An Introduction to Quantum Walks and application to Search Problems

Riddhiman Bhattacharya

July 14, 2023

Abstract

In the field of classical algorithms, classical random walks have provided useful approximation and optimization algorithms, For instance. **Schoning's** [6] [1] algorithm, achieved a complexity of $(4/3)^n$ for 3-SAT. Motivated by the results for classical walks, researchers in the early 2000s started developing the theory for its quantum analogue. One of the earliest successes was **Ambainis' quantum walk** for element distinctness problem [1]. In this paper, I'll provide an overall introduction of quantum walks [4] and its application to two search problems

1. The Historic **Quantum random-walk search algorithm** [7]
2. A recent generalization for **Spatial Search** [2]

.

1 Background

I'll first describe the quantum analogue of the **discrete time Markov chain**(**DTMC**). For each iteration of a classical random walk on a graph, a coin is flipped. The walker then moves to an adjacent node specified by the outcome of the coin flip.
An equivalent process occurs in the quantum walk, with the modification that the coin is a quantum coin and can therefore exist in a superposition of states. However, if the state of the coin is measured after each flip, then the quantum random walk reverts to a classical random walk [2].

Mathematically, this can be thought of as a repeated application of a unitary evolution operator U. U acts on the Hilbert space $\mathcal{H}^C \otimes \mathcal{H}^S$ where $\mathcal{H}^C$ is the Hilbert space associated with a quantum coin and $\mathcal{H}^S$ is the Hilbert space associated with the nodes of the graph. The operator U can be written as

$$U = S \cdot C \tag{1}$$

where S is a permutation matrix that performs a controlled shift based on the state of the quantum coin, and C is a unitary matrix that flips the quantum coin. An important point here is that if no measurement is made the quantum walk is controlled by a **unitary operator** rather than a stochastic one. This implies that there is **no limiting stationary distribution**.

[1] a random walk on a complete graph with vertices $\in \{0,1\}^n$ and $(x, y) \in E$ if the Hamming distance is 1
[2] even measuring the state of nodes destroys the superposition, reverting to classical

For a d-regular graph, the Shift operator S is as follows

$$S\left|j\right\rangle \otimes \left|v\right\rangle = \begin{cases} \left|j\right\rangle \otimes \left|w\right\rangle & \text{if } e_v^j = (v,w) \\ 0 & \text{otherwise} \end{cases} \tag{2}$$

The coin space $\mathcal{H}^C$ for the above is of dimension d, i.e $j \in \{1 \dots d\}$. The Coin flip $C = C_0 \otimes I$, C_0 is a $d \times d$ unitary acting on $\mathcal{H}^C$

Important Note: The Shift operator S is straightforward and constructed from the underlying graph structure, but there's a lot of freedom for the choice of C. Different C's generate different walks, and if nothing is known about where a marked vertex exists the authors suggest using the following family of unitary and permutation symmetric coins:

$$G_{a,b} = \begin{pmatrix} a & b & b & \dots & b & b \\ b & a & b & \dots & b & b \\ b & b & a & \dots & b & b \\ & & & \dots & & \\ b & b & b & \dots & a & b \\ b & b & b & \dots & b & a \end{pmatrix} \tag{3}$$

where a and b are real, $1 - \frac{2}{d} \leq |a| \leq 1$ and $b = \pm(1-a)$. Among all these coins $a = \frac{2}{d} - 1$ and $b = \frac{2}{d}$ is called the **Grover coin**. Among all coins $G_{a,b}$ this is the coin which is the farthest away from the identity operator ($G_{1,0} = I$).

2 Quantum walk search algorithm

This coined walk was used by **Kempe**, where the underlying graph was an n-dimensional hypercube, and the marked node $\vec{x}_{target}$. The search space of the algorithm is all n-bit binary strings, $x \in \{0,1\}^n$. The function $f(\vec{x}) \in \{0,1\}$ is such that $f(x) = 1$ for exactly one input $\vec{x}_{target}$. The goal is to find $\vec{x}_{target}$. Using the mapping of n-bit binary string to nodes on the hypercube, this search problem is then equivalent to searching for a single marked node amongst the $N = 2^n$ nodes on the n-cube. For purposes of the proof, we set the marked node to be $\vec{x}_{target} = \vec{0}$, but the location of the marked node has no significance.

The Shift operator S maps a state $\left|d, \vec{x}\right\rangle$ onto the state $\left|d, \vec{x} \oplus \vec{e_d}\right\rangle$, where $\vec{e_d}$ is the d^{th} basis vector on the hypercube. Explicitly

$$S = \sum_{d=0}^{n-1} \sum_{\vec{x}} \left|d, \vec{x} \oplus \vec{e_d}\right\rangle \left\langle d, \vec{x}\right| \tag{4}$$

$C_0 = G$, the Grover coin we saw earlier can be equivalently represented as

$$C_0 = G = -I + 2\left|s^C\right\rangle \left\langle s^C\right| \tag{5}$$

where $\left|s^C\right\rangle = \frac{1}{\sqrt{n}} \sum_{d=1}^{n} \left|d\right\rangle$. C_0 preserves the permutation symmetry of the hypercube as it is invariant to all permutations of n dimensions. However the C_{pert} that is applied in the algorithm is "perturbed", i.e C_1 [3] is applied to the marked node and C_0 to the unmarked nodes by the coin oracle [4]

$$C_{pert} = C_0 \otimes I + (C_1 - C_0) \otimes \left|\vec{0}\right\rangle \left\langle \vec{0}\right| . \tag{6}$$

[3] $C_1 = -I$ is considered in the analysis

[4] The cost for this algorithm the number of calls to this oracle

Finally, the Unitary operator U_{pert} to be applied t times is:

$$\begin{aligned} U_{pert} &= S \cdot C_{pert} \\ &= U - 2S \cdot \Big(|s^C\rangle \langle s^C| \otimes |\vec{0}\rangle \langle \vec{0}| \Big) \end{aligned} \tag{7}$$

Algorithm 1 Search with coin oracle

```
1: procedure Find Marked
2:     $\psi_0 \leftarrow |s^C\rangle \otimes |s^S\rangle$ i.e equal superposition over all states
3:     Compute $U^t_{pert}|\psi_0\rangle$ where $t = \frac{\pi}{2}\sqrt{2^n}$
4:     Measure in $|d, \vec{x}\rangle$ basis
```

With probability $\frac{1}{2} - O(1/n)$, the measured outcome is the marked state. As with any probabilistic algorithm this algorithm can be run a number of times, so that the marked state can be determined with arbitrarily small degree of error.

Important Note: The key idea used in this algorithm is that U^t_{pert}, applied on $|\psi_0\rangle$ is the same as a 2D rotation, from $\frac{1}{\sqrt{2}}(|w_0\rangle + |-w_0\rangle)$[5] to $\frac{1}{\sqrt{2}}(-|w_0\rangle + |-w_0\rangle)$[6]. The angle between these two is $\pi/2$ and each application of U_{pert} corresponds to a rotation angle of $1/\sqrt{2^{n-1}}$, so $t = \frac{\pi}{2}\sqrt{2^{n-1}} = O(\sqrt{N})$ coin oracle calls. Note: Here w_0 and $-w_0$ are "relevant" eigenvectors for the eigendecomposition of U_{pert}.

3 Linking the problems, main result

I'll now discuss a related but more general problem of spatial search. Here we are interested in finding a marked vertex in a graph. **Szedegy** [8] showed that **if a classical random walk hits a marked element in Hitting time HT steps, then there is a detection algorithm that runs in time $O(\sqrt{HT})$, i.e it can detect whether a marked vertex exists, but not actually find it** [7]. Krovi's[5] algorithm finds a marked element in $O(\sqrt{HT^+})$, HT^+ is the extended hitting time. $HT^+ = HT$ when there is exactly one marked element, but in [2] the authors show that there are special graphs with multiple marked vertices, where $HT^+ >> HT$ and even $\sqrt{HT^+} >> HT$ [8].They also provide an algorithm that runs the find problem (for multiple marked vertices) in $\tilde{O}(\sqrt{HT})$.
The running time for the **Quantum walk search algorithm** in the previous section is actually a special case for the spatial search problems, we can obtain it by running [5] or [2] quantum walk on the complete graph with only one marked element and obtain $O(\sqrt{HT})$ runtime.

4 Quadratic speedup for finding marked vertices by quantum walks

The two main components for the $\tilde{O}(\sqrt{HT})$ algorithm are the Interpolated walk, and the Quantum fast forwarding algorithm. There is also a simple heuristic which does not use Quantum fast forwarding, it performs well empirically and is conjectured to run in $O(\sqrt{HT})$

[5]an approximation of $|\psi_0\rangle$
[6]an approximation of $|\vec{x}_{target}\rangle$
[7]While the detection and finding problem in classical walks are the same this isn't the case for quantum walks
[8]Implying that even the classical walk is faster

4.1 Brief note on DTMC

$\mathcal{P}$ is *ergodic* if for a large enough $t \in \mathbb{N}$ all elements of $\mathcal{P}^t$ are non-zero. For an ergodic $\mathcal{P}$ we have a unique stationary distribution π such that $\pi\mathcal{P} = \pi$, the corresponding *time-reversed* Markov chain is defined as $\mathcal{P}^* := \operatorname{diag}(\pi)^{-1} \cdot \mathcal{P}^T \cdot \operatorname{diag}(\pi)$. $\mathcal{P}$ is *reversible* if it is ergodic and $\mathcal{P}^* = \mathcal{P}$.

For an ergodic Markov chain $\mathcal{P}$, discriminant matrix D is defined such that its xy-entry is $\sqrt{\mathcal{P}_{xy}\mathcal{P}^*_{yx}}$. Following from this definition we get

$$D = \operatorname{diag}(\pi)^{\frac{1}{2}} \cdot \mathcal{P} \cdot \operatorname{diag}(\pi)^{-\frac{1}{2}}. \tag{8}$$

4.2 Interpolated walk

Works as follows for a reversible Markov chain $\mathcal{P}$: first checks whether the current node is marked. If the node is *unmarked*, then it performs a *normal step of the walk*; but if it is *marked*, then it performs a *normal walk step with probability* $1-s$, *with probability* s *it stays at the current marked node*. For $\mathcal{P}$ the marked set $M \subset X$. $\mathcal{P}'$ is the absorbing markov chain. The states of X are arranged such that $U := X \setminus M$ come first, matrices $\mathcal{P}$ and $\mathcal{P}'$ have the following block structure:

$$\mathcal{P} := \begin{pmatrix} \mathcal{P}_{UU} & \mathcal{P}_{UM} \\ \mathcal{P}_{MU} & \mathcal{P}_{MM} \end{pmatrix}, \qquad \mathcal{P}' := \begin{pmatrix} \mathcal{P}_{UU} & \mathcal{P}_{UM} \\ 0 & I \end{pmatrix}.$$

The *interpolated walk* operator, for $s \in [0,1)$, is :

$$\mathcal{P}(s) := (1-s)\mathcal{P} + s\mathcal{P}', \tag{9}$$

The corresponding discriminant matrix is $D(s)$. Π_M denotes the projector onto marked vertices and $\Pi_U := I - \Pi_M$ denotes the projector onto unmarked vertices.

Quantum walk search - General Structure

- $\texttt{Check}(M)$: Check if a given vertex is marked by mapping $|x\rangle|b\rangle$ to $|x\rangle|b\rangle$ if $x \notin M$ and $|x\rangle|b \oplus 1\rangle$ if $x \in M$, $|x\rangle$ is the vertex register and $b \in \{0,1\}$;
- $\texttt{Setup}(\mathcal{P})$: Construct the superposition $|\sqrt{\pi}\rangle = \sum_{x\in X}\sqrt{\pi_x}\,|x\rangle$;
- $\texttt{Update}(\mathcal{P})$: Apply $\textsc{Shift}$, $\textsc{Ref}$, and $V(\mathcal{P})^{\pm 1}$.

Important Note: Checking is performed by the oracle. Each of these operations has a corresponding associated implementation cost, denoted by C, S, and U. The interpolated walk + quantum fast forwarding finds a marked vertex in complexity $\widetilde{\mathcal{O}}\left(\mathsf{S} + \sqrt{HT}(\mathsf{U} + \mathsf{C})\right)$.

For the interpolated quantum walk, the quantum update operator $V(\mathcal{P}, s)$ for all $x \in U$ acts as $I \otimes V(\mathcal{P})$ on the initial state $|0\rangle|\bar{0}\rangle|x\rangle$ [9], and for $x \in M$ acts as $|0\rangle|\bar{0}\rangle|x\rangle \mapsto \sqrt{1-s}\,|0\rangle V(\mathcal{P})|\bar{0}\rangle|x\rangle + \sqrt{s}\,|1\rangle|\bar{0}\rangle|x\rangle$. Explicitly, the corresponding unitary that is applied is:

$$W(s) := V^\dagger(\mathcal{P}, s)\,\textsc{Shift}'\,V(\mathcal{P}, s)\,\textsc{Ref}', \tag{10}$$

where $\textsc{Shift}' := |0\rangle\langle 0| \otimes \textsc{Shift} + |1\rangle\langle 1| \otimes I$ and $\textsc{Ref}' := (2|0\rangle\langle 0| \otimes |\bar{0}\rangle\langle\bar{0}| - I) \otimes I$ and $\textsc{Shift}$ is defined by the action $|x, y\rangle \mapsto |y, x\rangle$, for all $x, y \in X$ It is easy to see that

$$\langle 0|\langle\bar{0}|\langle x|\,W(s)\,|0\rangle|\bar{0}\rangle|y\rangle = D_{xy}(s). \tag{11}$$

[9] the first $|0\rangle$ refers to the coin that flips wp s and $1-s$, it is generally omitted and $|0\rangle|\bar{0}\rangle$ is just denoted by $|\bar{0}\rangle$

Note that $W(s)$ can be implemented[10] for any $s \in [0,1)$ in cost of order $\mathsf{C}+\mathsf{U}$, the following way. First check whether $x \in X$ is marked, and if it is, then apply the map $|0\rangle \mapsto \sqrt{1-s}\,|0\rangle + \sqrt{s}\,|1\rangle$ to the first qubit. Controlled by the first qubit's state being $|0\rangle$ apply $V(\mathcal{P})$ to the last two registers.

While a classical random walk can find a marked vertex in complexity[11] $\mathcal{O}\left(\mathsf{S} + HT(\mathsf{U}+\mathsf{C})\right)$, showed that using the the quantum walk $W(s)$ one can find a marked vertex in complexity $\mathcal{O}\big(\mathsf{S} + \sqrt{HT^{+}}(\mathsf{U}+\mathsf{C})\big)$.

4.3 Algorithm with known s and t

The following algorithm describes a quantum walk algorithm with a fixed interpolation parameter $s \in [0,1)$ and a predetermined number of uantum walk steps $t \in \mathbb{N}$(it is a good heuristic method). The basis states $|x\rangle$ identified with the vertices of the graph, are in the n-dimensional Hilbert space $\mathcal{H}$ [12]. The algorithm uses two registers R_1, R_2 with underlying state space $\mathcal{H}$ for each of them, initialized to some reference state $|\bar{0}\rangle$. An ancillary register R_3 initialized to $|0\rangle \in \mathbb{C}^2$ will be attached to check if the current vertex is marked. (Hope to obtain 1 from R_3 on final measurement, this serves as verification that the state in R_2 is a marked state).

Algorithm 2 Known s,t

```
Search($\mathcal{P}$, $M$, $s$, $t$)
  1. Prepare the state $|\bar{0}\rangle|\sqrt{\pi}\rangle$ with Setup($\mathcal{P}$).
  2. Apply $t$ times the operator $W(s)$ on $\mathsf{R}_1\mathsf{R}_2$.
  3. Attach $\mathsf{R}_3$, apply Check($M$) on $\mathsf{R}_2\mathsf{R}_3$, measure $\mathsf{R}_3$.
  4. If $\mathsf{R}_3 = 1$, measure $\mathsf{R}_2$ in the vertex basis, output the outcome. Otherwise, output No marked vertex found.
```

The above algorithm relies on the following conjecture:

Conjecture 1 *Let $\mathcal{P}$ be a reversible, ergodic Markov chain with stationary distribution π; suppose that M is a set of marked states which satisfies $p_M = \sum_{x\in M} \pi_x < 0.5$. Then there exists a value $s \in [0,1)$ and a positive integer $t = \mathcal{O}\big(\sqrt{HT}\big)$ such that Algorithm 2 succeeds with probability $\Omega(1)$. $q_t(s) = \Omega(1)$, where $q_t(s)$ is defined by* (**??**).

If true, the runtime of the algorithm is of the order $\mathcal{O}\left(\mathsf{S} + \sqrt{HT}\cdot(\mathsf{U}+\mathsf{C})\right)$. In the paper, the authors provide some directions towards proving this, but they conclude this algorithm with the empirical results.

4.4 Quantum fast-forwarding

Quantum fast-forwarding technique(**QFF**) [3], will be used on the interpolated walk. In an adapted form, it is as follows:

[10] We note that [**?**, Appendix B.2] also describes a way to implement the interpolated quantum walk operator with similar complexity but additionally require (query) access to the diagonal entries of $\mathcal{P}$.

[11] We note that in the classical case, S can be replaced with the cost of *classically* sampling from π, and U with the cost of classically sampling a neighbour of the current vertex. These classical sampling operations may be cheaper than `Setup` and `Update`, but in practice, they are often the same.

[12] We shouldn't get confused with Kempes notation for quantum walk search algorithm, now any graph structure is allowed

Theorem 2 *Let $\varepsilon \in (0,1)$, $s \in [0,1]$ and $t \in \mathbb{N}$. Let $\mathcal{P}$ be any reversible Markov chain on state space X, and let Q be the cost of implementing the (controlled) quantum walk operator $W(s)$. There is a quantum algorithm with complexity $\mathcal{O}\left(\sqrt{t \log(1/\varepsilon)}\mathsf{Q}\right)$ that takes input $|\bar{0}\rangle |\psi\rangle \in \operatorname{span}\{|\bar{0}\rangle |x\rangle : x \in X\}$, and outputs a state that is ε-close to a state of the form*

$$|0\rangle^{\otimes a} |\bar{0}\rangle D^t |\psi\rangle + |\Gamma\rangle$$

where $a = \mathcal{O}(\log(t \log(1/\varepsilon)))$ and $|\Gamma\rangle$ is some garbage state that has no support on states containing $|0\rangle^{\otimes a} |\bar{0}\rangle$ in the first two registers and $\| |\Gamma\rangle \|^2 = 1 - \| D^t |\psi\rangle \|^2$ (which may depend on $|\psi\rangle$ and t).

Important Note: QFF allows us to, in some very "quantum" sense, apply t steps of a classical walk in only $\sqrt{t}$ calls to its update operation, but the **caveat** : there is no guarantee on the amplitudes for each state by the quantum fast forwarding technique, we only have a guarantee on it being non zero. In particular, applied naively we could have a very high probability of reading the garbage state. The main contribution by [2] is providing a non trivial lower bound on measuring a marked state. This is proved by a series of lemmas, some combinatorial and some from prior results on ergodic chains.
But the main theorem to focus on is the following:

Theorem 3 *Let $\mathcal{P}$ be a reversible ergodic Markov chain, and let π be its stationary distribution. If $p_M \leq 1/9$ and $T \geq 3HT$, then choosing $s \in S = \{1 - \frac{1}{r} : r \in R\}$ and $t \in [24T]$ uniformly at random we get, that*

$$\mathbb{E}\left[\left\| \Pi_M D^t(s) |\sqrt{\pi_U}\rangle \right\|^2 \right] = \Omega\left(\frac{1}{\log(T)}\right).$$

What the above theorem means in words: the expected value[13] of obtaining a measurement $\in$ the marked states when starting off in $\pi_U = |\sqrt{\pi_U}\rangle = \sum_{x \in U} \sqrt{\pi_x} |x\rangle$ is lower bounded by $1/log(T)$.

This directly leads to the following algorithm:

Algorithm 3 Fast-forwarding-based search algorithm

Search($\mathcal{P}$, M, T)
Using $\mathcal{O}\left(\sqrt{\log(T)}\right)$ rounds amplitude amplification to amplify the success probability of steps 1-3:

1. Using `Setup`($\mathcal{P}$) to prepare the state
$$\sum_{t=1}^{T} \frac{1}{\sqrt{T}} |t\rangle \sum_{s \in S} \frac{1}{\sqrt{|S|}} |s\rangle |\sqrt{\pi}\rangle.$$
2. Measuring $\{\Pi_M, I - \Pi_M\}$ on the last register. If the outcome is "marked", measure in the computational basis, and output the entry in the last register. Otherwise, continue with the (subnormalized) post-measurement state
$$\sum_{t=1}^{T} \frac{1}{\sqrt{T}} |t\rangle \sum_{s \in S} \frac{1}{\sqrt{|S|}} |s\rangle |\sqrt{\pi_U}\rangle.$$
3. Using quantum fast-forwarding, controlled on the first two registers, to map $|t\rangle |s\rangle |\sqrt{\pi_U}\rangle$ to $|1\rangle |t\rangle |s\rangle D^t(s) |\pi_U\rangle + |0\rangle |\Gamma\rangle$ for some arbitrary $|\Gamma\rangle$, with precision $\mathcal{O}\left(\frac{1}{\log(T)}\right)$. Finally, measure the last register and output its content if marked, otherwise output `No marked vertex`.

[13]actually the square of the probability of measuring a marked state

If $T \geq 72HT(\mathcal{P}, M)$, then the success probability of the above steps 1-3 is $\Omega\left(\frac{1}{\log(T)}\right)$, as shown by Corollary 4.4. Thus, after $\mathcal{O}\left(\sqrt{\log(T)}\right)$ steps of amplitude amplification, the success probability becomes $\Omega(1)$.

By Theorem 2 the complexity of step 3 is $\mathcal{O}\left(\sqrt{T \log\log(T)}(\mathsf{U}+\mathsf{C})\right)$, since $W(s)$ can be implemented in cost $\mathcal{O}(\mathsf{U}+\mathsf{C})$.

The complexity of steps 1-3 [14] is $\mathcal{O}\left(\mathsf{S}+\sqrt{T \log\log(T)}(\mathsf{U}+\mathsf{C})\right)$ Amplitude amplification gives a $\sqrt{\log(T)}$ multiplicative overhead, giving the final complexity

$$\mathcal{O}\left(\mathsf{S}\sqrt{\log(T)}+\sqrt{T}(\mathsf{U}+\mathsf{C})\sqrt{\log(T)\log\log(T)}\right)$$

References

[1] Andris Ambainis. Quantum walk algorithm for element distinctness. *45th Annual IEEE Symposium on Foundations of Computer Science*, pages 22–31, 2003.

[2] Andris Ambainis, András Gilyén, Stacey Jeffery, and Martins Kokainis. Quadratic speedup for finding marked vertices by quantum walks. *ArXiv*, abs/1903.07493, 2019.

[3] Simon Apers and Alain Sarlette. Quantum fast-forwarding: Markov chains and graph property testing. *Quantum Information Computation*, 19:181–213, 2018.

[4] J. Kempe. Quantum random walks: An introductory overview. 2003.

[5] Hari Krovi, Frédéric Magniez, Maris Ozols, and Jérémie Roland. Quantum walks can find a marked element on any graph. *Algorithmica*, 74:851–907, 2015.

[6] Torsten Schoning. A probabilistic algorithm for k-sat and constraint satisfaction problems. *40th Annual Symposium on Foundations of Computer Science (Cat. No.99CB37039)*, pages 410–414, 1999.

[7] Neil Shenvi, J. Kempe, and K. Birgitta Whaley. Quantum random-walk search algorithm. 2003.

[8] M. Szegedy. Quantum speed-up of markov chain based algorithms. *45th Annual IEEE Symposium on Foundations of Computer Science*, pages 32–41, 2004.

[14] By Theorem 7 in [2]