# A Unified Framework of Quantum Walk Search

**Simon Apers** ✉
Université libre de Bruxelles, Brussels, Belgium
CWI, Amsterdam, The Netherlands

**András Gilyén** ✉
California Institute of Technology, Pasadena, CA, USA

**Stacey Jeffery** ✉
QuSoft, CWI, Amsterdam, The Netherlands

## Abstract

Many quantum algorithms critically rely on quantum walk search, or the use of quantum walks to speed up search problems on graphs. However, the main results on quantum walk search are scattered over different, incomparable frameworks, such as the *hitting time framework*, the *MNRS framework*, and the *electric network framework*. As a consequence, a number of pieces are currently missing. For example, recent work by Ambainis et al. (STOC'20) shows how quantum walks starting from the stationary distribution can always *find* elements quadratically faster. In contrast, the electric network framework allows quantum walks to start from an arbitrary initial state, but it only *detects* marked elements.

We present a new quantum walk search framework that unifies and strengthens these frameworks, leading to a number of new results. For example, the new framework effectively *finds* marked elements in the electric network setting. The new framework also allows to interpolate between the hitting time framework, minimizing the number of walk steps, and the MNRS framework, minimizing the number of times elements are checked for being marked. This allows for a more natural tradeoff between resources. In addition to quantum walks and phase estimation, our new algorithm makes use of *quantum fast-forwarding*, similar to the recent results by Ambainis et al. This perspective also enables us to derive more general complexity bounds on the quantum walk algorithms, e.g., based on Monte Carlo type bounds of the corresponding classical walk. As a final result, we show how in certain cases we can avoid the use of phase estimation and quantum fast-forwarding, answering an open question of Ambainis et al.

**2012 ACM Subject Classification** Theory of computation → Graph algorithms analysis; Theory of computation → Quantum complexity theory

**Keywords and phrases** Quantum Algorithms, Quantum Walks, Graph Theory

**Digital Object Identifier** 10.4230/LIPIcs.STACS.2021.6

**Related Version** *Full Version*: `https://arxiv.org/abs/1912.04233`

**Funding** *Simon Apers*: Work done while being partially supported by the CWI-Inria International Lab and the Dutch Research Council (NWO) through QuantERA ERA-NET Cofund project QuantAlgo 680-91-034.
*András Gilyén*: Funding provided by Samsung Electronics Co., Ltd., for the project "The Computational Power of Sampling on Quantum Computers", as well as by the Institute for Quantum Information and Matter, an NSF Physics Frontiers Center (NSF Grant PHY-1733907).
*Stacey Jeffery*: Supported by an NWO Veni Innovational Research Grant under project number 639.021.752, an NWO WISE Grant, and QuantERA project QuantAlgo 680-91-03. SJ is a CIFAR Fellow in the Quantum Information Science Program.



# 1 Introduction

Quantum walk search refers to the use of quantum walks to solve a search problem on a graph. Such algorithms can often be thought of as speeding up a classical algorithm based on a random walk, which makes them relatively easy to design, using only classical intuition. In the last two decades, this topic has received a great deal of attention, with a rich literature attesting to the progress on understanding quantum walk algorithmic techniques [1, 25, 3, 22, 19, 7, 13, 2] and developing applications [11, 21, 16, 8, 9, 23, 17, 14, 18]. Despite this long line of progress, the main results on quantum walk search lie somewhat scattered in different frameworks, and a number of pieces are currently missing.

The quantum walk search frameworks that we consider are the *hitting time framework* originally due to Szegedy [25], the *MNRS framework* due to Magniez, Nayak, Roland and Santha [22], the *electric network framework* due to Belovs [7], and the *controlled quantum amplification framework* by Dohotaru and Høyer [13]. We summarize these frameworks, as well as the corresponding complexities, in Table 1. In this work we unify these different frameworks, leading to a number of new results. For example, algorithms developed using the electric network framework could only *detect* marked elements. Our unified approach can be used to develop algorithms that *find* marked elements, while incurring at most a logarithmic overhead. We also derive quantum speedups with respect to Monte Carlo type bounds on the hitting times, as opposed to the usual Las Vegas type (expectation) bounds.

Our result bridges the conceptual gaps between the recent result of Ref. [2] and the original approaches by Szegedy [25] and Krovi, Magniez, Ozols and Roland [19]. The latter showed that combining quantum walks with phase estimation or time averaging allows one to quadratically improve the hitting time of a single marked element, when starting from the stationary distribution. Ambainis et al. [2] used a novel technique called *quantum fast-forwarding* [6] to improve these results to yield quadratic speedups on the hitting time of arbitrary sets. In this work we show that a similar result can be obtained using only simple quantum walks, thereby proving a conjecture from [2].

## Different Frameworks

While the frameworks we consider are similar, each has advantages and disadvantages. The earliest *hitting time framework* was due to Szegedy [25], inspired by an algorithm of Ambainis for element distinctness [1]. To illustrate this framework, imagine a classical algorithm that begins by sampling a state from the stationary distribution $\pi$ of some random walk, described by a transition matrix $P$. The algorithm starts from a vertex distributed according to $\pi$, and simulates the random walk. After every step of the walk it checks whether the current vertex is "marked". The algorithm terminates after $\mathcal{O}(\mathrm{HT}(P, M))$ steps, with $\mathrm{HT}(P, M)$ the *hitting time*, i.e., the expected number of steps from $\pi$ before a marked vertex in $M$, the marked set, is reached. As such, the algorithm has a constant probability of having found a marked vertex. To bound the complexity of this algorithm, let the *setup cost* $\mathcal{S}$ denote the complexity of sampling from $\pi$, the *update cost* $\mathcal{U}$ denote the complexity of simulating a step of the walk, and the *checking cost* $\mathcal{C}$ denote the complexity of checking whether a vertex is marked. The complexity of the resulting algorithm is then of order $\mathcal{S} + \mathrm{HT}(P, M)(\mathcal{U} + \mathcal{C})$. The hitting time framework essentially shows how to construct a quantum algorithm with complexity

$$\mathsf{S} + \sqrt{\mathrm{HT}(P, M)}(\mathsf{U} + \mathsf{C}),$$

where $\mathsf{S}$, $\mathsf{U}$ and $\mathsf{C}$ are quantum analogues of $\mathcal{S}$, $\mathcal{U}$ and $\mathcal{C}$, respectively, denoting the costs in terms of *coherent* quantum samples. One of the major drawbacks of the original framework was that, while the quantum algorithm could *detect* the presence of a marked vertex, it was not always guaranteed to *find* one. In the special case where there is only a single marked element, Krovi, Magniez, Ozols, and Roland [19] showed how to also find the marked element in the same complexity. To this end they introduced the concept of *interpolated walks*. Combining interpolated walks with another technique called *quantum fast-forwarding*, introduced in [6], Ref. [2] more recently showed how to also find a marked element in the general case. We will refer to this final result as the hitting time framework.

The second framework that we consider is the *MNRS framework* introduced by Magniez, Nayak, Roland and Santha [22]. This framework is always capable of *finding* a marked vertex, but it can be understood as the quantum analogue of a slightly different random walk algorithm. Consider a random walk that begins in the stationary distribution. Rather than checking if the current vertex is marked after every step, the walk takes $1/\delta$ steps between checks, where $\delta$ is the *spectral gap* of $P$. Since $1/\delta$ is approximately the *mixing time* of the random walk, effectively this process repeatedly samples from the stationary distribution until a marked vertex is found during a check. If $\varepsilon$ is the probability that a vertex sampled from the stationary distribution is marked, then a marked element is found with constant probability after $\mathcal{O}(1/\varepsilon)$ samples.[1] As such, the complexity of this classical algorithm is $\mathcal{S} + \frac{1}{\varepsilon}(\frac{1}{\delta}\mathcal{U} + \mathcal{C})$. The MNRS framework shows how to get a quantum algorithm for finding a marked vertex with complexity

$$\mathsf{S} + \frac{1}{\sqrt{\varepsilon}}\Big(\frac{1}{\sqrt{\delta}}\mathsf{U} + \mathsf{C}\Big).$$

Since $\mathrm{HT}(P, M) \leq \frac{1}{\varepsilon\delta}$, this requires at least as many steps of the walk as the hitting time framework. On the other hand, $\mathrm{HT}(P, M) \geq \frac{1}{\varepsilon}$, and so the number of checks can be significantly smaller than in the hitting time framework. In fact, this amount of checks performed in the MNRS framework is easily seen to be optimal by a lower bound on black-box search.[2]

The third framework that we consider is the *electric network framework* by Belovs [7] (published in [8]). This is a generalization of the hitting time framework, allowing for the walker to start from an arbitrary initial distribution $\sigma$ (such as a single vertex), rather than necessarily the stationary distribution. If $\mathsf{S}(\sigma)$ is the complexity of sampling (coherently) from $\sigma$, then the resulting quantum algorithm has complexity

$$\mathsf{S}(\sigma) + \sqrt{C_{\sigma,M}}(\mathsf{U}(\sigma) + \mathsf{C}),$$

where $\mathsf{U}(\sigma)$ is the complexity of implementing a step of a slightly modified random walk. The quantity $C_{\sigma,M}$ (see Section 3.1) is a generalization of the *commute time*. When $\sigma$ is supported on a single vertices $u$, then $C_{\sigma,M}$ equals the commute time from $u$ to $M$, which is the expected number of steps starting from $u$ to reach some $m \in M$ and then return to $u$. When $\sigma$ equals the stationary distribution then $C_{\sigma,M} = HT(P, M)$, thus retrieving the hitting time framework. The obvious advantage of the electric network framework is that it

[1] This yields a trivial classical search algorithm with complexity $(\mathcal{S} + \mathcal{C})/\varepsilon$ which does not walk just uses sampling. By using amplitude amplification this gives rise to a quantum algorithm with complexity $(\mathsf{S} + \mathsf{C})/\sqrt{\varepsilon}$. In case the cost of setup is much larger than that of checking, a random walk that uses the setup only once can be advantageous, potentially leading to interesting quantum walk based speedups.

[2] Consider for instance a quantum walk search algorithm on the complete graph on $N$ vertices. Finding a single marked element then requires $\Omega(\sqrt{N})$ checks by the optimality of Grover's search algorithm.

**Table 1** Comparison of different quantum walk frameworks.

| Framework | Complexity | Finds? |
|---|---|---|
| Hitting time framework [25, 19, 2] | $\mathsf{S} + \sqrt{\mathrm{HT}(P,M)}(\mathsf{U}+\mathsf{C})$ | Yes |
| MNRS framework [22] | $\mathsf{S} + \frac{1}{\sqrt{\varepsilon}}(\frac{1}{\sqrt{\delta}}\mathsf{U}+\mathsf{C})$ | Yes |
| Electric network framework [8, 7] | $\mathsf{S}(\sigma) + \sqrt{C_{\sigma,M}}(\mathsf{U}(\sigma)+\mathsf{C})$ | No |
| Controlled quantum amplification [13] | $\mathsf{S} + \sqrt{\mathrm{HT}(P,\{m\})}\mathsf{U} + \frac{1}{\sqrt{\varepsilon}}\mathsf{C}$ | Yes |

does not necessarily require quantum samples from the stationary distribution of $P$, which might be very costly, and can instead begin in a much easier to produce state. A major disadvantage of this framework, however, is that the quantum algorithm only *detects* the presence of marked vertices, as in the original hitting time framework, rather than actually finding marked vertices.

Finally, we also consider the *controlled quantum amplification framework* by Dohotaru and Høyer [13]. They use an extra qubit to control the quantum walk operator[3], leading to an additional degree of freedom. For the case of a *unique* marked element $M = \{m\}$, and starting from a quantum sample of the stationary distribution, they achieve complexity

$$\mathsf{S} + \sqrt{\mathrm{HT}(P,\{m\})}\mathsf{U} + \frac{1}{\sqrt{\varepsilon}}\mathsf{C},$$

which simultaneously has the lowest number of walk steps (matching the hitting time framework) and the lowest number of checks (matching the MNRS framework). The clear downside of this approach is that it is restricted to cases where there is a single marked element, and one needs to start from the stationary distribution.

## 2 Contributions

### 2.1 Finding in the electric network framework

The electric network framework [7] generalizes the hitting time framework [25] by allowing for arbitrary initial distributions. The downside is that algorithms in this framework only detect rather than actually find marked vertices. On the other hand, the improved hitting time framework of [2] shows how to actually find marked vertices in the hitting time framework, provided that the walk starts from a quantum sample of the stationary distribution. Both works hence provide complementary but incomparable improvements over the initial hitting time framework.

In this work we fill this gap by generalizing the results of [2] to the electric network setting, designing a quantum algorithm that not only detects but also *finds* marked elements for any starting distribution $\sigma$. This improved version strictly generalizes the results of [2], and it loses at most a log factor in the complexity compared to the original electric network framework [7]. In particular, we show the following:

▶ **Theorem 1** (Informal). *For any distribution $\sigma$, there is a quantum walk search algorithm that finds a marked element from $M$ with constant probability in complexity (up to log factors)*

$$\mathsf{S}(\sigma) + \sqrt{C_{\sigma,M}}(\mathsf{U}(\sigma)+\mathsf{C}).$$

[3] In fact they consider more general operators, but we will focus on their result about quantum walk search.

To analyze our new algorithm, we use techniques similar to those employed in [2] for finding in the hitting time framework. However, there is an additional difficulty we must overcome. The hitting time, $\mathrm{HT}(P, M)$, has a useful interpretation in terms of the classical random walk – that is, with high probability, a marked vertex is encountered within the first $\mathcal{O}(\mathrm{HT}(P, M))$ steps – and this fact is crucial in the analysis of the quantum algorithm in [2]. In contrast, to the best of our knowledge, the generalized quantity $C_{\sigma,M}$ is not well understood. If $\sigma$ is supported on a single vertex $u$, and $M$ contains a single vertex, $m$, then $C_{\sigma,M}$ is exactly the *commute time* between $u$ and $m$. This means that within the first $\mathcal{O}(C_{\sigma,M})$ steps, with high probability, a walker starting from $u$ has visited $m$, and then returned to $u$. For general $\sigma$ and $M$, no such interpretation was known. We prove that, under certain conditions, a similar interpretation holds: with high probability, a walker starting from $\sigma$ will hit $M$, and then return to the support of $\sigma$, within the first $\mathcal{O}(C_{\sigma,M})$ steps. We can ensure that the required conditions hold by using a slightly modified walk as in [7], adding a weighted edge to each vertex in $\mathrm{supp}(\sigma)$. The new combinatorial understanding of $C_{\sigma,M}$ then enables us to employ a similar analysis to that of [2].

## 2.2 A Unified Framework

While the electric network framework is a generalization of the hitting time framework, the MNRS framework is incomparable. Since $\mathrm{HT}(P, M) \leq \frac{1}{\varepsilon\delta}$, the hitting time framework always finds a marked vertex using a number of quantum walk steps (updates) less than or equal to that used by the MNRS framework. On the other hand, $\mathrm{HT}(P, M) \geq \frac{1}{\varepsilon}$, and hence the MNRS framework may make fewer calls to the check operation. When the complexity of implementing the checking operation is much larger than that of the update operation, the MNRS framework may hence be preferable to both the hitting time framework and the electric framework. The controlled quantum amplification framework achieves the best of both worlds, but only for a unique marked element.

In this work we present a new framework that unifies all these individual approaches. For the sake of intuition, we first describe this framework when the initial state $\pi$ is used, which can be seen as a unification between the hitting time framework, the MNRS framework and the controlled quantum amplification framework. To this end, recall that the hitting time framework is the quantum analogue of a random walk algorithm that takes $\mathrm{HT}(P, M)$ steps of the random walk described by $P$, checking at each step if the current vertex is marked. In contrast, the MNRS framework is the quantum analogue of a random walk algorithm that takes $\frac{1}{\delta}$ steps of $P$, thus approximately sampling from the stationary distribution $\pi$, and then checks if the sampled vertex is marked. Since $\varepsilon$ is the probability that a sampled vertex is marked, this process is repeated $\frac{1}{\varepsilon}$ times.

We can define a natural interpolation between both classical algorithm. To this end, take any $t$, and consider a classical random walk that repeatedly takes $t$ steps, and then checks whether the current vertex is marked. The expected number of iterations is then $\mathrm{HT}(P^t, M)$, the hitting time of the $t$-step random walk, described by transition matrix $P^t$. This classical algorithm finds a marked vertex in complexity $\mathcal{S} + \mathrm{HT}(P^t, M)(t\mathcal{U} + \mathcal{C})$. We give a quantum analogue of this algorithm, generalized to arbitrary[4] initial distributions.

[4] In case a the initial distribution $\sigma$ differs from $\pi$ we need to account for the complexity $\mathsf{U}(\sigma)$ of the modified update operator, see the full version [5] for details.

▶ **Theorem 2** (Informal). *For any $t \in \mathbb{N}$ and any distribution $\sigma$, there is a quantum walk search algorithm that finds a marked element from $M$ with constant probability in complexity (up to log factors)*

$$\mathsf{S}(\sigma) + \sqrt{C_{\sigma,M}(P^t)}(\sqrt{t}\mathsf{U}(\sigma) + \mathsf{C}).$$

This theorem and the corresponding quantum walk algorithm gives a common generalization of all major previous quantum walk algorithms, and recovers their complexity in the corresponding special cases (up to log factors). Indeed setting $t = 1$ we recover Theorem 1. When $\sigma = \pi$, then $C_{\sigma,M}(P^t) = \mathrm{HT}(P^t, M)$, and hence we find the quantum analogue of the aforementioned random walk algorithm. As such, when $\sigma = \pi$ and $t = 1$, we recover the hitting time framework. When $\sigma = \pi$ and $t = \frac{1}{\delta}$, we recover the MNRS framework, since a $1/\delta$-step random walk essentially samples from $\pi$ at every step, and so $\mathrm{HT}(P^{1/\delta}, M) \in \mathcal{O}\big(\frac{1}{\varepsilon}\big)$. Setting $t = \varepsilon\mathrm{HT}(P, \{m\})$, we recover the controlled quantum amplification framework. To see this, we use a result from [13, Section 6] which proves that $\mathrm{HT}(P^t, \{m\}) = 1/\varepsilon$ if $t \in \Omega(\varepsilon\mathrm{HT}(P, \{m\}))$. For multiple marked elements, and other intermediate values of $t$, we obtain new types of algorithms. We summarize these special cases in the table below.

**Table 2** The new quantum walk search framework.

| New quantum walk search framework: | $\mathsf{S}(\sigma) + \sqrt{C_{\sigma,M}(P^t)}(\sqrt{t}\mathsf{U}(\sigma) + \mathsf{C})$ |
|---|---|
| Hitting time framework | $\sigma = \pi$, $t = 1$ |
| MNRS framework | $\sigma = \pi$, $t = \frac{1}{\delta}$ |
| Electric network framework | any $\sigma$, $t = 1$ |
| Controlled quantum amplification | $\sigma = \pi$, $M = \{m\}$, $t = \varepsilon\mathrm{HT}(P, M)$ |

### An improved application: backtracking

Electric network based quantum walks have several applications, but arguably the most well-known application is Montanaro's quantum speedup for backtracking algorithms [23]. Montanaro's algorithm uses quantum walk based search in order to find a marked vertex in a bounded degree tree structure, with an unknown structure, starting from the root. The binary tree corresponds to a search tree typically coming from a constraint satisfaction problem. Since already partial assignments can lead to unsatisfiable constraints, the corresponding parts of the search tree are pruned in order to increase performance. This means that the graph on which the walk is performed is a priori unknown, but can be explored locally.

Since earlier versions of the electric network based walks did not always allow finding a marked element, Montanaro needed to construct a modified algorithm that could always find a marked vertex by directly exploiting the special tree structure. If the search tree has size $T$, and the depth of the tree is at most $n$, then Montanaro's algorithm finds a marked vertex in time

$$\widetilde{\mathcal{O}}\Big(\sqrt{T}n^{3/2}\log(n)\Big).$$

In order to understand the performance of our walk for finding a marked element, we need to understand the quantities in Theorem 2. Since we have a simple graph, we can assume that every edge has weight 1 initially, and $C_{r,M} = TR_{r,M}$, where $T$ is the number of edges in the tree, and $R_{r,M}$ is the effective resistance between the root $r$ and the set marked vertices $M$ (cf. Section 3.1). Due to Theorem 2 (for more formal details, see [5]) we can

find a marked element in complexity $\mathcal{O}\Big(\sqrt{TR_{r,M}\log(TR_{r,M})\log\log(TR_{r,M})}\Big)$. If there is a marked element (or solution), then there is a path of length $\leq n$ for m the root to a marked element, which means that the effective resistance is $R_{r,M} \leq n$. Also by using $\log(T) \in \mathcal{O}(n)$ we can see that a corollary of our results is that we can find a marked element in complexity

$$\mathcal{O}\Big(\sqrt{T\log(n)n}\Big),$$

which gives a $\sqrt{n\log(n)}$ factor improvement over the result of Montanaro.

For comparison, we shall note that another improvement over Montanaro's algorithm was achieved by [15]. Their algorithm has complexity

$$\mathcal{O}\Big(\sqrt{TR_{\max}}\log^4(kR_{r,M})\Big), \tag{1}$$

where $R_{\max}$ is the maximal finite ($< \infty$) effective resistance over subtrees of the graph, and $k$ is the number of marked vertices. If there are $\mathcal{O}(1)$ marked vertices, then the complexity (1) can be bounded as $\widetilde{\mathcal{O}}\Big(\sqrt{Tn}\log^4(n)\Big)$, which is asymptotically even lower than our bound. However, the interplay between $R_{\max}$, $R_{r,M}$, $n$ and $k$ suggest that our complexity will be typically lower when there is a fair number ($> 2^{\sqrt[4]{n\log(n)}}$) of marked vertices.

Finally, we note that another improvement over Montanaro's algorithm was suggested by Ambainis and Kokainis [4], employing a clever binary search on the size of the search subtrees $T$ in order to improve the overall runtime. This improvement relies on adaptively modifying the search graph, rather than optimizing the walk on a fixed graph, and therefore this approach does not directly fit into our framework.

## 2.3 Bounds based on Monte Carlo type guarantees of the classical walk

We study quantum walk speedups with respect to Monte Carlo type hitting time bounds of the corresponding classical random walks. Monte Carlo type hitting times can be upper bounded in terms of the Las Vegas hitting time of the same walk, but can be also substantially smaller (see the full version, [5], for an explicit example). Therefore, a quadratic speedup in terms of Monte Carlo type hitting times is preferable compared to the usual Las Vegas bounds.

Monte Carlo type hitting time bounds and quadratic speedups for the detection problem were studied in [20], however, to our knowledge quadratic speedups for finding a marked vertex were only established for vertex-transitive graphs with a unique marked vertex. Using our techniques we are able to show that a quadratic speedup (up to a log factor) for finding a marked vertex can be always achieved for any random walk on undirected weighted graphs.

We define the *Monte Carlo hitting time for probability* $p$, denoted $\mathrm{HT}_p(P, M)$, as the minimum number of time-steps $t$, such that $P$ visits a marked vertex within the first $t$ steps with probability at least $p$, when started from the stationary distribution $\pi$.

▶ **Theorem 3.** *Given an upper bound* $\mathrm{HT}_p(P, M) \leq \mathrm{HT}_p$ *on the Monte Carlo hitting time of a reversible Markov Chain* $P$*, there is a quantum algorithm that finds a marked vertex with high probability in complexity*

$$\sqrt{\frac{\log(\mathrm{HT}_p)}{p}}\left(\mathsf{S} + \sqrt{\mathrm{HT}_p\log(\log(\mathrm{HT}_p)/p)}(\mathsf{U}+\mathsf{C})\right).$$

Versions of the above theorem including the trade-off analogous to Theorem 2 also easily follow from our framework. Additionally, in the full version, we also prove some related Monte Carlo type bounds for the general case when we start in an arbitrary distribution $\sigma$ as opposed to the stationary.

## 2.4 Simpler algorithm for hitting time and electric network framework

Similar to the recent work by Ambainis et al. [2], our new quantum algorithm makes use of a somewhat involved technique called quantum fast-forwarding. For the case $t = 1$ (recovering the hitting time and electric network framework), we show that our algorithm can be much simplified while maintaining essentially the same complexity guarantees. The simple algorithm works by (classically) choosing random interpolation parameters, and applying the interpolated quantum walk operator for an appropriately chosen (random) number of steps, starting from $|\sqrt{\sigma}\rangle$, the quantum state whose amplitudes are the square roots of the probabilities of $\sigma$. It was already conjectured in [2] that such a simple approach would work (for the special case of the hitting time framework). In this work we prove that this simple algorithm indeed finds a marked vertex, and incurs at most a logarithmic overhead over the complexity of the more involved fast-forwarding algorithm. Interestingly, our proof relies on the proof of correctness of the fast-forwarding algorithm.

## 2.5 Related independent work

While finalizing this manuscript, the authors became aware of the concurrent and independent work of Stephen Piddock, who developed an alternative refinement of Belovs' results for finding marked elements in the electric network framework [24].

# 3 Summary of technical contributions

We now give a high-level summary of some of our technical results. For the sake of intuition, we suppose that the random walk $P$ is symmetric – this assumption is not necessary for our results, but serves to simplify the overview of this section. For the fully detailed statements and proofs of our results, see the technical version of this article [5].

## 3.1 Finding in the electric network framework

The electric network framework is almost a strict generalization of the hitting time framework, except that in the hitting time framework we know how to not only detect, but actually find a marked vertex, whereas the electric network framework only detects the presence of a marked vertex. Our first contribution is to show how to find in the electric network framework. The resulting framework thus generalizes both the hitting time and electric network frameworks.

### Interpolated walks

Letting $P$ be the transition matrix of a random walk, and $s \in [0, 1]$ an *interpolation parameter*, we define $P(s)$ as the transition matrix of the *interpolated walk* – the random walk that acts as $P$, except that when a marked vertex is encountered, with probability $s$, the walker remains in that vertex, and with probability $1 - s$, the walker transitions according to $P$. That is, if we let $P_M$ denote the *absorbing walk* that behaves as $P$, except that for any vertex $u \in M$, a walker at $u$ remains there with probability 1, then we can write[5]

$$P(s) = (1 - s)P + sP_M.$$

[5] Note that even if $P$ was symmetric, the interpolated walk $P(s)$ can be non-symmetric, however it is reversible. Fortunately, our results generalize to any reversible (non-symmetric) Markov Chain as well.

Note that if $s$ is sufficiently close to 1, after $10\mathrm{HT}(P,M)$ steps of $P(s)$ starting from the stationary distribution $\pi$, the walker will be in a marked vertex with high probability. In other words, letting $\Pi_M$ denote the orthogonal projector onto marked vertices, $\Pi_M P(s)^{10\mathrm{HT}(P,M)}\pi$ will be large in $\ell_1$-norm.

If $\mathsf{U}(P)$ is the complexity of taking one step of the random walk $P$, and $\mathsf{C}$ is the complexity of checking if a vertex is marked, then one step of the random walk $P(s)$ can be implemented in complexity $\mathsf{U}(P(s)) = \mathcal{O}(\mathsf{U}(P) + \mathsf{C})$.

### Quantum fast-forwarding and finding in the hitting time framework

The original hitting time framework also suffered from the drawback that it could detect but not find marked vertices, until this was recently improved in Ref. [2]. This improvement was based on a technique called *quantum fast-forwarding* [6], which shows how, for a symmetric random walk $P$, to map any state $|\psi\rangle$ to within $\xi$-distance of some state of the form $|0\rangle P^t|\psi\rangle + |1\rangle|\gamma\rangle$, where $\||\gamma\rangle\|^2 = 1 - \|P^t|\psi\rangle\|^2$, in complexity $\mathcal{O}\Big(\sqrt{t\log\frac{1}{\xi}}\mathsf{U}(P)\Big)$.

The algorithm of Ref. [2] first checks if the initial state is marked, then applies fast-forwarding to the absorbing walk $P(s)$ for some appropriately chosen $s$, using $|\sqrt{\pi}\rangle$ as the initial state, resulting in a state of the form $|0\rangle P(s)^t(I-\Pi_M)|\sqrt{\pi}\rangle + |1\rangle|\gamma\rangle$ for $t \approx \mathrm{HT}(P,M)$, using roughly $\sqrt{\mathrm{HT}(P,M)}$ calls to the update operator for $P(s)$, for a total cost of about $\sqrt{t}\mathsf{U}(P(s)) \approx \sqrt{\mathrm{HT}(P,M)}(\mathsf{U}(P)+\mathsf{C})$. The probability that measuring this state results in a marked vertex is $\|\Pi_M P(s)^t(I-\Pi_M)|\sqrt{\pi}\rangle\|^2$. Ref. [2] lower bounds this probability by the square of the probability that a random walk, beginning from the distribution $\pi$, is in a marked vertex after $t$ steps, and then in an unmarked vertex again after $t'$ steps, for *any* $t' > t$. That is, for all $t' > t$:

$$\left\|\Pi_M P(s)^t(I-\Pi_M)|\sqrt{\pi}\rangle\right\|^2 \geq \Pr_{Y_0\sim\pi}(Y_0 \notin M, Y_t \in M, Y_{t'} \notin M)^2. \tag{2}$$

Then all that remains is to prove a statement about the classical random walk: for some appropriately chosen $s$, and some value $t' > t$, with high probability, after $t \approx \mathrm{HT}(P,M)$ steps, a random walker is in a marked vertex – easy to achieve by taking $s$ to be sufficiently close to 1 – and after $t'$ steps, the walker is not in a marked vertex – necessitating that $s$ is not *too* close to 1. The proof of this, while not straightforward, relies only on combinatorial arguments about the classical random walk.

### The electric network framework

A similar statement to Equation (2) can be proven for the electric network framework, but utilizing it requires significantly more work in understanding the classical random walk.

The electric network framework can be understood in analogy to a classical random walk that begins in an arbitrary distribution $\sigma$, and walks until a marked vertex is found. The quantum analogue of this process begins in the quantum state $|\sqrt{\sigma}\rangle = \sum_u \sqrt{\sigma_u}|u\rangle$, and takes $\sqrt{C_{\sigma,M}}$ steps of a random walk $P'(s)$ that is similar to an interpolated walk $P(s)$, except that each vertex $u \in \mathrm{supp}(\sigma)$ has an extra edge, connecting it to a new degree-1 vertex, with weight proportional to $\sigma_u$, see Figure 1 (and the full version, [5], for more explanation).

Using fast-forwarding, we can find a marked vertex with probability $\|\Pi_M P'(s)^t|\sqrt{\sigma}\rangle\|^2$, in complexity $\mathsf{S}(\sigma) + \sqrt{t}(\mathsf{U}+\mathsf{C})$ as in [2]. Analogous to Equation (2), we can show that for any $t' > t$, this probability is lower bounded by:

$$\left\|\Pi_M P'(s)^t|\sqrt{\sigma}\rangle\right\|^2 \geq \Pr_{Y_0\sim\sigma}(Y_t \in M, Y_{t'} \in \mathrm{supp}(\sigma))^2. \tag{3}$$

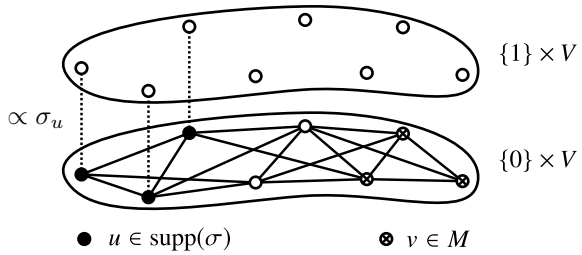


**Figure 1** Modified graph.

Taking $t \approx C_{\sigma,M}$ to be roughly the commute time, we get an algorithm with the right complexity, and it remains only to show that there is some appropriate choice of $s$, and some value $t' > t$ such that with high probability, a random walk beginning in the distribution $\sigma$ is in a marked state after $C_{\sigma,M}$, and has returned to the support of $\sigma$ after $t'$ steps. This requirement on returning to the support of $\sigma$ sheds a light on why the *commute* time is the relevant quantity for the analysis of the quantum walk. Although, the commute time has some intuitive combinatorial meaning in some special cases, for example when $\sigma$ is supported on a single vertex, in the general case it is difficult to grasp the intuitive interpretation of $C_{\sigma,M}$. Our main technical contribution here is to give such an interpretation, that can then be used to lower bound the expression in Equation (3).

### Interpretation of $C_{\sigma,M}$

A weighted graph can be thought of as modeling an electrical network, where an edge $e$ of weight $w_e$ represents a resistor with resistance $1/w_e$ (or equivalently, conductance $w_e$). The effective resistance $R_{\sigma,M}$ denotes the minimum energy of any *unit flow* entering the graph with current $\sigma_u$ at vertex $u$, and exiting the graph through the vertices of $M$. Then if $W$ denotes the *total weight* of all edges in the graph, we define $C_{\sigma,M} = WR_{\sigma,M}$. To motivate this perhaps strange-seeming definition, we highlight two special cases:

- When $\sigma = \pi$ is the stationary distribution of the graph, $C_{\sigma,M} = \mathrm{HT}(P, M)$.
- When $\sigma$ is supported on a unique vertex $s$, and $M = \{t\}$ is a singleton, then $C_{\sigma,M}$ is the *commute time* between $s$ and $t$, or the expected number of steps starting from $s$ to reach $t$ and then return to $s$.

However, to the best of our knowledge, no such operational interpretation of $C_{\sigma,M}$ in the general case is known. In order to lower bound the expression in Equation (3), we would like to be able to say that, for appropriately chosen parameter $s$, with high probability, after $t \approx C_{\sigma,M}$ steps, a random walker beginning in distribution $\sigma$ will be in a marked vertex, and after sufficiently many more steps, the random walker will have returned to the support of $\sigma$. In the special cases outlined above, there is intuitive reason to believe that such a statement should hold, but in the general case, there is no interpretation that would lead us to believe that such a statement is true.

We show that, under certain assumptions (that are always satisfied for the modified graphs described above, cf. Figure 1), we can interpret $C_{\sigma,M}$ as a kind of commute time between the support of $\sigma$ and the marked set $M$. In particular, in the full version, [5], we prove the following, which is sufficient to give a lower bound on Equation (3):

▷ Claim 4 (Informal). Whenever $\sigma$ is proportional to $\pi$ on its support, and $\pi(\mathrm{supp}(\sigma)) \approx \frac{1}{C_{\sigma,M}}$, with high probability, a random walk starting from $\sigma$ first hits $M$ and then returns to $\mathrm{supp}(\sigma)$ within $10C_{\sigma,M}$ steps.

The condition that $\sigma$ be proportional to $\pi$ on its support – that is, there exists $\alpha$ such that $\sigma_u = \alpha\pi_u$ whenever $\sigma_u \neq 0$ – may seem strong, but we can satisfy it by modifying the graph by adding an edge to each $u \in \mathrm{supp}(\sigma)$ to a new vertex of degree 1, with edge weight proportional to $\sigma_u$. If we call this new vertex $u$ (and appropriately rename the vertex formerly called $u$), then this change has negligible impact on the dynamics of a random walk starting from $\sigma$, but it ensures that the stationary probability of each of these new vertices $u$ is proportional to $\sigma_u$. This graph modification is already made in the original electric network framework, so we automatically satisfy the first condition required for the claim. The second condition is satisfied by appropriately scaling the weights of the new edges.

## 3.2 A unified framework

Once we show how to find in the electric network framework, it is a strict generalization of the hitting time framework. However, it is still incomparable to the MNRS framework. Our next contribution is to give a new framework that captures both the MNRS framework and the electric network framework as a special case, and also recovers the main application of the controlled quantum amplification framework regarding the optimized checking cost.

The main idea of the unified framework is to apply the electric framework machinery, but to the Markov chain that performs $t$-steps of $P$ at once. The main technical challenge is to perform $t$ consecutive update steps of the walk with using the update unitary only about $\sqrt{t}$ times. This can be performed using fast-forwarding, but we need to be careful, because after fast-forwarding we still need to apply the modifications to the walk required by the algorithm for the electric network framework.

Fortunately, we are able to show that these operators, namely interpolating the walk and modifying the graph on the support of $\sigma$, are compatible with fast-forwarding, if one considers a suitable and general-enough notion of the update operation. To show this we need to view the quantum walk operators from an angle which is different from the perspective of earlier quantum walk results, that did not consider fast-forwarding. To be able to grasp and extract the essential features of the update operator, we view the update operator as a block-encoding of the Markov chain (or more precisely its discriminant matrix). Then we describe the effects of the desired modifications using this formalism and then show how to implement them.

## 3.3 A plain quantum walk algorithm for finding marked elements

As we described above, our main quantum walk algorithm and its analysis relies on the use of quantum fast-forwarding. This makes it more complicated than the original "plain" quantum walk algorithms in e.g. [25, 22, 7]. We nevertheless show that the correctness of our algorithm implies the correctness of a much simpler algorithm, at least in the regimes corresponding to the hitting time framework and the electric network framework. Namely, if we simply pick a random interpolation parameter, run the corresponding plain quantum walk for at most about $\sqrt{C_{\sigma,M}}$ steps, and finally measure the walk register, then we find a marked element with constant probability. This algorithm was proposed in [2, Section 4] for the hitting time framework, but was only conjectured to be correct.

To derive this result, we literally "dissect" the more complicated fast-forwarding algorithm by considering an explicit construction of the quantum fast-forwarding routine based on the linear combination of unitaries technique [12, 10]. The structural properties of this quantum circuit then imply that the simpler routine should also succeed. Indeed, quantum fast-forwarding can be represented as a convex combination of plain quantum walk steps,

which makes it equivalent to a stochastic algorithm which picks a random iteration number $t$ according to the convex combination coefficients, and then runs the plain quantum walk for $t$ steps. To illustrate this, in the full version, Ref. [5], we give a proof of the correctness of the fast-forwarding technique, and analyze the structure of the constructed circuit.

## 3.4 Open questions

Our unified framework resolves most open questions about quantum walk based search, and recovers essentially all major prior upper bounds (up to log factors). However, there could be room for improving the "Monte Carlo type" bounds in the case of arbitrary initial distributions. Also in the case of arbitrary initial distributions we lack a good combinatorial intuition for the quantity $C_{\sigma,M}$ appearing in our upper bounds. Finally, we lack a good general lower bound matching our upper bounds in terms of $\mathsf{U}$ and $\mathsf{C}$ simultaneously.

## References


1 Andris Ambainis. Quantum walk algorithm for element distinctness. In *Proceedings of the 45th IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 22–31, 2004. arXiv:quant-ph/0311001, doi:10.1109/FOCS.2004.54.

2 Andris Ambainis, András Gilyén, Stacey Jeffery, and Martins Kokainis. Quadratic speedup for finding marked vertices by quantum walks. In *Proceedings of the 52nd ACM Symposium on the Theory of Computing (STOC)*, page 412–424, 2020. arXiv:1903.07493, doi:10.1145/3357713.3384252.

3 Andris Ambainis, Julia Kempe, and Alexander Rivosh. Coins make quantum walks faster. In *Proceedings of the 16th ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 1099–1108, 2005. arXiv:quant-ph/0402107.

4 Andris Ambainis and Martins Kokainis. Quantum algorithm for tree size estimation, with applications to backtracking and 2-player games. In *Proceedings of the 49th ACM Symposium on the Theory of Computing (STOC)*, page 989–1002, 2017. arXiv:1704.06774, doi:10.1145/3055399.3055444.

5 Simon Apers, András Gilyén, and Stacey Jeffery. A unified framework of quantum walk search, 2019. arXiv:1912.04233.

6 Simon Apers and Alain Sarlette. Quantum fast-forwarding: Markov chains and graph property testing. *Quantum Information and Computation*, 19(3&4):181–213, 2019. arXiv:1804.02321, doi:10.26421/QIC19.3-4.

7 Aleksandrs Belovs. Quantum walks and electric networks, 2013. arXiv:1302.3143.

8 Aleksandrs Belovs, Andrew M. Childs, Stacey Jeffery, Robin Kothari, and Frédéric Magniez. Time-efficient quantum walks for 3-distinctness. In *Proceedings of the 40th International Colloquium on Automata, Languages, and Programming (ICALP)*, pages 105–122, 2013. doi:10.1007/978-3-642-39206-1_10.

9 Daniel J. Bernstein, Stacey Jeffery, Tanja Lange, and Alexander Meurer. Quantum algorithms for the subset-sum problem. In *Proceedings of the 5th International Conference on Post-Quantum Cryptography (PQCrypto)*, pages 16–33, 2013. doi:10.1007/978-3-642-38616-9_2.

10 Dominic W. Berry, Andrew M. Childs, and Robin Kothari. Hamiltonian simulation with nearly optimal dependence on all parameters. In *Proceedings of the 56th IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 792–809, 2015. arXiv:1501.01715, doi:10.1109/FOCS.2015.54.

11 Harry Buhrman and Robert Špalek. Quantum verification of matrix products. In *Proceedings of the 17th ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 880–889, 2006. arXiv:quant-ph/0409035.

**12** Andrew M. Childs and Nathan Wiebe. Hamiltonian simulation using linear combinations of unitary operations. *Quantum Information and Computation*, 12(11&12):901–924, 2012. arXiv:1202.5822, doi:10.26421/QIC12.11-12.

**13** Cătălin Dohotaru and Peter Høyer. Controlled quantum amplification. In *Proceedings of the 44th International Colloquium on Automata, Languages, and Programming (ICALP)*, pages 18:1–18:13, 2017. doi:10.4230/LIPIcs.ICALP.2017.18.

**14** Alexander Helm and Alexander May. Subset sum quantumly in $1.17^n$. In *Proceedings of the 13th Conference on the Theory of Quantum Computation, Communication, and Cryptography (TQC)*, pages 5:1–5:15, 2018. doi:10.4230/LIPIcs.TQC.2018.5.

**15** Michael Jarret and Kianna Wan. Improved quantum backtracking algorithms using effective resistance estimates. *Physical Review A*, 97(2):022337, 2018. arXiv:1711.05295, doi:10.1103/PhysRevA.97.022337.

**16** Stacey Jeffery, Robin Kothari, and Frédéric Magniez. Nested quantum walks with quantum data structures. In *Proceedings of the 24th ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 1474–1485, 2012. arXiv:1210.1199.

**17** Ghazal Kachigar and Jean-Pierre Tillich. Quantum information set decoding algorithms. In *Proceedings of the 8th International Conference on Post-Quantum Cryptography (PQCrypto)*, pages 69–89, 2017. arXiv:1703.00263, doi:10.1007/978-3-319-59879-6_5.

**18** Elena Kirshanova. Improved quantum information set decoding. In *Proceedings of the 9th International Conference on Post-Quantum Cryptography (PQCrypto)*, pages 507–527, 2018. arXiv:1808.00714, doi:10.1007/978-3-319-79063-3_24.

**19** Hari Krovi, Frédéric Magniez, Maris Ozols, and Jérémie Roland. Quantum walks can find a marked element on any graph. *Algorithmica*, 74(2):851–907, 2016. arXiv:1002.2419, doi:10.1007/s00453-015-9979-8.

**20** Frédéric Magniez, Ashwin Nayak, Peter C. Richter, and Miklos Santha. On the hitting times of quantum versus random walks. *Algorithmica*, 63(1):91–116, 2012. Earlier version in SODA'09. arXiv:0808.0084 doi:10.1007/s00453-011-9521-6.

**21** Frédéric Magniez, Miklos Santha, and Mario Szegedy. Quantum algorithms for the triangle problem. *SIAM Journal on Computing*, 37(2):413–427, 2007. arXiv:quant-ph/0310134, doi:10.1137/050643684.

**22** Frédéric Magniez, Ashwin Nayak, Jérémie Roland, and Miklos Santha. Search via quantum walk. *SIAM Journal on Computing*, 40(1):142–164, 2011. Earlier version in STOC'07. arXiv:quant-ph/0608026 doi:10.1137/090745854.

**23** Ashley Montanaro. Quantum-walk speedup of backtracking algorithms. *Theory of Computing*, 14(15):1–24, 2018. arXiv:1509.02374, doi:10.4086/toc.2018.v014a015.

**24** Stephen Piddock. Quantum walk search algorithms and effective resistance, 2019. arXiv:1912.04196.

**25** Mário Szegedy. Quantum speed-up of Markov chain based algorithms. In *Proceedings of the 45th IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 32–41, 2004. arXiv:quant-ph/0401053, doi:10.1109/FOCS.2004.53.