Quantum-walk-based search algorithm performance on 2D lattices with bond percolation as decoherence

A thesis submitted in fulfilment of the requirements

for the degree of Bachelors in Mathematics and Physics

at Delft University of Technology

by

Leon Conde Miranda Schmittgens

August 2025

Abstract

Name of the student: **Leon Conde Miranda Schmittgens**

Degree for which submitted: **Double Bachelor Applied Mathematics & Physics**

Thesis title: **Quantum-walk-based search algorithm performance on 2D lattices with bond percolation as decoherence**

Thesis supervisors: **Dr. Miriam Blaauboer** and **Johan Dubbeldam, Ph.D.**

Month and year of thesis submission: **August 2025**

This thesis investigates the performance of quantum walk search algorithms (QWSAs) on two-dimensional rectangular lattices and on bond percolated two-dimensional square lattices. The focus is on how structural disorder, modelled by static and dynamic percolation, affects the success probability and optimal runtime of the algorithm. In the unpercolated case, the principal eigenvalue technique provides asymptotic expressions for runtime and success probability. These approximations agree well with simulations on square grids, but underestimate the runtime for rectangular grids. This seems to be a consequence of breaking the grid symmetry, which enhances higher-frequency spectral components and results in interference effects that delay the optimal runtime. The introduction of dynamic percolation causes a rapid drop in success probability, especially for larger grids. This models strong decoherence, where the short timescale of structural change disrupts coherent amplitude buildup. Static percolation degrades performance more gradually, with a sharp decline only near the percolation threshold, where global connectivity is lost. These results show that QWSAs are adversely affected by changing graph structures over time, highlighting the importance of sufficiently long coherence times. This study provides a way to test how well quantum search algorithms perform in disordered environments and future work could extend this to other types of graphs or quantum walk algorithms.

Contents

Chapter 1

Introduction

Computers are vital to our society. They are capable of quickly executing instructions and performing computations. This makes them incredibly useful. For instance, modern mobile phones effectively serve as portable computing devices. Moreover, computers are also used to let robots and machines execute tasks and by scientists or engineers to make numerical approximations [1][2]. Since computers have such a widespread use, it is important to exploit them to the best of their potential. This requires efficient algorithms [3][4]. These algorithms, which are a list of instructions, dictate what the computer is supposed to do. Algorithms can be either deterministic or probabilistic. Deterministic algorithms always produce the same output for a given input and follow the same sequence of steps every time [5]. Probabilistic algorithms, on the other hand, are algorithms that use randomness in their decision making. In some cases, it can be beneficial to use these probabilistic algorithms due to being able to quickly obtain good approximations or having faster average performances while still producing a correct answer with high probability [6].

In recent decades, there has been a lot of interest in quantum computers [7][8][9]. These are computers that use qubits (quantum bits), following the principles of quantum mechanics [10], instead of classical bits that behave as described by the laws of classical mechanics. Quantum mechanics predicts various non-classical behaviors, such as superposition and interference. An object in superposition means that it is in multiple different states at the same time. How these states are combined is described by the quantum state $|\psi\rangle$ of the object. A measurement of the object then follows the Born rule; a given state S is measured with a probability equal to the square of the absolute value of the amplitude of the state S in the quantum state ψ: $P[S] = |\langle S|\psi\rangle|^2$. After measurement, the quantum state collapses and the object is now in the state that was measured, so if state S was measured: $|\psi\rangle = |S\rangle$. The use of qubits allows these computers to take advantage of these quantum phenomena to achieve a quantum speed-up [11][12]. However, to actually achieve this speed-up, new algorithms specifically designed for quantum computers are required. Notable examples are Shor's algorithm for factoring numbers [13] and Grover's algorithm for searching unstructured databases [14]. Grover's algorithm uses a technique called amplitude amplification to increase the amplitude of the marked element and thereby decrease all other amplitudes. Grover's algorithm achieves a quadratic speed-up over classical counterparts.

A classical random walk is a random process where a "walker" makes a probabilistic walk on a lattice or graph [15][16]. A well-known example is the drunkard's walk on the integers $\mathbb{Z}$: assume a walker starts at the origin and flips a coin. Depending on the outcome of the coinflip, he either takes one step to the left or one step to the right. Now the walker is at -1 or 1 and the flip of the coin and the step are repeated. The position of the walker cannot be predicted at any time after $t = 0$. However, for every time step, it is possible to determine the likelihood of finding the walker at each position. This is the probability distribution of the classical walker. Classical random walks are famous for modelling diffusion and Brownian motion, but they are also useful for many efficient algorithms. In algorithms, they are often used for searching or testing properties, such as in: Monte Carlo Markov chain methods [17], in stochastic optimization algorithms (simulated annealing)[18] or in graph and network analysis [19].

The rise of quantum mechanics in the past century has also resulted in the development of quantum analogues to classical random walks, called quantum walks (QWs) [21][22][23]. Instead of a classical "walker", the moving object in a quantum walk is a quantum object: the quantum state. The movement of a quantum walk is created by two operators, instead of the coin flip and subsequent step of the classical walk, that act on the quantum state. Quantum walks exhibit certain interesting behaviors that classical walks do not exhibit, such as complex interference patterns

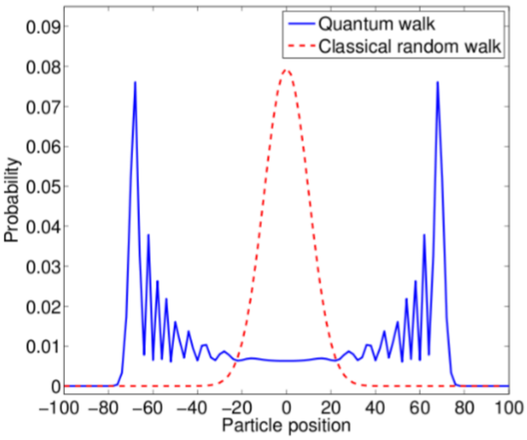

Figure 1.1: Probability distributions of a symmtric classical random walk and a symmetric quantum walk after N = 100 steps. Both walks starting at the origin[20].

and ballistic spreading. Interference patterns arise when an object interacts with itself such as waves; a peak might be higher or lower when two waves come together. Ballistic spreading means that the standard deviation of the walker's probability distribution grows linearly in time; $\sigma_x \sim t$. Intuitively, it can be seen as an object that propagates as fast as a particle that moves with constant velocity. In contrast, a classical walk exhibits diffusive behavior, which means that the standard deviation increases proportionally to the square root of time; $\sigma_x \sim \sqrt{t}$. How these quantum behaviors result in probability distributions that are significantly different from the probability distributions of classical random walks is illustrated in Figure 1.1. Another important difference between classical random walks and quantum walks to note is that the probability distribution in the case of classical random walks arises from the randomness in the dynamics, whereas the dynamics of quantum walks are deterministic. In quantum walks it is the walking object itself that gives rise to a probability distribution, following the Born rule.

These QWs can be used in quantum counterparts to classical random walk based algorithms. In particular, they can be combined with Grover's algorithm to obtain a quantum-walk-based search algorithm (QWSA) [24]. Grover's algorithm is meant for unstructured search as it disregards any underlying structure. QWSAs, on the other hand, are specifically meant for structured search, these search problems typically involve an underlying graph or lattice.

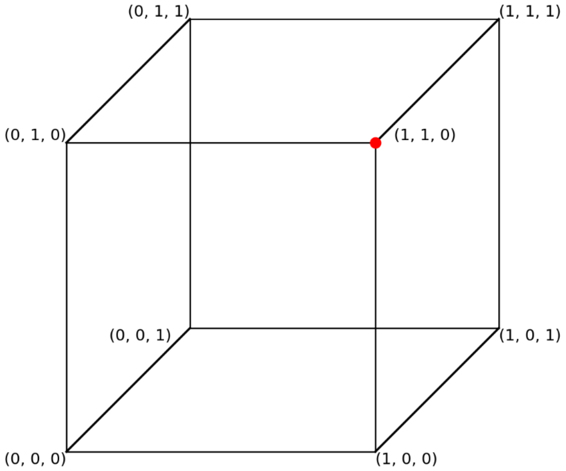

FIGURE 1.2: Cube (or 3D hypercube) representing the Boolean expression "A and B and not C" with the red dot representing the solution.

An example of a search problem is a Boolean satisfiability (SAT) problem. SAT problems are NP-complete which means there is no guaranteed way to quickly find the solution, but a computer can quickly check whether a solution is correct [25]. Due to the NP-completeness property, the best way to find a solution is to guess and verify solutions. The goal of a SAT problem is to find a truth assignment of a Boolean expression. A Boolean expression such as: "A AND B AND NOT C" which has as a solution the truth assignment A, B = TRUE and C = FALSE. A truth assignment such as A, B = TRUE and C = FALSE can be represented in binary $(1, 1, 0)$, this can be seen as a coordinate. This way of assigning coordinates to truth assignments creates a hypercube associated with the search problem. For the example of A, B = TRUE and C = FALSE this hypercube is a 3D cube, as presentend in Figure 1.2. A possible search method is to use a classical random walk on the hypercube and stop once the walker encounters and verifies the solution. A quantum computer could apply Grover's algorithm, but it ignores the underlying structure. So with a quantum computer, given the presence of a natural underlying structure, this problem lends itself to the use of QWSAs.

Both classical and quantum walks require an underlying structure, typically a graph or lattice, on which the walk is performed. Structures can however degrade or vary over time. These variations can be modeled using percolation theory, which describes the random process governing the presence of nodes or edges on a graph. This thesis uses bond percolation, where each edge in a graph is open (with a probability p) or closed (with a probability $1 - p$), independently of each other. For random walks it also matters how long the percolation is applied. Static percolation percolates the structure once and then keeps that percolated structure for the complete run of the walk. Whereas dynamic percolation reconfigures the percolated structure each time step. It is also possible to have some percolation period T_p, to define for how many time steps the percolated structure remains the same.

Random walks, both quantum and classical, have been studied in the context of percolation. With classical random walks, percolation can result in subdiffusive behavior or even localization. Subdiffusive behavior means that the spreading is of a lower order than diffusive spreading; $\sigma_x \sim t^\alpha, \alpha < \frac{1}{2}$. And localization means that the walker gets trapped in a finite region of the graph. Quantum walks can also exhibit a type of localization, in the presence of percolation, called Anderson localization. In the case of Anderson localization the localization occurs due to destructive interference and the quantum state decreases exponentially away from the localized region [26]. Furthermore, in the context of quantum walks, percolation has been found to act as a decoherence model [27][28]. Decoherence, characterized by the loss of quantum coherence and interference, causes the quantum walk to exhibit increasingly classical behavior as the degree of percolation increases.

Since the performance of quantum-walk-based search algorithms (QWSAs) is closely related to the properties of the underlying QWs, this raises questions about how QWSAs perform in percolated environments. This thesis studies the performance of QWSAs in finite two-dimensional rectangular lattices and percolated finite two-dimensional square lattices, both with periodic boundary conditions.

The performance of the QWSAs is measured by looking at two key performance indicators: the success probability and the optimal runtime. The thesis combines both analytical asymptotical approximations and numerical simulations. For unpercolated grids, the runtime and success probability can be approximated using the principal eigenvalue technique, which analyzes the dominant eigenvalues of the QWSA evolution operator. These asymptotic predictions perform well for square lattices, but are less accurate on rectangular lattices due to the influence of higher-frequency spectral components, which introduce beat frequencies in the quantum walk. As a result, rectangular grids exhibit a shifted optimal runtime and a reduced success probabilities compared to the square grids.

To model environmental noise or dynamic structural changes, bond percolation is introduced in two regimes: static percolation (fixed structure) and dynamic percolation (structure fluctuates at every timestep). Dynamic percolation significantly reduces QWSA performance, reducing the success probability by over 80% at percolation parameter $p = 0.95$, especially for larger grids that require longer coherence times. In contrast, static percolation degrades performance more gradually, and the success probability remains high until the percolation threshold is approached ($p \approx 0.5$), where the lattice loses global connectivity. Also notable is the fact that under static percolation, the optimal runtime increases with decreasing percolation parameter p, until it reaches the critical phase, after which it rapidly decreases again due to reduced connected regions.

In total this work is further proof of how percolation models can be used to systematically degrade QWSAs, providing a controlled way to study their robustness and the role of coherence. The analysis of QWSA performance applied here could be extended to other lattice geometries, percolation types, or quantum-walk-based algorithms in future work.

Chapter 2

Theory and Methods

2.1 Performing a discrete quantum walk (DQW)

A discrete-time quantum walk (DQW) is the quantum analogue of a classical random walk, in which the walker evolves according to both a "coin toss" and a shift. In the classical case, a coin flip determines the direction of motion at each time step. Analogously, a DQW requires two operators at each step: a *coin operator* C and a *shift operator* S, which moves the walker based on the coin outcome. The Hilbert space of the quantum walker is given by $\mathcal{H} = \mathcal{H}_C \otimes \mathcal{H}_P$, where $\mathcal{H}_C$ is the coin space and $\mathcal{H}_P$ is the position space.

For a one-dimensional walk with periodic boundary conditions and P positions, the coin space for each position is 2-dimensional, encoding left and right directions, and the position space is P-dimensional. The total Hilbert space then has dimension $2P$. A single time step is implemented by the unitary operator

$$U = S(I_P \otimes C), \tag{2.1}$$

where I_P is the identity operator on the position space. The coin operator is applied independently at each site, followed by a shift conditioned on the coin state.

In this thesis, the focus lies on quantum walks performed on finite two-dimensional rectangular lattice of size $N \times M$, with periodic boundary conditions. In this case, the coin space for each position becomes 4-dimensional to encode the directions `up`, `down`, `left`, and `right`. The position space becomes NM-dimensional, and the total Hilbert space is thus $4NM$-dimensional.

2.2 Grover's algorithm as a quantum walk

2.2.1 Grover's algorithm

Grover's algorithm is a quantum search algorithm designed to locate a marked item among N possibilities with a quadratic speedup over classical search. The algorithm operates in an N-dimensional Hilbert space and relies on two unitary operators: a *search oracle* and an *inversion-about-the-mean* operator. From a quantum mechanical perspective, the objective of Grover's algorithm can be stated as follows: the goal is to maximize the amplitude of the marked element basis state $|x_o\rangle$ in the system's quantum state. The search oracle is defined as:

$$R_f = I - 2\,|x_o\rangle\,\langle x_o|\,, \tag{2.2}$$

which reflects the state about the linear subspace orthogonal to $|x_o\rangle$. All amplitudes remain unchanged, except for the marked element component $|x_o\rangle$ which undergoes a sign inversion.

The *inversion-about-the-mean* operator is given by:

$$R_D = 2\,|D\rangle\,\langle D| - I, \tag{2.3}$$

where $|D\rangle$ is the uniform superposition over all basis states:

$$|D\rangle = \frac{1}{\sqrt{N}}\sum_{x=1}^{N}|x\rangle\,. \tag{2.4}$$

A single full Grover iteration, or step operator, is then defined as:

$$U = R_D R_f. \tag{2.5}$$

The Grover algorithm can be visualized, as illustrated in Figure 2.1, in a single plane. This is due to the fact that the wave vector remains in the same plane during the whole process, specifically the plane spanned by the marked wave vector and the uniform superposition vector.

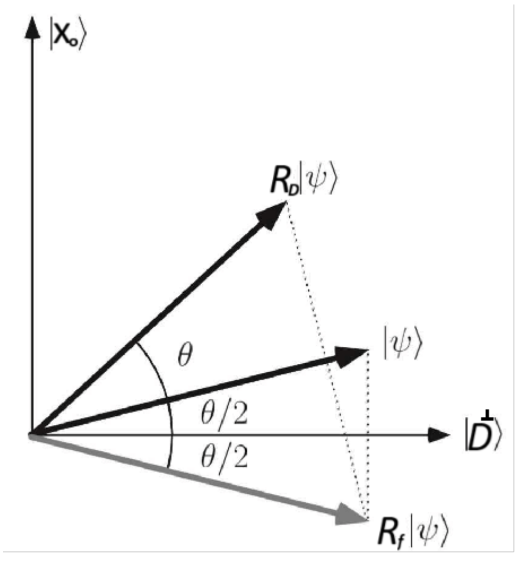

Figure 2.1: Geometric representation of a single Grover iteration. θ is defined by $\sin(\theta) = \frac{1}{N}$. The initial state ψ is taken as the uniform superposition state $|D\rangle$ and $|D^{\perp}\rangle$ is the normalized vector of $|D\rangle$ projected on the linear subspace orthogonal to $|x_o\rangle$.

2.2.2 QWSA

Quantum walks provide a natural framework to implement search algorithms on structures, such as graphs and lattices. In the coined discrete-time quantum walk model, the evolution of the walker is governed by two operators: a unitary coin operator C acting on an internal coin space, and a shift operator S that moves the walker across the graph based on the coin state. The main constraints are that C must be unitary and S must be local.

To incorporate Grover's algorithm into a quantum walk, the Grover coin is used as the coin operator [21]. For a d-regular graph, the Grover coin is defined as

$$G = 2\left|D\right\rangle\left\langle D\right| - I, \tag{2.6}$$

where $|D\rangle$ is the uniform superposition over the d coin basis states:

$$\left|D\right\rangle = \frac{1}{\sqrt{d}}\sum_{j=1}^{d}\left|j\right\rangle . \tag{2.7}$$

Explicitly, G is a $d \times d$ matrix in which the diagonal entries are $\frac{2}{d} - 1$ and the off-diagonal entries are $\frac{2}{d}$. This coin reflects any state about the uniform direction $|D\rangle$.

In the case of a finite two-dimensional rectangular lattice with periodic boundary conditions, each vertex is connected to its four nearest neighbors: up, down, left, and right. This makes the lattice a 4-regular graph, and thus $d = 4$ in the Grover coin.

The shift operator used is the *flip-flop shift* operator. This operator moves the walker to a neighboring site in the direction indicated by the coin state and simultaneously flips the coin state to its opposite. The flip-flop shift operator allows the amplitudes to convergence faster toward the marked state in the QWSA [29]. For a 2D grid, the action of S on position and coin states is given by:

$$S\left|x, y\right\rangle\left|r\right\rangle = \left|x+1, y\right\rangle\left|l\right\rangle , \tag{2.8}$$

$$S\left|x, y\right\rangle\left|l\right\rangle = \left|x-1, y\right\rangle\left|r\right\rangle , \tag{2.9}$$

$$S\left|x, y\right\rangle\left|u\right\rangle = \left|x, y+1\right\rangle\left|d\right\rangle , \tag{2.10}$$

$$S\left|x, y\right\rangle\left|d\right\rangle = \left|x, y-1\right\rangle\left|u\right\rangle . \tag{2.11}$$

In a finite $N \times M$ lattice with periodic boundary conditions, the operations $x \pm 1$ and $y \pm 1$ are interpreted modulo N and M, respectively.

The full step operator for the quantum walk is then given by:

$$U = S(I_P \otimes G), \tag{2.12}$$

where I_P is the identity operator on the position space, and G is the Grover coin.

Now only the search oracle still needs to be added to actually perform the search. In the coined model, the oracle must act on the full Hilbert space, including the coin space. The oracle operator is defined as:

$$R = (I_P - 2\left|x_o\right\rangle\left\langle x_o\right|) \otimes I_C, \tag{2.13}$$

where I_C is the identity operator on the coin space. This operator inverts the sign of the amplitude at the marked vertex for all coin states.

The combined step operator for the QWSA is then:

$$U' = UR. \tag{2.14}$$

A specific initial state is also required. Due to the symmetry of the 2D rectangular lattice with periodic boundary conditions, every site has the same environment and is therefore physically equivalent to any other position. This makes the natural choice for the initial state of the system:

$$\left|\psi(0)\right\rangle = \left|D_P\right\rangle \otimes \left|D_C\right\rangle, \tag{2.15}$$

where $\left|D_P\right\rangle$ is the uniform superposition over all NM positions and $\left|D_C\right\rangle$ is the uniform superposition over the coin states.

2.3 Computing optimal runtime and success probability

An important part of QWSAs is that they need to be stopped at the correct time so as to maximize the amplitude of the correct node. If it is stopped too early then the marked node amplitude has not reached its maximum value yet, but if it goes longer than the optimal runtime then the marked node amplitude starts to diminish due to the oscillatory nature of the quantum evolution in QWSAs [30].

To determine the optimal runtime and corresponding success probability asymptotically, one can employ the *principal eigenvalue technique* [31], which utilizes a partial spectral decomposition of the step evolution operator U'. While the full spectrum of U' may be difficult to compute analytically for arbitrary lattice sizes, this is often doable for the spectrum of the unperturbed operator U (without the search oracle), and can be used as the basis for approximation.

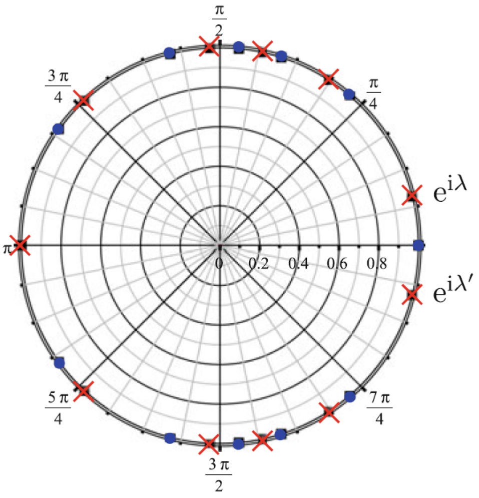

Figure 2.2: The blue dots are the eigenvalues of U and the red crosses are eigenvalues of U', for a 5x5 square lattice.

Let U' be the total step operator of the QWSA, and suppose $e^{i\lambda}$ is the eigenvalue of U' with the smallest $\lambda > 0$ and $e^{i\lambda'}$ the eigenvalue with the largest $\lambda' < 0$. Let $|\lambda\rangle$ and $|\lambda'\rangle$ denote the corresponding eigenvectors. Since U' is real and unitary, all eigenvalues lie on the unit circle and appear in complex-conjugate pairs; hence $\lambda' = -\lambda$. An example is given in Figure 2.2, which shows the eigenvalues of U (blue dots) and U' (red crosses) for a 5x5 square lattice.

Note that the partial spectral decomposition of the operator U' can be written as:

$$U' = e^{i\lambda}|\lambda\rangle\langle\lambda| + e^{-i\lambda}|\lambda'\rangle\langle\lambda'| + U_{\text{res}}, \quad (2.16)$$

with U_{res} the component of U' that only acts on the linear subspace orthogonal to $|\lambda\rangle$ and $|\lambda'\rangle$.

The probability $p(t)$ of measuring the marked node $|x_o\rangle$ at time t is then:

$$p(t) = |\langle x_o|(U')^t|\psi(0)\rangle|^2 = |e^{it\lambda}\langle x_o|\lambda\rangle\langle\lambda|\psi(0)\rangle + e^{it\lambda'}\langle x_o|\lambda'\rangle\langle\lambda'|\psi(0)\rangle + \epsilon|^2 \quad (2.17)$$

where $\epsilon = \langle x_o|(U_{res})^t|\psi(0)\rangle$.

Under suitable limit conditions, it can be shown that $\epsilon(t)$ becomes negligible as lattice size goes to infinity, in Appendix C it is proven that $\epsilon \to 0$ as $N, M \to \infty$ for a 2D rectangular lattice.

To evaluate Equation 2.17, it suffices to compute the following quantities:

- $\langle x_o|\lambda\rangle$, the projection of the marked state onto the principal eigenvector,
- $\langle\lambda|\psi(0)\rangle$, the overlap of the initial state with the eigenvector,
- The associated phase λ.
- And the corresponding quantities for λ'.

Following the method from Portugal [31], the above values can be related to the spectrum of the unperturbed operator U using certain parameters. Because U is real it has complex conjugate pair eigenvalues, the method then requires the two parameters:

$$A = 2\sum_{\phi_k=0} |\langle x_o|\psi_k\rangle|^2, \tag{2.18}$$

$$C = \sum_{\phi_k\neq 0} \frac{|\langle x_o|\psi_k\rangle|^2}{1-\cos(\phi_k)}. \tag{2.19}$$

Here, the ψ_k form an eigenbasis of U, and ϕ_k are the complex arguments of the associated eigenvectors. A is a sum that only uses zero eigenvalues and C only non-zero eigenvalues. This distinction arises in the derivation of λ, which is also where A and C come from. The zero eigenvalues allow the use of a Maclaurin expansion that is different from the Taylor expansion required for non-zero eigenvalues, resulting in two distinct sums that have to be calculated. Moreover, the Taylor expansions require that, for sufficiently big N, M, $\lambda << \phi_{min}$ with ϕ_{min} the smallest positive argument from the eigenvalues of U, this is proven in Appendix B.

The method from Portugal then states that, using A and C, the marked node probability becomes:

$$p(t) = \frac{\langle\psi(0)|x_o\rangle^2}{AC}\sin^2(\sqrt{(AC)}t) \tag{2.20}$$

The probability amplitude is maximal when:

$$t = \frac{\pi\sqrt{C}}{2\sqrt{A}}. \tag{2.21}$$

Since this QWSA employs a discrete quantum walk, the runtime from Equation 2.21 has to be rounded up or down when used for a QWSA run. Rounding Equation 2.21 should produce a good approximation for the runtime at which the marked node probability attains a maximum. The maximum probability amplitude for successfully finding the marked node, at the runtime given by Equation 2.21, is approximately:

$$p_{succ} = \frac{|\langle\psi(0)|x_o\rangle|^2}{AC}. \tag{2.22}$$

The last step to finding a formula for t_{opt} and p_{succ} in terms of N and M is to evaluate, or more specifically find bounds for, A and C. First notice that the only eigenvector of U that has non-zero overlap with $|x_o\rangle$ is the uniform superposition state, so

$$A = \frac{2}{\#nodes} = \frac{2}{NM} \tag{2.23}$$

The inner products in the sum C can be determined using the eigenvectors for an $N \times M$ grid (see Appendix A). Also ϕ_k can be found in the same section, C can then be reduced to:

$$C = \frac{1}{NM}\sum_{\substack{k,l=0\\(k,l)\neq(0,0)}}^{N-1,M-1} \frac{1}{1-\frac{1}{2}(\cos(\frac{2\pi k}{N})+\cos(\frac{2\pi l}{M}))} = \frac{1}{NM} \times S_{NM} \tag{2.24}$$

The identity $\frac{1-cos(a)}{2} = \sin^2(\frac{a}{2})$ and the fact that $\sin^2(x)$ is symmetric around $\frac{\pi}{2}$ combine to produce:

$$S_{NM} = \sum_{\substack{k,l=0\\(k,l)\neq(0,0)}}^{N-1,M-1} \frac{1}{\sin^2(\frac{\pi k}{N})+\sin^2(\frac{\pi l}{M})} \tag{2.25}$$

$$= 4 \times \sum_{k,l=1}^{\lfloor\frac{N}{2}\rfloor,\lfloor\frac{M}{2}\rfloor} \frac{1}{\sin^2(\frac{\pi k}{N})+\sin^2(\frac{\pi l}{M})} + 2 \times (\sum_{k=1}^{\lfloor\frac{N}{2}\rfloor}\frac{1}{\sin^2(\frac{\pi k}{N})} + \sum_{l=1}^{\lfloor\frac{M}{2}\rfloor}\frac{1}{\sin^2(\frac{\pi l}{M})}) \tag{2.26}$$

The following identity will be used to find bounds for the sums in Equation 2.25:

$$\frac{4a^2}{\pi^2} \leq \sin^2(a) \leq a^2 \tag{2.27}$$

Bounds for the term in brackets are found, using the inequalities from Equation 2.27 (details in Appendix D):

$$\frac{1}{\pi^2}(N^2+M^2) \leq \sum_{k=1}^{\lfloor\frac{N}{2}\rfloor} \frac{1}{\sin^2(\frac{\pi k}{N})} + \sum_{l=1}^{\lfloor\frac{M}{2}\rfloor} \frac{1}{\sin^2(\frac{\pi l}{M})} \leq \frac{\pi^2}{24}(N^2+M^2) \tag{2.28}$$

The double sum in Equation 2.25 is first bounded by using Equation 2.27 again. Afterwards, integral approximations are used for these bounds. These steps result in the following bounds:

$$\frac{1}{\pi^2}\int_1^{\frac{M}{2}}\int_1^{\frac{N}{2}} \frac{1}{(\frac{x}{N})^2+(\frac{y}{M})^2}dxdy \leq \sum_{k,l=1}^{\lfloor\frac{N}{2}\rfloor,\lfloor\frac{M}{2}\rfloor} \frac{1}{\sin^2(\frac{\pi k}{N})+\sin^2(\frac{\pi l}{M})} \leq \frac{1}{4}\int_{\frac{1}{2}}^{\frac{M}{2}}\int_{\frac{1}{2}}^{\frac{N}{2}} \frac{1}{(\frac{x}{N})^2+(\frac{y}{M})^2}dxdy. \tag{2.29}$$

These integrals can be solved analytically (see Appendix E) combining this result with Equation 2.28 produces the asymptotic bounds for S_{NM}:

$$\frac{1}{\pi}NM(\ln(NM)+D+\mathcal{O}(\frac{1}{N}+\frac{1}{M})) \leq S_{NM} \leq \frac{\pi}{4}NM(\ln(NM)+E+\mathcal{O}(\frac{1}{N}+\frac{1}{M})) \tag{2.30}$$

with D, E constants that depend on the ratio $\frac{N}{M}$. So in the limit of $N, M \to \infty$: $C = k_1(\ln(NM)+k_2)$.

Finally, substituting A and C into the previously obtained Equation 2.21, the approximation for the optimal runtime for an $N \times M$ grid is:

$$t_{opt} = \frac{\pi\sqrt{k_1 NM(\ln(NM)+k_2)}}{2\sqrt{2}} \tag{2.31}$$

with k_1 and k_2 constants that depend on the geometry (specifically on the ratio between N and M). Since QWSA performs a discrete quantum walk, the time needs to be rounded to an integer when performing the QWSA. Now note that $|\langle\psi(0)|x_0\rangle|^2 = \frac{1}{NM}$ since $\psi(0)$ is the uniform superposition state. Then the corresponding success probability is:

$$p_{succ} = \frac{1}{2k_1(\ln(NM)+k_2)}. \tag{2.32}$$

The asymptotic approximation of the optimal runtime, Equation 2.31, contains two undetermined constants. The sum of C will be determined exactly for several geometries and sizes, these values can be used to determine the runtime via Equation 2.21. Then the formula from Equation 2.31 will be fitted through these runtimes to obtain the parameters k_1 and k_2. The values from the principal eigenvalue technique obtained by exactly evaluating the sum C and the fitted formula will be compared to the actual optimal runtimes determined by performing quantum walks.

Moreover, given the parameters k_1 and k_2 also fix the Equation 2.31. This formula will also be compared to the maximum probabilities determined by performing the corresponding quantum walks in subsection 3.2.1.

2.4 Percolation

Percolation is a way to model connectivity or variation in a graph [32][33]. There are two types of percolation: site percolation and bond percolation. In site percolation each node of the graph is open with probability p and closed with probability $1-p$. Bond percolation does the same with the edges; a subgraph is chosen where each bond independently is either present with probability p or missing with probability $1-p$. The probabiliy p that each edge has of being present is called the percolation parameter. In this thesis only bond percolation is applied.

These models bring an element of uncertainty and variation into the graph. But connectivity also is an issue when some links are missing, as even a connected graph can become disconnected when percolation is applied. It turns out that such a percolated graph has three distinct connectivity phases dependent on the percolation parameter [32]. The supercritical phase, $p > 0.5$ for infinite square graphs, is characterized by large scale connectivity. To be precise, the probability that a node is a part of an infinite connected cluster (connected group of nodes) is equal to one. In the subcritical phase, $p < 0.5$ for infinite square graphs, there is no infinite cluster present. This means that all clusters are finite and connectivity is only local. The critical phase, $p = 0.5$ for an infinite square graphs, has clusters of all sizes but no infinite cluster. This means that the system is at a transition point, which is why the percolation parameter $p_c = 0.5$ is called the percolation threshold or critical percolation parameter [34]. In the case of finite graphs defining the critical percolation becomes more subtle. The most important result is that the critical phase becomes smeared out, for a finite square lattice (side length L) the critical regime becomes $p_c \pm \Delta p$ still with $p_c = 0.5$ and Δp scales with $L^{-\frac{3}{4}}$ [35][36].

In the context of a QWSA, percolation can be applied at various moments. For instance the graph could be percolated only once at the beginning (static percolation) or at every timestep the graph is percolated again (dynamic percolation). In the context of quantum walks, percolation can be seen as a decoherence model [37] [27]. Static percolation can be interpreted as a siuation where the change in time of the graph takes much more time than the runtime of the QWSA, so a decoherence time that is long compared to the required coherence time. And dynamic percolation can be seen as a situation where the change in time of the graph takes much less time than the timestep of the QWSA, in this case the decohorence time is short compared to the required coherence time.

Quantum walks, including quantum-walk-based search algorithms (QWSAs), evolve unitarily and deterministically: given a specific initial state, the final state is entirely determined by the applied operators. In such systems, coherence and interference play essential roles in enabling quantum speed-up over classical algorithms [28]. When percolation is added as a means to study the impact of disorder on QWSAs, then there is a stochastic element in the evolution of the system. This thesis examines the influence of bond percolation by performing multiple quantum walk simulations on percolated 2D rectangular lattices. This will be done in both the static percolation domain and the dynamic percolation domain. These simulations are then used to determine how the success probability after runtime Equation 2.31 changes as a function of the percolation parameters. Moreover, the runtime at which the average walk, of a certain percolation parameter value, attains its maximum probability is plotted against the percolation parameter.

Chapter 3

Results and Discussion

This section is largely split into two parts. Part 1 is made up of: section 3.1 and section 3.2. Part 2 comprises: section 3.3, section 3.4 and section 3.5. Part 1 compares the principal eigenvalue technique results from section 3.1 with data from unpercolated grid QWSA simulations for different grid dimensions. Part 2 is an analysis of the influence of percolation on QWSAs.

Both parts begin with the analysis of single QWSA simulation runs providing results that help in later discussions; section 3.1 contains unpercolated QWSA runs and section 3.3 cotainsstatically percolated QWSA runs. Furthermore, part 1 also compares the results from the different grid dimensions, $\frac{M}{N} = 1$ and $\frac{M}{N} = 3$, with each other. Similarly, part 2 also compares the influences of dynamic percolation compared to static percolation. Only the last section of part 2, section 3.5, exclusively looks at static percolation. The central node in the grid is the marked node in all simulations. Note, however, that due to the periodic boundary conditions all nodes are physically equivalent and hence the results should be the same irrespective of what node is chosen as the marked node. Moreover, the runtime of the QWSA refers to the number of discrete steps of the QWSA and hence is a dimensionless quantity (in the figures [steps] will be used as the dimension). The temporal frequency (with dimension $[f] = [\frac{1}{\text{runtime}}]) = \frac{1}{\text{steps}}$) of section 3.1 is therefore also a dimensionless quantity.

3.1 Analysis of unpercolated QWSA runs

This section presents an analysis of unpercolated QWSA runs to provide insight into the algorithm's behavior and support the discussions in later sections. The grid sizes chosen in this section have dimensions $1:1$ and $1:3$, and the number of nodes across the different grids is approximately the same; $29 \times 29 = 841$ and $17 \times 51 = 867$. The section will first point out notable observation and then in the last paragraph an explanation is discussed.

Figure 3.1 shows a single run for a square (29x29) grid, the runtime $t = 96$ steps corresponds to approximately one "period" of the walk. The QWSA simulation exhibits a high-frequency component that initially resembles noise. However, it corresponds to a period-2 oscillation, a known characteristic of most coin-based discrete-time quantum walks. Another noticeable part of the run is the shape of the peak. Instead of being rounded like a sine function as might be expected based on Equation 2.27 the top of the graph is more like a plateau; staying constant for a while.

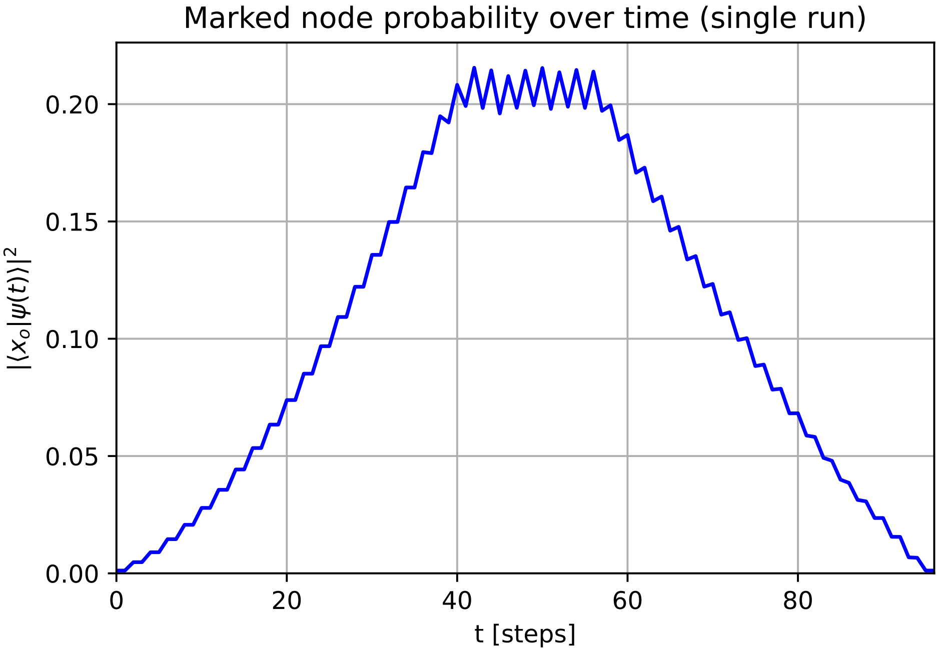

FIGURE 3.1: A single run of a QWSA simulation on an unpercolated 29x29 grid with runtime $t = 96$ steps.

Figure 3.2 shows a single run for a rectangular (17×51) grid. Here, a longer runtime $t = 1050$ (dimensionless time steps) is used to reveal more of the periodic structure. Note that, unlike the square grid, the first peak does not plateau. However, the first maximum is achieved slightly to the right of the center of the peak. This plot also has more unexpected characteristics such as the splitting of the peak at $t = 500$ steps and the variation in the peak height, which itself almost seems modulated by a lower frequency wave.

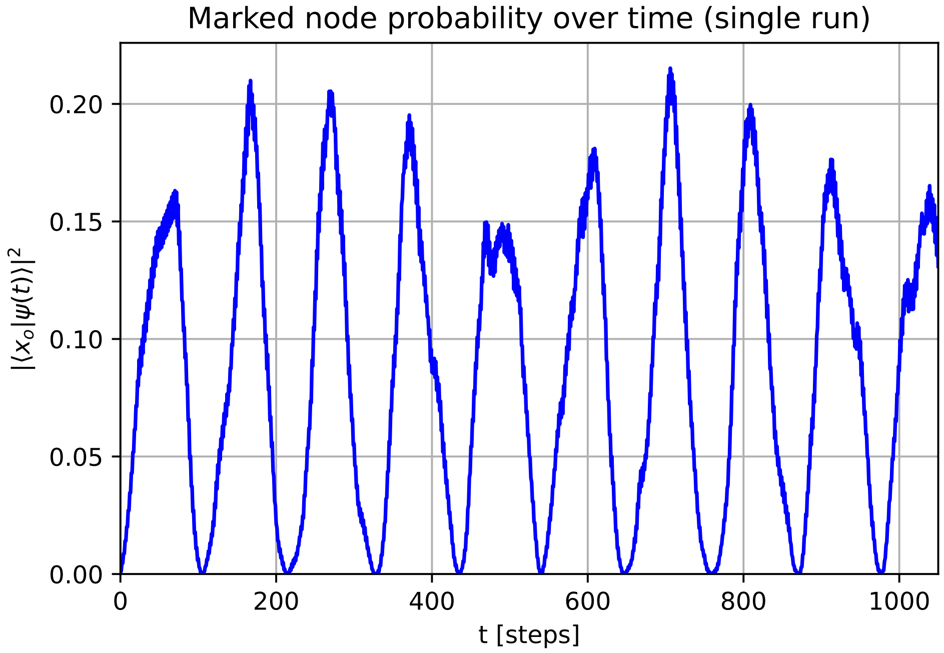

FIGURE 3.2: A single run of a QWSA simulation on an unpercolated 17x51 grid with runtime $t = 1050$ steps.

The square grid also exhibits these variations in the peak height, but the square grid peak oscillations are significantly smaller than in the rectangular grid case.

The frequency decompositions of these QWSAs, Figure 3.3 and Figure 3.4, help explain these phenoma. The small peaks at frequencies -0.5 and 0.5, correspond to the period 2 component typical of coin-based discrete time quantum walks. While both spectra contain higher frequency components, both frequency decompositions clearly have a dominant non-zero term. However, due to the broken symmetry in the rectangular grid, Figure 3.4 contains non-dominant frequencies with larger amplitudes than those in Figure 3.3. This explains why the 29x29 QWSA marked node probabilities have a smaller variation in peak height than the 17x51 QWSA marked node probabilities. Moreover, the plateau observed in Figure 3.1 can also be attributed to these higher-frequency components; they interfere constructively in such a way that the peak becomes flattened. The modulation of peak heights in the QWSA probability signal arises from interference between the dominant low-frequency component and the higher-frequency components, for instance in Figure 3.4 the frequency with approximate magnitude of 0.01 $\frac{1}{\text{steps}}$ is in the $N \times 3N$ grid the most important higher frequency. These additional components create beat frequencies, a type of lower frequency that occurs when two or more frequencies are superimposed. Beat frequencies modulate the amplitude of the main oscillation, resulting in an envelope that causes periodic variation in peak heights. This is a phenomen that has been observed in quantum walks [38].

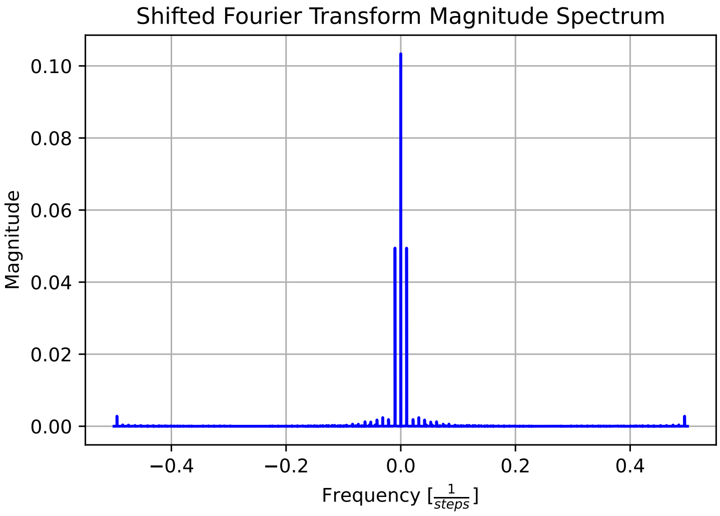

FIGURE 3.3: The figure depicts the magnitude frequency decomposition for the marked node probability of a QWSA with grid size 29x29. Runtime $t = 100000$ steps so as to get a high resolution decomposition. The highest non-zero peaks correspond to the frequencies: -0.0101 $\frac{1}{\text{steps}}$ and 0.0101 $\frac{1}{\text{steps}}$.

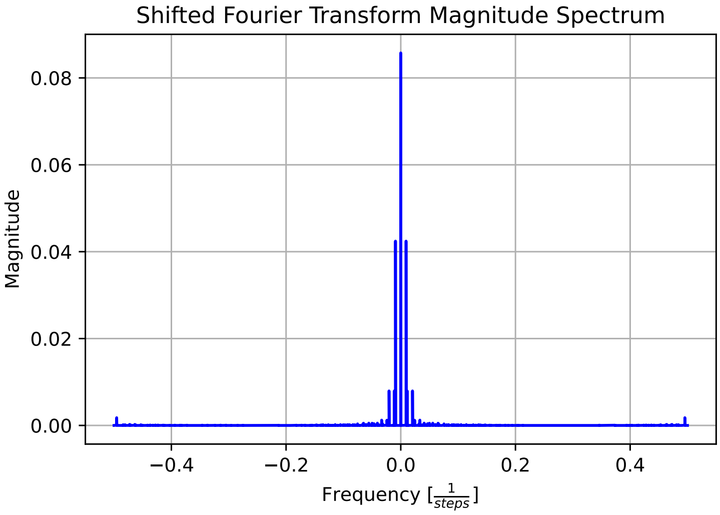

FIGURE 3.4: The figure depicts the magnitude frequency decomposition for the marked node probability of a QWSA with grid size 17x51. Runtime $t = 100000$ steps so as to get a high resolution decomposition. The highest non-zero peaks correspond to the frequencies: -0.0092 $\frac{1}{\text{steps}}$ and 0.0092 $\frac{1}{\text{steps}}$.

3.2 Comparison of asymptotic results with QWSA simulations

3.2.1 Optimal runtime

The asymptotic formula for the optimal runtime, given in Equation 2.31, is compared here against simulation results. This formula includes two constants, k_1 and k_2, which depend on the grid's aspect ratio $\frac{M}{N}$. These constants are obtained by fitting the expression in Equation 2.31 to runtimes determined by computing Equation 2.18 and using these in Equation 2.21. The constants are determined for two aspect ratios: $\frac{M}{N} = 1$ and $\frac{M}{N} = 3$.

For the square grid, the runtime predicted by the principal eigenvalue method appears to agree well with the QWSA simulations. However, there is a noticeable gap between the optimal runtimes observed in the simulations at $N = 29$ and $N = 31$. Upon inspection of the respective QWSA simulations, it turns out that the gap can be explained by the plateauing behavior of square grid QWSA peaks, that was discussed in section 3.1. Up until $N = 29$ the maximum is achieved at the beginning of the plateau, whereas from $N = 31$ onward the maximum is achieved more at the end of the plateau.

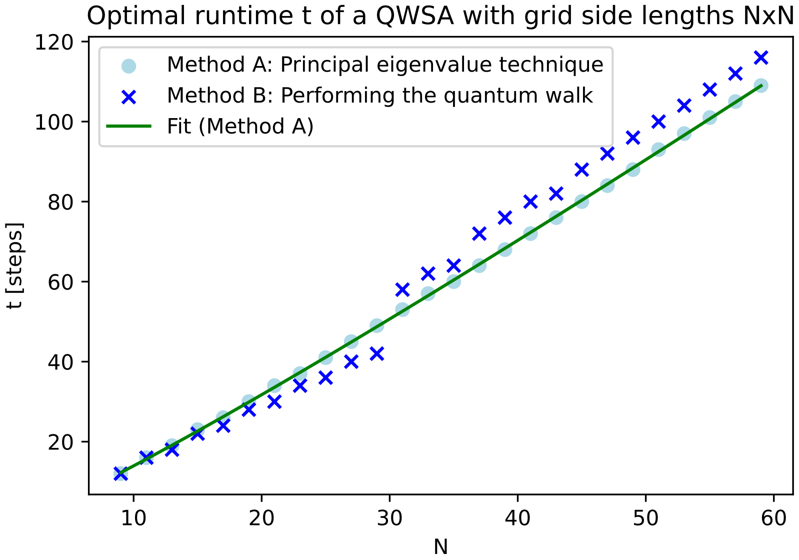

FIGURE 3.5: The figure shows the optimal runtime for a QWSAs with square grids obtained with two methods. Method A; the runtime obtained by exactly calculating the sums from Equation 2.18 and using this in the formula from Equation 2.21. Method B; the runtime where the probability amplitude of the marked node is maximal in a QWSA simulation. A fit of the formula from Equation 2.31 is made through the points obtained by method A, results of the fit: $k_1 = 0.335$ and $k_2 = 0.104$.

For the rectangular grid with aspect ratio 3, the optimal runtime observed in the QWSA simulations is consistently higher than the optimal runtime predicted by the principal eigenvalue method. This can be attributed to the presence of non-dominant higher-frequency components, as seen in Figure 3.4. Specifically, as depicted in Figure 3.2, these frequencies interfere in such a way that the peak is shifted to the right. As a result, the first maximum in the QWSA probability occurs later than would be predicted by considering only the dominant frequency.

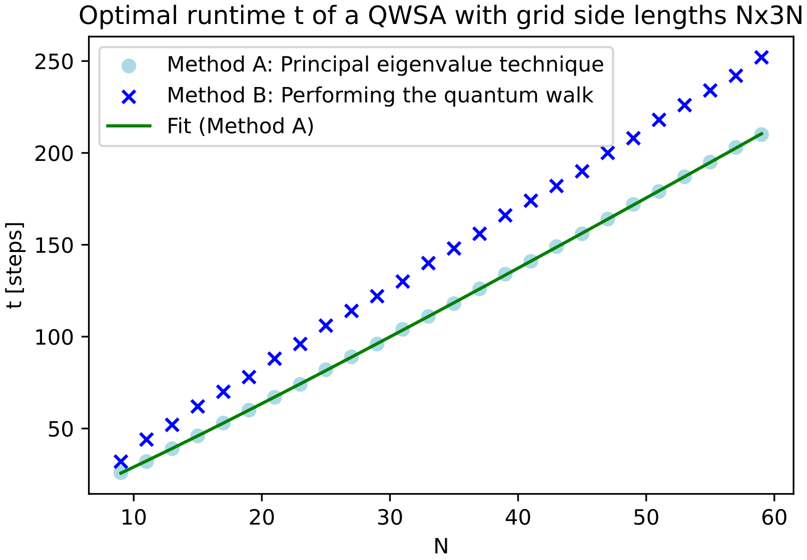

FIGURE 3.6: he figure shows the optimal runtime for QWSAs with rectangular grids ($N \times 3N$) obtained with two methods. Method A; the runtime obtained by exactly calculating the sums from Equation 2.18 and using this in the formula from Equation 2.21. Method B; the runtime where the probability amplitude of the marked node is maximal in a QWSA simulation. A fit of the formula from Equation 2.31 is made through the points obtained by method A, results of the fit: $k_1 = 0.327$ and $k_2 = 1.261$.

3.2.2 Success probability

This section compares the theoretical asymptotic success probability, given by Equation 2.32, with the success probability of a QWSA simulation using the runtime from Equation 2.31. The geometry-dependent parameters k_1 and k_2 were determined in subsection 3.2.1; see Figure 3.5 and Figure 3.6.

The results, shown in Figure 3.7 and Figure 3.8, support the previous findings.

In Figure 3.7, the curve for the theoretical approximation, using the fitting parameters from Figure 3.5, closely matches the values obtained from the QWSA simulations and lies slightly above them. This is expected, as the asymptotic formula in Equation 2.32 serves as an upper bound derived from dominant eigenvalues only.

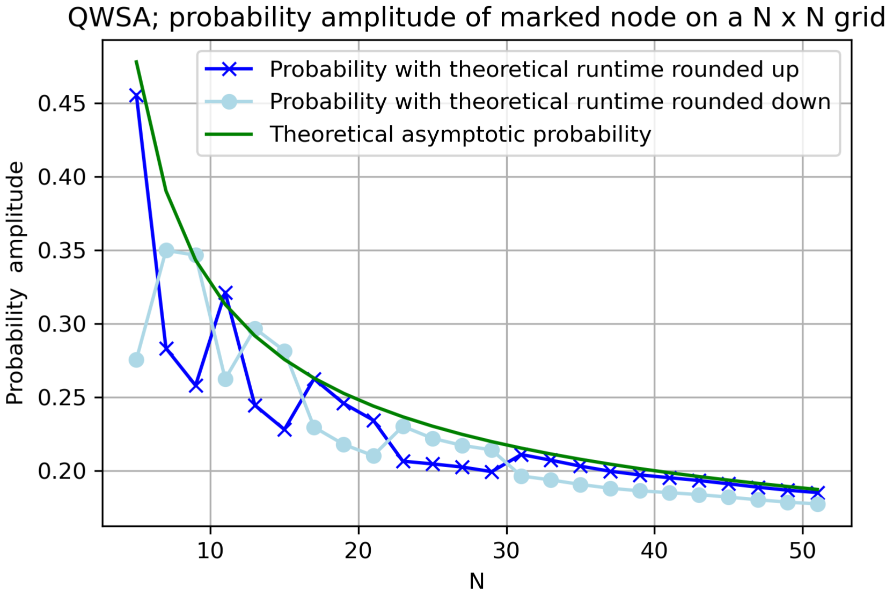

Figure 3.7: Graph with the probability amplitude of the marked node at the aymptotic optimal runtime given by Equation 2.31 compared with the success probability predicted by Equation 2.32 with fitting parameters k_1, k_2 from Figure 3.5.

Figure 3.8 for the Nx$3N$ gridsizes, on the other hand, shows a theoretical success probability with fitting parameters from Figure 3.2, that is significantly above the values obtained from the QWSA simulation.

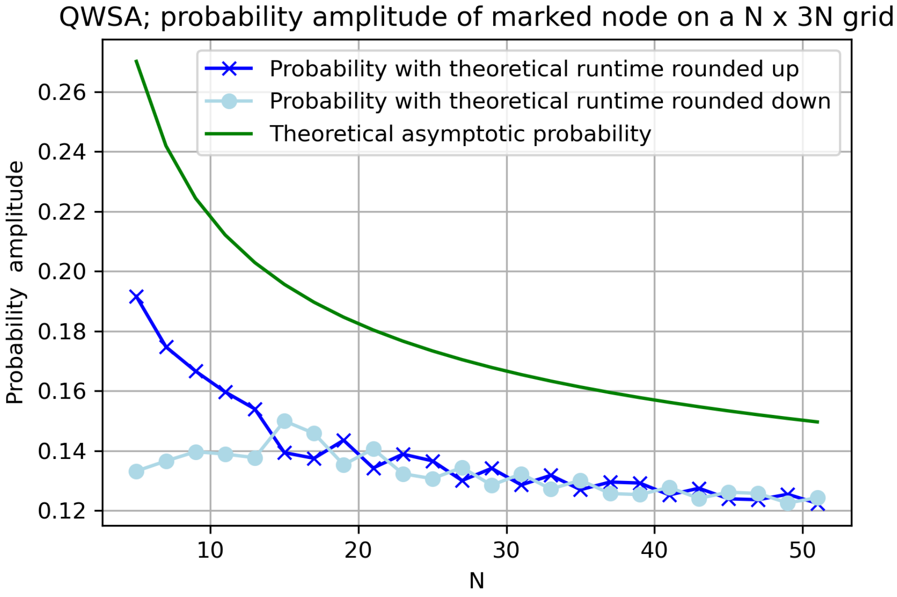

Figure 3.8: Graph with the probability amplitude of the marked node at the theoretical asymptotic optimal runtime given by Equation 2.31 compared with the success probability predicted by Equation 2.32 with fitting parameters k_1, k_2 from Figure 3.6.

This discrepancy can be explained by the fact that for $N \times 3N$ grids, the higher-frequency components have greater amplitudes than in square grids, as discussed in section 3.1. This results in a more complex wave pattern for the marked node probability, which systematically seems to lead to a smaller success probability as depicted in Figure 3.2.

For the same reason (that is, the presence of higher-frequency components with non-negligible amplitude) QWSAs on rectangular $N \times 3N$ grids are also expected to exhibit other peaks at later times where the actual success probability exceeds the theoretical prediction. Indeed, some of the peaks in Figure 3.2 are up to 10% higher than the 0.19 predicted that can be read from Figure 3.8.

3.3 Analysis of a statically percolated QWSA run

Figure 3.9 shows the marked node probability of a percolated 29×29 QWSA. Similar to Figure 3.2, but to a much greater extent, this plot reveals the presence of multiple frequency components in the percolated QWSA simulation. There are characteristics of the grid that help explain the presence of these additional frequencies with significant amplitudes.

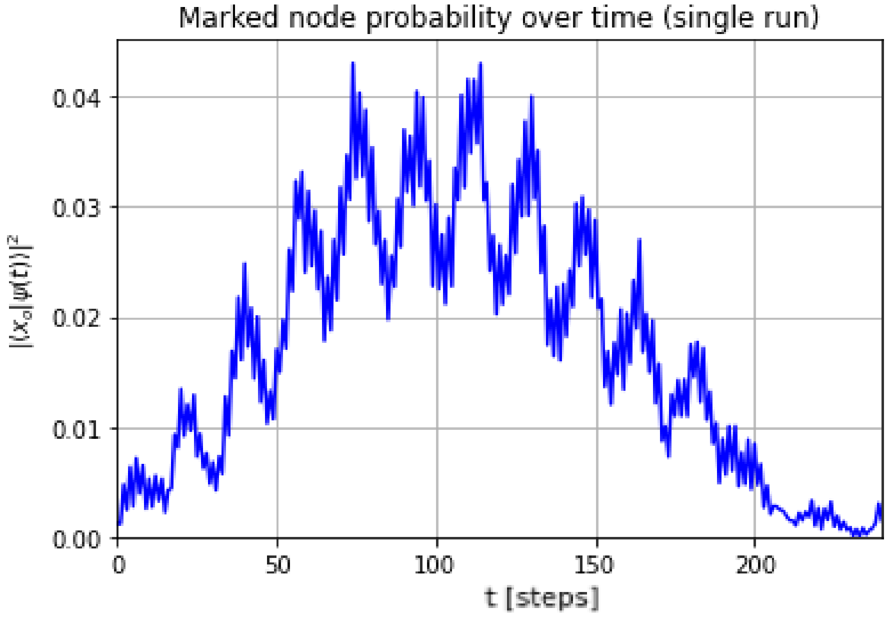

FIGURE 3.9: QWSA on a percolated 29x29 grid, Figure 3.10, with runtime $t = 240$ steps.

Figure 3.10 shows some regions that are only distantly connected to the marked node. For example, the top right and bottom left corners are only connected due to the periodic boundary conditions. Moreover, some paths form important connections from large areas to the marked node, such as the narrow path linking the marked node to the left side of the grid.

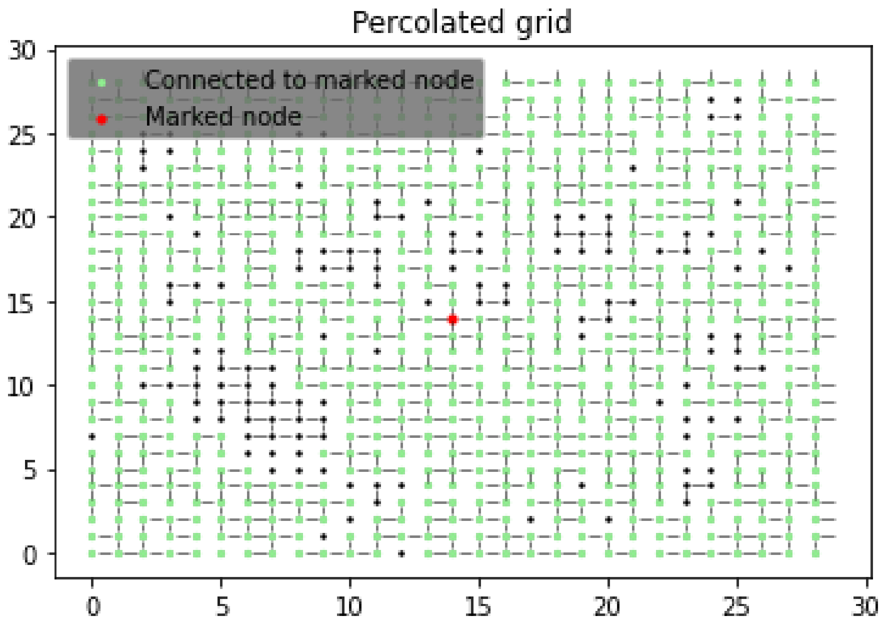

Figure 3.10: Percolated 29x29 grid; percolation parameter $\frac{1}{2}$.

Some of these features are also evident in the probability distributions shown in Figure 3.11, Figure 3.12, Figure 3.14 and Figure 3.15. In Figure 3.12. In Figure 3.12, the paths through which a significant part probability flows are visible as lighter regions, particularly the routes from the marked node to the left, upward, and toward the bottom right. The areas that are distantly connected to the marked node are illustrated well in Figure 3.15.

In this figure there are certain lighter areas, such as the bottom left and the top right, the same regions previously described as being weakly connected to the marked node. The bright yellow nodes in the bottom right depict a connecting path to the bottom left.

The spatial asymmetry, thus obtained, provides a possible explanation for the many frequencies present in the walk depicted in Figure 3.9, similar to the rectangular case compared to the square case. Furthermore, note how the maximum probability reached in this percolated 29x29 grid QWSA run, just above 0.04, is significantly lower than the maximum success probability reached in the unpercolated 29x29 grid QWSA run. This is to be expected: symmetry and connectivity of the unpercolated grid create a distinct dominant frequency in the QWSA, allowing a significantly greater portion of the amplitude to constructively interfere at the marked node at the same time. The disorder and asymmetry in the percolated grid, on the other hand, cause the amplitudes to disperse across multiple paths and frequencies making it less likely that a large portion of the probability converges at the marked node.

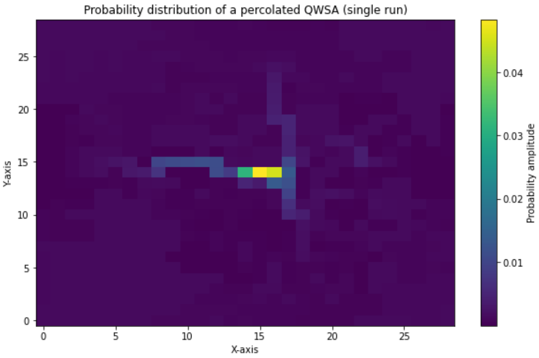

FIGURE 3.11: QWSA probability distribution at $t = 60$ steps of the run depicted in Figure 3.9.

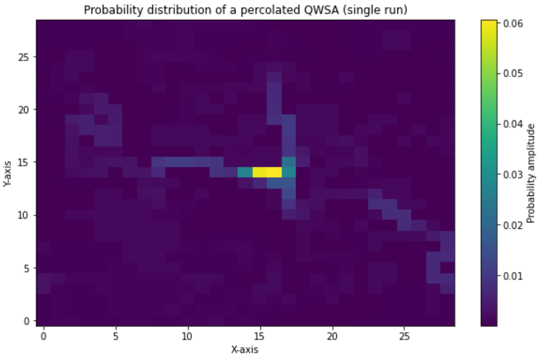

FIGURE 3.12: QWSA probability distribution at $t = 120$ steps of the run depicted in Figure 3.9.

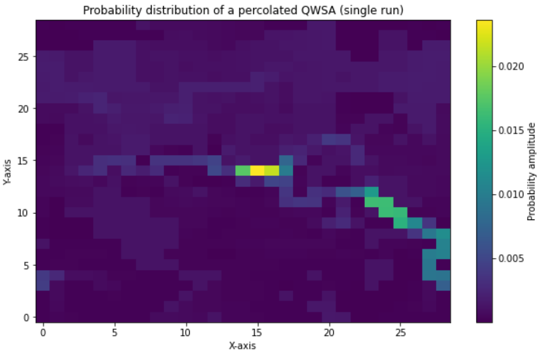

FIGURE 3.14: QWSA probability distribution at $t = 180$ steps of the run depicted in Figure 3.9.

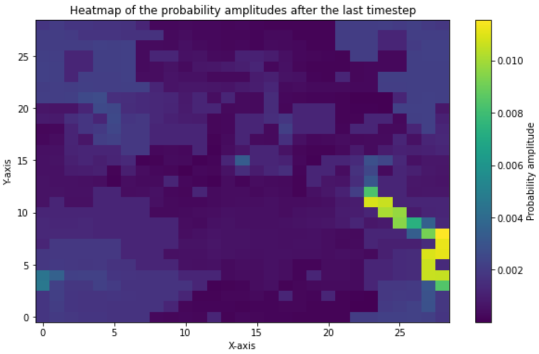

FIGURE 3.15: QWSA probability distribution at $t = 240$ steps of the run depicted in Figure 3.9.

3.4 Effect of percolation on QWSA success probability

3.4.1 Dynamic percolation

The success probability, with runtime Equation 2.31, is plotted against the percolation parameter in Figure 3.17. Figure 3.18 shows the maximum succes probability obtained during a run of $\lfloor 2.9 \times t_{opt} \rfloor$ timesteps. The figures are very similar; they both show a rapid decrease in success probability as the percolation parameter decreases.

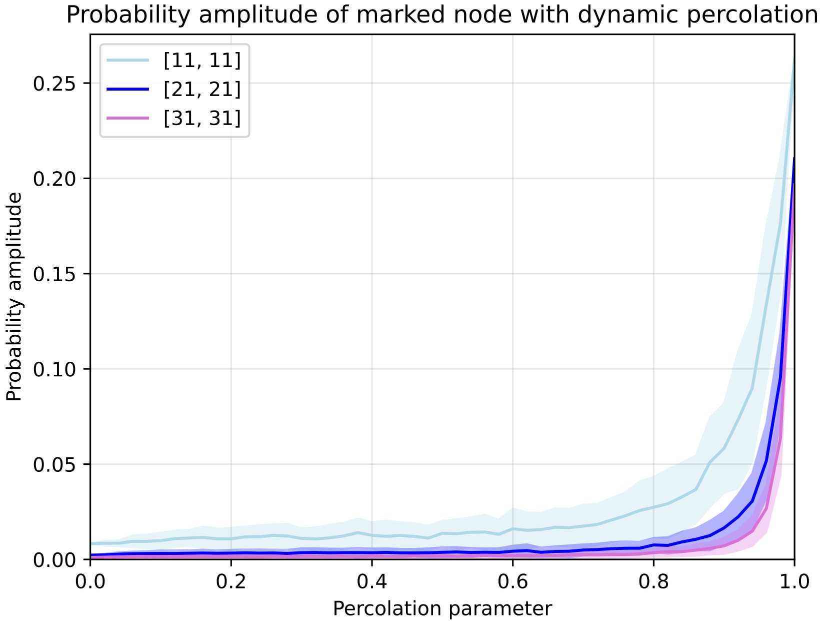

FIGURE 3.17: Mean and standard deviation (sample size $n = 100$) of the success probability for dynamically percolated QWSA simulations with runtime $\lfloor t_{opt} \rfloor$ for three grid sizes.

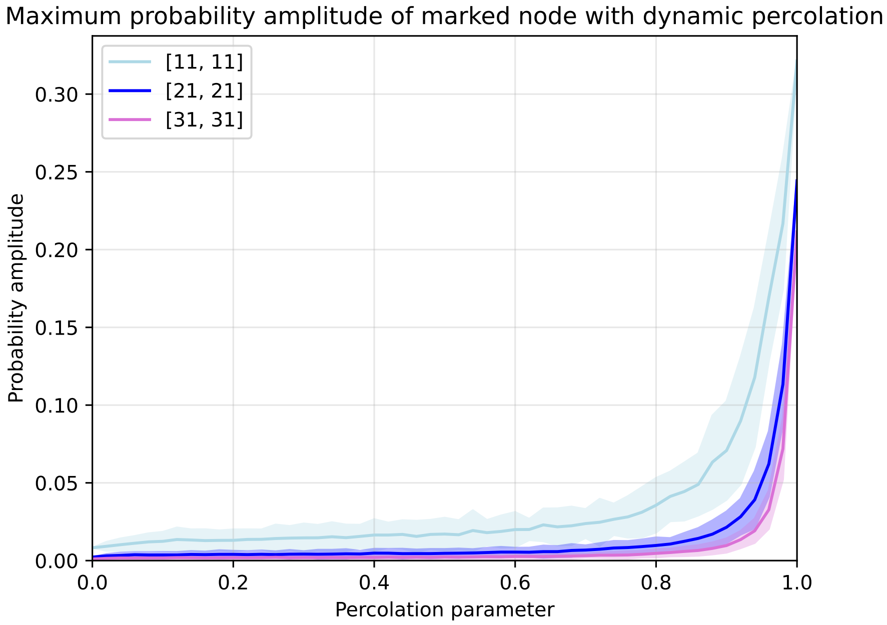

FIGURE 3.18: Mean and standard deviation (sample size $n = 100$) of the maximum marked node probability obtained from dynamically percolated QWSA simulations run for $\lfloor 2.9 \times t_{opt} \rfloor$ timesteps for three grid sizes.

Both figures reveal a sharp decline in success probability as the percolation parameter decreases. This is supported by the interpretation of dynamic percolation as a model for decoherence: when the percolation parameter is low, the graph connectivity changes rapidly, effectively scrambling quantum interference and severely diminishing the effectiveness of the QWSA. In the dynamic percolation model, the environment evolves on a much shorter timescale than the algorithm itself, mimicking strong environmental noise.

At a percolation parameter of 0.95, the success probability is reduced by more than 80%, clear evidence of the strong decoherence effect modelled by dynamic percolation. Furthermore, the impact appears to scale with system size: larger grids suffer a more pronounced decrease in success probability than smaller ones at the same percolation level. This could be explained by the fact that the QWSA has a longer optimal runtime on larger grids, effectively increasing the coherence time necessary for the QWSA to be performed, yet keeping the percolation and thus decoherence timescale the same.

3.4.2 Static percolation

In the static case, the success probability remains significantly closer to the unpercolated value across a wider range of percolation parameters compared to the dynamic case. This behavior is expected: unlike dynamic percolation, the static model represents a degraded but stable environment, where decoherence occurs on a much longer timescale than the QWSA runtime.

This means that while the success probability should decrease with a decreasing percolation parameter, indeed as is depicted in Figure 3.19 and Figure 3.20, it should decrease more gradually than in the dynamic case. It is really between percolation parameters 0.4 and 0.6 that a rapid decline in success probability occurs. This interval coincides with the percolation threshold for these grid geometries, where the graph begins to lose global connectivity. As a result of the loss of global connectivity, a significant portion of the amplitudes cannot reach the marked node, which drastically reduces the QWSA's success probability.

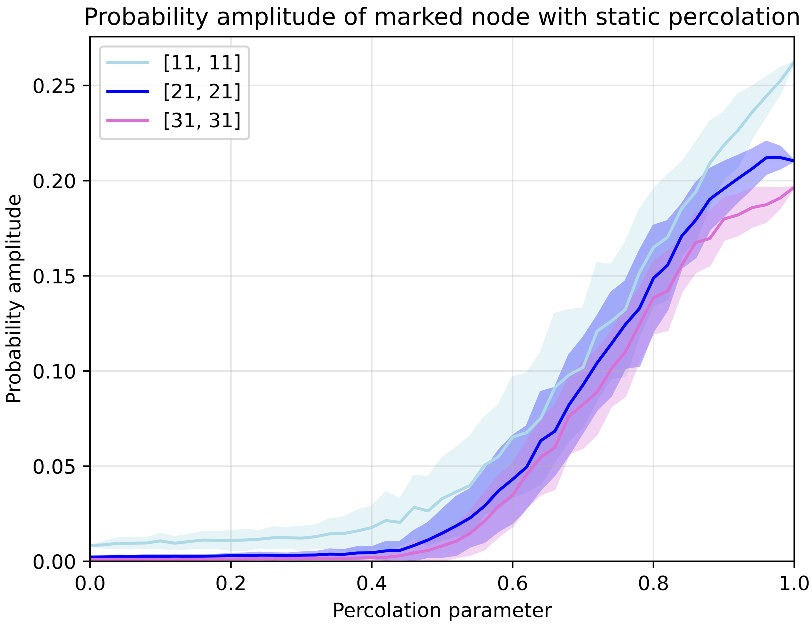

FIGURE 3.19: Mean and standard deviation (sample size n = 100) of the success probability for statically percolated QWSA simulations with runtime $\lfloor t_{opt} \rfloor$ for three grid sizes.

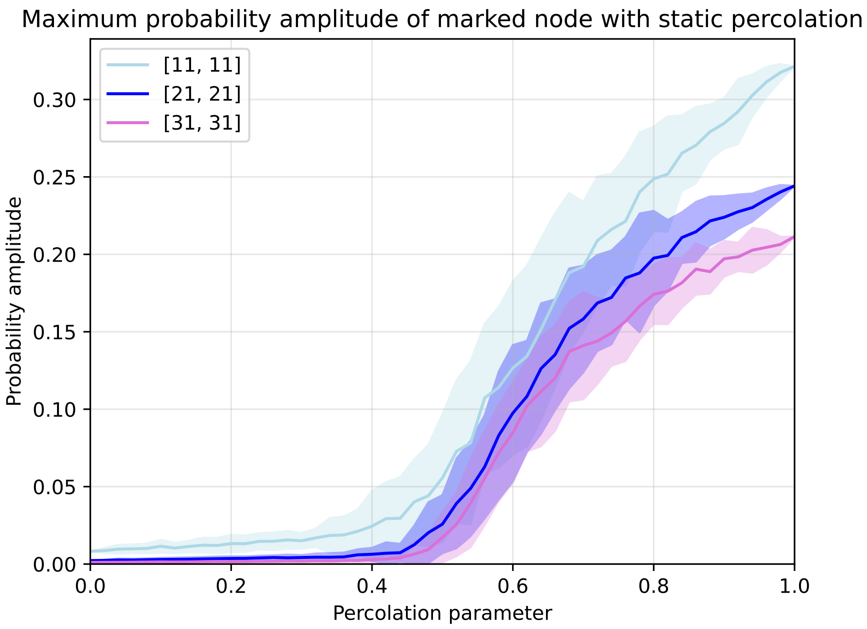

FIGURE 3.20: Mean and standard deviation (sample size n = 100) of the maximum marked node probability obtained from statically percolated QWSA simulations run for $\lfloor 2.9 \times t_{opt} \rfloor$ timesteps for three grid sizes.

The success probability in the static case remains much closer to the unpercolated success probability for a much greater range of percolation values than in the dynamic case. This can be explained by the idea that, in contrast to the dynamic case, static percolation can be interpreted as modeling a degraded environment where the decoherence time is significantly longer than the runtime. This means that while the success probability should decrease with a decreasing percolation parameter, indeed as is depicted in Figure 3.19 and Figure 3.20, it should decrease more gradually than in the dynamic case.

It is between percolation parameter values 0.4 and 0.6 that the success probability decreases quickly. This is right around the threshold of the percolation parameter value of these geometries, at this point the graph starts to become much more disconnected. This means that a lot of the probability cannot flow towards the marked node, drastically decreasing the success probability of the QWSA.

Also noticeable is that, between percolation parameters 0.5 and 0.8, the maximum marked node probability amplitude achieved during the QWSA significantly exceeds the probability amplitude at the asymptotic runtime $\lfloor t_{opt} \rfloor$. This suggests a possible shift in the average of the optimal runtime in the presence of static percolation. This is explored in the next subsection; section 3.5.

3.5 Optimal runtime for statically percolated QWSAs

The optimal runtime for the average walk, shown in Figure 3.21, supports the findings from subsection 3.4.2. As the percolation parameter decreases, the time step at which the maximum marked node probability is achieved increases progressively. This increase in optimal runtime continues until the longest optimal runtime is observed between percolation parameters 0.5 and 0.55.

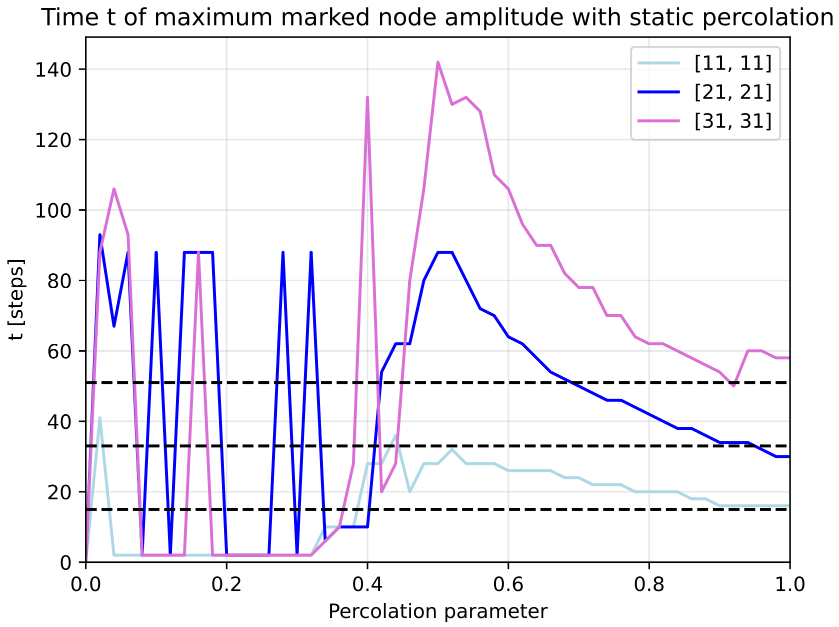

FIGURE 3.21: The optimal runtime of the average (sample size n = 100) statically percolated walk, the dashed lines represent $\lfloor t_{opt} \rfloor$ from Equation 2.31 with k_1 and k_2 from Figure 3.5.

section 3.3 and section 2.4 provide an explanation for this increase and the following decrease in optimal runtime. Until close to the percolation threshold, there is a high likelihood that much of the grid remains connected to the marked node, allowing amplitude to flow from many parts of the graph toward the marked node. But as the percolation parameter decreases, the number of paths to the marked node also decrease, and amplitude from distant nodes takes longer to arrive. This results in the marked node probability peaking at increasingly later time steps. But once the percolation parameter drops below around 0.55, the optimal runtime begins to decrease rapidly. This likely is a result of entering the critical and subcritical percolation regimes of these geometries. As the grids become more percolated, the area connected to the marked node tends to be smaller and smaller, hence the probabilities that can flow to the marked node do not need a lot of time to get there, thus reducing the optimal runtime.

Lastly, it is noted that the peaks at or below percolation parameter 0.4 are a result of the oscillatory behavior; in these cases it is a later peak that obtains the maximum value and not the first. So these big peaks do not actually represent the first optimal percolation time, the first optimal time is significantly smaller and better represented by the smallest runtimes in this percolation parameter regime below 0.4. This explanation was verified by examining the average walks. Above the percolation parameter of 0.4 this behavior does not interfere, so in that region the data points in Figure 3.21 do represent the first optimal times of the average QWSA.

Chapter 4

Conclusion

This thesis investigated the success probability and optimal runtime of quantum-walk-based search algorithms (QWSAs) on rectangular lattices and percolated square lattices.

For unpercolated grids, an asymptotic approximation of the runtime, based on the principal eigenvalue of the evolution operator, was determined. While this approximation performed well for square lattices, it consistently underestimated the optimal runtime for rectangular lattices. This discrepancy results from geometric asymmetry, which amplifies the role of higher frequency components not captured by the dominant-eigenvalue-based estimate. A similar effect was observed for the predicted success probability: the asymptotic formula provided reliable estimates for square grids but tended to overestimate the probability for rectangular grids during their first optimal peak. Due to the limited number of grid sizes studied, it is still uncertain whether this trend continues for significantly larger grids. However, in the limit of infinite grid size, the asymptotic approximation is expected to hold.

When percolation is introduced, the symmetry and connectivity of the grid are disrupted. Dynamic percolation in particular creates a lot of disorder and works as a model for strong decoherence. As a consequence, the success probability of QWSAs decreases quickly as the percolation parameter decreases. Moreover, as the size of the grid increases, the adverse effect of dynamic percolation on the effectiveness of the QWSA increases. At the same percolation value (less than 1), the success probability of a larger grid is reduced more, relative to the unpercolated case, than that of a smaller grid. This is likely due to the fact that the optimal runtime of a QWSA increases with grid size, while the timescale on which the grid fluctuates under dynamic percolation (every timestep) remains the same. In other words, larger grids require a longer coherence time for effective execution, but under dynamic percolation, the decoherence timescale of the grid remains fixed. This leads to a greater decrease in performance of QWSA in the presence of dynamic percolation in larger grids.

Static percolation also degrades the structure, but the percolated structure remains fixed throughout the walk. As long as the percolation parameter remains in the supercritical regime, static percolation on average still allows the QWSA to work reasonably effectively, although the average optimal runtime is longer and the average success probability is reduced. However, as the percolation parameter approaches the percolation threshold, the grid loses its global connectivity, leading to a rapid decline in performance even in the static case.

Further research could look at the effect of percolation on other types of structures; hexagonal lattices, hypercubes or different set of boundary conditions, for example. Also site percolation instead of bond percolation could be interesting to study in the context of QWSAs. Lastly, future research could look at the performance of other quantum-walk-based algorithms, such as element distinctness determination [39] or search algorithms based on Szegedy's quantum walk [12], in the presence of percolation.

Appendix A

Eigenvectors of an N by M grid

Using the method of Fourier transforms, the eigenvectors and values for an $N \times M$ grid are obtained, see Table A.1. Where $|k\rangle$ is the Fourier basis for the x-coordinates (side length N) and $|l\rangle$ is the Fourier basis for the y-coordinate (side length M). So: $|k,l\rangle = \sum_{x=0}^{N-1}\sum_{y=0}^{M-1}\omega^{xk}\alpha^{kl}|x\rangle|y\rangle$, with $\omega = e^{\frac{2\pi i}{N}}$ and $\alpha = e^{\frac{2\pi i}{M}}$. θ_{kl} is defined by:

$$\cos\theta_{kl} = \frac{1}{2}(\cos\frac{2\pi k}{N} + \cos\frac{2\pi l}{M}) \tag{A.1}$$

Table A.1: Eigenvectors of an $N \times M$ grid

Eigenvalues	Eigenvectors for $k = l = 0$	Eigenvectors for $k \neq 0$ or $l \neq 0$
$\lambda = 1$	$v_{+1} = \frac{1}{2}\begin{pmatrix}1\\1\\1\\1\end{pmatrix}\lvert 0,0\rangle$	$v_{+1} = \begin{pmatrix}\omega^k(\alpha^l-1)\\1-\alpha^l\\\alpha^l(1-\omega^k)\\\omega^k-1\end{pmatrix}\lvert k,l\rangle$
$\lambda = -1$	$v_{-1} = \frac{1}{2}\begin{pmatrix}1\\1\\-1\\-1\end{pmatrix}\lvert 0,0\rangle$	$v_{-1} = \begin{pmatrix}-\omega^k(1+\alpha^l)\\-(1+\alpha^l)\\\alpha^l(1+\omega^k)\\1+\omega^k\end{pmatrix}\lvert k,l\rangle$
$\lambda = e^{i\theta_{kl}}$	$v_{+1} = \frac{1}{\sqrt{2}}\begin{pmatrix}1\\-1\\0\\0\end{pmatrix}\lvert 0,0\rangle$	$v_{\theta} = \frac{i}{2\sqrt{2}\sin\theta_{kl}}\begin{pmatrix}e^{-i\theta_{kl}}-\omega^k\\e^{-i\theta_{kl}}-\omega^{-k}\\e^{-i\theta_{kl}}-\alpha^l\\e^{-i\theta_{kl}}-\alpha^{-l}\end{pmatrix}\lvert k,l\rangle$
$\lambda = e^{-i\theta_{kl}}$	$v_{+1} = \frac{1}{\sqrt{2}}\begin{pmatrix}0\\0\\1\\-1\end{pmatrix}\lvert 0,0\rangle$	$v_{-\theta} = \frac{-i}{2\sqrt{2}\sin\theta_{kl}}\begin{pmatrix}e^{i\theta_{kl}}-\omega^k\\e^{i\theta_{kl}}-\omega^{-k}\\e^{i\theta_{kl}}-\alpha^l\\e^{i\theta_{kl}}-\alpha^{-l}\end{pmatrix}\lvert k,l\rangle$

Appendix B

Requirement I: $\lambda << \theta_{min}$ for $N, M \to \infty$

The first requirement that is proven is that $\lambda << \theta_{min}$ for $N, M \to \infty$.The first goal is to make an asymptotic approximation for the smallest positive θ_{kl} in the limit of $N, M \to \infty$. Note that this argument goes to zero as N, M go to infinity, this allows a Maclaurin expansion of both sides of Equation A.1:

$$\cos\theta_{kl} = 1 - \frac{1}{2}\theta_{kl}^2 + \mathcal{O}(\theta_{kl}^4) \tag{B.1}$$

and

$$\frac{1}{2}(\cos\frac{2\pi k}{N} + \cos\frac{2\pi l}{M}) = 1 - \frac{1}{2}((\frac{2\pi k}{N})^2 + (\frac{2\pi l}{M})^2) + \mathcal{O}(\frac{1}{N^4} + \frac{1}{M^4}) \tag{B.2}$$

Combining the expressions through Equation A.1 and simplifying the equation, produces for the positive solution:

$$\theta_{kl} + \mathcal{O}(\theta_{kl}^2) = \sqrt{2(\frac{\pi k}{N})^2 + 2(\frac{\pi l}{M})^2 + \mathcal{O}(\frac{1}{N^4} + \frac{1}{M^4})} \tag{B.3}$$

So for the limit of $N, M \to \infty$, the following expression for θ_{kl} is obtained:

$$\theta_{kl} = \pi\sqrt{2(\frac{k}{N})^2 + 2(\frac{l}{M})^2} \tag{B.4}$$

The smallest positive argument must then have $(k, l) = (1, 0)$ under the assumption that N is the smaller side lengt.

Now, according to the method from [31], λ in the scenario of a real U is given by $\lambda = \sqrt{\frac{A}{C}}$. Using A and C from section 2.3, this is asymptotically equal to:

$$\lambda = \frac{\sqrt{2}}{\sqrt{NM \times k_1(\ln NM + k_2)}} \tag{B.5}$$

The ratio $\frac{\lambda}{\theta_{min}}$ is then asymptotically equal to:

$$\frac{\lambda}{\theta_{min}} = \frac{1}{\pi\sqrt{(\frac{N}{M} + \frac{M}{N}) \times k_1(\ln NM + k_2)}} \tag{B.6}$$

The sum of the ratios $\frac{N}{M}$ and $\frac{M}{N}$ is constant when grid dimensions are kept the same. So then the limit of $N, M \to \infty$ of the ratio goes to zero which proves the first requirement.

Appendix C

Requirement II: $\epsilon \to 0$ as $N, M \to \infty$

The second requirement that has to be proven is that $\epsilon \to 0$ as $N, M \to \infty$. To proof this, the following identities from Portugal [31] are used:

$$\langle\psi(0)|\lambda\rangle = \langle\lambda'|\psi(0)\rangle = \langle\psi(0)|x_o\rangle\,(\frac{1}{2\sqrt{C}} + \frac{i}{\sqrt{A}}) \tag{C.1}$$

$|\,\langle\psi(0)|x_o\rangle\,|^2 = \frac{1}{NM}$, so the square of the absolute value is then equal to:

$$|\,\langle\psi(0)|\lambda\rangle\,|^2 = |\,\langle\lambda'|\psi(0)\rangle\,|^2 = \frac{1}{NM}(\frac{1}{4C} + \frac{1}{A}) \tag{C.2}$$

Using A and C from section 2.3, in the asymtotic limit of $N, M \to \infty$ this identity is then equal to:

$$\lim_{N,M\to\infty} |\,\langle\psi(0)|\lambda\rangle\,|^2 = \lim_{N,M\to\infty} |\,\langle\lambda'|\psi(0)\rangle\,|^2 = \lim_{N,M\to\infty} \frac{1}{NM}(\frac{1}{4k_1(\ln NM + k_2)} + \frac{NM}{2}) = \frac{1}{2} \tag{C.3}$$

So $\lim_{N,M\to\infty}(|\,\langle\psi(0)|\lambda\rangle\,|^2 + |\,\langle\lambda'|\psi(0)\rangle\,|^2) = 1$, therefore the component in the initial conditions that is ortogonal to $|\lambda\rangle$ and $|\lambda'\rangle$ becomes vanishingly small. This means that $\lim_{N,M\to\infty} |\,\langle\psi(0)|U_{res}|\psi(0)\rangle\,|^2 = 0$, where U_{res} has eigenvalue 0 for $|\lambda\rangle$ and $|\lambda'\rangle$ and acts unitary on the subspace orthogonal to them. And since $|x_0\rangle$ is normalized this means that $\lim_{N,M\to\infty} \epsilon_{N,M} = \lim_{N,M\to\infty} |\,\langle x_0|U_{res}^t|\psi(0)\rangle = 0$. This proves the seconde requirement.

Appendix D

Inequality with intermediate steps

Including the intermediate steps in the inequality from Equation 2.28 is results in:

$$\frac{N^2}{\pi^2} = \frac{N^2}{\pi^2}\sum_{k=1}^{1}\frac{1}{k^2} \leq \frac{N^2}{\pi^2}\sum_{k=1}^{\lfloor\frac{N}{2}\rfloor}\frac{1}{k^2} \leq \sum_{k=1}^{\lfloor\frac{N}{2}\rfloor}\frac{1}{\sin^2(\frac{\pi k}{N})} \leq \frac{N^2}{4}\sum_{k=1}^{\lfloor\frac{N}{2}\rfloor}\frac{1}{k^2} \leq \frac{N^2}{4}\sum_{k=1}^{\infty}\frac{1}{k^2} = \frac{\pi^2 N^2}{24} \tag{D.1}$$

Appendix E

Solution of the integral

The integral solution is presented here; for arbitrary a the following integral needs to be evaluated:

$$\int_a^{\frac{M}{2}} \int_a^{\frac{N}{2}} \frac{1}{(\frac{x}{N})^2 + (\frac{y}{M})^2} dxdy \tag{E.1}$$

Using the substitutions $x' = \frac{x}{N}$ and $y' = \frac{y}{M}$ this becomes;

$$NM \int_{\frac{a}{M}}^{\frac{1}{2}} \int_{\frac{a}{N}}^{\frac{1}{2}} \frac{1}{(x')^2 + (y')^2} dx'dy' = NM \int_{\frac{a}{M}}^{\frac{1}{2}} \int_{\frac{a}{N}}^{\frac{1}{2}} \frac{1}{y'^2} \frac{1}{(\frac{x'}{y'})^2 + (1)^2} dx'dy' = \tag{E.2}$$

$$NM \int_{\frac{a}{M}}^{\frac{1}{2}} \frac{1}{y'^2} [y' \arctan(\frac{x'}{y'})]_{\frac{a}{N}}^{\frac{1}{2}} dy' = NM \int_{\frac{a}{M}}^{\frac{1}{2}} \frac{1}{y'} (\arctan(\frac{1}{2y'}) - \arctan(\frac{a}{Ny'}))dy' \tag{E.3}$$

Now the substitution $y = \frac{1}{y'}$ gives:

$$-NM \int_{\frac{M}{a}}^{2} \frac{1}{y} (\arctan(\frac{y}{2}) - \arctan(\frac{ay}{N}))dy = -NM(\int_{\frac{M}{2a}}^{1} \frac{1}{y} \arctan(y)dy - \int_{\frac{M}{N}}^{\frac{2a}{N}} \frac{1}{y} \arctan(y)dy) \tag{E.4}$$

This integral is in the form of a known function called the inverse tangent integral Ti_2. This produces the result:

$$\int_a^{\frac{M}{2}} \int_a^{\frac{N}{2}} \frac{1}{(\frac{x}{N})^2 + (\frac{y}{M})^2} dxdy = NM(-\mathrm{Ti}_2(1) - \mathrm{Ti}_2(\frac{M}{N}) + \mathrm{Ti}_2(\frac{M}{2a}) + \mathrm{Ti}_2(\frac{2a}{N})) \tag{E.5}$$

Note that the same results, but with M and N interchanged can be produced if the order of integration is reversed, therefore:

$$\int_a^{\frac{M}{2}}\int_a^{\frac{N}{2}} \frac{1}{(\frac{x}{N})^2+(\frac{y}{M})^2}dxdy = \tag{E.6}$$
$$\frac{1}{2}NM(\mathrm{Ti}_2(\frac{M}{2a})+\mathrm{Ti}_2(\frac{2a}{N})+\mathrm{Ti}_2(\frac{N}{2a})+\mathrm{Ti}_2(\frac{2a}{M})-2\mathrm{Ti}_2(1)-\mathrm{Ti}_2(\frac{M}{N})-\mathrm{Ti}_2(\frac{N}{M})) \tag{E.7}$$

The following inverse tangent integral identity:

$$\mathrm{Ti}_2(x)-\mathrm{Ti}_2(\frac{1}{x})=\frac{\pi}{2}\ln(x) \tag{E.8}$$

in combination with the fact that $\mathrm{Ti}_2(x)=\mathcal{O}(x)$ for $x \to 0$, so as to obtain the complete result:

$$\int_a^{\frac{M}{2}}\int_a^{\frac{N}{2}} \frac{1}{(\frac{x}{N})^2+(\frac{y}{M})^2}dxdy = \tag{E.9}$$
$$\frac{1}{2}NM(\frac{\pi}{2}(\ln(\frac{M}{2a}+\ln(\frac{N}{2a}))-2\mathrm{Ti}_2(1)-\mathrm{Ti}_2(\frac{M}{N})-\mathrm{Ti}_2(\frac{N}{M})+\mathcal{O}(\frac{1}{M}+\frac{1}{N}))= \tag{E.10}$$
$$\frac{\pi}{4}NM(\ln(NM)-(2\ln(2a)+\frac{2}{\pi}(\mathrm{Ti}_2(1)+\mathrm{Ti}_2(\frac{M}{N})+\mathrm{Ti}_2(\frac{N}{M})))+\mathcal{O}(\frac{1}{M}+\frac{1}{N})) \tag{E.11}$$

Bibliography

[1] A. Quarteroni, R. Sacco, and F. Saleri, *Numerical Mathematics.* Springer, 2nd ed., 2007.

[2] B. Siciliano and O. Khatib, eds., *Springer Handbook of Robotics.* Springer, 2nd ed., 2016.

[3] A. V. Aho, J. E. Hopcroft, and J. D. Ullman, *The Design and Analysis of Computer Algorithms.* Addison-Wesley, 1974.

[4] S. Arora and B. Barak, *Computational Complexity: A Modern Approach.* Cambridge University Press, 2009.

[5] T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein, *Introduction to Algorithms.* MIT Press, 3rd ed., 2009.

[6] R. Motwani and P. Raghavan, *Randomized Algorithms.* Cambridge University Press, 1995.

[7] S. J. Devitt, W. J. Munro, and K. Nemoto, “Quantum error correction for beginners,” *Reports on Progress in Physics*, vol. 76, no. 7, p. 076001, 2013.

[8] M. A. Nielsen and I. L. Chuang, *Quantum Computation and Quantum Information.* Cambridge University Press, 10th anniversary ed., 2010.

[9] A. Montanaro, “Quantum algorithms: an overview,” *npj Quantum Information*, vol. 2, p. 15023, 2016.

[10] D. J. Griffiths, *Introduction to Quantum Mechanics.* Cambridge University Press, 3rd ed., 2018.

[11] Y. Liu, S. Arunachalam, and K. Temme, “A rigorous and robust quantum speed-up in supervised machine learning,” *Nature Physics*, vol. 17, no. 9, pp. 1013–1017, 2021.

[12] M. Szegedy, “Quantum speed-up of markov chain based algorithms,” in *45th Annual IEEE symposium on foundations of computer science*, pp. 32–41, IEEE, 2004.

[13] P. W. Shor, “Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer,” *SIAM Journal on Computing*, vol. 26, no. 5, pp. 1484–1509, 1997.

[14] L. K. Grover, “A fast quantum mechanical algorithm for database search,” in *Proceedings of the twenty-eighth annual ACM symposium on Theory of computing*, pp. 212–219, ACM, 1996.

[15] F. Spitzer, *Principles of Random Walk.* Springer, 1964.

[16] L. Lovász, “Random walks on graphs: A survey,” in *Combinatorics, Paul Erdős is Eighty*, vol. 2, pp. 1–46, János Bolyai Mathematical Society, 1993.

[17] S. Brooks, A. Gelman, G. Jones, and X.-L. Meng, *Handbook of Markov Chain Monte Carlo.* CRC Press, 2011.

[18] S. Kirkpatrick, C. D. Gelatt, and M. P. Vecchi, “Optimization by simulated annealing,” *Science*, vol. 220, no. 4598, pp. 671–680, 1983.

[19] M. E. Newman, *Networks: An Introduction.* Oxford University Press, 2010.

[20] J. Nsofini, “Discrete quantum walk on a line with two entangled particles,” *arXiv preprint arXiv:1205.4357*, 2012.

[21] J. Kempe, “Quantum random walks: an introductory overview,” *Contemporary Physics*, vol. 44, no. 4, pp. 307–327, 2003.

[22] A. Ambainis, E. Bach, A. Nayak, A. Vishwanath, and J. Watrous, “One-dimensional quantum walks,” in *Proceedings of the 33rd annual ACM symposium on Theory of computing*, pp. 37–49, ACM, 2001.

[23] S. E. Venegas-Andraca, “Quantum walks: a comprehensive review,” *Quantum Information Processing*, vol. 11, no. 5, pp. 1015–1106, 2012.

[24] N. Shenvi, J. Kempe, and K. B. Whaley, “Quantum random-walk search algorithm,” *Physical Review A*, vol. 67, no. 5, p. 052307, 2003.

[25] M. R. Garey and D. S. Johnson, *Computers and Intractability: A Guide to the Theory of NP-Completeness.* W. H. Freeman, 1979.

[26] P. W. Anderson, “Absence of diffusion in certain random lattices,” *Physical Review*, vol. 109, no. 5, p. 1492, 1958.

[27] A. Romanelli, R. Siri, G. Abal, A. Auyuanet, and R. Donangelo, “Decoherence in the quantum walk on the line,” *Physica A: Statistical Mechanics and its Applications*, vol. 347, pp. 137–152, 2005.

[28] V. M. Kendon, “Decoherence in quantum walks—a review,” *Mathematical Structures in Computer Science*, vol. 17, no. 6, pp. 1169–1220, 2007.

[29] A. Tulsi, “Faster quantum-walk algorithm for the two-dimensional spatial search,” *Physical Review A*, vol. 78, no. 1, p. 012310, 2008.

[30] A. Ambainis, K. Prūsis, J. Vihrovs, and T. G. Wong, “Full characterization of oscillatory localization of quantum walks,” *arXiv preprint arXiv:1606.02139*, 2016.

[31] R. Portugal, *Quantum Walks and Search Algorithms.* Quantum Science and Technology, Springer, 2013.

[32] D. Stauffer and A. Aharony, *Introduction to Percolation Theory.* London: Taylor & Francis, 2nd ed., 1994.

[33] G. Grimmett, *Percolation.* Berlin: Springer, 2nd ed., 1999.

[34] B. Bollobás and O. Riordan, “A short proof of the harris–kesten theorem,” *Bulletin of the London Mathematical Society*, vol. 38, no. 3, pp. 470–484, 2006.

[35] X. Feng, Y. Deng, and H. W. Blöte, “Percolation transitions in two dimensions,” *Physical Review E—Statistical, Nonlinear, and Soft Matter Physics*, vol. 78, no. 3, p. 031136, 2008.

[36] M. E. J. Newman and R. M. Ziff, “Efficient monte carlo algorithm and high-precision results for percolation,” *Physical Review Letters*, vol. 85, no. 19, pp. 4104–4107, 2000.

[37] R. A. M. Santos and F. de L. Marquezino, “Decoherence on staggered quantum walks,” *Phys. Rev. A*, vol. 105, p. 032452, Mar 2022.

[38] P. L. Knight, E. Roldán, and J. E. Sipe, “Quantum walk on the line as an interference phenomenon,” *Physical Review A*, vol. 68, no. 2, p. 020301, 2003.

[39] A. Ambainis, “Quantum walk algorithm for element distinctness,” *SIAM Journal on Computing*, vol. 37, no. 1, pp. 210–239, 2007.