# Quantum Algorithms for Path and Cycle Containment Problems

**Arjan Cornelissen**
Simons Institute, University of California Berkeley, CA, USA

**Amin Shiraz Gilani**
QuICS, University of Maryland, College Park, MD, USA

**Subhasree Patro**
Eindhoven University of Technology, The Netherlands

## Abstract

The quantum query complexity of subgraph-containment problems, which ask whether a given subgraph $H$ is present in an input graph $G$, has been the subject of considerable study. This interest stems not only from the natural and well-motivated formulation of these problems, but also from a flurry of novel quantum algorithmic techniques that were developed specifically to solve them. Notably, even for relatively simple subgraphs, such as paths and cycles, a complete understanding of their query complexities remains elusive.

In this work, we consider several variants of path- and cycle-containment problems in the adjacency matrix model, where we search for paths or cycles of constant length $k \in O(1)$. We compare the settings where the graphs are directed or undirected, where the goal is to detect or find the existence of a path/cycle, and where the path/cycle we're looking for has length exactly $k$, or at most $k$. We also consider several promise versions of these problems, where we know beforehand that the input graph has a certain structure. We characterize the relative difficulty of these variants of the path- and cycle-containment problems, by relating them to one another using randomized reductions, and grouping them into several equivalence classes.

When we restrict our attention to path-containment problems, this implies a dichotomy result. Some of the path-containment problems can be solved using a linear number of queries, and all the others are equivalent to one another (and additionally to several cycle-containment problems as well) under randomized reductions and up to constant multiplicative overhead. For the latter equivalence class, we prove a novel quantum-walk-based algorithm that achieves query complexity $\widetilde{O}(n^{3/2-\alpha_k})$, where $\alpha_k \in \Theta(c^{-k})$ and $c = \sqrt{3+\sqrt{17}}/2 \approx 1.33$, beating the previous best upper bound $O(n^{3/2})$ on its query complexity. We also provide a conditional lower bound based on the graph-collision problem, which implies that this equivalence class does not admit linear-query quantum algorithms unless graph collision admits an $O(\sqrt{n})$ query algorithm.

**2012 ACM Subject Classification** Theory of computation → Problems, reductions and completeness

**Keywords and phrases** Quantum algorithms, query complexity, graph problems, fine-grained reductions

**Digital Object Identifier** 10.4230/LIPIcs.APPROX/RANDOM.2026.72

**Category** RANDOM

**Related Version** *Full Version*: `https://arxiv.org/abs/2605.09017`

**Funding** *Arjan Cornelissen*: Supported by a Simons-CIQC postdoctoral fellowship through NSF QLCI Grant No. 2016245.
*Amin Shiraz Gilani*: Acknowledges support from the U.S. Department of Energy, Office of Science, Accelerated Research in Quantum Computing, Fundamental Algorithmic Research toward Quantum Utility (FAR-Qu) and the National Institute of Standards and Technology (NIST).
*Subhasree Patro*: Acknowledges the support from the Dutch Ministry of Education, Culture, and Science through Gravitation project "Challenges in Cyber Security - 024.006.037" for this work.

**Acknowledgements** We would like to thank Andrew Childs and Matthew Coudron for many useful discussions.

# 1 Introduction

The subgraph containment problem asks whether a given (undirected) subgraph $H$ is contained in an (undirected) input graph $G$. In this work, we consider subgraphs $H$ of constant size that are known ahead of time, and input graphs $G$ consisting of $n$ vertices that we can access through adjacency-matrix queries. Now, we wish to decide whether the subgraph $H$ is present in the input graph $G$, while making a minimal number of queries to the adjacency matrix of $G$. We denote this problem concisely by $\texttt{Subgraph}^H$, and the minimal number of queries to the input required to solve it is known as its query complexity. We also consider a restricted version of this problem where we select several vertices $v_1, \dots, v_m$ from $H$ and additionally require that these vertices get mapped to predetermined positions $s_1, \dots, s_m$ in $G$. Under this restriction, we then ask whether $H$ is present in $G$, and we denote the resulting problem by $\texttt{Subgraph}^H_{v_1,\dots,v_m}$.

In the randomized setting, we decide which edges to query by a (classical) randomized algorithm, and we require the resulting algorithm to output the correct answer with probability at least 2/3. The randomized query complexity of the unrestricted subgraph containment problem, denoted by $\mathsf{R}(\texttt{Subgraph}^H)$, satisfies $\Theta(n^2)$ for any non-empty graph $H$. Moreover, it is not hard to show that the restricted problems are classified into three distinct classes, where the randomized query complexities $\mathsf{R}(\texttt{Subgraph}^H_{v_1,\dots,v_m})$ are $\Theta(1)$, $\Theta(n)$ and $\Theta(n^2)$, respectively.

In contrast, in the quantum setting, we can query the adjacency matrix coherently in superposition, and we aim to give a quantum algorithm whose final measurement outputs the right answer with probability at least 2/3. The corresponding quantum query complexities, denoted by $\mathsf{Q}(\texttt{Subgraph}^H)$ for the unrestricted problem, and $\mathsf{Q}(\texttt{Subgraph}^H_{v_1,\dots,v_m})$ for the restricted problem, seem to have a much richer structure than their classical counterparts. This is the focus of this work.

The study of the constant-size subgraph-containment problem in the quantum setting was pioneered by Magniez, Szegedy and Santha [24], who built on a quantum walk algorithm by Ambainis [2] to show that the existence of any subgraph $H$ consisting of $k$ vertices can be detected using $\widetilde{O}(n^{2-2/k})$ queries. This result was subsequently independently improved by Zhu [28], and Lee, Magniez and Santha [21], resulting in the current state-of-the-art complexity for general subgraphs $H$ through the following learning-graph-based result.

▶ **Theorem 1** ([21, Theorem 4.4]). *Let $H$ be an undirected graph with $k \geq 3$ vertices, $m$ edges, and minimum vertex-degree $d \geq 1$, where $k, m, d \in O(1)$. Then, $\mathsf{Q}(\texttt{Subgraph}^H) \in O\left(n^{2-2/k-t}\right)$, where*

$$t = \max\left\{\frac{k^2 - 2(m+1)}{k(k+1)(m+1)}, \frac{2k-d-3}{k(d+1)(m-d+2)}\right\}.$$

Even though this result gives a non-trivial upper bound on the query complexity for the subgraph containment problem for any subgraph $H$, it is not expected that this bound is optimal in general. Indeed, for several specific families of subgraphs, we know better bounds than those provided by Lee, Magniez and Santha. Examples of such subgraph families include the line graphs with $k$ edges ($L_k$) and cycle graphs with $k$ edges ($C_k$), for which we refer to the subgraph-containment problems as $\texttt{Path}^{=k} := \texttt{Subgraph}^{L_k}$ and $\texttt{Cycle}^{=k} := \texttt{Subgraph}^{C_k}$, respectively.

The path-containment problem was pioneered by Childs and Kothari [14], who showed that it can be solved using $\widetilde{O}(n^{3/2-1/\lceil k/2\rceil})$ quantum queries. This was later improved by Belovs and Reichardt [6] to $\mathsf{Q}(\texttt{Path}^{=k}) \in \Theta(n)$, for all $k \in O(1)$. Their algorithm involves reducing this problem to the promise problem, which we denote by $\mathtt{PromPath}_{s,t}^{\leq k}$, where two vertices $s$ and $t$ are fixed, and the task is to distinguish between the existence of a path between $s$ and $t$ of length at most $k$, and no path between $s$ and $t$ at all. They used a span-program approach to show that $\mathsf{Q}(\mathtt{PromPath}_{s,t}^{\leq k}) \in O(n\sqrt{k})$, even for non-constant $k$.

Despite Belovs's and Reichardt's remarkable progress on the path-containment problem, subsequent efforts to generalize their techniques were unsuccessful. Specifically, all attempts to solve the path-containment problem in the directed setting, or a version of their promise problem where one is required to not just detect the existence of a path, but actually output it, have not resulted in linear-query algorithms. In this work, we argue why these efforts are unlikely to work, with the following reasoning containing two parts. First, we show that all these directed and finding versions of the path-containment problem are equivalent up to randomized reductions (Section 3), and then we show a conditional quantum query lower bound for all these problems based on the graph-collision problem (Section 6). Since it is considered unlikely that we can make progress on the graph-collision problem with the current techniques, we conclude it's unlikely that we can make progress on these generalizations of Belovs's and Reichardt's work with the current techniques as well.

On the topic of cycle-containment problems, the problem of detecting 3-cycles (more commonly known as the triangle-detection problem) has played a central role in the literature on quantum query complexity. The study of this problem gave rise to many novel quantum algorithmic frameworks, and has been improved multiple times since it was first considered in [9]. Currently, the best-known algorithm is by Carette, Laurière and Magniez [11], who removed the log-factors from the algorithm by Le Gall [20].

▶ **Theorem 2** ([20, 11])**.** $\mathsf{Q}(\mathtt{Cycle}^{=3}) \in O(n^{5/4})$.

The more general constant-length cycle-finding problem was also considered by Childs and Kothari [14], who showed the following improved upper bounds for even values of $k$.

▶ **Theorem 3** ([14, Theorem 4.11,4.12])**.** *Let $4 \leq k \in \mathbb{N}$ be even. Then,* $\mathsf{Q}(\mathtt{Cycle}^{=k}) \in \widetilde{O}\left(n^{\frac{3}{2}-\frac{k-2}{k(k+2)}}\right)$*. Moreover,* $\mathsf{Q}(\mathtt{Cycle}^{=4}) \in \widetilde{O}(n^{5/4})$.

Finally, Cade, Montanaro and Belovs [10] considered a promise version of the cycle-containment problem, where they fix a vertex $s$ in the input graph, and then distinguish between graphs with a cycle of length at most $k$ passing through $s$ and graphs with no cycles at all. Much like the promise path problem, they present an algorithm making $O(n\sqrt{k})$ queries, even for non-constant values of $k$.

As in the path-containment setting, all attempts to generalize the above results to the directed setting proved to be futile. Moreover, generalizing the promise version of the problem considered by Cade, Montanaro and Belovs to the finding setting, where one additionally has to output the cycle, proved elusive as well. In this work, we connect the cycle-containment problems rigorously to the path-containment problems through randomized reductions, explaining in a similar fashion why these generalization efforts are unlikely to succeed.

From the lower-bound perspective, progress is hampered by the *certificate barrier* [27, 26], which states that lower bounds obtained through the non-negative-weighted adversary bound are upper bounded by the geometric average of the worst-case positive and negative certificate sizes. For all subgraph containment problems where the subgraph $H$ is of constant size, the positive certificate sizes are of constant size as well, and therefore the best possible lower bound on the quantum query complexity we can obtain through the non-negative-weighted adversary bound is, up to constants, equal to $\sqrt{n^2 \cdot 1} = n$.

Even though there are other methods available for proving lower bounds on quantum query complexity, like the polynomial method [4], and the general adversary bound [17], applying these techniques to the subgraph containment problem has so far not been fruitful. As such, proving any unconditional super-linear lower bound on the quantum query complexity for any subgraph containment problem is a major open problem in this area that has been open for about two decades.

Despite (or perhaps because of) these barriers, alternative progress was made by Balodis and Iraids [3] in showing a conditional lower bound for the triangle-detection problem, based on the hardness of the graph-collision problem. This problem is parametrized by a graph $G$ on $n$ vertices, which is completely known ahead of time. The input to the problem is a bit sting $x$ that we have query access to, where each bit of $x$ is associated with a distinct vertex of $G$. The graph collision problem on $G$ asks whether there exists an edge in $G$ for which both endpoints are labeled by a 1 in $x$. We refer to this problem as $\mathtt{GC}_G$, and we write $\mathsf{Q}(\mathtt{GC}_n)$ for the maximum of $\mathsf{Q}(\mathtt{GC}_G)$, over all graphs $G$ on $n$ vertices.

The best generic algorithm for this problem was given by Magniez, Szegedy and Santha [24, Theorem 3], which is essentially based on the algorithm of Ambainis [2] and has query complexity $\mathsf{Q}(\mathtt{GC}_n) \in O(n^{2/3})$. This complexity is believed by some in the community to be optimal. However, the best-known lower bound is only $\mathsf{Q}(\mathtt{GC}_n) \in \Omega(\sqrt{n})$, where further progress is again impeded by the certificate barrier.

Balodis and Iraids embedded the OR of $n$ instances of the graph-collision problem into a single instance of the triangle-detection problem on $3n$ vertices. In doing so, they prove a conditional lower bound on the family of cycle-containment problems.

▶ **Theorem 4** ([3, Theorem 1])**.** $\mathsf{Q}(\mathtt{Cycle}^{=3}) \in \Omega(\sqrt{n} \cdot \mathsf{Q}(\mathtt{GC}_n))$.

This result implies a plausible path toward lower bounding the query complexity of the triangle-detection problem. Indeed, any lower bound $\omega(\sqrt{n})$ for the graph-collision problem would immediately imply a super-linear lower bound $\omega(n)$ for the triangle-detection problem.

In this work, we provide similar graph-collision-based conditional lower bounds for the families of path- and cycle-containment problems. This solidifies the role of the graph-collision problem in the landscape of subgraph-containment problems, as a non-trivial lower bound for graph-collision would imply non-trivial lower bounds for both cycle- and path-containment problems.

## 1.1 Contributions

In this work, we provide a rigorous study of the different variants of subgraph-containment problems, with a specific focus on path- and cycle-containment problems. We present a number of simpler *observations* based on prior work that help paint the landscape of these problems, and we obtain three *main results* that improve our understanding of their query complexities.

For the general constant-size subgraph-containment problem, we consider relations between the undirected and the directed setting, as well as between their detection and finding versions. Even though it is not explicitly analyzed by Lee, Magniez and Santha, we make the simple observation that their learning-graph-based approach readily generalizes to the directed setting.

▶ **Observation 5** (Informal version of Proposition 25)**.** *Theorem 1 holds too whenever the subgraph $H$ and the input graph are directed.*

Next, we also observe that for non-promise subgraph-containment problems, detection and finding are equivalent up to randomized reductions and constant multiplicative overhead.

▶ **Observation 6** (Informal version of Lemma 23). *For any (directed/undirected) subgraph $H$, the quantum query complexities of the finding and detection versions of the non-promise (possibly restricted) subgraph-containment problem with subgraph $H$ are equal up to a multiplicative constant.*

We note that the above only holds for non-promise problems, i.e., where all graphs are valid inputs to the problem. That is, one can only have separations between the detection and finding versions of a subgraph-containment problem in a setting where the input satisfied some promise known a priori. Indeed, in such promise settings, there are several graph problems where the finding and detection versions seem to be very much distinct. For instance, there is a quantum algorithm that makes polylogarithmic queries to output the (first bit of the) EXIT vertex in the Welded tree graph [12], while outputting an ENTRANCE-to-EXIT path in this graph takes polynomially many queries for a certain natural class of quantum algorithms [13]. Moreover, as we will see, the promise problem $\texttt{PromFindPath}_{s,t}^{=k}$ would be polynomially harder than $\texttt{PromPath}_{s,t}^{=k}$ assuming graph collision is harder than its trivial lower bound.

We also show that there is a generic randomized reduction from the undirected version of a subgraph-containment problem to the directed one.

▶ **Observation 7** (Informal version of Lemma 24). *Every directed subgraph-containment problem is at least as hard (in terms of randomized/quantum query complexity) as its undirected counterpart.*

Next, we focus specifically on the path- and cycle-containment problems. For these problems, we consider the restricted and unrestricted versions of the problem, where in the restricted versions, we fix the start and end vertices $s$ and $t$ for the path-containment problem, and a single vertex $s$ on the cycle for the cycle-finding problem. We also consider the directed versions of these problems, and the finding versions where we ask to output the path or cycle. Finally, we consider versions where we search for a path or cycle of length exactly $k$, or at most $k$. We denote the directed and finding versions with the keywords `Dir` and `Find`, we attach a superscript $= k$ or $\leq k$ to indicate the length constraint, and we supply a subscript $s$ or $s, t$ for the restricted versions. Thus, for instance, the problem $\texttt{FindDirPath}_{s,t}^{=k}$ is the finding version of the directed path-finding problem between $s$ and $t$, of length exactly $k$.

Akin to Belovs and Reichardt [6], and Cade et al. [10], we also consider promise versions of this problem. In the restricted path problem, i.e., where we fix the start and end vertex $s$ and $t$, the promise version satisfies that a path of the required length exists between $s$ and $t$, or $s$ and $t$ are not connected at all. Similarly, in the restricted cycle-finding problem through a vertex $s$, either a cycle through $s$ exists with the required length, or no cycle through $s$ exists at all. We note that this promise is weaker than the promise considered in Cade et al. [10], and so our algorithmic results for this problem are stronger. Finally, for the unrestricted cycle-finding problem, we also consider a promise version, where either a cycle exists of the required length, or there exists no cycle at all (i.e., the graph is a forest). These problems are indicated with the `Prom` keyword, i.e., $\texttt{PromDirCycle}_{s}^{\leq k}$ is the promise problem of detecting a directed cycle through $s$ of length at most $k$.[1]

Our first observation in this direction is that every `PromFind`-version of these problems requires the same number of queries as the corresponding `Find`-version.

[1] Note that we don't consider a promise version of the unrestricted path-containment problem, because we don't see a meaningful way of defining it.

▶ **Observation 8** (Informal version of Lemmas 28 and 38). *Let* `P` *be a (restricted/unrestricted, directed/undirected) path- or cycle-containment problem, with length constraint* $= k$ *or* $\leq k$ *with* $k \in O(1)$*. Then, the finding version of* `P` *is equivalent to the promise-finding version of* `P`*, up to randomized reductions and multiplicative constants.*

Observations 6 and 8 show that for all path- and cycle-containment problems `P`, we have that `P`, `FindP` and `PromFindP` are equivalent up to randomized reductions and multiplicative constants. As such, these three problems fall in the same equivalence class, and it suffices to restrict our attention to `P` and `PromP` in what follows.

Now, we arrive at the first main result of this work. We classify all remaining path- and cycle-containment problems into several equivalence classes via randomized reductions between them. We aid the reader by providing a graphical overview of these reductions in Figure 1, and we obtain the following classification.

▶ **Main result 9.** *The unrestricted and undirected path-containment problem, and the undirected promise versions of the path- and cycle-containment problems admit linear-query quantum algorithms. All other path-containment problems are equivalent, and all other cycle-containment problems can be classified into at most 5 equivalence classes. See Figure 1.*

For some reductions we use the color-coding technique, introduced by Alon, Yuster and Zwick [1]. The idea is to assign distinct colors to the vertices of the path/cycle graph $L_k/C_k$, and then to randomly assign these colors to the vertices of the input graph $G$. Next, we only keep an edge in $G$ if the colors of the two vertices it connects are associated to two vertices in the path/cycle graph that are connected by an edge too. The resulting color-coded graph becomes a layered graph in the path-containment case, and a layered cycle graph in the cycle-containment case, as displayed in Figure 2. With at least constant probability, a path/cycle in the original graph gets mapped to a path/cycle in the layered graph that straddles across all the layers.

To show relations between the various subgraph-containment problems, we cleverly make subtle modifications to the input graphs, oftentimes combined with the aforementioned color-coding technique. For instance, we can insert layers into a layered graph to show that one can find short paths using an algorithm that finds longer paths. Similarly, we can merge the vertices $s$ and $t$ into a single vertex $s$ to convert an $st$-path containment problem into a cycle-containment problem through the vertex $s$, and similarly we can separate a vertex $s$ into two vertices $s$ and $t$ to prove a reduction in the reverse direction.

All these conversions, however, are subject to subtle caveats. For instance, in a layered cycle graph with $k$ layers, a cycle of length $k$ is guaranteed to go through all layers whenever $k$ is odd, whereas when $k$ is even it can exist between two consecutive layers. Similarly, when we merge two vertices $s$ and $t$ into a single vertex $s$, a cycle through $s$ in the resulting graph does not necessarily imply that the original graph contains an $st$-path. We obtain our classification, i.e., Main result 9, by carefully checking which reductions can be made to work despite these caveats. Since we don't expect Figure 1 to collapse to a single island, we don't expect these subtleties to be artifacts of the techniques used, but rather highlight the fundamental relations between these problems.

The overview of reductions presented in Figure 1 reveals some interesting relations. For instance, it shows that one can reduce the $\texttt{DirCycle}_s^{\leq k}$-problem to the $\texttt{Cycle}_s^{\leq k}$-problem, through a series of reductions that involve the $\texttt{DirCycle}_s^{=k}$- and $\texttt{Cycle}_s^{=k}$-problems. We remark here that making this connection without going through the intermediate steps is not immediate, since it's not clear how one can remove directions in a layered graph while preventing the introduction of new cycles. This highlights the benefit of this classification effort, as it uncovers relations between problems that would otherwise not be easily obtained.

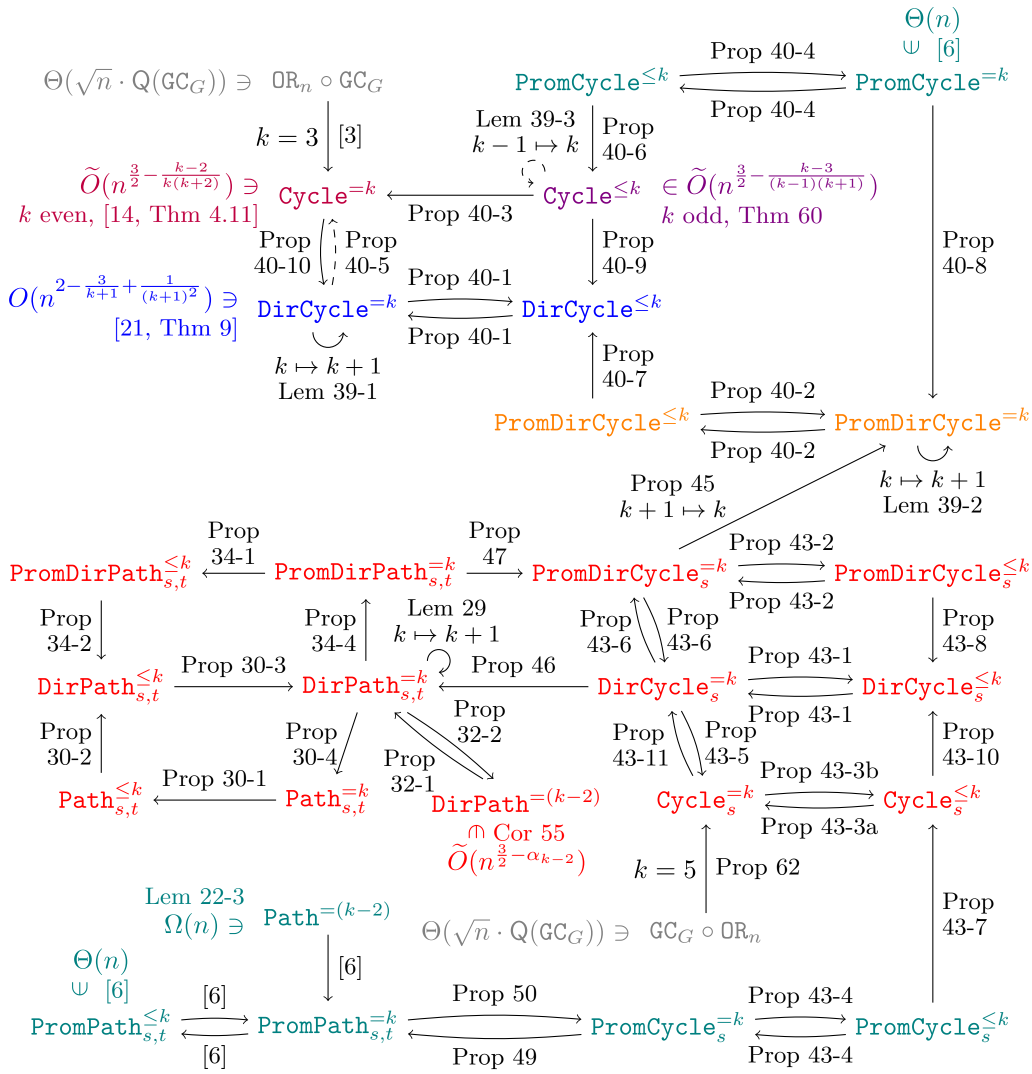


**Figure 1** Randomized reductions between problems for $k \geq 3$. The dashed connections hold only for odd values of $k$. For the graph-collision problems, we can take any graph $G$ with $n$ vertices. All problems' randomized query complexities are $\Theta(n^2)$. All complexity bounds stated in the figure are upper and lower bounds on the quantum query complexity. In the upper bound for $\mathsf{Q}(\mathtt{DirPath}^{=(k-2)})$, $\alpha_k \in \Theta(c^{-k})$, with $c \approx 1.33$. If two problems have the same color, they have the same quantum query complexity up to constants.

Next, the classification of path- and cycle-finding problems displayed in Figure 1 motivates the search for new quantum algorithms and query lower bounds for problems in these equivalence classes specifically. For instance, the $\mathtt{DirPath}^{=k}$-problem is part of an equivalence class containing 12 other path- and cycle-containment problems, hence an improved algorithm for this problem immediately provides an algorithm for all the others as well. Moreover, this equivalence class includes all directed versions of the path problems, as well as the finding versions of the promise-path problems, i.e., exactly the problems for which generalizing Belovs's and Reichardt's span-program-based construction failed.

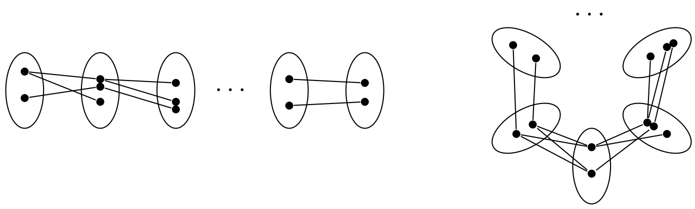

**Figure 2** A layered graph (left) and a layered cycle graph (right). These can be obtained from general graphs using the color-coding technique, where a path of length $k$ (resp. cycle of length $k$) is preserved with constant probability.

Our main algorithmic result is a novel quantum-walk-based algorithm for this $\texttt{DirPath}^{=k}$-problem, and hence for all problems in the equivalence class highlighted in red in Figure 1. The best previously-known algorithm for any of the problems in this equivalence class is an algorithm that finds the shortest path between $s$ and $t$, if it exists, and thereby solves the $\texttt{Path}_{s,t}^{\leq k}$-problem. This algorithm was originally developed by Dürr et al. [16], making $\widetilde{O}(n^{3/2})$ queries, and the log-factors were subsequently removed by a series of works culminating in a paper by Lin and Lin [22, Theorem 13]. We note that Beigi and Taghavi also developed $O(n^{3/2})$-query algorithms for the $\texttt{Cycle}_s^{=k}$ and $\texttt{Cycle}_s^{\leq k}$-problems [5, Proposition 9.iv-v]. Our result provides an improved algorithm for all these problems.

▶ **Main result 10** (Corollary 55). *Let* $5 \leq k \in \mathbb{N}$. *Then,* $\mathsf{Q}(\texttt{DirPath}^{=k}) \in \widetilde{O}(n^{3/2-\alpha_k})$, *where* $\alpha_k \in \Theta(c^{-k})$ *as* $k \to \infty$, *with* $c = \sqrt{3+\sqrt{17}}/2 \approx 1.33$.

The main algorithmic idea is to reduce the directed-path-detection problem of length $k$ to the same problem with length $k-2$. To that end, we employ the MNRS-framework [23] to search for vertices that have at least one incoming edge, and those that have at least one outgoing edge. Then, it remains to find a pair of such vertices that has a path of length $k-2$ between them, which we then solve recursively. This results in a nested quantum-walk construction, where we optimize the parameters of the quantum-walk framework to minimize the resulting query complexity. The resulting minimization problem involves solving a recurrence relation, whose solution yields an exponent that satisfies $3/2 - \Theta(c^{-k})$ for large values of $k$.

We additionally provide an improved algorithm for the $\texttt{Cycle}^{\leq k}$-problem, which is also displayed in Figure 1. A slight modification of Childs and Kothari [14] yields the following result.

▶ **Observation 11** (Theorem 60). *Let* $5 \leq k \in \mathbb{N}$ *be an odd integer. Then,* $\mathsf{Q}(\texttt{Cycle}^{\leq k}) \in \widetilde{O}(n^{\frac{3}{2}-\frac{k-5}{(k-1)(k+3)}})$.

Finally, we focus on the lower bounds. Our main result in this direction is relating the cycle-finding problem through a particular vertex $s$ to the graph-collision problem. In particular, through the reductions highlighted in Figure 1, we emphasize that this result also provides a conditional lower bound on the $\texttt{DirPath}^{=k}$-problem, and consequently on all path-containment problems that are in the same equivalence class.

▶ **Main result 12** (Proposition 62). *Let* $5 \leq k \in \mathbb{N}$ *be an odd integer. Then,* $\mathsf{Q}(\texttt{Cycle}_s^{\leq k}) \in \Omega(\sqrt{n} \cdot \mathsf{Q}(\texttt{GC}_n))$.

Even though the resulting lower bound is identical to the lower bound derived by Balodis and Iraids for the triangle-detection problem, the underlying construction is profoundly different. Whereas Balodis and Iraids embed $n$ instances of the graph-collision problem into a single instance of triangle-detection, we embed a single instance of the graph-collision problem composed with the OR-function into a single instance of the 5-cycle-detection problem through a fixed vertex $s$. In other words, Balodis and Iraids prove $\mathsf{Q}(\mathtt{Cycle}^{=3}) \in \Omega(\mathsf{Q}(\mathtt{OR}_n \circ \mathtt{GC}_n))$, whereas we prove that $\mathsf{Q}(\mathtt{Cycle}_s^{=5}) \in \Omega(\mathsf{Q}(\mathtt{GC}_n \circ \mathtt{OR}_n))$, where $\mathtt{OR}_n$ is the boolean functions encoding the OR-function on $n$ bits.

Finally, we remark that combining all our results provides conditional separations between the $\mathtt{Path}^{=k}$ and $\mathtt{DirPath}^{=k}$ problems, the $\mathtt{PromDirPath}_{s,t}^{=k}$ and $\mathtt{PromPath}_{s,t}^{=k}$ problems, and the $\mathtt{PromFindPath}_{s,t}^{=k}$ and $\mathtt{PromPath}_{s,t}^{=k}$ problems. Indeed, any non-trivial lower bound on the graph-collision problem implies a query separation between each pair of these problems. This in particular suggests that Belovs's and Reichardt's span-program-based approach can likely not be generalized to either the directed or finding settings, and, as such, that the undirected promise problems are indeed likely to be much easier than their directed and non-promise counterparts.

## 2 Preliminaries

### 2.1 Notation

Let $\mathbb{N} = \{1, 2, \dots\}$ be the set of all positive integers, and $\mathbb{N}_0 = \mathbb{N} \cup \{0\}$. For any integer $n \in \mathbb{N}$, we define $[n] \coloneqq \{1, \dots, n\}$ and $[n]_0 \coloneqq \{0, 1, \dots, n\}$.

Let $g : \mathbb{R} \supseteq D_g \to \mathbb{R}_{\geq 0}$. We define $O(g)$ to be the set of functions $f : D_g \supseteq D_f \to \mathbb{C}$ such that there exist $C, M > 0$ such that for all $x \in D_f$, $x > M \Rightarrow |f(x)| \leq C \cdot g(x)$. Similarly, we define $\widetilde{O}(g)$ to be the set of functions $f : D_g \supseteq D_f \to \mathbb{C}$ such that there exist $C, M, k > 0$ such that for all $x \in D_f$, $x > M \Rightarrow |f(x)| \leq C \cdot g(x) \cdot \log^k(g(x))$. We define $\Omega(g)$ and $\widetilde{\Omega}(g)$ analogously, but with the conditions being $|f(x)| \geq C \cdot g(x)$ and $|f(x)| \geq C \cdot g(x)/\log^k(g(x))$, respectively. Finally, we write $\Theta(g) \coloneqq O(g) \cap \Omega(g)$, and $\widetilde{\Theta}(g) \coloneqq \widetilde{O}(g) \cap \widetilde{\Omega}(g)$.

We denote undirected graphs by $G = (V, E)$, where every edge $\{v, w\} = e \in E$ is a set of two vertices, with $v, w \in V$. Similarly, we denote directed graphs by $G = (V, A)$, where every arc $(v, w) = a \in A$ is an ordered pair of vertices, with $v, w \in V$. If $V' \subseteq V$, then $G - V'$ denotes the graph $G$ with all vertices from $V'$ and their adjacent edges removed. All (undirected/directed) graphs considered in this work are simple, i.e., there are no duplicate edges and arcs. However, in the directed setting, we do allow for both arcs in opposite directions to exist simultaneously between two vertices.

We consider the adjacency-matrix model, where any algorithm can make queries to the entries of the adjacency matrix of an input graph $G$. Equivalently, for an undirected graph, the algorithm can supply a set of two vertices to an oracle, that outputs whether an edge exists between the two, and similarly in the directed setting the algorithm can ask whether an arc exists between two vertices from an ordered pair. In the randomized setting, the algorithm can employ randomness to select which vertices to query. In the quantum setting, the vertex pairs can be queried coherently in superposition, e.g., for an undirected graph $G = (V, E)$, the oracle behaves as a unitary $O_G$ that acts as

$$O_G : |v, w\rangle |b\rangle \mapsto |v, w\rangle |b \oplus [\{v, w\} \in E]\rangle, \qquad \forall v, w \in V, b \in \{0, 1\}.$$

We refer the reader to a quantum computing textbook for more details on the quantum computational model, e.g., [25]. The randomized and quantum query complexities of a given problem are the minimal numbers of queries required to solve the problem in these respective computational models.

Finally, we use regular expressions to denote sets of problems. That end, for two expressions $\mathtt{A}$ and $\mathtt{B}$, we write $(\mathtt{A}|\mathtt{B})$ to denote the set of expressions $\{\mathtt{A}, \mathtt{B}\}$. We use this in combination with concatenation, i.e., $\mathtt{A}(\mathtt{B}|\mathtt{C})$ denotes the set $\{\mathtt{AB}, \mathtt{AC}\}$. In particular, some of the expressions can be empty, i.e., $(|\mathtt{A})\mathtt{B}$ denotes the set $\{\mathtt{B}, \mathtt{AB}\}$.

## 2.2 Subroutines

We begin with a quantum subroutine that approximates the hamming weight of a Boolean string with a an error parameter $\epsilon$.

▶ **Theorem 13** (Approximate quantum counting [8, Theorem 15]). *Let $f : \{0,1\}^n \to \{0,1\}$ and $\epsilon \in (0,1]$. Let $t = |f^{-1}(1)|$. Then, there is a quantum algorithm that outputs $\tilde{t}$ such that $|t - \tilde{t}| \leq \epsilon t$ with probability at least $1 - \delta$ using $O\left(\frac{1}{\epsilon}\sqrt{n/t}\log\frac{1}{\delta}\right)$ queries to $f$.*

We will mainly need to use the approximate counting algorithm for approximating the number of edges in a graph and approximating the degree of a vertex. For that reason, we will need the following corollaries.

▶ **Corollary 14** ([14, Corollary 4.1]). *Let $A : \{0,1\}^{\binom{n}{2}} \to \{0,1\}$ be an adjacency matrix representation of an undirected graph $G$, $\epsilon \in (0,1]$ be a constant and $\overline{m}$ be an integer. Then, there is a quantum algorithm* ApproxEdgeCount *that accepts $G$ if the number of edges $|E(G)| = |A|$ in $G$ are at least $(1+\epsilon)\overline{m}$ and rejects $G$ if $|E(G)|$ is at most $(1-\epsilon)\overline{m}$ with probability at least $1 - \delta$ using $O\left(\frac{1}{\epsilon}\sqrt{n^2/\overline{m}}\log\frac{1}{\delta}\right)$ queries to $A$.*

▶ **Corollary 15.** *Let $A : \{0,1\}^{\binom{n}{2}} \to \{0,1\}$ be an adjacency matrix representation of an undirected graph $G$, $v$ be a vertex in $G$, $\epsilon \in (0,1]$ be a constant and $\overline{d}$ be an integer. Then, there is a quantum algorithm* ApproxDegreeCount *that accepts $(G, v)$ if the degree $\deg(v)$ of $v$ in $G$ are at least $(1+\epsilon)\overline{d}$ and rejects $G$ if $\deg(v)$ is at most $(1-\epsilon)\overline{d}$ with probability at least $1 - \delta$ using $O\left(\frac{1}{\epsilon}\sqrt{n/\overline{d}}\log\frac{1}{\delta}\right)$ queries to $A$.*

We will also need a subroutine that retrieves all the marked elements of a function.

▶ **Lemma 16** ([14, Lemma 4.1]). *Let $f : [n] \mapsto \{0,1\}$ and $t = |f^{-1}(1)|$. Then, there is a quantum algorithm that computes $f^{-1}(1)$ using $O(\sqrt{n(t+1)})$ queries to $f$.*

The following lemma relates the query complexity of problems that can be related by a randomized reduction with one-sided error.

▶ **Lemma 17.** *Let $\mathtt{P}$ and $\mathtt{P}'$ be arbitrary decision problems. Let $G$ be an instance of $\mathtt{P}$ and let $G'$ be obtained from $G$ via an efficient randomized reduction, i.e., each query to $G'$ can be made using $O(1)$ queries to $G$. Suppose that if $G$ is a* YES *instance of $\mathtt{P}$, then $G'$ is a* YES *instance of $\mathtt{P}'$ with probability at least $p \in \Omega(1)$, and if $G$ is a* NO *instance of $\mathtt{P}$, then $G'$ is a* NO *instance of $\mathtt{P}'$ with probability $1$. Then, for $\mathsf{M} \in \{\mathsf{R}, \mathsf{Q}\}$, we have $\mathsf{M}(\mathtt{P}) \in O(\mathsf{M}(\mathtt{P}'))$.*

**Proof.** Let $\mathcal{A}$ be the optimal (randomized/quantum) algorithm for the problem $\mathtt{P}'$. We first construct a new algorithm $\mathcal{B}$ with majority voting, where we run $\mathcal{A}$ on the same input a total of $\Theta(\log(1/p))$ times, so that it decides between YES and NO inputs with probability at least $1 - p/4$.

Now, we run the algorithm $\mathcal{B}$ on an instance $G'$ generated through the randomized reduction. Then, if $G$ is a positive input, this procedure will succeed with probability at least $p(1 - p/4) > 3p/4$, whereas if $G$ is a negative input, this procedure will succeed with probability at most $p/4$. Thus, by running this procedure a total of $\Theta(1/p)$ times, we can distinguish between these two cases with high probability. As such, the total number of calls to $\mathcal{A}$ is $\Theta((1/p)\log(1/p))$, which is constant as $p \in \Omega(1)$. Thus, $\mathsf{M}(\mathtt{P}) \in O(\mathsf{M}(\mathtt{P}'))$, for $\mathsf{M} \in \{\mathsf{Q}, \mathsf{R}\}$. ◀

▶ Remark 18. Note that, to be able to argue sub-quadratic quantum query bounds for graph problems on $n$ vertices through randomized reductions, these reductions must be query-efficient. In the reductions we use throughout this work, we express the edge set of the new graph in terms of the edge set of the original graph, such that checking a specific edge in the new graph requires $O(1)$ queries to the edge set of the original graph, unless explicitly stated otherwise.

## 2.3 Quantum walks

Throughout this paper, we will be making use of the MNRS-framework for quantum walks, developed by Magniez, Nayak, Rolland and Santha [23]. We recall their result here.

▶ **Theorem 19** ([23, Theorem 3]). *Let $P$ be a reversible ergodic Markov chain over a state space $V$, with spectral gap $\delta$. Let $M \subseteq V$ be a marked subset, such that if $M \neq \varnothing$, then $\mathbb{P}_{v\sim\pi}[v \in M] \geq \varepsilon > 0$, where $\pi$ is the stationary distribution of $P$. Let $\mathsf{S}$ be such that the number of queries required to prepare a q-sample of $|\pi\rangle$ up to norm-error at most $\delta$ is $\widetilde{O}(\mathsf{S}\mathrm{polylog}(1/\delta))$. Similarly, let $\mathsf{U}$ be such that the number of queries required to prepare a q-sample over the neighbors of a given vertex $v \in V$ with norm-error at most $\delta$ is $\widetilde{O}(\mathsf{U}\mathrm{polylog}(1/\delta))$. Finally, let $\mathsf{C}$ be the number of queries required to check whether a given state $v \in V$ is marked, with high probability. Then, we can decide whether $M$ is empty or non-empty with a total number of queries that satisfies*

$$\widetilde{O}\left(\mathsf{S} + \frac{1}{\sqrt{\varepsilon}}\left(\frac{1}{\sqrt{\delta}}\mathsf{U} + \mathsf{C}\right)\right).$$

On several occasions, we will recursively use this construction, which is known as a nested quantum walk. This has been considered ample times in previous works, for instance [14, 19]. These constructions typically suffer from polylogarithmic overhead, but recent work has shown that this can be avoided in certain settings [18, 15]. We leave using these more recent techniques to remove the polylogarithmic overhead from our algorithms for future work.

# 3 Reductions between subgraph containment problems

We start by formally introducing the different versions of the subgraph-containment problems we consider in this work.

▶ **Definition 20** (Subgraph containment). *We define several computational problems.*

1. *Let $H = (V_H, E_H)$ be a known undirected graph, and suppose we have access to an undirected graph $G = (V, E)$ through adjacency-matrix queries. Let $m \in \mathbb{N}_0$, $v_1, \dots, v_m \in V_H$ and $s_1, \dots, s_m \in V$ all distinct. The subgraph containment problem asks whether there exists an injection $\varphi : V_H \to V$, such that $\varphi : v_j \mapsto s_j$, for all $j \in [m]$, and such that the induced map $\varphi' : \binom{V_H}{2} \to \binom{V}{2}$ acting as $\{v, w\} \mapsto \{\varphi(v), \varphi(w)\}$ satisfies $\varphi'(E_H) \subseteq E$. The resulting boolean function, that evaluates to 1 if and only if such an injection exists, is denoted by*

$$\mathtt{Subgraph}^{H,v_1,\dots,v_m}_{V,s_1,\dots,s_m} : \{0,1\}^{\binom{V}{2}} \to \{0,1\},$$

*and we use the following shorthand notation for the sequence of boolean functions*

$$\mathtt{Subgraph}^{H}_{v_1,\dots,v_m} := (\mathtt{Subgraph}^{H,v_1,\dots,v_m}_{[n],1,\dots,m})_{n=1}^{\infty}.$$

*Whenever $m = 0$, we refer to the subgraph containment problem as unrestricted, in which case the boolean function is simply denoted by $\mathtt{Subgraph}^H_V$, and the sequence of boolean functions by $\mathtt{Subgraph}^H$. On the other hand, when $m \geq 1$, we say that the problem is restricted.*

2. *We define the restricted and unrestricted subgraph containment problems analogously for directed graphs, i.e., if $H = (V_H, A_H)$ is a directed graph and we have adjacency-matrix-query access to a directed graph $G = (V, A)$, we define the boolean function* $\texttt{Subgraph}_V^H : \{0,1\}^{\{(v,w):v,w\in V:v\neq w\}} \to \{0,1\}$*, and similarly in the restricted setting.*
3. *Finally, we define finding versions of these problems, which ask to output* $0$ *if the subgraph* $H$ *is not contained in* $G$*, and the injective map* $\varphi$ *otherwise. Formally, in the undirected unrestricted setting, this is a relation* $\texttt{FindSubgraph}_V^H \subseteq \{0,1\}^{\binom{V}{2}} \times (\{0\} \cup \{\varphi : V_H \to V \text{ injective}\})$*, and similarly for the restricted and directed settings.*

We can now phrase many graph problems formally as subgraph containment problems. For instance, if we let $H$ be an undirected cycle on three vertices, then the sequence of boolean functions $\texttt{Subgraph}^H$ represents the triangle finding problem, where we have the bounds $\mathsf{Q}(\texttt{Subgraph}^H) \in O(n^{5/4}) \cap \Omega(n)$ [20, 11].

In this work, we restrict our attention to subgraph-containment problems where the subgraph is of constant size. In this setting, we can prove several elementary structural properties of these problems. The proofs of these properties are available in the full version of the paper.

▶ **Lemma 21.** *Let $H$ be a graph of constant size, $m \in \mathbb{N}_0$ and $v_1, \dots, v_m \in V_H$. Suppose that $H_1, \dots, H_\ell$ are the connected components of $H$, and we write $i_1^{\ell'}, \dots, i_{m_{\ell'}}^{\ell'} \in [m]$ such that $v_{i_1^{\ell'}}, \dots, v_{i_{m_{\ell'}}^{\ell'}}$ are the vertices from $v_1, \dots, v_m$ that are contained in $H_{\ell'}$, for all $\ell' \in [\ell]$. Then, for $\mathsf{M} \in \{\mathsf{Q}, \mathsf{R}\}$, we have*

$$\mathsf{M}\left(\texttt{Subgraph}_{v_1,\dots,v_m}^H\right) \in \Theta\left(\max_{\ell' \in [\ell]} \left\{\mathsf{M}\left(\texttt{Subgraph}_{v_{i_1^{\ell'}},\dots,v_{i_{m_{\ell'}}^{\ell'}}}^{H_{\ell'}}\right)\right\}\right).$$

The above lemma informs us that we can without loss of generality focus on the case where $H$ is connected. Next, we provide a full classification of these problems in the randomized setting, and a similar partial classification in the quantum setting. This lemma is folklore (see for instance [6, Proposition 4] for the third item), but we provide a formal statement and proof here for completeness.

▶ **Lemma 22.** *Let $H$ be a connected non-empty graph of constant size, and let $m \in \mathbb{N}_0$ with $v_1, \dots, v_m \in V_H$. Then,*

1. *If $\{v_1, \dots, v_m\} = V_H$, then $\mathsf{Q}(\texttt{Subgraph}_{v_1,\dots,v_m}^H) \in \Theta(1)$ and $\mathsf{R}(\texttt{Subgraph}_{v_1,\dots,v_m}^H) \in \Theta(1)$.*
2. *If $H - \{v_1, \dots, v_m\}$ contains just isolated vertices, then $\mathsf{Q}(\texttt{Subgraph}_{v_1,\dots,v_m}^H) \in \Theta(\sqrt{n})$ and $\mathsf{R}(\texttt{Subgraph}_{v_1,\dots,v_m}^H) \in \Theta(n)$.*
3. *If $H - \{v_1, \dots, v_m\}$ contains an edge, then $\mathsf{Q}(\mathsf{Subgraph}_{v_1,\dots,v_m}^H) \in \Omega(n)$ and we have $\mathsf{R}(\texttt{Subgraph}_{v_1,\dots,v_m}^H) \in \Theta(n^2)$.*

Note that the above lemma completely characterizes the randomized query complexity of all detection versions of undirected subgraph containment problems. The simplicity of this characterization is in stark contrast with the quantum setting, where such a characterization seems elusive.

Next, we provide some generic reductions between the detection and finding versions of the subgraph containment problems, and between the undirected and directed versions.

▶ **Lemma 23.** *Let $H$ be a connected non-empty graph of constant size, $m \in \mathbb{N}_0$, and $v_1, \dots, v_m \in V_H$. Then, for $\mathsf{M} \in \{\mathsf{Q}, \mathsf{R}\}$, we have the characterization $\mathsf{M}(\texttt{Subgraph}_{v_1,\dots,v_m}^H) \in \Theta(\mathsf{M}(\texttt{FindSubgraph}_{v_1,\dots,v_m}^H))$.*

▶ **Lemma 24.** *Let $H$ be a connected non-empty undirected graph of constant size, $m \in \mathbb{N}_0$, and $v_1, \dots, v_m \in V_H$. Let $\vec{H}$ be the same graph as $H$, but then with arcs pointed in arbitrary directions. Then, for $\mathsf{M} \in \{\mathsf{Q}, \mathsf{R}\}$, we have $\mathsf{M}(\texttt{Subgraph}^H_{v_1,\dots,v_m}) \in O(\mathsf{M}(\texttt{Subgraph}^{\vec{H}}_{v_1,\dots v_m}))$.*

We note that the above lemmas also completely characterize the randomized query complexity for all directed and finding subgraph containment problems. We spend the rest of the document investigating the quantum counterparts of these results in more detail.

Finally, we recall that Theorem 1, as proved by Lee, Magniez and Santha, strictly speaking only holds for undirected subgraph-containment problems. However, we argue that their learning-graph-based algorithm also works in the directed setting, with the same complexity.

▶ **Proposition 25.** *Let $H = (V, A)$ be a directed graph, and let $H' = (V, E')$ be its undirected counterpart, i.e., $\{v, w\} \in E'$ if and only if $(v, w) \in A$ or $(w, v) \in A$. Then, $\mathsf{Q}(\texttt{Subgraph}^H)$ is upper bounded by the expression in Theorem 1 applied to $H'$.*

## 3.1 Path-containment reductions

We introduce shorthand notations for the subgraph-containment problems where the subgraph is a line graph. We refer to these as path-containment problems.

▶ **Definition 26** (Path-containment problems)**.** *Let $k \in \mathbb{N}$.*

1. *Let $L_k$ be the undirected line graph with $k$ edges, with the endpoint vertices labeled by $s$ and $t$. We define $\texttt{Path}^{=k} := \texttt{Subgraph}^{L_k}$, and $\texttt{Path}^{=k}_{s,t} := \texttt{Subgraph}^{L_k}_{s,t}$.*
2. *Let $L'_k$ be the directed version of $L_k$, where the edges are all pointing from $s$ to $t$. We define $\texttt{DirPath}^{=k} := \texttt{Subgraph}^{L'_k}$, and $\texttt{DirPath}^{=k}_{s,t} := \texttt{Subgraph}^{L'_k}_{s,t}$.*

*Finally, for $\texttt{P} \in (|\texttt{Dir})\texttt{Path}_{(|s,t)}$, we write*

$$\texttt{P}^{\leq k} := \bigvee_{\ell=1}^{k} \texttt{P}^{=\ell}.$$

*We also consider the finding versions of these problems, denoted by $\texttt{FindP}$ for all $\texttt{P} \in (|\texttt{Dir})\texttt{Path}^{(=k|\leq k)}_{(|s,t)}$. In the $(\leq k)$-problems, it suffices to output a path for any of the subproblems $\texttt{P}^{=\ell}$, with $\ell \in \{1, \dots, k\}$.*

We also consider promise versions of the path-containment problems.

▶ **Definition 27** (Promise path-containment problems)**.** *Let $k \in \mathbb{N}$.*

1. *The $\texttt{PromPath}^{(=k|\leq k)}_{s,t}$-problem is the $\texttt{Path}^{(=k|\leq k)}_{s,t}$-problem, restricted to inputs that satisfy the promise that if $s$ and $t$ are connected, then there exists a path of length exactly, or at most, $k$ between $s$ and $t$, respectively.*
2. *The $\texttt{PromDirPath}^{(=k|\leq k)}_{s,t}$-problem is the $\texttt{DirPath}^{(=k|\leq k)}_{s,t}$-problem, restricted to inputs that satisfy the promise that if $s$ and $t$ are connected by a directed path from $s$ to $t$, then there exists a directed path of length exactly, or at most, $k$ from $s$ to $t$, respectively.*

*The finding versions of these problems, $\texttt{PromFindPath}^{(=k|\leq k)}_{s,t}$ and $\texttt{PromFindDirPath}^{(=k|\leq k)}_{s,t}$ respectively, are the finding problems under the same restriction.*

Formally speaking the promise path-containment problems are also boolean functions, but on a restricted domain. As such, they are partial booelan functions, rather than total ones.

We start by showing some elementary reductions between the promise and non-promise versions of the path-containment problems. The proofs are available in the full version of the paper.

▶ **Lemma 28.** *Let* $\mathtt{P} \in (|\mathtt{Dir})\mathtt{Path}_{s,t}^{(=k|\leq k)}$.
1. *We have* $\mathsf{Q}(\mathtt{PromP}) \in O(\mathsf{Q}(\mathtt{P}))$.
2. *We have* $\mathsf{Q}(\mathtt{FindP}) \in \Theta(\mathsf{Q}(\mathtt{PromFindP}))$.

Through Lemmas 23 and 28, we conclude that the `Find` and `PromFind` versions of the path-containment problems considered above all have the same query complexity up to constants as the non-promise detection version of the problem. Thus, to understand the relative hardness of the path-containment problems introduced above, we will ignore the `Find` and `PromFind` versions in what follows. We provide a graphical overview of the remaining problems in Figure 3, with arrows representing the reductions we prove in the remainder of this section.

$$\begin{tikzcd}
 & \mathtt{PromDirPath}_{s,t}^{=k} \arrow[r, "\text{Prop 34-1}"] & \mathtt{PromDirPath}_{s,t}^{\leq k} \arrow[d, "\text{Prop 34-2}"] \\
\mathtt{DirPath}^{=(k-2)} \arrow[r, bend left, "\text{Prop 32-1}"] & \mathtt{DirPath}_{s,t}^{=k} \arrow[l, bend left, "\text{Prop 32-2}"] \arrow[u, "\text{Prop 34-4}"] \arrow[d, "\text{Prop 30-4}"] & \mathtt{DirPath}_{s,t}^{\leq k} \arrow[l, "\text{Prop 30-3}"'] \\
 & \mathtt{Path}_{s,t}^{=k} \arrow[r, "\text{Prop 30-1}"] & \mathtt{Path}_{s,t}^{\leq k} \arrow[u, "\text{Prop 30-2}"'] \\
\mathtt{Path}^{=(k-2)} \arrow[r, "{[6]}"] & \mathtt{PromPath}_{s,t}^{=k} \arrow[r, bend left, "{[6]}"] & \mathtt{PromPath}_{s,t}^{\leq k} \arrow[l, bend left, "{[6]}"] \quad \in \Theta(n) \; [6]
\end{tikzcd}$$

**Figure 3** Randomized reductions between path-containment problems. All the green problems can be solved in linear number of queries, and all the red problems are equivalent under randomized reductions and constant multiplicative overhead.

If we set $k = 1$, or $k = 2$ then the problems $(|\mathtt{Dir})\mathtt{Path}_{s,t}^{(=k|\leq k)}$ are instances of the first and second classes derived in Lemma 22, respectively. Thus, whenever we set $k = 1$ or $k = 2$, we already understand the exact query complexity of the problems in Figure 3, and so we assume without loss of generality that $k \geq 3$.

For the $\mathtt{PromPath}_{s,t}^{\leq k}$-problem, Belovs and Reichardt [6] proved that it can be solved using $O(n\sqrt{k})$ queries, which is simply $O(n)$ when $k$ is constant. They also realize that the $\mathtt{PromPath}_{s,t}^{=k}$-problem has a stronger promise, and so there is a generic reduction from this problem to the $\leq k$-version. Moreover, they provide a randomized reduction from the $\mathtt{Path}^{=(k-2)}$-problem to the $\mathtt{PromPath}_{s,t}^{=k}$-problem [6, Page 2]. This means that these three problems have linear query complexity, i.e., their query complexity is $\Theta(n)$ for $k \geq 3$.

We now prove that all the other path-containment problems form an equivalence class, i.e., they have the same query complexity up to a constant. We then prove a new quantum algorithm that solves all the problems from this equivalence class in Section 4.

We begin with showing monotonicity of $\mathsf{Q}(\mathtt{DirPath}_{s,t}^{\leq k})$ in terms of $k$. From our results later in this section, this monotonicity result also follows for all the problems in the same equivalence class as $\mathtt{DirPath}_{s,t}^{\leq k}$. All proofs are available in the full version of the paper.

▶ **Lemma 29.** *Let* $3 \leq \ell \leq k \in \mathbb{N}$. *Then,* $\mathsf{Q}(\mathtt{DirPath}_{s,t}^{=\ell}) \in O(\mathsf{Q}(\mathtt{DirPath}_{s,t}^{=k}))$.

▶ **Proposition 30.** *Let* $3 \leq k \in \mathbb{N}$. *Then,*
1. $\mathsf{Q}(\mathtt{Path}_{s,t}^{=k}) \in O(\mathsf{Q}(\mathtt{Path}_{s,t}^{\leq k}))$;
2. $\mathsf{Q}(\mathtt{Path}_{s,t}^{\leq k}) \in O(\mathsf{Q}(\mathtt{DirPath}_{s,t}^{\leq k}))$;
3. $\mathsf{Q}(\mathtt{DirPath}_{s,t}^{\leq k})) \in O(\mathsf{Q}(\mathtt{DirPath}_{s,t}^{=k}))$;
4. $\mathsf{Q}(\mathtt{DirPath}_{s,t}^{=k})) \in O(\mathsf{Q}(\mathtt{Path}_{s,t}^{=k}))$.

An immediate corollary of the above theorem is as follows.

▶ **Corollary 31.** *Let $3 \leq k \in \mathbb{N}$. Then, all the problems in* $(|\mathtt{Dir})\mathtt{Path}_{\mathtt{s,t}}^{(=\mathtt{k}|\leq\mathtt{k})}$ *have the same, up to constants, quantum query complexity.*

▶ **Proposition 32.** *Let $k \in \mathbb{N}$. Then,*
1. $\mathsf{Q}(\mathtt{DirPath}^{=k}) \in O(\mathsf{Q}(\mathtt{DirPath}_{s,t}^{=k+2}))$*;*
2. *For $k \geq 3$,* $\mathsf{Q}(\mathtt{DirPath}_{s,t}^{=k}) \in O(\mathsf{Q}(\mathtt{DirPath}^{=k-2}))$*.*

▶ **Corollary 33.** *Let $3 \leq k \in \mathbb{N}$. Then, the problems in* $\mathtt{DirPath}_{s,t}^{=k}$ *and* $\mathtt{DirPath}^{=k-2}$ *have the same, up to constants, quantum query complexity.*

▶ **Proposition 34.** *Let $3 \leq k \in \mathbb{N}$. Then,*
1. $\mathsf{Q}(\mathtt{PromDirPath}_{s,t}^{=k}) \in O(\mathsf{Q}(\mathtt{PromDirPath}_{s,t}^{\leq k}))$*;*
2. $\mathsf{Q}(\mathtt{PromDirPath}_{s,t}^{\leq k}) \in O(\mathsf{Q}(\mathtt{DirPath}_{s,t}^{\leq k}))$*;*
3. $\mathsf{Q}(\mathtt{DirPath}_{s,t}^{\leq k})) \in O(\mathsf{Q}(\mathtt{DirPath}_{s,t}^{=k}))$*;*
4. $\mathsf{Q}(\mathtt{DirPath}_{s,t}^{=k})) \in O(\mathsf{Q}(\mathtt{PromDirPath}_{s,t}^{=k}))$*.*

An immediate corollary of the above theorem is as follows.

▶ **Corollary 35.** *Let $3 \leq k \in \mathbb{N}$. Then, all the problems in* $(|\mathtt{Prom})\mathtt{DirPath}_{s,t}^{(=k|\leq k)}$ *have the same, up to constants, quantum query complexity.*

## 3.2 Cycle-containment reductions

We now introduce cycle-containment problems.

▶ **Definition 36** (Cycle-containment problems). *Let $3 \leq k \in \mathbb{N}$.*
1. *Let $C_k$ be the undirected cycle graph with $k$ edges, and let $s$ be any vertex on the cycle. Then,* $\mathtt{Cycle}^{=k} := \mathtt{Subgraph}^{C_k}$ *and* $\mathtt{Cycle}_s^{=k} := \mathtt{Subgraph}_s^{C_k}$*.*
2. *Let $C_k'$ be the directed cycle graph with $k$ arcs all pointing in the same direction, and let $s$ be any vertex on the cycle. Then,* $\mathtt{DirCycle}^{=k} := \mathtt{Subgraph}^{C_k'}$ *and* $\mathtt{DirCycle}_s^{=k} := \mathtt{Subgraph}_s^{C_k'}$*.*

*We also consider versions of the problem where we are looking for cycles of length at most $k$, i.e., for all* $\mathtt{P} \in (|\mathtt{Dir})\mathtt{Cycle}_{(|s)}$*, we write*

$$\mathtt{P}^{\leq k} = \bigvee_{\ell=3}^{k} \mathtt{P}^{=k}.$$

*Finally, we also consider the finding versions of these problems. That is, for all* $\mathtt{P} \in (|\mathtt{Dir})\mathtt{Cycle}_{(|s)}^{(=k|\leq k)}$*, we consider* $\mathtt{FindP}$ *where the task is to output the cycle. For the $(\leq k)$-problem, outputting a cycle of any length $3 \leq \ell \leq k$ suffices.*

For simplicity, we only consider cycles of length at least 3. Technically, one can also have a "cycle" of length 2 in the directed setting, if two arcs in opposite directions are present between two vertices, but we will refrain from calling these cycles in this work.

We also consider promise problems of these problems.

▶ **Definition 37** (Promise cycle-containment problems). *Let $3 \leq k \in \mathbb{N}$.*
1. *The* $\mathtt{PromCycle}^{(=k|\leq k)}$ *problem is the* $\mathtt{Cycle}^{(=k|\leq k)}$*-problem, restricted to inputs where if there exists any cycle in the graph, there must be at least one cycle of length exactly, or at most, $k$, respectively.*

2. *The* $\texttt{PromDirCycle}^{(=k|\leq k)}$ *problem is the* $\texttt{DirCycle}^{(=k|\leq k)}$*-problem, restricted to inputs where if there exists any directed cycle in the graph, there must be at least one directed cycle of length exactly, or at most,* $k$*, respectively.*
3. *The* $\texttt{PromCycle}_s^{(=k|\leq k)}$ *problem is the* $\texttt{Cycle}_s^{(=k|\leq k)}$*-problem, restricted to inputs where if there exists any cycle in the graph that passes through* $s$*, there must be at least one cycle of length exactly, or at most,* $k$*, respectively, that passes through* $s$*.*
4. *The* $\texttt{PromDirCycle}_s^{(=k|\leq k)}$ *problem is the* $\texttt{DirCycle}_s^{(=k|\leq k)}$*-problem, restricted to inputs where if there exists any directed cycle in the graph that passes through* $s$*, there must be at least one directed cycle of length exactly, or at most,* $k$*, respectively, that passes through* $s$*.*

*We consider the finding versions of these problems as well, which are defined to be the finding problems under the same restrictions on the input, denoted by* $\texttt{PromFind(|Dir)Cycle}_{(|s)}^{(=k|\leq k)}$*.*

We note here that the same reductions hold between the promise and non-promise versions of the cycle-containment problems as in the path case.

▶ **Lemma 38.** *Let* $3 \leq k \in \mathbb{N}$*, and* $\texttt{P} \in \texttt{(|Dir)Cycle}_{(|s)}^{(=k|\leq k)}$*.*
1. *We have* $\mathsf{Q}(\texttt{PromP}) \in O(\mathsf{Q}(\texttt{P}))$*.*
2. *We have* $\mathsf{Q}(\texttt{FindP}) \in \Theta(\mathsf{Q}(\texttt{PromFindP}))$*.*

Similar to the path-containment setting, the above lemma informs us that we don't need to consider the finding versions of the cycle-containment problems, since they are always equivalent to the non-promise detection versions up to randomized reductions. Thus, for all $\texttt{P} \in \texttt{(|Dir)Cycle}_{(|s)}^{(=k|\leq k)}$, we only consider the problems $\texttt{P}$ and $\texttt{PromP}$ in the remainder of this section.

We now prove several randomized reductions between these problems, and we give a concise graphical overview of them in Figure 4.

For the following problems we can show that the quantum query complexity is monotonically increasing in $k$. The proofs are available in the full version of the paper.

▶ **Lemma 39.** *Let* $3 \leq \ell \leq k \in \mathbb{N}$*. Then,*
1. $\mathsf{Q}(\texttt{DirCycle}^{=\ell}) \in O(\mathsf{Q}(\texttt{DirCycle}^{=k}))$*;*
2. $\mathsf{Q}(\texttt{PromDirCycle}^{=\ell}) \in O(\mathsf{Q}(\texttt{PromDirCycle}^{=k})$*;*
3. $\mathsf{Q}(\texttt{Cycle}^{\leq(k-1)}) \in O(\mathsf{Q}(\texttt{Cycle}^{\leq k}))$ *whenever* $k$ *is an odd integer.*

*The same monotonicity relations hold for* $\texttt{P}_s$ *where* $\texttt{P} \in \{\texttt{DirCycle}, \texttt{PromDirCycle}, \texttt{Cycle}\}$*.*

We will now prove a few cycle-containment reductions as illustrated in Figure 4.

▶ **Proposition 40.** *Let* $3 \leq k \in \mathbb{N}$ *be a constant. Then,*
1. $\mathsf{Q}(\texttt{DirCycle}^{=k}) \in \Theta(\mathsf{Q}(\texttt{DirCycle}^{\leq k}))$*;*
2. $\mathsf{Q}(\texttt{PromDirCycle}^{\leq k}) \in \Theta(\mathsf{Q}(\texttt{PromDirCycle}^{=k}))$*;*
3. $\mathsf{Q}(\texttt{Cycle}^{\leq k}) \in O(\mathsf{Q}(\texttt{Cycle}^{=k}))$*;*
4. $\mathsf{Q}(\texttt{PromCycle}^{=k}) \in \Theta(\mathsf{Q}(\texttt{PromCycle}^{\leq k}))$*;*
5. $\mathsf{Q}(\texttt{DirCycle}^{=k}) \in O(\mathsf{Q}(\texttt{Cycle}^{=k}))$ *whenever* $k$ *is an odd integer;*
6. $\mathsf{Q}(\texttt{PromCycle}^{\leq k}) \in O(\mathsf{Q}(\texttt{Cycle}^{\leq k}))$*;*
7. $\mathsf{Q}(\texttt{PromDirCycle}^{\leq k}) \in O(\mathsf{Q}(\texttt{DirCycle}^{\leq k})))$*;*
8. $\mathsf{Q}(\texttt{PromCycle}^{=k}) \in O(\mathsf{Q}(\texttt{PromDirCycle}^{=k}))$*;*
9. $\mathsf{Q}(\texttt{Cycle}^{\leq k}) \in O(\mathsf{Q}(\texttt{DirCycle}^{\leq k}))$*;*
10. $\mathsf{Q}(\texttt{Cycle}^{=k}) \in O(\mathsf{Q}(\texttt{DirCycle}^{=k}))$*.*

▶ **Theorem 41** (Implicit in [6, Theorem 8])**.** *Let* $3 \leq k \in \mathbb{N}$ *be a constant. Then, there exists a* $O(n)$*-query quantum algorithm that, given a simple graph* $G$ *with* $n$ *vertices, accepts if* $G$ *contains a cycle of length* $k$ *and rejects if* $G$ *is a forest, except with error probability at most* $1/3$*.*

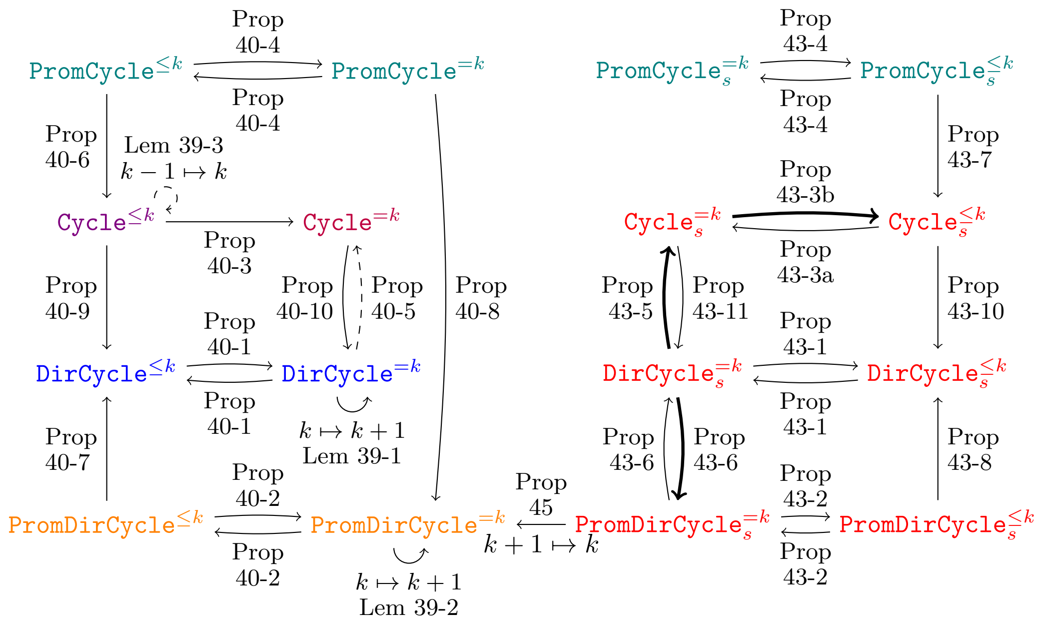


**Figure 4** Pictorial representation of the randomized reductions we prove in this section, for $k \geq 3$. The dashed connections only hold for odd values of $k$. Problems depicted in the same color have the same quantum query complexity up to constants. Note that the connections represented in bold only exist in the restricted cycle-containment version of the problem, i.e., the corresponding result cannot be obtained in the left-hand side of the picture using the same techniques.

Combining Item 4 of Proposition 40 and Theorem 41 we conclude the following.

▶ **Corollary 42.** *Let $3 \leq k \in \mathbb{N}$ be a constant. Then,* $\mathsf{Q}(\texttt{PromCycle}^{=k|\leq k}) \in \Theta(n)$.

We will now present a few other cycle-containment reductions where the cycles are required to pass through a specific vertex, let us denote by $s$, given as input. For these reductions we will use color-coding and layer insertion techniques as done earlier, but we will use the color-coding version that passes through $s$. It is interesting to note that for some of these problems we are able to prove stronger relation in contrast to its counterparts that don't have this additional requirement of the subgraph containing $s$. For example, Items 3b and 5 in Proposition 43 are to name a few. See Figure 4 for an illustrative and comparative summary of the results.

▶ **Proposition 43.** *Let $3 \leq k \in \mathbb{N}$ be a constant. Then,*

1. $\mathsf{Q}(\texttt{DirCycle}_s^{=k}) \in \Theta(\mathsf{Q}(\texttt{DirCycle}_s^{\leq k}))$;
2. $\mathsf{Q}(\texttt{PromDirCycle}_s^{\leq k}) \in \Theta(\mathsf{Q}(\texttt{PromDirCycle}_s^{=k}))$;
3. **a.** $\mathsf{Q}(\texttt{Cycle}_s^{\leq k}) \in O(\mathsf{Q}(\texttt{Cycle}_s^{=k}))$;
   **b.** $\mathsf{Q}(\texttt{Cycle}_s^{=k}) \in O(\mathsf{Q}(\texttt{Cycle}_s^{\leq k}))$;
4. $\mathsf{Q}(\texttt{PromCycle}_s^{=k}) \in \Theta(\mathsf{Q}(\texttt{PromCycle}_s^{\leq k}))$;
5. $\mathsf{Q}(\texttt{DirCycle}_s^{=k}) \in O(\mathsf{Q}(\texttt{Cycle}_s^{=k}))$;
6. $\mathsf{Q}(\texttt{PromDirCycle}_s^{=k}) \in \Theta(\mathsf{Q}(\texttt{DirCycle}_s^{=k}))$;
7. $\mathsf{Q}(\texttt{PromCycle}_s^{\leq k}) \in O(\mathsf{Q}(\texttt{Cycle}_s^{\leq k}))$;
8. $\mathsf{Q}(\texttt{PromDirCycle}_s^{\leq k}) \in O(\mathsf{Q}(\texttt{DirCycle}_s^{\leq k})))$;
9. $\mathsf{Q}(\texttt{PromCycle}_s^{=k}) \in O(\mathsf{Q}(\texttt{PromDirCycle}_s^{=k}))$;
10. $\mathsf{Q}(\texttt{Cycle}_s^{\leq k}) \in O(\mathsf{Q}(\texttt{DirCycle}_s^{\leq k}))$;
11. $\mathsf{Q}(\texttt{Cycle}_s^{=k}) \in O(\mathsf{Q}(\texttt{DirCycle}_s^{=k}))$.

Consequently, we get the following.

▶ **Corollary 44.** *Let $3 \leq k \in \mathbb{N}$. Then, all the problems in* $(|\texttt{Prom})\texttt{DirCycle}_s^{(=k|\leq k)}$ *have the same, up to constants, quantum query complexity.*

Finally, we prove a connection between the restricted and unrestricted version of the cycle-containment problems.

▶ **Proposition 45.** *Let $3 \leq k \in \mathbb{N}$ be a constant. Then, we have* $\mathsf{Q}(\texttt{PromDirCycle}_s^{=(k+1)}) \in O(\mathsf{Q}(\texttt{PromDirCycle}_s^{=k}))$.

## 3.3 Path-cycle-containment reductions

In this subsection, we present reductions between path and cycle problems (in both directions) to further refine the classification established in Sections 3.1 and 3.2 for path and cycle problems, respectively. All the proofs are available in the full version of the paper.

▶ **Proposition 46.** *Let $3 \leq k \in \mathbb{N}$ be a constant. Then,* $\mathsf{Q}(\texttt{DirCycle}_s^{=k}) \in O(\mathsf{Q}(\texttt{DirPath}_{s,t}^{=k}))$.

▶ **Proposition 47.** *Let $3 \leq k \in \mathbb{N}$ be a constant. Then, we have* $\mathsf{Q}(\texttt{PromDirPath}_{s,t}^{=k}) \in O(\mathsf{Q}(\texttt{PromDirCycle}_s^{=k}))$.

Using Corollary 35 from Section 3.1, Corollary 44 from Section 3.2 and Propositions 46 and 47 we conclude the following.

▶ **Corollary 48.** *Let $3 \leq k \in \mathbb{N}$ be a constant. Then, the path problems* $(|\texttt{Prom})\texttt{DirPath}_{s,t}^{(=k|\leq k)}$ *and the cycle problems* $(|\texttt{Prom})\texttt{DirCycle}_s^{(=k|\leq k)}$ *have the same, up to constants, quantum query complexity.*

▶ **Proposition 49.** *Let $3 \leq k \in \mathbb{N}$ be constant. Then,* $\mathsf{Q}(\texttt{PromCycle}_s^{=k}) \in O(\mathsf{Q}(\texttt{PromPath}_{s,t}^{=k}))$.

▶ **Proposition 50.** *Let $3 \leq k \in \mathbb{N}$ be constant. Then,* $\mathsf{Q}(\texttt{PromPath}_{s,t}^{=k}) \in O(\mathsf{Q}(\texttt{PromCycle}_s^{=k}))$.

As $\texttt{PromPath}_{s,t}^{=k}$ is known to have quantum query complexity of $\Theta(n)$ (see Figure 3), using Item 4 of Proposition 43 and Propositions 49 and 50 we can conclude the following:

▶ **Corollary 51.** *Let $3 \leq k \in \mathbb{N}$ be a constant. Then,* $\mathsf{Q}(\texttt{PromCycle}_s^{(=k|\leq k)}) \in \Theta(n)$.

# 4 Length-$k$ directed path-detection algorithm

In this section, we consider the subgraph containment problem $\texttt{DirPath}^{=k}$, and we develop a novel algorithm for this problem. This implies a query upper bound for all of the problems displayed in red in Figure 1.

We start by employing the color-coding technique, introduced by Alon, Yuster and Zwick [1], to partition the graph into $k+1$ subsets. All proofs are available in the full version of the paper.

▶ **Lemma 52.** *Let $k, n \in \mathbb{N}$, and let $G = (V, A)$ be a directed graph with $|V| = n$ and $k \in O(1)$. Let $V_0 \sqcup \cdots \sqcup V_k \subseteq V$ be a random partition of $V$ such that $|V_j| \in \Theta(n/k)$. We define $G' = (V, A')$, where $A' = A_1 \sqcup \cdots \sqcup A_k$, with $A_j = \{(v, w) \in A : v \in V_{j-1}, w \in V_j\}$. Now, if $\texttt{DirPath}^{=k}(G) = 0$, then $\texttt{DirPath}^{=k}(G') = 0$. On the other hand, if $\texttt{DirPath}^{=k}(G) = 1$, then $\mathbb{P}_{G'}[\texttt{DirPath}^{=k}(G') = 1] \in \Omega(1)$, where the probability distribution is over the randomization in the procedure that constructs $G'$.*

We can combine the previous theorem with Lemma 17, so without loss of generality we can consider the color-coded version of the problem, i.e., where the graph's vertices are already partitioned into $k+1$ disjoint sets and where we're looking for a path that traverses all the way from one end to the other, as displayed in Figure 5. To optimize the parameters in the recursion, we consider a version this problem where the first and last layer have a smaller size than all the intermediate layers, and setting $r \in \Theta(n)$ recovers the problem considered in Lemma 52.

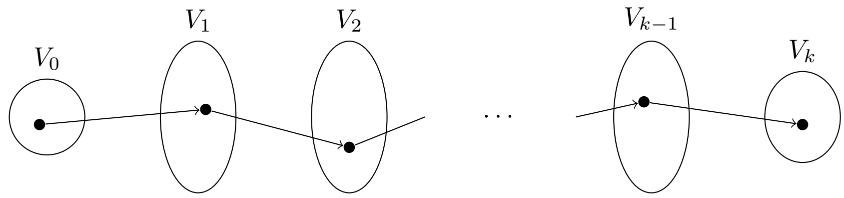


**Figure 5** The layered-path problem. Every set $V_j$ for $j \in [k-1]$ represents a set of $\Theta(n)$ vertices, and the first and last layer might have a different size, say $|V_0| = |V_k| = r$. We are looking for a directed path that traverses all the layers from $V_0$ to $V_k$.

We now provide a recursive nested walk construction that finds such a layered path, if it exists.

▶ **Lemma 53.** *Let $k, n, r \in \mathbb{N}$, with $3 \leq k \in O(1)$ and $r \leq n$. Let $n/2 \geq s \in \mathbb{N}$, and suppose that we can solve the layered path-finding problem with length $k-2$ and first and last layer size $s$ with $C_{n,s}^{(k-2)}$ queries. Then, we can solve the layered path-finding problem with complexity $C_{n,r}^{(k)}$, satisfying*

$$C_{n,r}^{(k)} \in \widetilde{O}\left(s\sqrt{r} + \left(\frac{n}{s}\right)\left(\sqrt{sr} + C_{n,s}^{(k-2)}\right)\right).$$

Now that we have described the recursive algorithm, we should work out what the resulting number of queries becomes, for a given value of $k \in O(1)$. This is the objective of the following lemma.

▶ **Lemma 54.** *Let $\alpha_r \in [0,1]$. For all $k \in \mathbb{N}$, let $\alpha_{k,r}$ be defined by the recurrence relation*

$$\alpha_{k+2,r} := \min_{\alpha_s \in [0,1]} \max\left\{\alpha_s + \frac{1}{2}\alpha_r, 1 - \frac{1}{2}\alpha_s + \frac{1}{2}\alpha_r, 1 - \alpha_s + \alpha_{k,s}\right\},$$

*with $\alpha_{1,r} = \alpha_r$, and $\alpha_{2,r} = \frac{1}{2} + \frac{1}{2}\alpha_r$. Next, let $n \in \mathbb{N}$, and $r \in \Theta(n^{\alpha_r})$. Then, we can solve the layered-path problem with first and last layer size $r$ with a number of queries that satisfies $C_{n,r}^{(k)} \in \widetilde{O}(n^{\alpha_{k,r}})$. Moreover, the solution to the recurrence relation satisfies $\alpha_{k,n} \in 3/2 - \Theta(c^{-k})$ with $c \approx 1.33$.*

Putting everything together, we now obtain the following result.

▶ **Corollary 55.** $\mathsf{Q}(\mathtt{DirPath}^{=k}) \in \widetilde{O}(n^{\frac{3}{2}-\alpha_k})$*, where $\alpha_k \in \Omega(c^{-k})$ as $k \to \infty$, with $c = \sqrt{3+\sqrt{17}}/2 \approx 1.33$.*

We compute the resulting query complexity for the first few values of $k$ for good measure.

## 5 Length-$\leq k$ cycle-detection algorithm

In this section, we modify the approach developed in [14], to obtain a novel quantum algorithm for the $\mathtt{Cycle}^{\leq k}$-problem, whenever $k$ is odd. The core result that we are using is the following theorem.

**Table 1** The query complexity of $\mathsf{Q}(\texttt{DirPath}^{=k})$ for the first few values of $k$.

| $k$ | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
|---|---|---|---|---|---|---|---|
| $\mathsf{Q}(\texttt{DirPath}^{=k})$ | $\widetilde{O}(n)$ | $\widetilde{O}(n)$ | $\widetilde{O}(n^{7/6})$ | $\widetilde{O}(n^{7/6})$ | $\widetilde{O}(n^{23/18})$ | $\widetilde{O}(n^{29/22})$ | $\widetilde{O}(n^{91/66})$ |

▶ **Theorem 56** ([14, Theorem 4.7]). *Let $\mathcal{P}$ be the property that an $n$-vertex graph either has more than $\overline{m}$ edges, where $\overline{m} \in \Omega(n)$, or contains a given subgraph $H$. Let $H'$ be the graph obtained by deleting all degree-1 vertices of $H$ that are not part of an isolated edge. Then $\mathsf{Q}(\mathcal{P}) \in \widetilde{O}(\sqrt{\overline{m}}n^{1-\frac{1}{vc(H')+1}})$, where $vc(H')$ denotes the vertex cover of $H'$, i.e., the minimum number of vertices in $H'$ required to cover all edges.*

Applying this to $k$-cycles, we get the following corollary.

▶ **Corollary 57.** *Let $k \geq 3$. Let $\mathcal{P}$ be the property that an $n$-vertex graph either has more than $\overline{m}$ edges, where $\overline{m} \in \Omega(n)$, or contains a $k$-cycle. Then, there is a quantum algorithm that computes $\mathcal{P}$ using $\widetilde{O}(\sqrt{\overline{m}}n^{1-\frac{1}{\lceil k/2\rceil+1}})$ queries.*

We modify the algorithm attaining the result in Theorem 56 (and Corollary 57) to get a slight improvement for the case when $k$ is odd (making a stronger assumption on $\overline{m}$ which will also be satisfied for our applications). The proof is available in the full version of the paper.

▶ **Theorem 58.** *Let $k \geq 3$. Let $\mathcal{P}$ be the property that an $n$-vertex graph either has more than $\overline{m}$ edges, where $\overline{m} \in \Omega(n^{1+1/(\ell+1)})$ and $\ell = \lfloor k/2 \rfloor$, or contains a $k$-cycle. Then, there is a quantum algorithm* SparseCycle *that computes $\mathcal{P}$ using $\widetilde{O}(\sqrt{\overline{m}}n^{1-\frac{1}{\ell+1}})$ queries.*

Now, we can combine this construction with a combinatorial result by Bondy and Simonovits [7], that connects the non-existence of an even-length cycle of a given length to a sparsity condition on the graph.

▶ **Theorem 59** ([7]). *Let $G$ be a graph on $n$ vertices. For any $\ell \geq 2$, if $|E(G)| > 100\ell n^{1+1/\ell}$ then $G$ contains $\mathcal{C}_{2\ell}$ as a subgraph.*

Combining the previous two results now gives us our algorithm that solves the $\texttt{Cycle}^{\leq k}$-problem. The proof is available in the full version of the paper.

▶ **Theorem 60.** *Let $k \geq 4$. Then, $\mathsf{Q}(\texttt{Cycle}^{\leq k}) \in \widetilde{O}(n^{\frac{3}{2}-\gamma(k)})$, where*

$$\gamma(k) = \begin{cases} \frac{k-2}{k(k+2)}, & \text{if } k \text{ is even,} \\ \frac{k-3}{(k-1)(k+1)}, & \text{if } k \text{ is odd.} \end{cases}$$

Finally, we summarize the best-known upper bounds on the query complexities for $\texttt{Cycle}^{=k}$ and $\texttt{Cycle}^{\leq k}$ in Figure 6. We observe that for all $k$, the $\texttt{Cycle}^{\leq k}$-problem can be solved in $o(n^{3/2})$ queries, whereas for the $\texttt{Cycle}^{=k}$-problem, we can only make that statement for even values of $k$. It would be a very interesting direction of future research to establish whether the $\texttt{Cycle}^{=k}$-problem for odd values of $k$ can be solved in $\widetilde{O}(n^{3/2})$ queries as well.

We also observe that the expressions we obtain for general values of $k$ do not recover the best-known upper bounds for the cases where $k = 3$ and $k = 4$. This suggests that for higher values of $k$, the upper bounds we know for these cycle-containment problems are likely not tight. We think it's a very nice future direction of research to improve these algorithms.

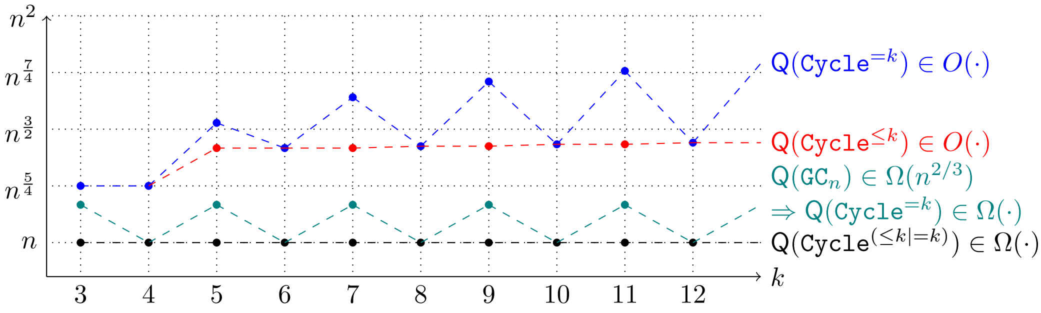


**Figure 6** The best-known upper bounds on the query complexity of the $\mathtt{Cycle}^{=k}$-problem (shown in blue) and the $\mathtt{Cycle}^{\leq k}$-problem (shown in red). The black vertices represent the best-known unconditional lower bounds on both problems, and the green dots represent the best-known conditional lower bounds on the $\mathtt{Cycle}^{=k}$-problem, conditional on $\mathsf{Q}(\mathtt{GC}_n) \in \Omega(n^{2/3})$.

## 6 Conditional lower bounds based on graph collision

In this final section, we prove a novel super-linear conditional lower bound on the query complexities of several problems in Figure 1. Our lower bounds will be based on the graph-collision problem, introduced in [24, Section 4.2].

▶ **Definition 61** (Graph collision problem [24, Section 4.2]). *Let $G = (V, E)$ be an undirected graph. We consider the boolean function $\mathtt{GC}_G = \mathtt{GraphCollision}_G : \{0,1\}^V \to \{0,1\}$, which evaluates to 1 on input $x \in \{0,1\}^V$ if and only if there exists an edge $vw \in E$, such that $x_v = x_w = 1$. This is the graph collision problem on $G$. For all $n \in \mathbb{N}$,*

$$\mathsf{Q}(\mathtt{GC}_n) := \max_{\substack{G=(V,E)\\|V|=n}} \mathsf{Q}(\mathtt{GC}_G).$$

Balodis and Iraids [3] showed that the quantum query complexity of triangle finding is related to the graph collision problem, which gives us a conditional lower bound on several of the equivalence classes displayed in Figure 1. We modify Balodis's and Iraids's approach to give the same lower bound to the cycle-finding problem through a vertex $s$. The proof is available in the full version of the paper.

▶ **Proposition 62.** $\mathsf{Q}(\mathtt{Cycle}_s^{=5}) \in \Omega(\sqrt{n} \cdot \mathsf{Q}(\mathtt{GC}_n))$.

### References

1 Noga Alon, Raphael Yuster, and Uri Zwick. Color-coding. *Journal of the ACM (JACM)*, 42(4):844–856, 1995. doi:10.1145/210332.210337.
2 Andris Ambainis. Quantum walk algorithm for element distinctness. *SIAM Journal on Computing*, 37(1):210–239, 2007. doi:10.1137/S0097539705447311.
3 Kaspars Balodis and Jānis Iraids. Quantum lower bound for graph collision implies lower bound for triangle detection. *Baltic Journal of Modern Computing*, 4(4), 2016. doi:10.22364/BJMC.2016.4.4.10.
4 Robert Beals, Harry Buhrman, Richard Cleve, Michele Mosca, and Ronald de Wolf. Quantum lower bounds by polynomials. In *Proceedings of the 39th Annual Symposium on Foundations of Computer Science*, 1998.
5 Salman Beigi and Leila Taghavi. Quantum speedup based on classical decision trees. *Quantum*, 4:241, 2020. doi:10.22331/Q-2020-03-02-241.

6 Aleksandrs Belovs and Ben W Reichardt. Span programs and quantum algorithms for st-connectivity and claw detection. In *European Symposium on Algorithms*, pages 193–204. Springer, 2012. doi:10.1007/978-3-642-33090-2_18.

7 J. Adrian Bondy and Miklós Simonovits. Cycles of even length in graphs. *Journal of Combinatorial Theory, Series B*, 16(2):97–105, 1974.

8 Gilles Brassard, Peter Hoyer, Michele Mosca, and Alain Tapp. Quantum amplitude amplification and estimation. *AMS Contemporary Mathematics Series*, 305, 2000.

9 Harry Buhrman, Ronald de Wolf, Christoph Dürr, Mark Heiligman, Peter H"yer, Frédéric Magniez, and Miklos Santha. Quantum algorithms for element distinctness. In *Proceedings of the 16th Annual Conference on Computational Complexity*, 2001.

10 Chris Cade, Ashley Montanaro, and Aleksandrs Belovs. Time and space efficient quantum algorithms for detecting cycles and testing bipartiteness. *Quantum Information and Computation*, 18(1-2):18–50, 2018. doi:10.26421/QIC18.1-2-2.

11 Titouan Carette, Mathieu Laurière, and Frédéric Magniez. Extended learning graphs for triangle finding. *Algorithmica*, 82(4):980–1005, 2020. doi:10.1007/S00453-019-00627-Z.

12 Andrew M. Childs, Richard Cleve, Enrico Deotto, Edward Farhi, Sam Gutmann, and Daniel A. Spielman. Exponential algorithmic speedup by a quantum walk. In *Proceedings of the Thirty-Fifth Annual ACM Symposium on Theory of Computing*, 2003.

13 Andrew M. Childs, Matthew Coudron, and Amin Shiraz Gilani. Quantum Algorithms and the Power of Forgetting. In *14th Innovations in Theoretical Computer Science Conference (ITCS 2023)*, 2023. doi:10.4230/LIPIcs.ITCS.2023.37.

14 Andrew M Childs and Robin Kothari. Quantum query complexity of minor-closed graph properties. *SIAM Journal on Computing*, 41(6):1426–1450, 2012. doi:10.1137/110833026.

15 Arjan Cornelissen. Quantum walks through generalized graph composition. *arXiv preprint*, 2025. doi:10.48550/arXiv.2510.04973.

16 Christoph Dürr, Mark Heiligman, Peter Hoyer, and Mehdi Mhalla. Quantum query complexity of some graph problems. *SIAM Journal on Computing*, 35(6):1310–1328, 2006. doi:10.1137/050644719.

17 Peter Hoyer, Troy Lee, and Robert Spalek. Negative weights make adversaries stronger. In *Proceedings of the Thirty-Ninth Annual ACM Symposium on Theory of Computing*, pages 526–535, 2007.

18 Stacey Jeffery. Quantum subroutine composition. *arXiv preprint*, 2022. doi:10.48550/arXiv.2209.14146.

19 Stacey Jeffery, Robin Kothari, and Frédéric Magniez. Nested quantum walks with quantum data structures. In *Proceedings of the twenty-fourth annual ACM-SIAM symposium on Discrete algorithms*, pages 1474–1485. SIAM, 2013. doi:10.1137/1.9781611973105.106.

20 François Le Gall. Improved quantum algorithm for triangle finding via combinatorial arguments. In *2014 IEEE 55th Annual Symposium on Foundations of Computer Science*, pages 216–225. IEEE, 2014. doi:10.1109/FOCS.2014.31.

21 Troy Lee, Frédéric Magniez, and Miklos Santha. Learning graph based quantum query algorithms for finding constant-size subgraphs. *Chicago Journal of Theoretical Computer Science*, 10:1–21, 2012.

22 Cedric Yen-Yu Lin and Han-Hsuan Lin. Upper bounds on quantum query complexity inspired by the elitzur-vaidman bomb tester. In *Proceedings of the 30th Conference on Computational Complexity*, pages 537–566, 2015. doi:10.4230/LIPIcs.CCC.2015.537.

23 Frédéric Magniez, Ashwin Nayak, Jérémie Roland, and Miklos Santha. Search via quantum walk. In *Proceedings of the thirty-ninth annual ACM symposium on Theory of computing*, pages 575–584, 2007. doi:10.1145/1250790.1250874.

24 Frédéric Magniez, Miklos Santha, and Mario Szegedy. Quantum algorithms for the triangle problem. *SIAM Journal on Computing*, 37(2):413–424, 2007. doi:10.1137/050643684.

25 Michael A Nielsen and Isaac L Chuang. *Quantum computation and quantum information.* Cambridge university press, 2010.

**26** Robert Špalek and Mario Szegedy. All quantum adversary methods are equivalent. *Theory of Computing*, 2(1):1–18, 2006. `doi:10.4086/TOC.2006.V002A001`.

**27** Shengyu Zhang. On the power of ambainis lower bounds. *Theoretical Computer Science*, 339(2-3):241–256, 2005. `doi:10.1016/J.TCS.2005.01.019`.

**28** Yechao Zhu. Quantum query complexity of subgraph containment with constant-sized certificates. *arXiv preprint*, 2011. `arXiv:1109.4165`.