Limits of Quantum Speed-Ups for Computational Geometry and Other Problems: Fine-Grained Complexity via Quantum Walks

Harry Buhrman
QuSoft, CWI Amsterdam, The Netherlands
University of Amsterdam, The Netherlands

Bruno Loff
University of Porto, Portugal
INESC-Tec, Porto, Portugal

Subhasree Patro
QuSoft, CWI Amsterdam, The Netherlands
University of Amsterdam, The Netherlands

Florian Speelman
QuSoft, CWI Amsterdam, The Netherlands
University of Amsterdam, The Netherlands

Abstract

Many computational problems are subject to a quantum speed-up: one might find that a problem having an $O(n^3)$-time or $O(n^2)$-time classic algorithm can be solved by a known $O(n^{1.5})$-time or $O(n)$-time quantum algorithm. The question naturally arises: *how much quantum speed-up is possible?*

The area of fine-grained complexity allows us to prove optimal lower-bounds on the complexity of various computational problems, based on the conjectured hardness of certain natural, well-studied problems. This theory has recently been extended to the quantum setting, in two independent papers by Buhrman, Patro and Speelman [7], and by Aaronson, Chia, Lin, Wang, and Zhang [1].

In this paper, we further extend the theory of fine-grained complexity to the quantum setting. A fundamental conjecture in the classical setting states that the 3SUM problem cannot be solved by (classical) algorithms in time $O(n^{2-\varepsilon})$, for any $\varepsilon > 0$. We formulate an analogous conjecture, the Quantum-3SUM-Conjecture, which states that there exist no sublinear $O(n^{1-\varepsilon})$-time quantum algorithms for the 3SUM problem.

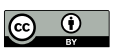
Based on the Quantum-3SUM-Conjecture, we show new lower-bounds on the time complexity of quantum algorithms for several computational problems. Most of our lower-bounds are optimal, in that they match known upper-bounds, and hence they imply tight limits on the quantum speedup that is possible for these problems.

These results are proven by adapting to the quantum setting known classical fine-grained reductions from the 3SUM problem. This adaptation is not trivial, however, since the original classical reductions require pre-processing the input in various ways, e.g. by sorting it according to some order, and this pre-processing (provably) cannot be done in sublinear quantum time.

We overcome this bottleneck by combining a quantum walk with a classical dynamic data-structure having a certain "history-independence" property. This type of construction has been used in the past to prove upper bounds, and here we use it for the first time as part of a reduction. This general proof strategy allows us to prove tight lower bounds on several computational-geometry problems, on Convolution-3SUM and on the 0-Edge-Weight-Triangle problem, conditional on the Quantum-3SUM-Conjecture.

We believe this proof strategy will be useful in proving tight (conditional) lower-bounds, and limits on quantum speed-ups, for many other problems.

2012 ACM Subject Classification Theory of computation → Problems, reductions and completeness; Theory of computation → Quantum complexity theory



13th Innovations in Theoretical Computer Science Conference (ITCS 2022).
Editor: Mark Braverman; Article No. 31; pp. 31:1–31:12

Leibniz International Proceedings in Informatics
Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

Keywords and phrases complexity theory, fine-grained complexity, 3SUM, computational geometry problems, data structures, quantum walk

Digital Object Identifier 10.4230/LIPIcs.ITCS.2022.31

Related Version *Full Version*: https://arxiv.org/abs/2106.02005

Funding Harry Buhrman, Subhasree Patro, and Florian Speelman are additionally supported by NWO Gravitation grants NETWORKS and QSC, and EU grant QuantAlgo.
Bruno Loff: Bruno Loff's research is supported by National Funds through the Portuguese funding agency, FCT - Fundação para a Ciência e a Tecnologia, within project UIDB/50014/2020.
Subhasree Patro: Subhasree Patro is supported by the Robert Bosch Stiftung.

1 Introduction

The world is investing in quantum computing because of so-called *quantum speed-ups*: quantum algorithms can solve many computational problems faster than their classical counterparts. However, the amount of speed-up that is possible varies among different computational problems. It is expected that quantum computers will remain an expensive resource for a long time, and the extent to which a quantum speed-up is possible, or not possible, may one day be a key factor in deciding whether or not to invest in the use of a quantum computation, for example in an industrial setting.

Such a consideration is not a mere abstraction. It is folk knowledge among researchers in quantum error-correction that current error-correction techniques are so costly, meaning the constant-factor overhead they impose is so great, that quadratic quantum speed-ups will offer no advantage, when compared to their classical counterparts. Babbush et al. [4] consider N-qubit quantum algorithms that work by making quantum calls to certain primitives, so that a quantum algorithm will do, e.g., M calls, and the corresponding classical algorithm will do M^2 calls, to the same primitive. Here $M \gg N \approx 100$ (so, e.g., we might using Grover to search for a satisfying assignment to a CNF with roughly 100 variables). They then estimate how large M must be, in order for quantum computers to offer a significant advantage over their classical counterparts, and conclude:

> *[We find that, even when] using state-of-the-art surface code constructions under a variety of assumptions, (...) quadratic speedups will not enable quantum advantage on early generations of fault-tolerant [quantum computers] unless there is a significant improvement in how we would realize quantum error-correction.*

It can further be said that the estimates appearing in [4] are extremely generous on the quantum side, in many respects. So, even allowing for incremental improvements to current quantum error correction, improvements in qubit technology, and so forth, this *uselessness* of quadratic quantum speedups is likely to assert itself in practice, for decades to come.

It is therefore essential to understand how much quantum speed-up is possible for specific computational problems (for example, so as not to overstate the potential of early-generation quantum computers). For this purpose we would need to have tight upper and lower-bounds on both classical and quantum algorithms.

Sadly, the state of affairs is such that we do not even know how to prove super-linear time lower-bounds (e.g., on a classical random-access machine). Hence, there are some computational problems which *do* have polynomial-time (e.g. quadratic-time) algorithms, classical or quantum, and these algorithms are conjectured to be optimal, but we presently have no way of proving this.

The theory of fine-grained complexity has been developed in the last decade to overcome this problem. Analogous to how NP-completeness allows us to prove super-polynomial lower-bounds, fine-grained complexity allows us to prove tight fixed-polynomial (e.g. quadratic) lower-bounds on the time complexity of many problems in P, conditioned on hardness conjectures for a few natural, well-studied problems. Three central hardness conjectures are the strong exponential-time hypothesis (SETH) for satisfiability, a conjectured cubic-hardness for the all-pairs shortest-path problem (APSP) and a conjectured quadratic hardness for the 3SUM problem.[1]

Recently, two independent works initiated the study of fine-grained complexity in the quantum setting. Both works studied quantum variants of SETH, and used these variants to prove (often tight) bounds on how much quantum speed-up is possible for various problems. Aaronson, Chia, Lin, Wang, and Zhang [1] presented linear quantum time lower bounds for Closest Pair, Bichromatic Closest Pair, and Orthogonal Vectors, conditioned on the quantum hardness of the Satisfiability (SAT) problem. In the same paper, they also present matching quantum upper bounds for these problems. Simultaneously, Buhrman, Patro, and Speelman [7] presented a framework for proving quantum time lower bounds for many problems in P conditioned on quantum hardness of variants of SAT, which they used to prove an $n^{1.5}$ quantum time lower bound for the Edit Distance and the Longest Common Subsequence problems.

In this work, we explore quantum fine-grained reductions to derive quantum time lower-bounds for several problems in P, conditioned on a natural, conjectured quantum hardness for the 3SUM problem. These lower-bounds will often tightly match upper-bounds given by known quantum algorithms, and similar tight upper and lower-bounds have also been proven in the classical setting. Together, these tight classical and quantum bounds are finally able to tell us exactly how much quantum speed-up is possible for various problems, which is the main goal of this line of research. For the problems we study, we will conclude that a quadratic speed-up is, in fact, the best possible.

1.1 The conjectured hardness of 3SUM

The 3SUM problem is defined as follows: We are given as input a list S of n integers, which we may assume to be between $-n^3$ and n^3,[2] and we wish to know if there exist a, b, c in S such that $a + b + c = 0$. There is a simple classical algorithm that solves this problem in $\tilde{O}(n^2)$ time which is as follows: Use $O(n \log n)$ time to make a *sorted* copy of the input S, let's denote by S'. Then go over n^2 pairs $(a, b) \in S \times S$ and search if $-(a + b)$ is present in S' using binary search. This algorithm takes $O(n \log n) + n^2 \log n = \tilde{O}(n^2)$ time[3], but even after many years of interest in the problem, the exponent has not been reduced. The conjecture naturally arises that there is no $\epsilon > 0$, such that 3SUM can be solved in $O(n^{2-\epsilon})$ classical time. We refer to this conjecture as the Classical-3SUM-Conjecture. Using this conjecture, one can derive conditional classical lower bounds for a vast collection of computational geometry problems, dynamic problems, sequence problems, etc. [10, 14, 13, 15].

However, the Classical-3SUM-Conjecture no longer holds true in the quantum setting, as there is a faster *quantum* algorithm for 3SUM: we may use Grover search as a subroutine in the $\tilde{O}(n^2)$ classical algorithm to solve the problem in $\tilde{O}(n)$ quantum time. Apart from this

[1] The survey by Vassilevska Williams contains an overview of many results within this area [15].
[2] This is because the 3SUM problem over lists with larger integers can be reduced to the 3SUM problem on n^3-bounded integers by a simple hashing technique.
[3] Note that there is also a classical algorithm that solves 3SUM in $O(n^2)$ time.

quadratic speedup, no further improvement to the quantum-time upper-bound is known. It is worth mentioning that there is a sub-linear $O(n^{3/4})$ quantum *query* algorithm for computing 3SUM [2, 8] – with a matching lower bound of $\Omega(n^{3/4})$ [5] – this query algorithm however, is not time efficient. Consequently, it was conjectured [3] that the 3SUM problem cannot be solved in sub-linear quantum time in the QRAM model:

▶ **Conjecture 1** (Quantum-3SUM-Conjecture [3]). *There does not exist a $\delta > 0$ such that 3SUM on a list of n integers can be solved in $O(n^{1-\delta})$ quantum time in the QRAM model.*

It is then natural to try to extend the classical 3SUM-based lower bounds to the quantum setting, and one may at first expect this task to be a simple exercise. However, one soon realizes that none of the existing classical reductions can be easily adapted to the quantum regime. Indeed, most of the existing classical reductions begin by pre-processing the input in some way, e.g., by sorting it according to some ordering, and this pre-processing turns out to be essential for the reduction to work efficiently. This is not an issue in the classical setting, as the classical conjectured lower bound for 3SUM is quadratic. Hence, the classical reductions can accommodate any pre-processing of the input that takes sub-quadratic time, such as e.g. sorting. However, this pre-processing becomes problematic in the quantum setting, since here we will need a sublinear-time quantum reduction, and even simple sorting requires linear quantum time on a quantum computer [12].

We present a workaround for this problem. The idea of the proof is to adapt Ambainis' quantum walk algorithm for element distinctness [2]. For example, to enable reductions that need a sorted input, then instead of having the reduction sort the entire list, we combine a data structure for dynamic sorting together with a quantum walk algorithm. As we will show, this approach only needs the reduction to sort a small part of the input and allows us to show that 3SUM remains hard, even when the entire input is sorted. As we will see, this idea can be extended to allow for any "structuring" of the input (not just sorting) which can be implemented by a dynamic data structure obeying a certain "history-independence" property. The proof will be sketched in Section 1.2.

This *quantum-walk plus data-structure* proof strategy has been used to prove upper-bounds on other problems (e.g., for the closest-pair problem [1]), and here we use it for the first time as part of a reduction in order to obtain a lower-bound. We expect that the same strategy will be applicable to other quantum fine-grained reductions, and our hope is that this will give rise to a landscape of results, that establish (conditional) tight lower-bounds for quantum algorithms. This, in turn, will precisely answer the question of how much quantum speed-up is possible for a variety of computational problems.

Using this strategy we are able to show that various "structured" versions of 3SUM are as hard as the original (unstructured) 3SUM problem, even in the quantum case. Once we have shown that these structured versions of 3SUM are hard, we may then construct direct quantum adaptations of the classical reductions, to show the quantum hardness of several computational-geometry problems, of Convolution-3SUM and of the 0-Edge-Weight-Triangle problem. This enables us to prove quantum time lower-bounds for these problems, conditioned on the Quantum-3SUM-Conjecture. All these results are formally stated and proven in Sections 3,4,5 of the full version of our paper [6].

1.2 Main Idea: Reductions via Quantum Walks

The Classical-3SUM-Conjecture states that there is no sub-quadratic classical algorithm to solve the 3SUM problem. However, the statement of this conjecture can be shown to be equivalent to the same statement for a promise version of 3SUM where the input S is

sorted. That is because, if there was a sub-quadratic algorithm for 3SUM on sorted inputs, then given any (unsorted) input one can first sort the entire input with additional $O(n \log n)$ pre-processing time, and then use the sub-quadratic algorithm for sorted 3SUM, resulting in a sub-quadratic algorithm for unsorted 3SUM. In fact, one can make a more general statement in this regard. An input to the 3SUM problem is a list $S \in \{-n^3, \ldots, n^3\}^n$ of n integers (possibly with repetitions). One may consider a family $\{q_i\}$ of *queries*, i.e., each $q_i : \{-n^3, \ldots, n^3\}^n \to A_i$ is a function on lists of integers, for some set A_i of possible answers to the query. (For example, $q_i(S)$ could output the i-th smallest integer in S.) We may then ask about static data-structures that allow us to efficiently answer these queries. (For example, we may consider the sorted version of S to be a data-structure that allows us to efficiently obtain the i-th smallest element of S.) Then we may generally state that, if it is possible to preprocess an input S in sub-quadratic time to produce a static data-structure that allows us to answer any query in $n^{o(1)}$ time, then the "structured" variant of Classical-3SUM-Conjecture, where we give the algorithm access to all the queries $q_i(S)$ for free, is equivalent to the original version of Classical-3SUM-Conjecture.

Most of the known fine-grained reductions from 3SUM, in the classical setting, can be explained in the following way: one first shows that a certain "structured" variant of 3SUM is just as hard as the original 3SUM problem, and then one reduces the structured variant of 3SUM to another problem. While for some reductions [10] require the input list to be sorted in the usual order of the integers, other reductions require the input to be structured in some other way, for example, reductions in [13, 14] require that the elements are hashed into buckets and every element in the bucket can be accessed efficiently.

The reduction from "unstructured" to "structured" 3SUM is usually trivial to do in classical sub-quadratic time, but not so in quantum sub-linear time (e.g., a quantum computer cannot sort in sublinear time [12]). This is the main difficulty in translating the classical reductions to the quantum setting.

Our main observation is that, if a certain analogous *dynamic* data-structure problem can be solved efficiently by a dynamic data-structure possessing a certain "history-independence" property, then it is possible to use a quantum walk in order to show that the "structured" variant of Quantum-3SUM-Conjecture, where we give the algorithm access to the queries for free, is equivalent to the original unstructured version of the Quantum-3SUM-Conjecture.[4] It is this insight that underlies all of our reductions, and which we expect will open up the way to many other fine-grained reductions in the quantum setting.

One might informally state our observation as follows.

[informal] Let $\{q_i\}$ be a collection of queries over 3SUM inputs, i.e., each q_i is a function over inputs $S \in \{-n^3, \ldots, n^3\}^n$ for 3SUM. Suppose that there exists an efficient classical dynamic data-structure that allows us to answer the queries q_i, under updates to S, where an update consists of replacing an element in the list S by a different element. By efficient we mean that any query or update can be carried out in $n^{o(1)}$ time. Suppose further that the dynamic data structure satisfies a certain "history-independence" property[5], which means that the data structure corresponding to each set S has a unique representation in memory, which only depends on the current value of S (so it is independent of the initial value of S, and of the subsequent updates which resulted in the current value of S).

[4] The *history-independence* is necessary to achieve the appropriate amplitude amplification/cancellation in the quantum walk: if two update histories for the data structure (e.g. insertions/removals) lead to the same data contents (e.g. same list), then there should be a single basis state that represents the result.

[5] Also mentioned in [2, 1]

Then, conditioned on the Quantum-3SUM-Conjecture, 3SUM cannot be done in $O(n^{1-\varepsilon})$ quantum time, for any $\varepsilon > 0$, even if the queries $q_i(S)$ can be done at unit cost.

Hereafter, we refer to these versions of 3SUM, where queries $q_i(S)$ have unit cost, as "structured" versions of the 3SUM problem. To be clear, by *being able to do the queries at unit cost*, we mean that the algorithm is given access to an oracle gate, implementing the unitary transformation:

$$|i, b\rangle \mapsto |i, b \oplus q_i(S)\rangle.$$

The distinction between an arbitrary dynamic data-structure and a history-independent solution should be understood as follows. Generally speaking, a solution to a dynamic data-structure problem could represent data in a way which depends on the specific sequence of updates which were applied to the initial data. For example, self-balancing trees are a solution to the dynamic sorting problem, but the specific balancing of the tree which is kept in memory depends on the sequence of updates which were applied, so different sequences of insertions and deletions might lead to the same list, but will nonetheless be represented differently in memory. A history-independent data-structure, however, has fixed a-priori representations for each possible data value. So, for example, in the dynamic sorting problem, a history-independent data-structure must represent each possible list in a unique, or canonical way in memory.

Our idea

Let $S = (x_1, \ldots, x_n)$ be an unstructured input to 3SUM. We will now discuss quantum query algorithms for solving 3SUM. Such algorithms can access the input only via a unitary $|i, b\rangle \mapsto |i, b \oplus x_i\rangle$. Each application of this unitary is called a *query*. But, in accordance to data-structure nomenclature, we have also called *queries* to the functions q_i. So to distinguish the two, in this section we will use *input queries* to refer to queries to the input, in the sense of query complexity, and let us use *data-structure queries*, to refer to the values $q_i(S)$.

Consider the quantum walk algorithm for Element Distinctness by Ambainis [2]. It was observed by Childs and Eisenberg [8] that this algorithm can be used to solve any problem, such as 3SUM, where we wish to find a constant-size subset that satisfies a given property. Although this algorithm is optimal and sub-linear for 3SUM when we only measure the number of input queries (it uses $\Theta(n^{3/4})$ input queries, and this is required [5]), the algorithm still requires linear time, essentially because an $\Omega(n^{1/4})$-time operation is performed between each input query.

This optimal query algorithm for 3SUM is a quantum walk on the *Johnson graph*, namely, the graph of $\binom{n}{r}$ vertices with each vertex of the graph labelled by an r-sized subset of $[n]$, and where there is an edge between two vertices if and only if the two corresponding sets differ by exactly two elements. This resulting graph $J(n, r)$ is a good-enough expander, so that a quantum walk will be able to find an r-sized subset of $[n]$ containing indices to three elements of S that sum to zero, in queries sublinear in n.[6] To do so, the quantum-walk algorithm maintains the list of values $(x_{i_1}, \ldots, x_{i_r})$ entangled together with the basis state representing the current r-sized subset $\{i_1, \ldots, i_r\} \subseteq [n]$ that is being traversed. Using this list of values, as a part of the quantum walk algorithm, a subroutine checks (in superposition) if there is a 3SUM solution in $(x_{i_1}, \ldots, x_{i_r})$. While this step requires no additional input queries, so the total number of input queries is $O(n^{3/4})$, the actual implementation of this subroutine requires a significant amount of time (namely time $r = \Omega(n^{1/4})$), which then makes the resulting quantum walk algorithm for 3SUM linear, at best.

[6] For an excellent introduction to quantum walks, see Chapter 8 of Ronald de Wolf's lecture notes [9].

It is this subroutine, i.e. the subroutine that checks for a 3SUM solution in the r-sized set of values, that we would like to further speed up. Now suppose that we had a faster-than-linear algorithm for a "structured" version of 3SUM. I.e., the algorithm works in sublinear time, provided it is given certain data-structure queries $q_i(S)$ as part of the input. Now, if we could efficiently answer these data-structure queries at any point during the entire quantum walk, then we could use this faster-than-linear algorithm to speed-up the subroutine. To do so, we need a dynamic data structure that allows us to efficiently answer the data-structure queries, under the kind of updates that are required at each step of the quantum walk. For the quantum walk on the Johnson graph, each update corresponds to replacing a single element in the list of values $(x_{i_1}, \ldots, x_{i_r})$.

An important detail remains: in order for the quantum walk to work, it is necessary that there is a unique basis state corresponding to each node in the quantum-walk graph (otherwise we won't have the desired amplitude interference). It is for this reason that the dynamic data-structure structure is required to have a history-independence property.

Proof of Theorem 1.2 (sketch). In order to prove this theorem, we will first go through the steps of the more general version of Ambainis' quantum walk algorithm for Element Distinctness given by [8].

Let $S \in \{-n^3, \ldots, n^3\}^n$ be an input to the 3SUM problem. Let $r = n^\beta$ for some $\beta \in (0, 1)$ which will be fixed later (so that r is an integer). The graph G used in Ambainis' construction is a Johnson graph $J(n, r)$ with vertices all labelled by r-sized subsets of $[n]$. Let $V, V' \subset [n]$ with $|V| = |V'| = r$. Vertices labelled by V and V' are connected if and only if $|V \cap V'| = r-1$, i.e., V' can be obtained by replacing a single element of V.

Given a subset $I \subset [n]$, we use $S[I]$ to denote all the elements $S[i], i \in I$. Now suppose we have a history-independent classical dynamic data structure for answering a family of data-structure queries $\{q_i\}$, where each $q_i : \{-n^3, \ldots, n^3\}^r$. For $V \subseteq [n]$ of size $|V| = r$, let $D(S[V])$ denote the (unique) state of the data-structure corresponding to $S[V]$. I.e., given $D(S[V])$, we are able to answer any query $q_i(S[V])$ in time $n^{o(1)}$. And if we change V to V' by replacing a single element of V, we are able to update $D(S[V])$ to $D(S[V'])$, also in time $n^{o(1)}$.

To define a quantum walk on G, define an orthonormal basis of quantum states $|V\rangle$, one for each r-subset V. We start with creating a uniform superposition over all the vertices of the Johnson graph $J(n, r)$:

$$|s_0\rangle = \frac{1}{\sqrt{c}} \sum_{|V|=r, V\subseteq[n]} |V\rangle \sum_{k \notin V} |k\rangle, \tag{1}$$

with $c = (n-r)\binom{n}{r}$ being the normalization constant.[7,8]

[7] Refer to the circuit construction in András Pál Gilyén's Master's thesis [11] for creating a uniform superposition over all the vertices of Johnson Graph $J(n, r)$. This construction uses $\widetilde{O}(r)$ elementary quantum gates in total and their results extend for any $r = n^\beta$ with $0 < \beta < 1$.

[8] Note that, the circuit construction that we refer to creates a uniform superposition of vertices in $J(n, r)$ with the vertices represented in $O(r \log n)$ sized array of qubits. This representation of vertices do not allow for time efficient insertions and deletions. Therefore, we first encode all the vertices V (in superposition) in a data structure similar to the data structure we use to store the query values, so that the (walk) updates on the states representing the vertices also occur time efficiently. In our paper, we use V to denote a vertex already encoded in the data structure. In the full version of the paper [6], we show that such encoding procedures exist and are (almost) linear (reversibly as well), i.e. run in $\widetilde{O}(r)$ time where r is the size of the set that is being encoded.

The key idea is to store values from the list, and the contents of the data-structure, along with the subset V. So the full quantum state has the form $|V, D(S[V]), k\rangle$ where $k \in [n]$. If $|V| = r$ then k denotes an element in $[n]\backslash V$ to be added to V. We say a vertex V is marked if $S[V]$ is a positive 3SUM instance (of smaller size), i.e., if there are $p, q, r \in V$ such that $S[p] + S[q] + S[r] = 0$.

The quantum walk algorithm is analogous to Grover's algorithm, where the aim is to make the amplitude on marked vertices large enough that with very high probability[9] the final measurement collapses on a marked vertex, i.e., a vertex labelled by an r-subset that contains a solution to 3SUM problem. The algorithm starts with a state

$$|s\rangle = \frac{1}{\sqrt{c}} \sum_{|V|=r} |V, D(S[V])\rangle \sum_{k \notin V} |k\rangle, \tag{2}$$

which is a uniform superposition of all the states on subsets of size r and $c = (n-r)\binom{n}{r}$ is the normalization constant.

There are two main operations in this algorithm: A walk operation U_{walk} and a phase flip operation $U_{phaseFlip}$ which is

$$U_{phaseFlip}|V, D(S[V])\rangle = \begin{cases} -|V, D(S[V])\rangle & \text{if } V \text{ is marked} \\ |V, D(S[V])\rangle & \text{if } V \text{ is not marked.} \end{cases} \tag{3}$$

The full algorithm is $(U_{walk}^{t_1} U_{phaseFlip})^{t_2}$ where $t_1 = O(\sqrt{r})$ and $t_2 = O((n/r)^{1.5})$. The total time taken by the algorithm is

$$T_{setup}(|s\rangle) + t_1 \cdot t_2 \cdot T_{unitary}(U_{walk}) + t_2 \cdot T_{unitary}(U_{phaseFlip}), \tag{4}$$

where $T_{setup}(|s\rangle)$ denotes the time taken to setup the initial state $|s\rangle$ that also includes the time taken to query values of the subset of indices of size r. The term $T_{unitary}(U)$ denotes the number of elementary gates required to implement a unitary U.

In the setup phase, for every vertex V we initialize the dynamic data-structure corresponding to $S[V]$. We may think of $S[V]$ as obtained via the $(0, \ldots, 0)$ list by updating each position i with $S[i]$. Hence, the setup time for each vertex, which consists of computing $D(S[V])$ for all V in superposition, is at most $rn^{o(1)}$.

Now, because the data structure supports efficient updates, the U_{walk} unitary can be implemented in time $n^{o(1)}$. It in is this U_{walk} operation that an element is inserted and some other element is deleted, hence it is sufficient that the dynamic data structure supports replacement of values.

The unitary $U_{phaseFlip}$ in Equation 3 adds a negative phase to the marked states and none to the unmarked states, which means $U_{phaseFlip}$ implements a subroutine that checks whether or not a vertex V is marked by going through its input-query values $S[V]$ and checking if there is a 3SUM solution present in $S[V]$. Currently, there is no known (time) efficient method to implement this subroutine.[10]

Instead, suppose that there exists a constant $\alpha > 0$ such that there is a subroutine that can solve this structured version of 3SUM on r elements in $O(r^{1-\alpha})$ quantum time. We can now implement $U_{phaseFlip}$ in the following way. Call the subroutine that is optimal for solving 3SUM on this of ordered input. The data-structure queries $q_i(S[V])$ to the structured

[9] Throughout the paper we say that something holds "with high probability" if it holds with probability at least $1 - o(1)$.

[10] One would require a dynamic data-structure for efficiently answering 3SUM queries, which is not known to exist.

input can be simulated with an $n^{o(1)}$ overhead in time, because the data structure $D(S[V])$ supports efficient data-structure queries. The time complexity of the original Ambainis' walk algorithm then becomes

$$r \cdot n^{o(1)} + t_1 \cdot t_2 \cdot n^{o(1)} + t_2 \cdot n^{o(1)} \cdot r^{1-\alpha}, \tag{5}$$

which, after ignoring all the $n^{o(1)}$ factors, becomes

$$r + t_1 t_2 + t_2 r^{1-\alpha}. \tag{6}$$

Substituting the values of $t_1 = O(\sqrt{r})$ and $t_2 = O((n/r)^{1.5})$ the total time taken in Equation 6 roughly becomes

$$r + \frac{n^{1.5}}{r} + \frac{n^{1.5}}{r^{1.5}} \cdot r^{1-\alpha}. \tag{7}$$

Given that $r = n^\beta$ for a $\beta \in (0,1)$, it is easy to see that for every $0 < \alpha < 1$, there exists a β such that $\max(\frac{1}{2}, \frac{1}{2\alpha+1}) < \beta < 1$, and then the value of (7) becomes strictly sublinear. It then follows that there is no sub-linear quantum time algorithm for solving the structured version of 3SUM, unless Quantum-3SUM-Conjecture is false. ◀

We have omitted several details from the above proof sketch. One omission is that we neglected to account for the error (in the quantum walk and in the invoked subroutine for 3SUM). This is simple to account for and we will do so in Section 3.1 of the full version of our paper [6]. The most crucial omission is that we will actually require *probabilistic* dynamic data-structures in our reductions. Randomness seems to be required because no dynamic sorting data-structure is known that is simultaneously time-efficient, space-efficient, history-independent, and deterministic. However, a solution exists if any of these four requirements is removed. We will first (in Section 3.1 of [6]) present a solution which uses a deterministic data-structure, but large space, and then (in Section 3.2 of [6]) a probabilistic solution which is also efficient in space. It is an interesting open question in classical data-structures to provide, or disprove the existence of, a dynamic data-structure that simultaneously satisfies all four requirements.[11]

1.3 Applications

We use our proof strategy to show, conditional on Quantum-3SUM-Conjecture, tight lower-bounds on several computational-geometry problems, on Convolution-3SUM, and on the 0-Edge-Weight-Triangle problem. Our lower-bounds show that the quantum speed-up is at most quadratic for all of these problems.

Our lower-bounds on Convolution-3SUM and 0-Edge-Weight-Triangle tightly match the Grover-based speed-up that quantum algorithms can get for these problems.

Our quantum reductions from 3SUM to computational-geometry problems are complementary to a recent paper by Ambainis and Larka [3], where they present quantum speed-ups for several such problems. Our results show, under the Quantum-3SUM-Conjecture, that all

[11] The question might arise: *why do we care for the structure to be space efficient?* This is for two reasons. On the one hand, we expect memory to be an expensive resource for quantum computers, so algorithms using a large amount of memory, even in regimes that are practical classically, might never be so quantumly. On the other hand, making our reductions space efficient allows us to weaken the Quantum-3SUM-Conjecture to say that no space-efficient quantum algorithm can solve 3SUM in sublinear time. We do not explicitly state this outside of this footnote, but all the lower-bounds in this paper also follow from this weaker conjecture.

of the speed-ups obtained by Ambainis and Larka are optimal. There are also computational-geometry problems for which the Quantum-3SUM-Conjecture gives us a lower-bound, but for which no quantum speed-up is known.

Table 1 (in page 11) summarizes our results. It also includes the best-known *classical* upper and lower-bounds.

1.4 Future directions and open questions

The study of quantum fine-grained complexity is just beginning. Classically, there are many fine-grained reductions laying out the structure of the class P, but only a few of such reductions have been established for BQP. This forms an appealing avenue for future work, as not only is the topic very much unexplored, any tight lower-bounds given by quantum fine-grained reductions will allow us to understand how much quantum speed-up is possible.

The following is a non-exhaustive list of questions which are currently open, and which we hope will benefit from the approach contained in our paper:

- Table 1 contains four problems for which we can prove some quantum lower-bound, conditioned on the Quantum-3SUM-Conjecture. Is this lower-bound tight, i.e., are there matching algorithms? Or can we prove a higher lower-bound, perhaps based on a different conjecture?
- In the classical setting, there are problems, other than 3SUM, which serve as a basis for fine-grained reductions, e.g. the Orthogonal Vectors problem, the all-pairs shortest-path problem [15]. What lower-bounds can we prove in the quantum setting, based on these problems? Can we prove tight bounds on quantum speed-ups?
- The Classical-3SUM-Conjecture itself gives various other lower-bounds in the classical setting, which we did not study in the quantum setting, namely lower-bounds against dynamic data-structure problems. Can these lower-bounds be proven in the quantum regime, also?
- More generally, for what other problems can we prove that the known quantum speed-up is optimal, under a reasonable hardness hypothesis such as the Quantum-3SUM-Conjecture?

The various papers using dynamic data-structures in quantum walks, including [2, 1] and our paper, give rise to an interesting question in classical data-structures. The vast majority of space-efficient dynamic data-structures are not history-independent: history-independence is a feature which cannot be properly motivated if one is only interested in classical algorithms, but which is fundamentally necessary for using the dynamic data-structure as part of a quantum walk. One can then attempt to understand for which problems do history-independent, memory and time-efficient dynamic data-structures exists. For sorting, the only known solution (skip lists) is randomized. Is this necessary? More generally, what dynamic data-structure problems have solutions that are simultaneously deterministic, time-efficient, space-efficient, and history-independent? Can we prove lower-bounds against data-structures obeying all four criteria simultaneously, which we cannot prove against data-structures obeying only three among the four criteria?

1.5 Full version of the paper

All the main theorems and their respective proofs are presented in the full version of this paper [6] whose structure is as follows. In Section 2.1 of [6] we describe our model of computation, and in Section 2.2 of [6] we describe various simple variants of the 3SUM problem and show that the Quantum-3SUM-Conjecture is equivalent for these versions. (These are not the structured versions we mentioned earlier, here the proof of equivalence is very simple.)

Using the approach we sketched above (in Section 1.2), we proceed to give a full proof that, under the Quantum-3SUM-Conjecture, two "structured" variants of 3SUM also require $\Omega(n)$ time on a quantum computer. We give two separate proofs: The first proof (in Section 3.1 of [6]) uses a deterministic data structure which is *space-inefficient*, and the second proof (in Section 3.2 of [6]) uses a probabilistic data structure which is space-efficient. As direct implications of these hardness results, in Section 4 of [6] we present conditional quantum time lower bounds for several computational geometry problems.

Lastly, in Section 5 of [6], we present conditional quantum time lower bound for Convolution-3SUM and 0-Edge-Weight-Triangle problems. This requires us to prove, under the Quantum-3SUM-Conjecture, that a third "structured" variant of 3SUM also requires $\Omega(n)$ time on a quantum computer.

Table 1 This is a summary of all the Quantum-3SUM-hard problems mentioned in this paper, with (almost) matching upper bounds for most of them.

3SUM-based quantum lower-bounds (our results) ↴		**Classical complexity** (∗∗) ↴	
Problems		**Quantum upper-bound**	
GeomBase	$\Omega(n)$	$\widetilde{O}(n)$ (∗)	$\Theta(n^2)$
3-Points-on-Line	$\Omega(n)$	$O(n^{1+o(1)})$ [3]	$\Theta(n^2)$
Point-on-3-Lines	$\Omega(n)$	$O(n^{1+o(1)})$ [3]	$\Theta(n^2)$
Separator	$\Omega(n)$	$O(n^{1+o(1)})$ [3]	$\Theta(n^2)$
Strips-Cover-Box	$\Omega(n)$	$O(n^{1+o(1)})$ [3]	$\Theta(n^2)$
Triangles-Cover-Triangle	$\Omega(n)$	$O(n^{1+o(1)})$ [3]	$\Theta(n^2)$
Point-Covering	$\Omega(n)$	$O(n^{1+o(1)})$ [3]	$\Theta(n^2)$
Visibility-Between-Segments	$\Omega(n)$	$O(n^{1+o(1)})$ [3]	$\Theta(n^2)$
Hole-In-Union	$\Omega(n)$	$O(n^{1+o(1)})$ (†)	$\widetilde{\Theta}(n^2)$
Triangle-Measure	$\Omega(n)$	Open!	$\Theta(n^2)$
Visibility-From-Infinity	$\Omega(n)$	Open!	$\Theta(n^2)$
Visible-Triangle	$\Omega(n)$	$O(n^{1+o(1)})$ (†)	$\Theta(n^2)$
Planar-Motion-Planning	$\Omega(n)$	Open!	$\Theta(n^2)$
3D-Motion-Planning	$\Omega(n)$	Open!	$\Theta(n^2)$
General-Covering	$\Omega(n)$	$O(n^{1+o(1)})$ [3]	$\Theta(n^2)$
Convolution-3SUM	$\Omega(n)$	$O(n)$ (∗)	$\Theta(n^2)$
0-Edge-Weight-Triangle	$\Omega(n^{1.5})$	$O(n^{1.5})$ (∗)	$\Theta(n^3)$

(∗) Using a simple Grover speed-up on the classical algorithm.

(†) Implicit in [3], by using the classical reduction to Triangles-Cover-Triangle and then using the corresponding quantum algorithm.

(∗∗) All upper-bounds are straightforward: For problems like Convolution-3SUM and 0-Edge-Weight-Triangle the best known algorithms use brute force, for the computational-geometry problems, the upper-bounds follow from geometry arguments [10]. All lower-bounds for computational geometry problems are from [10], the lower-bound for Convolution-3SUM follows from [13], and, the lower-bound for 0-Edge-Weight-Triangle follows from [14].

References

1. Scott Aaronson, Nai-Hui Chia, Han-Hsuan Lin, Chunhao Wang, and Ruizhe Zhang. On the quantum complexity of closest pair and related problems. In *Proceedings of the 35th Computational Complexity Conference*, CCC '20, Dagstuhl, DEU, 2020. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik. doi:10.4230/LIPIcs.CCC.2020.16.
2. A. Ambainis. Quantum walk algorithm for element distinctness. In *45th Annual IEEE Symposium on Foundations of Computer Science*, pages 22–31, 2004. doi:10.1109/FOCS.2004.54.
3. Andris Ambainis and Nikita Larka. Quantum Algorithms for Computational Geometry Problems. In Steven T. Flammia, editor, *15th Conference on the Theory of Quantum Computation, Communication and Cryptography (TQC 2020)*, volume 158 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 9:1–9:10, Dagstuhl, Germany, 2020. Schloss Dagstuhl–Leibniz-Zentrum für Informatik. doi:10.4230/LIPIcs.TQC.2020.9.
4. Ryan Babbush, Jarrod R McClean, Michael Newman, Craig Gidney, Sergio Boixo, and Hartmut Neven. Focus beyond quadratic speedups for error-corrected quantum advantage. *PRX Quantum*, 2(1), 2021.
5. Aleksandrs Belovs and Robert Špalek. Adversary lower bound for the k-sum problem. In *Proceedings of the 4th Conference on Innovations in Theoretical Computer Science*, ITCS '13, pages 323–328, New York, NY, USA, 2013. Association for Computing Machinery. doi:10.1145/2422436.2422474.
6. Harry Buhrman, Bruno Loff, Subhasree Patro, and Florian Speelman. Limits of quantum speed-ups for computational geometry and other problems: Fine-grained complexity via quantum walks, 2021. arXiv:2106.02005.
7. Harry Buhrman, Subhasree Patro, and Florian Speelman. A Framework of Quantum Strong Exponential-Time Hypotheses. In Markus Bläser and Benjamin Monmege, editors, *38th International Symposium on Theoretical Aspects of Computer Science (STACS 2021)*, volume 187 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 19:1–19:19, Dagstuhl, Germany, 2021. Schloss Dagstuhl – Leibniz-Zentrum für Informatik. doi:10.4230/LIPIcs.STACS.2021.19.
8. Andrew M. Childs and Jason M. Eisenberg. Quantum algorithms for subset finding. *Quantum Info. Comput.*, 5(7):593–604, November 2005.
9. Ronald de Wolf. Quantum computing: Lecture notes, 2021. arXiv:1907.09415.
10. Anka Gajentaan and Mark H Overmars. On a class of o(n2) problems in computational geometry. *Computational Geometry*, 5(3):165–185, 1995. doi:10.1016/0925-7721(95)00022-2.
11. András Pál Gilyén. Quantum walk based search methods and algorithmic applications. Master's thesis, Budapest University of Technology and Economics, 2014. URL: https://web.cs.elte.hu/blobs/diplomamunkak/msc_mat/2014/gilyen_andras_pal.pdf.
12. Peter Høyer, Jan Neerbek, and Yaoyun Shi. Quantum complexities of ordered searching, sorting, and element distinctness. *Lecture Notes in Computer Science*, pages 346–357, 2001. doi:10.1007/3-540-48224-5_29.
13. Mihai Patrascu. Towards polynomial lower bounds for dynamic problems. In *Proceedings of the Forty-Second ACM Symposium on Theory of Computing*, STOC '10, pages 603–610, New York, NY, USA, 2010. Association for Computing Machinery. doi:10.1145/1806689.1806772.
14. Virginia Vassilevska and Ryan Williams. Finding, minimizing, and counting weighted subgraphs. In *Proceedings of the Forty-First Annual ACM Symposium on Theory of Computing*, STOC '09, pages 455–464, New York, NY, USA, 2009. Association for Computing Machinery. doi:10.1145/1536414.1536477.
15. Virginia Vassilevska Williams. Hardness of easy problems: Basing hardness on popular conjectures such as the strong exponential time hypothesis. *IPEC*, 2015.