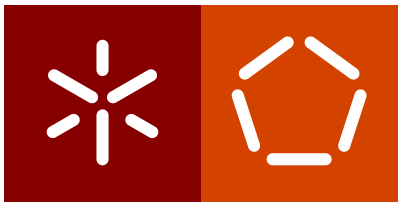


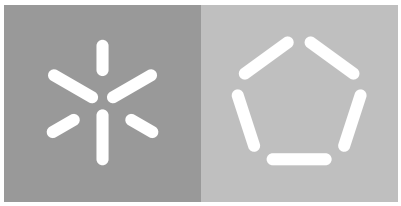
Universidade do Minho
Escola de Engenharia??
Departamento de Informática??

Jaime Santos

Staggered Quantum Walks in Qiskit??
Second Part of Title

First Part of Subtitle
Second part of Subtitle

May 2021



Universidade do Minho
Escola de Engenharia??
Departamento de Informática??

Jaime Santos

Staggered Quantum Walks in Qiskit??
Second Part of Title

First Part of Subtitle
Second part of Subtitle

Master dissertation
Master Degree in Física da Informaçao

Dissertation supervised by
Luís Barbosa
Bruno Chagas

May 2021

ACKNOWLEDGEMENTS

Write acknowledgements here

ABSTRACT

Implementation of staggered quantum walks in qiskit.

RESUMO

Pensar no que escrever aqui. Hello!

CONTENTS

1 INTRODUCTION 1
- 1.1 Brief history of quantum computing 1
- 1.2 Classical and Quantum Random Walks 3
- 1.3 State of the Art quantum walks implementations 6
- 1.4 Text overview and contributions 7

2 QUANTUM COMPUTING 9
- 2.1 Mathematical foundations QM 9
- 2.2 Quantum Fourier Transform 9

3 QUANTUM WALKS 12
- 3.1 Classical Random Walk 12
- 3.2 Coined Quantum Walk 13
- 3.3 Continuous-Time Quantum Walk 18
- 3.4 Staggered Quantum Walk 23

4 SEARCHING PROBLEMS 26
- 4.1 Grover's algorithm 26
  - 4.1.1 One marked element 28
  - 4.1.2 Multiple marked elements 30
  - 4.1.3 Single-Shot Grover 31
- 4.2 Coined Quantum Walk 32
- 4.3 Continuous-Time Quantum Walk 34
- 4.4 Staggered Quantum Walk 36

5 IMPLEMENTATIONS AND APPLICATIONS 39
- 5.1 Coined 39
- 5.2 Continuous 44
- 5.3 Staggered 48
- 5.4 Search Problems with Qiskit 48
  - 5.4.1 Grover 48
  - 5.4.2 Coined 50
  - 5.4.3 Continuous 52
  - 5.4.4 Staggered 55

A SUPPORT MATERIAL 63

LIST OF FIGURES

Figure 1 Temp 10

Figure 2 Classical Walk Temp. 12

Figure 3 Probability distribution for the coined quantum walk on a line, after 100 steps, with initial condition $|\Psi(0)\rangle = |0\rangle |x = 0\rangle$ and the Hadamard coin. 16

Figure 4 Probability distribution for the coined quantum walk on a line, after 100 steps, with initial condition $|\Psi(0)\rangle = -|1\rangle |x = 0\rangle$ and the Hadamard coin. 16

Figure 5 Probability distribution for the coined quantum walk on a line, after 100 steps, with initial condition $|\Psi(0)\rangle = \frac{|0\rangle - i|1\rangle}{\sqrt{2}} |x = 0\rangle$ and the Hadamard coin. 17

Figure 6 Probability distribution for the continuous-time quantum walk on a line, at t = 100, with initial condition $|\Psi(0)\rangle = |0\rangle$ and $\gamma = \frac{1}{2\sqrt{2}}$. 20

Figure 7 Probability distribution for the continuous-time quantum walk on a line, after 100 steps, with initial condition $|\Psi(0)\rangle = |0\rangle$ and $\gamma = \frac{1}{6\sqrt{2}}$. 21

Figure 8 Temporary 21

Figure 9 Probability distribution for the continuous-time quantum walk on a line, after 100 steps, with initial condition $|\Psi(0)\rangle = \frac{|0\rangle + |1\rangle}{\sqrt{2}}$ and $\gamma = \frac{1}{2\sqrt{2}}$. 22

Figure 10 Temporary 22

Figure 11 Probability distribution for the staggered quantum walk on a line after 50 steps, with initial condition $|\Psi(0)\rangle = \frac{|0\rangle + |1\rangle}{\sqrt{2}}$, for multiple angles. 25

Figure 12 $|\Psi(0)\rangle = |0\rangle$ 25

Figure 13 $|\Psi(0)\rangle = |1\rangle$ 25

Figure 14 Grover one marked element temp. 29

Figure 15 Grover Multiple marked temp. 30

Figure 16 Single Shot temp 31

Figure 17 Discrete-time coined quantum walk search for a complete graph with 16, 32 and 64 nodes. 34

Figure 18 Value of the difference between the largest eigenvalue and the second largest, plotted as a function of γN, for $N = 512$. 35

Figure 19 Continuous quantum walk search for a complete graph with 16, 32 and 64 vertices. 36
Figure 20 Maximum probability of the marked element as a function of the θ value plotted from 0 to π for number of nodes $N = 64, 128$ and 256. 37
Figure 21 Staggered quantum walk search for a complete graph with 16, 32 and 64 nodes. 38
Figure 22 Douglas wang shift operator 40
Figure 23 Douglas wang coined quantum walk circuit 40
Figure 24 Toffoli decomposition 40
Figure 25 General decomposition 41
Figure 26 Temp 42
Figure 27 Temp 42
Figure 28 Temp 43
Figure 29 Temp 43
Figure 30 Temp 46
Figure 31 Temp 46
Figure 32 temp 46
Figure 33 Temp 47
Figure 34 Temp 47
Figure 35 Temp 49
Figure 36 Temp 49
Figure 37 Temp 49
Figure 38 Temp 50
Figure 39 Douglas wang coined quantum walk circuit 51
Figure 40 Temp 51
Figure 41 Temp 51
Figure 42 Temp 52
Figure 43 Temp 52
Figure 44 Temp 53
Figure 45 Temp 54
Figure 46 Temp 54
Figure 47 Temp 54
Figure 48 Temp 54
Figure 49 Temp 55

LIST OF TABLES

1

INTRODUCTION

1.1 BRIEF HISTORY OF QUANTUM COMPUTING

The modern understanding of computer science was firstly announced by Turing (1936) where he developed the abstract concept of what is now called a *Turing machine*. These machines are the mathematical foundation of programmable computers, and Turing showed that there is a *Universal Turing Machine* that can be used to simulate any other Turing Machine. This means that if an algorithm can be executed in any piece of hardware, then there is a Universal Turing Machine that can acompish the same task. This is known as the *Church-Turing thesis*, which connects the concept of what classes of algorithms can be run in some physical device with the mathematical framework of a Universal Turing Machine.

The paper published by Turing set in motion a series of events which led to the rapid advancement of eletronic computers and computer science. One of the earliest theoretical models developed by John von Neumann (work later published in von Neumann (1993)), presented how to assemble all the necessary parts to create a computer with all the capabilities of a Universal Turing Machine. The true explosion of innovation in this field came after the invention of the transistor in 1947 by John Bardeen and Walter Brattain. The creation of the transistor led to an unpredecent growth quantified by Moore (1965) where he created *Moore's law*, stating that computer power will double with constant cost approximately every two years. Moore's law has roughly held true throughout the decades, by the ever increasing miniaturization of the transistor technology. However, conventional fabrication methods run into a problem of scale, as quantum effects begin to interfere more as the size of the devices becomes smaller.

Feynman (1959) recognized such a miniturization was the way forward, and even predicted the problems quantum effects presented to a classical computer. With an amazing stroke of insight, Feynmann imagined that these effects could be exploited given the right computational paradigm. Quantum computing begins to take form in later work developed by Benioff (1980), where the earliest quantum mechanical model of a computer was described. In this paper, Benioff showed that a computer working under the laws of quantum mechanics could be used to express a Shrodinger equation description of Turing machines. Shortly

after, Feynman (1982) pointed out that simulating quantum systems on classical computers is inneficient, and suggested using quantum computers for this purpose. Additional work in the following decade further explored this idea and showed that there are systems that quantum computers can simulate, which have no known efficient simulation on a classical computer, and even today this continues to be one of the most promising fields in quantum computing.

Driven by the work of Turing, Deutsch (1985) questioned if a stronger version of the Church-Turing thesis could be derived from the laws of physics. The strong Church-Turing thesis states that any algorithmic process can be simulated efficiently using a probabilistic Turing machine, and Deutsh was set to define some device that could efficiently simulate an arbitrary physical system. Whether Deutsh's formulation of a Universal Quantum Computer is sufficient for this function is still an open question, but what he accomplished was a challenge to the strong Church-Turing thesis by suggesting that there are tasks a quantum computer can accomplish efficiently that a probabilistic Turing machine cannot. Deutsch and Jozsa (1992) present an example of a quantum algorithm that is exponentially faster than a classical counterpart, the *Deusth-Josza algorithm* that determines if a function is constant or balanced. Even though of little practical use, this is one of the first examples of possible advantages a quantum computer may have over a classical one.

Even though the Deutsh-Josza algorithm might not have real world applications, it led to further research on finding other such types of algorithms. Shor (1994a) showed that the problem of finding prime factors of an integer and the *discrete logarithm* problem can be effieciently solved by a quantum computer by an exponential factor. This brought a lot of interest to quantum computing, since both of these problems have real world applications and no efficient classical solution was/is known. Furthermore, most modern popular algorithms used for cryptography rely on the fact that the integer factorization or discrete logarithm problems are not effiently solvedi. Since this is no longer the case, a new field has emerged called *Post-quantum cryptography*, whose purpose is to find suitable classical algorithms for cryptography that are not efficiently solved by quantum computing.

A more modest, but very relevant advantage was presented by Grover (1996) where he presented a quantum algorithm that promised to speed up unstructured database searches quadratically. Even though it's not an exponential improvement like Shor's algorithm, search-based algorithms are useful in many contexts, so even a "small" quadratic gain generated a lot of interest.

Contemporary to computer science, information theory is another field very relevant to this topic. Shannon (1948) revolutionized how communication and information are understood. In his paper, Shannon was interested in defining what resources are requiried to send information over a communication channel and how to reliably send that information mitigating the effects of noise. This led to the creation of the two fundamental theorems of

information theory. Firstly, Shannon's *noiseless channel coding theorem* specifies what resources are needed to store information sent from a source. Secondly, the *noisy channel coding theorem*, specifies how much information can be sent through a channel subject to noise. Even though Shannon's second theorem does not define any specific methodology to reduce noise, it sets an upper limit on how much noise can be mitigated through said methodology. These are known as *error-correcting codes* and research has developed better and better codes that get closer and closer to Shannon's limit, and they are used wherever there is need to store or transmit information.

Similar progress was made in quantum information theory. Schumacher (1995) developed a quantum version of Shannon's noiseless coding theorem, where he defined a *quantum bit* as a physical resource. There is no analogue for the second Shannon theorem, but that didn't stop the development of quantum error-correcting theory. For example, Calderbank and Shor (1996) and Steane (1996) proposed an important class of quantum error-correcting codes known as CSS.

Error-correcting was designed to protect quantum states, but another discovery by Bennett and Wiesner (1992) showed another interesting aspect about quantum information when transmitting classical information through a quantum channel. They explained how to send two classical bits of information using only one qubit, in a phenomenon known as *superdense coding*.

Another interesting application of quantum information is in the field of cryptography. Wiesner (1983) showed how quantum mechanics could be used to make sure that a information sent could not be interfered with without destroying it. Building on this work, Bennett and Brassard (1984) proposed a quantum key distribution protocol between sender and receiver that could not spied upon without notice. Many other protocols have since been proposed and experimental prototypes developed.

Finally, another interesting field within quantum computation is based on the concept of *distributed quantum computation*. Quantum clusters show promise since they require exponentially less communication to solve certain problems, such as modeling quantum systems, but are stil in their infancy due to technical restrictions. There has been increasing international interest in taking advantage of these systems to build a *quantum internet* which promises better and safer transmission of information, but there are still many technological improvements to be made before this becomes a mainstream reality.

1.2 Classical and Quantum Random Walks

The strong Church-Turing thesis, that states that any algorithmic process can be simulated efficiently using a Turing Machine, was challenged by Solovay and Strassen (1977) where they presented what is known as the *Solovay-Strassen primality test*. They showed that it

is possible to test whether a integer is prime or composite using a randomized algorithm. The implication is that, because of the randomness, the Solvay-Strassen primality test does not determine with certainty whether a integer is prime or composite, rather it computes that a number is *probably* prime or else *certainly* composite. This is of significance since no deterministic test for primality was known at the time (nor it is now), meaning that this was an example of a class of problems that could not be efficiently solved by a conventional deterministic Turing Machine.

This led to a modification of the Church-Turing thesis, now stating that any algorithm can be simulated efficiently using a *probabilistic* Turing machine. The discovery of more instances of such algorithms followed, Motwani and Raghavan (1995) and Papadimitrious (1994) show several problems that can be solved based on randomized algorithms. For example, the *Quicksort* algorthim, developed by Hoare (1961), has a high probability of finishing in $O(n \log n)$. In contrast to many deterministic algorithms that require $O(n^2)$ time. They also show algorithms that take advantage of *Markov chains* and the *Monte Carlo method*. The volume of a convex body, proposed by Dyer et al. (1991), can be estimated by a randomized algorithm in polynomial time; the permanent of a nonnegative entry matrix can also be approximately calculated in probabilistic polynomial time as was shown by Jerrum et al. (2004) and the *k-SAT* and satisfiability with restrictions problem by Schöning (1999).

Random walks, as the name suggests, belong to this class of algorithms. Pearson (1905) coined the term random walk, and they can be described as path consisting of a succession of steps determined by a stochastic process, over a mathematical space. They are a special case of *Markov chains*, which are stochastic processes that assume discrete values and whose next state is dictated by a deterministic or random rule based only on the current state. This is a useful framework, since it can be used to explain the behaviour of systems across many fields, from the Brownian movement of particles moving through a gas, to the price of a fluctuating stock as shown by Cootner (1967).

The quantum analogue of the random walk was firstly developed by Aharonov et al. (1993), where they defined the *coined quantum random walk*. This model consists of a walker and a coin that determines the movement of the walker, which are both quantum systems where time is a discrete variable dictated by the successive quantum coin flips and shifts in position. Nayak and Vishwanath (2000) and Aharonov et al. (2001) presents the first analysis of the quantum walk on a graph described by a line. Further work by Inui et al. (2003) studies the behaviour of the walk on grids and Aharonov et al. (2001) on general regular graphs. The first algorithmic applications appear in the work of Shenvi et al. (2003) where they constructed a search problem based on the quantum random walk, and Ambainis (2007) applied it to the element distinction problem. On a more theoretical note, Konno (2002) demonstrated how the classical and quantum models of the random walk on the line differ, and Grimmett et al. (2003) generalized this to higher dimensions. Lovett

et al. (2010) demonstrated that any quantum algorithm can be reformulated as a discrete time quantum walk algorith, effectively showing that this model can be used for universal quantum computation.

A different model for quantum random walks emerged from the work of Farhi and Gutmann (1998), where a different way of computing a search problem was presented. They showed that evolving a system in time between an initial and final Hamiltonian is analogous to the Grover algorithm, but continuous in time. Farhi et al. (2000) revised their work, now known as an adiabatic evolution, to solve boolean *sat* problems. Childs and Goldstone (2004) formulated a model of a quantum walk in terms of adiabatic evolution, known as *continuous time quantum walk* or *adiabatic quantum walk*. Aharonov et al. (2007) showed that any quantum algorithm can be efficiently simulated using adiabatic evolution, meaning that it is polynomially equivalent to the conventional quantum computation model. Further work by Childs (2009) showed that this model is indeed universal. The main difference between these two quantum walk models relies on the fact that in the discrete case, the system evolves with the flipping of a coin and subsequent movement of the walker, whilst in the adiabatic case the system evolves smoothly in time.

Yet another way of thinking about quantum walks was announced by Szegedy (2004), where he describes a discrete model based on Markov chain random walks. At the foundation of this model is the duplication of a gaph, process by which a bipartite graph is created. Magniez et al. (2007) show how to use this walk for triangle detection in an undirected graph. Magniez et al. (2006) proposed a search problem, which takes advantage of ergodicity and simmetry properties of Markov chains, with quadratic gain compared to classical algorithms. Problems like element distinctness, matrix product verification and others were formulated within this framework by Santha (2008). Further work by Portugal (2015) established a connection between the coined and Szegedy's quantum walk, by defining a model that encompasses and expands the latter.

Patel et al. (2005) pointed that, at the time, there was confusion surrounding the scaling behaviour of discrete and adiabatic quantum walk algorithms. They argued that this was because the former model used a coin, which is an extra resource, and the latter didnt't. So, in an attempt to resolve this confusion, they showed that a discrete time quantum walk could be constructed without the use of a coin. A new way of thinking about quantum walks came with the development of the methods behind the construction of evolution operators, by Falk (2013), introducing the concept of*tesselations*, based on local diffusion operators. Portugal et al. (2015) studied this model applied to a line graph, using the tesselation idea to construct the evolution operators. Further work by Portugal et al. (2016) formalized this approach naming it *staggered quantum walk*, and showed instances where the Szegedy quantum walk is equivalent. This suggests that this is a more general model, being able to describe other discrete time quantum walks. Portugal et al. (2017) delved deeper into this

topic, adding Hamiltonians to the model, and Portugal and Fernandes (2017) shows how this can be instanced as a search problem in a grid. Finally, Coutinho and Portugal (2018) analyze how a continuous-time quantum walk can be casted into this discrete model and Moqadam et al. (2017) show a possible physical implementation of this walk.

1.3 State of the Art quantum walks implementations

One of the earliest works on a more computational approach to quantum random walks was by Marquezino and Portugal (2008), where they created a general simulator for discrete-time quantum walks on one- and two-dimensional lattices. They argued that this framework allowed reasearchers to focus more on the mathematical aspect of quantum walks, instead of the specific numerical implementations. Further work on these lattices was later presented by Sawerwain and Gielerak (2010), where they studied the simulation of quantum walks by taking advantage of the GPU and CUDA techonlogy. Another interesting program for simulating discre-time quantum walks came with the work of Berry et al. (2011). This package allows for direct simulation of these walks, and visualisation of the time-evolution on arbitrary undirected graphs. It also allowed for plotting of continuous-time quantum walks, provided the data was provided externally. There are, however, direct simulators of continuous-time quantum walks, the earliest being by Izaac and Wang (2015). Their distributed memory software claims to be able to perform efficient simulation of multi-particle continuous-time quantum walk based systems, on *High Performance Computing* platforms. Falloon et al. (2017) provide a *Mathematica* package that implements a simulator of *Quantum Stochastic Walks*, which are a generalization of the continuous-time model. These walks incorporate both coherent and incoherent dynamics, which means that quantum stochastic walks can be instantiated as both quantum walks and classical random walks. What this paper then provides is a way of implementing quantum walks on directed graphs, opening the door to applications ranging from the capture of energy by photosyntethic protein complexes, to page ranking algorithms used by search engines. This package was ported to and expanded in the *Julia* programming language by Glos et al. (2018).

In order to truly harness the power of quantum computing, however, one must be able to perform these algorithms on quantum hardware. For this purpous, various implementations of quantum walks on quantum circuits have been purposed. Douglas and Wang (2009) pioneered this approach by developing efficient quantum circuits for discrete-time quantum walks on highly symmetric graphs, whose resources scale logarithmically with the size of the state space. Shakeel (2020) presented a new approach for building circuits for the discrete model, reducing resource requirement by using the quantum Fourier transform. For the continuous-time quantum walk, work by Qiang et al. (2016) presents efficient quantum circuits for the circulant graph class, and also an experimental implementation on a

photonic quantum processor. In the same year, Loke and Wang (2017a) showed how to build continous-time quantum walk circuits for composite graphs, namely commuting graphs and Cartesian product of graphs. Considering the Szegedy quantum walk, Chiang et al. (2009) proposed an efficient method of creating quantum circuits for this model. In their work, they showed how to derive a quantum version of the arbitrary sparse classical random walk by approximating a *quantum update rule* with circuit complexity scaling linearly with the degree of sparseness of the structure. Loke and Wang (2017b) developed this method by showing that an efficient circuit for the Szegedy quantum walk can be constructed even if the structures are not sparse, given they posses translational symmetry in the columns of the transitional matrix. More specifically, they identified that the class of cyclic and bipartite graphs are compatible with this approach. Another interesting result in this paper was the creation of circuits that implement a quantum analogue of Google's *Page Rank* algorithm, in terms of Szegedy walks.

On the experimental side, there have been various implementations of quantum walks on quantum computers. On IBM's hardware, work by Balu et al. (2017) presents an efficient implementation of topological quantum walks where they ran the circuit on a five qubit computer over a 4 vertex lattice. Georgopoulos and Zuliani (2019) implements two instances of the discrete-time quantum walk. The first is based on the work of Douglas and Wang (2009) using generalized CNOT gates, and the second uses a rotational approach to the CNOT decomposition, which saves using an extra ancilla qubit register. They noted that IBM's simulator backend corresponded to the theoretical predictions, however the circuit for over three qubits was too much for the quantum hardware at that time. The paper presented by Shakeel (2020) also uses IBM's hardware to perform their formulation of the discrete-time quantum walk. For the staggered quantum walk model, work by Acasiete et al. (2020) obtains meaningful results when using the quantum computers to study the dynamics of this model on various graphs with 16 elements, requiring 4 qubits. They also modify these circuits to accomodate an oracle, showing that a spatial search algorithm could be performed on IBM's quantum computers for a search space of size 8, and with a bit of noise but still satisfactory for size 16.

1.4 Text overview and contributions

The motivation behind the contents in this dissertation is to create an expanded overview on the topic of quantum random walks. To better contextualize the rest of this thesis, chapter 2 presents the basic concepts and mathematical tools needed to study quantum walks.

Chapter 3 consists of the study of three major quantum random walk models, more specifically the discrete-time coined quantum walk, the continuous-time quantum walk and the staggered quantum walk. For each of these models, this work presents the theoretical

framework as well as Python implementations. In these simulations, the dynamics of the walks are analised by changing various parameters, plotting the resulting probability distributions and seeing how these parameters alter the shape, propagation and other features of the quantum random walks.

Chapter 4 follows this approach, but now the structure where these walks take place is a complete graph and the goal is to find a marked element. For this purpose, the section about Grover's algorithm is used to introduce the notion of quantum searching problems. The following sections show how to change the various models of quantum walks to accomodate an oracle, thus performing an element search in time similar to Grover's algorithm.

Finally, chapter 5 is dedicated to constructing circuits for the models previously defined, using IBM's software *Qiskit* and their hardware. The first three sections are used to create circuits for the dynamics of the walks. The biggest contribution being the circulant graph approach to building diagonal operators for the continuous-time quantum walk, which can be easily translated to Qiskit circuits. Although work by Qiang et al. (2016) pioneered this approach, the work presented in this thesis aims to give a clear description on how to build these circuits in Qiskit and an original analysis on how the approximate quantum Fourier transform affects the accuracy of the results and the number of operations needed to perform the quantum walk. The last section shows how introduce an oracle to the various circuits in order to perform a searching problem. For the continuous-time case, circulant graphs are again used in an original implementation of the searching problem making use of diagonal operators and the Suzuki-Trotter expansion.

2

QUANTUM COMPUTING

2.1 MATHEMATICAL FOUNDATIONS QM

2.2 QUANTUM FOURIER TRANSFORM

As was seen in section **??**, quantum computers can perform certain tasks more efficiently than classical computers. Another such example is the problem of finding the prime factorization of an n-bit integer, which the most efficient solution to date, proposed by Pollard et al. (1994), requires $e^{O(n^{\frac{1}{3}} \log^{\frac{2}{3}} n)}$ operations. In contrast, a quantum algorithm proposed by Shor (1994b) accomplishes the same task in $O((\log n)^2 (\log \log n)(\log \log \log n))$ operations, which is an exponential gain due to the efficiency of the quantum Fourier transform.

The quantum Fourier transform is an implementation of the discrete Fourier transform over amplitudes of quantum states. It offers no speed ups when used in computing Fourier transforms of classical data, since the amplitudes cannot be accessed directly by measurement. Moreover, there is no known generalized efficient way of preparing the initial state to be Fourier Transform. What this means is that the uses of the QFT are not in the straightforward way of calculating discrete Fourier transforms, but in the form of algorithms, such as *phase estimation*, that take advantage of it's properties. This transform can be described as the following operation over an orthonormal basis $|0\rangle, |1\rangle, \cdots, |N-1\rangle$

$$QFT(|j\rangle) = \frac{1}{\sqrt{N}} \sum_{k=0}^{N-1} e^{\frac{2\pi i j k}{N}} |k\rangle, \tag{1}$$

where $N = 2^n$. With a little bit of algebra, this can be rewritten as a product

$$\begin{aligned}
\frac{1}{\sqrt{N}} \sum_{k=0}^{N-1} e^{\frac{2\pi i j k}{2^n}} |k\rangle &= \frac{1}{\sqrt{N}} \sum_{k_1=0}^{1} \cdots \sum_{k_n=0}^{1} e^{2\pi i j (\sum_{l=1}^{n} k_l 2^{-l})} |k_1 \cdots k_n\rangle \\
&= \frac{1}{\sqrt{N}} \sum_{k_1=0}^{1} \cdots \sum_{k_n=0}^{1} \bigotimes_{l=1}^{n} e^{2\pi i j k_l 2^{-l}} |k_l\rangle \\
&= \frac{1}{\sqrt{N}} \bigotimes_{l=1}^{n} (\sum_{k_l=0}^{1} e^{2\pi i j k_l 2^{-l}} |k_l\rangle) \\
&= \frac{1}{\sqrt{N}} \bigotimes_{l=1}^{n} (|0\rangle + e^{2\pi i j 2^{-l}} |1\rangle)
\end{aligned} \tag{2}$$

$$QFT(|x_1, ...x_n\rangle) = \frac{(|0\rangle + e^{2\pi i 0.x_n} |1\rangle)(|0\rangle + e^{2\pi i 0.x_{n-1}x_n} |1\rangle) \cdots (|0\rangle + e^{2\pi i 0.x_1 x_2 \cdots x_n} |1\rangle)}{2^{\frac{N}{2}}}, \tag{3}$$

where $x = x_1 2^{n-1} + x_2 2^{n-2} + \cdots + x_n 2^0$ and the notation $0.x_1 x_{l+1} \cdots x_n$ represents the binary fraction $\frac{x_l}{2^{i^0}} + \frac{x_{l+1}}{2^1} \cdots \frac{x_m}{2^{m-l+1}}$. This is a very useful representation because it makes constructing an efficient circuit much simpler, as can be seen in figure 1. However, the circuit implementation of the QFT requires exponentially smaller phase-shift gates as the number of qubits increases. This can be somewhat mitigated by eliminating the smaller phase-shitft gates at the cost of some accuracy, as was shown in Coppersmith (1994) where he defined the *approximate* quantum Fourier transform. This approximation requires only $O(n \log n)$ gates, and work by Barenco et al. (1996) and Cheung (2004) established lower bounds for the probability of the approximate state accurately representing the state without approximation.

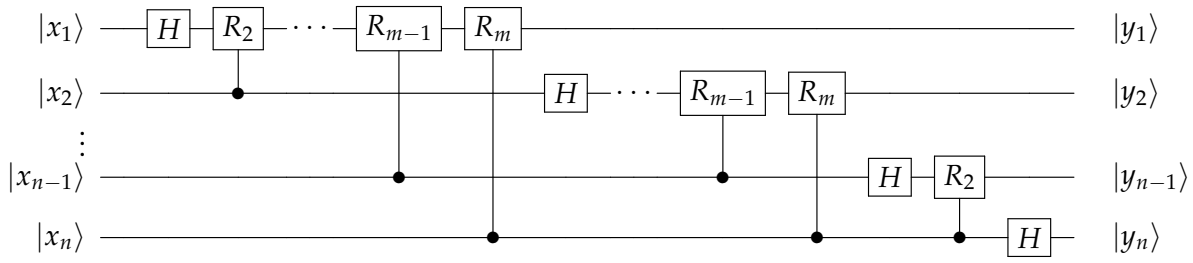


Figure 1: Temp

The rotation R_k in figure 1 is defined as the controlled version of

$$R_k = \begin{pmatrix} 1 & 0 \\ 0 & e^{\frac{2\pi i}{2^k}} \end{pmatrix}. \tag{4}$$

To verify that this circuit is the QFT, consider the state $|x_1 \cdots x_n\rangle$ as input. Applying the Hadamard gate on the first qubit produces the state

$$H|x_1 \cdots x_n\rangle = \frac{1}{\sqrt{N}}(|0\rangle + e^{2\pi i 0.x_1}|1\rangle)|x_1 \ldots x_n\rangle . \tag{5}$$

The next operation is the rotation R_2, controlled by the second qubit, resulting in state

$$\frac{1}{\sqrt{N}}(|0\rangle + e^{2\pi i 0.x_1 x_2}|1\rangle)|x_1 \ldots x_n\rangle . \tag{6}$$

Applying the succesive rotations up to R_n appends an extra bit to the phase of the first $|1\rangle$, ultimately becoming

$$\frac{1}{\sqrt{N}}(|0\rangle + e^{2\pi i 0.x_1 x_2 \cdots x_n}|1\rangle)|x_1 \ldots x_n\rangle . \tag{7}$$

A similar process is applied to the second qubit, and at the end of the rotations the state is

$$\frac{1}{\sqrt{N}}(|0\rangle + e^{2\pi i 0.x_1 x_2 \cdots x_n}|1\rangle)(|0\rangle + e^{2\pi i 0.x_2 \cdots x_n}|1\rangle)|x_1 \ldots x_n\rangle , \tag{8}$$

and the successive application of this process to the remaining qubits results in state

$$\frac{1}{\sqrt{N}}(|0\rangle + e^{2\pi i 0.x_1 x_2 \cdots x_n}|1\rangle)(|0\rangle + e^{2\pi i 0.x_2 \cdots x_n}|1\rangle) \cdots (|0\rangle + e^{2\pi i 0.x_n}|1\rangle)|x_1 \ldots x_n\rangle , \tag{9}$$

confirming that this is indeed the Fourier transform derived in equation 3 minus the order of the qubits, which is reversed. It also shows that the QFT is unitary, since all operations in the circuit are unitary.

Counting the number of gates on the circuit, one can conclude that the first qubit will have 1 Hadamard gate followed by $n-1$ controlled rotations. The second qubit is another Hadamard followed by $n-2$ controlled rotations. After n qubits, the total number of gates will be $\frac{n(n+1)}{2}$. This means the circuit provides a $O(n^2)$ algorithm, compared to the fastest classical algorithm, the *Fast Fourier Transform* which requires $O(n2^n)$ operations. This is an exponential gain, which can be improved upon at the cost of accuracy, but it's not a replacement for calculating classical Fourier transforms for the aforementioned reasons.

3

QUANTUM WALKS

3.1 CLASSICAL RANDOM WALK

The term *random walk*, firstly introduced by Pearson (1905), is classically defined as a stochastic process that models the path a walker would take through a mathematical space, where each step made by the walker is random. This can be used to model systems such as a molecule displaying Brownian motion in a fluid, or even fluctuating stock prices as can be seen in Sottinen (2001).

The simplest instance of this walk is on a discretely numbered line, whose mathmatical space is composed of integer numbers. Here, the walker can only advance with equal probability in one of two directions, depending on the outcome of a random event such as tossing a coin. This was coded in Python, and the result of iterating the walk several times is a binomial distribution centered around the starting position.

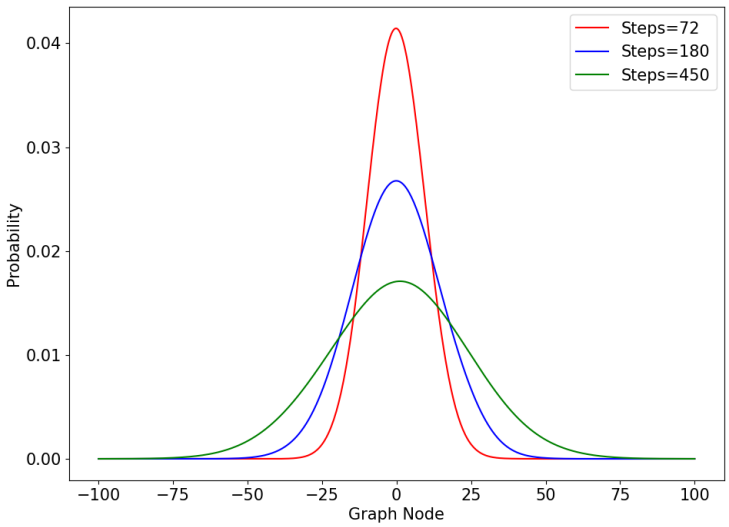


Figure 2: Classical Walk Temp.

The number of steps (iterations) directly affects how far the walker can reach, as can be seen in figure 2. As the number of steps increases, the height of each curve at the starting position decreases and the width of the curves increases. This relationship can be captured by the *position standard deviation*, and Portugal (2018) shows that the standard deviation is

$$\sigma(t) = \sqrt{t}. \tag{10}$$

In other words, equation 10 represents the rate at which a walker moves away from the origin.

Note that this algorithm can be abstracted to graphs of higher dimensions. For example, in a two dimensional lattice, a walker would be transversing a plane with integer coordinatesi, choosing one of four directions in every intersection. Notably, Pólya (1921) proved that a walker in a two dimensional lattice will almost surely return to the origin at some point. However, the probability of returning to the origin decreases as the number of dimensions increases, as shown by Montroll (1956) and Finch (2003).

It is worth noting that a random walk, over a graph whose nodes are weighed and directed, is analagous to a *discrete-time Markov chain*[1].

The following sections will be used to describe various models of a quantum counterpart of the classical random walk.

3.2 COINED QUANTUM WALK

In the quantum case, the walker is a quantum system whose position on a discretely numbered line, is described by a vector $|x\rangle$ in Hilbert Space. The next position of the system will depend, in part, of a unitary operator, which can be viewed as a quantum coin. The analogy is, if the coin is tossed and rolls "heads", for example, the system transitions to position $|x+1\rangle$, otherwise it advances to $|x-1\rangle$. From a physical perspective, this coin can be the spin of an electron or the chirality of a particle, for example, and the outcome of measuring these properties decides whether the walker moves left or right. The coin is a unitary operator defined as

$$\begin{cases} C\,|0\rangle\,|x\rangle = a\,|0\rangle\,|x\rangle + b\,|1\rangle\,|x\rangle \\ C\,|1\rangle\,|x\rangle = c\,|0\rangle\,|x\rangle + d\,|1\rangle\,|x\rangle\,, \end{cases} \tag{11}$$

1 A Markov chain can be described as a sequence of stochastic events where the the probability of each event depends only on the state of the previous event.

where a, b, c and d are the amplitudes associated with each outcome of the coin toss. One of the most commonly used coins is the unbiased coin, also known as Hadamard operator

$$H = \begin{pmatrix} a & c \\ b & d \end{pmatrix} = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}, \tag{12}$$

which will be the one used in this example.

The Hilbert space of the system is $\mathscr{H} = \mathscr{H}_C \otimes \mathscr{H}_P$, where $\mathscr{H}_C$ is the two-dimensional Hilbert space associated with the coin and $\mathscr{H}_P$ is the Hilbert space of the walker.

The transition from $|x\rangle$ to either $|x+1\rangle$ or $|x-1\rangle$ must be described by a unitary operator, the *shift operator*

$$\begin{cases} \mathcal{S} |0\rangle |x\rangle = |0\rangle |x+1\rangle \\ \mathcal{S} |1\rangle |x\rangle = |1\rangle |x-1\rangle , \end{cases} \tag{13}$$

that can also be described by

$$S = |0\rangle \langle 0| \otimes \sum_{x=-\infty}^{x=\infty} |x+1\rangle \langle x| + |1\rangle \langle 1| \otimes \sum_{x=-\infty}^{x=\infty} |x-1\rangle \langle x| . \tag{14}$$

It follows that the operator that describes the dynamics of the quantum walk will be given by

$$U = S(C \otimes I) = S(H \otimes I). \tag{15}$$

Consider a quantum system located at $|x=0\rangle$ with coin state $|0\rangle$, for $t=0$. It's state will be described by

$$|\Psi(0)\rangle = |0\rangle |x=0\rangle . \tag{16}$$

After t steps

$$|\Psi(t)\rangle = U^t |\Psi(0)\rangle , \tag{17}$$

more explicitly

$$|\Psi(0)\rangle \xrightarrow{U} |\Psi(1)\rangle \xrightarrow{U} |\Psi(2)\rangle \xrightarrow{U} (...) \xrightarrow{U} |\Psi(t)\rangle . \tag{18}$$

In other words, the coined quantum walk algorithm consists on applying the coin operator followed by the shift operator a certain number of times. Iterating this twice, evolves the system to the following respective states

$$|\Psi(1)\rangle = \frac{|0\rangle |x=-1\rangle + |1\rangle |x=1\rangle}{\sqrt{2}} \tag{19}$$

$$|\Psi(2)\rangle = \frac{|0\rangle |x=-2\rangle + |1\rangle |x=0\rangle + |0\rangle |x=0\rangle - |1\rangle |x=2\rangle}{2} \tag{20}$$

$$\tag{21}$$

If one were to measure the system after the first application of $\mathcal{U}$, it would be expected to see the walker at $x = 1$ with probability $P(x) = \frac{1}{2}$, and at $x = -1$ with $P(x) = \frac{1}{2}$ aswell. Measure the system t times, after each application of $\mathcal{U}$, and the result is a binomial probability distribution similar to the one in 2. The conclusion is that repetitive measurement of a coined quantum walk system reduces to the classical case, which means that any desired quantum behaviour is lost.

It is possible, however, to make use of the quantum correlations between different positions to generate constructive or destructive interference, by applying the Hadamard and shift operators successively without intermediary measurements. The consequences of interference between states become very apparent after only 3 iterations

$$|\Psi(3)\rangle = \frac{|1\rangle\,|x=-3\rangle - |0\rangle\,|x=-1\rangle + 2(|0\rangle + |1\rangle)\,|x=1\rangle + |0\rangle\,|x=3\rangle}{2\sqrt{2}}. \tag{22}$$

Even though an unbiased coin was used, this state is not symmetric around the origin and the probability distributions will not be centered in the origin. Moreover, Portugal (2018) shows that the standard deviation will be

$$\sigma(t) \approx 0.54t. \tag{23}$$

This means that the standard deviation for the coined quantum walk grows linearly in time, unlike the classical case which grows with $\sqrt{t}$, as was seen in equation 10. The implication is that the quantum walk displays *ballistic* behaviour, as is reviewed in Venegas-Andraca (2012). This behaviour is usually defined in the context of a moving free particle with unit velocity in a single direction, which is expected to be found at $x = t$ after t steps. The velocity of a walker in a Hadamard quantum walk is approximately half of the free particle example, which is still a quadratic improvement over the classical random walk.

This quadratic gain implies exponentially faster hitting times in certain graphs, as shown by Childs et al. (2002), meaning improvements to problems that require transversing graphs. Ambainis (2007) also shows advantages of the coined quantum walk model in element distinctness problems, and Childs and Goldstone (2004) show advantages in spatial search problems, which will be studied in a later chapter.

In order to study this distribution, a simulation of the coined quantum walk was coded in *Python*. Figure 3 is the result of using the Hadamard coin and the initial condition in equation 16, for varying numbers of steps. Analyzing the plot, it is noticeable that the distributions are asymmetric. The probability of finding the walker on the right-hand side is much larger than on the left, with a peak around $x \approx \frac{t}{\sqrt{2}}$. Regardless of number of steps, this peak is always present (albeit in varying positions), which is to say that the walker can always be found moving in a uniform fashion away from the origin, consistent with ballistic behaviour.

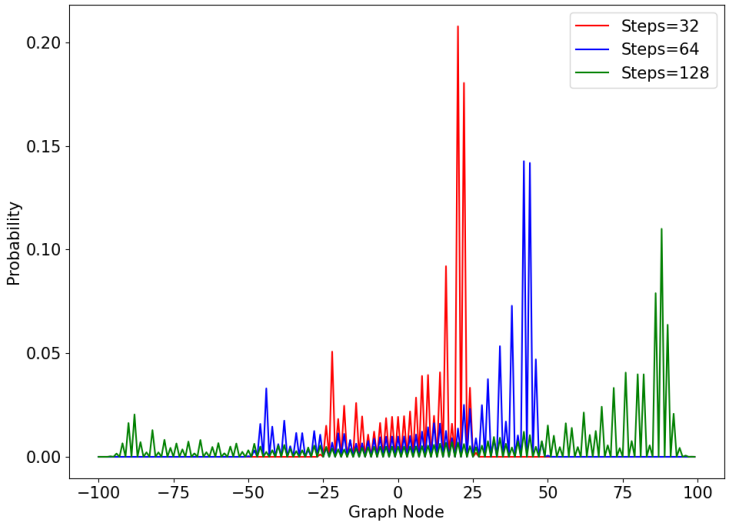


Figure 3: Probability distribution for the coined quantum walk on a line, after 100 steps, with initial condition $|\Psi(0)\rangle = |0\rangle\,|x = 0\rangle$ and the Hadamard coin.

Another interesting case study is to find if this behaviour is preserved for a symmetric distribution around the origin. For this purpose, one must first understand where the asymmetry comes from. The Hadamard operator flips the sign of state $|1\rangle$, hence more

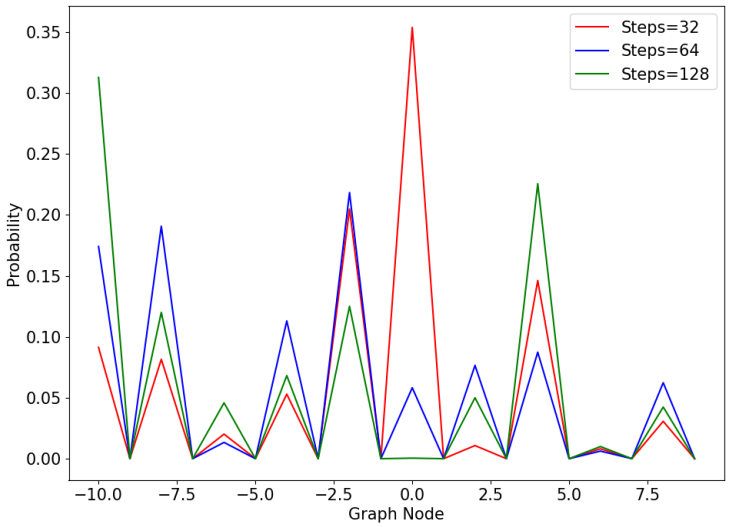


Figure 4: Probability distribution for the coined quantum walk on a line, after 100 steps, with initial condition $|\Psi(0)\rangle = -|1\rangle\,|x = 0\rangle$ and the Hadamard coin.

terms are cancelled when the coin state is $|1\rangle$. Since $|0\rangle$ was defined to induce movement

to the right, the result is as shown in 3. Following this logic, it would be expected that an initial condition

$$|\Psi(0)\rangle = |1\rangle\,|x=0\rangle\,, \tag{24}$$

would result in more cancellations when the coin state is $|0\rangle$, thus the walker would be more likely found in the left-hand side of the graph. This is indeed what happens, as figure 4 is a mirror image of figure 3. The walker still moves away from the origin with ballistic behaviour, but in opposite direction. The peaks behave in a similar fashion, being instead found at $x \approx -\frac{t}{\sqrt{2}}$.

In order to obtain a symmetrical distribution, one must superpose the state in equation 16 with the state in equation 24. However, in order to not cancel terms before the calculation of the probability distribution, one must multiply state $|1\rangle$ with the imaginary unit, i

$$|\Psi(0)\rangle = \frac{|0\rangle + i\,|1\rangle}{\sqrt{2}}\,|x=0\rangle\,. \tag{25}$$

This works because the entries of the Hadamard operator are real numbers. Terms with the imaginary unit will not cancel out with terms without it, thus the walk can proceed to both left and right, as it is shown in figure 5.

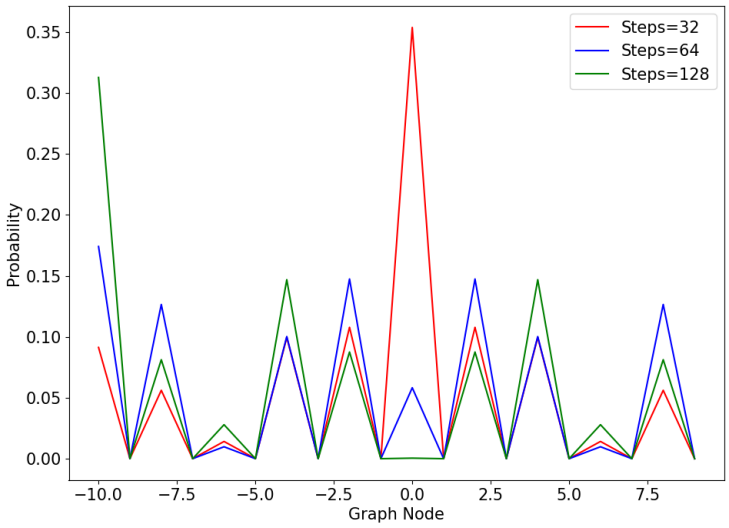


Figure 5: Probability distribution for the coined quantum walk on a line, after 100 steps, with initial condition $|\Psi(0)\rangle = \frac{|0\rangle - i|1\rangle}{\sqrt{2}}\,|x=0\rangle$ and the Hadamard coin.

The probability distribution is now symmetric and it is spread over the range $[-\frac{t}{\sqrt{2}}, -\frac{t}{\sqrt{2}}]$ with peaks around $x \approx \pm\frac{t}{\sqrt{2}}$. This means that if the position of the walker was measured at

the end, it would be equally probable to find him either in the left side or the right side of the graph , which is not possible in a classical diffusive motion.

All of the previous examples are in sharp contrast with the classical random walk distribution in figure 2. There, maximum probability is reached at $x = 0$ since there are approximately equal steps in both directions. Furthermore, the further the vertex is away from the origin, the less likely the walker is to be found there. However, in the quantum case, the walker is more likely to be found away from the origin as the number of steps increases. More specifically, the walk spreads quadratically faster than the classical counterpart.

This is but one model of a quantum random walk. As it will be seen in further sections, there are other approaches to creating both discrete and continuous quantum walk models that do not use a coin.

3.3 CONTINUOUS-TIME QUANTUM WALK

The continuous-time random walk model on a graph is a Markov process where transitions have a fixed probability per unit time γ of moving to adjacent vertices, firstly introduced by Montroll and Weiss (1997). Consider a graph G with N vertices and no self-loops, this walk can be defined by the linear differential equation that describes the probability of jumping to a connected vertex in any given time

$$\frac{dp_i(t)}{dt} = \gamma \sum_j L_{ij} p_j(t), \tag{26}$$

where L is the Laplacian defined as $L = A - D$. A is the adjacency matrix that represents each vertex connection, given by

$$A_{ij} = \begin{cases} 1, & \text{if } (i,j) \in G \\ 0, & \text{otherwise,} \end{cases} \tag{27}$$

and D is the diagonal matrix $D_{jj} = deg(j)$ corresponding to the degree[2] of a vertex j.

In the quantum case, the nodes are quantum states that form the basis for the Hilbert space. The continuous-time quantum walk model will also be described by a differential equation, the Schrödinger equation

$$i\hbar \frac{d\,|\Psi(t)\rangle}{dt} = \hat{H}\,|\Psi(t)\rangle\,, \tag{28}$$

2 The degree of a vertex refers to the number of edges that it is connected to.

where $\hat{H} = -\gamma L$ is the Hamiltonian of the system. More explicitly,

$$\hat{H}_{ij} = \begin{cases} deg(j)\gamma, & \text{if } i = j; \\ -\gamma, & \text{if } i \neq j \text{ and adjacent;} \\ 0, & \text{if } i \neq j \text{ and not adjacent.} \end{cases} . \tag{29}$$

A general state of a system $|\Psi(t)\rangle$ can be written as a function of it's complex amplitudes

$$q_i = \langle i|\Psi(t)|i|\Psi(t)\rangle\,, \tag{30}$$

which means 28 can be rewritten as

$$i\hbar\frac{dq_i(t)}{dt} = \sum_j \hat{H}_{ij}q_j(t). \tag{31}$$

This highlights the similarities between the Schrödinger equation and 26. One of the main differences is the complex phase i, which will result in a very different behaviour. Setting $\hbar = 1$ and solving the differential equation results in the evolution operator of this walk

$$U(t) = e^{-iHt} = e^{i(-\gamma L)t} = e^{-i\gamma(A+D)t} \tag{32}$$

In the regular graph case, where D is simply the degree of the whole graph multiplied by the identity matrix, A and D will commute, meaning that the evolution operator can be written in terms of the adjacency matrix

$$U(t) = e^{-i\gamma At+i\gamma Dt} = e^{-i\gamma At}e^{i\gamma Dt} = \phi(t)e^{-i\gamma At} \tag{33}$$

since the degree matrix becomes a global phase. Applying this operator to an initial condition $\Psi(0)$, will give the state of the system at a time t

$$|\Psi(t)\rangle = U(t)\,|\Psi(0)\rangle\,. \tag{34}$$

Considering a uni-dimensional quantum system, each vertex will have at most 2 other neighboring vertices, reducing equation 29 to

$$\hat{H}_{ij} = \begin{cases} 2\gamma, & \text{if } i = j; \\ -\gamma, & \text{if } i \neq j \text{ and adjacent;} \\ 0, & \text{if } i \neq j \text{ and not adjacent.} \end{cases} \tag{35}$$

For a more detailed visualization, this quantum walk model was coded in python and figure 6 was obtained setting the transition rate to $\gamma = \frac{1}{2\sqrt{2}}$ the initial condition to $|\Psi(0)\rangle = |0\rangle$

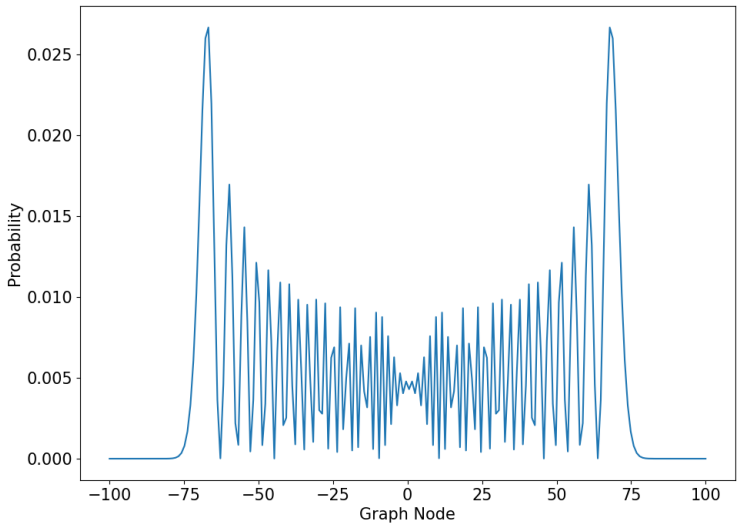


Figure 6: Probability distribution for the continuous-time quantum walk on a line, at t = 100, with initial condition $|\Psi(0)\rangle = |0\rangle$ and $\gamma = \frac{1}{2\sqrt{2}}$.

A brief look at figure 6 reveals several similarities to the coined quantum walk model of figure **??**. Both have two peaks away from the origin and low probability near the origin. However, in the previous quantum walk, these characteristics were altered as a function of the chosen coin and initial condition, whereas in this case different values of γ will influence the probability distribution. For example, a lower value of γ will limit the spread of the probability distribution, as is shown in figure 8.

Moreover, the effects of altering the initial condition will also differ in the continuous-time example. For example, setting the initial condition to the balanced superposition of states $|0\rangle$ and $|1\rangle$ has no effect on the overall pattern of the probability distribution as can be seen in figure 10. Both peaks still are still present and at the same distance from the origin, with intermediate amplitudes being attenuated relative to figure 6. This behaviour is in contrast with the discrete-time case, where a change in the initial condition would dictate the number of peaks and where they would appear.

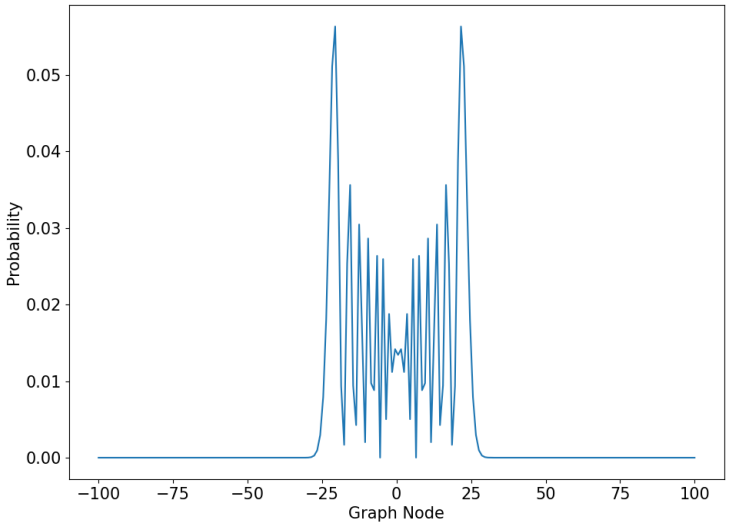


Figure 7: Probability distribution for the continuous-time quantum walk on a line, after 100 steps, with initial condition $|\Psi(0)\rangle = |0\rangle$ and $\gamma = \frac{1}{6\sqrt{2}}$.

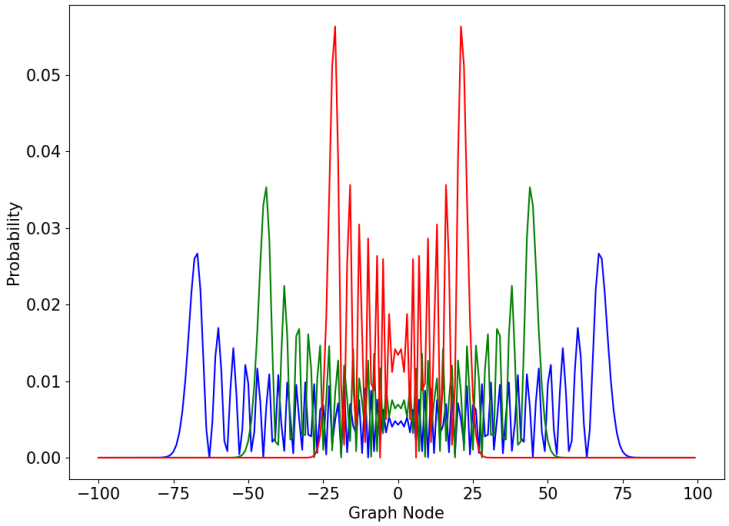


Figure 8: Temporary

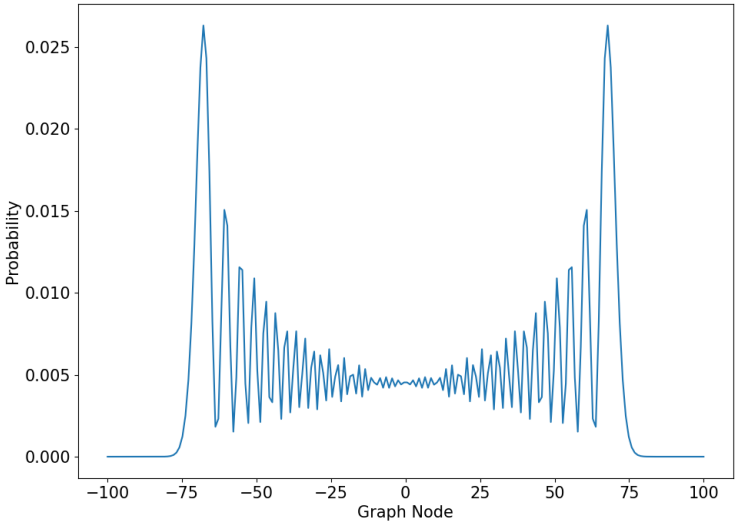


Figure 9: Probability distribution for the continuous-time quantum walk on a line, after 100 steps, with initial condition $|\Psi(0)\rangle = \frac{|0\rangle+|1\rangle}{\sqrt{2}}$ and $\gamma = \frac{1}{2\sqrt{2}}$.

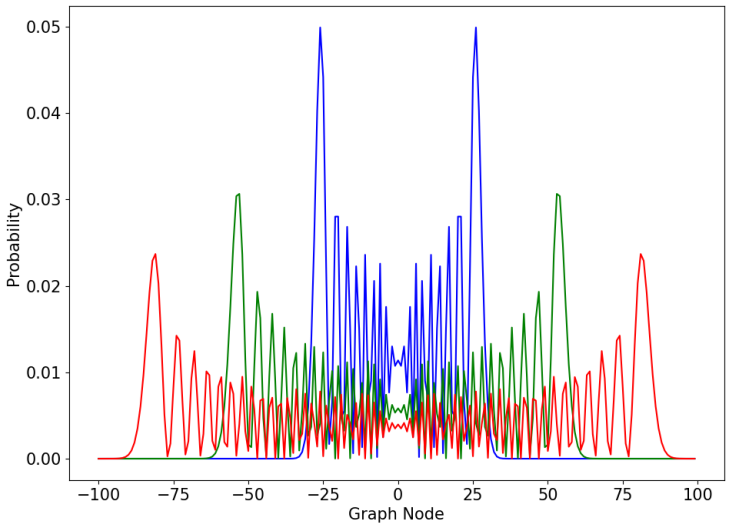


Figure 10: Temporary

3.4 STAGGERED QUANTUM WALK

Similarly to the continuous-time quantum walk, the staggered case aims to spread a transition probability to neighboring vertices but with discrete time steps. The notion of adjacency now comes from cliques[3], and the initial stage of this walk consists in partitioning the graph in several different cliques. This is called tessellation, and it is defined as the division of the set of vertices into disjoint cliques. An element of a tessellation $\mathscr{T}$ is called a polygon, and it's only valid if if all of its vertices belong to the clique in $\mathscr{T}$. The set of polygons of each tessellation must cover all vertices of the graph, and the set of tessellations ($\mathscr{T}_1$,$\mathscr{T}_2$,...,$\mathscr{T}_k$) must cover all edges.

These definitions allow the construction of operators H_1,H_2,...,H_k that will be used to propagate the probability amplitude locally, in each polygon. The state associated to each polygon is

$$\left|u_j^k\right\rangle = \frac{1}{\sqrt{|\alpha_j^k|}} \sum_{l \in \alpha_j^k} |l\rangle \tag{36}$$

where α_j^k is the $j-th$ polygon in the $k-th$ tessellation.

The unitary and Hermitian operator H_k, associated to each tessellation is defined in Portugal et al. (2017) as

$$H_k = 2 \sum_{j=1}^{p} \left|u_j^k\right\rangle \left\langle u_j^k\right| - I \tag{37}$$

Solving the time-independent Schrodinger equation for this Hamiltonian gives the evolution operator

$$U = e^{i\theta_k H_k} ... e^{i\theta_2 H_2} e^{i\theta_1 H_1} \tag{38}$$

where

$$e^{i\theta_k H_k} = \cos(\theta_k) I + i \sin(\theta_k) H_k \tag{39}$$

since $H_k^2 = I$.

The simplest use case of this quantum walk model is the one-dimensional lattice, where the minimum tessalations are

$$\mathscr{T}_\alpha = \{\{2x, 2x+1\} : x \in \mathbb{Z}\} \tag{40}$$

$$\mathscr{T}_\beta = \{\{2x+1, 2x+2\} : x \in \mathbb{Z}\} \tag{41}$$

3 A clique is defined as the subset of vertices of an undirected graph such that every two distinct vertices in each clique are adjacent.

Each element of the tessellation has a corresponding state, and the uniform superposition of these states is

$$|\alpha_x\rangle = \frac{|2x\rangle + |2x+1\rangle}{\sqrt{2}} \tag{42}$$

$$|\beta_x\rangle = \frac{|2x+1\rangle + |2x+2\rangle}{\sqrt{2}} \tag{43}$$

One can now define Hamiltonians H_α and H_β as

$$H_\alpha = 2\sum_{x=-\infty}^{+\infty} |\alpha_x\rangle \langle\alpha_x| - I \tag{44}$$

$$H_\beta = 2\sum_{x=-\infty}^{+\infty} |\beta_x\rangle \langle\beta_x| - I \tag{45}$$

The Hamiltonian evolution operator reduces to

$$U = e^{i\theta H_\beta} e^{i\theta H_\alpha} \tag{46}$$

and applying it to an initial condition $|\Psi(0)\rangle$ results in the time evolution operator

$$U|\Psi(t)\rangle = U^t |\Psi(0)\rangle \tag{47}$$

Having defined the time evolution operator, the walk is ready to be coded with a certain initial condition and θ value, to better understand how the probability distribution spreads through time.

For the first case study, the initial condition will be a uniform superposition of states $|0\rangle$ and $|1\rangle$ and the θ value will be varied in order to understand how this parameter impacts the walk

The overall structure of the probability distribution remains the same, the difference is that the walker is more likely to be found further away from the origin as the angle increases.

Another interesting case study is to see how the initial condition affects the dynamics of the system

Similarly to the coined case, each initial condition results in asymmetric probability distributions, $|\Psi(0)\rangle = |0\rangle$ leads to a peak in the left-hand side while condition $|\Psi(0)\rangle = |1\rangle$ results in a peak in the right-hand side. As was shown in 11, the uniform superposition of both these conditions results in a symmetric probability distribution.

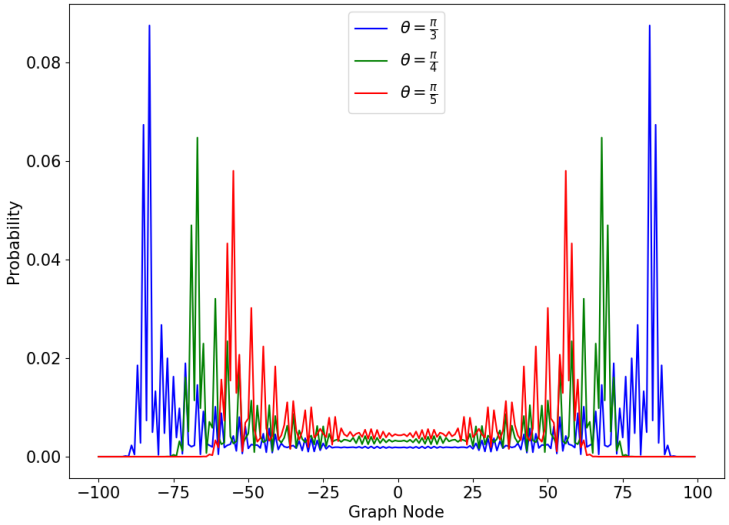


Figure 11: Probability distribution for the staggered quantum walk on a line after 50 steps, with initial condition $|\Psi(0)\rangle = \frac{|0\rangle+|1\rangle}{\sqrt{2}}$, for multiple angles.

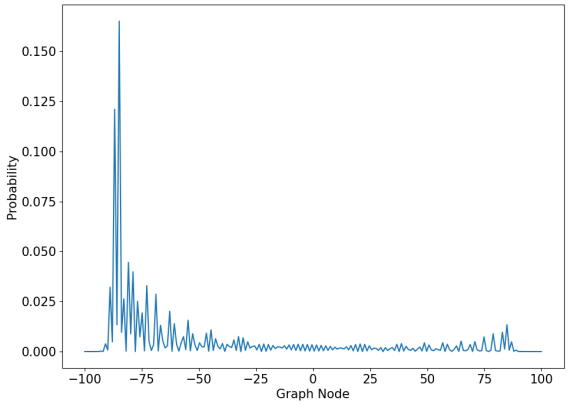


Figure 12: $|\Psi(0)\rangle = |0\rangle$

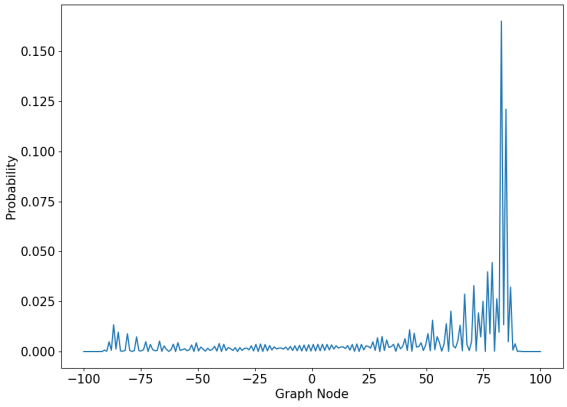


Figure 13: $|\Psi(0)\rangle = |1\rangle$

4

SEARCHING PROBLEMS

4.1 GROVER'S ALGORITHM

Searching through an unstructured database is a task classically achieved by exhaustively evaluating every element in the database. Assume there exists a black box (oracle) that can be asked to find out if two elements are equal. Since we're looking for a specific element in a database of size N, we'd have to query the oracle on average $\frac{N}{2}$ times, or in the worst case N times.

Grover's algorithm, presented in Grover (1996), comes as a quantum alternative to this type of problems, taking advantage of superposition by increasing desirable states' amplitudes through a process called *amplitude amplification*. This method has a quadratic gain over the classical counterpart Boyer et al. (1998), being able to find a target element in expected time $\mathcal{O}(\sqrt{N})$.

Let us now expand on the inner workings of the black box. We start by focusing on searching indexes instead of directly evaluating the element and we assume $N = 2^n$, n being a positive integer. We can now define a function $f : \{0, 1, ..., N-1\}$ that returns 1 when evaluating the desired (marked) element and 0 otherwise. Since this function is to be applied to a quantum system, we must build a unitary operator $\mathcal{O}$

$$\mathcal{O} |x\rangle |i\rangle = |x\rangle |i \oplus f(x)\rangle . \tag{48}$$

where $|x\rangle$ is the index register, $\oplus$ is the binary sum operation and $|i\rangle$ is a qubit that is flipped if $f(x) = 1$.

The action of the oracle on state $|0\rangle$ will be

$$\mathcal{O} |x\rangle |0\rangle = \begin{cases} |x_0\rangle |1\rangle , & \text{if } x = x_0 \\ |x\rangle |0\rangle , & \text{otherwise.} \end{cases} \tag{49}$$

where x_0 is the marked element. More generically, $\mathcal{O}$ can be written as

$$\mathcal{O}|x\rangle = (-1)^{f(x)}|x\rangle . \tag{50}$$

This offers a bit of insight into the oracle, it *marks* the solutions to the search problem by applying a phase shift to the solutions. The question now is, what is the procedure that determines a solution x_0 using $\mathcal{O}$ the minimum number of times? The answer lies in the amplitude amplification section of Grover's search, starting with the creation of a uniform superposition

$$|\Psi_0\rangle = H^{\otimes n}|x\rangle = \frac{1}{\sqrt{N}}\sum_{x=0}^{N-1}|x\rangle \tag{51}$$

where $H^{\otimes n}$ is the *Hadamard* operator applied to an arbitrary number of states.

If we were to measure $|x\rangle$ at this point, the superposition would collapse to any of the base states with the same probability $\frac{1}{N} = \frac{1}{2^n}$, which means that on average, we'd need to try $N = 2^n$ times to guess the correct item. This is where amplitude amplification comes into effect, by means of a second unitary operator

$$\mathcal{D} = (2|\Psi_0\rangle\langle\Psi_0| - I) = H^{\otimes n}(2|0\rangle\langle 0| - I)H^{\otimes n} \tag{52}$$

This operator applies a conditional phase shift, with every computational basis state except $|0\rangle$ receiving a phase shift. This can also be described as the *inversion about the mean*, for a state of arbitrary amplitudes

$$|\phi\rangle = \sum_{k=0}^{N-1}\alpha_k|k\rangle \tag{53}$$

the action of $\mathcal{D}$ on state ϕ will be

$$\mathcal{D}|\phi\rangle = \sum_{k=0}^{N-1}(-\alpha_k + 2\langle\alpha\rangle)|k\rangle \tag{54}$$

where $\langle\alpha\rangle$ is the average of α_k

$$\langle\alpha\rangle = \frac{1}{N}\sum_{k=0}^{N-1}\alpha_k|k\rangle \tag{55}$$

The evolution operator that performs one step of the algorithm is then

$$\mathcal{U} = \mathcal{D}\mathcal{O} \tag{56}$$

and after t steps the state of the system is

$$|\Psi(t)\rangle = \mathcal{U}^t|\Psi_0\rangle . \tag{57}$$

4.1.1 *One marked element*

The optimal number of steps is, as aforementioned, proportional to $\sqrt{N}$. More precisely, if there's only one solution, maximum probability can be reached in *approximately* $\frac{\pi}{4}\sqrt{N}$ iterations. In order to show that this is the case, an iteration will be formally defined here as the process that transforms the state

$$|\Psi(k,l)\rangle = k\,|i_0\rangle + \sum_{i\neq i_0} l\,|i\rangle \tag{58}$$

into state $\left|\Psi(\frac{N-2}{N}k + \frac{2(N-1)}{N}l, \frac{N-2}{N}l - \frac{2}{N}k)\right\rangle$. Amplitudes l and k are real numbers that statisfy $k^2 + (N-1)l^2 = 1$. Running m iterations over state $|\Psi_0\rangle$ will eventually lead to state $|\Psi_j\rangle = |\Psi(k_j, l_j)\rangle$ after the $j-th$ iteration, where $k_0 = l_0 = \frac{1}{\sqrt{N}}$ and

$$\begin{cases} k_{j+1} = \frac{N-2}{N}k_j + \frac{2(N-1)}{N}l_j \\ l_{j+1} = \frac{N-2}{N}l_j + \frac{2}{N}k_j. \end{cases} \tag{59}$$

After the last iteration, the system will be in state $|\Psi_m\rangle$ with a certain amplitude. If that amplitude corresponds to the marked element x_0, then it is said that the algorithm was successful.

Grover (1996) proves that there exists a value of $m < \sqrt{2N}$ such that the probability of success is at least $\frac{1}{2}$. However the probability of success does not linearly increase with the number of iterations, in fact for $m = \sqrt{2N}$ the system will succeed less that 1 in 10 times. Boyer et al. (1998) argues that an explicit value of m is needed, and it is achieved by finding a closed form formula for k_j and l_j. The first step is to define an angle θ so that $\sin^2\theta = \frac{1}{N}$, and equation 59 will become

$$\begin{cases} k_{j+1} = \sin\left((2j+1)\theta\right) \\ l_{j+1} = \frac{1}{\sqrt{N-1}}\cos\left((2j+1)\theta\right). \end{cases} \tag{60}$$

In order to maximize the probability of success, one must find a value of m so that $k_m = 1$ and l_m is as close to 0 as possible. The value of k after m iterations will be at it's maximum when $\sin\left((2m+1)\theta\right) = \frac{\pi}{2}$, and solving the trigonometric equation leads to a value of $m = \frac{\pi-2\theta}{4\theta}$. Conversely, $l_{\widetilde{m}} = 0$ when $\widetilde{m} = \frac{\pi-2\theta}{4\theta}$ for an integer number of $\widetilde{m}$. Setting m to the nearest lower integer of $\frac{\pi}{4\theta}$ will lead to

$$|m - \widetilde{m}| \leq \frac{1}{2} \iff |(2m+1)\theta - (2\widetilde{m}-1)\theta| \leq \frac{\pi}{2}. \tag{61}$$

By definition, $(2\widetilde{m}+1)\theta = \frac{\pi}{2}$ which means that $|\cos((2m+1)\theta)| \leq |\sin\theta|$. The probability of failure after m iterations can then be written as

$$(N-1)l_m^2 = \cos^2((2m+1)\theta) \leq \sin^2\theta = \frac{1}{N}. \tag{62}$$

Failure decreases as the number of elements increases. The run time of the algorithm will be

$$m \leq \frac{\pi}{4\theta} \leq \frac{\pi}{4}\sqrt{N} \tag{63}$$

since $\theta \geq \sin\theta = \frac{1}{\sqrt{N}}$. This means that, for a large N, the number of iterations that maximizes the probability of success will be very close to $\frac{\pi}{4}\sqrt{N}$.

Figure 14 was obtained by coding the apropriate operators as to simulate the system presented in equation 57. The unitary evolution operator was applied approximately

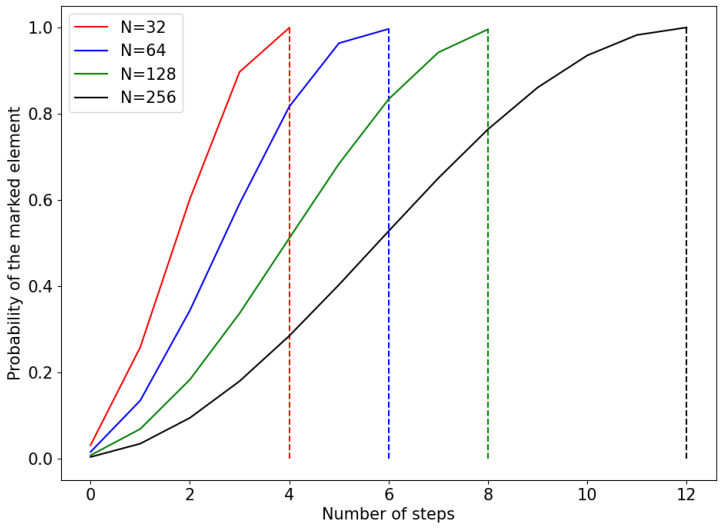


Figure 14: Grover one marked element temp.

$\frac{\pi}{4}\sqrt{N}$ times and the amplitudes associated with those states were stored as a probability distribution. Filtering the probability of the marked element and plotting it against the number of steps, shows that the maximum is indeed reached after the said number of iterations. It also shows that the maximum probability for $N = 16$ is lower than for $N = 128$, which makes sense since the the probability of success is maximized for larger values of N.

4.1.2 *Multiple marked elements*

When there's more than one element marked by the oracle, the number of iterations to achieve maximum probability changes. In fact, the latter part of this section will be used to discuss the case where one single iteration of this algorithm is enough to achieve maximum probability.

Firstly, one must define a set A that is composed of all the marked elements and set B of the remaining. The state from equation 58 will become

$$|\Psi(k,l)\rangle = \sum_{i\in A} k\,|i\rangle + \sum_{x\in B} l\,|x\rangle\,. \tag{64}$$

Assuming t marked elements, iterating over this state will result in

$$\left|\Psi\left(\frac{N-2t}{N}k + \frac{2(N-1)}{N}l, \frac{N-2}{N}l - \frac{2}{N}k\right)\right\rangle. \tag{65}$$

Choosing an angle θ such that $sin^2\theta = \frac{t}{N}$, allows the definition of the amplitudes associated with the states after j iterations

$$\begin{cases} k_j = \frac{1}{\sqrt{t}}\sin\left((2j+1)\theta\right) \\ l_j = \frac{1}{\sqrt{N-t}}\cos\left((2j+1)\theta\right). \end{cases} \tag{66}$$

Similarly to the one solution case, it can be shown that setting the number of iterations

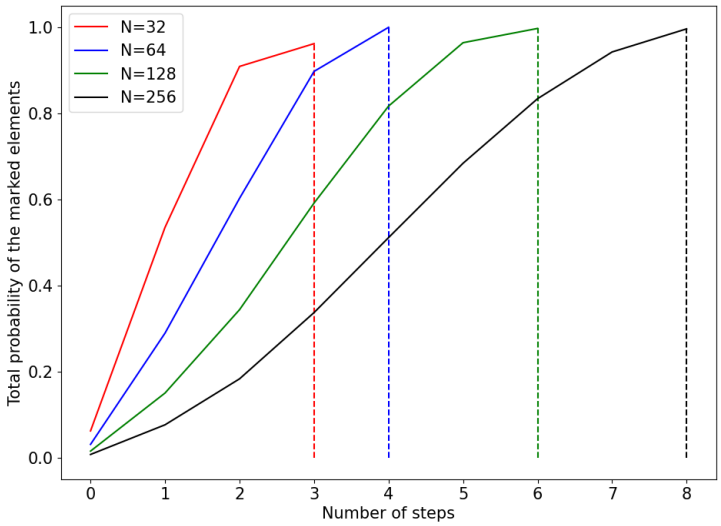


Figure 15: Grover Multiple marked temp.

m, to the nearest lower integer of $\frac{\pi}{4\theta}$ will result in a probability of failure $(N-t)l_m^2 \leq \frac{t}{N}$. Because $\theta \geq \sin\theta = \sqrt{\frac{t}{N}}$ then

$$m \leq \frac{\pi}{4\theta} \leq \frac{\pi}{4}\sqrt{\frac{N}{t}}. \tag{67}$$

From a more practical perspective, if one were to mark two elements of a 64 element set, maximum probability is expected to be reached in approximately 4 steps, since $\lfloor \frac{\pi}{4}\sqrt{\frac{64}{2}} \rfloor = 4$. Likewise, for $N = 256$, the number of iterations is rounded to 8, which is plotted along several other values of N in figure 15. The y-axis is now the sum total probability of the marked elements and the x-axis represents the range of steps that spans from 0 to $\lfloor \frac{\pi}{4}\sqrt{\frac{N}{2}} \rfloor$ for each N. Again, the probability of success approaches 1 as N increases. However, comparing to figure 14, the number of iterations that maximize probability is lower for each N, in agreement with equation 67.

4.1.3 *Single-Shot Grover*

An interesting case arises when the number of marked elements is set to $t = \frac{N}{4}$, because

$$sin^2\theta = \frac{\frac{N}{4}}{N} = \frac{1}{4} \iff \sin\theta = \frac{1}{2} \iff \theta = \frac{\pi}{6}. \tag{68}$$

Note that there are an infinite number of negative and positive solutions, but equation 68

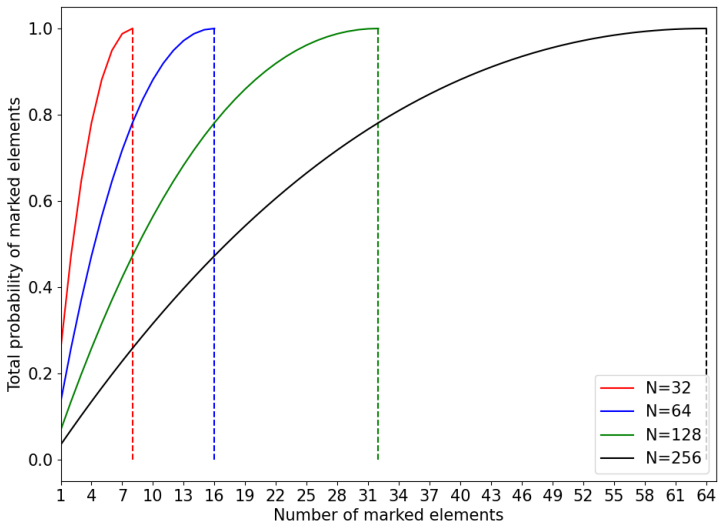


Figure 16: Single Shot temp

reflects the only relevant one in this context. As a consequence, amplitudes associated with state $|\Psi(k_1, l_1)\rangle$ become

$$\begin{cases} k_1 = \frac{1}{\sqrt{t}} \sin\left((2+1)\theta\right) = \frac{1}{\sqrt{\frac{N}{4}}} \sin\left(3\frac{\pi}{6}\right) = \frac{2}{\sqrt{N}} \\ l_1 = \frac{1}{\sqrt{N-t}} \cos\left((2+1)\theta\right) = \frac{1}{\sqrt{N-\frac{N}{4}}} \cos\left(3\frac{\pi}{6}\right) = 0. \end{cases} \tag{69}$$

These results show that the amplitudes associated with the marked states double in relation to $|\Psi_0\rangle$ and the remaining states disappear after only one iteration. This behaviour can be seen in figure 16, where the total probability of marked elements reaches 1 once the number of marked elements is $\frac{1}{4}$ of the total elements.

4.2 COINED QUANTUM WALK

In classical computation, a *spatial search problem* focuses on finding marked points in a finite region of space. Defining this region with graphs is fairly straightforward, the vertices of the graph are the search space, and the edges define what transitions are possible through the search space. As was previously mentioned in **??**, exhaustively searching through an unstructured space, by means of a classical random walk for example, would mean that in the worst case, one would have to take as many steps to find the marked points as there are vertices in the graph. Quantum computing provides an alternative to this complexity through Grover's algorithm, and applying some of his ideas to the coined quantum walk not only allows a quantum counterpart to the random walk search, but also further insight into the algorithm itself.

Following Portugal (2018)'s definition, a good first step is to borrow the diffusion from Grover's algorithm and invert the sign of the state corresponding to the marked vertex while leaving unmarked vertices unchanged. This is done through the following operator

$$\mathcal{O} = I - 2 \sum_{x \in M} |x\rangle \langle x| \tag{70}$$

where M is the set of marked vertices and $\mathcal{O}$ is an analogue to Grover's oracle. For one marked vertex, this oracle can be written as

$$\mathcal{O} = I - 2 |0\rangle \langle 0| \tag{71}$$

Notice that there is no loss of generality by choosing the marked vertex as 0, since the labelling of the vertices is arbitrary.

The next step is to combine the evolution operator from the coined quantum walk model with the oracle

$$U' = U\mathcal{O} \tag{72}$$

Similarly to the simple coined case, the walker starts at $|\Psi(0)\rangle$ and evolves following the rules of an unitary operator U followed by the sign inversion of marked vertices. The walker's state after an arbitrary number of steps will be

$$\Psi(t) = (U')^t \, |\Psi(0)\rangle \, . \tag{73}$$

For a better understanding of the search problem in the coined quantum walk model, consider a graph where all the vertices are connected and each vertex has a loop that allows transitions to itself, as shown in figure **??**. The next step is to label the arcs using notation $\{(v, v'), v \geqslant 0 \wedge v' \leqslant N-1\}$ where N is the total number of vertices and (v, v') are the position and coin value, respectively, in the coined model. The shift operator, now called *flip-flop* shift operator, is

$$S \, |v1\rangle \, |v2\rangle = |v2\rangle \, |v1\rangle \, . \tag{74}$$

The coin operator is defined as

$$C = I_N \otimes G \tag{75}$$

where

$$G = 2 \, |D\rangle \, \langle D| - I \tag{76}$$

is the Grover coin with $|D\rangle$ being the diagonal state of the coin space. Given both of these operators, the evolution is defined for the unmarked case similarly to **??**

$$U = S(I \otimes G). \tag{77}$$

Marking an element in a complete graph is done through the following oracle

$$\mathcal{O}' = \mathcal{O} \otimes I = (I_N - 2 \, |0\rangle \, \langle 0|) \otimes I_N = I_{N^2} - 2 \sum_v |0\rangle \, |v\rangle \, \langle 0| \, \langle v| \, , \tag{78}$$

that can be seen, in the arc notation, as an operator that marks all arcs leaving 0.

Recalling 72, the modified evolution operator can be written as

$$U' = S(I \otimes G)\mathcal{O}' = S(I \otimes G)\mathcal{O} \otimes I = S(\mathcal{O} \otimes G), \tag{79}$$

and the state of the system will evolve according to equation 73.

As was shown in Portugal (2018), maximum probability of the marked vertex is achieved after $\frac{\pi}{2}\sqrt{N}$ steps. Figure 17 is the result of coding and plotting the evolution of this probability distribution, for graphs of varying sizes. It shows that the probability is close

to one at *approximately* the predicted ideal steps, because of the discrete nature of the walk. The probability distributions have a stair-like shape, because transitions in this model only occur on even numbered time steps, because of how the unmodified evolution operator was constructed.

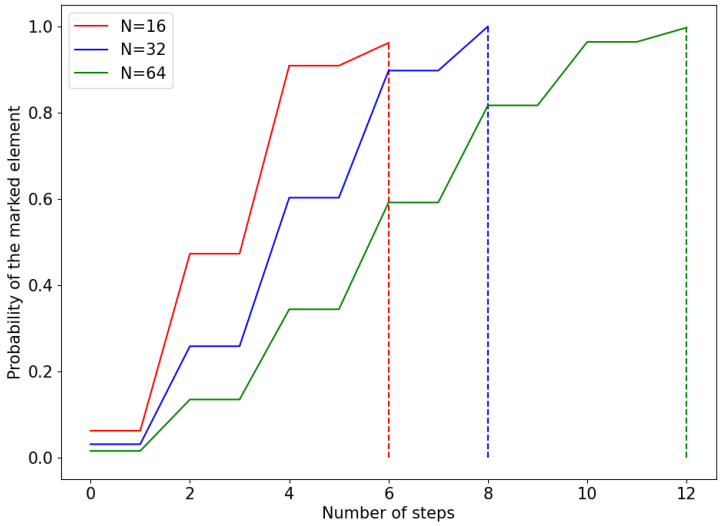


Figure 17: Discrete-time coined quantum walk search for a complete graph with 16, 32 and 64 nodes.

4.3 CONTINUOUS-TIME QUANTUM WALK

As was previously seen, the continuous-time quantum walk model is defined by an evolution operator obtained by solving Schrödinger's equation

$$U(t) = e^{-iHt}. \tag{80}$$

The search problem requires introducing an oracle to the Hamiltonian, that will mark an arbitrary vertex m

$$H' = -\gamma L - |m\rangle \langle m| . \tag{81}$$

Since the complete graph is a regular graph, the operator can be rewritten in terms of the adjacency matrix plus the marked element. Considering $|0\rangle$ is marked,

$$U'(t) = e^{iH't} = e^{i(-\gamma L - |0\rangle\langle 0|)t} = e^{i(-\gamma A + \gamma D - |0\rangle\langle 0|)t} = e^{-i\gamma(A + |0\rangle\langle 0|)t + i\gamma Dt}. \tag{82}$$

The degree matrix is again $D = dI$, which means it will commute with $A + |0\rangle\langle 0|$ and become a global phase

$$U'(t) = e^{-i\gamma(A+|0\rangle\langle 0|)t}e^{i\gamma Dt} = \phi(t)e^{-i\gamma(A+|0\rangle\langle 0|)t}. \tag{83}$$

As was show by Zalka (1999), the value of γ is crucial for the success of the search. As γ increases, the contribution of the marked element in the Hamiltonian decreases, and as γ approaches 0 the contribution of the adjacency matrix decreases. To find the optimum value, the Hamiltonian can be rewritten by adding multiples of the identity matrix to the adjacency matrix

$$H' = -\gamma(A + NI) - |0\rangle\langle 0| = -\gamma N |s\rangle\langle s| - |0\rangle\langle 0| \tag{84}$$

where $|s\rangle = \frac{1}{\sqrt{N}}\sum_i |i\rangle$. Now it is obvious that, for $\gamma = \frac{1}{N}$, the Hamiltonian is $H = -|s\rangle\langle s| - |0\rangle\langle 0|$. It's eigenstates are proportional to $|s\rangle \pm |w\rangle$ and eigenvalues are $-1 - \frac{1}{\sqrt{N}}$ and $-1 + \frac{1}{\sqrt{N}}$, respectively. This means that the evolution rotates from the state of balanced superposition to the marked vertex state in time $\frac{\pi}{\Delta E} = \frac{\pi}{2}\sqrt{N}$ which is, as was shown by Zalka (1999), optimal and equivalent to Grover's algorithm. Plotting ΔE as a function of γN, as can be seen in figure 18, has a minimum at $\gamma N = 1$. The difference between the largest eigenvalue and second largest, plotted in the y-axis, is the smallest for a value of $\gamma N = 1 \implies \gamma = \frac{1}{N}$, which will correspond to the maximum probability for the marked vertex, in optimal steps.

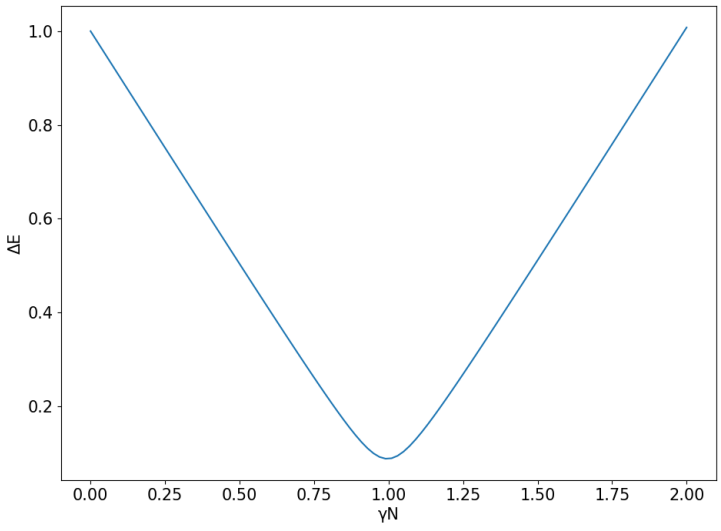


Figure 18: Value of the difference between the largest eigenvalue and the second largest, plotted as a function of γN, for $N = 512$.

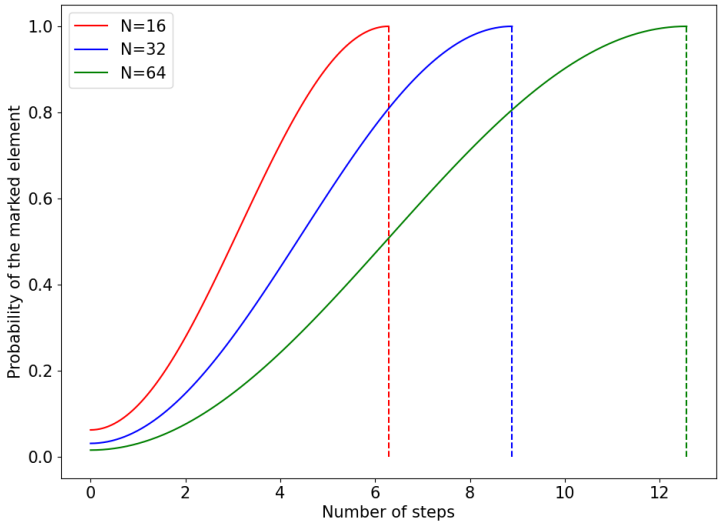


Figure 19: Continuous quantum walk search for a complete graph with 16, 32 and 64 vertices.

Figure 19 shows the evolution of the probability of the marked vertex in time, which is continuous in this model. In contrast with previous models, the distributions are smooth and reach exactly one, since the walk is allowed to evolve to exactly the ideal time steps.

4.4 STAGGERED QUANTUM WALK

The process for defining the search problem in this model is similar to the coined quantum walk case. The oracle still inverts the sign of a certain state and amplifies it, and the system's state will still be described by equation 73. However,instead of using a coin, the staggered model takes advantage of the notions of cliques and tessellations, as was shown in chapter **??**, which means the unmodified evolution operator has to be defined for an undirected complete graph.

As was shown in figure **??**, the vertices in a complete graph are all neighbors. This is a special case because this is the only connected graph where the tessellation cover can be done by one tessellation, since the graph is it's own clique. The minimum tessellations required to cover this structures are defined by the one clique that encompasses all N nodes of the graph

$$\mathscr{T}_\alpha = \{\{0, 1, 2, ..., N-1\}\}. \tag{85}$$

The associated polygon can then be described as the balanced superposition of all the nodes in the graph

$$|\alpha\rangle = \frac{1}{\sqrt{N}} \sum_{v=0}^{N-1} |v\rangle \,. \tag{86}$$

The Hamiltonian, as defined in 37, is

$$H_\alpha = 2 \sum_{0}^{1} |\alpha\rangle \langle\alpha| - I = 2 |\alpha_0\rangle \langle\alpha_0| - I \tag{87}$$

The unmodified evolution operator from equation 38

$$U = e^{i\theta_k H_k} ... e^{i\theta_2 H_2} e^{i\theta_1 H_1} \tag{88}$$

reduces to the single Hamiltonian case

$$U = e^{i\theta H_\alpha}. \tag{89}$$

The choice of the θ value is an important one, since maximum probability is achieved at $\theta = \frac{\pi}{2}$, as shown in figure 20. Since $H_\alpha^2 = I$, equation 89 can be rewritten as

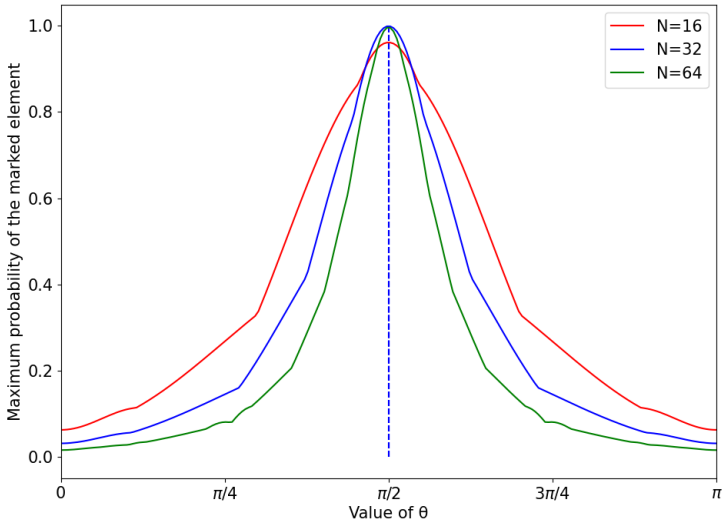


Figure 20: Maximum probability of the marked element as a function of the θ value plotted from 0 to π for number of nodes $N = 64, 128$ and 256.

$$U = e^{-i\frac{\pi}{2}H_\alpha} = \cos\frac{\pi}{2} I + i \sin\frac{\pi}{2} H_\alpha = iH_\alpha = i(2 |\alpha_0\rangle \langle\alpha_0| - I). \tag{90}$$

Having defined the the evolution operator associated with the complete graph, the next step is to use the oracle

$$\mathcal{O} = I_N - 2\left|0\right\rangle\left\langle 0\right|, \tag{91}$$

to create the modified evolution operator associated with the search

$$U' = U\mathcal{O}. \tag{92}$$

The walk achieves the same result as Grover's algorithm after $\frac{\pi}{4}\sqrt{N}$ steps, as shown in figure 21. This plot also shows that the probabilities converge to 1 as N increases, this is because time is discretized and deviations to the ideal steps will matter less for bigger values of N.

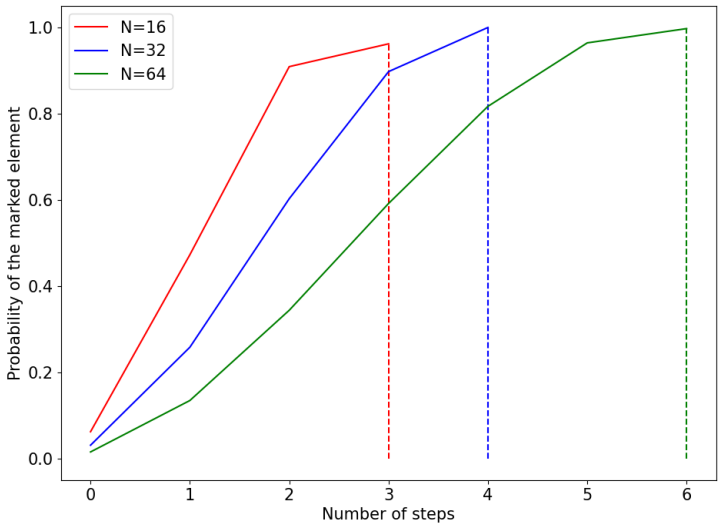


Figure 21: Staggered quantum walk search for a complete graph with 16, 32 and 64 nodes.

5

IMPLEMENTATIONS AND APPLICATIONS

5.1 COINED

Consider the example of a quantum walker on a discretely numbered cycle. It was seen that the evolution operator associated with such a system is, as was defined in equation 15

$$U = S(C \otimes I), \tag{93}$$

where S is a shift operator, defined in equation 14 as

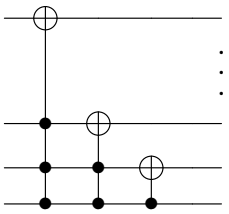
$$S = |0\rangle \langle 0| \otimes \sum_{x=-\infty}^{x=\infty} |x+1\rangle \langle x| + |1\rangle \langle 1| \otimes \sum_{x=-\infty}^{x=\infty} |x-1\rangle \langle x|, \tag{94}$$

that increments or decrements the position of the walker according to the coin operator C.

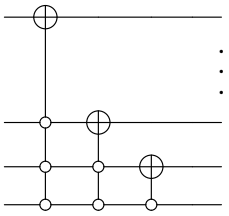
Previously, this system was simulated in Python by implenting it's equations. Now, the focus is to study a quantum circuit based on the work presented by Douglas and Wang (2009). This approach relies on multi-controlled CNOT gates in order to shift the state of the walker by $+1$ or -1, each with a probability associated with the chosen coin, as can be seen in figure 22.

The generalized CNOT gates act on the node states as a cyclic permutator, where each node is mapped to an adjacent state. This can be seen as the walker moving left or right, in the uni-dimensional graph example.

The coin operator will simply be a Hadamard gate acting on a single qubit. For a graph with 16 nodes, for example, 4 qubits are required to encode each node and an extra qubit for the coin. The circuit will then be as shown in figure 23. Note that this circuit limits the number of graph nodes to powers of 2, and an arbitrary implementation of 2^n nodes requires $n+1$ qubits. However, it is possible to have any number of nodes, given that the proper correction is made as can be seen in Douglas and Wang (2009). The method used was *Gray Code Ordering* proposed by Slepoy (2006), whereby a certain arrangement of CNOT gates results in control states only differing by a single bit.



(a) Douglas wang increment



(b) Douglas wang Decrement

Figure 22: Douglas wang shift operator

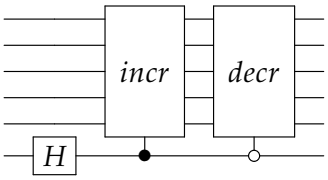


Figure 23: Douglas wang coined quantum walk circuit

In order to run this circuit on a real quantum computer using Qiskit, one must first find a way of creating generalized CNOT gates, since it is not available in the base package. One approach to this problem is to decompose an arbitrarily controlled CNOT gate into elementary gates, as was done by Barenco et al. (1995). In this context, the main idea is that for any unitary operator U, there exists operators such that

$$U = \phi AXBXC, \tag{95}$$

where $ABC = I$, X is the Pauli-X and ϕ is a phase operator described by $\phi = e^{i\delta} \times I$.

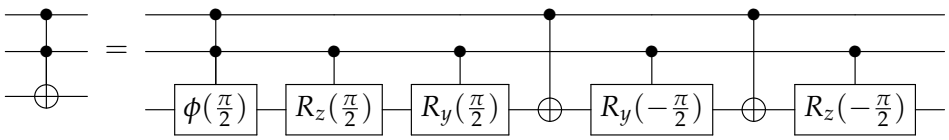


Figure 24: Toffoli decomposition

In order to understand this method, a good first example is the Toffoli gate, as is shown in figure 24. The first rotation in the circuit is defined by the R_z matrix

$$R_z(\theta) = \begin{pmatrix} e^{i\frac{\phi}{2}} & 0 \\ 0 & e^{i\frac{\phi}{2}} \end{pmatrix}, \tag{96}$$

where $\theta = \frac{\pi}{2}$. Secondly, the R_y rotation is

$$R_y(\phi) = \begin{pmatrix} cos(\frac{\theta}{2}) & -sin(\frac{\theta}{2}) \\ sin(\frac{\theta}{2}) & cos(\frac{\theta}{2}) \end{pmatrix}, \tag{97}$$

and $\phi = \frac{\pi}{2}$. The following rotations are simply $R_z^\dagger$ and $R_y^\dagger$. Lastly, the phase operator ϕ is

$$\phi(\delta) = \begin{pmatrix} e^{i\delta} & 0 \\ 0 & e^{i\delta} \end{pmatrix} \tag{98}$$

where $\delta = -\frac{\pi}{2}$. This phase correction is considered because otherwise

$$R_z(\frac{\pi}{2})R_y(\frac{\pi}{2})XR_y(-\frac{\pi}{2})XR_z(-\frac{\pi}{2}) = \begin{pmatrix} 0 & -i \\ -i & 0 \end{pmatrix} \neq X. \tag{99}$$

Introducing the phase correction results in

$$\phi(\frac{\pi}{2}) \begin{pmatrix} 0 & -i \\ -i & 0 \end{pmatrix} = \begin{pmatrix} i & 0 \\ 0 & i \end{pmatrix} \begin{pmatrix} 0 & -i \\ -i & 0 \end{pmatrix} = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} = X \tag{100}$$

However, since this is a global phase, it won't be included since it has no effect on the result of the measurement.

A more generalized version of this method can be seen in figure 25. Each individual generalized CNOT gate in this circuit can be expanded as was done for the Toffoli gate example, stopping once the generalised inverter gates are simply Toffoli gates.

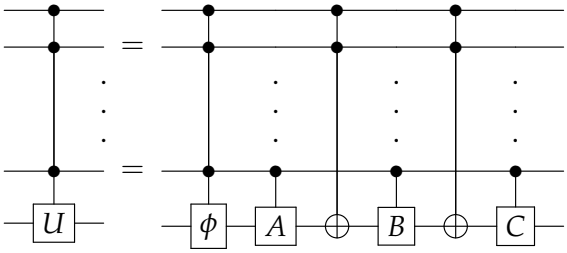


Figure 25: General decomposition

This was the chosen method because it provides a way of implementing arbitrarily controlled CNOT gates without the use of ancillary qubits, which are a scarce resource.

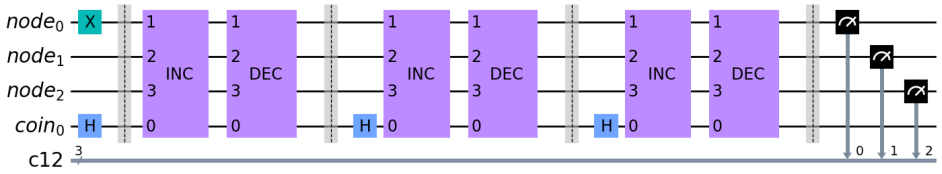


Figure 26: Temp

This circuit was implemented in Qiskit, as can be seen in figure 26. In this example, the increment and decrement sequence was applied three times on a graph of size $2^3 = 8$ nodes. The starting position of the walker was set to $\Psi(0) = |4\rangle$ and the Hadamard coin was used. The first block after the barrier is the sequence of operations that will increment the state of the walker, as is shown in figure 27. The circuit is simply the CNOT decomposition of figure 25 applied to the increment circuit of figure 22a for the $N = 4$, qubit case. The following

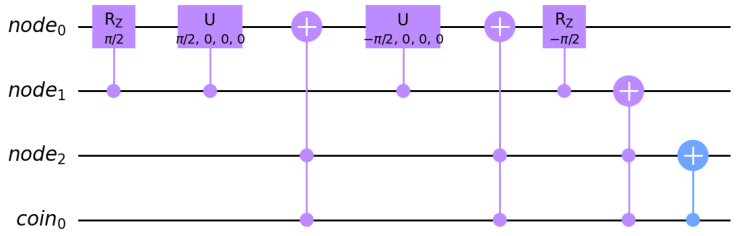


Figure 27: Temp

block represents the decrement of the state of the walker, which is just an increment block with it's controls negated as is shown in figure 28. The rest of the circuit is just the repetition of these operations as a function of the number of time steps required.

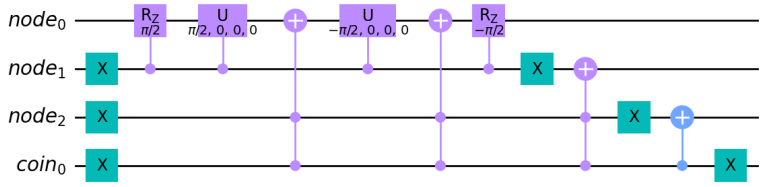


Figure 28: Temp

Lastly, the circuit is measured and the results can be seen in figure 29.These results can be verified by calculating the time evolution of the wave function associated with the system

$$|\Psi(0)\rangle = |4\rangle \tag{101}$$

$$|\Psi(1)\rangle = \frac{|0\rangle\,|x=3\rangle + |1\rangle\,|x=5\rangle}{\sqrt{2}} \tag{102}$$

$$|\Psi(2)\rangle = \frac{|0\rangle\,|x=2\rangle + |1\rangle\,|x=4\rangle + |0\rangle\,|x=4\rangle - |1\rangle\,|x=6\rangle}{2} \tag{103}$$

$$|\Psi(3)\rangle = \frac{|1\rangle\,|x=1\rangle - |0\rangle\,|x=3\rangle + 2(|0\rangle + |1\rangle)\,|x=5\rangle + |0\rangle\,|x=7\rangle}{2\sqrt{2}}. \tag{104}$$

Taking the modulus squared of the amplitudes associated with the states, confirms that the probability distribution presented in figure 29 is correct.

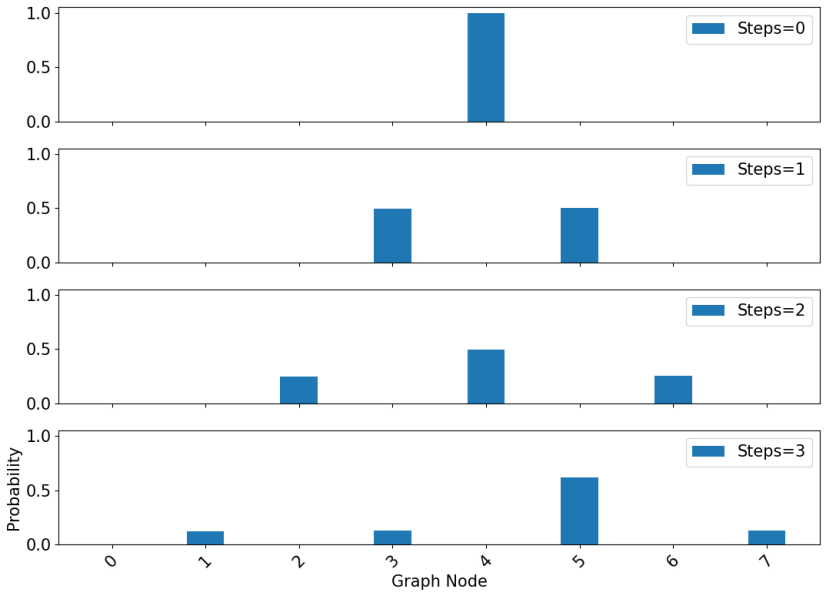


Figure 29: Temp

5.2 CONTINUOUS

As was seen in section 3.3, the unitary evolution operator of this model is defined as

$$U(t) = e^{-iHt} = e^{i(-\gamma L)t} = e^{-i\gamma(A+D)t}. \tag{105}$$

Considering a regular graph, this operator can be rewritten as

$$U(t) = \phi(t)e^{-i\gamma(A)t}, \tag{106}$$

where $\phi(t)$ is a global phase and A is the adjacency matrix associated with the graph.

Here, the study will focus on the circuit implementation of this walk in a cyclic graph, and a different approach was used to define the adjacency matrix. This approach relies on the concept of a circulant graph, which are a class of graphs defined by a circulanti matrix such that

$$A = \begin{pmatrix} c_0 & c_{N-1} & \cdots & c_3 & c_2 \\ c_1 & c_0 & c_{N-1} & & c_3 \\ \vdots & c_1 & c_0 & \ddots & \vdots \\ c_{N-2} & & \ddots & \ddots & c_{N-1} \\ c_{N-1} & c_{N-2} & \cdots & c_1 & c_0 \end{pmatrix}. \tag{107}$$

In order to generate the proper circulant graphs, restrictions on this matrix are needed. Firstly, $c_0 = 0$, since self-loops are not part of the structure. Secondly, the matrix must be symmetric, therefore $c_{n-j} = c_j$.

These matrices can be fully described by the first column of the matrix

$$v_1 = [c_0, c_1, \cdots, c_{N-2}, c_{N-1}]^T \tag{108}$$

with a discrete convolution operator performing cyclic permutations of c, on each column. For example,

$$Dv_1 = [c_{N-1}, c_0, \cdots, c_{N-3}, c_{N-2}]^T = v_2. \tag{109}$$

More specifically, for the cycle case

$$Dv_1 = D[0,1,0,\cdots,0,1]^T = [1,0,1,0,\cdots,0,0]^T = v_2. \tag{110}$$

The operator can then be recursively defined as

$$Dv_k = v_{k+1}. \tag{111}$$

and the matrix will be

$$A = \sum_{j=0}^{N-1} Dv_j \tag{112}$$

$$A^k = \sum_{j=0}^{N-1} c_{(j-1) \mod N} \tag{113}$$

The eigenvalues of a circulant matrix can be found by

$$\lambda_p = c_0 + \sum_{q=1} c_{N-q} \omega^{pq}, \tag{114}$$

and the eigenvectors are

$$|\varphi_p\rangle = \frac{1}{\sqrt{n}} \sum_{q=0}^{n-1} \omega^{pq}. \tag{115}$$

This given, it is possible to construct an operator that diagonalizes the circulant matrix through the eigenvectors, which is useful for constructing the circuit. For this purpous, the Quantum Fourier Transform can be used and it is defined as

$$F = \frac{1}{\sqrt{N}} \sum_{p,q} \omega^{pq} |p\rangle \langle q| . \tag{116}$$

The adjacency matrix of a circulant graph can then be diagonalized such that

$$A = F^\dagger \Lambda F, \tag{117}$$

where Λ is a diagonal operator that encodes the eigenvalues

$$\Lambda = \sum_j \lambda_j |j\rangle \langle j| . \tag{118}$$

The unitary operator of the walk can then be rewritten as

$$U = F^\dagger e^{i\gamma\Lambda t} F \tag{119}$$

where

$$e^{i\gamma\Lambda t} = \sum_j e^{i\gamma\lambda_j t} |j\rangle \langle j| . \tag{120}$$

The circuit can now be constructed making use of the *diagonal* function provided by Qiskit, which decomposes diagonal operators based on the method presented in theorem 7 of Shende et al. (2006). The other tool used was the Quantum Fourier Transform, also provided by the Qiskit package. Figure 30 shows the implementation of the circuit for 3 qubits or $2^3 = 8$ graph nodes and $t = 3$.

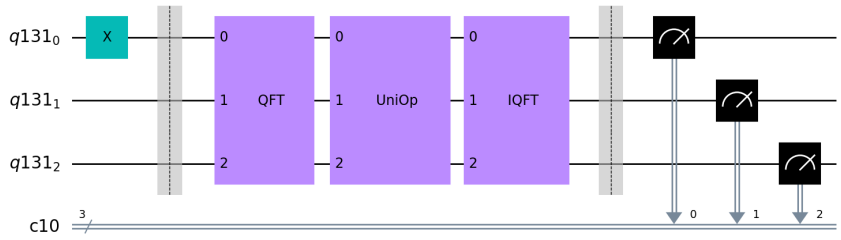


Figure 30: Temp

The circuit for the Quantum Fourier transform is well known and presented in figure 31. The circuit associated with $F^{\dagger}$ is similarly constructed by negating the previous figure's rotations.

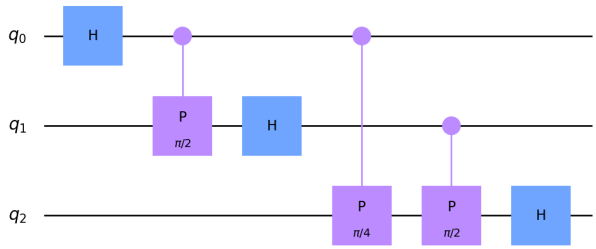


Figure 31: Temp

The circuit associated with the diagonal operator is show in figure 32. Furthermore equation 119 says that time is simply a constant inside the exponential, which means that the diagonal operator's circuit will not need extra gates when increasing time, it will only need different rotations and differ in global phase. This is an advantage when comparing to

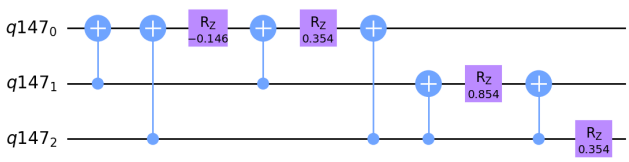


Figure 32: temp

the previous model, seen in figure 26, where each extra step required another increment

and decrement gates. Further evidence of this behaviour can be seen in figure 33, where it was created a circuit for $t = 0$ up to $t = 100$ with increments of 1. It was then counted the number of gates, for each circuit, and plotted against the respective time. This graph

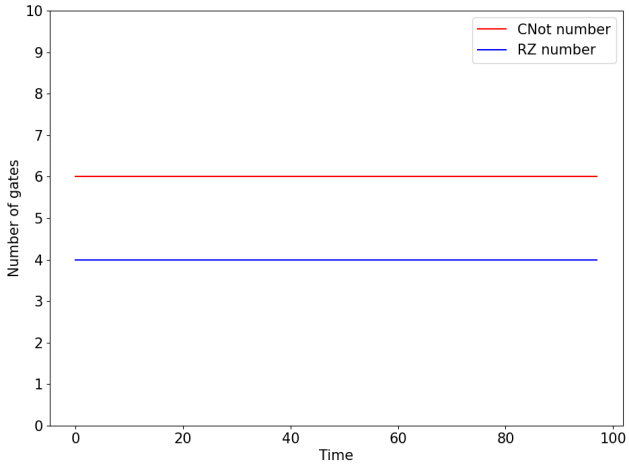


Figure 33: Temp

clearly shows that both the number of R_z rotations and CNot operations remain constant throughout the entire time interval.

Finally, the circuit was measured, and the result can be seen in figure 34

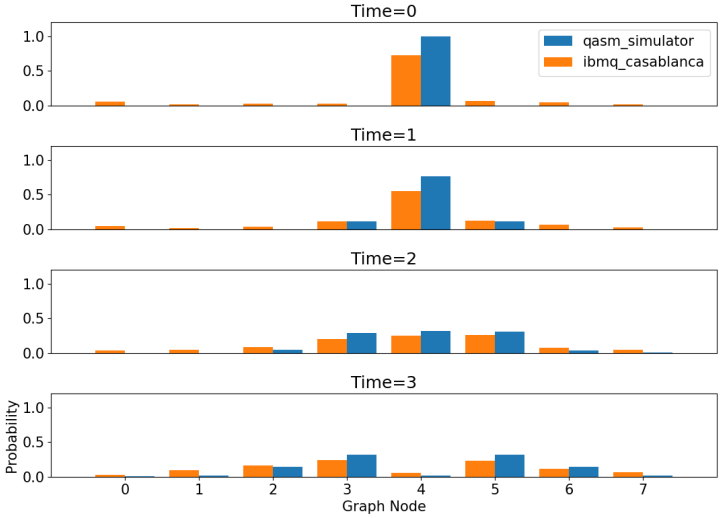


Figure 34: Temp

5.3 STAGGERED

5.4 SEARCH PROBLEMS WITH QISKIT

5.4.1 *Grover*

Like it was seen in section **??**, Grover's algorithm is a quantum alternative to unstructured search problems. Consider the case of finding element x_0 out of an unordered list of size N. Worst case scenario, a classical algorithm would need to check every element of the list, requiring N steps.

The first stage of Grover's algorithm is to create an uniform superposition of all states in the system

$$|\Psi_0\rangle = \frac{1}{\sqrt{N}} \sum_{x=0}^{N-1} |x\rangle . \tag{121}$$

Next is the application of the Grover iteration process, which starts with an oracle that adds a negative phase to the solution states

$$\mathcal{O} |x\rangle = (-1)^{f(x)} |x\rangle . \tag{122}$$

This operator can be seen as an identity matrix with negative entries corresponding to the solution states, and the operator can be rewritten as

$$\mathcal{O} = I - 2 \sum_{m \in M} |m\rangle \langle m| . \tag{123}$$

where I is the identity matrix and M is a set of solutions where $f(m) = 1$, otherwise it's 0. The matrix associated with this operator will be

$$\mathcal{O} = \begin{pmatrix} (-1)^{f(0)} & 0 & \cdots & 0 \\ 0 & (-1)^{f(1)} & \cdots & 0 \\ \vdots & 0 & \ddots & \vdots \\ 0 & 0 & \cdots & (-1)^{f(N-1)} \end{pmatrix} . \tag{124}$$

The second part of the iteration is an amplitude amplification process by means of the diffusion operator

$$\mathcal{D} = (2 |\Psi_0\rangle \langle \Psi_0| - I) = H^{\otimes n}(2 |0\rangle \langle 0| - I)H^{\otimes n}. \tag{125}$$

The unitary operator that describes the Grover iteration process will then be

$$\mathcal{U} = \mathcal{D}\mathcal{O}. \tag{126}$$

As was shown in section **??** this iteration process will be done several times, depending on the number of elements. Optimal probability of success finding a single solution will be reached after $\lfloor \frac{\pi}{4}\sqrt{N} \rfloor$ steps, and $\lfloor \frac{\pi}{4}\sqrt{\frac{N}{K}} \rfloor$ for K solutions, which is a quadratic gain when compared to the classical case.

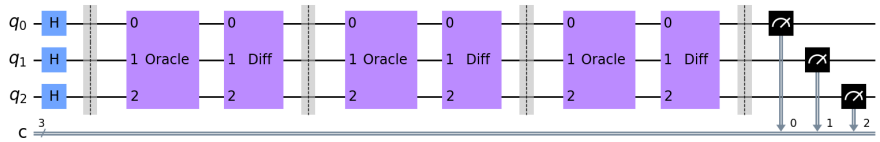


Figure 35: Temp

Consider the 3 qubit case, where $N = 8$ and solution state $|4\rangle$. The optimal number of iterations is approximately 2, and figure 35 is the circuit for 3 iterations. The system starts with the creation of an uniform superposition state, which means applying Hadamard gates to each qubit. Immediately following the barrier, the first operator of the iteration process is

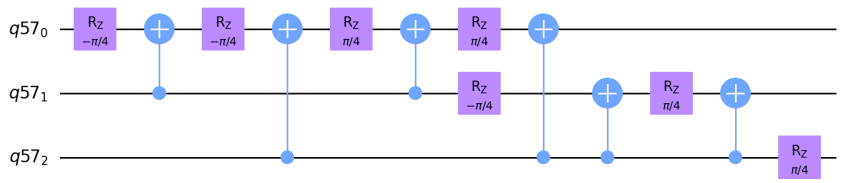


Figure 36: Temp

the oracle, which is shown in figure 36. Because the oracle operator is simply the identity matrix with negative entries corresponding to the solution states, it can be simply translated into a circuit by means of the diagonal function in Qiskit. The last part of the iteration is the diffusion operator, whose circuit is shown in figure 37.

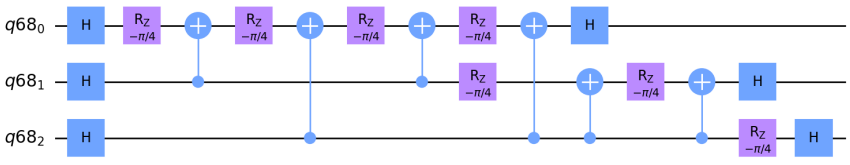


Figure 37: Temp

Comparing equations 123 and 125, it is easy to see why figures 36 and 37 are very similar. The diffusion circuit will simply be the oracle circuit for state $|0\rangle$ in between Hadamard operations.

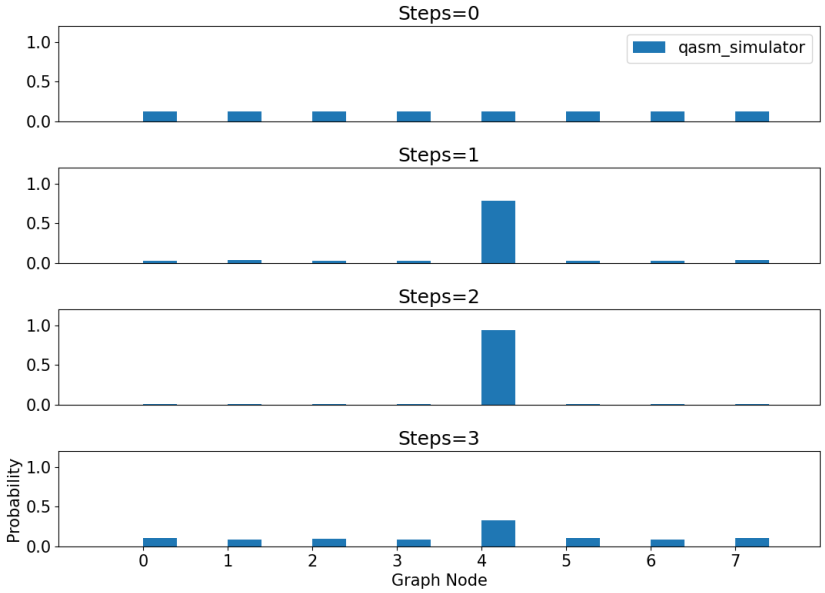


Figure 38: Temp

The results of measurement are shown in figure 38. As was expected, maximum probability for the marked element was reached after 2 iterations and it decreases in subsequent steps.

5.4.2 *Coined*

As previously done in section **??**, this chapter aims to expand the coined quantum walk model incorporating concepts like the oracle and diffusion operator of the Grover algorithm. In fact, for the complete graph case, the coined quantum walk and Grover's algorithm are equivalent.

The modified unitary evolution operator is

$$U' = S(\mathcal{O} \otimes G), \tag{127}$$

as was defined in equation 79, where S is the flip-flop shift operator, $\mathcal{O}$ is the oracle operator and G is the Grover diffusion as a coin operator.

Consider the case of a complete graph, where every vertex is adjacent to one another. The quantum circuit to implement this, as shown in figure 39, will require N qubits to represent the state of the walker and N qubits for the state of the coin. The shift operator

was constructed based on the work of Douglas and Wang (2009), where the state of the walker is flip-flopped with the state of the coin, which can be done through swap gates.

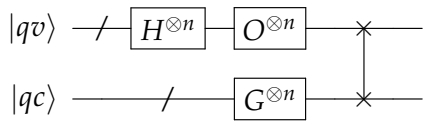


Figure 39: Douglas wang coined quantum walk circuit

This was implemented in Qiskit, for a graph of size $N = 2^3 = 8$, which means 6 qubits will be required. For the case of one marked element, the number of iterations that maximizes the amplitude of the solution state is $\lfloor \frac{\pi}{2}\sqrt{N} \rfloor$, and figure 40 shows the circuit for 5 iterations of the walk.

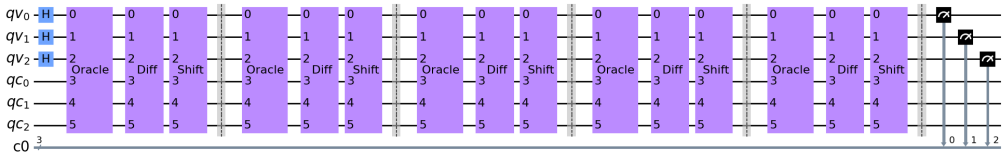


Figure 40: Temp

The circuit starts in a uniform superposition of the states corresponding to the vertices of the graph, and the first step of the iteration is the oracle. This operator will flip the amplitude of the vertex state $|4\rangle$, as is shown in figure 41. It is the same operator as in figure 36, but applied only to states belonging in the position space of the walk.

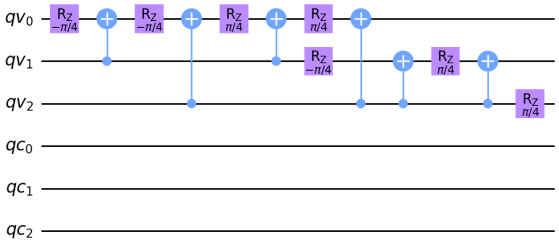


Figure 41: Temp

The states associated with the coin space of the walk will be transformed according to Grover's diffusion, as is seen in figure 42.

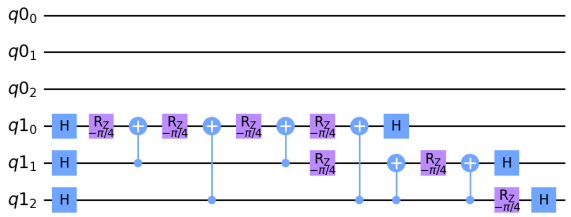


Figure 42: Temp

The final part of the iteration is the shift operator, as can be seen in figure 43. The flip-flop shift operator was defined in equation 74 as

$$S\left|v1\right\rangle\left|v2\right\rangle = \left|v2\right\rangle\left|v1\right\rangle, \tag{128}$$

where $|v1\rangle$ represents the position of the walker and $|v2\rangle$ is the state of the coin. It is trivial to implement this as shift operations between the vertex states and coin states.

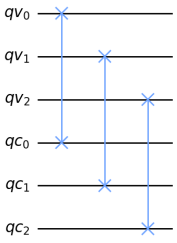


Figure 43: Temp

Lastly, measurements were performed, and the results plotted in figure 44. Maximum probability of the marked element was reached after 4 steps, and extra steps reduce said probability.

5.4.3 *Continuous*

As was seen in section **??**, the unitary operator associated with the continuous time quantum walk model can be modified as to mark an element for amplitude amplification

$$U'(t) = e^{iH't} = \phi(t)e^{-i\gamma(A+O)t}, \tag{129}$$

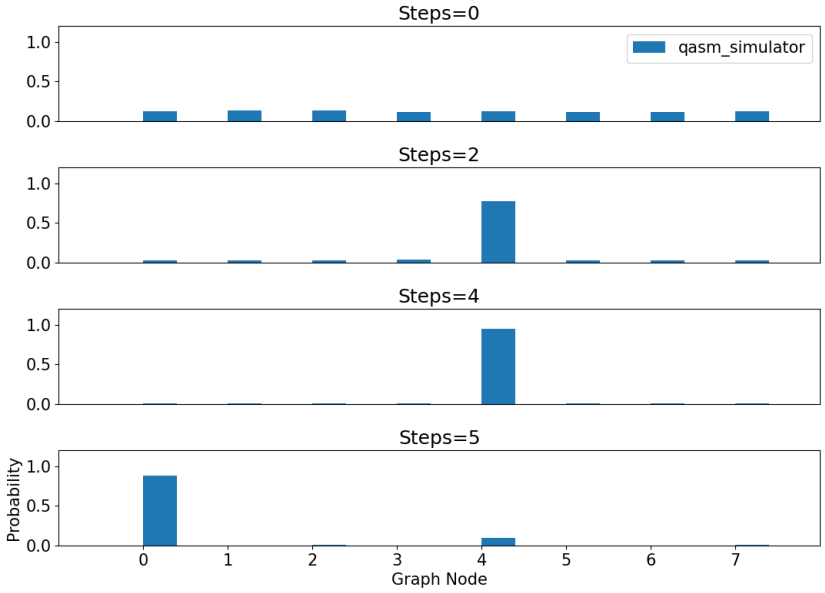


Figure 44: Temp

where $\phi(t)$ is a global phase, A is the adjacency matrix and the oracle is defined as

$$O = \sum_{m \in M} |m\rangle \langle m|, \tag{130}$$

where M is the set of marked elements.

This section will focus on constructing and analyzing the circuit form of the continuous-time quantum walk search problem, and the first step is to borrow the diagonal definition of the adjacency matrix from equation 117

$$A = F^{\dagger} \Lambda F, \tag{131}$$

and use the Suzuki-Trotter expansion

$$e^{i(H_0+H_1)t} = \lim_{n \to \infty} (e^{i\frac{H_0 t}{n}} e^{i\frac{H_1 t}{n}})^n, \tag{132}$$

to decompose the operator in equation 129

$$e^{i\gamma(A+O)t} = \lim_{n \to \infty} (F^{\dagger} e^{i\gamma\frac{\Lambda t}{n}} F e^{i\gamma\frac{Ot}{n}})^n, \tag{133}$$

which can be easily translated into circuit form as in figure 45.

Consider the case of a graph of size $N = 3$ and trotter number of $n = 2$. The corresponding Qiskit circuit is as shown in figure 45. The system starts out in an uniform superposition followed by a state transformation according to the oracle operator that can be seen in figure

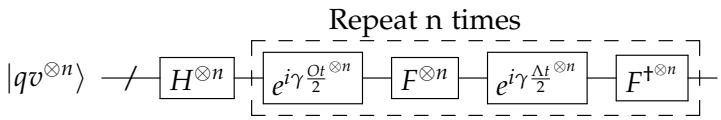


Figure 45: Temp

47. Note that the circuit was obtained by using the Qiskit diagonal function that takes the diagonal entries of the operator corresponding to the oracle, as in equation 133. The

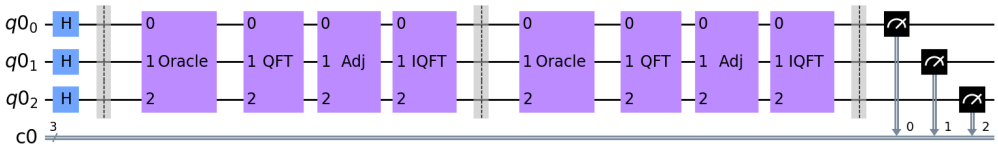


Figure 46: Temp

following transformations will be the quantum Fourier transform, which is the same as in figure 31, and the operator associated with the adjacency matrix. Since A is the diagonal adjacency matrix of a complete graph, it is easily implemented using Qiskit, as can be seen in figure 48.

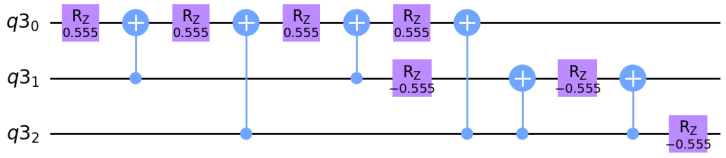


Figure 47: Temp

The results of measurement are shown in figure 49.

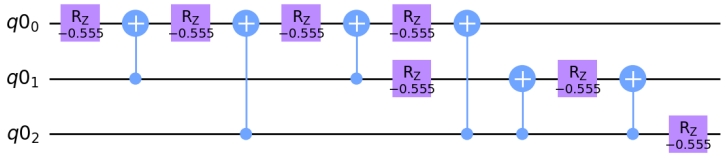


Figure 48: Temp

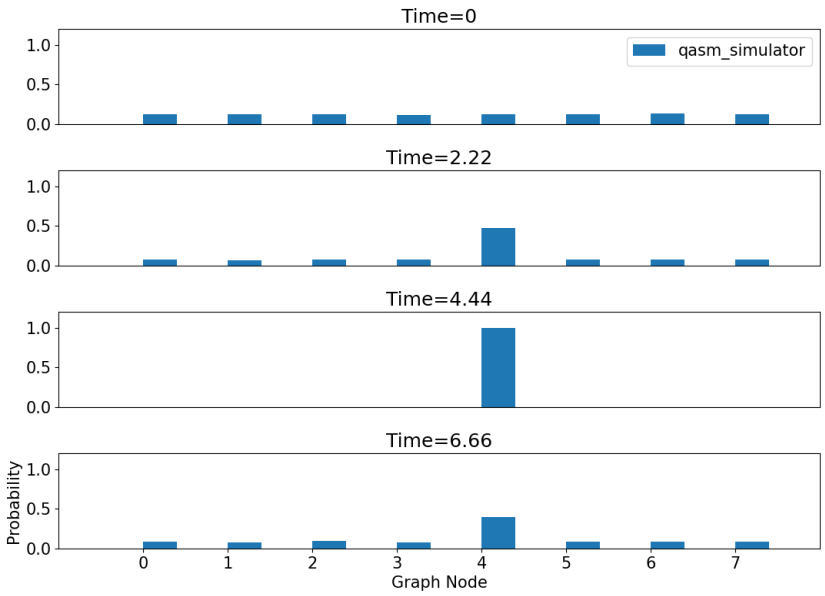


Figure 49: Temp

5.4.4 *Staggered*

BIBLIOGRAPHY

F. Acasiete, F. P. Agostini, J. Khatibi Moqadam, and R. Portugal. Implementation of quantum walks on ibm quantum computers. *Quantum Information Processing*, 19(12), Nov 2020. ISSN 1573-1332. doi: 10.1007/s11128-020-02938-5. URL http://dx.doi.org/10.1007/s11128-020-02938-5.

Dorit Aharonov, Andris Ambainis, Julia Kempe, and Umesh Vazirani. Quantum walks on graphs. *STOC '01 Proceedings of the thirty-third annual ACM symposium on Theory of computing*, pages 50–59, 2001.

Dorit Aharonov, Wim van Dam, Julia Kempe, Zeph Landau, Seth Lloyd, and Oded Regev. Adiabatic quantum computation is equivalent to standard quantum computation. *SIAM Journal on Computing*, 37(1), 2007. doi: 166-194.

Y. Aharonov, L. Davidovich, and N. Zagury. Quantum random walks. *Physical Review A*, 48 (2):1687–1690, 1993.

Andris Ambainis. Quantum walk algorithm for element distinctness. *SIAM Journal on Computing*, 37(1):210–239, 2007.

Radhakrishnan Balu, Daniel Castillo, and George Siopsis. Physical realization of topological quantum walks on ibm-q and beyond. 2017.

Adriano Barenco, Charles H. Bennett, Richard Cleve, David P. DiVincenzo, Norman Margolus, Peter Shor, Tycho Sleator, John A. Smolin, and Harald Weinfurter. Elementary gates for quantum computation. *Physical Review A*, 52(5):3457–3467, Nov 1995. ISSN 1094-1622. doi: 10.1103/physreva.52.3457. URL http://dx.doi.org/10.1103/PhysRevA.52.3457.

Adriano Barenco, Artur Ekert, Kalle-Antti Suominen, and Päivi Törmä. Approximate quantum fourier transform and decoherence. *Physical Review A*, 54(1):139–146, Jul 1996. ISSN 1094-1622. doi: 10.1103/physreva.54.139. URL http://dx.doi.org/10.1103/PhysRevA.54.139.

Paul Benioff. The computer as a physical system: A microscopic quantum mechanical hamiltonian model of computers as represented by turing machines. *Journal of Statistical Physics*, 22(5):563–591, 1980.

Charles Bennett and Gilles Brassard. Withdrawn: Quantum cryptography: Public key distribution and coin tossing. *Theoretical Computer Science - TCS*, 560:175–179, 01 1984. doi: 10.1016/j.tcs.2011.08.039.

Charles H. Bennett and Stephen J. Wiesner. Communication via one- and two-particle operators on einstein-podolsky-rosen states. *Phys. Rev. Lett.*, 69:2881–2884, Nov 1992. doi: 10.1103/PhysRevLett.69.2881. URL https://link.aps.org/doi/10.1103/PhysRevLett.69.2881.

Scott D. Berry, Paul Bourke, and Jingbo B. Wang. qwviz: Visualisation of quantum walks on graphs. *Computer Physics Communications*, 182(10):2295–2302, 2011. ISSN 0010-4655. doi: https://doi.org/10.1016/j.cpc.2011.06.002. URL https://www.sciencedirect.com/science/article/pii/S0010465511002128.

Michael Boyer, Gilles Brassard, Peter Høyer, and Alain Tapp. Tight bounds on quantum searching. *Fortschritte der Physik*, 46(4-5):493–505, 1998.

A. R. Calderbank and Peter W. Shor. Good quantum error-correcting codes exist. *Phys. Rev. A*, 54:1098–1105, Aug 1996. doi: 10.1103/PhysRevA.54.1098. URL https://link.aps.org/doi/10.1103/PhysRevA.54.1098.

D. Cheung. Improved bounds for the approximate qft. 2004.

Chen-Fu Chiang, Daniel Nagaj, and Pawel Wocjan. Efficient circuits for quantum walks. *Quantum Information and Computation*, 10, 03 2009. doi: 10.26421/QIC10.5-6-4.

Andrew M. Childs. Universal computation by quantum walk. *Physical Review Letters*, 102 (18):180501, 2009.

Andrew M. Childs and Jeffrey Goldstone. Spatial search by quantum walk. *Physical Review A*, 70(2):022314, 2004.

Andrew M. Childs, Richard Cleve, Enrico Deott, Edward Farhi, Sam Gutmann, and Daniel A. Spielman. Exponential algorithmic speedup by quantum walk. *Proc. 35th ACM Symposium on Theory of Computing (STOC 2003), pp. 59-68*, 2002. doi: 10.1145/780542.780552.

Paul H. Cootner. *The random character of stock market prices.* M.I.T. Press, Cambridge, Mass, rev. ed. edition, 1967.

D. Coppersmith. An approximate fourier tqquantum fourier transformuantum fourier transformransform useful n quantum factoring", ibm research report rc19642 ,; r. cle. 1994.

Gabriel Coutinho and Renato Portugal. Discretization of continuous-time quantum walks via the staggered model with hamiltonians. *Natural Computing*, pages 1–7, 2018.

David Deutsch. Quantum theory, the church-turing principle and the universal quantum computer. *Proceedings of The Royal Society A Mathematical Physical and Engineering Sciences*, 400(1818), 1985.

David Deutsch and Richard Jozsa. Rapid solution of problems by quantum computation. *Proceedings of The Royal Society A Mathematical Physical and Engineering Sciences*, 439(1907), 1992.

B.L. Douglas and J. B. Wang. Efficient quantum circuit implementation of quantum walks. 2009.

Martin Dyer, Alan Frieze, and Ravi Kannan. A random polynomial-time algorithm for approximating the volume of convex bodies. *Journal of the ACM*, 38(1):1–17, 1991.

Matthew D. Falk. Quantum search on the spatial grid. 2013.

Peter E. Falloon, Jeremy Rodriguez, and Jingbo B. Wang. Qswalk: A mathematica package for quantum stochastic walks on arbitrary graphs. *Computer Physics Communications*, 217: 162–170, 2017. ISSN 0010-4655. doi: https://doi.org/10.1016/j.cpc.2017.03.014. URL https://www.sciencedirect.com/science/article/pii/S0010465517301029.

Edward Farhi and Sam Gutmann. An analog analogue of a digital quantum computation. *Physical Review A*, 57(4):2403–2406, 1998.

Edward Farhi, Jeffrey Goldstone, Sam Gutmann, and Michael Sipser. Quantum computation by adiabatic evolution. 2000.

Richard P. Feynman. There's plenty of room at the bottom. *Feynman and computation*, pages 63–76, 1959.

Richard P. Feynman. Simulating physics with computers. *International Journal of Theoretical Physics*, 21(6-7):467–488, 1982.

Steven Finch. "pólya's random walk constant." in mathematical constants. *Cambridge University Press*, pages 322–331, 2003.

Konstantinos Georgopoulos and P. Zuliani. One-dimensional hadamard quantum walk on a cycle with rotational implementation. *arXiv: Quantum Physics*, 2019.

Adam Glos, Jarosław Adam Miszczak, and Mateusz Ostaszewski. Qswalk.jl: Julia package for quantum stochastic walks analysis. *Computer Physics Communications*, 235:414–421, Feb 2018. ISSN 0010-4655. doi: 10.1016/j.cpc.2018.09.001. URL http://dx.doi.org/10.1016/j.cpc.2018.09.001.

Geoffrey Grimmett, Svante Janson, and Petra F. Scudo. Weak limits for quantum random walks. *Physical Review E*, 69(2):026119, 2003.

Lov K. Grover. A fast quantum mechanical algorithm for database search. *STOC '96 Proceedings of the twenty-eighth annual ACM symposium on Theory of computing*, pages 212–219, 1996.

C. A. R. Hoare. Algorithm 64: Quicksort. *Commun. ACM*, 4(7):321, 1961. ISSN 0001-0782. doi: 10.1145/366622.366644. URL https://doi.org/10.1145/366622.366644.

Norio Inui, Yoshinao Konishi, and Norio Konno. Localization of two-dimensional quantum walks. *Physical Review A*, 69(5):052323, 2003.

Josh Izaac and Jb Wang. Pyctqw: A continuous-time quantum walk simulator on distributed memory computers. *Computer Physics Communications*, 186, 01 2015. doi: 10.1016/j.cpc.2014.09.011.

Mark Jerrum, Alistair Sinclair, and Eric Vigoda. A polynomial-time approximation algorithm for permanent of a matrix with nonnegative entries. *Journal of the ACM*, 51(4):671–697, 2004.

Norio Konno. A new type of limit theorems for the one-dimensional quantum random walk. *J. Math. Soc. Japan*, 57(4):1179–1195, 2002.

T Loke and J B Wang. Efficient quantum circuits for continuous-time quantum walks on composite graphs. *Journal of Physics A: Mathematical and Theoretical*, 50(5):055303, Jan 2017a. ISSN 1751-8121. doi: 10.1088/1751-8121/aa53a9. URL http://dx.doi.org/10.1088/1751-8121/aa53a9.

T. Loke and J.B. Wang. Efficient quantum circuits for szegedy quantum walks. *Annals of Physics*, 382:64–84, Jul 2017b. ISSN 0003-4916. doi: 10.1016/j.aop.2017.04.006. URL http://dx.doi.org/10.1016/j.aop.2017.04.006.

Neil B. Lovett, Sally Cooper, Matthew Everett, Matthews Trevers, and Viv Kendon. Universal quantum computation using the discrete quantum walk. *Physical Review A*, 81(4):042330, 2010.

Frédéric Magniez, Ashwin Nayak, Jérémie Roland, and Miklos Santha. Search via quantum walk. *SIAM Journal on Computing*, 40(1):142–164, 2006.

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

F.L. Marquezino and R. Portugal. The qwalk simulator of quantum walks. *Computer Physics Communications*, 179(5):359–369, Sep 2008. ISSN 0010-4655. doi: 10.1016/j.cpc.2008.02.019. URL http://dx.doi.org/10.1016/j.cpc.2008.02.019.

Elliott Montroll. Random walks in multidimensional spaces, especially on periodic lattices. *Journal of the Society for Industrial and Applied Mathematics*, 4(4):241–260, 1956. doi: 10.1137/0104014.

Elliott Waters Montroll and George Herbert Weiss. Random walks on lattices. ii. *Journal of Mathematical Physics*, page 167–181, 1997.

Gordon E. Moore. Cramming more components onto integrated circuits. *Electronics*, 38(8): 114–117, 1965.

J .K. Moqadam, M. C. de Oliveira, and Renato Portugal. Staggered quantum walks with superconducting microwave resonators. *Physical Review B*, 95(14):144506, 2017.

Rajeev Motwani and Prabhakar Raghavan. *Randomized Algorithms*. Cambridge University Press, 1995.

Ashwin Nayak and Ashvin Vishwanath. Quantum walk on the line. 2000.

Christos Papadimitrious. *Computational Complexity*. Pearson, 1994.

Apoorva Patel, K.S. Raghunathan, and Pranaw Rungta. Quantum random walks do not need a coin toss. *Physical Review A*, 71(3):032347, 2005.

Karl Pearson. The problem of the random walk. *Nature*, 72(1865):294, 1905. doi: 10.1038/072294b0.

J. Pollard, Lenstra Arjen, H. Lenstra, and Mark Manasse. The number field sieve. 1994.

R. Portugal, R.A.M. Santos, T.D. Fernandes, and D.N. Goncalves. The staggered quantum walk model. *Quantum Information Processing*, 15(1):85–101, 2016.

Renato Portugal. Establishing the equivalence between szegedy's and coined quantum walks using the staggered model. *Quantum Information Processing*, 15(4):1387–1409, 2015.

Renato Portugal. *Quantum Walks and Search Algorithms*. Springer, 2018.

Renato Portugal and T. D. Fernandes. Quantum search on the two-dimensional lattice using the staggered model with hamiltonians. *Physical Review A*, 95(4):042341, 2017.

Renato Portugal, Stefan Boettcher, and Stefan Falkner. One-dimensional coinless quantum walks. *Physical Review A*, 91(5):052319, 2015.

Renato Portugal, M. C. de Oliveira, and J. K. Moqadam. Staggered quantum walks with hamiltonians. *Physical Review A*, 95(1):012328, 2017.

George Pólya. Über eine aufgabe der wahrscheinlichkeitsrechnung betreffend die irrfahrt im straßennetz. *Mathematische Annalen*, 84:149–160, 1921. doi: 10.1007/BF01458701.

Xiaogang Qiang, Thomas Loke, Ashley Montanaro, Kanin Aungskunsiri, Xiaoqi Zhou, Jeremy L. O'Brien, Jingbo B. Wang, and Jonathan C. F. Matthews. Efficient quantum walk on a quantum processor. *Nature Communications*, 7(1), May 2016. ISSN 2041-1723. doi: 10.1038/ncomms11511. URL http://dx.doi.org/10.1038/ncomms11511.

Miklos Santha. Quantum walk based search algorithms. *International Conference on Theory and Applications of Models of Computation. TAMC 2008. Lecture Notes in Computer Science*, 4978:31–46, 2008.

Marek Sawerwain and Roman Gielerak. Gpgpu based simulations for one and two dimensional quantum walks. *Communications in Computer and Information Science*, page 29–38, 2010. ISSN 1865-0937. doi: 10.1007/978-3-642-13861-4_3. URL http://dx.doi.org/10.1007/978-3-642-13861-4_3.

Benjamin Schumacher. Quantum coding. *Phys. Rev. A*, 51:2738–2747, Apr 1995. doi: 10.1103/PhysRevA.51.2738. URL https://link.aps.org/doi/10.1103/PhysRevA.51.2738.

Uwe Schöning. A probabilistic algorithm for k-sat and constraint satisfaction problems. *40th Annual Symposium on Foundations of Computer Science*, pages 410–, 1999.

Asif Shakeel. Efficient and scalable quantum walk algorithms via the quantum fourier transform. *Quantum Information Processing*, 19(9), Aug 2020. ISSN 1573-1332. doi: 10.1007/s11128-020-02834-y. URL http://dx.doi.org/10.1007/s11128-020-02834-y.

C. E. Shannon. A mathematical theory of communication. *The Bell System Technical Journal*, 27(3):379–423, 1948. doi: 10.1002/j.1538-7305.1948.tb01338.x.

V.V. Shende, S.S. Bullock, and I.L. Markov. Synthesis of quantum-logic circuits. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 25(6):1000–1010, Jun 2006. ISSN 1937-4151. doi: 10.1109/tcad.2005.855930. URL http://dx.doi.org/10.1109/TCAD.2005.855930.

Neil Shenvi, Julia Kempe, and Birgitta Whaley. A quantum random walk search algorithm. *Physical Review A*, 67(5):052307, 2003.

Peter W. Shor. Algorithms for quantum computation: Discrete logarithms and factoring. *Proceedings 35th Annual Symposium on Foundations of Computer Science*, pages 124–134, 1994a.

P.W. Shor. Algorithms for quantum computation: discrete logarithms and factoring. *Proceedings 35th Annual Symposium on Foundations of Computer Science*, pages 124–134, 1994b. doi: 10.1109/SFCS.1994.365700.

Alexander Slepoy. Quantum gate decompostition algorithms. *Sandia National Laboratories*, 2006.

R. Solovay and V. Strassen. A fast monte-carlo test for primality. *SIAM Journal on Computing*, 6(1):84–85, 1977. doi: 10.1137/0206006. URL https://doi.org/10.1137/0206006.

Tommi Sottinen. Fractional brownian motion, random walks and binary market models. *Finance and Stochastics*, (5):343–355, 2001. doi: 10.1007/PL00013536.

Andrew Steane. Multiple-particle interference and quantum error correction. *Proceedings of the Royal Society of London. Series A: Mathematical, Physical and Engineering Sciences*, 452 (1954):2551–2577, 1996. doi: 10.1098/rspa.1996.0136. URL https://royalsocietypublishing.org/doi/abs/10.1098/rspa.1996.0136.

Mario Szegedy. Quantum speed-up of markov chain based algorithms. *45th Annual IEEE Symposium on Foundations of Computer Science*, 2004.

Alan Turing. On computable numbers, with an application to the entscheidungsproblem. *Proceedings of the London Mathematical Society*, s2-42(1):230–265, 1936.

Salvador Elías Venegas-Andraca. Quantum walks: a comprehensive review. *Quantum Information Processing*, 11(5):1015–1106, 2012.

J. von Neumann. First draft of a report on the edvac. *IEEE Annals of the History of Computing*, 15(4):27–75, 1993. doi: 10.1109/85.238389.

Stephen Wiesner. Conjugate coding. *SIGACT News*, 15(1):78–88, 1983. ISSN 0163-5700. doi: 10.1145/1008908.1008920. URL https://doi.org/10.1145/1008908.1008920.

Christof Zalka. Grover's quantum searching algorithm is optimal. *Physical Review A*, 60(4): 2746–2751, 1999.

A

SUPPORT MATERIAL