Hitting times of quantum random walks

Joep Claes

(4935802)

Double Bachelor Applied Physics and
Applied Mathematics
Bachelor-Thesis TWN
Technische Universiteit Delft

Delft, August 2022

Supervisor: J. Dubbeldam and A. Akhmerov

Abstract

Random walks have been applied in a many different fields for a long time. More recently, classical random walks are being used in wide variety of computer algorithms used to solve complex computational problems like 2-SAT, 3-SAT and the estimation of the volume of complex bodies. In this thesis we look into the quantum version of the familiar random walk, distinguishing between the discrete- and continuous-time quantum random walk.

Some general properties of random walks are studied throughout this thesis. First we look at the behaviour of both the classical and the random walk on a simple 1-dimensional lattice. Then, to investigate the speed and efficiency of algorithms that use quantum walks, we analyse quantum random walks in graphs, as graphs do a good job of representing the decision trees of algorithms. The graph we look at specifically is the n-dimensional hypercube.

The criterion we use to see how fast a quantum walk can traverse certain types of graph is the hitting time. The hitting time is the expected amount of time a random walk takes to reach a certain vertex in a graph for the first time. Literature has shown that classical random walks have a hitting time that scales exponentially with the dimension n of the hypercube. We find that quantum random walks offer a significant speed-up to its classical counterpart. Generally they have a polynomial hitting time, that depends on the Hamming distance between the start and sink vertex. Furthermore we find that quantum walks can have interesting properties such as a non-unitary hitting probability, meaning the quantum walker will never visit the sink vertex. Finally we see that, in some simple, symmetric cases, the hitting time of the quantum walk even is sub-linear, scaling almost like a square root, rather than a polynomial.

Table of Contents

1 Introduction

Random walks , also known as Markov chains, are, in essence, very simple concepts. These random processes describe a path that consists of a succession of random steps on some mathematical space, a graph for example. Random walks on graphs can be applied in a wide variety of fields, not only in natural science such as physical systems and mathematical modeling of life phenomena, but also in social science such as financial systems. More recently, algorithms based on random walks have shown to be immensely successful in computer science. They have been applied to many different problems such as 2-SAT , approximation of the permanent [11], estimation of the volume of convex bodies [6] and 3-SAT. Thus, it is natural to consider quantum generalization of classical random walks, which may be very useful in constructing efficient quantum algorithms, for which only a few general algorithmic tools have been developed.

So far the quantum random walk, QRW for short, has been subdivided into 2 types: the continuous- and discrete-time. The continuous-time QRW in a graph is based on the solution of a system of ordinary differential equations, which relies on the Hamiltonian of the quantum system, which, in turn, is very closely related to the adjacency matrix of the graph. The evolution of a discrete-time QRW is specified by two unitary operators: the "coin" and shift operator, which are repeatedly applied to the state of the walk. Both have many applications and are useful in different types of algorithms.

Classical and quantum random walks know some striking differences, as we will explore in this thesis. These differences are due to phenomena we know from quantum mechanics, like quantum coherence and superposition.

For many computational problems, like 3-SAT, the most efficient solution is based on the hitting time of a classical random walk [20]. Properties of a random walk like mixing and hitting time are very good criterion to analyse how fast that walk is able to traverse certain graphs. In this thesis we focus on the hitting time of the random walk in a graph, which can be seen as the average time it takes for the walk to reach a specific position in the graph, from a certain starting point or state. Literature has shown that, in many cases, classical random walks have a hitting time that scales exponentially with the size of the graph, whereas quantum random walks show polynomial (and sometimes even faster) hitting times [15][22][13].

In Chapter 2 an introduction is given on classical random walks, first on a simple 1-dimensional lattice, then in various types of graphs. We look into the applications of CRW and after which we give some background on quantum mechanics. From there we move onto the discrete- and then continuous-time quantum random walk, again, first in a simple lattice and then in graphs, the hypercube in specific. After giving definitions for the hitting time of a CRW and QRW in a graph, we are ready to see some results in Chapter 3. In this chapter we first approach the hitting time analytically. Then we use numerical calculations to compute the hitting time of both quantum walks in an n-dimensional hypercube, varying the position of the hitting and starting position of our walk. In Chapter 4 and Chapter 5 we present a discussion and conclusion of our research. Finally a link to the code used throughout this thesis can be found in Appendix A.

This research is done as a BSc thesis for the BSc Applied Physics and BSc Applied Mathematics at the faculties of Applied Sciences and Electrical Engineering,Mathematics and Computer Science at the Delft University of Technology.

2

Random Walks

Quantum random walks, which we'll call QRW from now on, behave very differently from classical random walks (CRW). To study the interesting properties of the QRW we will first analyse the CRW and then move onto the QRW and analyse the remarkable differences.

2.1. Classical Random Walks

A random walk is a random process describing a series of random steps in a certain mathematical space. They come in many different forms, from a single walker taking random steps left or right on a line to random walks in d-dimensional space. Classical random walks have been studied for a very long time and are applied in the fields of computer science, physics, chemistry, biology, economics, etc.[21]

2.1.1. Discrete Random Walk on a 1D Lattice

Imagine a person or walker, who can only take steps left or right and it does so randomly. We call the starting position 0 and for every step this walker takes, we could flip a coin, heads means it moves right, tails means a step to the left. A biased random walk is when the chance to go left, p, isn't the same as the chance to go right, $1-p$. In general, the probability mass function of the position S_n of a 1-dimensional random walker after n steps is given by the binomial distribution:

$$\mathbb{P}(S_n = d) = \binom{n}{\frac{n+d}{2}} p^{\frac{n+d}{2}} (1-p)^{\frac{n-d}{2}}. \tag{2.1}$$

Here $d = r - l$ means the distance traveled from the origin by the walker. It is equal to the number of steps taken to the right, r, minus the number of steps to the left, l, where $n = r + l$. An important property of a random walk is the variance, denoted by σ^2, which describes how much the walk disperses over a given time. To calculate the variance, we need the expected position of the walker after n time steps. Let X_t be the independent random variable that tells us which way the walker moves in a step t, meaning $\mathbb{P}(X = 1) = p$ and $\mathbb{P}(X = -1) = 1 - p$. This allows us to rewrite and eventually find the formula for the expected position after n steps, S_n.

$$\mathbb{E}(S_n) = \mathbb{E}(\sum_{t=0}^{n} X_t) = \sum_{t=0}^{n} \mathbb{E}(X_t) = n \cdot ((1)(p) + (-1)(1-p)) = n \cdot (2p-1) \tag{2.2}$$

Not the formula for the standard deviation is, classically, given by

$$\sigma^2 = \mathbb{E}(S_n^2) - \mathbb{E}(S_n)^2 = \sum_{t=0}^{n} \mathbb{E}(X_t^2) = n \cdot ((1)^2(p) + (-1)^2(1-p)) = n \tag{2.3}$$

For an unbiased walker, $p = 0.5$, logically, the probability to end up at the starting position after many steps is the largest. Simulating 100,000 classical random walkers that all take 100 unbiased steps, we see that this is indeed the case and in figure 2.1 we see the following fit with the Gaussian/normal distribution given by the probability density function:

$$f(x) = 200391 \cdot \frac{1}{10\sqrt{2\pi}} e^{-\frac{1}{2}(\frac{x}{10})^2} \tag{2.4}$$

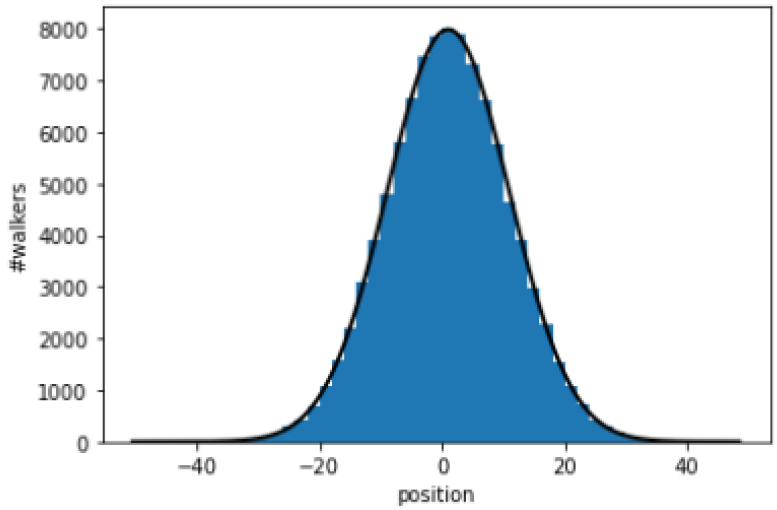

Figure 2.1: Simulation of end position of 100.000 classical random walkers that each take 100 steps in blue, fitted to the Gaussian described by 2.4 in black.

Eventually, as $n \longrightarrow \infty$, the probability distribution of a 1D random walker goes to a Gaussian distribution. In accordance to formula 2.2 and 2.3, we indeed see this distribution has an expected position of zero and a $\sigma = \sqrt{100} = 10$ (68%-rule). This is a simple model of the CRW, 1 walker moving on a 1-dimensional lattice. Classical random walks appear in many forms, however, and can be considered in many different mathematical spaces, such as higher dimensional lattices and graphs.

2.1.2. CRW in a graph

Many interesting computational problems can be reformulated in terms of a decision tree [8][2]. A natural classical algorithm is then the same as letting a classic random walk run on the tree[7]. Therefore, it is much more interesting to generalize a CRW on a 1-dimensional lattice to a walk on a graph, $G = (V, E)$, where V is the set of all vertices and E describes all edges between vertices. The *degree* of a vertex is the number of edges connected to that vertex. Now, for G we can make a so-called generator matrix which is defined as:

$$M_{ab} = \begin{cases} \gamma & a \neq b, a \text{ is connected to } b \text{ by an edge} \\ 0 & a \neq b, a \text{ and } b \text{ not connected} \\ -k\gamma & a = b, k \text{ is the degree of vertex } a \end{cases} \tag{2.5}$$

where γ is the jumprate, which, for simplicity, we will set equal to 1. Now, if $p_a(t)$ denotes the probability for the walker to be at vertex a at time t, we have [4]

$$\frac{dp_a(t)}{dt} = -\sum_b M_{ab} p_b(t). \tag{2.6}$$

This system of ordinary differential equations can be solved , which gives

$$\boldsymbol{p}(t) = \boldsymbol{p}(0) \cdot e^{-iMt}. \tag{2.7}$$

This formula describes the evolution of the CRW in a graph with generator matrix M. Using this solution we are able to model the behaviour of a random walker through different types of graphs. One specifically interesting type of graph is a connected binary tree, of order n, called G_n, as has been extensively investigated by *Childs* et al. [4]. A connected binary tree starts at a vertex, s, which is connected to 2 vertices, which are in turn both connected to 2 more vertices. After n branches that connect every vertex to 2 new vertices, the graph stops growing and starts to shrink by connecting two vertices to one new vertex until the last two vertices are connected to the last vertex t. Figure 2.2 shows an example of a connected binary tree with $n = 4$. We see vertex s, t and the vertices in the middle (or n-th) column/layer of the graph have degree two and all

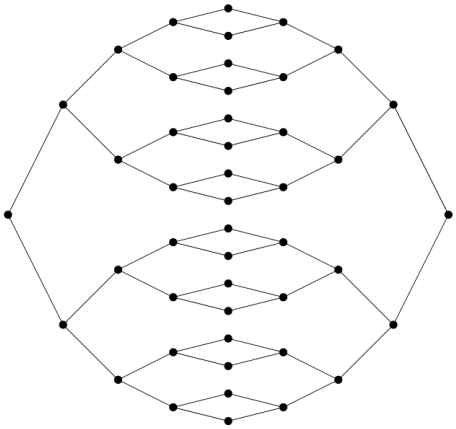

Figure 2.2: The graph G_4

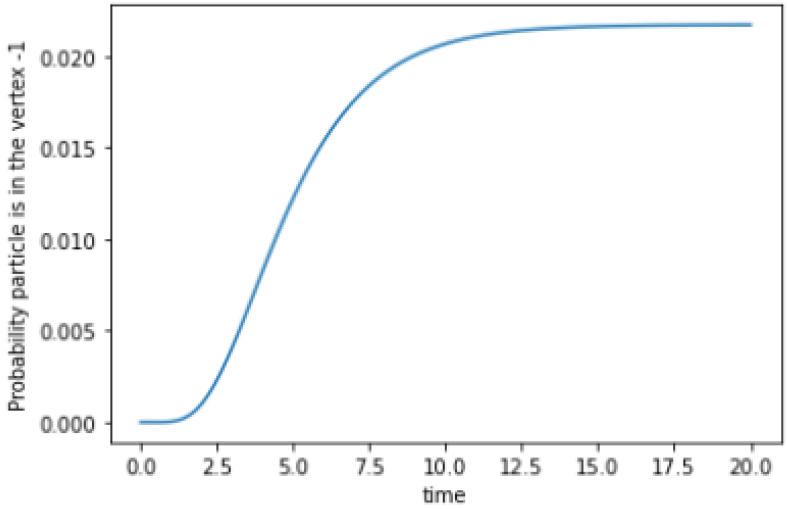

Figure 2.3: Probability of a walker, starting at vertex 0, to be at the final vertex, t, of a G_4 graph over time.

other vertices have degree three. Now, using 2.6, we can calculate and analyse how the probability of the walker being in a certain vertex of G_n after a time t changes, when it starts at vertex s on the left. Figure 2.3 shows the progression of the probability of the walker to be in vertex t (the outer vertex, furthest away from the starting vertex s) over time.

Since the walker starts at the other end of the graph, it takes some time before it reaches the outer vertex, t. After some time, the probability of the walker to be in vertex t slowly converges to an equilibrium value. This is also true for the probabilities of the walker to be in any other vertex, as can be seen in figure 2.4. We see the probability converges to a uniform distribution over all vertices over time. Also, the probability in vertices in the same columns (see 2.2) is the same and it is clear this graph can be simplified into a line of nodes, where each node represents a column of the binary tree. Adding the probability of all vertices in a column gives the total probability of the walker to be in that column. If we do this, we get the probability distribution shown in figure 2.5 after a certain time t. As can be seen the simplification of the binary tree into a line of nodes behaves very similarly. After enough time-steps the walker is (almost completely) evenly spread throughout all vertices, which means the probability to be in a certain column is a function of the number of vertices in that given column. In figure 2.2 we see that the column in the centre contains the most vertices, which is why the walker is most likely to be in that column after many time steps.

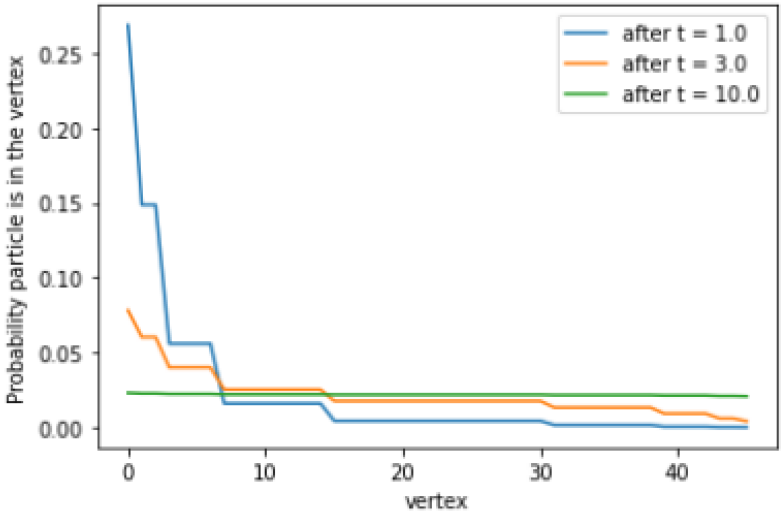

Figure 2.4: Probability of the walker, starting at vertex 0, to be in any vertex of the G_4 graph after 1, 3 and 10 time steps

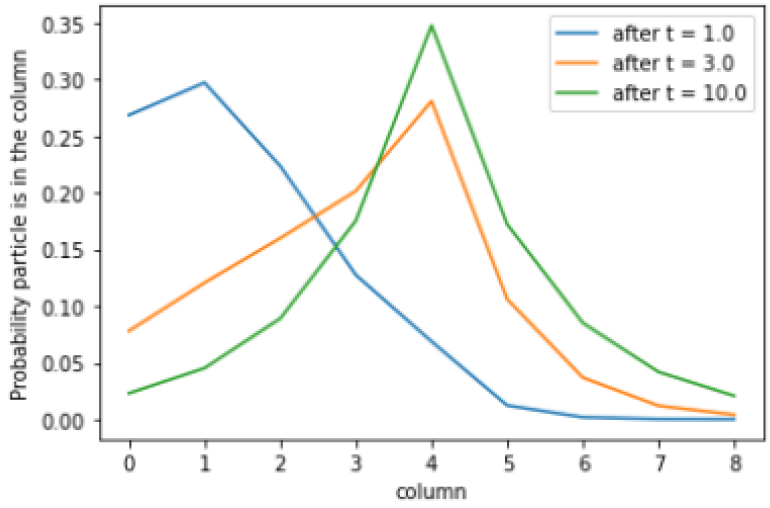

Figure 2.5: Probability of a random walker to be in every column of a G_4 graph after 1, 3 and 10 time steps

2.1.3. Applications of a CRW

Having studied how a CRW works, it is interesting to see where all this theory is applied. Many of today's algorithms work similarly to graphs, so that is where we see many random walks being used.

One famous example is the Google PageRank algorithm [18], which is the algorithm that ranks all the webpages for every Google search. It works as follows: once a search term is entered, the algorithm selects a webpage containing that term; from there a random walker goes from that page to a site linked to that page. Here the webages can be seen as the vertices and if one page links another page, it can be entrepeted as an edge between the two pages. The walker traverses between the vertices until it has visited every webpage many times. Obviously, it visits pages that are referenced a lot in other pages (and thus have a high degree) more often than others. Now, the degree of an individual vertex is just as important as the degree of the vertices connected to that vertex. If many important webpages (with a high degree) link to a certain webpage, this must mean this page matches very well to the search term. This way the algorithm is able to rank the importance and relevance of different pages for a single search term.

Furthermore, CRW are widely used in all types of real life models. It is a great way to simulate the "random" behaviour of a liquid or gas, but also in biology, ecology and medicine [5].

2.2. Quantum Mechanics

In classical mechanics we are used to think of particles in terms of their mass m and position $x(t)$. From there we can calculate any dynamical variable of interest. To find $x(t)$ we apply Newton's second law: $F = ma$. In quantum mechanics we don't look at particles using these properties, instead we use a more probabilistic model.[9]

2.2.1. The Wavefunction and Schrödinger Equation

In the quantum world a particle is described by its wave function, $\Psi(x,t)$ and we get it by solving the Schrödinger equation:

$$i\hbar\frac{\partial\Psi}{\partial t} = -\frac{\hbar^2}{2m}\frac{\partial^2\Psi}{\partial x^2} + V\Psi \tag{2.8}$$

The wavefunction doesn't describe the exact position of a particles, instead is gives us information about the probability of finding the particle in a certain position x at a specific time t. The probability of finding a particle, which is located on a line, between position a and b at time t, $\mathbb{P}([a,b],t)$, is given by

$$\mathbb{P}([a,b],t) = \int_a^b |\Psi(x,t)|^2 dx \tag{2.9}$$

Since the wavefunction describes a probability, normalization tells us that as $a \to -\infty$ and $b \to \infty$ the total probability must equal 1, therefore all solutions of the Schrödinger equation (2.8) must satisfy this requirement. This statistical interpretation means that at any t, the exact position of the particle is unknown until it is measured. When a measurement of the particle is done, we say its wavefunction "collapses". Later on we will see the collapsing of the wavefunction plays a vital role in the way the quantum random walk differs from its classical counterpart.

2.2.2. Hamiltonian

Now that we have learned about the wavefunction and the Schrödinger equation (2.8), it is time to solve it for a specified potential $V(x,t)$. To make this a bit easier we will assume that the potential only depends on the position, so $V(x,t) = V(x)$. Then, using separation of variables, we are able to find the time-independent Schrödinger equation:

$$-\frac{\hbar^2}{2m}\frac{d^2\psi(x)}{dx^2} + V\psi(x) = E\psi(x) \tag{2.10}$$

which can be rewritten as,

$$\hat{H}\psi = E\psi \tag{2.11}$$

where $\hat{H} = \frac{\hat{p}^2}{2m} + V(x)$ is the Hamiltonian operator, which describes the total energy of the particle, and $\hat{p} = -i\hbar(\frac{\partial}{\partial x})$ is the momentum operator. The solutions to this time-independent SE, $\psi_E(x)$, correspond to

eigenstates of the full SE (2.8). These solutions allow us to express the solution to the full SE as:

$$\Psi_E(x,t) = \psi_E(x)e^{-iEt/\hbar} \tag{2.12}$$

Since these eigenstates are all orthonormal and complete and the operator $U = e^{-iEt/\hbar}$ is unitary, the probability density $|\Psi(x,t)|^2 = |\psi(x)|^2$ is also unitary (total probability is equal to 1) and therefore not time-dependent.

2.2.3. Hilbert space

Similar to ordinary vectors in two dimensions, which are described in Euclidean space, in quantum mechanics we describe a state using a vector, $|S(t)\rangle$, in Hilbert space. In Euclidean space a vector can be described using different bases (Cartesian or polar coordinates, etc.) and the same can be done for a vector in Hilbert space. The wave function $\Psi(x,t)$ is actually the x "component" in the expansion of $|S(t)\rangle$ in the basis of position eigenfunctions:

$$\Psi(x,t) = \langle x|S(t)\rangle \tag{2.13}$$

with $|x\rangle$ standing for the eigenfunction of $\hat{x}$ with eigenvalue x [9]. Similarly, the vector can be described using the momentum space wave function or the energy coefficients using the expansions of $|S(t)\rangle$ in the basis of momentum and energy eigenfunctions resp.

2.2.4. Dirac notation

The previous subsection introduces a new form of notation for the inner-product, called the Dirac or bra-ket notation. In quantum mechanics, a particle has many different possible states. These different states can be represented by a linear combination of unit vectors in a complex Hilbert space. As we have seen, a certain state can be denoted as $|\Psi\rangle$. The $|.\rangle$ notation is called a *ket* and represents a column vector. In this thesis we will mostly describe $|\Psi\rangle$ as a linear combination of the n orthogonal basis vectors for a complex Hilbert space, $|\psi_1\rangle, ..., |\psi_n\rangle$, such that

$$|\Psi\rangle = c_1|\psi_1\rangle + c_2|\psi_2\rangle + ... + c_n|\psi_n\rangle \tag{2.14}$$

To satisfy the unitary law of quantum mechanics it is necessary that $\sum_{i=1}^{n} |c_i|^2 = 1$. The Hermitian conjugate of the *ket* is called the *bra* and is denoted as $\langle .|$ and describes a linear map or a row vector. The inner-product between a bra and ket is written as $\langle .|.\rangle$

2.2.5. Spin

In classical mechanics, an object, like the earth, has two types of angular momentum: orbital, which is attributed to its motion around the sun, and spin, which is caused by its rotation *about* its own centre of mass. In quantum mechanics an electron also has orbital angular momentum due to its rotation around the nucleus, but an electron is a structureless point so it doesn't have the same type of spin as the earth. Rather, elementary particles carry an *intrinsic* angular momentum, which we call spin.

Different types of particles carry different spins: π mesons have spin 0; photons have spin 1; electrons have spin $1/2$ etc. In this thesis we will mainly look at walkers that posses many of the properties of $1/2$-spin particles (like electrons), which have two eigenstates $|\frac{1}{2}\frac{1}{2}\rangle$, which we call spin *up* and $|\frac{1}{2}(-\frac{1}{2})\rangle$, which we call spin *down*. Informally we can write these states as $|\uparrow\rangle$ and $|\downarrow\rangle$ resp. Another way of denoting these eigenstates is using *spinors*. Then, the general state of a spin-$1/2$ particle can be represented by a two-element column matrix (a spinor):

$$\chi = \begin{pmatrix} a \\ b \end{pmatrix} = a\chi_+ + b\chi_- \tag{2.15}$$

with $\chi_+ = \begin{pmatrix} 1 \\ 0 \end{pmatrix}$ representing spin up and $\chi_- = \begin{pmatrix} 0 \\ 1 \end{pmatrix}$ for spin down. Using this notation we can denote the intrinsic angular momentum ($\mathbf{S}$) as $\mathbf{S} = (\hbar/2)\boldsymbol{\sigma}$ (the $\hbar$ is often left out for simplicity), where

$$\sigma_x = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}, \sigma_y = \begin{pmatrix} 0 & -i \\ i & 0 \end{pmatrix}, \sigma_z = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}, \tag{2.16}$$

These are the famous *Pauli spin matrices* and allow us to measure the spin when being multiplied by the spinor notation of the two eigenstates.

2.2.6. Qubits

In this thesis, however, we will write spin up as $|0\rangle$ and spin down as $|1\rangle$. As we know, in quantum mechanics, a particle can be in a superposition of their two eigenstates, written as

$$|\psi\rangle = \alpha|0\rangle + \beta|1\rangle \tag{2.17}$$

where $|\alpha|^2$ is the probability to find the particle in state $|0\rangle$ and the same goes for β, so $|\alpha|^2 + |\beta|^2 = 1$.

The particle that we have described in (2.17) can be seen as a superposition between two classical computer bits, which can have value 0 or 1. This means an electron could act like a regular bit, but rather than only taking on the value 1 or 0, it can have an infinite number of superpositions of the two states. Due to the similarity to classical computer bits, these particles are referred to as quantum bits or *qubits*. It turns out that this property allows for much faster computing when these qubits are used in quantum computers[10].

2.3. Quantum Random Walks

Now that we have briefly looked at the CRW and discussed some quantum mechanics formalities, we can now move onto the Quantum Random Walk. QRWs are first proposed by Aharonov et al. [1] in 1993 and they can be regarded as the counterpart of classical random walks in quantum mechanics. The main difference between a CRW and QRW is that quantum walks don't converge to some limiting distributions. They can spread significantly faster or slower than classical random walks because of quantum interference [21].

2.3.1. Discrete QRW

In order to analyse a discrete QRW walk on a 1-dimensional lattice, we need to define the Hilbert space, $\mathscr{H}_P$, spanned by the positions of the particle, so in this case $\mathscr{H}_P = \{|i\rangle : i \in \mathbb{Z}\}$. This will act as our lattice. The states in this basis will make up the wave function $|\psi\rangle$, with $|i\rangle$ corresponding to a particle localized in position *i*. In addition to the position state space we will need another Hilbert space, $\mathscr{H}_C$, to simulate a the coin-flip (or dice roll for dimensions higher that 1) that determines which way the walker will travel. We call this the *coin-state space* and it is spanned by two basis states $\{|\uparrow\rangle, |\downarrow\rangle\}$, which represent the spin up and down mentioned in the previous section. Thus, states in the total system are in the space $\mathscr{H} = \mathscr{H}_C \otimes \mathscr{H}_P$.

The first step of the quantum random walk is a rotation in the coin-space, which we will refer to as the "coin-flip" C. Ideally, C is unitary and unbiased, meaning that if the walk is in a localized state $|0\rangle$, it will go right (to position $|1\rangle$) with probability $\frac{1}{2}$ and left (to $|-1\rangle$) with probability $\frac{1}{2}$. A frequently used unitary and balanced coin is the Hadamard coin H [12]

$$H = \frac{1}{\sqrt{2}}\begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}. \tag{2.18}$$

When the walker is at position $|i\rangle$ on the lattice it can only move to either $|i-1\rangle$ or $|i+1\rangle$. The conditional translation of the system can now be described by the unitary shift operation

$$S = |\uparrow\rangle\langle\uparrow| \otimes \sum_i |i\rangle\langle i+1| + |\downarrow\rangle\langle\downarrow| \otimes \sum_i |i-1\rangle\langle i| \tag{2.19}$$

where the index i runs over $\mathbb{Z}$. Basically S transforms the basis state $|\uparrow\rangle \otimes |i\rangle$ to $|\uparrow\rangle \otimes |i+1\rangle$ and $|\downarrow\rangle \otimes |i\rangle$ to $|\downarrow\rangle \otimes |i-1\rangle$. This operation, in combination with the Hadamard coin, allows us to simulate the walk.

Measuring the walk after every iteration means there is no correlation between different positions left and doing this at every step results in a classic random walk as described in section (2.1.1). However, we will of course not measure the coin register during intermediate iterations, but rather keep the quantum correlations between different positions and let them interfere in subsequent steps. This interference is what causes the quantum random walk to behave so radically different from its classical counterpart.

A discrete QRW of N steps is defined as the transformation U^N, where U, which acts on our Hilbert space $\mathscr{H} = \mathscr{H}_C \otimes \mathscr{H}_P$, is the unitary operator that combines the coin-flip C and positional shift S:

$$U = S \cdot (C \otimes I) \tag{2.20}$$

where for C we will choose the Hadamard coin H. Using this operator U we can now produce a N-step quantum random walk. If we let the walk start in a symmetric initial state, $|\psi_{in}\rangle = \frac{1}{\sqrt{2}}(|\uparrow\rangle + i|\downarrow\rangle) \otimes |0\rangle$, we get the probability distribution seen in figure 2.6 after N=100 steps.

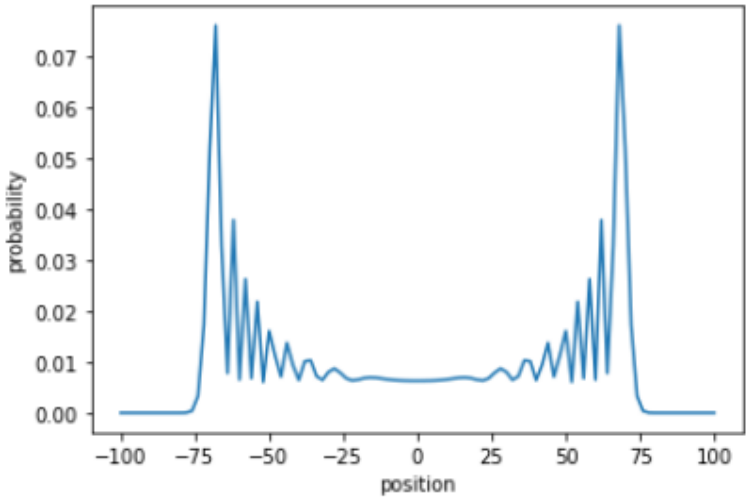

Figure 2.6: Probability distribution of a discrete quantum random walk after 100 steps, starting in a symmetric state.

Now that we have seen how a QRW behaves on an infinite 1-dimensional lattice, we can look into what happens if we added a wall of some sort. First we insert a completely absorbing wall (also know as a *sink*) at position $|20\rangle$, meaning absorbing boundary conditions are added to our model. This BC sets the amplitude of the wavefunction at position $|20\rangle$ to zero after every iteration of the walk. It does so by checking the amplitude of the wavefunction at the position of the wall and setting it back to zero. We know that measurement of a quantum system causes the wavefunction to "collapse", however since this measurement is only partial (we only measure at the position of the wall) the quantum coherence is conserved for the most part. Now, since the wall absorbs all of the "probability" that hits the wall, the walk is no longer unitary. Figure 2.7 shows the resulting probability distribution.

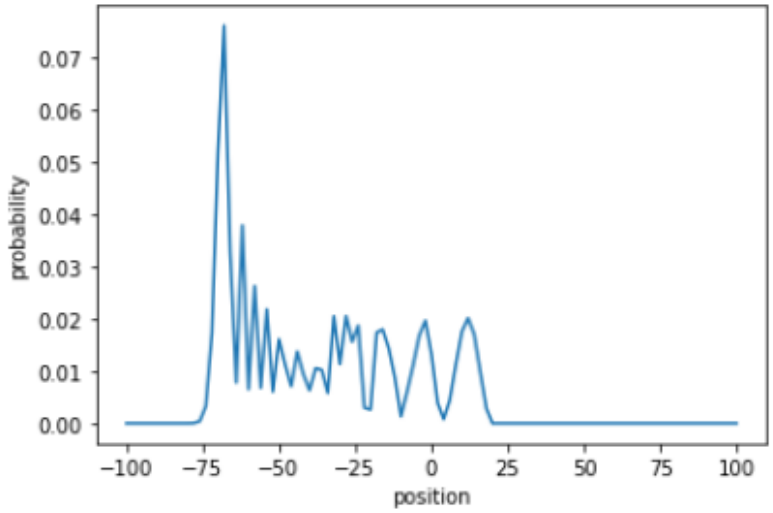

Figure 2.7: Probability distribution of a QRW with an absorbing wall at position $|20\rangle$.

We see figure 2.7 is not the same as 2.6 with all probabilities after position $|20\rangle$ set to zero. This is because the sink at our wall sets the amplitude of the wavefunction to zero at every iteration, which changes the interference that will occur at the next iteration, compared to our QRW without wall. Now, if we want the walk

to remain unitary, the wall needs to be reflective, rather than absorbing. As mentioned before, when applying the coin operator, the phase is rotated in coin space. A reflection of the walker is nothing more than applying a rotation of $\pi/2$ to the phase, when the rotation matrix for this rotation is given by

$$R(\theta) = \begin{pmatrix} \cos\theta & \sin\theta \\ \sin\theta & -\cos\theta \end{pmatrix} \tag{2.21}$$

Clearly, inserting $\theta = \frac{\pi}{4}$ results in the Hadamard coin described in formula 2.18 and $\theta = \frac{\pi}{2}$ completely flips the phase. However, θ can also take on intermediary values, $\frac{3\pi}{8}$ for example. This means the walk will only be partially reflected. Adding a completely or partially reflective wall results in the following different distributions:

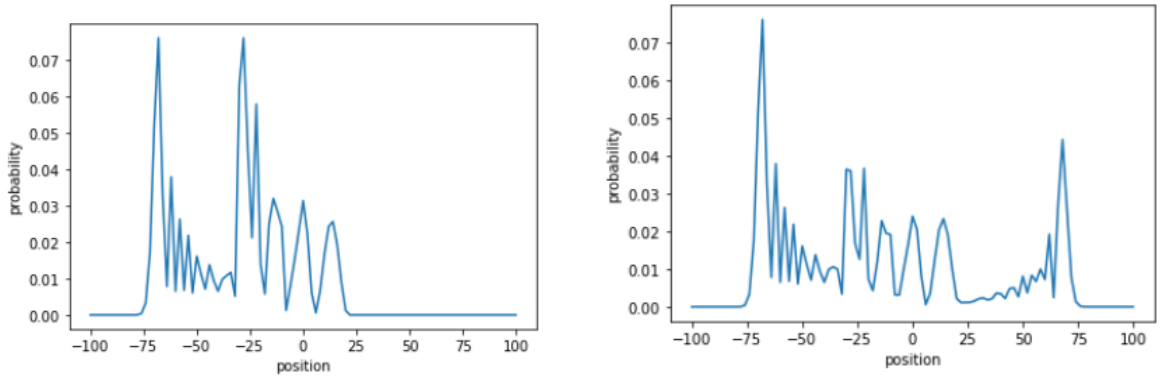

Figure 2.8: Probability distribution of a QRW with a completely reflective wall at position $|20\rangle$ (left) and a partially reflective wall, with a phase rotation of $\pi/4$, at position $|20\rangle$ (right).

Now that we have seen the behaviour of a Discrete Quantum Random Walk with different kinds of boundary conditions, we can start to expand our theory.

2.3.2. Continuous QRW

In section 2.1.2. formula 2.6 is an ordinary differential equation describing the movement of a continuous classic random walk. The quantum counterpart of this formula, which therefore describes a continuous quantum random walk, looks as follows:

$$i\frac{d}{dt}\langle a|\psi(t)\rangle = \sum_b \langle a|H|b\rangle\langle b|\psi(t)\rangle \tag{2.22}$$

This is simply the Schrödinger equation rewritten for quantum evolution in a v-dimensional Hilbert space for a Hamiltonian H, in a basis $|1\rangle, |2\rangle, ..., |v\rangle$. This QRW is given by the quantum Hamiltonian with matrix elements equal to the elements of the adjacency matrix of the graph through which the QRW propagates[7]

$$\langle a|H|b\rangle = M_{ab} \tag{2.23}$$

with M_{ab} as described by 2.5. To analyze the continuous QRW through any graph with an adjacency matrix, M, we simply need to solve the system of ordinary equations given by:

$$\dot{\psi}(t) = -iM \cdot \psi(t) \tag{2.24}$$

where ψ is the vector containing the amplitude of the wavefunction in every position $|1\rangle, |2\rangle, ..., |v\rangle$. Solving this differential equation is straightforward and gives the expression for the QRW in any graph or lattice, which is:

$$\psi(t) = e^{-iMt} \cdot \psi(0) \tag{2.25}$$

2.3.3. QRW in a graph

Given that the Hamiltonian is equal to the adjacency matrix of the space in which the quantum particle walks, the QRW can be extended to all sorts of graphs. Using formula 2.5 to make the generator matrix, the quantum

random walk can be simulated in a binary tree, by inserting it into the solution 2.25. This gives insights to some very interesting characteristics of the QRW in graphs. Plotting the probability of the wavefunction to be in the last vertex t, over time, like in figure 2.3, the differences are remarkable, as can be seen in figure 2.9.

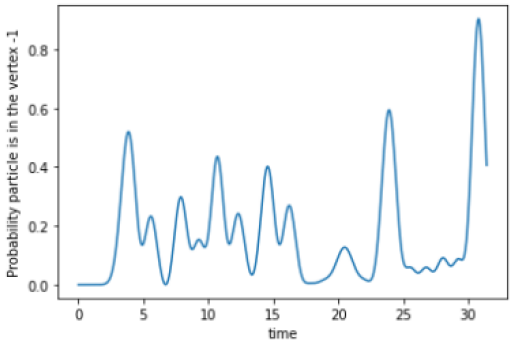

Figure 2.9: Probability of a quantum random walker, starting at vertex s, to be at vertex t of a G_4 graph over time.

The first thing that stands out is that, rather than converging to an equilibrium value, the probability seems to fluctuate aperiodically. Furthermore, at $t = 30.7$ the probability in the outer vertex t reaches a peak value of 0.904. Since the initial value of the walk was to start at vertex 0 with probability 1, this means that after this time t the wavefunction has magically almost completely "reassembled" in the outer vertex due to constructive interference. So the odds of finding our quantum random walker at this point at this time are very high.

It turns out that other graphs show similar results. A very interesting case is that of the n-dimensional hypercube. Most of us are familiar with the 1-, 2- and 3-dimensional hypercube (a line, square and cube resp.). The dimension of a hypercube describes the amount of (binary) bits in its vertices. To illustrate the figure below shows a 4-dimensional hypercube, with all its corresponding vertices.

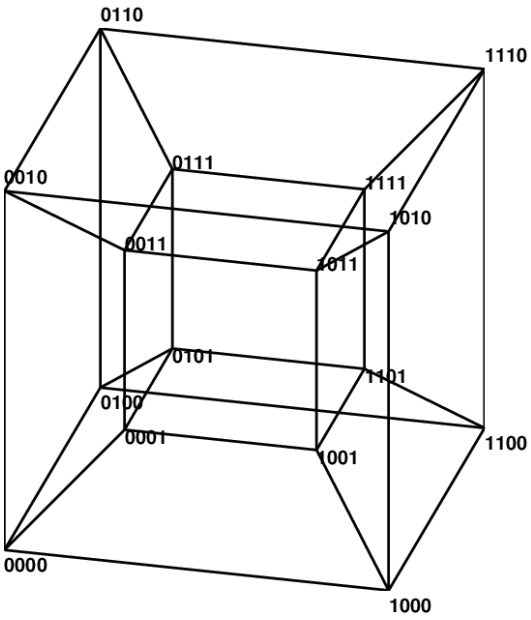

Figure 2.10: A 4-dimensional hypercube with all its vertices labeled using binary bit notation.

As can be seen in figure 2.10, two vertices are connected if they differ in only 1 bit. Therefore the adjacency matrix, and thus the Hamiltonian (remember formula 2.23), of an n-dimensional hypercube can simply be

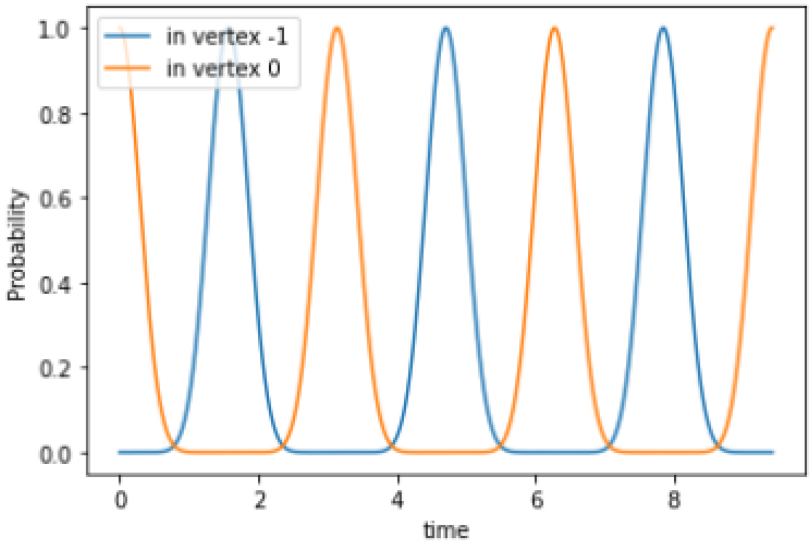

Figure 2.11: The probability of a quantum random walker to be in the first (orange) and last (blue) vertex of a 6-dimensional hypercube, over time.

written as the $2^n \times 2^n$-matrix [17]:

$$H(\vec{x},\vec{y}) = \begin{cases} 1 & d(\vec{x},\vec{y}) = 1 \\ 0 & d(\vec{x},\vec{y}) > 1 \\ -n & \vec{x} = \vec{y} \end{cases} \tag{2.26}$$

where $d(\vec{x},\vec{y})$ is the Hamming distance between two vertices. Now, plotting the evolution of a continuous quantum random walk through a hypercube, using formula **??**, sheds light on some interesting behaviour. The initial condition of the walker is set at probability 1 to be in vertex s = (000...00) at time $t = 0$. First, the probability in the last vertex t = (111111) of a 6-dimensional hypercube is plotted over time and compared to the probability in the first vertex (000000), as seen in figure 2.11.

We notice the walker has a period of π when traversing a 6-dimensional hypercube. To better understand the nature of this behaviour, we first simplify the graph of the hypercube by representing each column of vertices with a single node (as in section 2.1.2). The vertices in each column all have the same Hamming weight. The Hamming weight of a vertex is equal to the number of 1's that vertex contains in its bit-string notation. This means that at the very left is our starting vertex s = (000...00), which is the only vertex with a Hamming weight of zero. Next to that column is the column containing all the vertices with weight one ((100...00),(010..00),(001...00), etc.) and all the way at the right is our end vertex t = (111...11). Moreover, we see that after $t = \frac{\pi}{2}n$, for any odd n, the probability of the walker to be in the last vertex is equal to 1. So, after starting in vertex 0, at these times the wavefunction has completely reassembled in the last vertex due to the interference property of the QRW. Figure **??**shows the evolution of the walk between $t = 0$ and $t = \frac{\pi}{2}$. In this figure the green line is the probability distribution across the columns after $t = \frac{\pi}{4}$. The symmetry resembles that of the classic random walk very closely, as can be seen when comparing figure **??** to 2.5. It turns out to be the case that a quantum random walker traversing a hypercube reaches an almost exact uniform distribution across all vertices at time $t = \frac{\pi}{4}n$, for any odd n, as can be seen in figure 2.13.

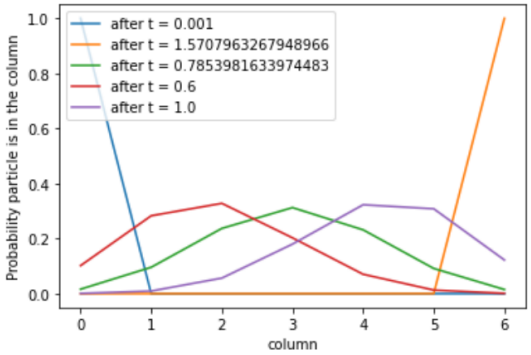

Figure 2.12: The probability distribution of a quantum random walker in the columns of a 6-dimensional hypercube after different times.

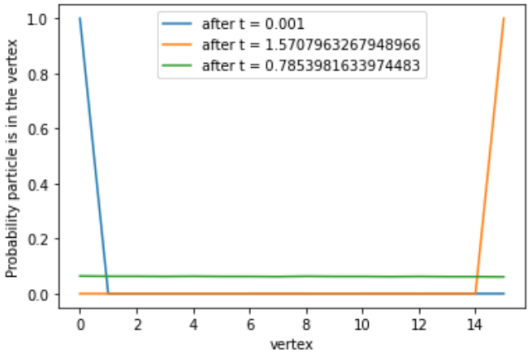

Figure 2.13: The probability of a quantum random walk in every vertex of a 4-dimensional hypercube after $t \approx 0$, $t = \frac{\pi}{4}$ and $t = \frac{\pi}{2}$.

Now the question arises, where does this mysterious periodicity of π come from. The answer lies in the Hamiltonian of the system. Looking at the most simple case, the 1-dimensional hypercube (figure 2.14), it is very clear.

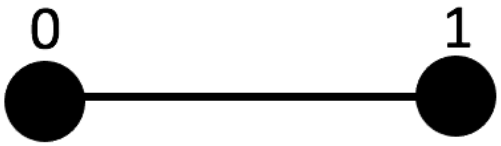

Figure 2.14: A 1-dimensional hypercube with the two vertices {0,1}.

As we know from 2.26, the Hamiltonian of this system is

$$H = \begin{pmatrix} -1 & 1 \\ 1 & -1 \end{pmatrix} \tag{2.27}$$

Diagonalizing this matrix allows us to calculate our unitary time evolution operator:

$$U(t) = e^{-iHt} = \begin{pmatrix} 1 & -1 \\ 1 & 1 \end{pmatrix} \begin{pmatrix} 1 & 0 \\ 0 & e^{2it} \end{pmatrix} \begin{pmatrix} 1/2 & 1/2 \\ -1/2 & 1/2 \end{pmatrix} = e^{it} \begin{pmatrix} \cos t & -i\sin t \\ -i\sin t & \cos t \end{pmatrix} \tag{2.28}$$

This means that in this graph the continuous-time quantum walk has a time-dependent amplitude vector given by [14]

$$|\psi(t)\rangle = e^{-iHt}|0\rangle = e^{it} \begin{pmatrix} \cos(t) \\ -i\sin(t) \end{pmatrix} \tag{2.29}$$

which implies that $\mathbb{P}(t) = \langle\psi(t)|\psi^*(t)\rangle = (\cos^2(t)\ \sin^2(t))^T$. Clearly, this system starts at vertex $(1\,0)^T$ at $t = 0$ and completely transfers to the other vertex $(0\,1)^T$ at $t = \pi/2$. For hypercubes of higher dimensions it becomes very cumbersome to show by hand as their size scales with 2^n. To generalize our theory, we first note that adding terms to the diagonal of the Hamiltonian does not change the behaviour of our walk. Adding a constant c, to the diagonal simply results in the new time evolution operator $U_{new(t)} = e^{-ict}U(t)$, which obviously does not change the probability (2.28 shows an example for $c = -1$). Setting the diagonal of the Hamiltonian for an n-dimensional hypercube equal to zero means we can rewrite 2.26 as [17]

$$H = \sum_{j=1}^{n} \mathbf{1} \otimes ... \otimes \sigma_x \otimes ... \otimes \mathbf{1} \tag{2.30}$$

where σ_x is the Pauli matrix described by 2.16 and where the jth term in the sum means σ_x appears in the jth place in the tensor product. Now, we use and we can say

$$U = e^{-iHt} = \prod_{j=1}^{n} \mathbf{1} \otimes ... \otimes e^{-it\sigma_x} \otimes ... \otimes \mathbf{1} = [e^{it\sigma_x}]^{\otimes n} = \begin{pmatrix} \cos t & -i\sin t \\ -i\sin t & \cos t \end{pmatrix}^{\otimes n} \tag{2.31}$$

where $A^{\otimes n}$ is the tensor product of n copies of A. Plugging this into formula 2.25 with starting position $\psi_0 = |0...0\rangle = |0\rangle^{\otimes n}$, gives

$$\psi_t = U_t\psi_0 = [(\cos t)|0\rangle + (i\sin t)|1\rangle]^{\otimes n} \tag{2.32}$$

This means that the amplitude of the wavefunction of the QRW at vertex v with Hamming weight d is

$$\psi_t(v) = (\cos t)^{n-d}(i\sin t)^d \tag{2.33}$$

This explains the observed oscillatory behaviour of the quantum walk in an n-dimensional hypercube. Evidently, when looking at the corner-to-corner hitting time the last vertex t has Hamming weight n and plugging this into 2.33 we see the same behaviour as we found in figure 2.11.

2.3.4. Hitting Times

To study the speed at which a random walker can traverse a certain graph we need to look at the *hitting time*. Classically, the hitting time is the average or expected time for a walker to reach a certain vertex, say v, for the first time. For simplicity reasons we will call the vertex, s, at which the walk starts, the start vertex and the vertex, t, at which we measure the hitting time, the end vertex. To investigate the first time the end vertex is visited we use a *sink*, like in figure 2.7, which sets the probability of the walker to be at that vertex equal to zero after every time step. In the classical case we do this be setting all entries to the matrix given by 2.5 equal to zero. This causes the probability that the walker stays in the system or graph, p_{stay}, to not be unitary. The hitting time can be seen as the expected time for the particle to escape the system. Analytically the expected value of a random variable X can be written as:

$$\mathbb{E}(X) = \int_{-\infty}^{\infty} x \cdot f(x)dx \tag{2.34}$$

where $f(x)$ is the probability distribution function of X. In the case of our random walk, X is the time it takes the walker to leave the system (or hit the end vertex) and $T = \mathbb{E}(X)$ is the hitting time. Therefore, we know the hitting time, T, is given by the formula

$$T = \sum_{t=0}^{\infty} t \cdot p_{esc}(t) \tag{2.35}$$

where $p_{stay}(t)$ is the probability distribution function that describes the probability the walker escapes from the system between time t and $t + dt$ (and thus 'hits'). Now, logically, we can say

$$p_{esc}(t) = P_{stay}(t) - P_{stay}(t + dt) \approx dP_{stay}(t) \tag{2.36}$$

where $P_{stay}(t)$ is the cumulative distribution function describing the probability the walker has stayed in the system until a time t. This can easily be calculated by summing over all the "remaining probability" left in the system (the rest has disappeared into the sink). Now, combining 2.35 and 2.36, we get the formula that can be used to calculate the average hitting time of our walker:

$$T = \sum_{t=0}^{\infty} t \cdot dP_{stay}(t) \tag{2.37}$$

which sums over t from zero to infinity in small steps of dt.

2.3.4.1 Hitting time in a QRW

In a classical random walk, measurement does not interfere with the evolution of the walk, however, as we know, quantum random walks behave differently. In a quantum system, measurement of any kind (e.g. measuring the position of the walker) causes the walk to lose its quantum coherence and the wavefunction to collapse [13]. If we were do measure the QRW at every iteration the walk would reduce to the standard CRW, in which case we cannot expect any non-classical behaviour or speed-ups. To get out of this dilemma we need to redefine the "quantum hitting time".

One definition is known as the "one-shot p hitting time" and allows the walk to evolve without being observed at all. After a (previously selected) time T, the position of the walk is measured. If the probability p to be at the end vertex v is larger or equal to a predetermined threshold probability, p_{th}, T is said to be our hitting time". In most cases the threshold probability is an inverse polynomial in the graph size [13].

Another way to determine the hitting time of a QRW is by using "concurrent measurements". Rather than measuring the position of the walk at every iteration, only the question "Is the position v or not v?" is asked, where v is the end vertex. This measurement perturbs the walk slightly but does not kill all quantum coherence at once. If the answer is that the walk has indeed hit the end vertex, the walk is stopped, otherwise this concurrent measurement is repeated after the next step. Say after a time T the probability p is bounded below by our threshold probability p_{th}, from the previous paragraph, we call T the "concurrent p hitting time". The partial measurement is described by the two projectors $\Pi_0 = |v\rangle\langle v|$ and $\Pi_1 = \mathbf{1} - \Pi_0$ (with end vertex $|v\rangle$).

In section 2.3.3 we took notice of the oscillatory behaviour of the QRW in a hypercube. This means that at times $t = n \cdot \frac{\pi}{2}$ the wavefunction has amplitude 1 (and thus probability 1) in the end vertex. Therefore the "one-shot p hitting time" is not a very suitable way to define the hitting time of a quantum random walk in this case. This is why the method involving concurrent measurements is much more suitable for analysing the hitting time of quantum walks in hypercubes. Now the question arises how often we should preform the partial measurement. When preformed too frequently, any type of measurement (partial or not) gives rise to a phenomenon called the Quantum Zeno Effect [3]. It states that frequent measurement can slow down the time evolution of a quantum walk. This can be seen as measuring a system, in its known initial state, so often, that you can freeze the evolution of the system. Therefore, using the concurrent measurement method would increase the hitting time. How much the hitting time is increased depends on the frequency of the partial measurement.

In this thesis, however, our main objective is to compare the hitting times in a QRW to that of its classical counterpart. To make sure this comparison is as close and as fair as possible we try to measure the hitting time in both cases in the same way. That is why, in this thesis, we opt for using a sink to measure the hitting time of the quantum walk in an n-dimensional hypercube. This allows us to derive a formula for the quantum hitting time in the same way as done for the classic hitting time. Simply integrating 2.37 by parts gives us the formula for the quantum hitting time, T:

$$T = \int_0^\infty |\psi_{stay}(t)|^2 dt \tag{2.38}$$

where $|\psi(t)|^2$ is the probability the walker is still in the system. In the classical case we implement a sink by setting the probability in the "sink vertex" equal to zero after every time step. In a quantum system this can be done by introducing a complex potential at the sink vertex. We do this by adding a negative imaginary term, $i\epsilon$, to our Hamiltonian matrix. By analyzing the simple case of a 2-dimensional hypercube we are able to see what adding this term changes for the evolution of the walk. A 2-dimensional hypercube has four vertices, $|00\rangle, |01\rangle, |10\rangle, |11\rangle$ and the Hamiltonian matrix is

$$H = \begin{pmatrix} -2 & 1 & 1 & 0 \\ 1 & -2 & 0 & 1 \\ 1 & 0 & -2 & 1 \\ 0 & 1 & 1 & -2 \end{pmatrix}. \tag{2.39}$$

For this matrix the eigenvalues and eigenvectors can easily be calculated (see Appendix B) and we find the following eigenvectors:

$$v_1 = \begin{pmatrix} 1 \\ -1 \\ -1 \\ 1 \end{pmatrix}, v_2 = \begin{pmatrix} 1 \\ 1 \\ 1 \\ 1 \end{pmatrix}, v_3 = \begin{pmatrix} 1 \\ 0 \\ 0 \\ 1 \end{pmatrix}, v_4 = \begin{pmatrix} 0 \\ 1 \\ -1 \\ 0 \end{pmatrix} \tag{2.40}$$

with correspond eigenvalues $\lambda_1 = -4$, $\lambda_2 = 0$, $\lambda_3 = -2$, $\lambda_4 = -2$ respectively. Since we know our walker starts of in the state $(1,0,0,0)^T$ we can write the time-dependent amplitude vector as

$$|\psi(t)\rangle = \frac{1}{4}e^{-4it}\begin{pmatrix} 1 \\ -1 \\ -1 \\ 1 \end{pmatrix} + \frac{1}{4}\begin{pmatrix} 1 \\ 1 \\ 1 \\ 1 \end{pmatrix} + \frac{1}{2}e^{-2it}\begin{pmatrix} -1 \\ 0 \\ 0 \\ 1 \end{pmatrix} + 0 \cdot e^{-2it}\begin{pmatrix} 0 \\ -1 \\ 1 \\ 0 \end{pmatrix} \tag{2.41}$$

We know from our analysis of the QRW in n-dimensional hypercubes in this case (without sink), that this walker will traverse back and forth and, therefore, have an "infinite" hitting time. Now we add our complex

ϵ	λ_1	λ_2	λ_3	λ_4
0.5	-4.0 + 0.12i	-0.02 + 1.2i	-2 + 0.25i	-2
1	-3.9 + 0.23i	-0.07 + 0.23i	-2 + 0.53i	-2
2	-3.7 + 0.35i	-0.27 + 0.35i	-2 + 1.3i	-2
5	-3.5 + 0.20i	-0.54 + 0.20i	-2 + 4.6i	-2
10	-3.4 + 0.10i	-0.57 + 0.10i	-2 + 9.8i	-2
20	-3.4 + 0.05i	-0.58 + 0.05i	-2 + 19.9i	-2
100	-3.4 + 0.01i	-0.58 + 0.01i	-2 + 99.9i	-2

Table 2.1

potential, which results in the new Hamiltonian:

$$H = \begin{pmatrix} -2 & 1 & 1 & 0 \\ 1 & -2 & 0 & 1 \\ 1 & 0 & -2 & 1 \\ 0 & 1 & 1 & -2+i\epsilon \end{pmatrix}. \tag{2.42}$$

Immediately we see that as $\epsilon \to 0$ the hitting time will grow infinitely large (almost as if there was no potential). In order to be able to rewrite the evolution of our walk as a function of time like in 2.46, we need to calculate the eigenvalues and eigenvectors of our new Hamiltonian 2.42. This is much more cumbersome, as setting $\det(H - \lambda I) = 0$ results in a cubic equation. Using the mathematical software of WolframAlpha we find the eigenvalues for different strengths of the potential ϵ shown in table 2.1. We see the real parts of the eigenvalues are quite similar to the eigenvalues of B.1, with λ_4 still completely the same as before. The eigenvector corresponding to this eigenvalue also turns out to be these same as when there is no potential present, namely $v_4 = (0,1,-1,0)^T$. Now, for the other eigenvalues, we see that as $\epsilon \to \infty$ our eigenvalues becomes approximately equal to: $\lambda_1 = -3.412 + \frac{i}{\epsilon}$, $\lambda_2 = -0.586 + \frac{i}{\epsilon}$, $\lambda_3 = -2 + i\epsilon$, $\lambda_4 = -2$. If we let ϵ approach ∞ we get the following eigenvectors respectively:

$$v_1 = \begin{pmatrix} \frac{1}{2} + \frac{\epsilon}{\sqrt{2}}i \\ -\frac{1}{\sqrt{2}} - \frac{\epsilon}{2}i \\ -\frac{1}{\sqrt{2}} - \frac{\epsilon}{2}i \\ 1 \end{pmatrix}, v_2 = \begin{pmatrix} \frac{1}{2} - \frac{\epsilon}{\sqrt{2}}i \\ \frac{1}{\sqrt{2}} - \frac{\epsilon}{2}i \\ \frac{1}{\sqrt{2}} - \frac{\epsilon}{2}i \\ 1 \end{pmatrix}, v_3 = \begin{pmatrix} \frac{-2}{\epsilon^2} \\ \frac{-i}{\epsilon} \\ \frac{-i}{\epsilon} \\ 1 \end{pmatrix}, v_4 = \begin{pmatrix} 0 \\ 1 \\ -1 \\ 0 \end{pmatrix} \tag{2.43}$$

Now we can write the time-dependent amplitude vector when ϵ becomes very large as:

$$|\psi(t)\rangle = c_1 \cdot e^{-3.412it} e^{-\frac{1}{\epsilon}t} \begin{pmatrix} \frac{1}{2} + \frac{\epsilon}{\sqrt{2}}i \\ -\frac{1}{\sqrt{2}} - \frac{\epsilon}{2}i \\ -\frac{1}{\sqrt{2}} - \frac{\epsilon}{2}i \\ 1 \end{pmatrix} + c_2 \cdot e^{-0.586it} e^{-\frac{1}{\epsilon}t} \begin{pmatrix} \frac{1}{2} - \frac{\epsilon}{\sqrt{2}}i \\ \frac{1}{\sqrt{2}} - \frac{\epsilon}{2}i \\ \frac{1}{\sqrt{2}} - \frac{\epsilon}{2}i \\ 1 \end{pmatrix} + c_3 \cdot e^{-2it} e^{-\epsilon t} \begin{pmatrix} \frac{-2}{\epsilon^2} \\ \frac{-i}{\epsilon} \\ \frac{-i}{\epsilon} \\ 1 \end{pmatrix} + c_4 \cdot e^{-2it} \begin{pmatrix} 0 \\ -1 \\ 1 \\ 0 \end{pmatrix} \tag{2.44}$$

which, as $\epsilon \to \infty$, simplifies to:

$$|\psi(t)\rangle = c_1 \cdot e^{-3.412it} \begin{pmatrix} \frac{\epsilon}{\sqrt{2}}i \\ -\frac{\epsilon}{2}i \\ -\frac{\epsilon}{2}i \\ 1 \end{pmatrix} + c_2 \cdot e^{-0.586it} \begin{pmatrix} -\frac{\epsilon}{\sqrt{2}}i \\ -\frac{\epsilon}{2}i \\ -\frac{\epsilon}{2}i \\ 1 \end{pmatrix} + c_3 \cdot e^{-2it} \cdot 0 \begin{pmatrix} 0 \\ 0 \\ 0 \\ 1 \end{pmatrix} + c_4 \cdot e^{-2it} \begin{pmatrix} 0 \\ -1 \\ 1 \\ 0 \end{pmatrix} \tag{2.45}$$

We we can apply our initial condition $\psi(0) = (1,0,0,0)^T$ and find $c_1 = \frac{-i}{\sqrt{2}\epsilon}$, $c_2 = \frac{i}{\sqrt{2}\epsilon}$, $c_3 = 0$, $c_4 = 0.$, so our time-dependent amplitude vector becomes:

$$|\psi(t)\rangle = e^{-3.412it} \begin{pmatrix} \frac{1}{\sqrt{2}} \\ \frac{-1}{2\sqrt{2}} \\ \frac{-1}{2\sqrt{2}} \\ 0 \end{pmatrix} + e^{-0.586it} \begin{pmatrix} \frac{1}{2} \\ \frac{1}{2\sqrt{2}} \\ \frac{1}{2\sqrt{2}} \\ 0 \end{pmatrix} \tag{2.46}$$

We can now clearly see that when the strength of the potential $\epsilon \to \infty$, the walker will never reach the final vertex, $|11\rangle$.

It turns out that this method of defining the sink in our graph is essentially the same as preforming a concurrent measurement, where the strength ϵ of our potential is synonymous with the time in between measurements. Intuitively by looking at the formulas we see that if the value of our potential is close to zero, it is almost as if there is no sink at all, meaning the hitting time will approach infinity as $\epsilon \to 0$. When the potential is very high, however, it is very difficult for the walker to reach the sink vertex at all, this corresponds to preforming the concurrent measurement very frequently and will also results in a very large hitting time.

3

Results

In this chapter, we try to show that a quantum walker is faster than a classical walker at traversing a computer algorithm, which can be see as a computational decision tree. We do this by analysing the hitting times of a CRW and a QRW in an n-dimensional hypercube.

3.1. Hitting times in a Hypercube

To see if a quantum random walk is indeed faster at traversing a hypercube than its classical counterpart, we need to look at the *corner-to-corner hitting time* of both walks. This means the starting vertex, s = (000...000), has a Hamming-distance of n to the end vertex, t = (111...111). Literature has shown the corner-to-corner hitting time in the classical case to be in exponential time (in relation to n) and predicts the quantum case to be in polynomial time [13]. This would mean a very significant speed-up for large values of n (which is the case for most big algorithms).

3.1.1. Analytical approach

Firstly we'll try to derive an analytical expression for the hitting time of a classical random walk in a hypercube. To do this, the graph of a hypercube must be simplified, by looking at the graph as a line, where each vertex represents a column (like in section 2.3.3) containing all vertices with the same Hamming weight. It is important to remember that every vertex in an n-dimensional hypercube is connected to n other vertices, one for each bit that can be flipped (changed from zero to one and vice versa). In this case we'll say that for every step, the walker has an equal probability to move to any adjacent vertex ($p = \frac{1}{2}$. This allows us to clearly see that when a walker is located in a vertex with Hamming weight 2, $v \in V_2$, it is connected to 2 vertices with a weight of 1 (corresponding to flipping one of the two 1-bits in v) and it is connected to $n-2$ vertices with a weight of 3 (corresponding to flipping one of the $n-2$ 0-bits in v). We can now represent our n-dimensional hypercube in the following manner:

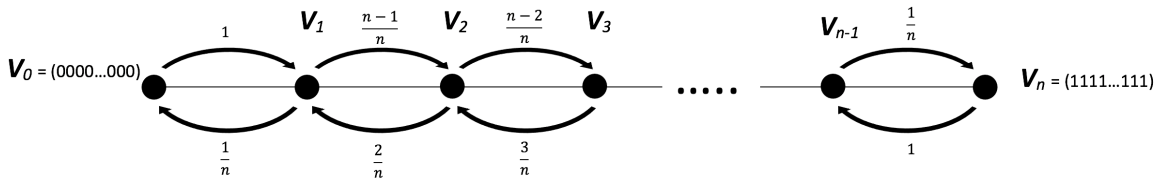

Figure 3.1: An n-dimensional hypercube reduced to a single line with $n+1$ nodes, where each node V_i represents a column containing all vertices with Hamming weight i. Also shown are the probabilities to jump to a specific neighbouring column, from a vertex in a certain column.

Our walker starts at the vertex (000.00). The first thing we calculate is the recurrence time, $\mathbb{E}_0$, which is the expected number of steps it takes the walker to revisit the start vertex for the first time. This value is given by the inverse of the stationary probability distribution of our graph, $\pi(v)$ [19]. Since there are 2^n vertices in total,

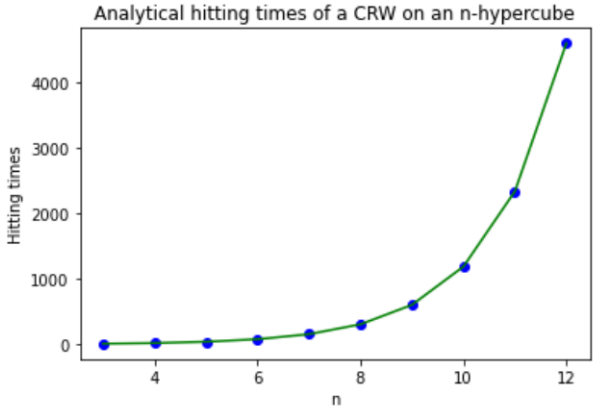

Figure 3.2: The corner-to-corner hitting time, calculated using formula 3.4, for different values of n (blue dots), fitted to an exponential function (green).

we have $\pi(v) = 1/2^n$, so:

$$\mathbb{E}_0 = \frac{1}{\pi(v)} = 2^n \tag{3.1}$$

We know that the only way the walker can revisit (000.00) is if it was in a vertex with weight 1 in the step before. From a certain vertex in V_1 the probability to move to V_0, rather than V_2, is $\frac{1}{n}$ (see figure 3.1). It is also known that V_1 contains n vertices, so V_0 could be reached from any of these n vertices. Using this logic, we can derive the following formula:

$$\mathbb{E}_0 = 1 + \frac{1}{n}\mathbb{E}_1 \cdot n \tag{3.2}$$

where $\mathbb{E}_1$ is the expected first hitting time for a specific vertex in V_1. Now, since we know $\mathbb{E}_0$, we can calculate $\mathbb{E}_1 = 2^n - 1$. Applying the same logic, we see that you can only get to V_1 from the starting position V_0 or neighbouring column V_2. Any specific vertex in V_1 is connected to 1 vertex in V_0 and $n-1$ vertices in V_2, so there are $n-1$ vertices that each have probability $\frac{1}{n}$ to map to that specific vertex in V_1. This gives the following formula for the expected hitting time for a vertex in V_1:

$$\mathbb{E}_1 = 1 + \frac{1}{n} \cdot 0 + \frac{1}{n}\mathbb{E}_2 \cdot (n-1) \tag{3.3}$$

Here the 0 comes from the fact that our walk starts in V_0. Now we can rewrite this to isolate $\mathbb{E}_2$ and plug in our found value for $\mathbb{E}_2$. If we keep applying this logic we end u with the following recursion formula for the hitting time of a specific vertex in V_k:

$$\mathbb{E}_k = 1 + \frac{k}{n} \cdot \mathbb{E}_{k-1} + \frac{n-k}{n}\mathbb{E}_{k+1}, \quad 1 < k < n \tag{3.4}$$

We can now use this recursive formula to calculate the hitting times of any vertex in a n-dimensional hypercube. Calculating the corner-to-corner hitting time, $\mathbb{E}_n$, for different values of n allows us to find the polynomial relation between n and T, seen in figure 3.2. Fitting an exponential function to the calculated hitting times we find that the hitting time of the classical random walk is $\mathcal{O}(2^n)$.

For the QRW it is much harder to analytically find a solution, as we don't have an analytical expression for the wavefunction.

3.1.2. Numerical calculations

Using the formulas described in the Theory section, we are able to compute the hitting times of both a CRW and QRW in an n-dimensional hypercube for different values of n. More specifically we calculate 2.7 for the classical random walk and 2.25 for the QRW by dividing time into small steps dt. As we have seen in figure 3.2, the hitting time of a random walk can scale exponentially with n ($\mathcal{O}(2^n)$ in case of the CRW), so the amount of computer power needed grows rapidly as we calculate the hitting times for bigger values of n. To speed up these calculations, formula 2.25 was modified to

$$\psi(t + dt) = e^{-iM\,dt} \cdot \psi(t) \tag{3.5}$$

which significantly decreased the calculation time.

Literature suggests the corner-to-corner hitting time is exponential in the CRW (as seen in the previous section) and at most polynomial in the quantum case [13]. Fitting these functions to the found data allows us to predict the behaviour of the walks for larger values of n (as these would take too long to calculate numerically). The plots in figure 3.3 show the found hitting times for n up to 12.

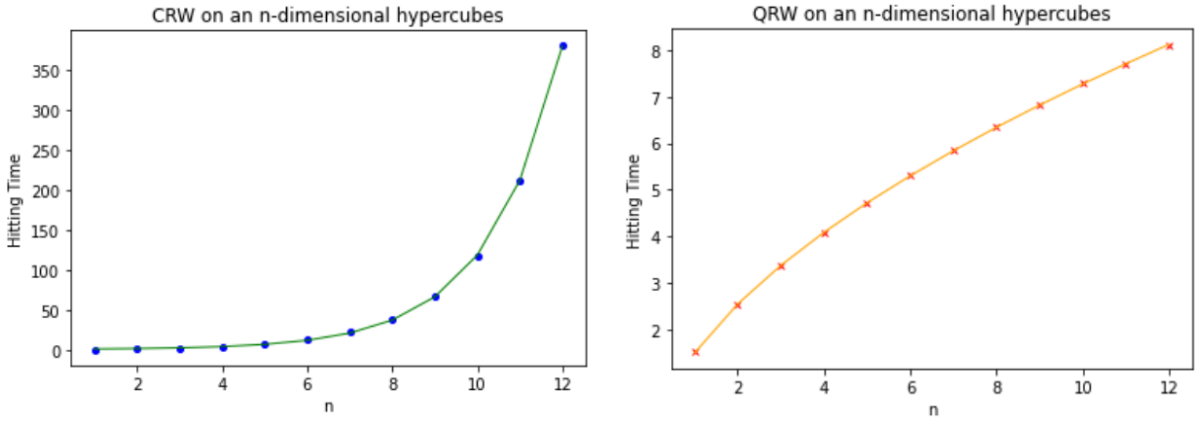

Figure 3.3: The hitting time of a CRW (blue dots) and QRW (red crosses) in an n-dimensional hypercube, for different values of n. The data for the CRW has been fitted with an exponential function (green) and the QRW with a polynomial function (orange).

We immediately see that for large n, the hitting time of the quantum random walk is much faster than its classical counterpart. The exponential and polynomial function fitted to our data for the CRW and QRW resp. come out as:

$$T_{\text{classic}} = 0.34 \cdot 1.80^n + 1.72 \tag{3.6}$$

$$T_{\text{quantum}} = 2.25 \cdot n^{0.55} - 0.75 \tag{3.7}$$

We see the hitting time for the CRW is not quite $\mathcal{O}(2^n)$, but it definitely increases exponentially. In the quantum case we see the relation between n and the hitting times is not a polynomial, but looks more like a square root. To give an idea of how much faster the hitting time for the CRW increase compared to its quantum counterpart, at $n = 20$ we have $T_{\text{quantum}} = 11.0$, whereas $T_{\text{classic}} = 40557$ and for $n = 100$: $T_{\text{quantum}} = 27.9$, whereas $T_{\text{classic}} = 8.14 \cdot 10^{24}$.

3.1.2.1 Hamming distance of sink vertex

Now that we have seen how the corner-to-corner hitting time relates to n, it is interesting to study what happens when the sink vertex is located in a different vertex. Rather than looking at a sink located in a vertex that has a Hamming distance equal to n from the start vertex (as was previously that case, we now look at what happens to the hitting time when, for example, $d(s,t) = n-1$. In the classical case the relation between the Hamming distance (d for short) between the start and sink vertex can be seen in figure 3.4. We see that the hitting time increases only very slightly as the Hamming distance to the sink vertex increase, but for the most part the relation between n and the hitting time (the shape of the graph) stays very much the same. When an exponential function is fitted to the calculated data we can see this more clearly:

$$T_{d=1} = 0.26 \cdot 1.82^n + 0.94 \tag{3.8}$$

$$T_{d=2} = 0.30 \cdot 1.81^n + 1.74 \tag{3.9}$$

$$T_{d=3} = 0.30 \cdot 1.81^n + 2.64 \tag{3.10}$$

These exponential functions also match very closely to the formula 3.6 found in the previous section.

It turns out that when we look at cases of the QRW where $d(s,t) \neq n$, we need to introduce a new term, the *hitting probability*. It is given by:

$$P_{\text{hit}} = \sum_{t=0}^{\infty} p(t) \tag{3.11}$$

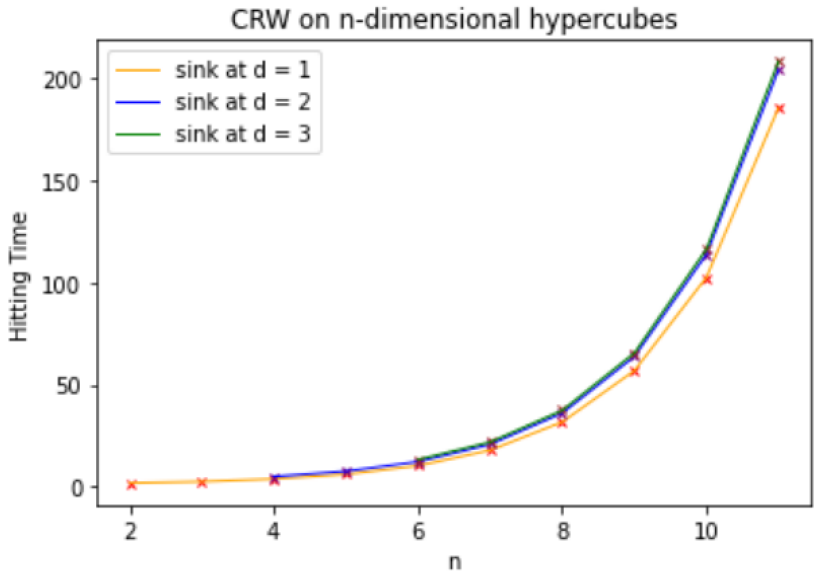

Figure 3.4: Hitting times for a CRW in n-dimensional hypercubes for different Hamming distances d between the start and sink vertex.

Hamming distance	T	T_{real}	P_{hit}
0	0.000	0.000	1.0000
1	1.237	9.899	0.1250
2	0.733	20.522	0.0357
3	0.564	31.571	0.0179
4	0.523	36.608	0.0143
5	0.572	32.028	0.0179
6	0.770	21.557	0.0357
7	1.493	11.944	0.1250
8	6.422	6.422	1.0000

Table 3.1: Hitting times (both normal and real) and probability for a QRW in an 8-dimensional hypercube for different Hamming distances between the start and sink vertex

where $p(t)$ is the probability that the walker hits the sink vertex at time t. We have always assumed that a random walker will, eventually, hit as $t \to \infty$, but this is not necessarily always the case [22]. When the hitting probability is not 1, it is possible for the walker to move through the graph forever, without hitting the sink vertex. Say a walk has hitting probability $\mathcal{O}(1/p)$ when the sink vertex is at a certain point. We would then need to repeat that walk p times to get a hitting probability $\mathcal{O}(1)$. Therefore we cannot regard the hitting time, T, described in 2.38 as a criterion of how fast a quantum walk spreads through a graph. Instead, we define the *real hitting time* as follows:

$$T_{\text{real}} = \frac{T}{P_{\text{hit}}} \tag{3.12}$$

To see how this definition of the hitting time differs from the previous one (2.38), table 3.1 shows the value for both definitions, for sink vertices with different Hamming distances from the start vertex. Since the real hitting time is a much better criterion for the speed at which a quantum walk traverses a graph, we will, from now on, only look at this definition and simply refer to it as the hitting time.

First of all, we see a pattern in the hitting probability, namely that, in an n-dimensional hypercube, it is given by

$$P_{\text{hit}}(d) = \frac{1}{\binom{n}{d}} \tag{3.13}$$

where d is the Hamming distance between the start vertex and sink vertex. This matches very closely to results found in [22]. It is also quite logical when we have the insight that the column $\mathbf{V}_i$ of an n-dimensional

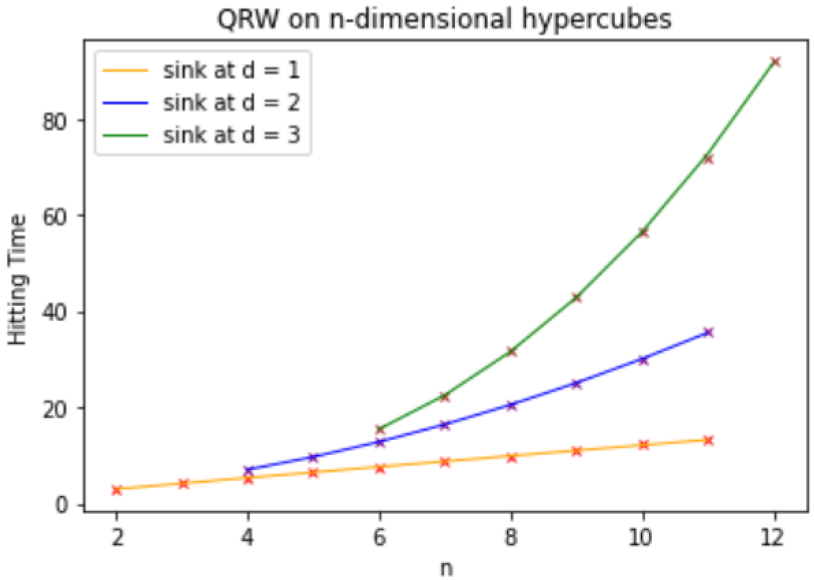

Figure 3.5: Hitting times of a QRW on n-dimensional hypercubes with sink vertices at different Hamming distance d.

hypercube contains $\binom{n}{i}$ vertices in total. What we can conclude from this is that, somehow, the interference pattern evolves in such a way that the walk traverses back and forth across the columns. Then, when a single vertex in that column contains a sink, it absorbs $\frac{1}{\binom{n}{d}}$ of the total probability over time, but the rest of the wavefunction continues to oscillate through the graph, skipping the vertex containing the sink.

Table 3.1 also shows that the hitting time increases as the Hamming distance increases, for $d \leq \frac{n}{2}$. In figure 3.5 we see the relation between the hitting time of a quantum walk through n-dimensional hypercubes and the Hamming distance d. In table 3.1 we see when $d > \frac{n}{2}$ the hitting time starts to decrease with the Hamming distance, because the number of vertices in the columns starts to decrease. This is why we have only included hypercubes for which $d \leq \frac{n}{2}$ holds. Fitting polynomial functions to the data in figure 3.5 allows us to find the relation between the (real) hitting time, T, and n when the sink vertices are located at incrementally increasing Hamming distances from the start vertex. We find the following fits:

$$T_{d=1} = 1.26 \cdot n^{0.96} + 0.52 \tag{3.14}$$

$$T_{d=2} = 0.39 \cdot n^{1.85} + 1.87 \tag{3.15}$$

$$T_{d=3} = 0.10 \cdot n^{2.73} + 1.95 \tag{3.16}$$

Right away we see that the degree of the polynomial increases as the Hamming distance between the start and sink vertex. It turns out that the degree of the polynomial when $d = i$ is the same as when $d = n - i$. This is probably due to the symmetry of the hypercube. We can conclude that, in general, the hitting times is larger, the closer the sink vertex is to the centre column(s) of the hypercube.

3.1.2.2 Initial distribution

So far have only considered a quantum walk in a hypercube with an initial distribution

$$\psi(0) = \begin{pmatrix} 1 \\ 0 \\ \vdots \\ 0 \end{pmatrix} \tag{3.17}$$

Obviously, studying initial distributions where the walk starts at a different vertex than (000.00) is the same as varying the Hamming distance of the sink vertex, which we did in section 3.1.2.1. It is, however, interesting to see what happens when the walk starts in a superposition of different vertices. Since physically a classical

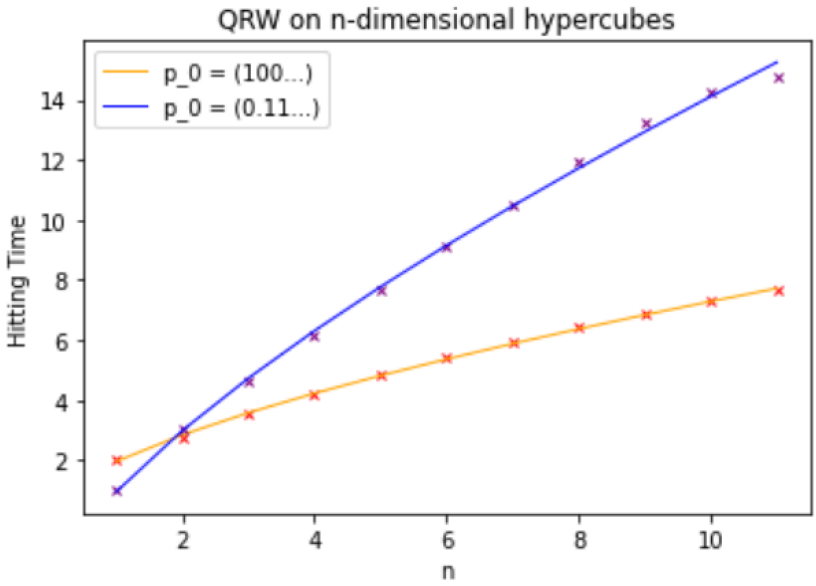

Figure 3.6: Hitting times of a QRW in n-dimensional hypercubes where the initial distributions is described by 3.18 (blue) and the case when the walk starts of in the vertex s=(000...00) (orange).

random walker cannot be in a superposition, we will only study the case of the quantum random walk. We can, for example, look at what happens when we set the initial distribution of the quantum walk equal to an even distribution across all vertices with Hamming weight 1 (all vertices in column $\mathbf{V}_1$). Since column $\mathbf{V}_1$ of an n-dimensional hypercube contains n vertices we have the initial distribution:

$$\psi_i(0) = \begin{cases} \frac{1}{\sqrt{n}} & w(i) = 1 \\ 0 & \text{otherwise} \end{cases}, \quad i = 0,1,2,...,2^n \tag{3.18}$$

where $w(i)$ is the Hamming weight of vertex i. We might expect the hitting time to be shorter in this case, as the quantum walk has already spread out and is closer to the sink vertex t=(111.11). Figure 3.6 shows the hitting times in n-dimensional hypercubes in the case where the walk has an initial distributions as described by 3.18 compared to when the quantum walk starts of in the vertex s=(000...00). Interestingly we see that, even though the walker is "closer" to the sink vertex, the hitting time is still longer than when the walker starts at the vertex s. Fitting a polynomial function to our found data allows us to analyse the behaviour of the hitting time in relation to the dimension of the hypercubes, n, and gives the following formula:

$$T_{\text{quantum}} = 3.24 \cdot n^{0.70} - 2.29 \tag{3.19}$$

We see that when the quantum walk starts in the distribution described by 3.18 T increases faster with n than the hitting time described by 3.7. In this case the Hamming distance between the start vertices and sink vertex is 1, but we see in this case the hitting time is faster than in the case described by 3.14, where the walk starts from a single vertex at Hamming distance 1. Lastly we notice that the hitting probability in this case is equal to 1. This is probably due to the fact that the initial distributions is symmetric, allowing symmetric evolution of the walk through the graph.

3.1.3. Potential strength and partial measurement

As discussed previously, the strength of our complex potential described in section 2.3.4.1 is very similar preforming a partial measurement operator, $\Pi - |v\rangle\langle v|$, on the sink vertex v. In this section we will try to compare both methods of calculating the hitting time.

Formula 2.32 allowed us to explain the oscillatory behaviour seen in 2.11. Knowing the probability of the walker to be in the last vertex, p_v oscillates with a period of π with $p_v = 0$ at $t = k\pi, k = 0,1,2,...$, we expect infinitely large hitting times when the partial measurements are preformed at these exact times. Also, as mentioned before, when concurrent measurements occur too frequently causes the evolution of our system

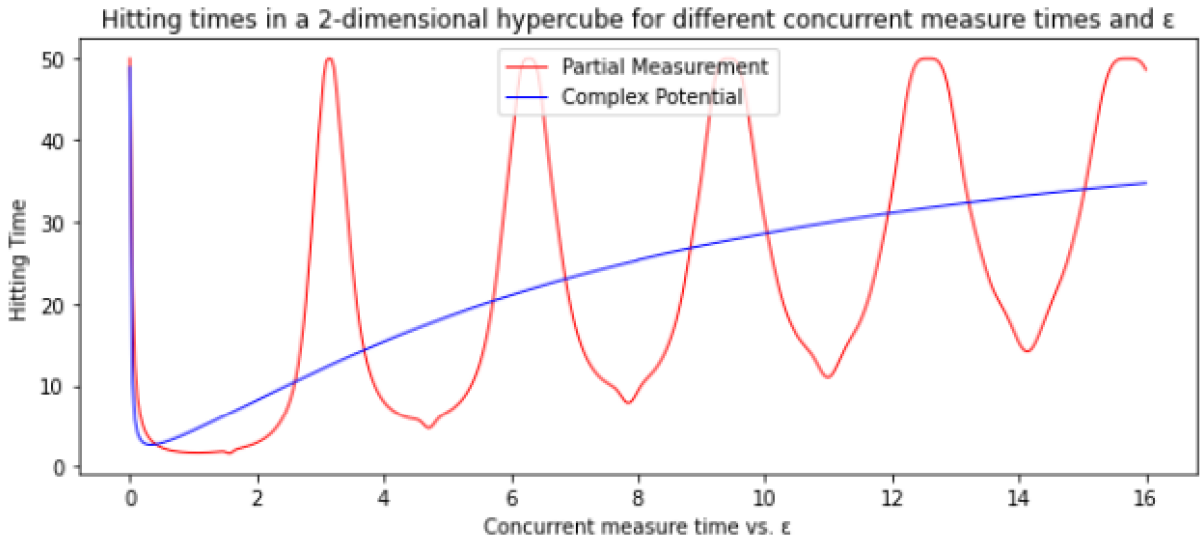

Figure 3.7: Hitting time of a continuous-time QRW in a 2-dimensional hypercube measured using two different methods. In red, the hitting times for partial measurements preformed after different times and in blue, the hitting times for a systems containing a complex potential of different strengths.

to freeze due to the Quantum Zeno effect. However, when the time between two concurrent measurement is too large, our walker is able to traverse the graph freely without hinder, resulting in longer hitting times. Figure 3.7 allows us to see the similarities, but also the striking differences between the concurrent measurement and complex potential method. We see that, in general, that the hitting time becomes infinitely large as the time t between measurements approaches 0 or as $t \to \infty$ and that this is also the case when $\epsilon \to 0$ or $\epsilon \to \infty$. Furthermore, we see the expected oscillatory behaviour, with peaks at $t = k\pi, k = 0,1,2,...$ and local minima at $t = (2k+1)\frac{\pi}{2}, k = 0,1,2,....$

Because at the times $t = k\pi, k = 0,1,2,...$ the hitting time is infinite, a maximum time was set at $t = 50$, after which the walk is halted. To give a better view of how similar the results for the two different methods of measuring the hitting time of a QRW in a 2-dimensional hypercube (concurrent measurement and complex potential), we look at the average of the results for the concurrent measurement. Figure 3.8 shows the results for the hitting times averaged over the 10 time steps before and 10 steps after that concurrent measurement time.

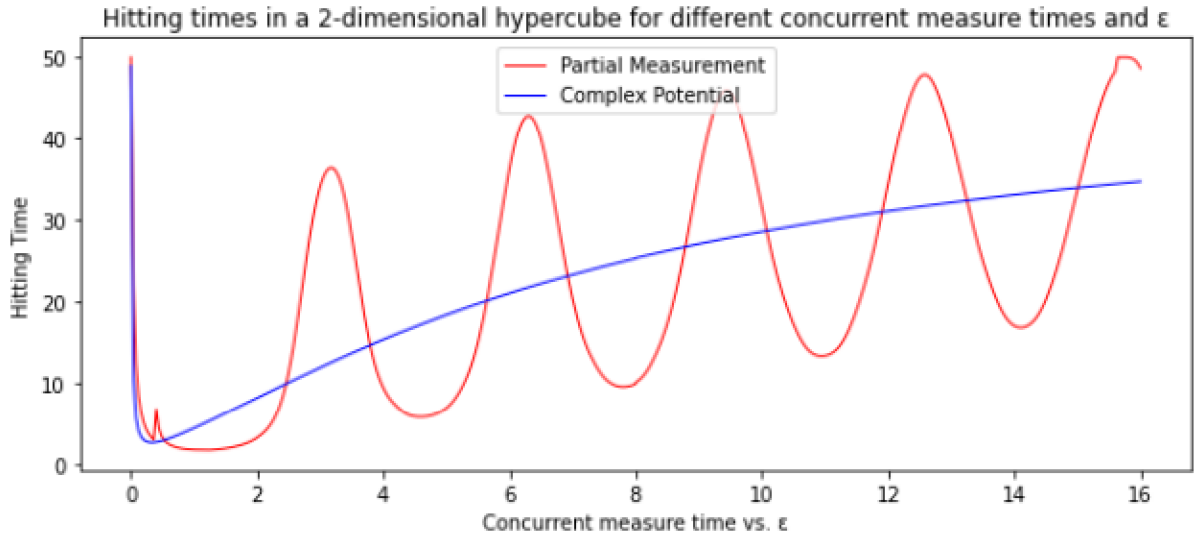

Figure 3.8: Hitting time of a continuous-time QRW in a 2-dimensional hypercube measured using two different methods. In red, the average hitting times for partial measurements preformed after different times and in blue, the hitting times for a systems containing a complex potential of different strengths.

4

Discussion

In this thesis a theoretical analysis of random walks was given. Based on this theory numerical simulations were done. This often meant using approximations, which is always going to lead to some minor errors. These errors could explain some of the deviations of our data from the values predicted our analytically derived formulas, as was the case for 3.6 that was not exactly equal to the expected 3.2.

Having analysed a quantum walk through a hypercube, we found an expression for the relation between the hitting time and the dimension of the hypercube. Although our findings match closely to some of the literature studied [22][15][13], we cannot confirm it with or compare it to experimental findings, as I was not able to find an example of an experimental quantum walk through a hypercube.

Furthermore we found that, when not looking at the corner-to-corner hitting time, the hitting probability was not always equal to one. This translates to the fact that there is a chance the walker can traverse the graph forever, without being absorbed into the sink (hitting). In other words, when the sink vertex is not exactly opposite the starting vertex, the asymmetry in the path of the walk causes the wavefunction to reach a certain distribution that allows the walk to oscillate through the graph without hitting (so essentially avoiding) the sink vertex, which is frankly amazing.

Moreover, a hitting probability lower than one should, intuitively, lead to an infinite hitting time, but by redefining the hitting time, using 3.12, we were still able to find a finite hitting time. Not only was this hitting time finite, but it was still polynomial in relation to n. Literature has shown that infinite hitting times do occur, when using different coin operators or when studying different graphs, and that suggest that the speed-up or slow-down of the quantum walk is largely caused by the symmetry of the graph that is being traversed [15].

After having studied how fast the different types of random walks are able to traverse a graph the big question remains: "How well does the hitting time of a random walk in a graph translate to the speed of a certain algorithm?" The graphs analysed in this thesis were very convenient in their simplicity and symmetry, whereas most computer algorithms are far from those things. So, even though we can confidently say that the hitting times of a quantum walk in a hypercube is lower than its classical counterpart, this doesn't help us confirm that a quantum walk will be faster or more efficient at traversing different types of algorithms. An example for which it has been shown is Grover's algorithm, that is able to search through an unstructured database in $\mathcal{O}(\sqrt{n})$ rather than the classical $\mathcal{O}(n)$ [16]. This is why looking at what types of algorithms are closely represented by what types of graphs would make for an interesting followup study.

5

Conclusion

In this thesis the hitting time of random walks was studied. Specifically, we investigated the hitting time of the classical random walk and continuous-time quantum random walks in an n-dimensional hypercube. We looked into different definitions of the hitting time, we varied the position of the sink vertex with respect to the starting vertex and changed the initial distribution of the walk, to see how these slight changes influence the hitting time.

Analytical calculations told us the corner-to-corner hitting time of a CRW in an n-dimensional hypercube is $\mathcal{O}(2^n)$, meaning it is exponential in relation to n. Numerical calculations confirmed this exponential hitting, even when the sink vertex was positioned closer to the start vertex.

In the case of the quantum walk we found that in the ideal scenario of the corner-to-corner hitting time in an n-dimensional hypercube the speed-up compared to the classical walk was the greatest. The hitting time was $\mathcal{O}(n^{0.55})$ in relation to the dimension n. When the initial distribution of the walker was changed to an evenly spread superposition across the whole first column of the hypercube we found a hitting time of $\mathcal{O}(n^{0.70})$.

Now, when the Hamming distance d between the start and sink vertex (which was n) was decreased, something very interesting happened. It was no longer the case that as $t \to \infty$ the walker definitely hits at some point in time. We found that the probability of hitting, at all, in an n-dimensional hypercube was equal to $\frac{1}{\binom{n}{d}}$. This meant we needed to alter our definition of the hitting time. Having done this, we found the relation between the hitting time and n was still polynomial, although the degree of the polynomial had increased pretty significantly.

Finally we made a comparison between the hitting times of a QRW using two different definitions of the hitting time, one that preforms a concurrent or partial measurement and another that introduces a complex potential in the sink vertex. It turn out that the behave similarly, when comparing extreme values of the concurrent hitting time and the potential strength.

So, to conclude, where classical random walk always have exponential hitting times in an n-dimensional hypercubes, the quantum random walk can hit in linear time (or even faster) in some special cases. In general, however, the hitting time of a QRW is polynomial with the dimension of the hypercube it is traversing. This is a very promising and significant speedup compared to the CRW, so quantum algorithms seem to be a much faster alternative to their classical counterparts.

Further research into relation between different types of graphs and algorithms could prove interesting to test the significance of the hitting time of a random walk in predicting the speed of an algorithm. Additionally, future studies could look into what happens when the hitting probability is smaller than 1 and the walk traverses a hypercube indefinitely, essentially avoiding the sink vertex forever.

References

[1] Y. Aharonov, L. Davidovich, and N. Zagury. Quantum random walks. *Phys. Rev. A*, 48:1687–1690, Aug 1993. doi: 10.1103/PhysRevA.48.1687. URL https://link.aps.org/doi/10.1103/PhysRevA.48.1687.

[2] Michael Ben-Or. Lower bounds for algebraic computation trees. *Proceedings of the fifteenth annual ACM symposium on Theory of computing*, 15(1):80–86, 1983.

[3] C. M. Chandrashekar. Zeno subspace in quantum-walk dynamics. , 82(5):052108, November 2010. doi: 10.1103/PhysRevA.82.052108.

[4] Andrew M. Childs, Edward Farhi, and Sam Gutmann. An example of the difference between quantum and classical random walks. *arXiv e-prints*, art. quant-ph/0103020, March 2001.

[5] E. Codling, M. J. Planck, and S. Benhamou. Random walk models in biology. *J R Soc Interface*, 2:834, May 2008. doi: 10.1098/rsif.2008.0014.

[6] Martin Dyer, Alan Frieze, and Ravi Kannan. A random polynomial-time algorithm for approximating the volume of convex bodies. *J. ACM*, 38(1):1–17, jan 1991. ISSN 0004-5411. doi: 10.1145/102782.102783. URL https://doi.org/10.1145/102782.102783.

[7] Edward Farhi and Sam Gutmann. Quantum computation and decision trees. , 58(2):915–928, August 1998. doi: 10.1103/PhysRevA.58.915.

[8] Johnson S Ford L. A tournament problem. *The American Mathematical Monthly*, 66, 1959.

[9] David J. Griffiths. *Introduction to Quantum Mechanics*. Cambridge University Press, University Printing House, Cambridge, third edition, 2018.

[10] Tony Hey. Quantum computing: An introduction. *Computing Control Engineering Journal*, 10(1):105–112, 1999.

[11] Mark Jerrum and Alistair Sinclair. Approximating the permanent. *SIAM Journal on Computing*, 18(6): 1149–1178, 1989. doi: 10.1137/0218077. URL https://doi.org/10.1137/0218077.

[12] J. Kempe. Quantum random walks: an introductory overview. *Contemporary Physics*, 44(4):307–327, April 2003. doi: 10.1080/0010715103100011077 6.

[13] Julia Kempe. Quantum random walks hit exponentially faster. 2002. doi: 10.48550/ARXIV.QUANT-PH/0205083. URL https://arxiv.org/abs/quant-ph/0205083.

[14] V. Kendon and C. Tamon. Perfect state transfer in quantum walks on graphs. *Journal of Computational and Theoretical Nanoscience*, 8:422–433, Mar 2011. doi: 10.1166/jctn.2011.1706. URL https://doi.org/10.1166/jctn.2011.1706.

[15] Hari Krovi and Todd A. Brun. Hitting time for quantum walks on the hypercube. *Phys. Rev. A*, 73:032341, Mar 2006. doi: 10.1103/PhysRevA.73.032341. URL https://link.aps.org/doi/10.1103/PhysRevA.73.032341.

[16] C. Lavor, L. R. U. Manssur, and R. Portugal. Grover's Algorithm: Quantum Database Search. *arXiv e-prints*, art. quant-ph/0301079, January 2003.

[17] Cristopher Moore and Alexander Russell. Quantum Walks on the Hypercube. *arXiv e-prints*, art. quant-ph/0104137, April 2001.

[18] G. D. Paparo and M. A. Martin-Delgado. Google in a Quantum Network. *Scientific Reports*, 2:444, June 2012. doi: 10.1038/srep00444.

[19] Gomez R. On mean recurrence times of markov chains and spanning tree invariants. *Linear Algbebra and its Applications*, 433, 2010.

[20] Uwe Schöning. A probabilistic algorithm for k-sat and constraint satisfaction problems. In *FOCS*, 1999.

[21] Feng Xia, Jiaying Liu, Hansong Nie, Yonghao Fu, Liangtian Wan, and Xiangjie Kong. Random Walks: A Review of Algorithms and Applications. *arXiv e-prints*, art. arXiv:2008.03639, August 2020.

[22] Tomohiro Yamasaki, Hirotada Kobayashi, and Hiroshi Imai. Analysis of absorbing times of quantum walks. , 68(1):012302, July 2003. doi: 10.1103/PhysRevA.68.012302.

A

Appendix A

The most important code used in the making of all plots and numerical calculations is available at this link.

B

Appendix B

Here we try to calculate the eigenvectors, v_1, v_2, v_3, v_4, and corresponding eigenvalues, $\lambda_1, \lambda_2, \lambda_3, \lambda_4$ of the matrix:

$$H = \begin{pmatrix} -2 & 1 & 1 & 0 \\ 1 & -2 & 0 & 1 \\ 1 & 0 & -2 & 1 \\ 0 & 1 & 1 & -2 \end{pmatrix}. \tag{B.1}$$

We do his by solving the equation: $\det(H - \lambda I) = 0$. Firstly we introduce the new eigenvalue, λ^*, where $-\lambda^* = 2 - \lambda$, to simplify our initial matrix. We will still denote it as λ, to save time, but we will take care of this later. We find

$$\begin{vmatrix} -\lambda & 1 & 1 & 0 \\ 1 & -\lambda & 0 & 1 \\ 1 & 0 & -\lambda & 1 \\ 0 & 1 & 1 & -\lambda \end{vmatrix} = -\lambda \begin{vmatrix} -\lambda & 0 & 1 \\ 0 & -\lambda & 1 \\ 1 & 1 & -\lambda \end{vmatrix} - \begin{vmatrix} 1 & 1 & 0 \\ 0 & -\lambda & 1 \\ 1 & 1 & -\lambda \end{vmatrix} + \begin{vmatrix} 1 & 1 & 0 \\ -\lambda & 0 & 1 \\ 1 & 1 & -\lambda \end{vmatrix} \tag{B.2}$$

This allows us to calculate the characteristic polynomial. Switching our notation of the temporary eigenvalue back, we find

$$p(\lambda^*) = -(\lambda^*)^4 + 4(\lambda^*)^2 = -\lambda^4 - 8\lambda^3 - 20\lambda^2 - 16\lambda = -(\lambda+2)^2\lambda(\lambda+4) = 0 \tag{B.3}$$

We find $\lambda_1 = -4$, $\lambda_2 = 0$, $\lambda_3 = -2$, $\lambda_4 = -2$. Now all that rests us to do is find the vectors v that satisfy the equation $H\vec{v} = \lambda\vec{v}$ or $(H - \lambda I)\vec{v} = \vec{0}$ for all eigenvalues. This results in a series of equations that can be solved very simply and we find:

$$v_1 = \begin{pmatrix} 1 \\ -1 \\ -1 \\ 1 \end{pmatrix}, v_2 = \begin{pmatrix} 1 \\ 1 \\ 1 \\ 1 \end{pmatrix}, v_3 = \begin{pmatrix} 1 \\ 0 \\ 0 \\ 1 \end{pmatrix}, v_4 = \begin{pmatrix} 0 \\ 1 \\ -1 \\ 0 \end{pmatrix} \tag{B.4}$$