convex combination of such 0-extension induced graphs. It is worth noting that Moitra could not find better results by allowing the sparsifier to include polynomially many (i.e. $k^{\mathcal{O}(1)}$) nodes instead of just $k := |K|$ many [Moi13]. It remains an open question how such a relaxation could be exploited to improve the quality. On the other hand, if we would allow 2^{2^k} many nodes, we could construct a quality-1, "perfect" sparsifier G', which is then called a *mimicking network* [KR12]. Another important topic, which we did not discuss, is the efficient construction of sparsifiers. Our approach lead to an exponential-time algorithm; for better algorithms, we point out to [Moi13], [Eng+10] and [Chu12].

References

[Pau89] L. Paul Chew. "There are planar graphs almost as good as the complete graph". In: *Journal of Computer and System Sciences* 39.2 (1989), pp. 205–219.

[SS08] D. A. Spielman and N. Srivastava. "Graph Sparsification by Effective Resistances". In: *Proceedings of the Fortieth Annual ACM Symposium on Theory of Computing*. STOC '08. New York, NY, USA: Association for Computing Machinery, 2008.

[Kar99] D. Karger. "Random Sampling in Cut, Flow, and Network Design Problems". In: *Mathematics of Operations Research* 24 (Feb. 1999), pp. 383–413.

[BK02] A. Benczúr and D. Karger. "Randomized Approximation Schemes for Cuts and Flows in Capacitated Graphs". In: *SIAM Journal on Computing* 44 (Aug. 2002).

[Moi13] A. Moitra. "Vertex Sparsification and Oblivious Reductions". In: *SIAM Journal on Computing 42, no. 6* (2013).

[Eng+10] M. Englert, A. Gupta, R. Krauthgamer, H. Räcke, I. Talgam-Cohen, and K. Talwar. "Vertex Sparsifiers: New Results from Old Techniques." In: Jan. 2010, pp. 152–165.

[Chu12] J. Chuzhoy. "On Vertex Sparsifiers with Steiner Nodes". In: *Proceedings of the Annual ACM Symposium on Theory of Computing* (Apr. 2012).

[Gha20] M. Ghaffari. *Advanced Algorithms (lecture notes)*. Available at: https://people.inf.ethz.ch/gmohsen/AA20/AAscript.pdf. Dec. 2020.

[LR99] T. Leighton and S. Rao. "Multicommodity Max-Flow Min-Cut Theorems and Their Use in Designing Approximation Algorithms". In: *J. ACM* 46.6 (1999).

[Räc02] H. Räcke. "Minimizing congestion in general networks". In: *The 43rd Annual IEEE Symposium on Foundations of Computer Science, 2002. Proceedings.* 2002, pp. 43–52.

[Aza+03] Y. Azar, E. Cohen, A. Fiat, H. Kaplan, and H. Räcke. "Optimal Oblivious Routing in Polynomial Time". In: *Proceedings of the Thirty-Fifth Annual ACM Symposium on Theory of Computing*. STOC '03. New York, NY, USA: Association for Computing Machinery, 2003, pp. 383–388.

[Kar98] A. V. Karzanov. "Minimum 0-Extensions of Graph Metrics". In: *European Journal of Combinatorics* (1998), pp. 71–101.

[Fak+03] J. Fakcharoenphol, C. Harrelson, S. Rao, and K. Talwar. "An improved approximation algorithm for the 0-extension problem". In: *Symposium on Discrete Algorithms* (2003), pp. 257–265.

[CKR01] G. Calinescu, H. Karloff, and Y. Rabani. "Approximation Algorithms for the 0-Extension Problem". In: SODA '01. Washington, D.C., USA: Society for Industrial and Applied Mathematics, 2001, pp. 8–16.

[FT03] J. Fakcharoenphol and K. Talwar. "An Improved Decomposition Theorem for Graphs Excluding a Fixed Minor". In: *RANDOM-APPROX*. 2003.

[KR12] R. Krauthgamer and I. Rika. "Mimicking Networks and Succinct Representations of Terminal Cuts". In: *Proceedings of the Annual ACM-SIAM Symposium on Discrete Algorithms* (July 2012).