# Quantum speedups for graph sparsification, graph cut problems and Laplacian solving*

Simon Apers† Troy Lee‡ Ronald de Wolf§

## 1 Introduction

The complexity of many graph problems naturally scales with the number of edges in the graph. Graph sparsification aims to reduce this number of edges, while preserving certain quantities of interest. If we wish to approximate the weight of a minimum cut then the sparsified graph should approximately preserve the cut values of the original graph. If we wish to solve a linear system in the Laplacian of the graph, then the sparsified graph should approximately preserve certain spectral structure of the original graph. Through a long line of work [Kar94, BK96, ST11] the existence of so-called *spectral sparsifiers* was established. An $\varepsilon$-spectral sparsifier $H$ of a graph $G$ is a reweighted subgraph such that

$$(1-\epsilon)L_G \preceq L_H \preceq (1+\epsilon)L_G,$$

with $L_H$ and $L_G$ the Laplacian matrices associated to $H$ resp. $G$. This ensures that both the cut values and the spectral structure of $G$ are approximately preserved in $H$. It was shown that for any $G$ with $n$ vertices and $m$ edges there exists an $\varepsilon$-spectral sparsifier $H$ with only $\widetilde{O}(n/\varepsilon^2)$ edges, and moreover that $H$ can be constructed in time $\widetilde{O}(m)$.

This result and its predecessors lie at the basis of a huge range of efficient graph algorithms. It led to the first near-linear time algorithms for cut problems such as computing the edge connectivity or weight of a minimum cut [Kar00], or for finding an approximate sparsest cut or balanced separator [ARV09]. Spectral sparsifiers also were a critical building block of the first near-linear time algorithms for solving Laplacian systems [ST14]. This has applications in broad areas such as learning, computer vision and image processing, and graph clustering (see [Ten10] for a survey of the so-called "Laplacian paradigm").

## 2 Quantum speedup for graph sparsification, cut approximation and Laplacian solving (based on [AdW19])

In this work we give a *quantum* algorithm for spectral sparsification. We consider both the adjacency array model, in which we can query for the degree of a vertex or the $i$-th neighbor of a vertex (and the weight of the corresponding edge), and the adjacency matrix model, in which we can query for the existence and weight of an edge between an arbitrary pair of vertices.

**Theorem 1.** *Let $G$ be a weighted graph with $n$ vertices and $m$ edges. There is a quantum algorithm that outputs with high probability the explicit description of an $\epsilon$-spectral sparsifier of $G$ with $\widetilde{O}(n/\epsilon^2)$ edges. The algorithm has time complexity $\widetilde{O}(\sqrt{mn}/\epsilon)$ in the adjacency array model and $\widetilde{O}(n^{3/2}/\epsilon)$ in the adjacency matrix model.*

*This submission is a merge of the papers "Quantum speedup for graph sparsification, cut approximation and Laplacian solving." by Simon Apers and Ronald de Wolf [AdW19] and "Quantum query complexity of edge connectivity" by Simon Apers and Troy Lee [AL20].

†ULB, Brussels and CWI, Amsterdam. Email: smgapers@gmail.com
‡Centre for Quantum Software and Information, University of Technology Sydney. Email: troyjlee@gmail.com
§QuSoft, CWI and University of Amsterdam, Amsterdam. Email: rdewolf@cwi.nl

This improves on the classical time complexity which is $\Omega(m)$ in the adjacency array model and $\Omega(n^2)$ in the adjacency matrix model. The algorithm builds on a range of quantum and classical results. As a blueprint we use the iterative sparsification algorithm by Koutis and Xu [KX16]. This algorithm relies on so-called *graph spanners* which are sparse subgraphs that approximately preserve all shortest-path distances in the graph. The Koutis-Xu algorithm first constructs a near-constant number of *graph spanners* in the graph, after which it randomly downsamples the remaining set of edges. Applying this scheme a logarithmic number of times leads to a spectral sparsifier with $\widetilde{O}(n/\varepsilon^2)$ edges. Our quantum algorithm builds on a faster quantum algorithm for constructing spanners. To this end we combine a classical spanner algorithm by Thorup and Zwick [TZ05] with a quantum algorithm for constructing single-source shortest-path trees by Dürr, Heiligman, Høyer and Mhalla [DHHM06]. This allows us to speed up the first phase of the Koutis-Xu algorithm. The second phase amounts to downsampling the remaining set of edges. While this is trivially done classically by iterating over the full edge set, our quantum algorithm works in sublinear time and hence we must rely on an efficient $k$-independent hash function by Christiani, Pagh and Thorup [CPT15] to *implicitly* downsample the edge set. Combined with the quantum spanner algorithm this gives a quantum sparsification algorithm with time complexity $\widetilde{O}(\sqrt{mn}/\varepsilon^2)$ in the adjacency array model and $\widetilde{O}(n^{3/2}/\varepsilon^2)$ in the adjacency matrix model. Finally, we quadratically improve the $\varepsilon$-dependency by using a *bootstrap* trick based on work by Spielman and Srivastava [SS11].

We prove a matching lower bound, showing that the runtime of our quantum algorithm is optimal up to polylogarithmic factors.

**Theorem 2.** *A quantum algorithm that for any graph with $n$ vertices and $m$ edges explicitly constructs with high probability an $\epsilon$-cut sparsifier of the graph must make $\widetilde{\Omega}(\sqrt{mn}/\epsilon)$ queries in the adjacency array model and $\widetilde{\Omega}(n^{3/2}/\varepsilon)$ queries in the adjacency list model.*

Roughly, we obtain the lower bound by embedding an "unsparsifiable" graph in a larger graph. This unsparsifiable graph is derived from a construction by Andoni, Chen, Krauthgamer, Qin, Woodruff and Zhang [ACK$^+$16], where it was used for the purpose of proving lower bounds on the space complexity, rather than the time complexity. The lower bound on the quantum query complexity is then proven using a lower bound on the quantum query complexity of the composition of a relational problem with a Boolean function, which was proven by Belovs and Lee [BL20], prompted by our question to them.

We can combine our sparsification algorithm with classical algorithms to achieve a quantum speedup for many of the aforementioned applications. Examples include a quantum speedup for approximating the weight of a maximum cut, a minimum cut, a minimum $st$-cut or a sparsest cut in the original graph. Combining our sparsification algorithm with classical algorithms for Laplacian solving gives the following theorem, where we use the induced norm $\|y\|_L = \sqrt{y^T L y}$ for a positive semi-definite matrix $L$.

**Theorem 3.** *Let $L$ be the Laplacian of a weighted graph with $n$ vertices and $m$ edges. There exists a quantum algorithm that outputs with high probability an explicit description of an approximate solution $\tilde{x} \in \mathbb{R}^n$ to the linear system $Lx = b$ such that $\|\tilde{x} - x\|_L \leq \varepsilon \|x\|_L$. The algorithm has time complexity $\widetilde{O}(\sqrt{mn}/\varepsilon)$ in the adjacency array model and $\widetilde{O}(n^{3/2}/\varepsilon)$ in the adjacency matrix model.*

This leads to quantum speedups for approximating effective resistances, random walk commute and cover times, and eigenvalues of the Laplacian. We also demonstrate a quantum speedup for spectral $k$-means clustering.

## 3 Quantum query complexity of edge connectivity (based on [AL20])

At first sight, sparsifiers can only yield approximate solutions to cut problems. However, Karger [Kar00] showed that sparsifiers can also lead to an efficient algorithm for *exactly* determining the weight of a minimum cut in a graph. This requires a significantly more involved usage of sparsifiers. Since our quantum sparsification led to a quantum speedup for *approximating* the weight of a minimum cut, this raises the question of whether we can also find a quantum speedup for *exactly* computing its weight.

First we show that for general weighted graphs no quantum speedup is possible for exactly computing the weight of a minimum cut. Indeed, we show that in the worst case $\Omega(n^2)$ queries are needed to compute the weight of a minimum cut by a quantum algorithm in both the adjacency matrix and adjacency array models. The same lower bound applies if we wish to output the edge set corresponding to a minimum cut, or the shores (induced bipartition) of a minimum cut.

Next we consider the same problem for simple (unweighted) graphs, in which case the weight of a minimum cut is called the *edge connectivity*, and it corresponds to the least number of edges whose removal disconnects the graph. Computing the edge connectivity of a graph is a fundamental computational problem and has great practical importance, with applications to clustering algorithms and evaluating network reliability, amongst others [PQ82]. It was shown only recently that the edge connectivity can be computed in time $\widetilde{O}(m)$ even by *deterministic* algorithms [KT15].

In contrast to the case for weighted graphs, we show that a quantum speedup in the query complexity of computing the edge connectivity is possible.

**Theorem 4.** *Let $G$ be a simple graph with $n$ vertices and $m$ edges. There is a quantum algorithm that outputs with high probability the edge connectivity of $G$ using $\widetilde{O}(\sqrt{mn})$ queries in the adjacency array model and $\widetilde{O}(n^{3/2})$ queries in the adjacency matrix model.*

Note that, in contrast to the algorithms of the previous section, these upper bounds are only for the *query* complexity, and we leave the existence of *time-efficient* quantum algorithms for edge connectivity as an open question. Apart from the edge connectivity, with the same query complexity the algorithms can also output the edge set corresponding to a minimum cut, or the shores of a minimum cut. These bounds improve on the trivial classical query complexities $\Omega(m)$ and $\Omega(n^2)$ in the adjacency array and matrix models, respectively. The upper bound in the adjacency matrix model is tight up to polylogarithmic factors, whereas the best lower bound in the adjacency array model that we know of is $\Omega(n)$. These bounds follow from $\Omega(n^{3/2})$ and $\Omega(n)$ lower bounds for deciding graph connectivity in the adjacency matrix and array models, respectively, proven by Dürr, Heiligman, Høyer and Mhalla [DHHM06].

For comparison, if we directly combine a classical edge connectivity algorithm with the quantum sparsification algorithm, then we can only get an $\varepsilon$-approximation of the edge connectivity using $\widetilde{O}(\sqrt{mn}/\varepsilon)$ queries in the adjacency array model and $\widetilde{O}(n^{3/2}/\varepsilon)$ queries in the adjacency matrix model. Our bounds improve on both of these upper bounds. We do so by more carefully combining the quantum sparsification algorithm with a lemma on the structure of near-minimum cuts given by Rubinstein, Schramm, and Weinberg [RSW18].

## References

[ACK+16] Alexandr Andoni, Jiecao Chen, Robert Krauthgamer, Bo Qin, David P. Woodruff, and Qin Zhang. On sketching quadratic forms. In *Proceedings of the 2016 Innovations in Theoretical Computer Science Conference (ITCS)*, pages 311–319. ACM, 2016. arXiv: 1511.06099

[AdW19] Simon Apers and Ronald de Wolf. Quantum speedup for graph sparsification, cut approximation and Laplacian solving. In *Proceedings of the 61st IEEE Symposium on Foundations of Computer Science (FOCS)*. IEEE, 2019. arXiv: `1911.07306`

[AL20] Simon Apers and Troy Lee. Quantum query complexity of edge connectivity. arXiv: `2011.09823`, 2020.

[ARV09] Sanjeev Arora, Satish Rao, and Umesh Vazirani. Expander flows, geometric embeddings and graph partitioning. *Journal of the ACM*, 56(2):5, 2009. doi: `10.1145/1502793.1502794`

[BK96] András A. Benczúr and David R. Karger. Approximating $s-t$ minimum cuts in $\widetilde{O}(n^2)$ time. In *Proceedings of the 28th ACM Symposium on Theory of Computing (STOC)*, volume 96, pages 47–55. ACM, 1996. doi: `10.1145/237814.237827`

[BL20] Aleksandrs Belovs and Troy Lee. The quantum query complexity of composition with a relation. arXiv: `2004.06439`, 2020.

[CPT15] Tobias Christiani, Rasmus Pagh, and Mikkel Thorup. From independence to expansion and back again. In *Proceedings of the 47th ACM Symposium on Theory of Computing (STOC)*, pages 813–820. ACM, 2015. arXiv: `1506.03676`

[DHHM06] Christoph Dürr, Mark Heiligman, Peter Høyer, and Mehdi Mhalla. Quantum query complexity of some graph problems. *SIAM Journal on Computing*, 35(6):1310–1328, 2006. arXiv: `quant-ph/0401091`

[Kar94] David R. Karger. Using randomized sparsification to approximate minimum cuts. In *Proceedings of the 5th ACM-SIAM Symposium on Discrete Algorithms (SODA)*, volume 94, pages 424–432, 1994.

[Kar00] David R. Karger. Minimum cuts in near-linear time. *Journal of the ACM*, 47(1):46–76, 2000. arXiv: `cs/9812007`

[KT15] Ken-ichi Kawarabayashi and Mikkel Thorup. Deterministic global minimum cut of a simple graph in near-linear time. In *Proceedings of the 47th ACM Symposium on Theory of Computing (STOC)*, pages 665–674, 2015. arXiv: `1411.5123`

[KX16] Ioannis Koutis and Shen Chen Xu. Simple parallel and distributed algorithms for spectral graph sparsification. *ACM Transactions on Parallel Computing (TOPC)*, 3(2):1–14, 2016. arXiv: `1402.3851`

[PQ82] Jean-Claude Picard and Maurice Queyranne. Selected applications of minimum cuts in networks. *INFOR: Information Systems and Operational Research*, 20(4):394–422, 1982. doi: `10.1080/03155986.1982.11731876`

[RSW18] Aviad Rubinstein, Tselil Schramm, and S. Matthew Weinberg. Computing exact minimum cuts without knowing the graph. In *Proceedings of the 9th Innovations in Theoretical Computer Science Conference (ITCS)*, pages 39:1–39:16. LIPICS, 2018. arXiv: `1711.03165`

[SS11] Daniel A. Spielman and Nikhil Srivastava. Graph sparsification by effective resistances. *SIAM Journal on Computing*, 40(6):1913–1926, 2011. arXiv: `0803.0929`

[ST11] Daniel A. Spielman and Shang-Hua Teng. Spectral sparsification of graphs. *SIAM Journal on Computing*, 40(4):981–1025, 2011. arXiv: `0808.4134`

[ST14] Daniel A. Spielman and Shang-Hua Teng. Nearly linear time algorithms for preconditioning and solving symmetric, diagonally dominant linear systems. *SIAM Journal on Matrix Analysis and Applications*, 35(3):835–885, 2014. arXiv: `cs/0607105`

[Ten10] Shang-Hua Teng. The Laplacian paradigm: Emerging algorithms for massive graphs. In *International Conference on Theory and Applications of Models of Computation*, pages 2–14. Springer, 2010. doi: `10.1007/978-3-642-13562-0_2`

[TZ05] Mikkel Thorup and Uri Zwick. Approximate distance oracles. *Journal of the ACM*, 52(1):1–24, 2005. doi: `10.1145/1044731.1044732`