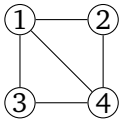
Hiding, Shuffling, and Cycle Finding: Quantum Algorithms on Edge Lists

Amin Shiraz Gilani[1], Daochen Wang[2], Pei Wu[3], and Xingyu Zhou[2]
[1]*University of Maryland,* [2]*University of British Columbia,* [3]*The Pennsylvania State University*

The study of graph problems forms a cornerstone of research in theoretical computer science. These problems have been studied when the input model is the *adjacency matrix* or *adjacency list*. Both models build structural cues into the input: an adjacency matrix assigns each vertex a given row and column, while an adjacency list uses vertices to order edges. These cues allow for fast computation but are arguably not intrinsic to the underlying graph.

In this work, we study graph problems in the edge list model, a minimalist model where the graph is given as an unordered multiset of edges. When computing on edge lists, algorithms must build the graph's structure for themselves and not rely on any input cues. Thus, studying computation on edge lists offers a new window into the relationship between a graph problem's *structure* and *complexity*.



Edge list: {1,2} {3,4} {1,4} {1,3} {2,4}

Adjacency list: 1: 2 4 3 2: 4 1 3: 1 4 4: 1 3 2

An edge list can be seen as a shuffled adjacency list: both inputs specify the same displayed graph.

We study the edge list model in the context of *quantum query complexity*, meaning that the complexity of a graph problem is characterized by the number of times a quantum computer queries an input edge list. Inspired by the fact that the triangle problem in the adjacency matrix model has long driven innovations in quantum query complexity [1–6], we were driven to study the same problem but in the edge list model. Specifically, we study three variants of the triangle problem.

- TriangleEdge asks whether there exists a triangle containing a known target edge and raises general questions about how a problem's complexity increases if its input is *hidden* among irrelevant data.
- TriangleVertex asks whether there exists a triangle containing a known target vertex and raises general questions about how a problem's complexity increases if its input is *shuffled*.
- Triangle asks whether a triangle exists (without placing constraints). We show its deep structural connection to the 3-distinctness problem, which has been extensively studied both in the worst-case setting by complexity theorists seeking to better understand problem structure [7–10] and in the average-case setting by cryptographers seeking to undergird the security of cryptosystems [11, 12].

We provide tight or nearly tight results for these problems as well as some first answers to the general questions they raise. Furthermore, we generalize our results of Triangle to k-cycle finding, a problem motivated by its close ties to the well-studied problem of k-distinctness.

Overview of main results

We write $Q(\cdot)$ for the usual worst-case (bounded-error) quantum query complexity.

TriangleEdge and Hiding. In TriangleEdge, when searching for a triangle containing a target edge $\{u, v\}$, any edge in the input of the form $\{u', v'\}$ such that $\{u', v'\} \cap \{u, v\} = \emptyset$ can be viewed as junk data *hiding* the relevant part of the input. This observation motivates our general definition of hiding.

Definition 1 (Hiding)**.** Given a function f on length-d strings and $m \geq d$, $\mathsf{HIDE}_m[f]$ is the function whose input $\tilde{x}$ is a length-m string having (i) some length-d subsequence x that is an input to f, and (ii) $(m-d)$ junk symbols that fill the remaining positions; and $\mathsf{HIDE}_m[f](\tilde{x})$ is defined to be $f(x)$.

Let ED_d denote the element distinctness function on length-d strings that decides whether the input string contains a collision. We show that $Q(\mathsf{HIDE}_m[\mathrm{ED}_d]) = \widetilde{\Theta}(\sqrt{m}d^{1/6})$ by exploiting "block-inputs", i.e., inputs consisting of blocks of symbols each with exactly one non-junk symbol. As a corollary, $Q(\mathsf{TriangleEdge}) = \widetilde{\Theta}(\sqrt{m}d^{1/6})$, where the input is a length-m edge list and the target edge has d neighboring edges. In fact, block-inputs help us prove $Q(\mathsf{HIDE}_m[f]) = \Omega(\sqrt{m/d}\cdot Q(f))$ for *any* $f : \Sigma^d \to \{0,1\}$. We conjecture a matching upper bound *for all symmetric* f. We support the conjecture by proving in two cases: (i) $\Sigma = \{0,1\}$; and (ii) Q is replaced by R, the randomized query complexity.

TriangleVertex and Shuffling. In TriangleVertex, when seeking a triangle containing a target vertex u, it is helpful to think of the input edge list as consisting of two parts: part A containing edges that are incident to u, and part B containing the remaining edges. We do not apriori know where parts A and B are but neither contains junk data; indeed, two edges of the triangle must come from A and one from B. Part of the problem's difficulty is due to the *shuffling* between parts A and B. Moreover, an edge list can be viewed as a shuffled adjacency list. These observations motivate our general definition of shuffling.

Definition 2 (Shuffling)**.** Given a function f, we define $\mathsf{SHUFFLE}[f]$ to be the function whose input $\tilde{x}$ is a permutation of some input x to f such that each symbol of $\tilde{x}$ additionally contains its position pre-permutation; and $\mathsf{SHUFFLE}[f](\tilde{x})$ is defined to be $f(x)$.

For general functions f, it is easy to see that $Q(\mathsf{SHUFFLE}[f])$ can be massively larger than $Q(f)$, while for symmetric f, $Q(\mathsf{SHUFFLE}[f]) = Q(f)$ is clear. Thus, the interesting question is what happens when f is *partially symmetric*? We show massive separations can exist between $Q(\mathsf{SHUFFLE}[f])$ and $Q(f)$ even if f is partially symmetric by being a graph property. Our result has the interpretation that using the edge list, as opposed to the adjacency list or adjacency matrix, can massively slow down the quantum computation of certain graph properties.

Triangle and Cycle Finding. We now ask if a triangle or, more generally, a length-k cycle exists in a length-m edge list. Here, $k \geq 3$ is an arbitrary but fixed integer. Our main result is

Theorem 3 (Cycle finding, informal)**.** *The quantum query complexity of finding a length-k cycle in a length-m edge list is $Q(k\text{-}\mathsf{CYCLE}_m) \geq \widetilde{\Omega}(m^{3/4-1/(2^{k+2}-4)})$. Furthermore, if the input has max-degree $o(\log(m))$, then $Q(k\text{-}\mathsf{CYCLE}_m) = m^{3/4-1/(2^{k+2}-4)\pm o(1)}$.*

We find Theorem 3 interesting for several reasons. First, proving the theorem requires us to develop new tools for the recording query framework [12, 13, 13, 14, 14–18]. This framework is well suited for proving quantum query lower bounds when the input is sampled from a *product distribution*. Extending the framework to non-product distributions is a well-known challenge. Ideally, we would like to consider a product distribution supported *only* on graphs with low max-degree. This means the graph must be sparse, so the low max-degree condition makes finding a cycle harder, and hence proving the lower bound easier. To see this, consider triangle finding. Without a degree bound, the edge queried at the t-th step could help an algorithm find a large number ($\Omega(t)$) of *wedges*, i.e., length-2 paths. As each wedge can be completed to a triangle, the more wedges the algorithm finds, the easier triangle finding becomes. Unfortunately, a product distribution on edge lists cannot be *only* supported on those with low max-degree. While sampling a graph with high max-degree is a rare event, it is always *possible*.

Our main technical contributions are the Mirroring and Exclusion lemmas (lemmas 5.14 and 5.18 in the technical manuscript) that can handle such rare events within the recording query framework. These tools can be used together to isolate and bound the contribution to an algorithm's success probability when rare events occur. They extend the recording query framework to treat any non-product distribution that results from conditioning a product distribution on the absence of rare events.

Theorem 3 is also interesting because the lower bound there matches the best-known upper bound on $Q(k\text{-DISJ}_m)$, up to logarithmic factors. Here, $k\text{-DISJ}_m$ is the k-distinctness function that computes whether a length-m input string contains k repetitions of the same symbol. For every $k \geq 3$, the upper and lower bounds on $Q(k\text{-DISJ}_m)$ are far from matching and we summarize the state of affairs in Table 1. This is despite numerous attempts to raise the lower bound using the (dual) polynomial method [19, 20], adversary method [21, 22], and even recording queries method [12].

Problem	k-DISJ$_m$ **Lower Bound**	k-DISJ$_m$ **Upper Bound**	k-CYCLE$_m$ **Tight Bound**
$k = 3$	$\Omega(m^{2/3})$ [23]	$O(m^{5/7})$ [7]	$m^{5/7 \pm o(1)}$ (this work)
$k \geq 4$	$\widetilde{\Omega}(m^{3/4-1/(4k)})$ [20]	$O(m^{3/4-1/(2^{k+2}-4)})$ [7]	$m^{3/4-1/(2^{k+2}-4) \pm o(1)}$ (this work)

Table 1: Best known quantum query complexity bounds for k-DISJ and k-CYCLE on length-m inputs. For k-CYCLE, we assume the input has max-degree $o(\log(m))$. For $k = 3$, the k-DISJ lower bound is inherited from that for $k = 2$, proven over two decades ago [23]. Can k-CYCLE help us close the long-standing gaps for k-DISJ?

Currently, the best-known upper bound on $Q(k\text{-DISJ}_m)$ is due to Belovs using an (adaptive) learning graph algorithm. This bound is also the conjectured tight lower bound on $Q(k\text{-DISJ}_m)$. The matching of this conjectured tight bound with our proven bounds in Theorem 3 for k-CYCLE$_m$ is *not a coincidence*. The upper bounds match because we adapt Belovs's learning graph algorithm for k-DISJ to k-CYCLE. The quantum query complexity of our adaptation involves an extra factor that is exponential in the max-degree, which is $m^{o(1)}$ when the max-degree is $o(\log(m))$. The lower bounds match because the "collision structure" of a uniformly random length-m edge list on $n = \Theta(m)$ vertices resembles that of the conjectured hard distribution on inputs to k-DISJ$_m$ from [9, 24]. The former contains $\Theta(m)$ length-$(k-1)$ paths and $O(1)$ length-k cycles, while the latter similarly contains $\Theta(m)$ size-$(k-1)$ collisions and $O(1)$ size-k collisions. The intuition is that the many size-$(k-1)$ collisions (resp. length-$(k-1)$ paths) present many dead-ends that effectively hide the location of the size-k collision (resp. length-k path). Thus, the lower bound of Theorem 3 proves that this intuition is correct for k-CYCLE. Moreover, k-CYCLE is the *first* example of a problem for which Belovs's learning graph algorithm is provably optimal, up to an $m^{o(1)}$ factor.

Given the striking similarities between the collision structures and algorithms for k-DISJ and k-CYCLE, it is natural to wonder if the lower bound of Theorem 3 can be lifted to a tight lower bound for k-DISJ. We show that this is indeed the case, *provided* the following conjecture holds.

Conjecture 4 (Cycle finding conjecture, informal)**.** *Given an edge list for a graph on n vertices and $n/\log n$ edges that can be partitioned into $\log n$-sized components disconnected from each other, the quantum query complexity of finding a k-cycle does not depend much on whether the partition is known in advance or not.*

An interpretation of the conjecture is that $Q(k\text{-CYCLE})$ should not decrease under certain reductions of the input alphabet. Interestingly, statements of a similar flavor are known to hold, see [25].

References

[1] Frédéric Magniez, Miklos Santha, and Mario Szegedy. Quantum algorithms for the triangle problem. *SIAM Journal on Computing*, 37(2):413–424, 2007.

[2] Aleksandrs Belovs. Span programs for functions with constant-sized 1-certificates. In *Proceedings of the 44th ACM Symposium on Theory of Computing (STOC)*, pages 77–84, 2012.

[3] Stacey Jeffery, Robin Kothari, and Frédéric Magniez. Nested quantum walks with quantum data structures. In *Proceedings of the 24th ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 1474–1485, 2013.

[4] Troy Lee, Frédéric Magniez, and Miklos Santha. Improved quantum query algorithms for triangle finding and associativity testing. In *Proceedings of the 24th ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 1486–1502, 2013.

[5] François Le Gall. Improved quantum algorithm for triangle finding via combinatorial arguments. In *Proceedings of the 55th IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 216–225, 2014.

[6] Zhiying Yu and Shalev Ben-David. Quantum algorithms for hypergraph simplex finding, 2024.

[7] Aleksandrs Belovs. Learning-graph-based quantum algorithm for k-distinctness. In *Proceedings of the 53rd IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 207–216, 2012.

[8] Aleksandrs Belovs and Robert Spalek. Adversary lower bound for the k-sum problem. In *Proceedings of the 4th Innovations in Theoretical Computer Science Conference (ITCS)*, pages 323–328, 2013.

[9] Aleksandrs Belovs, Andrew M. Childs, Stacey Jeffery, Robin Kothari, and Frédéric Magniez. Time-efficient quantum walks for 3-distinctness. In *Proceedings of the 40th International Colloquium on Automata, Languages, and Programming (ICALP)*, pages 105–122, 2013.

[10] Aleksandrs Belovs and Ansis Rosmanis. On the power of non-adaptive learning graphs. *computational complexity*, 23(2):323–354, 2014.

[11] David Wagner. A generalized birthday problem. In *Advances in Cryptology — CRYPTO 2002: 22nd Annual International Cryptology Conference*, pages 288–304, 2002.

[12] Qipeng Liu and Mark Zhandry. On finding quantum multi-collisions. In *Advances in Cryptology – EUROCRYPT 2019*, pages 189–218, 2019.

[13] Mark Zhandry. How to record quantum queries, and applications to quantum indifferentiability. In *Advances in Cryptology — CRYPTO 2019: 39th Annual International Cryptology Conference*, pages 239–268, 2019.

[14] Yassine Hamoudi and Frédéric Magniez. Quantum time–space tradeoff for finding multiple collision pairs. *ACM Transactions on Computation Theory*, 15(1–2):1–22, 2023.

[15] Kai-Min Chung, Serge Fehr, Yu-Hsuan Huang, and Tai-Ning Liao. On the compressed-oracle technique, and post-quantum security of proofs of sequential work. In *Advances in Cryptology – EUROCRYPT 2021*, pages 598–629, 2021.

[16] Paul Beame, Niels Kornerup, and Michael Whitmeyer. Quantum time-space tradeoffs for matrix problems. In *Proceedings of the 56th ACM Symposium on Theory of Computing (STOC)*, pages 596–607, 2024.

[17] Christian Majenz, Giulio Malavolta, and Michael Walter. Permutation superposition oracles for

quantum query lower bounds. In *Proceedings of the 57th ACM Symposium on Theory of Computing (STOC)*, page 1508–1519, 2025.

[18] Fermi Ma and Hsin-Yuan Huang. How to construct random unitaries. In *Proceedings of the 57th ACM Symposium on Theory of Computing (STOC)*, page 806–809, 2025.

[19] Mark Bun, Robin Kothari, and Justin Thaler. The polynomial method strikes back: tight quantum query bounds via dual polynomials. In *Proceedings of the 50th ACM Symposium on Theory of Computing (STOC)*, pages 297–310, 2018.

[20] Nikhil S. Mande, Justin Thaler, and Shuchen Zhu. Improved approximate degree bounds for k-distinctness. In *Proceedings of the 15th Conference on the Theory of Quantum Computation, Communication and Cryptography (TQC)*, volume 158, pages 2:1–2:22, 2020.

[21] Robert Spalek. Adversary lower bound for the orthogonal array problem, 2013.

[22] Ansis Rosmanis. Adversary lower bound for element distinctness with small range, 2014.

[23] Scott Aaronson and Yaoyun Shi. Quantum lower bounds for the collision and the element distinctness problems. *Journal of the ACM*, 51(4):595–605, 2004.

[24] Andris Ambainis. Quantum walk algorithm for element distinctness. *SIAM Journal on Computing*, 37(1):210–239, 2007.

[25] Andris Ambainis. Polynomial degree and lower bounds in quantum complexity: Collision and element distinctness with small range. *Theory of Computing*, 1(3):37–46, 2005.