UTS41076 - Assignment 3
Quantum Walk Search

Arjun David Rao

Oct 2021

Abstract

Search algorithms appear in many different flavours across the computer science community and are frequently used to find marked elements in structured and unstructured graphs. One of these is the classical walk search, where a random walk is used to traverse the edges of a Markov chain and find a marked vertex. Quantum walk searches are a quantum mechanical version of a classical walk search implemented on a quantum state space and offer a speed up comparatively. Although these protocols are yet to be demonstrated experimentally, they have strong mathematical rigour as they exponentially decrease hitting times and also have a quadratic speed up in query complexity.

1 Classical random walks

Quantum walks are derived from the principle behind a classical walk; the latter being a common mathematical model used to simulate stochastic motion. It can be described by using an example of a coin toss, where by flipping the coin and it landing on a face determines the movement of a particle on a 1-D axis. For a discrete classical walk, the path can be quantified by the position, X,

$$X = \sum_{i=1}^{n} x_i \tag{1}$$

$$x_i = \begin{cases} -1, & p = \frac{1}{2} \\ +1, & p = \frac{1}{2} \end{cases} \tag{2}$$

where p is the probability, n is the time step and x_i is the length of each step.

A general random walk also has the following properties, with the mean $\bar{X}$ and variance $\mathrm{Var}\,(X)$,

$$\bar{X} = nL(2p-1) \tag{3}$$

$$\mathrm{Var}\,(X) = 4nL^2p(1-p) \tag{4}$$

where L is the length of each step and N is the total number of steps. For the simple example above, $\bar{X} = 0$ and $\mathrm{Var}(X) = n$. For algorithms which rely on traversing a large state-space, a larger variance will lead to a higher probability of finding any given item in the state space sooner, and hence shorter halting times.

Whilst the random walk poses an interesting mathematical concept, it has a broad range of applications to many different areas of modern science and technology. Models using random walks range from Brownian motion, stock market fluctuations and even neurons firing in the brain. One of these applications is performing a search for a marked element in a set of data.

1.1 Search via a walking algorithm

Search algorithms are very useful in the modern world, millions of users use the famous Page Rank algorithm finds everyday to find and index webpages. Different approaches have been made to find marked elements in an array of data. One of those involves using a classical walk, where an algorithm can be created to perform a search on any graph where there are marked elements [1]. This class of problem is known as spatial search problems, where in an undirected graph G, there are M marked vertices. The goal is to find a marked element by moving across the edges of the graph. The classical approach is to perform a random walk on G until a marked element is found, where the running time is known as the hitting time $\mathrm{HT}(G, M)$, quantifying the quality of the algorithm. An algorithm like this can be achieved using the below pseudo-code.

1. Generate $x \in X$ where a sample is drawn for the stationary distribution π of the Markov Chain P
2. Check is x is a marked element, if so then output x and end
3. Else, update x using a classical random walk (one random walk step along P)

The marked element will be found eventually, but it is not particularly efficient. This is due to statistics of the random walk, where the state space is not quickly traversed by a classical walker. The memory required for this is also of concern, as each vertex has to be loaded. With algorithms that have no deletion, this may not be scale-able to large data set searches [2]. The classical complexity of this algorithm can be bound as

$$S + T \cdot (U + C), \tag{5}$$

where $T = \sqrt{\mathrm{HT}(P, \{z\})}$, S is the set-up cost C is the checking cost and U is the update cost [1].

In this classical random walk, the hitting time can be estimated as

$$\mathrm{HT}(P, M) = \sum_{k=1}^{n-|M|} \frac{|\langle v'_k \mid U \rangle|^2}{1 - \lambda'_k} \tag{6}$$

To reduce the hitting time and have a better algorithm, one can increase the speed of the steps taken of alter the statistics of the walk process to match the structure of the data. As the step in the markov chain is determined by a random walk, the distribution of the outcomes will follow a Gaussian profile as seen in Section 1. It has a mean centered on zero and also the standard deviation as $\sqrt{n}$.

Classically the walking algorithms perform well in unstructured databases, but if the database has a structure then there will the algorithm may not be as efficient. In a weighted Markov chain [3], a faster traversal through the state-space may be preferable.

2 Quantum walk

The quantum equivalent of the random search walk was proposed initially in Ref.[4], a qubit state space is traversed randomly using the stochastic

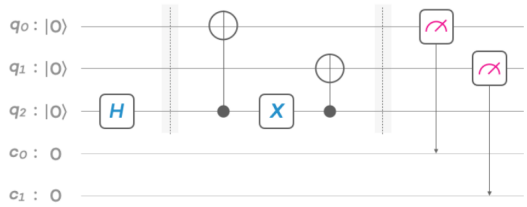

Figure 1: A quantum random walk circuit on four nodes, showing the use of the Hadamard gate and 2 CNOT gates to generate a stochastic process.

processes in quantum systems. Typically in the 1-D case, a superposition state is used to generate the randomness. A circuit for a quantum walk on four nodes is shown in fig.1.

To analyse the properties of this walk, the final position of a particle after N steps can be plotted against the classical version. The probability density is then compared in fig.2. It can clearly be seen that there is a quantum effect of interference occurring here, where the distribution does not follow the same Gaussian profile as seen in the classical case.

In the quantum case, the standard deviation is N, the number of steps taken. This has interesting implications for algorithms that use a quantum random walk.

A discrete quantum walk can be shown using a spin-1/2 particle on a linear array of discrete sites. The state

$$|\Psi\rangle = |s\rangle \otimes |\psi\rangle \tag{7}$$

with the $|s\rangle$ being the spin state and $|\psi\rangle$ being the position state.

By then applying a unitary transformation such as the Hadamard H gate, a superposition is obtained on the spin state. Then a unitary operation which conditionally shifts the particle left or right then will apply the quantum walk.

$$|\uparrow\rangle \otimes |0\rangle \xrightarrow{H} \frac{1}{\sqrt{2}}(|\uparrow\rangle + |\downarrow\rangle) \otimes |0\rangle \xrightarrow{S} \frac{1}{\sqrt{2}}(|\uparrow\rangle \otimes |1\rangle + |\downarrow\rangle \otimes |-1\rangle) \tag{8}$$

Returning to the quantum circuit with four nodes in fig.1, the state space is propagated by assigning each final qubit state to a node and then performing CNOT gates as a conditional shift operator.

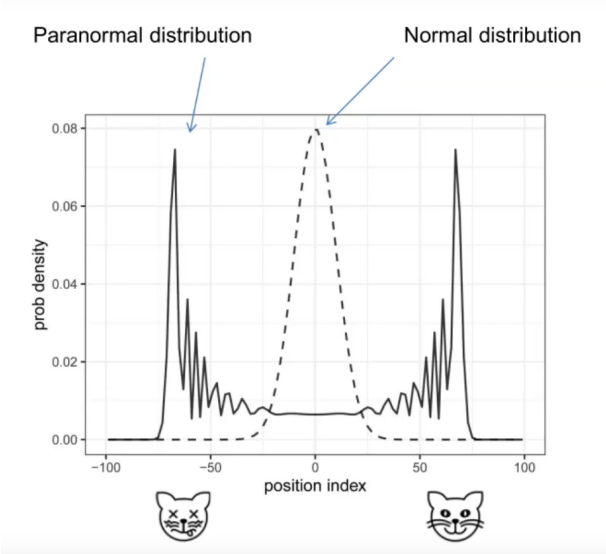

Figure 2: The probability distribution of the particle's position after N steps.

2.1 Quantum walk search algorithms

Quantum random walks have been thoroughly investigated in the recent decade as they have the potential to provide speed-ups in algorithms over classical counterparts. The famous Grover search algorithm can be shown to be a quantum random walk on the edges of a complete graph [3] and is therefore one of the first quadratic speed-ups demonstrated. However, the algorithmic potential of the quantum walk was first recognised in [5], where a quantum walk was used to emulate a Grover's search. After this initial proof of utility, a wave of the applications followed and can be decomposed into two categories. The first are derived from the fact that quantum walks can propagate through a tree exponentially faster than classical walks, therefore giving faster hitting times [6]. The second are algorithms based on quantum walks, giving quadratic speed ups to complexities.

An example is the work done by Ambainis [7], one of the first variations of a search using quantum walk which surpasses Grover search. This algorithm used quantum random walks for element distinctness, a problem of finding 2 marked elements that are the same in an array of n items. It can be classically shown that the element distinctness problem has a query complexity of $\mathcal{O}(n)$ [7], given that there is enough space to store all of the items n. The Ambainis

seminal paper reduced the number of queries to $\mathcal{O}(n^{2/3})$ with the use of a quantum walk.

Szegedy then collated the results from previous algorithms and showed that there was a generalisation to them by adapting the classical hitting time theory to a quantum one [3]. In the paper, the it is demonstrated that the quantum hitting time for finding a marked element in an ergodic Markov chain P with $P = P^T$ is the square root of the classical version. To give an example of a quantum search algorithm and the cost involved, the framework presented in [8] will be demonstrated. Firstly, the quantum state $|\pi\rangle$ is initialised where

$$|\pi\rangle = \sum_{x \in X} \sqrt{\pi_x} |x\rangle |p_x\rangle \tag{9}$$

which is the quantum equivalent of starting in a stationary distribution π in the classical search algorithm. This invokes a cost $S + U$, where S is the set-up cost and U is the update cost. The rest of the algorithm is as follows

Quantum search algorithm in the MNRS framework

1. 5 times:
   - Sample a state x from a stationary distribution π of P
   - if x is a marked element (i.e. $x \in M$), then stop!
2. Choose the number of repetitions T randomly from a set, which then defines the number of ancilla qubits (ks)
3. Prep the initial state $|\pi\rangle_d |0^{Tks}\rangle$
4. Repeat T times:
   - Apply a phase flip on the state $|x\rangle_d |y\rangle_d |z\rangle$ if $x \in M$
   - Perform a quantum circuit R(P) which is similar to a reflection ref(π). Where the quantum walk $W(P)_d$ is described by $W(P)_d =$ ref$(B)_d \cdot$ ref$(A)_d$

The quantum circuit R(P) involves a similar process to a Grover search as it involves a phase estimation circuit, however the difference is that it steps along the markov chain using a quantum random walk. Returning to the

cost of this algorithm, preparing $|\pi\rangle_d$ has an associated cost of $S + U$ and each iteration and phase flip costs C. In the quantum circuit R(P), each iteration a The advantage of using a quantum walk search is embedded in the mathematical derivation of a search algorithm and the cost of the Quantum search can be shown to be

$$\mathrm{S} + \frac{1}{\sqrt{\varepsilon}} \left[\left(\frac{1}{\sqrt{\delta}} \log \frac{1}{\sqrt{\varepsilon}} \right) \mathrm{U} + \mathrm{C} \right] \tag{10}$$

where ε is the lower bound on the probability than an element chosen from P is marked. In Szegedy's framework, Quantum random walks are supposed to give an exponential speed up in hitting times and other search algorithms have already been shown to give quadratic speed up in queries compared to their classical versions. The result in the MNRS framework corroborates this derviation for the search algorithm above.

2.2 Realising these Quantum Walks

Quantum walks have been demonstrated on 1-D state spaces [9] on trapped ions and also by Rohde in 2-D in optical hardware [10]. In the first paper, a single trapped ion's electronic states are used to provide the superposition state. Then, the motional degrees of freedom of the ion in its harmonic well are used to provide the random walk. A coupling interaction using carrier and displacement Raman beams are used to drive the spin dependent transition and thus generate a random walk. In the latter, an optical fibre network is used to create a coherent quantum walk over 12 steps and 169 positions. Although a range of other experimental proposals have been realised regarding quantum walks, to date no search algorithms have been performed given that the number of qubits required to perform any one of these algorithms is high. In the future, as the the FTQC regime is approached the implementation of quantum walk searches could be feasible.

References

[1] Hari Krovi et al. "Quantum Walks Can Find a Marked Element on Any Graph". In: *Algorithmica* 74.2 (Mar. 2015), pp. 851–907. ISSN: 1432-0541. DOI: 10.1007/s00453-015-9979-8. URL: http://dx.doi.org/10.1007/s00453-015-9979-8.

[2] Feng Xia et al. “Random Walks: A Review of Algorithms and Applications”. In: *CoRR* abs/2008.03639 (2020). arXiv: 2008.03639. URL: https://arxiv.org/abs/2008.03639.

[3] Mario Szegedy. “Quantum speed-up of Markov Chain based algorithms”. In: *Proceedings - Annual IEEE Symposium on Foundations of Computer Science, FOCS* (Nov. 2004), pp. 32–41. DOI: 10.1109/FOCS.2004.53.

[4] David A. Meyer. “From quantum cellular automata to quantum lattice gases”. In: *Journal of Statistical Physics* 85.5-6 (1996), pp. 551–574. DOI: 10.1007/bf02199356.

[5] Neil Shenvi, Julia Kempe, and K. Birgitta Whaley. “Quantum random-walk search algorithm”. In: *Physical Review A* 67.5 (2003). DOI: 10.1103/physreva.67.052307.

[6] Andrew M. Childs, Edward Farhi, and Sam Gutmann. “An example of the difference between quantum and classical random walks”. In: *Quantum Information Processing* 1.1/2 (2002), pp. 35–43. DOI: 10.1023/a:1019609420309.

[7] Andris Ambainis. “Quantum Walk Algorithm for Element Distinctness”. In: *SIAM Journal on Computing* 37.1 (2007), pp. 210–239. DOI: 10.1137/s0097539705447311.

[8] Frédéric Magniez et al. “Search via Quantum Walk”. In: *SIAM Journal on Computing* 40.1 (2011), pp. 142–164. DOI: 10.1137/090745854.

[9] Peng Xue, Barry C. Sanders, and Dietrich Leibfried. “Quantum Walk on a Line for a Trapped Ion”. In: *Physical Review Letters* 103.18 (2009). DOI: 10.1103/physrevlett.103.183602.

[10] Andreas Schreiber et al. “A 2D Quantum Walk Simulation of Two-Particle Dynamics”. en. In: *Science* 336.6077 (Apr. 2012). arXiv: 1204.3555, pp. 55–58. ISSN: 0036-8075, 1095-9203. DOI: 10.1126/science.1218448. URL: http://arxiv.org/abs/1204.3555 (visited on 10/29/2021).