Review on Quantum Walk Algorithm

Wenda Zhou1, *

1School of Electronics Engineering and Computer Science, Peking University, Beijing, 100871, China.

*Corresponding author e-mail: zhouwenda@pku.edu.cn

Abstract. Quantum walk is the quantum version of classical random walk, which has shown great advantage over classical algorithms. In this article, main discoveries and breakthroughs along with several applications in the last two decades are reviewed and discussed. Reducing hitting time and diminishing mixing time, as two hot research fields, are addressed. Possibility of universal computation with more generic case of quantum walk is also mentioned. In the future, we can expect new methods and models being feasible ways to approach the theoretical lower bounds of related problems, as well as more efficient applications on specific scene.

1. Introduction
Quantum computation is an interdisciplinary scientific field that exploits the property of quantum mechanism to build quantum computers and design specific quantum algorithms. It has been shown that quantum computation outperforms classical algorithm in many specific areas. For example, Shor proposed an efficient quantum factorization algorithm in 1994, which is dramatically faster than any classical solution [1]. Although no general-purpose quantum computer has been invented yet, research on quantum algorithm has been carried on for the last few decades.

Random walk is a frequently used and powerful computation model in mathematics, natural science and computer science. For instance, Google's search engine is based on PageRank algorithm [2], which performs a random walk on the graph with vertices representing different web pages. Quantum walk is the quantum counterpart of classical random walk. By utilizing the property of quantum superposition, quantum walk sometimes achieves exponential algorithmic speedup over classical computers [3]. Specific surveys on quantum walk have been made by Ambainis [4] and Kempe [5], however, many years have passed and there have been lots of new discoveries and some of the open problems have been well studied and solved. In this article, we will review the progress made in the last two decades.

The structure of this article is as follows: In Section 2, the effort done to speed up search procedure is reviewed, which is mainly related with the quadratically faster hitting time of quantum walk; In Section 3, another advantage of quantum walk is presented, i.e., faster mixing time along with its application in sampling; The value for universal computation is mentioned in Section 4 while some concrete examples of applying quantum walk to specific problems are given in Section 5; and finally, the possible development in the future are discussed and demonstrated in Section 6.

2. Search and Hitting time
One of the most important concepts in classical Markov chain theory is hitting time. Given initial state $x$ and marked state $y$, the hitting time $H(x, y)$ is defined as the expected number of steps before visiting $y$. Denote $M$ as the set of marked nodes, average hitting of random walk with respect to $M$ is defined as:
Hitting time is closely related to search problems, because smaller hitting time implies the possibility of approaching target faster. In this section, the faster-hitting property of quantum walk and its contribution in search procedure speed up are discussed.

2.1. Quantum walk on hypercube

In order to validate the feasibility of quantum walk, Shenvi, Kempe and Whaley [6] presented a discrete quantum walk algorithm for the search problem where \( N = 2^n \) elements were arranged in an \( n \)-dimensional hypercube. As same as classical random walk, the state of coin (a variable deciding next move) and walker's current state (probability distribution) should be recorded. A \( n \)-bit long coin register and a \( 2^n \)-bit long state register is used to store the information. In this way, entire quantum walk is built on the Hilbert space \( \mathcal{H} = \mathcal{H}_C \otimes \mathcal{H}_S \), where the former Hilbert space is related with the quantum coin, and the latter is associated with the vertices on the hypercube. Discrete time random walk can be seen as repeatedly applying sequence of unitary operator \( U \), which can be written as:

\[
U = S \cdot C
\]  

(2)

\( S \) is the unitary permutation operator deciding next state of state register based on the coin register. \( C \) is a unitary coin-tossing transformation on coin register. It will not directly change the state register, which means \( C \) is equivalent to an identity operator \( I \) acting on the state register. \( C \) can be decomposed into:

\[
C = C_0 \otimes I
\]  

(3)

Grover diffusion operator \( G \) [7] is a frequently chosen option for \( C_0 \), i.e.,

\[
C_0 = G = -I + 2|s_C⟩⟨s_C|
\]  

(4)

where \( |s_C⟩ \) is an equal unit superposition over all \( n \) directions.

2.1.1. Coin oracle

In general search problems circuits, an operator named oracle is usually used (see [8] for introduction). The oracle computes a function which will output 1 if the target has been found and 0 otherwise. The number of oracles needed is used to analyze the complexity of the algorithm. A similar coin oracle is utilized for search problem on hypercube, which can be easily derived from standard oracle as illustrated in Figure 1. A standard oracle \( U_f \) is applied to the state register to decide whether the marked node has been arrived. The answer is placed on the middle wire to manipulate the controlled coin operator. If the marked node has not been found, \( |f(x)⟩ \) is \( |0⟩ \), \( C_0 \) will be applied to coin register. Otherwise, \( |f(x)⟩ \) is \( |1⟩ \), another arbitrary operator \( C_1 \) is applied. The operator \( C \) in equation (2) turns into \( C' \), and \( U \) turns into \( U' = S \cdot C' \).

![Figure 1. Modified standard oracle that simulates the coin oracle [6].](image)

The standard oracle acts on the node space as \( U_f: |x⟩\otimes|y⟩ \rightarrow |x⟩\otimes|y\oplus f(x)⟩ \), the controlled coin operation, denoted by \( C_0/C_1 \), applies \( C_0 \) on the coin space if the control qubit is in the state \( |0⟩ \), and
3

\( C_1 \) if it is in the state \(|1\rangle\).

2.1.2. Search algorithm

Shenvi, Kempe and Whaley [6]'s algorithm can be described as follows: start from the equal superposition over all states in Hilbert space \( \mathcal{H} \), then repeatedly apply the \( U' \) for \( \frac{\pi}{2} \sqrt{\frac{N}{2}} \) times, where \( C_0 = G, \ C_1 = -I \). After that measuring the state of walker yields the marked node with probability \( \frac{1}{2} - O\left(\frac{1}{n}\right) \). Since the oracle \( U' \) is needed applying for \( \frac{\pi}{2} \sqrt{\frac{N}{2}} \) times, while the probability approximates constant \( \frac{1}{2} \), the complexity is \( O(\sqrt{N}) \), which achieved the same order of efficiency as Grover’s search algorithm [7].

To sum up, earliest study on quantum walk is quite intuitive. Imitating classical random walk, scientists addressed the problem in two independent Hilbert spaces and substantiate one quantum walk step as the unitary operator. If someone measures the state register and coin register after each step, the superposition will collapse and the algorithm then degrades to a classical random walk.

The controlled coin operation in coin oracle allows different walker behavior on marked nodes, which offers the possibility of undergoing specific tasks. Their algorithm, along with many later algorithms can be classified into stochastic algorithms. That is, single measurement may not give the correct answer, nevertheless, by repeating the procedure several times, the probability of getting a wrong answer can be diminished and controlled below the threshold.

2.2. Quadratic speed-up

A generic method for quantizing classical random walks was proposed by Szegedy [9]. Even though his proof is based on bipartite walk, the result can be generalized to ordinary graphs, since every graph can be mapped onto a bipartite graph. \( X \) and \( Y \) are two finite sets representing two parts of the graph. \( P = (p_{x,y}) \) and \( Q = (q_{x,y}) \) are the transition matrix between \( X \) and \( Y \). On the Hilbert space with basis states \( \{|x\rangle|y\rangle : x \in X, y \in Y\} \), two superpositions \( \phi_x = \sum_{y \in Y} \sqrt{p_{x,y}} |x\rangle|y\rangle \) and \( \psi_y = \sum_{x \in X} \sqrt{q_{y,x}} |x\rangle|y\rangle \) are defined. \( A = (\phi_x)_x \) and \( B = (\psi_y)_y \) are two matrices composed of column vectors \( \phi_x \) and \( \psi_y \).

\[
\text{ref}_1 = 2AA^\dagger - I, \text{ref}_2 = 2BB^\dagger - I \tag{5}
\]

Desired quantization is defined by the product \( W_p = \text{ref}_2 \text{ref}_1 \). Denote \( M \) as the set of marked nodes, hitting time \( \mathcal{H}(W_p, M) \) of quantum walk \( W_p \) with respect to \( M \) is defined as the number of steps \( T \) such that

\[
\mathcal{H}(W_p, M) = \frac{1}{T+1} \sum_{t=0}^{T} |W_p^t \phi_0 - \phi_0|^2 \geq 1 - \frac{|M|}{|x|} \tag{6}
\]

where \( \phi_0 = \frac{1}{\sqrt{n}} \sum_{x,y} \sqrt{p_{x,y}} |x\rangle|y\rangle \) is an equal superposition over all possible states, while \( P' \) is the transition matrix derived from \( P \) as \( p'_{x,y} = \delta_{x,y} \) for \( x \in M \).

Szegedy verified that the hitting time of an ergodic Markov chain with symmetric transition matrix is in the order of square root of average hitting time in classical random walk, i.e.,

\[
\mathcal{H}(W_p, M) \in O\left(\frac{1}{n} \sum_{x \in X} H(x, M)\right) \tag{7}
\]

Applying the result on classical result for the \([n]^d \) torus, Szegedy got the results, as displayed in Table 1. Szegedy’s work gave out not only a theoretical define of hitting time but also a method of generalize quantum walk. Though his research mainly focused on symmetric walks, which is only a specific case of random walk, his work is thought to be a milestone.
Table 1. Classical result for the [n]^d torus and the result after Szegedy’s “quantize” [9].

| Dimension | Average Hitting Time | Hitting Time |
|-----------|----------------------|--------------|
| 1         | n^2                  | n            |
| 2         | n^2 log n            | n sqrt log n |
| d ≥ 3     | n^d                  | n^d/2        |

2.3. Finding a marked element

Szegedy’s method can decide the existence of marked elements with quadratic speed-up, however fails to find a marked element with the same complexity, since the probability of measuring a marked element is $\Omega(n/h)$, where $h$ is average hitting time of corresponding random walk [9].

Magniez et al. provided an algorithm for finding a marked element [10]. Furthermore, they extended the scope of algorithm to Markov chains with possibly non-symmetric transition matrix, by replacing $Q$ with $P^* = (p_{x,y}^*)$, (In the work of Szegedy, Q is equal to P), the time-reversed Markov chain corresponding to $P$, which by define is:

$$\pi_x P_{xy} = \pi_y P^*_{y,x} \quad (8)$$

Phase estimation was used to create a quantum circuit that approximates the reflection operator $ref$. Meanwhile, a recursive amplitude amplification algorithm, which fits the situation where imperfect reflection operator was used. Another criterion to measure the cost of quantum walk was proposed in their work. Cost was divided into three types: set-up cost, update cost and checking cost, denoted by $S$, $U$ and $C$ respectively. Under this new criterion, their cost is $S + \frac{1}{\sqrt{\delta}} \left( \frac{1}{\sqrt{\delta}} U + C \right)$, compared to Szegedy’s result [11] of bipartite walk $S + \frac{1}{\sqrt{\delta}} (U + C)$, where $\delta$ is the spectral gap of the transition matrix $P$, $\epsilon$ is a lower bound on the probability of choosing a marked element from the stationary distribution of $P$. Even though their method has a wider scope, it may still fail to arrive a quadratic speed-up in some graphs, e.g., the $\sqrt{N} \times \sqrt{N}$ 2D grid, whose classical hitting time is $\Theta(N \log N)$.

Either the method of Szegedy [9] or that of Ambainis et al. [12] finds the target with probability of $\Theta(\log N)$ in $O(\sqrt{N \log N})$ steps of quantum walk. $\sqrt{\log N}$ iterations of amplitude amplification [13] are required to enlarge the possibility. Therefore, the total complexity is $O(\sqrt{N \log N})$, which is by one logarithmic factor larger than expectation.

Tulsi [14] replaced amplitude amplification with a controlled quantum walk circuit and eliminated the $\sqrt{\log N}$ factor in AKR’s algorithm [12], and thus achieved a quadratic speed-up at the cost of a ancilla qubit. The structure of the controlled quantum walk gate is shown in Figure 2.

Figure 2. Circuit diagram for the controlled quantum walk search algorithm. The Reflect and Walk boxes denote the reflection operator and the walk operator [14].

Magniez et al. extended Tulsi’s result to any state-transitive Markov chain with unique marked state
[15]. Hoyer and Komeili came up with an algorithm in 2016, using divide and conquer for finding a marked element on the grid where there are multiple marked elements with a quadratic speed-up [16]. Recently, Ambainis et al. presented a new algorithm for finding a marked vertex in any graph and with any set of marked vertices with a quadratic speed-up [17]. Their algorithm was based on the interpolated walks by Krovi et al. [18].

In this section, we review how quantum walk is applied to search problem, from intuitive analogue to general procedure so as to well-designed model. Hitting time is defined and demonstrated to be faster than its classical counterpart quadratically. Although amplitude amplification is a powerful tool for improving the possibility of algorithms, it may also be a bottleneck for further improving the complexity. On the other hand, well-designed structures like controlled quantum walk gate and new technologies (e.g. quantum interpolation) could be depended on approaching the theoretical lower bound.

3. Sampling and Mixing time

For an ergodic Markov chain, a classical random walk will lead to its stationary distribution $\pi$. Starting from some initial state, the required number of steps to achieve a distribution that is sufficiently close to $\pi$ is referred as mixing time. In other words, mixing time reflects how fast a random walk can reach its stationary distribution. In the case of classical random walk, mixing time has a lower bound $\Omega(\delta^{-1})$ [19], where $\delta$ is the spectral gap.

One of the most common applications for random walk is sampling. Given a probability distribution $f$ (either discrete or continuous), a Markov chain can be generated, for example by Metropolis-Hastings algorithm [20], such that the stationary distribution $\pi$ of the Markov chain is equal to desired $f$. After a few heat-up steps, samples generated by the random walker will subject to the distribution $f$. In a word, the speed of mixing process determines the efficiency of sampling procedure. Thus, it is worthwhile to study whether quantum walk can reduce mixing time, and therefore improve sampling algorithms.

Usually, a sequence of slowly-varying Markov chains $P_0, P_1, ..., P_r = P$, with stationary distribution $\pi_0, \pi_1, ..., \pi_r$ is used. Here, slowing-varying means $\pi_0$ is close enough to $\pi_1$, $\pi_1$ is sufficiently close to $\pi_2$, etc. Aharonov and Ta-Shma [21] proved that if $\pi_0$ can be efficiently sampled, so can $\pi_r$.

Wocjan and Abeyesinghe proposed a new method to speed up this mixing process [22]. They used the slowly-varying Markov chains, and improved results by applying Szegedy’s quantum walk operators $W$ [9] and amplitude amplification [13].

They defined unitary operator $R_t = \omega \Pi_t + \Pi_t^\perp$, where $\Pi_t$ is the projection on the subspace spanned by state $|t\rangle$ and $\Pi_t^\perp$ is the projection on the orthogonal subspace, and $\omega = e^{\pi i / 3}$ is used for Grover’s amplitude amplification. Another operator $U$ is defined recursively as:

$$U_{l,0} = I$$
$$U_{l,m+1} = U_{l,m} \cdot R_{1} \cdot U_{l,m}^\dagger \cdot R_{l+1} \cdot U_{l,m}$$

Figure 3. Quantum circuit $\hat{R}$ for approximate $R$ [22].

Then they designed a quantum circuit $\hat{R}$, which applies $W$ with control for $2^{\alpha} - 1$ times ($\alpha$ is the
number of ancilla qubits), to approximate $R$, as illustrated in Figure 3. This algorithm bounds the total variation distance between the final state and the target within tolerance $\epsilon$. Although his result has a $\sqrt{\delta^{-1}}$ factor, quadratically smaller than classical $\delta$, it also has a factor related with the number of Markov chains.

Further improvement was made by Orsucci et al. with complexity of $O\left(\sqrt{\delta^{-1}/N}\right)$, which still carries a factor of system size $N$ [23]. An open problem remains unsolved is that whether the complexity of $O(\sqrt{\delta^{-1}})$ can be achieved, neglecting some polynomial factors.

4. Universal computation

As pointed out in [24], universality is a highly desired property for a computation model since it shows the capability of emulating other models. For the sake of showing the power of quantum walk as a general model of computation, several studies have been conducted.

Childs [25] pointed out that continuous time quantum walk is suitable for any problems that can be solved by a general-purpose quantum computer. Main idea of that work is to encode the quantum computation into a graph connected by quantum wires. Scattering on graphs is used and three widgets are designed (controlled-not gate, phase shift gate and basis-changing gate as illustrated in Figure 4, where open circles indicate vertices where previous or successive widgets can be attached: (a) Controlled-not gate, (b) Phase shift and (c) Basis-changing gate.) to satisfy the criteria, for building a universal quantum computer. The construction of the graph needs exponential larger space than the input, it is to say, for a circuit with $n$ qubits, the induced graph will have a $2^n$ wires. Lovett et al. [26] proved the universality of discrete time quantum walk. A different group of controlled-not gate, Hadamard gate and phase shift gate was chosen to create a universal set. Structure used to implement a controlled-not gate is shown in Figure 5. In this case the first qubit is the control and the second is the target. The target qubit’s wires are interchanged and the control qubit is left unaltered. The dotted lines represent wires passing underneath the solid lines - there is no interaction between these wires.

Instead of single edge in continuous case, two edges per wire were used to avoid possible reflection on vertices and to guarantee a one-directional propagation, as illustrated in Figure 6. Therefore, their gates require twice the number of edges than the continuous counterpart.
Combining the advantages of both schemes, Underwood and Feder [27] presented a hybrid solution named discontinuous quantum walk, taking discrete steps of continuous evolution. According to [24], discontinuous quantum walk propagates on a line composed of two kinds of alternating segments. The walker starts on the first segment and walks for a long time to ensure the perfect state transformation (PST) to the end of the segment. Subsequently, the first segment is turned off while the second segment is switched on, i.e., the walker repeats the PST to the end. Although moving in discrete timesteps, coin register is eliminated in their scheme; instead, an additional amount of global control is used.

Though quantum walks discussed above differs from each other, a common strategy can be abstracted: quantum computation is firstly encoded into a graph connected by quantum wires, then several widgets which form a complete set to build any gate are designed and applied to the graph. Once the mapping from quantum computation to the graph is proved to be equivalent, the universality can be obtained.

5. Application
The first algorithm using quantum walk that went beyond search problem was made by Ambainis when he was handling the problem of element distinctness [28]. Inspired by his work, many specific algorithms, e.g., triangle finding [29], verification of matrix products [30], group commutativity testing [31] are proposed.

5.1. Element distinctness
The problem is defined as: given a set of numbers \( x_1, ..., x_N \in \mathbb{M} \), are there any \( i, j \in [N], i \neq j \) such that \( x_i = x_j \)? The generalization of this problem is called element k-distinctness: given a set of numbers, are there k distinct indices (which is also called k-collision), such that the numbers corresponding are all the same. Subset with size \( r \) or \( r + 1 \) are selected from these \( N \) number and mapped to vertices in graph \( G \). If two subsets differ in one element, corresponding vertices are connected by an edge. If the subset contains such \( x_i = x_j \), corresponding vertex is defined as marked. Therefore, the problem of element distinctness is reduced to marked vertex finding in the graph \( G \).

Central idea of speeding up is based on re-using the information in last query rather than querying all the elements in every vertex in naive version of Grover Search. In this way, only the elements in first vertex are queried for \( O(r) \) times. The quantum walk happens on two Hilbert space \( \mathcal{H} \) and \( \mathcal{H}' \) whose basis states are related with the subset of indices and power set of values. Each walk step performs a mapping from \( \mathcal{H} \) to \( \mathcal{H}' \), and then in opposite direction. The transformation between two Hilbert space is equivalent to moving from one vertex on the graph to another.

Similar to [6], walker has different behavior on marked nodes, a conditional phase flip operator is applied if the node contains a k-collision. For the case when there is at most one k-collision, \( t_2 = O\left(\sqrt{r} \right) \) steps of the quantum walk are needed in every iteration, while \( t_4 = O\left((N/r)^{k/2}\right) \) iterations are required. Thus, the element k-distinctness can be solved with \( O\left( \max\left(\frac{N^{k/2}}{r^{(k-1)/2}}, r \right) \right) \) queries. Through setting \( r = \left[ N^{2/3} \right] \) and \( k = 2 \), the exact results are obtained addressing the element distinctness with \( O\left(N^{2/3}\right) \) queries. The case when there could be multiple solutions can be solved by applying the algorithm several times.

His work shows a general procedure of solving a specific problem with quantum walk: map the original problem to search problem on a graph, set the target to be marked nodes, then design oracles that will drive the walker from one step to another. Different behavior for walker can be achieved through some kind of controlled (or conditional) quantum gate. Besides, Ambainis illustrated how to re-use the information of last query to avoid unnecessary queries.

5.2. Triangle Finding
Given an undirected graph with \( n \) vertices, there are \( \binom{n}{3} \) different choice of selecting 3 vertices, whether a triple form a triangle can be decided with 3 queries. The question can be answered with
\( O\left(\binom{n}{\frac{3}{2}}\right) = O(n^3) \). By applying Grover search, a better result with \( O\left(n^{3/2}\right) \) can be achieved. Buhrman et al. [32] designed an algorithm for triangle finding with amplitude amplification and proved a complexity of \( O(n + \sqrt{nm}) \), where \( m \) is the number of edges. Their algorithm has a good performance on sparse graphs, while still requires \( \Theta(n^{3/2}) \) queries on dense graphs. Subsequently, Magniez et al. [29] presented a new algorithm using the result of collision problem and its generalization named graph collision problem, which achieves the complexity of \( O(n^{3/10}) \). Later efforts are summarized in Table 2. Except for the recent improvements, new notions about quantum walk is more important. Some of them, such as span program, can even be applied to general quantum methods.

**Table 2. List of different algorithms and their complexity**

| Algorithm            | Complexity                          |
|----------------------|-------------------------------------|
| Brute Force          | \( O(n^3) \)                        |
| Grover [7]           | \( O(n^{3/2}) = O(n^{1.5}) \)      |
| Buhrman et al. [32]  | \( O(n + \sqrt{nm}) \)             |
| Magniez et al. [29]  | \( O(n^{13/10}) = O(n^{1.3}) \)    |
| Belovs [33]          | \( O(n^{35/27}) \approx O(n^{1.296}) \) |
| Lee et al. [34]      | \( O(n^{9/7}) \approx O(n^{1.286}) \) |
| Jeffery et al. [35]  | \( O(n^{9/7}) \approx O(n^{1.286}) \) |
| Gall [36]            | \( O(n^{5/4}) = O(n^{1.25}) \)     |

**6. Conclusion**

In this article, discoveries related with quantum walk in last two decades have been reviewed and discussed. The major interests in this field are demonstrated to be focusing mainly on three topics: reducing hitting time, diminishing mixing time and the possibility of universal computation. According to several concrete examples, one can see the strength of quantum walk and in which way the algorithm is improved step by step.

Various models and new algorithms have been proposed based on the original intuitive analogue to classical random walk, aiming at offering a simple but general solution to the problem. Moreover, though great breakthrough has taken place recently, the theoretical lower bound has not been reached, which implies the possibility of improvement. The appearances of better models are expected to be witnessed in the near future.

**References**

[1] P. W. SHOR, Algorithms for quantum computation: discrete logarithms and factoring, Proceedings 35th annual symposium on foundations of computer science, Ieee, 1994, pp. 124-134.
[2] L. PAGE, S. BRIN, R. MOTWANI and T. WINOGRAD, The PageRank citation ranking: Bringing order to the web, Stanford InfoLab, 1999.
[3] A. M. CHILDS, R. CLEVE, E. DEOTTO, E. FARHI, S. GUTMANN and D. A. SPIELMAN, Exponential algorithmic speedup by a quantum walk, Proceedings of the thirty-fifth annual ACM symposium on Theory of computing, 2003, pp. 59-68.
[4] A. AMBAINIS, Quantum walks and their algorithmic applications, International Journal of Quantum Information, 1 (2003), pp. 507-518.
[5] J. KEMPE, Quantum random walks: an introductory overview, Contemporary Physics, 44 (2003), pp. 307-327.
[6] N. SHENVI, J. KEMPE and K. B. WHALEY, Quantum random-walk search algorithm, Physical Review A, 67 (2003), pp. 052307.
[7] L. K. GROVER, A fast quantum mechanical algorithm for database search, Proceedings of the
twenty-eighth annual ACM symposium on Theory of computing, 1996, pp. 212-219.

[8] A. AMBAINIS, Quantum query algorithms and lower bounds, Classical and New Paradigms of Computation and their Complexity Hierarchies, Springer, 2004, pp. 15-32.

[9] M. SZEGEDDY, Quantum speed-up of Markov chain based algorithms, 45th Annual IEEE symposium on foundations of computer science, IEEE, 2004, pp. 32-41.

[10] F. MAGNIEZ, A. NAYAK, J. ROLAND and M. SANTHA, Search via quantum walk, SIAM journal on computing, 40 (2011), pp. 142-164.

[11] M. SZEGEDDY, Spectra of quantized walks and a√δε rule, arXiv preprint quant-ph/0401053 (2004).

[12] A. AMBAINIS, J. KEMPE and A. RIVOSH, Coins make quantum walks faster, arXiv preprint quant-ph/0402107 (2004).

[13] G. BRASSARD, P. HOYER, M. MOSCA and A. TAPP, Quantum amplitude amplification and estimation, Contemporary Mathematics, 305 (2002), pp. 53-74.

[14] A. TULSI, Faster quantum-walk algorithm for the two-dimensional spatial search, Physical Review A, 78 (2008), pp. 012310.

[15] F. MAGNIEZ, A. NAYAK, P. C. RICHTER and M. SANTHA, On the hitting times of quantum versus random walks, Algorithmica, 63 (2012), pp. 91-116.

[16] P. HOYER and M. KOMEILLI, Efficient quantum walk on the grid with multiple marked elements, arXiv preprint arXiv:1608.08958 (2016).

[17] A. AMBAINIS, A. GILYén, S. JEFFERY and M. KOKAINIS, Quadratic speedup for finding marked vertices by quantum walks, Proceedings of the 52nd Annual ACM SIGACT Symposium on Theory of Computing, 2020, pp. 412-424.

[18] H. KROVI, F. MAGNIEZ, M. OZOLS and J. ROLAND, Quantum walks can find a marked element on any graph, Algorithmica, 74 (2016), pp. 851-907.

[19] D. ALDOUS, L. LOVÁSZ and P. WINKLER, Mixing times for uniformly ergodic Markov chains, Stochastic Processes and their Applications, 71 (1997), pp. 165-185.

[20] W. K. HASTINGS, Monte Carlo sampling methods using Markov chains and their applications, (1970).

[21] D. AHARONOV and A. TA-SHMA, Adiabatic quantum state generation and statistical zero knowledge, Proceedings of the thirty-fifth annual ACM symposium on Theory of computing, 2003, pp. 20-29.

[22] P. WOCJAN and A. ABYEYESINGHE, Speedup via quantum sampling, Physical Review A, 78 (2008), pp. 042336.

[23] D. ORSUCCI, H. J. BRIEGEL and V. DUNJKO, Faster quantum mixing for slowly evolving sequences of Markov chains, Quantum, 2 (2018), pp. 105.

[24] S. E. VENEGAS-ANDRACA, Quantum walks: a comprehensive review, Quantum Information Processing, 11 (2012), pp. 1015-1106.

[25] A. M. CHILDS, Universal computation by quantum walk, Physical review letters, 102 (2009), pp. 180501.

[26] N. B. LOVETT, S. COOPER, M. EVERITTT, M. TREVERS and V. KENDON, Universal quantum computation using the discrete-time quantum walk, Physical Review A, 81 (2010), pp. 042330.

[27] M. S. UNDERWOOD and D. L. FEDER, Universal quantum computation by discontinuous quantum walk, Physical Review A, 82 (2010), pp. 042304.

[28] A. AMBAINIS, Quantum walk algorithm for element distinctness, SIAM Journal on Computing, 37 (2007), pp. 210-239.

[29] F. MAGNIEZ, M. SANTHA and M. SZEGEDDY, Quantum algorithms for the triangle problem, SIAM Journal on Computing, 37 (2007), pp. 413-424.

[30] H. BUHRMAN and R. SPALEK, Quantum verification of matrix products, arXiv preprint quant-ph/0409035 (2004).

[31] F. MAGNIEZ and A. J. A. NAYAK, Quantum complexity of testing group commutativity, 48 (2007), pp. 221-232.

[32] H. BUHRMAN, C. DURR, M. HEILIGMAN, P. HOYER, F. MAGNIEZ, M. SANTHA and R. DE
WOLF, Quantum algorithms for element distinctness, Proceedings 16th Annual IEEE Conference on Computational Complexity, IEEE, 2001, pp. 131-137.

[33] A. BELOVS, Span programs for functions with constant-sized 1-certificates, Proceedings of the forty-fourth annual ACM symposium on Theory of computing, 2012, pp. 77-84.

[34] T. LEE, F. MAGNIEZ and M. SANTHA, Improved quantum query algorithms for triangle finding and associativity testing, Proceedings of the twenty-fourth annual ACM-SIAM symposium on Discrete algorithms, SIAM, 2013, pp. 1486-1502.

[35] S. JEFFERY, R. KOTHARI and F. MAGNIEZ, Nested quantum walks with quantum data structures, Proceedings of the twenty-fourth annual ACM-SIAM symposium on Discrete algorithms, SIAM, 2013, pp. 1474-1485.

[36] F. LE GALL, Improved quantum algorithm for triangle finding via combinatorial arguments, 2014 IEEE 55th Annual Symposium on Foundations of Computer Science, IEEE, 2014, pp. 216-225.