University of Calgary

PRISM Repository **https://prism.ucalgary.ca**

The Vault Open Theses and Dissertations

2023-07

Quantum Algorithms for Memoryless Search and Perfect Matching

Leahy, Janet

Leahy, J. (2023). Quantum algorithms for memoryless search and perfect matching (Master's thesis, University of Calgary, Calgary, Canada). Retrieved from https://prism.ucalgary.ca.

https://hdl.handle.net/1880/116736

Downloaded from PRISM Repository, University of Calgary

UNIVERSITY OF CALGARY

Quantum Algorithms for Memoryless Search and Perfect Matching

by

Janet Leahy

A THESIS

SUBMITTED TO THE FACULTY OF GRADUATE STUDIES

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE

DEGREE OF MASTER OF SCIENCE

GRADUATE PROGRAM IN COMPUTER SCIENCE

CALGARY, ALBERTA

JULY, 2023

© Janet Leahy 2023

Abstract

In this thesis, we present two new quantum algorithms for graph problems.

The first algorithm we give is a memoryless walk that can find a unique marked vertex on a two-dimensional grid. Our walk is based on a construction proposed by Falk, which tessellates the grid with squares of size 2×2. Our walk uses minimal memory, $O(\sqrt{N \log N})$ applications of the walk operator, and outputs the marked vertex with vanishing error probability. To accomplish this, we apply a selfloop to the marked vertex—a technique we adapt from interpolated walks. We prove that with our explicit choice of selfloop weight, this forces the action of the walk asymptotically into a single rotational space. We characterize this space and as a result, show that our memoryless walk produces the marked vertex with a success probability asymptotically approaching one.

Our second algorithm decides whether a graph contains a perfect matching. This is the first quantum algorithm based on the algebraic characterization by Tutte, which reduces the problem of detecting perfect matchings to deciding whether a matrix has nonzero determinant. The key part of our algorithm is a new span program that can decide whether a matrix is singular. Our span program has a simple structure and its witness size matches that of a related span program by Belovs for matrix rank-finding, up to a constant factor. Using a transformation given by Reichardt, our span program can be compiled into a quantum algorithm, which we use as a subroutine in our algorithm to detect perfect matchings. We also show that there are families of graphs for which our perfect matching detection algorithm may have exponential query complexity. These graphs could be a useful tool in determining the tight quantum query complexity of the perfect matching detection problem, which remains an open problem.

Preface

The results in this thesis are original work between the author, Janet Leahy, and supervisor Peter Høyer. Fellow graduate student Zhan Yu participated in the early stages of the research on memoryless walks by running numerical simulations to test the main statements.

Chapter 3 on memoryless walks is part of an article that has been pubilshed in *Physical Review A*. The article is available as “Spatial Search via an Interpolated Memoryless Walk" at `doi:10.1103/PhysRevA.78.012310`. The contents of the article are used in this thesis either verbatim or with minor modifications.

Acknowledgments

Research, as with so many things, is collaborative work, and there are many people without whom this thesis would not have been written.

First off, I am deeply thankful to my supervisor, Peter Høyer, for his guidance, ideas, patience, and volumes of feedback. His passion for growth as both a researcher and a person has inspired me since I was a student in his algorithms class years ago, and has given me a goal to always strive towards.

Sincere thanks are also due to Dr. David Feder and Dr. Renate Scheidler for reading my thesis and for serving along with Peter as my examiners. I am grateful for the time they spent providing valuable and thoughtful feedback, which has helped me improve as a researcher.

My time as a graduate student has been greatly enriched by the camaraderie of the quantum walks research group: Dante Bencivenga, Shang Li, Xining Chen, Zhan Yu, Yehonatan Shabash, and Elliot Evans. The conversations we had, both technical and otherwise, have both inspired me and helped to pull me through the tougher times. I'm glad to have them as friends.

I would also like to acknowledge to the organizations that provided funding my graduate research: the Alberta Graduate Excellence Scholarship program (AGES), the Alberta Innovates Graduate Student Scholarships program, and the National Sciences and Engineering Research Council of Canada (NSERC).

Finally, I am eternally grateful to my friends and family for all of the much-needed support and cups of tea. I am so blessed to know you all, and hope to see more of you soon.

Table of Contents

List of Figures

List of Symbols

Chapter 2 – Background

n	number of vertices in a graph G
M	the set of marked vertices in a graph
X_0	initial state of a random walk
X_k	state after k steps of a random walk
P	row-stochastic transition matrix for a random walk
$\deg(v)$	degree of the vertex v
π	stationary distribution of a random walk
T	number of steps in a walk-based search algorithm
S	the cost of sampling from π
C	the cost of checking whether a vertex is marked
U	the cost of taking one step of a random walk according to P
$\mathsf{HT}(\mathsf{P}, M)$	hitting time of a random walk with transition matrix P and marked vertices M
N	number of vertices in the square two-dimensional toric grid
$e^{\mathsf{H}t}$	evolution operator for a continuous quantum walk
d	degree of vertices in a regular graph (Section 2.2); dimension of vectors in a span program (Section 2.3)
$\{0,1\}^n$	binary strings of length n
$\mathscr{P}$	span program
$f_{\mathscr{P}}$	function computed by the span program $\mathscr{P}$
$\lvert\tau\rangle$	target vector in a span program
V_{free}	free vectors in a span program
$V_{i,b}$	a single input vector in a span program
V	set of all input vectors in a span program
$V(x)$	available vectors in a span program
$\lvert w^+\rangle, \lvert w^-\rangle$	positive, negative witness
$\text{wsize}(\mathscr{P})$	witness size of the span program $\mathscr{P}$

Chapter 3 – Memoryless quantum search

dim	dimension of a vector space
span	linear span of a set of vectors
n_r, n_c	number of rows, columns in the grid
N	number of vertices in the grid
$\lvert\circlearrowleft\rangle$	selfloop state
A, B	reflection about even, odd tessellation of the grid
$\lvert g\rangle$	marked vertex on the grid
$\lvert\tilde{g}\rangle$	interpolated marked state
s	interpolation parameter
$\tilde{\mathsf{G}}$	reflection of interpolated marked state
U	memoryless walk operator
U_z	memoryless walk operator, transformed to z-basis for analysis
W, F	input independent, dependent components of U_z
$\lvert\pi\rangle$	starting state of the walk, which is a uniform superposition over vertices
$\lvert\pi_z\rangle$	the state $\lvert\pi\rangle$ after transformation to the z-basis
$\lvert+\rangle, \lvert-\rangle$	orthonormal states spanning the two-dimensional space rotated by F
$\mathsf{F}_1, \mathsf{F}_2$	one-dimensional reflections generating the two-dimensional rotation $\mathsf{F} = \mathsf{F}_1\mathsf{F}_2$
$\lvert f_1\rangle, \lvert f_2\rangle$	vectors reflected by F_1, F_2
W_{kl}	invariant subspace of W indexed by k and l
θ_{kl}	eigenphase associated with subspace W_{kl}
Π_{kl}	projection onto subspace W_{kl}
φ_1	smallest positive eigenphase of $\mathsf{W}\mathsf{F}_1$
β	smallest positive eigenphase of $\mathsf{W}\mathsf{F}$

Chapter 4 – Perfect matchings and 0-determinant verification

Tr	trace of a matrix
Pr	probability of an event
$\mathbb{E}$	expectation of a random variable
n	number of vertices in a graph (perfect matching), or dimension of a square matrix (0-determinant verification)
ω	exponent of matrix multiplication
$c(\mathsf{A})$	square root of the average squared singular value of A^{-1}
T	upper bound on $c(\mathsf{A})$
L	cost of loading a matrix into a high-level span program
$\mathsf{T}(G)$	Tutte matrix of the graph G
S	set used for random instantiation of the Tutte matrix
$\mathscr{P}$	span program
$f_{\mathscr{P}}$	function computed by the span program $\mathscr{P}$
$\|\tau\rangle$	target vector in a span program
V_{free}	free vectors in a span program
$V_{i,b}$	a single input vector in a span program
$V(x)$	available vectors in a span program
$\|w^+\rangle, \|w^-\rangle$	positive, negative witness
wsize($\mathscr{P}$)	witness size of the span program $\mathscr{P}$
A	$n \times n$ matrix, input to 0-determinant verification
$\|r\rangle$	column vector of dimension $n \times 1$ whose entries are samples from the Rademacher distribution
A'	$(n+1)\times n$ matrix whose first row is $\langle r\|$ and whose remaining $n\times n$ entries are those of A
$\|\tau'\rangle$	$(n+1)\times 1$ vector whose first entry is 1 and whose remaining n entries are 0
A'_{ext}	matrix whose columns are the available vectors in our span program for 0-determinant verification
$\|\tau'_{\text{ext}}\rangle$	target vector in our span program for 0-determinant verification

Chapter 1

Overview

One of the driving reasons for interest in quantum computing is the discovery of quantum algorithms that can solve certain problems with a marked asymptotic speedup over what is possible with classical computation. A famous example of this is Shor's factoring algorithm [Sho97]. It requires exponentially fewer operations than the best-known classical algorithms for factoring, and as consequence, has had profound applications on modern cryptography.

An important and ongoing question is finding problems for which quantum algorithms can achieve asymptotic improvements in the required resources relative to what can be done classically. Not only is this interesting from a theoretical perspective, it is practically motivated. Early quantum computers are still subject to significant constraints on both the amount of memory and the number of operations that can be applied before the computation is lost to decoherence. Therefore, quantum algorithms that achieve an asymptotic improvement in the use of these resources are more feasible for implementation, and finding such algorithms is an active area of study.

Several models have been developed to characterize the complexity of quantum algorithms. We use the quantum query model, in which the problem input is accessed through quantum queries to a black box. The query complexity is defined to be the number of queries needed to solve the problem with bounded error. The advantage of this model is that counting queries is much simpler than counting quantum operations or gates. Because each query takes at least one time step, the query complexity gives a lower bound on the step complexity of an algorithm. For most known quantum algorithms, the step complexity is within a polylogarithmic factor of the query complexity, so the quantum query complexity is often a good indicator of the step complexity of an algorithm. Many nice results have been proven in the query model, including tight bounds on the query complexity of a number of problems and the elegant relationship between span programs, witness size, and the adversary lower bound [HLŠ07, Rei09].

In this thesis, we present two new quantum algorithms. These algorithms solve spatial search and perfect matching detection, respectively. Both of these are foundational problems in theoretical computer science. Their problem statements are simple, and yet their study reveals important differences between classical and quantum models of computation.

Our first algorithm is presented in Chapter 3, where we define a new type of quantum walk and show that it can optimally solve spatial search on the grid. The goal in spatial search is to find a marked vertex on a graph, where operations are subject to locality constraints. These constraints can be imposed by the physical layout of a system [AA05], or they can arise from the computational cost of moving from one vertex to another. An example of the latter can be found in [Amb07], where the element distinctness problem is reduced to spatial search in a structured graph. A query-optimal algorithm for element distinctness is then found using a query-optimal algorithm for spatial search.

A natural classical algorithm for spatial search uses a random walk to explore the graph. A walker is placed at a vertex on the graph following some given initial distribution. At each time step, the walker then moves to a vertex adjacent to its current location, chosen according to a fixed probability distribution. If at any point in the process, the walker's current vertex is marked, the search is complete, so the walker will stop and return the vertex it found.

There have been several quantum analogues developed for classical random walks, all of which are broadly classed as quantum walks. A primary motivation for developing quantum walks is the ubiquity of random walks as a component in classical algorithms. The hope is that by developing efficient quantum walks, they could be used to obtain a quantum speedup for a number of problems.

One of the standard types of quantum walk is the Szegedy quantum walk [Sze04], which is based on the discrete-time random walk. Another standard type is the continuous walk [FG98, CG04], which is based on a continuous-time evolution of the starting distribution according to the walk P. Both of these types of quantum walk exhibit markedly different behaviour from their classical counterparts. For example, on a one-dimensional line of vertices, both types of walk propagate quadratically faster than their classical counterparts [ABN$^+$01]. Continuous walks have also been proven to give an exponential speedup for finding a marked vertex on certain black-box graphs, relative to any possible classical algorithm [CCD$^+$03].

It was recently shown that Szegedy quantum walks can be used to solve the spatial search problem quadratically faster than classical random walks for any graph and any configuration of marked vertices [AGJK20]. This result answered a long-standing open question in the field of quantum walks, giving a method to obtain a step-optimal spatial search algorithm for any graph.

In this thesis, we give a quantum walk that improves on the Szegedy model in *space* complexity, while maintaining a simple structure and optimal step complexity. To do this, we use a memoryless walk.

Memoryless walks are notably different from the Szegedy and continuous models. They possess the features common to all quantum walks in that they can be divided into setup, step, and checking operations. Furthermore, like Szegedy walks, their discrete-time evolution can be decomposed into the product of two reflections. However, unlike Szegedy walks, which operate on the directed edge space of the graph, memoryless walks operate directly on the vertex space. This means that for a graph with N vertices and E edges, the evolving quantum state for a Szegedy walk lies in a Hilbert space of dimension at least E, while for a memoryless walk the dimension is simply N. The term “memoryless" captures this property, as this type of walk is designed to minimize the use of quantum memory.

Using a memoryless walk, we give a spatial search algorithm with a quadratic improvement in memory requirements relative to Szegedy’s edge-space walk. We prove that our walk can find a unique marked vertex on the two-dimensional grid with a quadratic speedup over classical walks. Thus, our walk is both memory-optimal and matches the best edge-space quantum walks in step complexity.

In addition, we show that our walk improves on previous memoryless walks for spatial search on the grid in its probability of finding a marked vertex. Inspired by the selfloops applied by [KMOR16] to Szegedy quantum walks, we show how to introduce selfloops into the memoryless setting. We give a precise description of how these selfloops affect the dynamics of our memoryless walk. As a result, we prove that adding a selfloop boosts the probability of finding the marked vertex from $O(\frac{1}{\log n})$ to $1-O(\frac{1}{\log n})$.

Our second algorithm is presented in Chapter 4, where we consider the problem of deciding whether a graph contains a perfect matching. Finding matchings is a well-studied problem in computer science, and yet there remains a gap between the best-known upper and lower bounds for the quantum query complexity of the problem. This gap applies to both the problem of finding a perfect matching and deciding whether one exists.

To investigate this gap, we give a new quantum algorithm that decides whether a graph contains a perfect matching. Our algorithm uses a different approach from previous quantum algorithms, drawing inspiration from an algebraic characterization of the problem by Tutte [Tut47].

Tutte's characterization reduces the problem of deciding whether a graph contains a perfect matching to the problem of deciding whether the determinant of the associated Tutte matrix is nonzero. Classically, this reduction has been used to obtain algorithms that can find a matching on a graph in $O(n^{\omega})$ steps with vanishing error probability [RV89, MS04], where $\omega < 2.37$ is the exponent of matrix multiplication. These algebraic algorithms are asymptotically the fastest-known classical algorithms for the problem.

In this thesis, we investigate whether an algebraic approach could be applied in the quantum setting. We give a natural algebraic quantum algorithm that decides whether a graph contains a perfect matching, and prove bounds on its query complexity.

The key component of our algorithm is a new quantum subroutine that decides whether the determinant of a matrix is nonzero. We refer to this problem as 0-determinant verification. As might be expected, the query complexity of our subroutine depends on the spectrum of the input matrix—if the input matrix is close to singular, the query complexity of the subroutine becomes very large.

As we show, there are classes of pathological graphs for which the matrix passed to this subroutine may have exponentially small eigenvalues, even when accounting for adjustable parameters in our algorithm. This would imply that our algorithm has exponential query complexity. As a result, our algorithm does not give an improvement in query complexity over the current best quantum algorithms for *finding* perfect matchings, which implicitly solve the decision problem and only require $O(n^{7/4})$ queries [LL16]. Furthermore, these results may suggest a fatal flaw with the algebraic approach to the matchings problem in the quantum setting.

On the other hand, our analysis reveals a class of hard-case graphs that have not yet been considered in the context of matchings. Studying these graphs may lead to new insights on the problem. Our algorithm may also still improve on the best existing quantum algorithms when there are restrictions placed on the allowed input graphs.

Finally, as part of our analysis we show that there remains a gap between the best upper and lower bounds for the query complexity of 0-determinant verification. This gap extends to the problem of determining the rank of a matrix. We do this in Appendix 5 by reconsidering a statement by Dörn and Theirauf [DT09]. This leaves the open question of proving a tight bound on the quantum query complexity of these two problems.

Chapter 2

Background

We begin with a chapter presenting the requisite terminology and some context useful for understanding our work. The remaining chapters and appendices focus on our original contributions in the thesis.

For this thesis, we assume familiarity with the basic principles of quantum computation and algorithm design. In particular, this includes an understanding of quantum states, quantum evolution and measurement, the quantum circuit model, complexity measures of quantum algorithms, and quantum search via amplitude amplification. From an algorithmic perspective, we assume a standard knowledge of algorithmic design and analysis, including bounded-error randomized algorithms and asymptotic notation. Readers may refer to the canonical textbook [NC00] for an introduction to each of the topics listed above.

Random walks, introduced in Section 2.1, are a natural classical solution to search problems with spatial constraints, and have many applications in algorithm design. There have been several quantum analogues developed for random walks, collectively known as quantum walks. We discuss some of their common features and applications in Section 2.2. Our first key result in this thesis is a new addition to this family of algorithmic tools. Our quantum walk is defined on the two-dimensional toric grid, defined formally in Section 3.2. We reference this graph here in our discussion of both random and quantum walks.

Our second main result relies on a subroutine defined in the span program framework. Span programs are a computational model which can be used to design quantum algorithms. Although span programs themselves do not capture any quantum dynamics, they can be converted into quantum algorithms whose query complexity depends on the span program's witness size. We introduce span programs and their witness size in Section 2.3.

2.1 Random walks

An important problem in computer science is that of spatial search. In spatial search, the search domain is modelled as a graph, where vertices represent elements in the search space. The problem differs from unstructured search in the existence of locality constraints on operations. In an unstructured search algorithm, the complexity is determined solely by the number of elements that are inspected. In spatial search, however, there is an additional cost associated with traversing the graph. Moving between adjacent vertices incurs a single unit of cost, so the cost of travelling from one vertex to another depends on the number of edges between them.

For this thesis, we limit our scope to unweighted, undirected graphs, i.e. the case where all edges are bidirectional and have weight one. We also assume that the underlying graph G is **strongly connected**, meaning that any pair of vertices in the graph is connected by some path.

Given a graph, the goal of a spatial search algorithm is to find a marked vertex or output “None" if none exist. A natural solution to this problem is to use a random walk.

Definition 1 *Given a graph G with n vertices, a **discrete-time random walk** on G is a sequence of vertices chosen according to a stochastic process. The walk begins at some vertex, captured by random variable X_0. At each time step, the walk transitions randomly to a neighbour of its current vertex according to a fixed distribution. This distribution is captured by a $n \times n$ row-stochastic matrix P, where the value P_{ij} is interpreted as the probability of transitioning from vertex i to vertex j. Note that P_{ij} can only be nonzero if vertex i is adjacent to vertex j.*

We denote the vertex reached after k steps of the walk with the random variable X_k, and refer to this as the ***state*** *of the walk after k steps. We refer to X_0 as the* ***initial state*** *of the walk and* P *as the* ***transition matrix****.*

We remark that with this definition, any non-initial state X_k of a random walk depends solely on its previous state, X_{k-1}. Thus, our random walks are equivalent to discrete-time Markov chains over a finite state space.[1]

For an unweighted graph, the default way to define the entries of P is to set $\mathsf{P}_{ij} = \frac{1}{\deg(i)}$, so the walk is equally likely to progress to each of the neighbours of its current vertex. For this thesis, we will always set the initial state to be chosen according to the **stationary distribution** of the walk, denoted π, where the probability of starting at vertex i is given by

$$\pi(i) = \frac{\deg(i)}{\sum_{j \in V} \deg(j)}.$$

For regular graphs, where every vertex has the same degree, each vertex has equal probability in the stationary distribution. On the grid with toric boundaries, which we use in Chapter 3, every vertex has degree four. Therefore, when starting a random walk on the grid, the initial vertex is simply chosen uniformly at random from the set of all vertices.

A random walk gives a means of exploring a graph using only local transitions. We can turn this into a search algorithm in the following way. We note that the behaviour of a random walk is completely specified by three factors: the transition matrix, the initial state, and the number of steps it is run for. It is therefore common to refer to the transition matrix P itself as a random walk, with the implication that the initial state is π and that the number of steps is either infinite or determined from context.

[1] Note that stochastic processes with this property can be referred to as memoryless. However, we reserve the term “memoryless" for a different context in this thesis, using it instead to refer to a class of quantum walks that do not store the previous location of the walker.

Algorithm 1 (Search via Random Walk) *Given a graph G with a set M of marked vertices, a random walk P on G, and a number of steps T, perform the following:*

1. *Choose a starting vertex X_0 according to the distribution π.*
2. *Execute the following T times:*
   - 2a) *Check if the current vertex is marked. If so, output the current vertex and terminate the algorithm.*
   - 2b) *Take one step of the random walk according to P.*
3. *Output "None".*

This probabilistic algorithm solves the spatial search problem with one-sided error. If the set M is empty, this algorithm always produces the correct output. If there are marked vertices in M, however, the algorithm could incorrectly return "None" if a marked vertex is not found in the given number of steps, T.

We emphasize that the search algorithm above consists of three distinct operations, for which the cost of each contributes separately to the total. We let S denote the cost of sampling from the initial state π. We also use C to denote the cost of checking whether a vertex is marked, and U (update) to denote the cost of moving from a vertex to its neighbour. With this notation, the total cost of Algorithm 1 is $\mathsf{S} + T(\mathsf{C} + \mathsf{U})$.

It therefore remains to choose an appropriate value for T. Recall that we assume G is strongly connected, so any vertex in G can be reached from any other vertex in G after a finite number of transitions according to P. By choosing a larger value for T, we can increase the probability that Algorithm 1 returns a marked vertex, provided one exists. The difficulty of finding a marked vertex depends on both the number of marked vertices and their placement. Thus, the appropriate choice of T depends on both the random walk P and the set M of marked vertices.

The quantity we wish to capture is known as the hitting time, and is defined as follows.

Definition 2 *Given a graph G, a random walk P on G, and a nonempty set M of marked vertices, the **hitting time** is the expected number of transitions according to P before the walk reaches a marked vertex, given that the initial vertex is unmarked. The hitting time is denoted $\mathsf{HT}(\mathsf{P}, M)$.*

The expectation in this case is taken with respect to both the randomness introduced by the movement of the walker and the randomness in the initial choice of starting vertex.

The hitting time exactly captures the asymptotic complexity of Algorithm 1. The hitting time is the expected number of steps before the algorithm will find a marked vertex, given one exists. Therefore, by Markov's inequality, setting T to be twice the worst-case hitting time will result in the algorithm returning the correct answer with probability at least $\frac{1}{2}$.

There are several approaches for computing the hitting time of a random walk. For a large class of random walks known as ergodic walks, there exists a spectral formula for computing the hitting time [Sze04, KMOR16]. One can show that the hitting time on the two-dimensonal $\sqrt{N} \times \sqrt{N}$ toric grid is $O(N \log N)$. The goal of quantum walks is to improve on the performance of random walks using quantum operations. Indeed, many random walks are able to achieve a quadratic speedup, within logarithmic factors, relative to the hitting time in the number of queries required.

Finally, we note that in this section, we have presented random walks as a means of developing spatial search algorithms. However, one of the primary motivations for studying random walks is their use in developing algorithms for a variety of other useful problems.

As an example of this, Aleliunas et al. [AKL⁺79] apply random walks in an algorithm for st-connectivity. In st-connectivity, the input is an undirected graph $G = (V, E)$ with $|V| = n$, $|E| = m$, and two designated vertices $s, t \in V$. The goal is to decide whether there is a path in G from s to t. A standard classical method for solving this problem would be to run a breadth-first search of G starting at vertex s. As soon as vertex t is encountered, the algorithm returns "True". If the search terminates without finding t, the algorithm returns "False". The step complex-

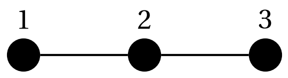


Figure 2.1: Consider a random walk on the graph above. If a walker starts at vertex 1, then after one step of the walk it will land at vertex 2. The same state will be reached after a single step of the walk if the walker begins at vertex 3. Thus, given that the walker is at vertex 2, it is not possible to uniquely determine the previous state of the walk.

ity of breadth-first search is optimal at $O(n+m)$. However, it requires up to n vertices to be stored at once in the queue, so its space complexity is $O(n \log n)$.

By exploring the input graph using a random walk, Aleliunas et al. give a bounded-error randomized algorithm that terminates in $O(nm)$ steps, at which point the walker outputs the correct answer with constant probability. Their algorithm only stores the current location of the walker, and therefore requires only $O(\log n)$ space in total. In this way, their algorithm improves on other approaches in its use of memory by means of a random walk.

2.2 Quantum walks

Translating random walks into the quantum setting leads to a class of algorithmic tools called quantum walks. The motivation for studying quantum walks is similar to that of random walks. Much as random walks have been a useful tool in developing efficient classical algorithms for a number of problems, quantum walks have proven to be a valuable tool in quantum algorithm design. Some examples of their applications include algorithms for element distinctness [Amb07], triangle finding [MSS07], and span program evaluation [Rei09]. Quantum walks have also been shown to be a universal model of computation [Chi09].

Quantum walks, like random walks, are required to satisfy the locality constraints imposed by the structure of the underlying graph. A quantum walk on a graph G must satisfy the restriction that in a single application of the walk operator, amplitude can only be moved between adjacent vertices in the graph.

One challenge when translating random walks to the quantum setting is the fact that quantum evolution must be unitary, and therefore invertible. Random walks by nature are not an invertible process (see Figure 2.1), nor is the transition matrix P itself unitary. There have been multiple classes of quantum walk developed, each of which address this challenge in a different way.

The continuous time quantum walk [CFG02, CG04] evolves an initial quantum state in continuous time using the operator $e^{\mathsf{H}t}$, where the parameter t represents time. Here, H is a $n \times n$ matrix such that $\mathsf{H}_{ij} = 0$ unless vertex i is adjacent to vertex j. Common choices for H include the adjacency matrix and the graph Laplacian, under the constraint that $e^{\mathsf{H}t}$ must be unitary. In this sense, the continuous walk operates directly on the vertices of graph, since it acts on a space of dimension n. There are a number of notable results proven for continuous walks, including an exponential speedup over any classical algorithm for searching certain black-box graphs [CCD+03]. On the other hand, continuous walks as defined above are not known to provide a quadratic speedup over random walks for spatial search on general graphs. On the $\sqrt{N} \times \sqrt{N}$ toric grid, the continuous walk defined by Childs and Goldstone [CG04] requires $\Omega(N)$ queries, which is no better than classical algorithms. By allowing H to have dimension $n^2 \times n^2$, it is possible to obtain a quadratic speedup over random walks for any graph and any number of marked vertices [ACNR21], although this requires operating on a larger Hilbert space.

In the discrete-time setting, one method to address the invertability requirement is to introduce extra quantum registers to track the previous location of the walker. This approach is taken in both coined [AAKV01, ABN+01] and Szegedy [Sze04] quantum walks.

Coined quantum walks are defined on **regular graphs**, where each of the n vertices has the same degree, d. A coined quantum walk is defined on the space $\mathbb{C}^d \oplus \mathbb{C}^n$, where the space $\mathbb{C}^d$ stores the state of a d-dimensional quantum coin. Updating the location of the walker involves “flipping" this coin and then updating the vertex of the walker conditioned on the result. Coined quantum walks are used in many of the first algorithms based on quantum walks [Amb07, MSS07, SKW03]. Ambainis et al. [AKR05] show that on the $\sqrt{N} \times \sqrt{N}$ toric grid with a unique marked vertex, the probability of measuring a marked vertex using a coined quantum walk is maximized at $T = \Theta(\sqrt{N \log N})$ steps. Measuring after this many steps of the walk yields a marked vertex with probability $O(\frac{1}{\log N})$, so by using amplitude amplification, the total query complexity of their algorithm is $O(\sqrt{N} \log N)$.

Szegedy quantum walks [Sze04] give a general approach to constructing a quantum walk from a Markov chain. They generalize coined quantum walks, and on graphs where coined walks can be defined, the two models are isometrically equivalent. On a graph with n vertices, Szegedy quantum walks operate on the space $\mathbb{C}^n \oplus \mathbb{C}^n$. Each of these two registers of dimension n encodes a vertex, so as a pair, the two registers capture a directed edge of the graph. In this way, Szegedy walks can be considered to be walking on the edges of the graph rather than the vertices. The walk state is considered marked if the vertex in the first register is marked.

Szegedy shows how to construct a walk operator consisting of two operations. The first operation is a reflection that distributes amplitude between the neighbours of the vertex in the first register, and the second operation swaps the contents of the two registers. Using this operator, Szegedy gives a quantum algorithm for spatial search. Szegedy's walk can decide whether a graph contains marked vertices with $O(\sqrt{\mathsf{HT}(\mathsf{P}, M)})$ quantum queries, giving a quadratic speedup over classical random walks. For certain regular graphs with a single marked vertex, such as the toric grid, the Szegedy quantum walk can also find a marked vertex with $O(\sqrt{\mathsf{HT}(\mathsf{P}, M)})$ quantum queries.

An important modification to Szegedy quantum walks is that of interpolated walks. First proposed by Krovi et al. [KMOR16], interpolated walks introduce weighted selfloops on the marked vertices in the underlying graph. A selfloop is simply an edge connecting a vertex v to itself, captured by the state $|v, v\rangle$. Adjusting the weight of the selfloop edge changes the dynamics of the interpolated quantum walk. Krovi et al. [KMOR16] show that this approach can be used to find a marked vertex on any graph. Their algorithm gives a quadratic speedup over the classical hitting time in the case where only a single vertex is marked. Ambainis et al. [AGJK20] strengthen this result by proving that interpolated walks can find a marked vertex on any graph, with any number of marked vertices, with the same asymptotic improvement.

Memoryless quantum walks are a separate class of discrete-time quantum walk. We discuss the development of memoryless walks in Chapter 3. Unlike coined or edge-based quantum walks, memoryless walks do not use extra registers to capture the previous location of the walker. Instead, memoryless walks operate directly on the vertex space of the graph. Amplitude is distributed across the graph using alternating reflections derived from vertex tessellations. Memoryless walks thus minimize the amount of quantum memory required.

Our new quantum walk in Chapter 3 is an example of a memoryless walk. It operates on a space of dimension $N+1$, which is minimal for the problem. Our walk finds a single marked vertex on the N-vertex square grid in $O(\sqrt{N \log N})$ quantum queries, with success probability asymptotically approaching one. This matches the performance of the best quantum walks for the problem, and is the first walk to achieve these parameters with optimal quantum memory. To obtain this result, we adapt the selfloop technique of Krovi et al. [KMOR16] to the memoryless setting. We introduce a weighted extra state, denoted $|\circlearrowleft\rangle$, that can exchange amplitude with the marked vertex, and which alters the spectrum of the walk operator. We show that this state forces the action of the walk into a two-dimensional subspace, and we use this property to bound the algorithm's complexity. We believe that our success with using this technique to obtain optimal parameters for spatial search on the grid shows new potential for the application and further development of memoryless quantum walks.

2.3 Span programs

Span programs are a linear-algebraic model of computation introduced by Karchmer and Wigderson [KW93]. Informally, a span program computes a boolean function by determining whether a fixed target vector lies in an input-dependent linear space. If the target vector lies in this space, then the function evaluates to one on the input, otherwise it evaluates to zero.

Span programs are closely connected to quantum algorithms, and are the primary tool we use to design our algorithm in Chapter 4. In this section, we define the terminology associated with span programs and discuss how complexity is measured in this model. We illustrate this discussion with two examples of span programs in Section 2.3.1.

There are multiple equivalent definitions for span programs. We use the following, as presented by Belovs and Reichardt [BR12].

Definition 3 *A* ***span program*** $\mathscr{P}$ *is a tuple* $(n, d, |\tau\rangle, V_{\text{free}}, \{V_{i,b}\})$*, where*

- *inputs to the span program are elements of* $\{0,1\}^n$*,*
- $|\tau\rangle \in \mathbb{C}^d$ *is the* ***target*** *vector,*
- $V_{\text{free}} \subseteq \mathbb{C}^d$ *is a set of* ***free*** *vectors,*
- $V_{i,b} \subseteq \mathbb{C}^d$ *for* $1 \leq i \leq n$*,* $b \in \{0,1\}$*, and*
- $V = \bigcup V_{i,b}$ *is the set of* ***input*** *vectors.*

The span program $\mathscr{P}$ *computes a boolean function* $f_{\mathscr{P}} : \{0,1\}^n \to \{0,1\}$*. Given an input* $x \in \{0,1\}^n$*, the* ***available*** *vectors are defined by*

$$V(x) = V_{\text{free}} \cup \left(\bigcup_{i=1}^{n} V_{i,x_i} \right).$$

Then $f_{\mathscr{P}}(x) = 1$ *if and only if* $|\tau\rangle \in \text{span}(V(x))$.

We also use V, V_{free} and $V(x)$ to denote the matrices whose columns are the vectors in the respective sets.

Classically, span programs have been studied for their applications to complexity theory and secret sharing schemes [KW93]. They also have a surprisingly tight and beautiful connection to quantum algorithms, which was first discovered by examining read-once formulas [RŠ12]. As shown by Reichardt [Rei09, Rei11], any span program can be compiled into a bounded-error quantum algorithm for the same function. This is done by converting the span program to a bipartite graph and applying phase estimation, with the complexity determined by a span program quantity known as the witness size.

Intuitively, one would expect the cost of evaluating a span program to be high when $|\tau\rangle$ is almost orthogonal to the span of the available vectors. The witness size captures this difficulty, which in turn characterizes the query complexity of the corresponding quantum algorithm.

Definition 4 *Consider a span program $\mathscr{P}$ and an input $x \in \{0,1\}^n$.*

- *If $f_{\mathscr{P}}(x) = 1$, then there exist vectors $|w_{\text{free}}\rangle$ and $|w^+\rangle$ such that $V_{\text{free}}|w_{\text{free}}\rangle + V(x)|w^+\rangle = |\tau\rangle$. These vectors contain the coefficients of a linear combination of available vectors producing the target. Together, they form a **positive witness** for x. The size of the witness is defined to be $\||w^+\rangle\|^2$.*
- *If $f_{\mathscr{P}}(x) = 0$, then $|\tau\rangle$ is orthogonal to the available vectors. Therefore, there exists a vector $|w^-\rangle$ such that $\langle w^-|\tau\rangle = 1$ and $|w^-\rangle \perp \text{span}(V(x))$. The vector $|w^-\rangle$ is a **negative witness** for x with size $\|V^\dagger|w^-\rangle\|^2$. This quantity is the sum of the squared inner products of $|w^-\rangle$ with all the vectors in V that are not available.*

The positive witness size of a span program $\mathscr{P}$ on input x is the minimum size over all positive witnesses for x, denoted $\text{wsize}_1(\mathscr{P}, x)$. *The negative witness size of $\mathscr{P}$ on x is defined similarly, and is denoted* $\text{wsize}_0(\mathscr{P}, x)$. *Then for $b \in \{0,1\}$, we define*

$$\text{wsize}_b(\mathscr{P}) = \max_{\substack{x \in \{0,1\}^n \\ f_{\mathscr{P}}(x) = b}} \text{wsize}_b(\mathscr{P}, x).$$

*The **witness size** of a span program $\mathscr{P}$ is defined to be*

$$\text{wsize}(\mathscr{P}) = \sqrt{\text{wsize}_0(\mathscr{P})\text{wsize}_1(\mathscr{P})}.$$

Reichardt gives a transformation that can convert a span program $\mathscr{P}$ into a bounded-error quantum algorithm with query complexity $O(\text{wsize}(\mathscr{P}))$. This supports the use of witness size as an appropriate measure of span program complexity. The details of Reichardt's transformation are beyond the scope of this thesis. However, we require Reichardt's main result, which is the following.

Theorem 1 ([Rei11]) *For any boolean function $f : \{0,1\}^n \to \{0,1\}$, if $\mathscr{P}$ is a span program computing f, then there exists a quantum algorithm that evaluates f with two-sided bounded error using $O(\text{wsize}(\mathscr{P}))$ quantum queries.*

By Theorem 1, span program witness size gives an upper bound on the quantum query complexity of a boolean function. In fact, witness size *exactly* characterizes quantum query complexity, up to a constant factor. For any boolean function, the general adversary bound of Høyer, Lee, and Špalek [HLŠ07] gives a tight lower bound on its quantum query complexity. The general adversary bound is expressed as the solution to a semidefinite program, whose dual is a semidefinite program that minimizes witness size. Using this property, Reichardt [Rei09] shows that for any boolean function, the optimal span program witness size is exactly equal to the general adversary bound. This reduces the problem of finding query-optimal quantum algorithms to finding span programs with optimal witness size.

Span programs are thus a very appealing tool for the design of quantum algorithms. The model does not directly capture any notion of quantum dynamics, and yet can be used to construct query-optimal quantum algorithms. As a result, span programs have been used to design some highly intuitive and query-efficient quantum algorithms. Just a few examples of this include algorithms for st-connectivity and subgraph detection [BR12], graph bipartiteness [Āri16], and formula evaluation [RŠ12, Rei09, JK17]. In Section 4.3, we add to this list by giving a new span program for 0-determinant verification, which we then apply in Section 4.4 to decide whether a graph contains a perfect matching.

2.3.1 Two span programs

To illustrate, we present two examples of span programs. The first is a simple span program that computes the OR function on n bits. The second is an elegant span program by Belovs and Reichardt [BR12] that solves the graph problem of st-connectivity. In both cases, we define the span program and analyse its complexity by computing the witness size.

One of the simplest span programs is the following span program for computing the logical OR function on n bits of input. We let $\mathscr{P} = (n, d, |\tau\rangle, V_{\text{free}}, \{V_{i,b}\})$, where $d = 1$, $|\tau\rangle = [1]$, $V_{\text{free}} = \emptyset$, and $V_{i,0} = \emptyset$ and $V_{i,1} = \{[1]\}$ for $1 \leq i \leq n$. With this definition, we can see that if any of the n input bits are equal to one, then [1] will be included in the available vectors. If all input bits are zero, then there will be no available vectors. Thus, the target $|\tau\rangle$ is in the span of the available vectors if and only if at least one of the n input bits is equal to one. In this way $\mathscr{P}$ computes the OR function on the n input bits as claimed.

To compute the witness size of this span program $\mathscr{P}$, we consider the positive and negative cases separately.

- If $f_{\mathscr{P}}(x) = 1$, then k of the input bits must be equal to one, for some $k \geq 1$. In this case, the minimum-norm positive witness $|w^+\rangle$ has entries $\frac{1}{k}$ for all indices i where $x_i = 1$, and is zero everywhere else. The positive witness size for input x is then $\||w^+\rangle\|^2 = \frac{1}{k}$. This quantity is maximized when $k = 1$, so $\text{wsize}_1(\mathscr{P}) = 1$.

- If $f_{\mathscr{P}}(x) = 0$, then all of the input bits are zero. In this case, $|\tau\rangle$ is orthogonal to all available vectors (since $V(x) = \emptyset$). The negative witness $|w^-\rangle$ must be exactly [1] in this case, due to the requirement $\langle w^-|\tau\rangle = 1$. The negative witness size is given by the sum of the squared inner products of $|w^-\rangle$ with all of the unavailable vectors. There are n unavailable vectors in this case, all equal to [1], so $\text{wsize}_0(\mathscr{P}) = n$.

Combining these cases, we obtain a total witness size of

$$\text{wsize}(\mathscr{P}) = \sqrt{\text{wsize}_0(\mathscr{P})\text{wsize}_1(\mathscr{P})} = \sqrt{n}.$$

Thus, the given span program can be converted to a quantum algorithm that computes the OR function on n input bits in $O(\sqrt{n})$ quantum queries. This matches the asymptotic query complexity of amplitude amplification, and is known to be optimal for the problem.

To show another case where a span programs yields a query-optimal quantum algorithm, we also present a span program for st-connectivity developed by Belovs and Reichardt [BR12]. This is an example of a span program being used to decide whether a graph has a given property, similar to our span program-based algorithm for perfect matchings in Chapter 4.

Recall that in st-connectivity, the input is an undirected graph $G = (V, E)$ with $|V| = n$, and two designated vertices $s, t \in V$, and the goal is to decide whether there is a path in G from s to t. Because span programs compute boolean functions, we will consider the input to be the adjacency matrix for G. We index this input using tuples (u,v), where $1 \le u, v \le n$, and the value at index (u,v) is one if and only if $(u,v) \in E$.

A span program solving st-connectivity can be defined as follows. Let $\mathscr{P} = (n, d, |\tau\rangle, V_{\text{free}}, \{V_{i,b}\})$, where $d = n$. In the space $\mathbb{C}^n$, we identify each of the n vertices with a standard basis state, so vertex k corresponds to state $|k\rangle$. Using this correspondence, let $|\tau\rangle = |s\rangle - |t\rangle$, and set $V_{\text{free}} = \emptyset$. The input-dependent vectors are defined by $V_{(u,v),1} = \{|u\rangle - |v\rangle\}$ and $V_{(u,v),0} = \emptyset$.

We now argue both correctness and witness size simultaneously, separating the positive and negative cases.

- If there is a path from s to t in G, then $|\tau\rangle$ will be in the span of the available vectors. The edges in the path from s to t give a linear combination of available vectors that produces $|\tau\rangle$. For example, if there is a path from s to t of the form $(s, x v_1, v_2, \ldots, v_k, t)$, then $|s\rangle - |v_1\rangle$, $|v_i\rangle - |v_{i+1}\rangle$ for $1 \le i \le k-1$, and $|v_k\rangle - |t\rangle$ are all in the span of the available vectors, and their sum is equal to $|\tau\rangle$. A path from s to t can contain at most $n-1$ edges, so $\| |w^+\rangle \|^2 \le n-1$ for any input graph. Thus, $\text{wsize}_1(\mathscr{P}) = n - 1 \in O(n)$.

- If there is no path from s to t in G, then s and t are in separate connected components of the graph. In this case, we can define $|w^-\rangle$ by $\langle w^-|v\rangle = 1$ if and only if v is in the same connected component as vertex s. Then $\langle w^-|\tau\rangle = \langle w^-|s\rangle = 1$. Because all edges in the graph connect two vertices in the same connected component as s, or two vertices in a different component from s, $|w^-\rangle$ is orthogonal to all of the available vectors. Thus, $|w^-\rangle$ is a negative witness for G.

  To compute the negative witness size, note that each vector that is not available corresponds to an edge that is not in the graph. The squared inner product of $|w^-\rangle$ with each unavailable vector is at most 2, and there are $O(n^2)$ unavailable vectors. Therefore, $\text{wsize}_0(\mathscr{P}) \in O(n^2)$.

Combining these two results, we can see that

$$\mathrm{wsize}(\mathscr{P}) = \sqrt{\mathrm{wsize}_0(\mathscr{P})\mathrm{wsize}_1(\mathscr{P})} \in O(n^{3/2}).$$

This matches the lower bound of $\Omega(n^{3/2})$ on the query complexity of st-connectivity proven by [DHHM06]. Thus, Belovs' and Reichardt's span program yields a query-optimal quantum algorithm.

Finally, we recall that Reichardt proved that for any boolean function, the optimal span program witness size is exactly equal to the general adversary bound [Rei09]. This guarantees that for any boolean function, there is a span program computing it that can be compiled into a query-optimal quantum algorithm. However, finding optimal span programs is not a simple task. So far, there are limited techniques for developing span programs, and few span programs are as naturally structured as the examples given in this section.

In Chapter 4 we give a new span program that decides whether an input matrix is singular. Our span program has two unusual elements in its design. First, the available vectors in our span program have some randomized values. We then show that the our span program's witness size is within our stated bound with probability at least $\frac{5}{6}$ under this distribution. This is sufficient to show that when our span program is converted into a quantum algorithm, our bound on the witness size also applies to the query complexity of the algorithm. Second, our span program is designed so that the available vectors each depend on multiple bits of the input. To achieve this, we apply a higher-level construction also used by Belovs [Bel11], which we discuss in Section 4.3 and illustrate with an example in Section 4.4.1.

These techniques are another reason our span program from Chapter 4 is interesting, beyond its applications to the matching problem. We believe that these techniques could be useful in the search for new span programs, and by extension, the development of query-efficient quantum algorithms.

Chapter 3

Memoryless quantum search

3.1 Introduction

Search problems are one of the foundational applications of quantum algorithms, and are one of the situations in which quantum algorithms are proven to provide a speedup over their classical counterparts. For example, Grover's search algorithm [Gro96] can find a single element in an N-element database with $\Theta(\sqrt{N})$ quantum queries, while any classical algorithm requires $\Omega(N)$ queries for the same task.

In spatial search, the goal is to find a "marked" element on a graph, with the restriction that in a single time step, amplitude can only be moved between adjacent vertices on the graph. Such restrictions can emerge from some underlying physical structure [AA05], or from the computational cost of moving from one vertex to another [Amb07]. Quantum spatial search was first considered by Benioff [Ben02], who showed that direct application of Grover's search on the grid does not yield a speedup over classical algorithms. A near-quadratic speed-up was then discovered using a divide-and-conquer quantum algorithm [AA05], and using quantum walks [AKR05, CG04, San08].

Of the types of quantum walk that have been developed, memoryless, or coinless, walks have several advantages. Foremost among these is that they operate directly on the vertex space of the graph. This differentiates them from coined or Szegedy-style quantum walks, which use extra registers to encode the previous location of the walker. Memoryless walks can therefore be used to produce spatial search algorithms with optimal memory requirements, as captured by their name. They also have a simple structure, alternating two or more non-commuting reflections that can be derived from tessellations of the underlying graph. This simplicity makes them a natural candidate for implementation. A proposal for implementing memoryless walks using superconducting microwave resonators is given by [MdOP17], and an implementation of memoryless walks on IBM quantum computers is in [AAMP20].

Despite their advantages, memoryless walks are less commonly studied than other types of quantum walk. Most of the literature on memoryless walks considers staggered walks, a class of memoryless walks defined by Portugal et al. in [PSFG16]. Staggered walks are based on tessellating a graph with cliques, which obeys the spatial search constraint and gives a method for constructing memoryless walks on general graphs. Ref. [PSFG16] shows that any quantum walk on a graph G in the standard Szegedy model [Sze04] can be converted to a staggered walk on the underlying line graph of G. The conversion preserves the asymptotic cost, the success probability, and the space requirement. Staggered walks have similarly been used to derive relationships between memoryless, coined, Szegedy, and continuous quantum walks [PBF15, Por16a, Por16b, PdOM17, CP18, KPSS18].

Memoryless walks have been applied to the spatial search problem on the grid in the following cases. A memoryless walk on the line was given by Patel, Raghunathan, and Rungta in [PRR05], who give and analyse a walk with Hamiltonians, noting that their walk operator resembles the staggered fermion formalism. In [PRR10], Patel, Raghunathan and Rahaman present numerical simulations on the extension of this walk to the N-vertex grid. Their results show that it finds a unique marked vertex in $\Theta(\sqrt{N \log N})$ applications of the walk operator with success probability $\Theta(\frac{1}{\log N})$, and that by using an ancilla qubit, the success probability can be improved to $\Theta(1)$.

Falk [Fal13] gives a memoryless walk on the two-dimensional grid, constructing a discrete walk operator by reflecting about two alternating tessellations of the grid. The walk by Falk was analysed by Ambainis, Portugal and Nahimovs [APN15], who proved that it finds a unique marked vertex on the grid using $\Theta(\sqrt{N\log N})$ applications of the walk operator with success probability $\Theta(\frac{1}{\log N})$. Portugal and Fernandes [PF17] give a memoryless walk with Hamiltonians that also finds a unique marked vertex on the two-dimensional grid with $\Theta(\sqrt{N\log N})$ applications of the walk operator and success probability $\Theta(\frac{1}{\log N})$.

The success probability of [PRR05], [Fal13] and [PF17] is sub-constant. The success probability can be improved to $\Theta(1)$ by applying amplitude amplification [BHMT02], but that would increase the number of steps by a factor of $\Theta(\sqrt{\log N})$ and introduce additional operators in an implementation. Two alternative methods [APN15, PF17] for increasing the success probability are the post-processing local neighborhood search used in [ABN$^+$12] and Tulsi's proposal of adding an ancilla qubit to the grid [PRR10, Tul08].

In this chapter, we present a memoryless walk that finds the marked vertex in $\Theta(\sqrt{N\log N})$ steps with vanishing error probability. Our walk uses minimal memory and preserves the simple structure given by alternating tessellations of the grid. To do this, we show how the interpolated walks of Krovi et al. [KMOR16], which introduce selfloops to walks on the edge space of a graph, can be adapted to the vertex space of a graph. We define our memoryless walk using the tessellations shown in Figure 3.1, which divide the grid into squares of size 2×2. This is the same tessellation structure analysed by [APN15], who proved that using Falk's construction, the maximum success probability of the corresponding memoryless walk scales with $\Theta(\frac{1}{\log N})$. By using a selfloop to force the action of the walk into a single two-dimensional subspace, we modify their walk so that the maximum success probability asymptotically approaches one.

Classically, the interpolated version of a random walk is constructed by adding weighted selfloop edges to marked vertices. Applying Szegedy's isometry [Sze04] to interpolated walks produces quantum walks that can find a unique marked vertex on any graph [KMOR16]. This approach has recently been used to solve the spatial search problem on any graph with a quadratic speedup over classical random walks, even for the case of multiple marked vertices. Solutions of this form have been obtained for both the discrete [AGJK20, AGJ21] and continuous [ACNR21] models using quantum walks that operate on the edge space of a graph.

To extend the approach of [KMOR16] to the memoryless setting, we introduce a new state corresponding to a selfloop on the marked vertex. Rather than reflecting about the marked vertex, we reflect about an interpolation between the selfloop state and the marked vertex. The result is a memoryless walk parametrized by the weight of the selfloop, s. We prove in our analysis that with our explicit choice of weight, this forces the evolution of the initial state into a single rotational subspace of our walk operator. As a result, our walk achieves a success probability $1-O(\frac{1}{\log N})$ with $\Theta(\sqrt{N \log N})$ steps and minimal memory.

Most of our analysis considers the eigenvector of an operator with the smallest positive eigenphase. We use the term "slowest eigenvector" to refer to this eigenvector, and "slowest rotational subspace" to refer to the subspace spanned by this eigenvector and its conjugate.

To prove our main result, we present a set of techniques for analysing memoryless walks. We analyse an operator W, whose spectrum is completely known, composed with a two-dimensional rotation F. As part of our proof, we determine the asymptotic behaviour of both the smallest positive eigenphase of WF and its associated eigenvector. The result is a precise asymptotic description of the slowest rotational subspace of our walk operator. The techniques we use to obtain this description are general enough that they could be applied in other contexts as well.

We begin by defining our walk in Section 3.2. An overview of the chapter layout and proof structure is given in Section 3.2.1.

3.2 Walk construction and main result

We consider the task of finding a unique marked vertex on a two-dimensional grid with n_r rows and n_c columns, where both n_r and n_c are even. The grid boundaries are those of a torus, so there are edges between vertices $(i, n_c - 1)$ and $(i, 0)$ for $0 \leq i < n_r$, and between vertices $(n_r - 1, j)$ and $(0, j)$ for $0 \leq j < n_c$. The total number of vertices is given by $N = n_r \times n_c$. After Lemma 5, we restrict to the case where $n_r = n_c = \sqrt{N}$.

We construct a memoryless quantum walk with a selfloop. Our walk operates on a space of dimension $N+1$, which is optimal for this task. The vertex at position (i,j) is represented by the quantum state $|i,j\rangle$. In addition to these N orthogonal basis states, we introduce a new state $|\circlearrowleft\rangle$ corresponding to a self-loop on the marked vertex. The state $|\circlearrowleft\rangle$ is a basis state orthogonal to the vectors $|i,j\rangle$, giving a space of total dimension $N+1$. Our approach and terminology are inspired by the interpolated walks of Krovi et al [KMOR16], in which additional basis states are added to a Szegedy-style walk. In their Szegedy-style interpolated walks, the new states correspond classically to adding selfloop edges to the marked vertices, which is how our state $|\circlearrowleft\rangle$ gets its name.

Our walk applies two alternating reflections about the faces of the graph. Here, we follow the construction used in [APN15]. For $0 \le i < \frac{n_r}{2}$, $0 \le j < \frac{n_c}{2}$, define

$$|a_{ij}\rangle = \frac{1}{2}\sum_{i',j'=0}^{1} |2i+i', 2j+j'\rangle, \tag{3.1}$$

$$|b_{ij}\rangle = \frac{1}{2}\sum_{i',j'=0}^{1} |2i+1+i', 2j+1+j'\rangle. \tag{3.2}$$

The sets $\{|a_{ij}\rangle\}_{i,j}$ and $\{|b_{ij}\rangle\}_{i,j}$ each specify a partition of the grid into 2×2 squares, positioned at even and odd indices, respectively. These sets form the tessellations depicted in Figure 3.1.

Define projections onto the even and odd partitions as

$$\Pi_e = \sum_{i=0}^{\frac{n_r}{2}-1}\sum_{j=0}^{\frac{n_c}{2}-1} |a_{ij}\rangle\langle a_{ij}|, \qquad \Pi_o = \sum_{i=0}^{\frac{n_r}{2}-1}\sum_{j=0}^{\frac{n_c}{2}-1} |b_{ij}\rangle\langle b_{ij}|,$$

and let

$$\mathsf{A} = 2\Pi_e + 2|\circlearrowleft\rangle\langle\circlearrowleft| - \mathsf{I}, \tag{3.3}$$

$$\mathsf{B} = 2\Pi_o + 2|\circlearrowleft\rangle\langle\circlearrowleft| - \mathsf{I}. \tag{3.4}$$

In [APN15], these reflections are alternated with a reflection about the marked state. This is the standard way of adding finding behaviour to memoryless walks,

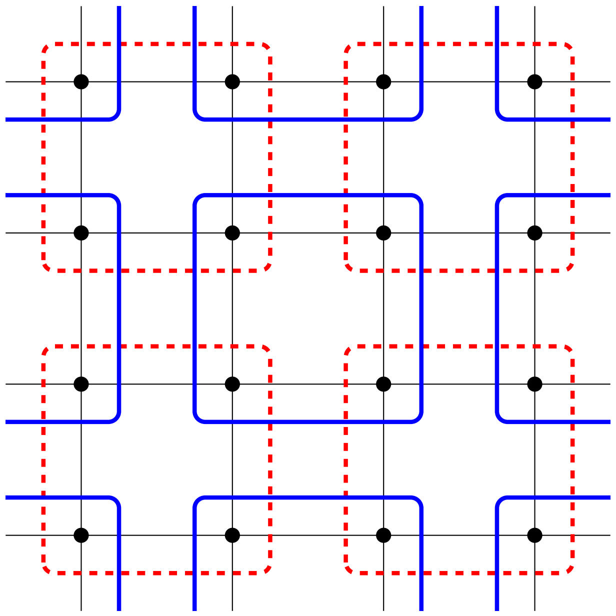


Figure 3.1: Staggered tessellations of the two-dimensional lattice into squares of size 2×2. The dashed red lines represent the states $|a_{ij}\rangle$ and solid blue lines represent the states $|b_{ij}\rangle$, as defined in equation (3.1) and equation (3.2). The construction based on alternating reflections about this tessellation structure was originally proposed by [Fal13].

where an operator composed of two or more non-commuting reflections is alternated with a reflection of the marked vertices. In our walk, we replace the reflection of the marked vertex with a reflection of an interpolated state with parameter $0 \leq s \leq 1$. Letting $|g\rangle$ denote the marked vertex, we define the interpolated state to be $|\tilde{g}\rangle = \sqrt{s}|g\rangle + \sqrt{1-s}|\circlearrowleft\rangle$. Our input-dependent reflection is then

$$\tilde{\mathsf{G}} = \mathsf{I} - 2|\tilde{g}\rangle\langle\tilde{g}|. \tag{3.5}$$

By selecting an appropriate value for s, we are able to force the action of the walk asymptotically into a single two-dimensional subspace. We set $s = 1 - \frac{1}{N+1}$, which is close to the value $s = 1 - \frac{1}{N}$ used by [KMOR16] for interpolated walks.

Given a fixed value for s, we define a single step of our walk to be the operator

$$\mathsf{U} = \mathsf{B}\tilde{\mathsf{G}}\mathsf{A}\tilde{\mathsf{G}}. \tag{3.6}$$

We apply the walk to the initial state $|\pi\rangle$, which is defined by $\langle i,j|\pi\rangle = \frac{1}{\sqrt{N}}$ for all vertices (i,j), and with $\langle \circlearrowleft|\pi\rangle = 0$. This allows us to present our main result.

Theorem 2 (Main result) *Fix $s = 1-\frac{1}{N+1}$ and suppose $n_r = n_c$. Then there exists a constant $c > 0$ such that after $c\sqrt{N\log N}$ applications of U to $|\pi\rangle$, measuring the state will produce $|\circlearrowleft\rangle$ with probability $1-e(N)$, where $e(N) \in O(\frac{1}{\log N})$.*

We remark that given the state $|\circlearrowleft\rangle$, one can obtain the marked state $|g\rangle$ through amplitude amplification [BHMT02]. This can be done by alternating the reflection $\tilde{\mathsf{G}}$ with a reflection about either $|g\rangle$ or $|\circlearrowleft\rangle$. After $\lfloor \frac{\pi}{2}(\arcsin(\frac{1}{\sqrt{N+1}}))^{-1}\rfloor \in \Theta(\sqrt{N})$ steps of amplification, measuring the resulting state will produce $|g\rangle$ with probability $1-e(N)$, where $e(N) \in O(\frac{1}{N})$.

Note that both the error probability and the query complexity for obtaining $|g\rangle$ from $|\circlearrowleft\rangle$ are dominated by the cost of finding the state $|\circlearrowleft\rangle$ as in Theorem 2. Thus, this final step does not affect the asymptotic parameters of our memoryless search algorithm. We also emphasize that this final amplitude amplification is conceptually different from applying amplitude amplification or Tulsi's method as suggested for previous memoryless walks. Both of the latter methods involve running the entire quantum walk nested inside amplitude amplification, or with a global control qubit. Ours is a simple adjustment to the final state of the walk, constructed from the existing operator $\tilde{\mathsf{G}}$ and a reflection about $|g\rangle$ or $|\circlearrowleft\rangle$.

3.2.1 Proof strategy

Our proof of Theorem 2 is based on the analysis of an intermediate walk operator, which we define below.

To simplify the analysis, we also introduce a change of basis. This will allow us to compute necessary properties of the walk's spectrum, as the eigenvectors of BA factor into product states under this basis change. Let cz be an operator with the action defined by $|i,j\rangle \mapsto -|i,j\rangle$ if both i and j are even, and being the identity otherwise. If the coordinates i and j are represented as a tensor product of qubits, then cz only operates on the least-significant bits in the tensor product space. Let cz act trivially on $|\circlearrowleft\rangle$. For any operator X, let $\mathsf{X}_z = \mathsf{czXcz}$, and let $|\pi_z\rangle = \mathsf{cz}|\pi\rangle$.

We define

$$\mathsf{W} = (\mathsf{BA})_z, \tag{3.7}$$

$$\mathsf{F} = (\mathsf{A}\tilde{\mathsf{G}}\mathsf{A}\tilde{\mathsf{G}})_z, \tag{3.8}$$

so that

$$\mathsf{U} = \left(\mathsf{BA}\right)\left(\mathsf{A}\tilde{\mathsf{G}}\mathsf{A}\tilde{\mathsf{G}}\right) = (\mathsf{WF})_z. \tag{3.9}$$

This is analogous to the decomposition of U used by [APN15] in their analysis. Recall that our algorithm from Theorem 2 consists of applying the operator U to $|\pi\rangle$. In the analysis we give for this algorithm, we instead consider applying the operator $\mathsf{U}_z = \mathsf{WF}$ to the initial state $|\pi_z\rangle$. Using the fact that

$$\mathsf{U}_z^k|\pi_z\rangle = \mathsf{cz}\mathsf{U}^k|\pi\rangle$$

for any positive integer k, we can see that the probability of measuring $|\circlearrowleft\rangle$ in state $\mathsf{U}_z^k|\pi_z\rangle$ is the same as in state $\mathsf{U}^k|\pi\rangle$. Thus, using U_z and $|\pi_z\rangle$ for our analysis simplifies the calculations, while maintaining the same success probability as our algorithm from Theorem 2.

Note that W is input-independent, with eigenvectors that factor into product states as we show in Section 3.4. Meanwhile, F is input-dependent, and depends on both s and the marked vertex. We show in Section 3.3 that F is a two-dimensional rotation, and therefore can be decomposed as the product of two reflections, which we write as $\mathsf{F} = \mathsf{F}_1\mathsf{F}_2$.

Our intermediate walk consists of W composed with only F_1. This choice of intermediate walk has the advantage that it consists of a real operator whose spectrum is completely known, composed with a one-dimensional reflection. This allows us to apply existing results about operators of this type, including the eigenvector analysis of [Amb07] and the flip-flop theorem from [DH17]. An overview of these results is given in Appendix 5.

Our proof is based on a tight characterization of the slowest rotational subspace of the intermediate walk WF_1. This characterization is developed in Section 3.5, where we prove asymptotic properties of the rotational angle and the spanning vectors. To our knowledge, this is the first case of the flip-flop theorem being used to derive properties of a subspace in this way. We show that with our choice of selfloop, the action of WF_1 on $|\pi_z\rangle$ can be reduced asymptotically to a Grover-like rotation in the slowest two-dimensional rotational subspace of WF_1. This key property is what allows our main algorithm to achieve a success probability asymptotically close to 1.

To prove our main result, we relate the slowest rotational subspaces of WF_1 and WF in Section 3.6. We show that composing WF_1 with the reflection F_2 does not significantly alter the slowest rotational subspace, and therefore that WF has the same asymptotic behaviour as WF_1 when applied to $|\pi_z\rangle$. The proof of Theorem 2 follows from the basis-change relationship between U and WF given in equation (3.9).

With our approach, we are able to derive precise statements about the behaviour of an operator composed with a two-dimensional rotation. This addresses a more general challenge in analysing quantum algorithms, and may have applications outside of memoryless walks.

3.3 Decomposition of F

In this section, we derive the exact form of the rotation F. We show that it can be decomposed into two one-dimensional reflections, F_1 and F_2, which we compose sequentially with W in Sections 3.5 and 3.6.

Without loss of generality, assume $|g\rangle = |0,0\rangle$ is the marked vertex. Consider the three-dimensional subspace spanned by $|g\rangle$, $|\circlearrowleft\rangle$ and $|a_{00}\rangle$, the even-indexed square containing $|0,0\rangle$. The operator F only acts non-trivially in this subspace, which is spanned by $|\circlearrowleft\rangle$ and the two orthonormal states

$$|+\rangle = \frac{1}{\sqrt{3}}(|g\rangle + |a_{00}\rangle), \tag{3.10}$$

$$|-\rangle = \qquad (|g\rangle - |a_{00}\rangle). \tag{3.11}$$

Lemma 1 *Let* $0 \le \eta \le \frac{\pi}{3}$ *be such that*

$$\sin^2(\eta) = \frac{3}{4}s. \tag{3.12}$$

Set $\lambda = e^{\imath 4\eta}$. *Then* $\mathsf{F} = \mathsf{I} + (\lambda - 1)|f_+\rangle\langle f_+| + (\lambda^{-1} - 1)|f_-\rangle\langle f_-|$, *where the two non-trivial eigenvectors are*

$$|f_+\rangle = \frac{1}{\sqrt{2}}\left[|+\rangle - \imath \frac{1}{\sqrt{4-3s}}\Big(\sqrt{s}|-\rangle - 2\sqrt{1-s}|\circlearrowleft\rangle\Big)\right], \tag{3.13}$$

$$|f_-\rangle = \frac{1}{\sqrt{2}}\left[|+\rangle + \imath \frac{1}{\sqrt{4-3s}}\Big(\sqrt{s}|-\rangle - 2\sqrt{1-s}|\circlearrowleft\rangle\Big)\right]. \tag{3.14}$$

Proof Observe that F is a real-valued operator that only acts non-trivially on the span of $|+\rangle$, $|-\rangle$ and $|\circlearrowleft\rangle$. Therefore, any complex eigenvalues of F must come in conjugate pairs, and F can have at most three non-trivial eigenvectors. Using the observation that $(\mathsf{A}\tilde{\mathsf{G}})_z$ is real-valued, any (-1)-eigenspace of F would necessarily have even dimension. Thus, F must have either two or zero non-trivial eigenvectors.

Define $|f_+{}^{un}\rangle = \sqrt{2(4-3s)}|f_+\rangle$ and compute

$$\begin{aligned}(\mathsf{A}\tilde{\mathsf{G}})_z|f_+{}^{un}\rangle &= \left[-(2-3s)\frac{\sqrt{4-3s}}{2} - \imath\sqrt{3s}(4-3s)\right]|+\rangle \\ &+ \left[-\frac{\sqrt{4-3s}}{2}\sqrt{3}s + \imath\sqrt{s}(2-3s)\right]|-\rangle \\ &+ \left[\sqrt{3s(1-s)(4-3s)} - \imath\sqrt{1-s}(2-3s)\right]|\circlearrowleft\rangle \\ &= -\frac{1}{2}\Big((2-3s) + \imath\sqrt{3s}\sqrt{4-3s}\Big)|f_+{}^{un}\rangle \\ &= -e^{\imath 2\eta}|f_+{}^{un}\rangle.\end{aligned}$$

This shows that $|f_+\rangle$ is an eigenvector of $\mathsf{F} = (\mathsf{A}\tilde{\mathsf{G}})_z(\mathsf{A}\tilde{\mathsf{G}})_z$ with eigenvalue $(-e^{\imath 2\eta})^2 = \lambda$. It follows that the entrywise conjugate of $|f_+\rangle$, given by $|f_-\rangle$, must be an eigenvector of F with eigenvalue λ^{-1}. □

Lemma 1 shows that F is a rotation by 4η of a single two-dimensional space, spanned by $|f_+\rangle$ and $|f_-\rangle$. Therefore, F can be decomposed as the product of two one-dimensional reflections. We choose these reflections as follows.

Lemma 2 *Define*

$$|f_1\rangle = \frac{1}{\sqrt{4-3s}}\Big(\sqrt{s}|{-}\rangle - 2\sqrt{1-s}|\circlearrowleft\rangle\Big), \tag{3.15}$$

$$|f_2\rangle = \sin(2\eta)|{+}\rangle + \cos(2\eta)|f_1\rangle, \tag{3.16}$$

and let

$$\mathsf{F}_1 = \mathsf{I} - 2|f_1\rangle\langle f_1|, \tag{3.17}$$

$$\mathsf{F}_2 = \mathsf{I} - 2|f_2\rangle\langle f_2|. \tag{3.18}$$

Then $\mathsf{F} = \mathsf{F}_1\mathsf{F}_2$.

Proof We know by Lemma 1 that F is a real-valued rotation of a single two-dimensional space. Therefore, we can write F as the product of two one-dimensional reflections of real-valued vectors. To obtain this decomposition, we could choose any two real-valued unit vectors $|f_1\rangle$ and $|f_2\rangle$ in the rotational space of F for which $\langle f_1|f_2\rangle = \cos(2\eta)$.

From Lemma 1, we have formulas for the eigenvectors $|f_+\rangle$ and $|f_-\rangle$ of F. Thus, we set $|f_1\rangle = \frac{\imath}{\sqrt{2}}(|f_+\rangle - |f_-\rangle)$, which is a real-valued vector in the rotational space. Furthermore, $|{+}\rangle = \frac{1}{\sqrt{2}}(|f_+\rangle + |f_-\rangle)$ is real-valued and orthogonal to $|f_1\rangle$. By defining $|f_2\rangle = \sin(2\eta)|{+}\rangle + \cos(2\eta)|f_1\rangle$, we obtain a second vector with the desired properties. □

3.4 Structure of W

We give exact formulas for the eigenvectors and eigenphases of W, as well as a decomposition of the $(N+1)$-dimensional domain into subspaces that are invariant under W. These properties are required for the precise characterisation of WF_1 and WF in later sections.

3.4.1 Spectrum of W

Recall that $|\circlearrowleft\rangle$ is trivially a $(+1)$-eigenvector of W. The remaining N eigenvectors of W can be indexed by k and l as follows.

For $0 \le k < n_r/2$, let $\tilde{k} = \frac{2\pi k}{n_r}$, and for $0 \le l < n_c/2$, let $\tilde{l} = \frac{2\pi l}{n_c}$. Let $\epsilon_k = \operatorname{sign}(\cos\tilde{k})$ and $\epsilon_l = \operatorname{sign}(\cos\tilde{l})$. Define the sign of zero to be $+1$. This case occurs when n_r or n_c is divisible by 4, since $\cos(\tilde{k}) = 0$ when $k = n_r/4$ and $\cos(\tilde{l}) = 0$ when $l = n_c/4$.

Define

$$p_{kl} = \sqrt{1 - \cos^2\tilde{k}\cos^2\tilde{l}}, \tag{3.19}$$

$$\theta_{kl} = \epsilon_k\epsilon_l \arccos(1 - 2p_{kl}^2), \tag{3.20}$$

and

$$r_{kl}^{\pm} = \sqrt{2\left(1 \pm \frac{\sin\tilde{k}\cos\tilde{l}}{p_{kl}}\right)} = \sqrt{1 + \frac{\sin\tilde{l}}{p_{kl}}} \pm \epsilon_l\sqrt{1 - \frac{\sin\tilde{l}}{p_{kl}}},$$

$$c_{kl}^{\pm} = \sqrt{2\left(1 \pm \frac{\cos\tilde{k}\sin\tilde{l}}{p_{kl}}\right)} = \sqrt{1 + \frac{\sin\tilde{k}}{p_{kl}}} \pm \epsilon_k\sqrt{1 - \frac{\sin\tilde{k}}{p_{kl}}}.$$

If $k = l = 0$, then $p_{00} = 0$ and the division by zero is ill-defined. In this case, we define

$$r_{00}^{+} = r_{00}^{-} = c_{00}^{+} = c_{00}^{-} = \sqrt{2}.$$

For each $0 \le k < n_r/2$ and $0 \le l < n_c/2$, there is an eigenvector $|w_{kl}\rangle$ of W with eigenvalue $e^{\imath\theta_{kl}}$. This $|w_{kl}\rangle$ is the product state

$$|w_{kl}\rangle = |u_{kl}\rangle \otimes |v_{kl}\rangle, \tag{3.21}$$

where the factors are given by the normalized states

$$|u_{kl}\rangle = \sqrt{2}|\phi_r^k\rangle \circ (|1_{n_r/2}\rangle \otimes |r_{kl}\rangle), \qquad |r_{kl}\rangle = \frac{1}{2}\begin{bmatrix} r_{kl}^- \\ r_{kl}^+ \end{bmatrix},$$

$$|v_{kl}\rangle = \sqrt{2}|\phi_c^l\rangle \circ (|1_{n_c/2}\rangle \otimes |c_{kl}\rangle), \qquad |c_{kl}\rangle = \frac{1}{2}\begin{bmatrix} c_{kl}^- \\ c_{kl}^+ \end{bmatrix}.$$

Here, $\circ$ denotes the Hadamard product and $|1_n\rangle$ is the all-ones vector of dimension n. The Fourier states are $|\phi_r^k\rangle = \frac{1}{\sqrt{n_r}}\sum_{i=0}^{n_r-1}\omega_{n_r}^{ik}|i\rangle$ and $|\phi_c^k\rangle = \frac{1}{\sqrt{n_c}}\sum_{i=0}^{n_c-1}\omega_{n_c}^{ik}|i\rangle$, where $\omega_n = e^{2\pi\imath/n}$ denotes the nth root of unity.

Let

$$|r^1_{kl}\rangle = \mathsf{XZ}|r_{kl}\rangle,$$
$$|c^1_{kl}\rangle = \mathsf{XZ}|c_{kl}\rangle.$$

We let $|u^1_{kl}\rangle$ and $|v^1_{kl}\rangle$ be defined similarly to $|u_{kl}\rangle$ and $|v_{kl}\rangle$ above, but replacing $|r_{kl}\rangle$ and $|c_{kl}\rangle$ with $|r^1_{kl}\rangle$ and $|c^1_{kl}\rangle$, respectively. Then replacing $|u_{kl}\rangle$ and $|v_{kl}\rangle$ with $|u^1_{kl}\rangle$ and $|v^1_{kl}\rangle$ in the definition of $|w_{kl}\rangle$ yields an eigenvector with eigenphase $-\theta_{kl}$, and replacing either one of the two yields an eigenvector with eigenvalue $+1$. We denote $|u^0_{kl}\rangle = |u_{kl}\rangle$ and $|v^0_{kl}\rangle = |v_{kl}\rangle$. Let $|w^B_{kl}\rangle$ be defined accordingly for $B \in \{00, 01, 10, 11\}$.

By this definition, the N eigenvectors $\{|w^B_{kl}\rangle\}$ of W constitute an orthonormal basis for the grid, where the eigenvector $|w^B_{kl}\rangle$ has eigenphase $\theta_{kl}, 0, 0, -\theta_{kl}$ for $B = 00, 01, 10, 11$.

Lemma 3 *Both* $|\pi\rangle$ *and* $|\pi_z\rangle$ *are* $(+1)$*-eigenvectors of* W.

Proof Based on equation (3.20), we note that $\theta_{00} = 0$. The lemma follows from the observation that

$$|\pi\rangle = |w^{00}_{00}\rangle,$$
$$|\pi_z\rangle = \frac{1}{2}\Big(|w^{00}_{00}\rangle + |w^{01}_{00}\rangle + |w^{10}_{00}\rangle - |w^{11}_{00}\rangle\Big).$$

□

3.4.2 Invariant subspaces

We partition the domain of W into subspaces W_{kl}, which we define as follows. These subspaces will be useful later in our analysis, as they allow us to consider action of W on each subspace separately. This in turn allows for precise analysis of our walk's behaviour.

The number of subspaces depends on the parity of $\frac{n_c}{2}$. If $\frac{n_c}{2}$ is odd, there are $\frac{(n_r+2)(n_c-2)}{8} + 1$ invariant subspaces, and if $\frac{n_c}{2}$ is even, there are $\frac{(n_r+2)n_c}{8} + \lfloor \frac{n_r}{4} \rfloor - 1$ invariant subspaces.

Each W_{kl} is spanned by a set of eigenvectors of W with eigenphase θ_{kl}. The subspaces are defined such that each subspace is invariant under the action of W, in the sense that any vector in W_{kl} will remain in W_{kl} when acted on by W. Thus, our partition of the domain of W yields subspaces with the convenient property that W applies the same eigenphase θ_{kl} to all vectors in a given W_{kl}, and that applying W does not change which vectors are in which subspaces. The projection onto subspace W_{kl} is denoted Π_{kl}.

First, observe that $\theta_{00} = 0$, so the eigenvectors $|w_{00}^{00}\rangle$ and $|w_{00}^{11}\rangle$ are both (+1)-eigenvectors. The (+1)-eigenspace of W therefore has dimension $\frac{N}{2}+2+1$, where the last dimension comes from the selfloop state. We denote the (+1)-eigenspace as W_{00} and its associated projection as Π_{00}.

For any $0 \le k < n_r/2$ and $0 \le l < n_c/2$, define

$$k' = \begin{cases} \frac{n_r}{2} - k & \text{if } 0 < k < \frac{n_r}{2} \\ 0 & \text{if } k = 0 \end{cases} \qquad \text{and} \qquad l' = \begin{cases} \frac{n_c}{2} - l & \text{if } 0 < l < \frac{n_c}{2} \\ 0 & \text{if } l = 0. \end{cases}$$

Note that $k = k'$ exactly when $k = 0$ or $k = \frac{n_r}{4}$, and similarly for $l = l'$.

For $0 < k < \frac{n_r}{2}, k \neq \frac{n_r}{4}$ and $0 < l < \frac{n_c}{4}$, define

$$\mathsf{W}_{kl} = \operatorname{span}\Big\{|w_{kl}^{00}\rangle, |w_{k'l'}^{00}\rangle, |w_{k'l}^{11}\rangle, |w_{kl'}^{11}\rangle\Big\}.$$

Each of these subspaces has an associated eigenphase $\theta_{kl} \neq 0, \pi$. When n_r or n_c is a multiple of 4, W also has a (−1)-eigenspace. In this case, there are subspaces with eigenphase π given by

$$\mathsf{W}_{kl} = \operatorname{span}\Big\{|w_{kl}^{00}\rangle, |w_{k'l'}^{00}\rangle, |w_{k'l}^{11}\rangle, |w_{kl'}^{11}\rangle\Big\}.$$

for $k = \frac{n_r}{4}$, $0 < l < \frac{n_c}{4}$ and $l = \frac{n_c}{4}$, $0 < k \le \frac{n_r}{4}$. These four vectors will be distinct unless both $k = \frac{n_r}{4}$ and $l = \frac{n_c}{4}$, in which case $\dim(\mathsf{W}_{kl}) = 2$.

For $k = 0$ and $0 < l < \frac{n_c}{2}$, the corresponding invariant subspace is

$$\mathsf{W}_{kl} = \operatorname{span}\Big\{|w_{kl}^{00}\rangle, |w_{kl'}^{11}\rangle\Big\},$$

and similarly for $l = 0$ and $0 < k < \frac{n_r}{2}$,

$$\mathsf{W}_{kl} = \operatorname{span}\Big\{|w_{kl}^{00}\rangle, |w_{k'l}^{11}\rangle\Big\}.$$

Partitioning the domain in this way allows us to closely analyse the behaviour of $|+\rangle$ and $|-\rangle$ under the action of W. The following results will be used in our analysis of $\mathsf{U}_z = \mathsf{WF}$.

Lemma 4 *The following statements hold.*

$$\Pi_{kl}|+\rangle \perp \Pi_{kl}|-\rangle \qquad \text{for all subspaces } \mathsf{W}_{kl} \tag{3.22}$$

$$\|\Pi_{kl}|+\rangle\|^2 = \frac{2}{3}\frac{\dim(\mathsf{W}_{kl})}{N} \qquad \text{for all } k,l \text{ not both } 0 \tag{3.23}$$

$$\|\Pi_{kl}|-\rangle\|^2 = 2\frac{\dim(\mathsf{W}_{kl})}{N} \qquad \text{for all } k,l \text{ not both } 0 \tag{3.24}$$

$$\|\Pi_{00}|+\rangle\|^2 = \frac{2}{3}\frac{N+2}{N} \tag{3.25}$$

$$\|\Pi_{00}|-\rangle\|^2 = \frac{4}{N}. \tag{3.26}$$

Proof These results are obtained using direct calculations based on the definitions given in this section. The full proof is given in Appendix 5. □

3.5 Reduction to the slowest subspace

Our memoryless walk, given in Theorem 2, achieves a success probability asymptotically close to 1. This is possible because our choice of s reduces the walk asymptotically to a rotation in a single two-dimensional subspace. As we show, this subspace is exactly the slowest rotational subspace of the applied walk operator. We prove this by giving a tight description of the smallest positive eigenvalue of the walk operator and its associated eigenvector. We also show that the two-dimensional rotation induced by the walk maps the initial state to the desired state $|\circlearrowleft\rangle$.

We consider two walk operators. The first consists of the real operator W composed with the one-dimensional reflection F_1. The second consists of W composed with the two-dimensional rotation F. We discuss the first operator in this section, and then use the results to derive properties of the second operator in Section 3.6.

First, we prove three key lemmas about the slowest rotational subspace of WF_1. Lemma 6 gives a tight bound on the smallest positive eigenphase of WF_1, and Lemmas 8 and 9 characterize the asymptotic behaviour of its associated eigenvector. These three lemmas show that the action of WF_1 on $|\pi_z\rangle$ can be asymptotically reduced to a rotation in the slowest rotational subspace. These results are the basis of the methods used in Section 3.6 to prove our main result.

The intermediate operator WF_1 is fundamentally a tool for analysis. However, it is interesting to note that WF_1 can be used directly to find the marked state. The proof of the corollary below follows from a similar argument to our proof of the main theorem.

Corollary 1 *Fix $s = 1 - \frac{1}{N+1}$ and suppose $n_r = n_c$. Then there exists a constant $c > 0$ such that after $c\sqrt{N \log N}$ applications of WF_1 to $|\pi_z\rangle$, measuring the state will produce $|\circlearrowleft\rangle$ with probability $1 - e(N)$, where $e(N) \in O(\frac{1}{\log N})$.*

Comparing Corollary 1 with Theorem 2 shows that most of the finding behaviour of the memoryless walk comes from the first reflection F_1. In Section 3.6, we show that composing WF_1 with the second reflection F_2 only changes the behaviour of the walk slightly, leading to our proof of Theorem 2.

Our proofs of both Corollary 1 and our main theorem rely on the following observation.

Lemma 5 *The (unnormalized) vector*

$$|\mathsf{U}_0\rangle = |\pi_z\rangle - \sqrt{\frac{s}{(1-s)N}}|\circlearrowleft\rangle \tag{3.27}$$

is a $(+1)$-eigenvector for each of W, F_1, and F_2.

Proof Both $|\pi_z\rangle$ and $|\circlearrowleft\rangle$ are $(+1)$-eigenvectors of W, so $|\mathsf{U}_0\rangle$ is a $(+1)$-eigenvector of W. To show $|\mathsf{U}_0\rangle$ is a $(+1)$-eigenvector of F_1, we compute

$$\langle \mathsf{U}_0|f_1\rangle = \frac{2\sqrt{s}}{\sqrt{N(4-3s)}} + \frac{\sqrt{s}}{\sqrt{4-3s}}\langle \pi_z|-\rangle = 0.$$

Finally, $|+\rangle$ is orthogonal to both $|\pi_z\rangle$ and $|\circlearrowleft\rangle$, so $|\mathsf{U}_0\rangle$ is orthogonal to $|f_2\rangle$. Therefore, $|\mathsf{U}_0\rangle$ is also a $(+1)$-eigenvector of F_2. □

For the rest of the chapter, we assume a square grid, so $n_r = n_c = \sqrt{N}$. We also fix $s = 1 - \frac{1}{N+1}$. This choice of selfloop weight means $|\mathsf{U}_0\rangle = |\pi_z\rangle - |\circlearrowleft\rangle$, so we can decompose the initial state $|\pi_z\rangle$ as

$$|\pi_z\rangle = \frac{1}{2}|\mathsf{U}_0\rangle + \frac{1}{2}\Big(|\pi_z\rangle + |\circlearrowleft\rangle\Big). \tag{3.28}$$

We show in Section 3.5.2 that for our chosen s, $|\pi_z\rangle + |\circlearrowleft\rangle$ lies asymptotically in the slowest rotational subspace of WF_1. Therefore, WF_1 can be used to apply a negative phase to this portion of the initial state. This rotates the state $|\pi_z\rangle$ to the state $\frac{1}{2}|\mathsf{U}_0\rangle - \frac{1}{2}\big(|\pi_z\rangle + |\circlearrowleft\rangle\big) = -|\circlearrowleft\rangle$, as we make precise in our proof of Corollary 1.

When $s = 1 - \frac{1}{N+1}$, note that the vector $|f_1\rangle$ has the form

$$|f_1\rangle = \sqrt{\frac{N}{N+4}}|-\rangle - \frac{2}{\sqrt{N+4}}|\circlearrowleft\rangle. \tag{3.29}$$

3.5.1 Smallest eigenphase of WF_1

We choose the operator WF_1 as our intermediate step in the analysis of WF because it is the composition of a well-characterized real operator with a one-dimensional reflection. This allows us to apply results from the literature about operators of this type. Here, we show how these results can be used to obtain a tight bound on the smallest positive eigenphase of WF_1.

An overview of the applied results is given in Appendix 5.

Lemma 6 *The smallest positive eigenphase φ_1 of WF_1 satisfies $\varphi_1 \in \Theta(\frac{1}{\sqrt{N\log N}})$.*

Proof Observe that W is a real-valued operator and that F_1 is a reflection of a single real-valued vector, $|f_1\rangle$. Therefore, we can apply the results discussed in Appendix 5, with the correspondence $\mathsf{T} = \mathsf{W}$ and $|s\rangle = |f_1\rangle$.

First, we recall from Section 3.4.1 that the smallest positive eigenvalue of W is given by

$$\theta_{10} = \operatorname{acos}\left(2\cos^2\left(\frac{2\pi}{\sqrt{N}}\right) - 1\right) = \frac{4\pi}{\sqrt{N}}. \tag{3.30}$$

We know by Lemma 4 and the decomposition in equation (3.29) that $|f_1\rangle$ satisfies

$$\begin{aligned} \|\Pi_{00}|f_1\rangle\|^2 &= \frac{8}{N+4}, \\ \|\Pi_{kl}|f_1\rangle\|^2 &= \frac{2\dim(\mathsf{W}_{kl})}{N+4} \qquad\qquad \text{for all } k,l \text{ not both } 0. \end{aligned}$$

In particular, $|f_1\rangle$ overlaps all eigenspaces of W, including the eigenspace with eigenphase θ_{10}. Therefore, by Theorem 7, $\varphi_1 < \theta_{10} = \frac{4\pi}{\sqrt{N}}$.

By definition, φ_1 is an eigenphase of the operator WF_1, so by Lemma 19, φ_1 must also satisfy the constraint in equation (3), with $\alpha = \varphi_1$. This constraint depends on the projection of $|f_1\rangle$ onto each of the two-dimensional rotational subspaces of W. We rewrite the constraint in equation (3) as a sum over the invariant subspaces of W, so that it becomes

$$\|\Pi_{00}|f_1\rangle\|^2 \cot\Big(\frac{\varphi_1}{2}\Big) + \sum_{kl\neq 00} \|\Pi_{kl}|f_1\rangle\|^2 \cot\Big(\frac{\varphi_1 - \theta_{kl}}{2}\Big) = 0. \tag{3.31}$$

Here, the sum is taken over all invariant subspaces of W except the $(+1)$-eigenspace, W_{00}. Note that the (-1)-eigenspace is included this sum, while it is written as a separate term in Lemma 19. Because $\varphi_1 > 0$ and $\varphi_1 \in o(1)$, we have

$$\|\Pi_{00}|f_1\rangle\|^2 \cot\Big(\frac{\varphi_1}{2}\Big) \in \Theta\Big(\frac{1}{\varphi_1 N}\Big).$$

Therefore, for equation (3.31) to hold, it must be the case that

$$\sum_{kl\neq 00} \|\Pi_{kl}|f_1\rangle\|^2 \cot\Big(\frac{\theta_{kl} - \varphi_1}{2}\Big) \in \Theta\Big(\frac{1}{\varphi_1 N}\Big). \tag{3.32}$$

We argue that there cannot be a solution $\varphi_1 \in \Theta(\frac{1}{\sqrt{N}})$. By Fact 1, we know that for such a φ_1, the sum in equation (3.32) has order $\Omega(\frac{\log N}{\sqrt{N}})$. Therefore, it must be the case that $\varphi_1 \in o(\frac{1}{\sqrt{N}})$.

By Fact 1, we also know that if $\varphi_1 \in o(\frac{1}{\sqrt{N}})$, then the sum in equation (3.32) has order $\Theta(\varphi_1 \log N)$. Due to the requirement $\varphi_1 \log N \in \Theta(\frac{1}{\varphi_1 N})$, the only possible solution is $\varphi_1 \in \Theta(\frac{1}{\sqrt{N \log N}})$, as stated. □

3.5.2 Slowest eigenvector of WF_1

In this section, we use a constraint-solving approach to analyse the eigenvector of WF_1 associated with eigenphase φ_1. By determining its asymptotic behaviour, we show that the $|\pi_z\rangle + |\circlearrowleft\rangle$ component of equation (3.28) lies in the span of this eigenvector and its conjugate. This shows that WF_1 can be applied to rotate the initial state $|\pi_z\rangle$ to the target state $|\circlearrowleft\rangle$.

Define $|{-}\rangle^\perp = |{-}\rangle + \frac{2}{\sqrt{N}}|\pi_z\rangle$ to be the (unnormalized) component of $|{-}\rangle$ that is orthogonal to $|\pi_z\rangle$. Note that by Lemma 4, this vector is orthogonal to (+1)-eigenspace of W.

Let $|\zeta\rangle$ be an unnormalized eigenvector of WF_1 with eigenphase $\alpha \neq 0, \pi$ and $\langle\zeta|\pi_z\rangle \neq 0$, scaled such that $\langle\zeta|\pi_z\rangle = \frac{1}{2}$. Because $|\zeta\rangle$ is perpendicular to $|\mathsf{U}_0\rangle$, this implies $\langle\zeta|\circlearrowleft\rangle = \frac{1}{2}$. We decompose $|\zeta\rangle$ as

$$|\zeta\rangle = a|{-}\rangle^\perp + \frac{1}{2}|\pi_z\rangle + \frac{1}{2}|\circlearrowleft\rangle + |\psi\rangle, \tag{3.33}$$

where $|\psi\rangle$ is an unnormalized vector orthogonal to $|{-}\rangle^\perp$. By analysing the asymptotic behaviour of a and $|\psi\rangle$, we show that the real part of $|\zeta\rangle$ tends to $\frac{1}{2}(|\pi_z\rangle + |\circlearrowleft\rangle)$ when $\alpha = \varphi_1$.

Note that any eigenvector of W with eigenphase θ_{kl} that is orthogonal to $|f_1\rangle$ is also an eigenvector of WF_1 with eigenphase θ_{kl}. Therefore, $\Pi_{kl}|\psi\rangle$ is some scalar multiple of $\Pi_{kl}|{-}\rangle^\perp$ for each k, l. We determine this scalar factor in Lemma 7.

We further decompose both $|{-}\rangle^\perp$ and $|\psi\rangle$ into the invariant subspaces of W. Both $|{-}\rangle^\perp$ and $|\psi\rangle$ are orthogonal to the (+1)-eigenspace W_{00}, so we write the decomposition as

$$|{-}\rangle^\perp = \sum_{kl \neq 00} m_{kl}|{-}_{kl}\rangle, \tag{3.34}$$

$$|\psi\rangle = \sum_{kl \neq 00} |\psi_{kl}\rangle, \tag{3.35}$$

where the vectors $|{-}_{kl}\rangle$ are normalized for all k, l. We know by Lemma 4 that $m_{kl} = \sqrt{\frac{2\dim(\mathsf{W}_{kl})}{N}}$. The vectors $|\psi_{kl}\rangle$ in the decomposition of $|\psi\rangle$ are unnormalized.

Lemma 7 *The following equations must be satisfied.*

$$\frac{8a(N-4)}{\sqrt{N}(N+4)} - \frac{16}{N+4} = e^{\imath\alpha} - 1 \tag{3.36}$$

$$\langle -_{kl}|\psi_{kl}\rangle = m_{kl}\left[a - \frac{\sqrt{N}}{4}(e^{\imath\alpha}-1)\left(\frac{1}{1-e^{\imath(\alpha-\theta_{kl})}}\right)\right] \quad \text{for all } k,l \text{ not both } 0. \tag{3.37}$$

Proof By definition, $|\zeta\rangle$ is an eigenvector of WF_1 with eigenphase α. We obtain the lemma by expanding the equation $\mathsf{WF}_1|\zeta\rangle = e^{\imath\alpha}|\zeta\rangle$ and solving for constraints.

Observe that using equation (3.29),

$$\langle f_1|\zeta\rangle = \frac{1}{\sqrt{N+4}}\left(\frac{a(N-4)}{\sqrt{N}} - 2\right).$$

Using this property, we compute

$$\begin{aligned}\mathsf{WF}_1|\zeta\rangle &= \mathsf{W}|\zeta\rangle - 2\mathsf{W}\langle f_1|\zeta\rangle|f_1\rangle \\ &= \gamma_-\mathsf{W}|-\rangle^{\perp} + \gamma_*(|\pi_z\rangle + |\circlearrowleft\rangle) + \mathsf{W}|\psi\rangle,\end{aligned}$$

where

$$\begin{aligned}\gamma_- &= a - \frac{2}{N+4}\left(a(N-4) - 2\sqrt{N}\right), \\ \gamma_* &= \frac{1}{2} + \frac{4}{\sqrt{N}(N+4)}\left(a(N-4) - 2\sqrt{N}\right).\end{aligned}$$

Setting $\mathsf{WF}_1|\zeta\rangle = e^{\imath\alpha}|\zeta\rangle$ and comparing coefficients on $|\pi_z\rangle$, we get

$$\gamma_* = \frac{1}{2}e^{\imath\alpha},$$

which can be expanded to give equation (3.36).

To get equation (3.37), we first solve $\mathsf{WF}_1|\zeta\rangle = e^{\imath\alpha}|\zeta\rangle$ on the subspace W_{kl} to get

$$\langle -_{kl}|\psi_{kl}\rangle = m_{kl}\left(\frac{ae^{\imath\alpha} - \gamma_- e^{\imath\theta_{kl}}}{e^{\imath\theta_{kl}} - e^{\imath\alpha}}\right). \tag{3.38}$$

Next, we use equation (3.36) to rewrite γ_- as

$$\gamma_- = a - \frac{\sqrt{N}}{4}(e^{\imath\alpha} - 1).$$

Substituting this expression for γ_- into equation (3.38) produces equation (3.37). □

Now, fix $|\zeta\rangle$ to be the eigenvector with eigenphase φ_1. By Lemma 19, we know that $\langle\zeta|\pi_z\rangle \neq 0$, so the constraints given in Lemma 7 apply. These constraints, together with the bound on φ_1 from Lemma 6, define asymptotic bounds on a and the real part of $|\psi\rangle$. We use this to show that as N increases, the real part of $|\zeta\rangle$ converges to $\frac{1}{2}|\pi_z\rangle + |\circlearrowleft\rangle$.

Let $|\overline{\zeta}\rangle$ denote the entrywise conjugate of $|\zeta\rangle$. Then $|\zeta\rangle$ and $|\overline{\zeta}\rangle$ span the slowest rotational subspace of WF_1. In this way, the following lemmas provide a close description of the spanning eigenvectors for the slowest rotational subspace of WF_1.

Lemma 8 *Let $\alpha = \varphi_1$. Then $|a| \in \Theta(\frac{1}{\sqrt{\log N}})$.*

Proof By Lemma 6, we know that $|e^{\imath\varphi_1} - 1| \in \Theta(\frac{1}{\sqrt{N\log N}})$. Applying this to equation (3.36) produces the stated bound. □

Lemma 9 *Let $\alpha = \varphi_1$. Let $|\psi\rangle = \Re(|\psi\rangle) + \imath\Im(|\psi\rangle)$, where both $\Re(|\psi\rangle)$ and $\Im(|\psi\rangle)$ are vectors with real entries. Then $\|\Re(|\psi\rangle)\| \in O(\frac{1}{\sqrt{\log N}})$.*

Proof From equation (3.37), we know that

$$\begin{aligned}|\psi\rangle &= \sum_{kl\neq 00} m_{kl}\left[a - \frac{\sqrt{N}}{4}(e^{\imath\alpha} - 1)\left(\frac{1}{1 - e^{\imath(\alpha-\theta_{kl})}}\right)\right]|{-}_{kl}\rangle \\ &= a|{-}\rangle^{\perp} + \rho|v\rangle,\end{aligned}$$

where we define

$$\begin{aligned}\rho &= -\frac{\sqrt{N}}{4}(e^{\imath\alpha} - 1), \\ |v\rangle &= \sum_{kl\neq 00} m_{kl}\left(\frac{1}{1 - e^{\imath(\alpha-\theta_{kl})}}\right)|{-}_{kl}\rangle.\end{aligned}$$

We know from Lemma 8 that $\| a|{-}\rangle^{\perp}\| \in \Theta(\frac{1}{\sqrt{\log N}})$. Thus, it remains to consider $\rho|v\rangle$.

Examining ρ shows that for $\alpha \in \Theta(\frac{1}{\sqrt{N \log N}})$,

$$|\Re(\rho)| \in \Theta\left(\frac{1}{\sqrt{N}\log N}\right), \qquad |\Im(\rho)| \in \Theta\left(\frac{1}{\sqrt{\log N}}\right).$$

Therefore, we can prove the lemma by showing that $\|\Re(|v\rangle)\| \in O(\sqrt{N \log N})$ and $\|\Im(|v\rangle)\| \in O(1)$. Observe that for all k, l,

$$\Re\left(\frac{1}{1-e^{\imath(\alpha-\theta_{kl})}}\right) = \frac{1}{2}, \qquad \Im\left(\frac{1}{1-e^{\imath(\alpha-\theta_{kl})}}\right) = -\frac{1}{2}\cot\left(\frac{\theta_{kl}-\alpha}{2}\right).$$

Using this property, we split the coefficients of $|v\rangle$ into their real and imaginary parts, giving

$$\begin{aligned}|v\rangle &= \frac{1}{2}\sum_{kl\neq 00} m_{kl}|{-}_{kl}\rangle - \frac{\imath}{2}\sum_{kl\neq 00} m_{kl}\cot\left(\frac{\theta_{kl}-\alpha}{2}\right)|{-}_{kl}\rangle \\ &= \frac{1}{2}|{-}\rangle^{\perp} - \frac{\imath}{2}|v'\rangle,\end{aligned}$$

where

$$|v'\rangle = \sum_{kl\neq 00} m_{kl}\cot\left(\frac{\theta_{kl}-\alpha}{2}\right)|{-}_{kl}\rangle.$$

Note that $|{-}\rangle^{\perp}$ is real-valued and has norm $\Theta(1)$.

We now bound the real and imaginary parts of $|v'\rangle$. Recall that for each subspace W_{kl} with eigenphase $0 < \theta_{kl} < \pi$, there is a corresponding subspace with eigenphase $-\theta_{kl}$, which we denote $\overline{\mathsf{W}_{kl}}$. Because $|{-}\rangle^{\perp}$ is real-valued, it must be the case that the normalized projection of $|{-}\rangle^{\perp}$ onto $\overline{\mathsf{W}_{kl}}$ is $|\overline{{-}_{kl}}\rangle$, the entrywise conjugate of $|{-}_{kl}\rangle$. Using this property, we decompose $|v'\rangle$ as

$$\begin{aligned}|v'\rangle &= \sum_{0<\theta_{kl}<\pi} m_{kl}\left[\cot\left(\frac{\theta_{kl}-\alpha}{2}\right)|{-}_{kl}\rangle - \cot\left(\frac{\theta_{kl}+\alpha}{2}\right)|\overline{{-}_{kl}}\rangle\right] \\ &\quad + \sum_{\theta_{kl}=\pi} m_{kl}\cot\left(\frac{\theta_{kl}-\alpha}{2}\right)|{-}_{kl}\rangle \\ &= |v_1\rangle + |v_2\rangle + |v_3\rangle,\end{aligned}$$

where

$$|v_1\rangle = \sum_{0<\theta_{kl}<\pi} m_{kl}\left[\cot\left(\frac{\theta_{kl}+\alpha}{2}\right)|-_{kl}\rangle - \cot\left(\frac{\theta_{kl}+\alpha}{2}\right)\overline{|-_{kl}\rangle}\right],$$
$$|v_2\rangle = \sum_{0<\theta_{kl}<\pi} m_{kl}\left[\cot\left(\frac{\theta_{kl}-\alpha}{2}\right) - \cot\left(\frac{\theta_{kl}+\alpha}{2}\right)\right]|-_{kl}\rangle,$$
$$|v_3\rangle = \sum_{\theta_{kl}=\pi} m_{kl}\cot\left(\frac{\theta_{kl}-\alpha}{2}\right)|-_{kl}\rangle.$$

Note that the sums are taken over the invariant subspaces of W whose eigenphases lie in the indicated range. We bound the norms of these three components individually. First, observe that

$$|v_3\rangle = \tan\left(\frac{\alpha}{2}\right)\sum_{\theta_{kl}=\pi} m_{kl}|-_{kl}\rangle,$$

where $\sum_{\theta_{kl}=\pi} m_{kl}|-_{kl}\rangle$ is the projection of $|-\rangle$ onto the (-1)-eigenspace of W. Therefore, $|v_3\rangle$ must be entirely real-valued, with norm $\||v_3\rangle\| \in O(\frac{1}{\sqrt{N\log N}})$ by Lemma 6.

The vector $|v_1\rangle$ is entirely imaginary-valued, with norm

$$\begin{aligned}
\||v_1\rangle\|^2 &= \sum_{0<\theta_{kl}<\pi} {m_{kl}}^2\left[\cot\left(\frac{\theta_{kl}+\alpha}{2}\right)^2 + \cot\left(\frac{\theta_{kl}+\alpha}{2}\right)^2\right] \\
&\leq \frac{16}{N}\sum_{0<\theta_{kl}<\pi}\cot\left(\frac{\theta_{kl}+\alpha}{2}\right)^2 \\
&= \frac{16}{N}\sum_{0<\theta_{kl}<\pi}\left(\frac{\cot(\frac{\theta_{kl}}{2})\cot(\frac{\alpha}{2})-1}{\cot(\frac{\alpha}{2})+\cot(\frac{\theta_{kl}}{2})}\right)^2 \\
&\leq \frac{16}{N}\cot^2\left(\frac{\alpha}{2}\right)\sum_{0<\theta_{kl}<\pi}\left(\frac{\cot(\frac{\theta_{kl}}{2})}{\cot(\frac{\alpha}{2})+\cot(\frac{\theta_{kl}}{2})}\right)^2.
\end{aligned}$$

There are $O(N)$ terms in the final sum, each of which is at most 1, so $\||v_1\rangle\|^2 \in O(N\log N)$.

Finally, the vector $|v_2\rangle$ has both real and imaginary parts, and has norm

$$\begin{aligned}\||v_2\rangle\|^2 &= \sum_{0<\theta_{kl}<\pi} m_{kl}{}^2\left[\cot\Big(\frac{\theta_{kl}+\alpha}{2}\Big)-\cot\Big(\frac{\theta_{kl}-\alpha}{2}\Big)\right]^2\\ &= \frac{2}{N}\sum_{0<\theta_{kl}<\pi} \dim(\mathsf{W}_{kl})\left[\cot\Big(\frac{\theta_{kl}+\alpha}{2}\Big)-\cot\Big(\frac{\theta_{kl}-\alpha}{2}\Big)\right]^2\\ &= \frac{1}{N}\sum_{kl\neq 00} \dim(\mathsf{W}_{kl})\left[\cot\Big(\frac{\theta_{kl}+\alpha}{2}\Big)-\cot\Big(\frac{\theta_{kl}-\alpha}{2}\Big)\right]^2.\end{aligned}$$

By Fact 2, this implies $\||v_2\rangle\|^2 \in O(\frac{1}{\log N})$.

Combining the bounds on $|v_1\rangle$, $|v_2\rangle$ and $|v_3\rangle$, we bound the norm of the real and imaginary parts of $|v'\rangle$. Thus,

$$\begin{aligned}\|\Re(|v'\rangle)\| &\le \||v_3\rangle\| + \||v_2\rangle\| \in O\Big(\frac{1}{\sqrt{\log N}}\Big),\\ \|\Im(|v'\rangle)\| &\le \||v_1\rangle\| + \||v_2\rangle\| \in O(\sqrt{N\log N}).\end{aligned}$$

Because $|v\rangle = \frac{1}{2}|{-}\rangle^{\perp} - \frac{i}{2}|v'\rangle$, this shows in particular that $\|\Re(|v\rangle)\| \in O(\sqrt{N\log N})$ and $\|\Im(|v\rangle)\| \in O(1)$. We combine this with the bounds on ρ to obtain $\|\Re(|\psi\rangle)\| \in O(\frac{1}{\sqrt{\log N}})$ as stated. □

Lemmas 8 and 9 show that the real part of $|\zeta\rangle$ tends to $\frac{1}{2}(|\pi_z\rangle + |\circlearrowleft\rangle)$ as N increases. This implies that $|\pi_z\rangle + |\circlearrowleft\rangle$ lies asymptotically in the slowest rotational subspace of WF_1. We make this precise in the following lemma.

Lemma 10 *Let* Π_{φ_1} *denote the projection onto the slowest rotational subspace of* WF_1*, which is spanned by the eigenvectors with eigenphases* $\pm\varphi_1$*. Then*

$$\left\|\Pi_{\varphi_1}\Big(|\pi_z\rangle + |\circlearrowleft\rangle\Big)\right\| = \sqrt{2} - O\Big(\frac{1}{\log N}\Big). \tag{3.39}$$

Proof Let $\Re(a)$ denote the real part of a. Observe that

$$|\zeta\rangle + |\overline{\zeta}\rangle = |\pi_z\rangle + |\circlearrowleft\rangle + 2\Re(a)|{-}\rangle^{\perp} + 2\Re(|\psi\rangle).$$

By Lemma 8, we have $|\Re(a)| \in O(\frac{1}{\sqrt{\log N}})$, and by Lemma 9 we have $\|\Re(|\psi\rangle)\| \in O(\frac{1}{\sqrt{\log N}})$. Therefore,

$$\||\zeta\rangle + |\overline{\zeta}\rangle\| = \sqrt{2} + O\Big(\frac{1}{\log N}\Big).$$

We can also see that

$$\Big(\langle\zeta| + \langle\overline{\zeta}|\Big)\Big(|\pi_z\rangle + |\circlearrowleft\rangle\Big) = 2.$$

Noting that Π_{φ_1} denotes the projection onto span$\{|\zeta\rangle, |\overline{\zeta}\rangle\}$, this implies that

$$\Big\|\Pi_{\varphi_1}\Big(|\pi_z\rangle + |\circlearrowleft\rangle\Big)\Big\| \geq \sqrt{2} - O\Big(\frac{1}{\log N}\Big).$$

□

3.5.3 Proof of Corollary 1

Applying the operator WF_1 to the initial state $|\pi_z\rangle$ yields an optimal algorithm for finding $|\circlearrowleft\rangle$. The proof of this corollary follows from a similar argument to the proof of Theorem 2. We apply our characterization of the slowest rotational subspace of WF_1, given by Lemmas 6 and 10, to show the action of WF_1 on $|\pi_z\rangle$ is asymptotically restricted to this single subspace. The result is a Grover-like algorithm that rotates $|\pi_z\rangle$ to our desired state $|\circlearrowleft\rangle$.

Corollary 1 *Fix $s = 1 - \frac{1}{N+1}$ and suppose $n_r = n_c$. Then there exists a constant $c > 0$ such that after $c\sqrt{N \log N}$ applications of WF_1 to $|\pi_z\rangle$, measuring the state will produce $|\circlearrowleft\rangle$ with probability $1 - e(N)$, where $e(N) \in O(\frac{1}{\log N})$.*

Proof Recall the decomposition of $|\pi_z\rangle$ in equation (3.28). By Lemma 5, we know that $|\mathsf{U}_0\rangle$ is a $(+1)$-eigenvector of WF_1. Letting Π_{φ_1} denote the projection onto the slowest rotational subspace of WF_1, we decompose $|\pi_z\rangle + |\circlearrowleft\rangle$ as

$$|\pi_z\rangle + |\circlearrowleft\rangle = \Pi_{\varphi_1}\Big(|\pi_z\rangle + |\circlearrowleft\rangle\Big) + |\bot\rangle.$$

for some vector $|\bot\rangle$. By Lemma 10, we know that $\||\bot\rangle\| \in O(\frac{1}{\log N})$. We also know from Lemma 6 that the slowest rotational subspace has eigenphase $\varphi_1 \in \Theta(\frac{1}{\sqrt{N \log N}})$.

Therefore, there exists a constant c such that $c\sqrt{N\log N} = \lfloor\frac{\pi}{\varphi_1}\rfloor = k$. After k applications of WF_1 to $|\pi_z\rangle$, we get the state

$$\begin{aligned}(\mathsf{WF}_1)^k|\pi_z\rangle &= \frac{1}{2}\Big(|\pi_z\rangle - |\circlearrowleft\rangle\Big) - \frac{1}{2}\Pi_{\varphi_1}\Big(|\pi_z\rangle + |\circlearrowleft\rangle\Big) + |\rho\rangle \\ &= -|\circlearrowleft\rangle + \frac{1}{2}|\perp\rangle + |\rho\rangle.\end{aligned}$$

Here, $|\rho\rangle$ is some state that captures both the result of applying $(\mathsf{WF}_1)^k$ to $|\perp\rangle$ and the small error incurred by the rounding of $\frac{\pi}{\varphi_1}$, and has norm $\||\rho\rangle\| \in O(\frac{1}{\log N})$. Thus, measuring the state will produce $|\circlearrowleft\rangle$ with probability $1-e(N)$, where $e(N) \in O(\frac{1}{\log N})$ as stated. □

3.6 Finding with a memoryless walk

We now present the proof of our main theorem, which is stated as follows.

Theorem 2 (Main result) *Fix $s = 1 - \frac{1}{N+1}$ and suppose $n_r = n_c$. Then there exists a constant $c > 0$ such that after $c\sqrt{N\log N}$ applications of U to $|\pi\rangle$, measuring the state will produce $|\circlearrowleft\rangle$ with probability $1 - e(N)$, where $e(N) \in O(\frac{1}{\log N})$.*

Our proof uses the decomposition of $|\pi_z\rangle$ given in equation (3.28). We show that the state $\frac{1}{2}(|\pi_z\rangle + |\circlearrowleft\rangle)$ lies asymptotically in the slowest rotational subspace of WF. Therefore, WF can be used to rotate $|\pi_z\rangle$ to a state close to $-|\circlearrowleft\rangle$. Applying the change of basis cz yields the result as stated.

The proof is based on relating the slowest rotational subspaces of WF_1 and $\mathsf{WF} = \mathsf{WF}_1\mathsf{F}_2$. We continue to apply the results from Appendix 5, this time to analyse the real operator WF_1 composed with the one-dimensional reflection F_2. We show the slowest rotational subspaces of WF_1 and WF have the same asymptotic bound on the rotational angle, and that both asymptotically contain $\frac{1}{2}(|\pi_z\rangle+|\circlearrowleft\rangle)$. By using WF_1 as an intermediate operator, we are thus able to tightly characterize the slowest rotational subspace of a real operator W, composed with a two-dimensional rotation F.

3.6.1 Relationship with WF_1

We begin by relating the slowest rotational subspaces of WF and WF_1. Note that when $s = 1 - \frac{1}{N+1}$, the vector $|f_2\rangle$ has the form

$$|f_2\rangle = \frac{\sqrt{3N}\sqrt{N+4}}{2(N+1)}|+\rangle + \frac{2-N}{2(N+1)}|f_1\rangle. \tag{3.40}$$

Let the eigenphases of WF_1 different from $0, \pi$ be denoted by $\pm\varphi_k$ for $1 \le k \le m$, where $0 < |\varphi_1| \le |\varphi_2| \le \cdots \le |\varphi_m| < \pi$. Let the associated eigenvectors be $|A_k^\pm\rangle$. Both $|f_2\rangle$ and WF_1 are real-valued, so we can decompose $|f_2\rangle$ into the eigenbasis of WF_1 as

$$|f_2\rangle = g_0|A_0\rangle + \sum_{k=1}^{m} g_k\Big(|A_k^+\rangle + |A_k^-\rangle\Big) + g_{-1}|A_{-1}\rangle, \tag{3.41}$$

where $|A_0\rangle$ is a (+1)-eigenvector, $|A_{-1}\rangle$ is a (−1)-eigenvector, and all g_i are non-negative real numbers.

By Lemma 4, $\Pi_{kl}|+\rangle \perp \Pi_{kl}|f_1\rangle$ for each invariant subspace W_{kl} of W (including W_{00}). Therefore, the eigenvectors $\Pi_{kl}|+\rangle$ of W are also eigenvectors of WF_1 with the same eigenphases θ_{kl}.

Lemma 11 *The decomposition of $|f_2\rangle$ in equation (3.41) satisfies:*

$$g_0^2 = \frac{1}{2} + O\Big(\frac{1}{\sqrt{N}}\Big) \tag{3.42}$$

$$g_1^2 \in O\Big(\frac{1}{\log N}\Big). \tag{3.43}$$

Proof Recall that $0 < \varphi_1 < |\theta_{kl}|$ for all nonzero eigenphases θ_{kl} of W. Using the property that $\Pi_{kl}|+\rangle$ is an eigenvector of WF_1 with eigenvalue θ_{kl}, this implies that $\langle A_1^+|\Pi_{kl}|+\rangle = 0$ for all k, l, so $\langle A_1^+|+\rangle = 0$. Let $\Pi_{(+1)}$ denote the projection onto the (+1)-eigenspace of WF_1. Then we have $||\Pi_{(+1)}|+\rangle||^2 = ||\Pi_{00}|+\rangle||^2 = \frac{2(N+2)}{3N}$ by Lemma 4.

We bound $\|\Pi_{(+1)}|f_1\rangle\|^2$ by observing that

$$\begin{aligned}\|\Pi_{(+1)}|f_1\rangle\|^2 &= \sum_{\theta_{kl}=\pi} \|\Pi_{kl}|f_1\rangle\|^2 \\ &= \frac{N}{N+4}\sum_{\theta_{kl}=\pi} \|\Pi_{kl}|-\rangle\|^2 \\ &= \frac{N}{N+4}\sum_{\theta_{kl}=\pi} \frac{2\dim(\mathsf{W}_{kl})}{N} \in O\Big(\frac{1}{\sqrt{N}}\Big),\end{aligned}$$

where the final bound follows from the property that there are $O(\sqrt{N})$ terms in the sum.

Thus, using equation (3.40) and the property that $\Pi_{(+1)}|+\rangle$ and $\Pi_{(+1)}|f_1\rangle$ are orthogonal,

$$\begin{aligned}g_0^2 &= \frac{3N(N+4)}{4(N+1)^2}\|\Pi_{(+1)}|+\rangle\|^2 + \frac{(2-N)^2}{4(N+1)^2}\|\Pi_{(+1)}|f_1\rangle\|^2 \\ &= \frac{3N(N+4)}{4(N+1)^2}\Big(\frac{2(N+2)}{3N}\Big) + O\Big(\frac{1}{\sqrt{N}}\Big) \\ &= \frac{1}{2} + O\Big(\frac{1}{\sqrt{N}}\Big).\end{aligned}$$

To bound g_1^2, consider the eigenvector decomposition given in equation (3.33) in the case where $\alpha = \varphi_1$. Then $|A_1^+\rangle$ is the normalized version of $|\zeta\rangle$. By definition, we have $\||\zeta\rangle\|^2 \geq \frac{1}{2}$, so

$$\begin{aligned}g_1^2 &= |\langle A_1^+|f_2\rangle|^2 \\ &= \cos^2(2\eta)|\langle A_1^+|f_1\rangle|^2 + \sin^2(2\eta)|\langle A_1^+|+\rangle|^2 \\ &= \cos^2(2\eta)|\langle A_1^+|f_1\rangle|^2 \\ &\leq |\langle\zeta|f_1\rangle|^2/\||\zeta\rangle\|^2 \\ &\leq 2|\langle\zeta|f_1\rangle|^2 \\ &= 2\left|a\sqrt{\frac{N}{N+4}}\Big(1-\frac{4}{N}\Big) - \frac{2}{\sqrt{N+4}}\right|^2 \in O\Big(\frac{1}{\log N}\Big),\end{aligned}$$

where the final bound follows from Lemma 8. □

Lemma 11 shows that $|f_2\rangle$ has a large constant overlap with the $(+1)$-eigenspace of WF_1, and a vanishing overlap with the slowest rotational space. Intuitively, this suggests the reflection F_2 will have little effect on the slowest rotational subspace of the intermediate walk WF_1. As discussed in Section 3.5, this subspace is where most of the action of the walk takes place. This observation is the basis for the proof of Lemma 13.

In the next lemma, we use the constraints from Lemma 19 to show that the smallest positive eigenphase of WF is asymptotically close to φ_1.

Lemma 12 *Let β denote the smallest eigenphase of* WF. *Then*

$$\beta = \varphi_1 - O\left(\frac{1}{\sqrt{N}(\log N)^{3/2}}\right). \tag{3.44}$$

Note that in particular, this implies $\beta \in \Theta(\frac{1}{\sqrt{N \log N}})$.

Proof First, we derive an upper bound on β using the flip-flop theorem. By Lemma 4, $|+\rangle$ intersects every eigenspace of W. Each of the eigenvectors $\Pi_{kl}|+\rangle$ is also an eigenvector of WF_1, so in particular, this implies that $g_0 > 0$, $g_{-1} > 0$, and $g_k > 0$ for the k corresponding to the eigenphases θ_{kl}. Therefore, by Theorem 7, $0 < \beta < \varphi_1$. Applying Lemma 6, we obtain $\beta \in O(\frac{1}{\sqrt{N \log N}})$.

To obtain the upper bound on $\varphi_1 - \beta$, we apply Lemma 19, which states that β must satisfy

$$g_0^2 \cot\left(\frac{\beta}{2}\right) + \sum_{k=1}^{m} g_k^2 \left[\cot\left(\frac{\varphi_k + \beta}{2}\right) - \cot\left(\frac{\varphi_k - \beta}{2}\right)\right] - g_{-1}^2 \tan\left(\frac{\beta}{2}\right) = 0. \tag{3.45}$$

We apply trigonometric identities to rewrite this as

$$g_0^2 \cot\left(\frac{\beta}{2}\right) - 2\cot\left(\frac{\beta}{2}\right) \sum_{k=1}^{m} g_k^2 \left(\frac{\cot^2(\frac{\varphi_k}{2}) + 1}{\cot^2(\frac{\beta}{2}) - \cot^2(\frac{\varphi_k}{2})}\right) - g_{-1}^2 \tan\left(\frac{\beta}{2}\right) = 0.$$

Applying our upper bound on β, we know that $g_{-1}^2 \tan^2(\frac{\beta}{2}) \in O(\frac{1}{N \log N})$, so we have

$$2\sum_{k=1}^{m} g_k^2 \frac{\cot^2(\frac{\varphi_k}{2}) + 1}{\cot^2(\frac{\beta}{2}) - \cot^2(\frac{\varphi_k}{2})} = g_0^2 - O\left(\frac{1}{N \log N}\right) = \frac{1}{2} + O\left(\frac{1}{\sqrt{N}}\right), \tag{3.46}$$

where the last equality follows by Lemma 11.

Because the smallest positive eigenphase of W has order $\Theta(\frac{1}{\sqrt{N}})$, Theorem 7 implies that $\varphi_k \in \Omega(\frac{1}{\sqrt{N}})$ for $k \geq 2$. Therefore, $\cot^2(\frac{\varphi_k}{2}) \in O(N)$ for $k \geq 2$, while $\cot^2(\frac{\beta}{2}) \in \Omega(N \log N)$. Also note that $\sum_k g_k^2 \leq 1$. This means that

$$\sum_{k=2}^{m} g_k^2 \frac{\cot^2(\frac{\varphi_k}{2})+1}{\cot^2(\frac{\beta}{2})-\cot^2(\frac{\varphi_k}{2})} \in O\Big(\frac{1}{\log N}\Big). \tag{3.47}$$

Combining equation (3.46) and equation (3.47), we get

$$2g_1^2 \frac{\cot^2(\frac{\varphi_1}{2})+1}{\cot^2(\frac{\beta}{2})-\cot^2(\frac{\varphi_1}{2})} = \frac{1}{2} - O\Big(\frac{1}{\log N}\Big).$$

We know that $\cot^2(\frac{\varphi_1}{2}) \in O(N \log N)$ by Lemma 6. We also have $g_1^2 \in O(\frac{1}{\log N})$ by Lemma 11. This implies that

$$\cot^2\Big(\frac{\beta}{2}\Big) - \cot^2\Big(\frac{\varphi_1}{2}\Big) \in O(N).$$

Because $\cot(\frac{\beta}{2}) + \cot(\frac{\varphi_1}{2}) \in \Omega(\sqrt{N \log N})$, it must be the case that

$$\cot\Big(\frac{\beta}{2}\Big) - \cot\Big(\frac{\varphi_1}{2}\Big) \in O\Big(\frac{\sqrt{N}}{\sqrt{\log N}}\Big).$$

Applying the Taylor expansion for cotangent, we get

$$\frac{1}{\beta} - \frac{1}{\varphi_1} = \frac{\varphi_1 - \beta}{\varphi_1 \beta} \in O\Big(\frac{\sqrt{N}}{\sqrt{\log N}}\Big).$$

We know that $\frac{1}{\varphi_1} \in \Theta(\sqrt{N \log N})$, so this implies that $\frac{1}{\beta} \in \Theta(\sqrt{N \log N})$. Thus,

$$\varphi_1 - \beta \in O\Big(\frac{1}{\sqrt{N}(\log N)^{3/2}}\Big).$$

□

Finally, we show that $|\pi_z\rangle + |\circlearrowleft\rangle$ lies asymptotically in the slowest rotational subspace of WF. To do so, we apply our bounds from Lemmas 11 and 12 to show that the slowest eigenvectors of WF are asymptotically close to the slowest eigenvectors of WF_1.

Lemma 13 *Let Π_β denote the projection onto the slowest rotational subspace of* WF*, which is spanned by the eigenvectors with eigenphases $\pm\beta$. Then*

$$\left\| \Pi_\beta \Big(|\pi_z\rangle + |\circlearrowleft\rangle \Big) \right\| = \sqrt{2} - O\Big(\frac{1}{\log N}\Big). \tag{3.48}$$

Proof We use the decomposition of $|f_2\rangle$ in equation (3.41). By Lemma 19, the (unnormalized) eigenvector of WF associated with eigenphase β is $|e_\beta\rangle = |f_2\rangle + \imath|e_\beta^\perp\rangle$, where

$$\begin{aligned} |e_\beta^\perp\rangle = g_0 \cot\Big(\frac{\beta}{2}\Big)|A_0\rangle + \sum_{k=1}^{m} g_k \Big[\cot\Big(\frac{\beta-\varphi_k}{2}\Big)|A_k^+\rangle + \cot\Big(\frac{\beta+\varphi_k}{2}\Big)|A_k^-\rangle \Big] \\ - g_{-1}\tan\Big(\frac{\beta}{2}\Big)|A_{-1}\rangle. \end{aligned} \tag{3.49}$$

Let $|B_1^+\rangle$ denote the normalization of $|e_\beta\rangle$, and let $|B_1^-\rangle$ denote the conjugate of $|B_1^+\rangle$. Then Π_β is a projection onto the span of $|B_1^+\rangle$ and $|B_1^-\rangle$.

We know from the proof of Lemma 10 that

$$\left| \Big(\langle A_1^+| + \langle A_1^-| \Big) \Big(|\pi_z\rangle + |\circlearrowleft\rangle \Big) \right| = 2 - O\Big(\frac{1}{\log N}\Big),$$

where $|A_1^+\rangle, |A_1^-\rangle$ are the normalizations of $|\zeta\rangle$ and $|\overline{\zeta}\rangle$, respectively. We prove Lemma 13 by showing that $|A_1^+\rangle$ and $|A_1^-\rangle$ have large overlap with $|B_1^+\rangle$ and $|B_1^-\rangle$, respectively.

We know from equation (3.49) that

$$\begin{aligned} \||e_\beta\rangle\|^2 &= \||e_\beta^\perp\rangle\|^2 + \||f_2\rangle\|^2 \\ &= g_0^2 \cot^2\Big(\frac{\beta}{2}\Big) + \sum_{k=1}^{m} g_k^2 \Big[\cot^2\Big(\frac{\beta-\varphi_k}{2}\Big) + \cot^2\Big(\frac{\beta+\varphi_k}{2}\Big) \Big] + g_{-1}^2 \tan^2\Big(\frac{\beta}{2}\Big) + 1. \end{aligned}$$

By Lemmas 11 and 12, we know that $g_0^2\cot^2(\frac{\beta}{2}) \in O(N\log N)$ and that $g_{-1}^2\tan^2(\frac{\beta}{2}) \in O(\frac{1}{N\log N})$. Recall from equation (3.47) that

$$\sum_{k=2}^{m} g_k^2 \Big[\cot^2\Big(\frac{\beta-\varphi_k}{2}\Big) + \cot^2\Big(\frac{\beta+\varphi_k}{2}\Big) \Big] \in O\Big(\frac{1}{\log N}\Big).$$

Finally, we know from Lemma 12 that $\cot^2(\frac{\beta+\varphi_1}{2}) \in O(N \log N)$. Thus,

$$\| |e_\beta\rangle \|^2 = g_1^2 \cot^2\Big(\frac{\beta-\varphi_1}{2}\Big) + O(N \log N).$$

By Lemma 11, $g_1^2 \in O(\frac{1}{\log N})$, and by Lemma 12, $\cot^2(\frac{\beta-\varphi_1}{2}) \in \Omega(N(\log N)^3)$. Therefore,

$$\begin{aligned}
|\langle A_1^+|B_1^+\rangle|^2 &= \frac{|\langle A_1^+|e_\beta\rangle|^2}{\| |e_\beta\rangle\|^2} \\
&= \frac{|\langle A_1^+|f_2\rangle + \imath \langle A_1^+|e_\beta^\perp\rangle|^2}{\| |e_\beta\rangle\|^2} \\
&= \frac{\Big|g_1 + \imath g_1 \cot\Big(\frac{\beta-\varphi_1}{2}\Big)\Big|^2}{\| |e_\beta\rangle\|^2} \\
&= 1 - O\Big(\frac{1}{\log N}\Big),
\end{aligned}$$

so $|\langle A_1^+|B_1^+\rangle| = 1 - O(\frac{1}{\log N})$. Similarly, one can show that

$$\begin{aligned}
|\langle A_1^-|B_1^+\rangle| &\in O\Big(\frac{1}{\log^2 N}\Big), \\
|\langle A_1^+|B_1^-\rangle| &\in O\Big(\frac{1}{\log^2 N}\Big), \\
|\langle A_1^-|B_1^-\rangle| &= 1 - O\Big(\frac{1}{\log N}\Big).
\end{aligned}$$

Combining these results, we get

$$\Big|\Big(\langle B_1^+| + \langle B_1^-|\Big)\Big(|A_1^+\rangle + |A_1^-\rangle\Big)\Big| = 2 - O\Big(\frac{1}{\log N}\Big).$$

Therefore,

$$\begin{aligned}
\Big\|\Pi_\beta\Big(|\pi_z\rangle + |\circlearrowleft\rangle\Big)\Big\| &\geq \frac{1}{\sqrt{2}}\Big|\Big(\langle B_1^+| + \langle B_1^-|\Big)\Big(|\pi_z\rangle + |\circlearrowleft\rangle\Big)\Big| \\
&\geq \frac{1}{2\sqrt{2}}\Big|\Big(\langle B_1^+| + \langle B_1^-|\Big)\Big(|A_1^+\rangle + |A_1^-\rangle\Big)\Big|\Big|\Big(\langle A_1^+| + \langle A_1^-|\Big)\Big(|\pi_z\rangle + |\circlearrowleft\rangle\Big)\Big| \\
&= \sqrt{2} - O\Big(\frac{1}{\log N}\Big).
\end{aligned}$$

□

3.6.2 Proof of main result

Through Lemmas 12 and 13, we have an asymptotic description of the slowest rotational subspace of WF. The description shows that as N increases, the action of WF on $|\pi_z\rangle$ approaches a rotation in this slowest rotational subspace. This property is what allows us to map our initial state to the target state $|\circlearrowleft\rangle$ with probability approaching 1. Applying the relationship $\mathsf{WF} = (\mathsf{U})_z$, we thus obtain the proof of our main result.

Theorem 2 (Main result) *Fix* $s = 1 - \frac{1}{N+1}$ *and suppose* $n_r = n_c$*. Then there exists a constant* $c > 0$ *such that after* $c\sqrt{N \log N}$ *applications of* U *to* $|\pi\rangle$*, measuring the state will produce* $|\circlearrowleft\rangle$ *with probability* $1 - e(N)$*, where* $e(N) \in O(\frac{1}{\log N})$.

Proof First, observe that for any k,

$$\mathsf{U}^k|\pi\rangle = \mathsf{cz}(\mathsf{WF})^k\mathsf{cz}|\pi\rangle = \mathsf{cz}(\mathsf{WF})^k|\pi_z\rangle.$$

Thus, we prove that after $c\sqrt{N\log N}$ applications of WF to $|\pi_z\rangle$, measuring the state will produce $\mathsf{cz}|\circlearrowleft\rangle = |\circlearrowleft\rangle$ with the stated probability.

Recall from equation (3.28) that $|\pi_z\rangle$ can be decomposed as

$$|\pi_z\rangle = \frac{1}{2}|\mathsf{U}_0\rangle + \frac{1}{2}\Big(|\pi_z\rangle + |\circlearrowleft\rangle\Big).$$

By Lemma 5, we know that $|\mathsf{U}_0\rangle$ is a (+1)-eigenvector of WF. Letting Π_β denote the projection onto the slowest rotational subspace of WF, we decompose $|\pi_z\rangle + |\circlearrowleft\rangle$ as

$$|\pi_z\rangle + |\circlearrowleft\rangle = \Pi_\beta\Big(|\pi_z\rangle + |\circlearrowleft\rangle\Big) + |\perp\rangle.$$

for some vector $|\perp\rangle$. By Lemma 13, we know that $\||\perp\rangle\| \in O(\frac{1}{\log N})$. We also know from Lemma 12 that applying WF to a vector in the slowest rotational subspace will result in a rotation of the vector by the angle $\beta \in \Theta(\frac{1}{\sqrt{N\log N}})$. Therefore, there exists a constant c such that $c\sqrt{N\log N} = \lfloor\frac{\pi}{\beta}\rfloor = k$. After k applications of WF to $|\pi_z\rangle$, we get the state

$$\begin{aligned}(\mathsf{WF})^k|\pi_z\rangle &= \frac{1}{2}\Big(|\pi_z\rangle - |\circlearrowleft\rangle\Big) - \frac{1}{2}\Pi_\beta\Big(|\pi_z\rangle + |\circlearrowleft\rangle\Big) + |\rho\rangle \\ &= -|\circlearrowleft\rangle + \frac{1}{2}|\perp\rangle + |\rho\rangle.\end{aligned}$$

Here, $|\rho\rangle$ is some state that captures both the result of applying $(\mathsf{WF})^k$ to $|\perp\rangle$ and the small error incurred by the rounding of $\frac{\pi}{\beta}$, and has norm $\||\rho\rangle\| \in O(\frac{1}{\log N})$. Thus, measuring the state will produce $|\circlearrowleft\rangle$ with probability $1-e(N)$, where $e(N) \in O(\frac{1}{\log N})$. $\square$

3.7 Conclusion

We consider the canonical problem of spatial search on the two-dimensional grid. For this problem, we give a memoryless quantum walk that finds a unique marked vertex using minimal memory and $\Theta(\sqrt{N \log N})$ steps. Compared to previous memoryless walks on the grid, our walk boosts the probability of measuring the marked state from $O(\frac{1}{\log N})$ to $1-O(\frac{1}{\log N})$, while also preserving the simplicity of the tessellation-based structure.

We achieve this by adding a selfloop to the marked vertex. In doing so, we show how interpolated walks can be usefully adapted to the memoryless setting. We also give a precise analysis of how the selfloop affects the walk dynamics. With our chosen value for the selfloop weight, we show that our walk asymptotically reduces to a rotation in a single two-dimensional subspace. Applying this rotation evolves the initial state to the selfloop state, from which the marked state can be obtained by straightforward amplitude amplification.

As part of our proof, we give a precise description of the slowest rotational subspace of our memoryless walk operator. This is done using its decomposition into a real operator composed with a two-dimensional rotation. The techniques we use to analyse such an operator are general enough that they have the potential to be used in the analysis of other walks as well. This includes developing and analysing memory-optimal spatial search algorithms for other types of graph, as well as for handling graphs with multiple marked vertices.

Chapter 4

Perfect matchings and 0-determinant verification

4.1 Introduction

In the perfect matchings problem, the goal is to find a subset of edges in a graph such that every vertex is incident to exactly one edge in the set. This could be applied, for example, to pair all people at a party such that each pair consists of two people who have never met. Finding matchings in graphs is a canonical problem in computer science and has driven many new ideas, including the formulation of polynomial asymptotic growth as the standard for efficient computation [Edm65].

In this chapter, we consider the decision version of the perfect matchings problem, which we refer to as perfect matching detection. Given a graph G on n vertices, the goal is to decide whether there exists a subset of the edges in G that constitutes a perfect matching. The decision version of the problem is more natural to consider in our chosen model of computation, and is no harder than the problem of finding such a matching.

We examine the perfect matchings problem in the quantum query model, where the adjacency matrix for the graph is accessed through quantum queries to a black box. In this model, the complexity of an algorithm is given by the number

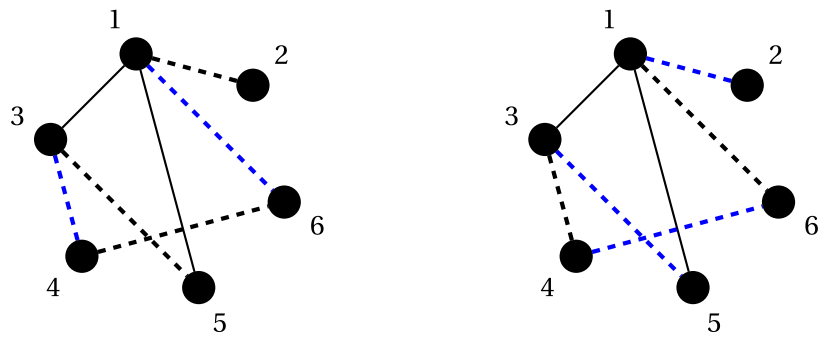


Figure 4.1: An example of an augmenting path for a graph, indicated by dotted lines, given the partial matching $\{(1,6),(3,4)\}$ shown in blue on the left graph. Observe that by exchanging edges in the path that are not in the matching with those that are, the total size of the matching increases by one to become $\{(1,2),(3,5),(4,6)\}$, as shown on the right.

of times we need to query this black box to solve the perfect matching decision problem with constant success probability. The advantage of the quantum query model is that it is simple to work with, and lower bounds on query complexity immediately imply lower bounds on step complexity. We remark that in this model, where the queries are to the adjacency matrix of a graph, any graph problem can be solved with $O(n^2)$ queries by simply querying all entries of the matrix.

The earliest classical algorithms for finding matchings are based on an incremental construction by augmenting paths, initially proposed by Edmonds [Edm65]. Given a partial matching, an augmenting path is one whose edges alternate between edges in the matching and ones that are not, and which starts and ends at vertices that are not incident to any edges in the matching. Then, by exchanging edges in the path that are not in the matching with those that are, the total size of the matching can be increased (see Figure 4.1). An algorithm by Hopkroft and Karp [HK73] uses this approach to find a maximum matching in a bipartite graph in $O(n^{5/2})$ steps. Remarkably, their algorithm can be extended to work for general graphs with no increase in asymptotic complexity, as shown by Micali and Vazirani [MV80].

Quantum algorithms for matchings have so far been developed by applying quantum techniques to speed up the search for augmenting paths. By using quantum search [BHMT02] to find new edges, one can find a maximum matching in $O(n^2 \log^2 n)$ steps for both bipartite [AŠ06] and general [Dör09] graphs. In the quantum query model, the first quantum algorithm to improve on the trivial query complexity of $O(n^2)$ was developed by Lin and Lin [LL16]. They give an algorithm to find a maximum matching in a bipartite graph using only $O(n^{7/4})$ queries to the adjacency matrix. Their algorithm applies a novel guessing-tree approach, based on the Elitzur-Vaidman bomb tester [EV93]. Kimmel and Witter [KW21] show how to apply similar techniques [BT19] to find a maximum matching in a general graph with $O(n^{7/4})$ queries.

Despite this recent progress, there remains a gap between the current upper bound of $O(n^{7/4})$ quantum queries and the best-known lower bound of $\Omega(n^{3/2})$, as proven by Zhang [Zha05]. This gap holds for both the problem of finding a perfect matching, and for the restricted problem of deciding whether a graph contains a perfect matching. In this chapter, we investigate this gap by proposing a new quantum algorithm for the perfect matching decision problem.

An alternative to the augmenting paths approach for finding matchings was first proposed by Lovász [Lov79], who gives an algorithm based on a characterization by Tutte [Tut47]. Tutte shows that a graph contains a perfect matching if and only if its associated Tutte matrix is nonsingular. Using this property, Lovász gives a classical randomized algorithm that can detect, but not find, a perfect matching in $O(n^\omega)$ steps, where $2 \leq \omega \leq 2.38$ is the exponent for matrix multiplication. Subsequent modifications by Rabin and Vazirani [RV89] and Mucha and Sankowski [MS04] show how this algorithm can be applied to find a maximum matching on any graph in $O(n^\omega)$ steps, also with vanishing error probability.

We give a quantum analogue to Lovász's algorithm for detecting perfect matchings. As with Lovász's algorithm, our algorithm randomly instantiates the Tutte matrix for the input graph, and then we apply quantum methods to decide whether the resulting matrix is singular. To our knowledge, this is the first quantum algorithm to use Tutte's algebraic characterization.

The key component of our algorithm is a new span program that solves the 0-determinant verification problem. In this problem, we are given an $n \times n$ input matrix A, and the task is to decide whether A is singular. We give an algorithm for this decision problem using the span program framework, which is a linear-algebraic model of computation that can also be used as a tool to design quantum algorithms. Because 0-determinant is a linear-algebraic problem, span programs are a natural choice for this purpose. Using Reichardt's transformation [Rei09, Rei11], any span program can be compiled into a quantum algorithm whose query complexity is determined by a span program quantity known as the witness size. Thus, bounding the quantum query complexity of our algorithm for 0-determinant verification reduces to bounding the witness size of our span program.

The witness size of our span program is determined by the upper bound on the quantity $c(\mathsf{A})$, which depends on the spectrum of the possible nonsingular input matrices. Letting $\sigma_1, \ldots \sigma_n$ denote the singular values of the nonsingular matrix A, we define

$$c(\mathsf{A}) = \sqrt{\frac{1}{n}\left(\frac{1}{\sigma_1^2} + \frac{1}{\sigma_2^2} + \ldots + \frac{1}{\sigma_n^2}\right)}. \tag{4.1}$$

This makes sense intuitively, as it should be hard to decide whether a matrix is singular when it has eigenvalues that are close to zero. Our first main result is then the following:

Theorem 3 *Our span program algorithm solves the* 0*-determinant verification problem with two-sided bounded error in $O(\sqrt{n}LT)$ quantum queries, under the promise that any input matrix has entries with norm at most one, and any input matrix with full rank satisfies $c(\mathsf{A}) \leq T$. Here, L represents the cost of loading a matrix into a span program.*

As discussed in Section 4.3, the loading cost for general matrices is $O(n)$, although the bound can be improved for sparse matrices.

Belovs gives a span program for matrix rank-finding [Bel11], which can be applied to solve 0-determinant verification. Our span program gives a simplified approach to the 0-determinant verification problem and matches Belovs' algorithm in asymptotic query complexity. Our span program also applies similar design principles to that of Belovs' algorithm. Both span programs use randomized vectors and are phrased in terms of a high level span program – a construction defined by Belovs that allows the span program vectors to be constructed from the columns of the input matrix A.

We apply our span program to detect perfect matchings using an algorithm inspired by that of Lovász. Given a graph G, we randomly instantiate the Tutte matrix with elements from a set S and then apply our span program for 0-determinant verification. This approach leads to our second main result.

Theorem 4 *Given a graph G on n vertices, a bound R, and a set S whose elements have norm at most one, our algorithm can decide whether G contains a perfect matching with two-sided bounded error in $O(n^{3/2}R)$ quantum queries, under the following condition. If G contains a perfect matching, then the Tutte matrix $\mathsf{T}(G)$ must be nonsingular with constant probability when instantiated uniformly at random from S, and given this is the case, $c(\mathsf{T}(G)) \leq R$ with constant probability.*

The complexity of our algorithm therefore depends on both the set of possible input graphs and the choice of S. There may be classes of graphs and sets S for which our algorithm gives an improvement over the $O(n^{7/4})$ algorithm by Kimmel and Witter [KW21]. We leave this as an interesting direction for future work.

We argue that there may also be classes of graphs for which the bound R must be exponential in n, regardless of the choice of set S. We suggest how these graphs could be constructed based on hard cases for 0-determinant verification. The resulting class of graphs has not yet been considered in context of matchings, so may yield new insights into the perfect matchings problem.

The chapter is organized as follows. In Section 4.2, we introduce the Tutte matrix and show how it can be used to solve the perfect matching detection problem. In Section 2.3, we give an overview of span programs, which are the framework we use to design our algorithm. In Section 4.3, we present our span program for 0-determinant verification. We then discuss how it can be applied to detect perfect matchings in Section 4.4, where we also provide an explicit example of how the loaded span program can be constructed for the small graph in Figure 4.2.

Finally, we include a discussion in Appendix 5 concerning the lower bound for 0-determinant verification, which is the problem solved by our span program from Section 4.3. We show that the lower bound of $\Omega(n^2)$ given by Dörn and Theirauf [DT09] only applies to general k-determinant verification, where the problem is to decide whether the determinant of a matrix is equal to some value k, and does not apply to the specific case of 0-determinant verification. This leaves a gap between the remaining lower bound of $\Omega(n)$ and the trivial upper bound of $O(n^2)$ for the problem.

4.2 The Tutte matrix

Let $G = (V, E)$ be an undirected graph with $|V| = n$. A **matching** of G is a subset $M \subseteq E$ such that no two edges in M share a vertex. We refer to $|M|$ as the **size** of the matching.

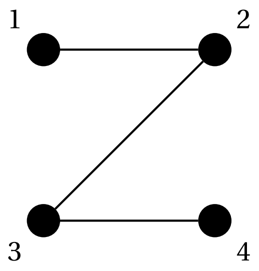
A matching is **perfect** if every vertex $v \in V$ is incident to some edge $e \in M$, or equivalently, if the matching has size $n/2$. Note that if n is odd, then the graph trivially has no perfect matching. A matching is **maximal** if there is no matching M' strictly larger than M such that $M \subset M'$. A matching is **maximum** if there are no matchings with size greater than M.

We consider the problem of deciding whether a graph G has a perfect matching. Tutte [Tut47] gave an algebraic characterization of this problem in terms of the following matrix.

Definition 5 *Given a graph G, the **Tutte matrix** associated with G is the $n \times n$ skew-symmetric matrix $\mathsf{T}(G)$ for which*

$$\mathsf{T}(G)_{ij} = \begin{cases} x_{ij} & \textit{if} (i,j) \in E \textit{ and } i < j \\ -x_{ij} & \textit{if} (i,j) \in E \textit{ and } i > j \\ 0 & \textit{otherwise} \end{cases} \tag{4.2}$$

and the x_{ij} are uninstantiated formal variables.



$$\mathsf{T}(G) = \begin{bmatrix} 0 & x_{12} & 0 & 0 \\ -x_{12} & 0 & x_{23} & 0 \\ 0 & -x_{23} & 0 & x_{34} \\ 0 & 0 & -x_{34} & 0 \end{bmatrix}$$

Figure 4.2: A graph with $n = 4$ vertices and the corresponding Tutte matrix. Observe that the Tutte matrix has nonzero determinant, which occurs if and only if the graph contains a perfect matching.

Note that an uninstantiated formal variable is one for which no value is yet assigned. Later, we will randomly instantiate the variables x_{ij} by assigning them numeric values from a set S according to some probability distribution.

An example of the Tutte matrix for a small graph is given in Figure 4.2. The Tutte matrix encodes useful information about the matchings of G, as shown by Tutte in the following theorem.

Theorem 5 ([Tut47]) $\det(\mathsf{T}(G)) \neq 0$ *if and only if* G *has a perfect matching.*

The Tutte matrix thus reduces the perfect matching decision problem to deciding whether a randomly instantiated matrix is singular.

Directly evaluating the symbolic determinant of $\mathsf{T}(G)$ is not feasible in practice, as the number of terms in the resulting polynomial can be exponential in n. However, randomly instantiating the variables x_{ij} allows a probabilistic algorithm for testing whether the determinant of $\mathsf{T}(G)$ is uniformly 0. This idea was proposed by Lovász [Lov79], who gave an algorithm that instantiates the x_{ij} with integers chosen uniformly at random from the set $S = \{1, 2, 3, \ldots, n^2\}$. This instantiation gives an algorithm with one-sided error (the possibility of a false negative) in the following way. If the graph G has a perfect matching, then the determinant of the instantiated Tutte matrix is nonzero with probability $1 - \frac{1}{n}$, otherwise the determinant is zero with certainty. Lovász's algorithm can therefore decide whether G has a perfect matching in time $O(n^{\omega})$ with vanishing error probability.

Lovász's choice of set S is based on the following lemma, which was published in different forms by Schwartz [Sch80], Zippel [Zip79], and DeMillo and Lipton [DL78].

Lemma 14 (Schwartz-Zippel) *Let $P \in \mathbb{C}[x_1,\dots,x_k]$ be a nonzero polynomial of degree $d \geq 0$. Let S be a finite subset of $\mathbb{C}$ and let $r_1,\dots,r_k$ be selected independently and uniformly at random from S. Then*

$$\Pr[P(r_1,\dots,r_k)=0] \leq \frac{d}{|S|}.$$

Our goal is to obtain an algorithm to detect perfect matchings with at most a constant probability of error. The symbolic determinant of $\mathsf{T}(G)$ is a degree-n polynomial, so to show the one-sided error probability is at most a constant by the bound in Lemma 14, one would require $|S|$ to grow asymptotically with n. Below, we prove a tighter bound using properties of the Tutte matrix. Our result shows that a constant-size set S is sufficient to achieve a constant bound on the error probability.

Lemma 15 *Let $P \in \mathbb{C}[x_1,\dots,x_k]$ be a nonzero polynomial of degree $d \geq 0$ such that for each term, the maximum degree of any variable x_i is 2. Let S be a finite subset of $\mathbb{C}$ with $0 \notin S$, and let $r_1,\dots,r_k$ be selected independently and uniformly at random from S. Then if $|S| \geq 12$,*

$$\Pr[P(r_1,\dots,r_k)=0] \leq \frac{1}{2}.$$

Proof We prove the inequality by induction on k. For the case $k=1$, we have

$$\Pr[P(r_1)=0] \leq \frac{1}{|S|} \leq \frac{1}{2},$$

by the fundamental theorem of algebra. Now, suppose $k>1$ and that for all polynomials P' in $0<k'<k$ variables,

$$\Pr[P'(r_1,\dots,r_{k'})=0] \leq \frac{1}{2}.$$

We can write

$$P(x_1,\dots,x_k) = \sum_{i=0}^{2} x_k^i P_i,$$

where P_0, P_1 and P_2 are polynomials in $x_1, \ldots, x_{k-1}$. To bound the probability that $P(r_1, \ldots, r_k) = 0$, we consider cases based on which of P_0, P_1 and P_2 are the uniformly zero polynomial. Note that P_0, P_1 and P_2 cannot simultaneously be the zero polynomial.

1. If any two of P_0, P_1 and P_2 are the zero polynomial, then using the fact that $0 \notin S$, the only way P can evaluate to 0 is if the remaining P_i evaluates to 0. Each of the P_i is a polynomial in fewer than k variables, so in this case,
$$\Pr[P(r_1, \ldots, r_k) = 0] \le \frac{1}{2}.$$
2. If only one of P_0, P_1 and P_2 is the uniformly zero polynomial, then either the other two P_i both evaluate to 0, or there are at most two values of x_k such that the polynomial P evaluates to 0. Therefore in this case,
$$\Pr[P(r_1, \ldots, r_k) = 0] \le \left(\frac{1}{2}\right)^2 + \frac{2}{|S|}.$$
3. If none of P_0, P_1 and P_2 are the zero polynomial, then there are three ways P could evaluate to 0. First, P_0, P_1 and P_2 could all evaluate to 0. Second, only one of P_0, P_1 and P_2 could evaluate to 0. If it is P_0 or P_2 that evaluates to 0, there is at most one value for x_k that will cause P to evaluate to 0, and if it is P_1 that evaluates to 0, then there are at most two possible values for x_k that would cause P to evaluate to 0. Third, none of P_0, P_1 and P_2 could evaluate to 0, in which case there are at most two values of x_k that would cause P to evaluate to 0. Therefore,
$$\Pr[P(r_1, \ldots, r_k) = 0] \le \left(\frac{1}{2}\right)^3 + \frac{1}{2}\left(\frac{4}{|S|}\right) + \frac{2}{|S|}.$$

Using the fact that $|S| \ge 12$, the probability that $P(r_1, \ldots, r_k) = 0$ is at most $\frac{1}{2}$ in all cases. We have already shown the inequality holds for $k = 1$, so by induction, it is true for all values of k. □

The symbolic determinant of $\mathsf{T}(G)$ satisfies the conditions of Lemma 15. Thus, any set S such that $0 \notin S$ and $|S| \ge 12$ can be used to obtain a classical algorithm for the perfect matching decision problem with one-sided bounded error probability at most $\frac{1}{2}$.

4.3 Span program for 0-determinant verification

We present a new span program that can decide whether a given $n \times n$ matrix A is singular. Our span program yields an algorithm that matches existing quantum algorithms in asymptotic query complexity, and has an attractively simple structure. We remark that in [Bel11], Belovs gives a span program for deciding whether the rank of a matrix is at least some threshold r, which solves 0-determinant verification when $r = n$. Our approach is different from that of Belovs, but has the same asymptotic query complexity as applying Belovs' algorithm to solve 0-determinant verification.

First, we present our span program and compare it to Belovs' approach. An analysis of the witness size for our span program is given in Section 4.3.1.

Both our algorithm and Belovs' algorithm are given in terms of a **high level** span program, for which the available vectors are constructed from the columns of the input matrix A. In the standard definition of a span program (see Definition 3), the input vectors are fixed by the program definition, and the input bits are used to determine which of these input vectors are available. The inclusion of each input vector depends only on a single bit of the input. Instead, we would like to have our available vectors equal to the columns of our input matrix A, and each column depends on multiple bits of the input. To achieve this, an additional construction is required. In [Bel11], Belovs shows how to build available vectors from the columns of a matrix by giving a method to construct a standard span program from a high level span program. This extra construction, known as **matrix loading**, incurs a multiplicative cost of L to the witness size.

Belovs shows that loading a general $n \times n$ matrix into a high-level span program requires $\Theta(n)$ queries, under the assumption that the matrix entries are bounded in norm by one. For sparse matrices, it is possible to reduce the loading cost, which is why it is kept as a separate factor in Theorems 3 and 6. We give a full example of how a high-level span program can be converted into a standard span program in Section 4.4.1, showing explicitly how this loading can be done.

Belovs' approach to 0-determinant verification is to test whether a matrix is singular by randomly generating a target vector whose entries are chosen from the normal distribution, and then testing whether it lies in the span of the matrix columns. We state Belovs' result in Theorem 6, restricted to the case of 0-determinant verification. The query complexity depends on the worst-case quadratic mean of the singular values of a nonsingular input matrix A. This is given as an upper bound on the value $c(\mathsf{A})$, which we recall from equation (4.1) to be defined as

$$c(\mathsf{A}) = \sqrt{\frac{1}{n}\left(\frac{1}{\sigma_1^2} + \frac{1}{\sigma_2^2} + \ldots + \frac{1}{\sigma_n^2}\right)}. \tag{4.3}$$

As noted by Belovs, an equivalent definition is

$$c(\mathsf{A}) = \frac{\|\mathsf{A}^{-1}\|_F}{\sqrt{n}}, \tag{4.4}$$

where $\|.\|_F$ denotes the Frobenius norm.

Theorem 6 ([Bel11]) *The 0-determinant verification problem can be solved with two-sided bounded error in $O(\sqrt{n}LT)$ quantum queries, under the promise that any input matrix has entries with norm at most one, and any input matrix with full rank satisfies $c(\mathsf{A}) \leq T$. Here, L represents the cost of loading the matrix into a span program.*

We present a new, complementary approach based on the idea that if a matrix A is singular, then there must be some nonzero linear combination of its columns that produces the zero vector. To test whether such a linear combination exists, we augment the input matrix A with a row of entries taken from the Rademacher distribution, i.e. chosen uniformly at random from $\{1,-1\}$. We define

$$\mathsf{A}' = \begin{bmatrix} \pm1 & \pm1 & \ldots & \pm1 \\ \mathsf{A}_{1,1} & \mathsf{A}_{1,2} & & \mathsf{A}_{1,n} \\ \mathsf{A}_{2,1} & \ddots & & \\ \vdots & & & \\ \mathsf{A}_{n,1} & & & \mathsf{A}_{n,n} \end{bmatrix}. \tag{4.5}$$

We let $|r\rangle = (r_i)$ denote the transposed first row of A', which is an $n \times 1$ vector whose entries follow the Rademacher distribution.

Now, define $|\tau'\rangle$ to be the $(n+1)\times 1$ vector

$$|\tau'\rangle = \begin{bmatrix} 1 \\ 0 \\ 0 \\ \vdots \\ 0 \end{bmatrix}. \tag{4.6}$$

If A has full rank, then $|\tau'\rangle$ will not be in the column space of A'. If A is singular, then there is some nonzero vector $|u\rangle$ such that $\mathsf{A}|u\rangle = \vec{0}$. In this case, $|\tau'\rangle$ will be in the column space of A' provided $\langle r|u\rangle \neq 0$. By Lemma 16, $\langle r|u\rangle \neq 0$ with probability at least $3/16$, so the probability that the instantiation of $|r\rangle$ introduces an error is at most constant. Thus, given a “good" instantiation of the first row of A', $|\tau'\rangle$ lies in the span of the columns of A' if and only if A is singular.

Our span program differs from Belovs' span program in two key ways. First, our span program samples from a binary distribution, rather from the normal distribution. This simplifies the implementation of the instantiation step, as sampling from a binary distribution is a standard operation in quantum computation. It also removes any complications arising from limited precision, and leads to a more straightforward analysis of the witness size. Second, our span program uses a fixed target vector, rather than having it determined by a random instantiation. This fixed target vector has the additional nice property that it is a standard basis state. The result is a simple and clean approach to solving 0-determinant verification.

4.3.1 Witness size

In this section, we compute the witness size of our high level span program, for which the available vectors are the columns of A' and the target is $|\tau'\rangle$. Recall that our high-level span program can be converted into a standard span program with an additional multiplicative cost of $L \in \Theta(n)$ to the witness size, and can then be converted into a quantum algorithm.

The witness size of a high level span program is computed in a similar way to that of a standard span program. The main difference is in the negative witness size, as in a high level span program, there are no unavailable vectors. Instead, the negative witness size is taken to be the norm of the entire negative witness.

Definition 6 *Consider a high level span program $\mathscr{P}$ whose available vectors are given by the columns of an $n \times m$ matrix B, where B is loaded from some input x in a domain $\mathscr{D}$ of binary input strings. Let the target vector be denoted $|\tau\rangle$, and let $f : \mathscr{D} \to \{0,1\}$ denote the function computed by $\mathscr{P}$.*

- *If $|\tau\rangle \in \operatorname{span}(\mathsf{B})$, then $f(x) = 1$ and there exists some vector $|w^+\rangle$ such that $\mathsf{B}|w^+\rangle = |\tau\rangle$. Then $|w^+\rangle$ is a **positive witness** for B with size $\||w^+\rangle\|^2$.*
- *If $|\tau\rangle \notin \operatorname{span}(\mathsf{B})$, then $f(x) = 0$ and $|\tau\rangle$ is orthogonal to the column space of B. Then there exists a vector $|w^-\rangle$ such that $\langle w^-|\tau\rangle = 1$ and $|w^-\rangle$ is orthogonal to the column space of B. Then $|w^-\rangle$ is a **negative witness** for B with size $\||w^-\rangle\|^2$.*

*The **witness size** for $\mathscr{P}$ is then defined in the standard way (see Definition 4).*

We note that by this definition, rescaling B affects the positive witness size but not the negative witness size. This is unlike in a standard span program, where uniformly rescaling the input vectors has no net effect. For a high level span program, this is accounted for in that rescaling B affects both the witness size and the loading cost. To avoid this complication, we simply assume the entries of B are bounded in norm by one.

Using Definition 6, we compute the positive and negative witness sizes of our high level span program. We begin with the following lemma, which we apply to obtain the positive witness size. The entries of $|r\rangle$ follow a Rademacher distribution, and both concentration and anti-concentration bounds are proven for random sequence sums of the form $\langle r|u\rangle$. The lower bound of 3/16 for the following inequality was recently proven by Dvořák and Klein [DK21], improving on the previous best value of 1/10 by Oleszkiewicz [Ole96].

Lemma 16 ([DK21]) *Let $|r\rangle \in \mathbb{R}^n$ be a vector whose entries are instantiated uniformly at random from* $\{1, -1\}$ *Then for any vector* $|u\rangle \in \mathbb{R}^n$,

$$\Pr\Big[|\langle r|u\rangle| \geq \||u\rangle\|\Big] \geq \frac{3}{16}.$$

Using this bound, we show that our span-program-based quantum algorithm outputs the correct answer to the 0-determinant verification problem with probability at least 2/3. This is the standard requirement for bounded-error randomized algorithms.

The first source of random error in our span program algorithm is due to the randomness in A', whose first row $\langle r|$ consists of entries sampled from the Rademacher distribution. As discussed previously, a “bad" instantiation could result in $|\tau'\rangle$ not being in the span of the columns of A', even though A' is nonsingular. The randomness in A' also has an effect on the witness size of the resulting span program, as we specify in Lemma 17.

The second source of random error in our span program algorithm comes from applying Reichardt's quantum algorithm to solve the span program. Recall from Theorem 1 that a span program can be compiled into a quantum algorithm with bounded error. By repeating this quantum algorithm a constant number of times if necessary, we can assume that the probability of an error being introduced by this step is at most 1/6. Therefore, to obtain an overall success probability of at least 2/3 in our span program algorithm, we need to show that with probability at least 5/6, A' is instantiated such that the resulting span program is correct and has a witness size within our claimed bound.

To show this, we begin with the following result. The case where A is singular follows almost immediately from Lemma 16. Our probability bounds for the case where A is non-singular are chosen so that the negative witness size is $O(c(\mathsf{A}))^2 n$ with high probability. Later, we will rebalance the success probability and witness size between the singular and nonsingular cases so that our span program is correct and has bounded witness size with probability at least 5/6 in both cases.

Lemma 17 *Consider the randomized high level span program $\mathscr{P}$ for which, given input matrix $\mathsf{A} \in \mathscr{D}$, the available vectors are the columns of A' and the target* *vector is* $|\tau'\rangle$. *Let* $f : \mathscr{D} \to \{0,1\}$ *be the function computed by* $\mathscr{P}$. *Then the following statements hold.*

- *Suppose* A *is singular. Then with probability at least* $\frac{3}{16}$, $f(\mathsf{A}) = 1$ *and the positive witness size is at most* 1.
- *Suppose* A *is invertible. Then* $f(\mathsf{A}) = 0$, *and with probability at least* $\frac{99}{100}$, *the negative witness size is at most* $100c(\mathsf{A})^2n + 1$.

Proof Suppose A is singular. A positive witness will be a vector $|w^+\rangle$ such that $\mathsf{A}|w^+\rangle = \vec{0}$ and $\langle r|w^+\rangle = 1$. Let $|u\rangle$ be any vector such that $\mathsf{A}|u\rangle = \vec{0}$. By Lemma 16, $|\langle r|u\rangle| \geq \||u\rangle\| > 0$ with probability at least $\frac{3}{16}$. If this condition holds, then $|w^+\rangle = \frac{1}{|\langle r|u\rangle|}|u\rangle$ is a positive witness for A, and $\||w^+\rangle\|^2 \leq 1$.

Next, consider the case where A is invertible. A negative witness will be a vector $|w^-\rangle$ such that $\langle\tau'|w^-\rangle = 1$ and $|w^-\rangle$ is orthogonal to the columns of A'. The requirement $\langle\tau'|w^-\rangle = 1$ implies that the first entry of $|w^-\rangle$ is equal to 1. The requirement that $|w^-\rangle \perp \mathsf{A}'$ implies that the remaining n entries of $|w^-\rangle$, which we denote $|w\rangle$, must satisfy

$$\mathsf{A}^T|w\rangle = -|r\rangle. \tag{4.7}$$

The negative witness size of the high-level span program is then $\||w^-\rangle\|^2 = 1 + \||w\rangle\|^2$.

To compute the norm of $|w\rangle$, we use the fact shown by Hutchinson [Hut90] that for any nonsingular symmetric matrix B and vector $|r\rangle$ with entries drawn from the Rademacher distribution,

$$\mathbb{E}\big[\langle r|\mathsf{B}|r\rangle\big] = \mathrm{Tr}(\mathsf{B}).$$

Therefore,

$$\begin{aligned}\mathbb{E}\Big[\||w\rangle\|^2\Big] &= \mathbb{E}\Big[\langle r|(\mathsf{A}^{-1})(\mathsf{A}^{-1})^T|r\rangle\Big] \\ &= \mathrm{Tr}\Big((\mathsf{A}^{-1})(\mathsf{A}^{-1})^T\Big) \\ &= \|\mathsf{A}^{-1}\|_F \\ &= c(\mathsf{A})^2 n.\end{aligned}$$

where $c(\mathsf{A})$ is defined as in equation (4.1). We can then apply Markov's inequality to show that $\||w^-\rangle\|^2 \le 100c(\mathsf{A})^2 n + 1$ with probability at least 99/100. □

By Lemma 17, the expected correctness and witness size of our span program differ depending on whether A is singular or not. If A is singular, then with probability at least 3/16, the program is correct and the witness size is bounded by a constant. On the other hand, if A is nonsingular, then with the much higher probability of at least 99/100, the program is correct and the witness size is $O(c(\mathsf{A})^2 n)$. To obtain a span program whose instantiation introduces an error probability of at most 1/6 for any input, we can balance these cases against each other by making the following modification to our span program.

Rather than using the columns of A' as the available vectors for the span program, we use the columns of the extended matrix $\mathsf{A'}_{\text{ext}}$, which has size $(9n+1) \times 9n$. The entries in the first row of $\mathsf{A'}_{\text{ext}}$ are chosen uniformly at random from $\{1, -1\}$. For notational convenience, we divide these entries into 9 blocks of length n, which we denote $\langle r_1| \ldots \langle r_9|$. The remaining $9n \times 9n$ matrix is block diagonal, consisting of 9 copies of A as shown.

$$\mathsf{A'}_{\text{ext}} = \begin{bmatrix} \langle r_1| & \langle r_2| & \ldots & & \langle r_9| \\ \mathsf{A} & 0 & \ldots & & 0 \\ 0 & \mathsf{A} & 0 & \ldots & \vdots \\ \vdots & 0 & \ddots & & \\ & & & & 0 \\ 0 & & & 0 & \mathsf{A} \end{bmatrix}. \tag{4.8}$$

The target vector $|\tau'_{\text{ext}}\rangle$ for the span program is defined similarly to $|\tau'\rangle$, where the first entry is 1 and the remaining $9n$ entries are all 0.

Modifying the high level span program in this way does not affect the asymptotic loading cost, but it affects the error probabilities. If A is singular, then $|\tau'_{\text{ext}}\rangle$ will lie in the span of the columns of $\mathsf{A'}_{\text{ext}}$ and the positive witness size will be less than one as long as at least one of the $\langle r_i|$ satisfies the properties required for a "good" instantiation. By Lemma 17, this will occur with probability $1-(\frac{13}{16})^9 > \frac{5}{6}$. If A is nonsingular, then any negative witness must be orthogonal to all columns of A'_{ext}. The negative witness size is therefore

$$\||w^-\rangle\|^2 = 1 + \||w_1\rangle\|^2 + \||w_2\rangle\|^2 + \ldots + \||w_9\rangle\|^2,$$

where the $|w_i\rangle$ are defined to satisfy $\mathsf{A}^T|w_i\rangle = -|r_i\rangle$. Therefore, the negative witness size of the modified span program will be less than $900c(\mathsf{A})^2 n + 1$ as long as all of the $\langle r_i|$ satisfy the properties required for a "good" instantiation. By Lemma 17, this occurs with probability at least $(\frac{99}{100})^9 > \frac{5}{6}$. Thus, by using the columns of A'_{ext} in our span program instead of the columns of A', we obtain the desired bound for the error stemming from the random instantiation of our span program.

Let T be defined such that for any input matrix A with full rank, $c(\mathsf{A}) \leq T$. To perform the algorithm, one would then run our modified high level span program with the bound

$$30T\sqrt{n} + 1 \geq \sqrt{900T^2 n + 1} \geq \text{wsize}(\mathscr{P}).$$

If the error probability of the span program-solving algorithm is at most 1/6, then our algorithm outputs the correct answer with probability at least 2/3 in both the positive and negative cases. Thus, we obtain the following theorem, which is analogous to Belovs' result in Theorem 6.

Theorem 3 *Our span program algorithm solves the* 0*-determinant verification problem with two-sided bounded error in $O(\sqrt{n}LT)$ quantum queries, under the promise that any input matrix has entries with norm at most one, and any input matrix with full rank satisfies $c(\mathsf{A}) \leq T$. Here, L represents the cost of loading a matrix into a span program.*

Finally, we show that there are families of matrices for which $c(\mathsf{A})$ grows exponentially with n, even when restricted to the case of binary matrices.

Lemma 18 *Consider the family of matrices $\{\mathsf{M}_n\}$, where M_n is the $n \times n$ upper antitriangular matrix for which the main antidiagonal and the first and third an-*

tidiagonals above it contain all ones, and all other entries are zero. For example,

$$\mathsf{M}_6 = \begin{bmatrix} 0 & 0 & 1 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 & 1 & 0 \\ 1 & 0 & 1 & 1 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}. \tag{4.9}$$

Then $c(\mathsf{M}_n) \in \omega(1.4^n)$.

Proof To prove this statement, we consider the (n,n) entry of the matrix M_n^{-1}. We begin by noting that because M_n is upper antitriangular, it has determinant ± 1. Therefore, the (n,n) entry of M_n^{-1} has norm equal to that of the $(n,n)^{\text{th}}$ minor of M_n. This minor is given by $\det(\mathsf{R}_{n-1})$, where R_{n-1} is defined to be the $(n-1)\times(n-1)$ matrix obtained by deleting the n^{th} row and the n^{th} column from M_n.

To illustrate, the norm of the $(n,n)^{th}$ entry of M_6^{-1} is norm of the determinant of R_5, where

$$\mathsf{R}_5 = \begin{bmatrix} 0 & 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & 1 & 1 \\ 1 & 0 & 1 & 1 & 0 \\ 0 & 1 & 1 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 \end{bmatrix}.$$

By inspection, we can see that the determinant of the matrix R_k can be expressed by the recurrence

$$\det(\mathsf{R}_k) = \det(\mathsf{R}_{k-1}) + \det(\mathsf{R}_{k-3}).$$

The base cases for this recurrence are given by $\det(\mathsf{R}_0) = 1$, $\det(\mathsf{R}_1) = 1$ and $\det(\mathsf{R}_2) = 1$. Thus, the norm of the $(n,n)^{\text{th}}$ entry of M_n^{-1} is given by the n^{th} term in Narayana's cows sequence[1].

[1]In fact, the antidiagonals of the matrix M_n^{-1} are elegantly given by the first n terms in this sequence, with alternating signs.

Narayana's cows sequence grows asympotically as $\Theta(c^n)$, where $c \approx 1.4656$ is the real root of $x^3 - x^2 - 1$. Therefore, the largest entry of M_n^{-1} is also $\Theta(c^n)$. Using the definition of $c(\mathsf{A})$ from equation (4.4), we conclude that

$$c(\mathsf{M}_n) = \frac{\|\mathsf{M}_n\|_F}{\sqrt{n}} \in \Omega(\frac{c^n}{\sqrt{n}}) \in \omega(1.4^n).$$

□

Lemma 18 shows that the algorithms in Theorems 3 and 6 are not optimal in terms of query complexity, as for certain classes of matrix the query complexity would exceed the trivial upper bound of $O(n^2)$. However, we note the possibility that our span-program-based algorithm may still be optimal in terms of quantum time complexity. This is based on the observation that the matrices for which our algorithm has exponential query complexity have exponentially small eigenvalues, and are therefore intuitively hard cases for 0-determinant verification.

4.4 Algorithm for perfect matching

We apply our span program from Section 4.3 to detect whether a graph G, given through black-box access to an adjacency matrix, contains a perfect matching. Our algorithm is based on Tutte's matrix-based characterization of perfect matchings, and is thus a quantum analogue of the classical randomized algorithm by Lovász [Lov79].

There are two steps to our algorithm. First, we instantiate the formal variables in the Tutte matrix $\mathsf{T}(G)$ uniformly at random from a sample space S. To do this, one would instantiate the Tutte matrix for the complete graph on n vertices, which would be stored in quantum memory. The second step in our algorithm is to apply our span program for 0-determinant verification as a subroutine to test whether the instantiated Tutte matrix is nonsingular. Each query to the Tutte matrix for the input graph would consist of querying both the adjacency matrix and the stored Tutte matrix, and then multiplying the query outcomes. If our span program decides the Tutte matrix for the input graph is nonsingular, we can conclude that G has a perfect matching.

This method gives the following result.

Theorem 4 *Given a graph G on n vertices, a bound R, and a set S whose elements have norm at most one, our algorithm can decide whether G contains a perfect matching with two-sided bounded error in $O(n^{3/2}R)$ quantum queries, under the following condition. If G contains a perfect matching, then the Tutte matrix $\mathsf{T}(G)$ must be nonsingular with constant probability when instantiated uniformly at random from S, and given this is the case, $c(\mathsf{T}(G)) \leq R$ with constant probability.*

For the instantiation step, we intentionally leave some freedom in the choice of the set S. In [Lov79], Lovász chooses the set $S = \{1, 2, 3, \ldots, n^2\}$, which by Lemma 14 means that the probability that a nonsingular $\mathsf{T}(G)$ becomes singular when instantiated is at most $\frac{1}{n}$. The elements in the set S can be rescaled to have norm at most one without changing this result. However, this may not be the optimal choice of sample space to minimize R.

To account for this, we refer to Lemma 15, which shows that any set S with $|S| \geq 12$ and $0 \notin S$ can be used to obtain a one-sided bounded error of at most $\frac{1}{2}$ during the instantiation of the Tutte matrix. This allows more flexibility when choosing a set S for a particular class of input graphs, with the goal of minimizing the quantity R.

Given a restricted domain of graphs, if there exists a set S such that $R \in O(1)$, then our algorithm achieves a query complexity of $O(n^{3/2})$. However, there may also be families of graphs for which R must grow exponentially with n, regardless of the choice of S. To see how such graphs could arise, we consider the family of matrices $\{\mathsf{M}_n\}$ defined in Lemma 18, for which the smallest eigenvalue decreases exponentially in n.

Let M_n be the biadjacency matrix of a bipartite graph G_n on $2n$ vertices. We define $\mathsf{M}_n(S)$ to be the matrix whose (i, j) entry is zero if the (i, j) entry of M_n is zero, and whose entries are otherwise instantiated uniformly at random from the set S. Then for each instantiation of $\mathsf{M}_n(S)$, there is a corresponding instantiation of the Tutte matrix $\mathsf{T}(G_n)$, which is given in block form by

$$\mathsf{T}(G_n) = \begin{bmatrix} 0 & \mathsf{M}_n(S) \\ -\mathsf{M}_n(S) & 0 \end{bmatrix}.$$

If λ is an eigenvalue of $\mathsf{M}_n(S)$, then $\pm\imath\lambda$ are eigenvalues of $\mathsf{T}(G_n)$. Therefore, $c(\mathsf{T}(G_n)) = c(\mathsf{M}_n(S))$. We know from Lemma 18 that when $\mathsf{M}_n(S)$ is instantiated from the set $S = \{1\}$, we have $c(\mathsf{M}_n(S)) \in \omega(1.4^n)$. Therefore, our algorithm from Theorem 4 does not run in polynomial time for $S = \{1\}$. We suspect this holds for any choice of S, namely that the class of graphs constructed in this way from the matrices M_n require exponentially large R, under the condition that the elements in S are bounded in norm by one.

This approach leads to a new class of interesting graphs for perfect matchings, namely graphs whose adjacency matrices have exponentially small eigenvalues, such as the family G_n of bipartite graphs described above. These graphs seem to be inherently resistant to our algebraic approach, and their study may lead to new insights on the perfect matchings problem.

4.4.1 Full span program for perfect matching

In this section, we show explicitly how one can construct a randomized span program to solve the perfect matching detection problem. In Section 4.4, we give an algorithm to solve the perfect matching decision problem in two steps. First, the Tutte matrix of the graph is randomly instantiated from the set S, and then we apply our span program from Section 4.3 to the instantiated matrix.

Alternatively, one could directly apply a span program to solve the perfect matching detection problem. Such a span program is obtained by combining the two steps above. We give the resulting span program here and show its explicit construction for the graph in Figure 4.2. Although our span program for perfect matching detection is essentially the same as our quantum algorithm described earlier, we show it here for clarity. This has the added benefit of illustrating the matrix loading routine, which has allowed us to prove our witness size bounds using the high-level version of our span program for 0-determinant verification.

For this construction, we tailor Belovs' matrix loading routine to our application by using the fact that the Tutte matrix is antisymmetric and randomly instantiated. As a result, our specialized loading routine reduces the number of extra dimensions from n^2k, where k is the number of bits of precision used, to $2e = n(n-1)$. Our loading incurs the same cost in query complexity as Belovs' routine, introducing a multiplicative factor of $L \in \Theta(n)$ to the witness size.

In our span program for perfect matching detection, the vectors have length $d = 1 + n + 2e$. Our target vector is

$$|\tau\rangle = \begin{bmatrix} 1 \\ 0 \\ 0 \\ \vdots \\ 0 \end{bmatrix}. \tag{4.10}$$

Let K_n denote the complete graph on n vertices. Our span program has n free vectors, which are derived from the columns of $\mathsf{T}'(K_n)$. The first $n+1$ entries of the k^{th} free vector are given by the k^{th} column of $\mathsf{T}'(K_n)$. Let the remaining $2e$ entries be indexed by ordered pairs (u,v), for $1 \leq u, v \leq n$ and $u \neq v$. These entries correspond to the e input bits, acting as flags to indicate which of the input bits influence a particular column. More precisely, for the k^{th} free vector, the bits corresponding to (u,k) for all $u \neq k$ are equal to 1. The sets $V_{i,b}$ of input vectors are defined as follows.

If $b = 0$, meaning edge $i = (u,v)$, $u < v$, is not in G, then $V_{i,b}$ contains two vectors. In the first vector, $\mathsf{T}(K_n)_{u,v}$ is at index v, the flag at position (u,v) is set to 1, and the remaining entries are 0. Similarly, in the second vector, $\mathsf{T}(K_n)_{v,u}$ is at index u, the flag at position (v,u) is set to 1, and the remaining entries are 0. Recall that $\mathsf{T}(K_n)_{v,u} = -\mathsf{T}(K_n)_{u,v}$, and that $\mathsf{T}(K_n)_{u,v}$ is instantiated uniformly from set S.

If $b = 1$, then edge $i = (u,v)$ is in G. In this case, $V_{i,b}$ contains two different vectors. Both vectors have all entries equal to 0, except for a single flag set to 1. In the first vector, the flag is set at position (u,v), and in the second vector, the flag is set at position (v,u).

The reason for this construction is that if edge (u,v) is not in G, the variable x_{uv} is not present in $\mathsf{T}(G)$, so we want to enforce $\mathsf{T}(G)_{u,v} = \mathsf{T}(G)_{v,u} = 0$. By our choice of target, the (u,v) and (v,u) flags must be 0 in $|\tau\rangle$. Therefore, if the k^{th} free vector is used to reach the target, then the input vectors with flags (u,k) must be used to cancel out the flags. If the edge (u,v) is not in G, then using the vectors

$$V_{free} = \begin{bmatrix} \pm1 & \pm1 & \pm1 & \pm1 \\ 0 & x_{12} & x_{13} & x_{14} \\ -x_{12} & 0 & x_{23} & x_{24} \\ -x_{13} & -x_{23} & 0 & x_{34} \\ -x_{14} & -x_{24} & -x_{34} & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \qquad V_{i,0} = \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ x_{23} & 0 \\ 0 & -x_{23} \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 1 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 1 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \end{bmatrix} \qquad V_{i,1} = \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 1 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 1 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \end{bmatrix}$$

Figure 4.3: Vectors in our span program for detecting perfect matchings when $n = 4$, as in our graph from Figure 4.2. The free vectors and the input vectors corresponding to the edge $i = (2,3)$ are shown. The columns of $V_{i,0}$ would be available if the edge $(2,3)$ was not in G, otherwise the columns of $V_{i,1}$ would be available. The relevant flags are highlighted in red.

in $V_{i,0}$ prevents the corresponding values $\mathsf{T}(G)_{u,v}$ and $\mathsf{T}(G)_{v,u}$ from contributing to reaching the target $|\tau\rangle$. On the other hand, if edge (u,v) is in G, then the vectors in $V_{i,1}$ must be used to cancel out the two flags for the corresponding edge, and cannot affect other entries in $\mathsf{T}(K_n)$.

It is then straightforward to check how the witness size of the resulting span program relates to the witness size of the original high level span program. If $|w_h^+\rangle$ is a positive witness for the high level span program, then the size of the corresponding positive witness for the full span program will be $\||w^+\rangle\|^2 = (n-1)\||w_h^+\rangle\|^2$. If $|w_h^-\rangle$ is a negative witness for the high level span program, then the size of the corresponding negative witness $|w^-\rangle$ for the full span program will be

$$\|V^\dagger|w^-\rangle\|^2 = \sum_{1\leq i<j\leq n} (x_{ij}w_i)^2 + (x_{ij}w_j)^2 \leq (n-1)\||w_h^-\rangle\|^2,$$

using the fact that $|x_{ij}| \leq 1$ by the restriction on S.

Therefore, the total witness size of the loaded span program is equal to that of the high level span program, with an extra multiplicative factor of $L \in \Theta(n)$ as claimed.

4.5 Conclusion

Tutte's characterization allows the problem of detecting perfect matchings to be reduced to the problem of deciding whether a given matrix is singular. We apply this to give an algebraic quantum algorithm for deciding whether a graph contains a perfect matching.

As part of our algorithm for detecting matchings, we give a new span program to solve 0-determinant verification. Our span program simplifies aspects of Belovs' existing span program for the rank problem, considered in the restricted setting of 0-determinant verification, and achieves the same asymptotic witness size. It also serves as a new example of how both randomness and matrix loading can be used advantageously in designing span programs.

When we apply this span program to solve the perfect matching detection problem, we obtain an algorithm whose query complexity depends on both the sample space S used for the random instantiation and the class of allowed graphs. For certain classes of graphs, this may lead to an improvement over the $O(n^{7/4})$ query complexity of the current best augmenting-paths-based algorithms. We also show how to find classes of graphs for which our algorithm could have exponential query complexity. These graphs have not yet been considered when studying perfect matchings, and may therefore provide new insights into the problem. To close the remaining gap with Zhang's lower bound of $\Omega(n^{3/2})$, new ideas will be needed, indicating that finding matchings will continue to be a fruitful problem in algorithm design.

Chapter 5

Concluding remarks

In this thesis, we give two new quantum algorithms. In doing so, we provide new insights on two canonical problems in computer science.

Our first algorithm is a new type of memoryless walk, which we obtain by adding selfloops to marked vertices. We prove that this modification gives an algorithm for spatial search that matches the best-known algorithms in space, query, and step complexity. An unusual and interesting feature of our proof is the precision with which we analyse the effect of the selfloop on the dynamics of our walk. We are able to show that with our chosen value of selfloop weight, the action of the walk is asymptotically forced into a single two-dimensional rotational subspace. This property is what allows our spatial search algorithm to find the marked vertex with success probability asymptotically close to one.

Our algorithm finds a unique marked vertex on the grid. This suggests two natural extensions. The first extension is modifying our walk so that it can find a marked vertex when multiple vertices are marked. We discuss this possibility in [HL22], where we propose an idea for how this case could be handled.

The second extension is applying our memoryless walk to solve spatial search on other types of graph. In our work, we analysed the effect of adding a selfloop to a walk on the two-dimensional grid. This choice was made because the structure of the grid is simple enough that we could obtain a complete description of the spectrum of our input-independent walk operator. This description is crucial to our analysis. However, we believe that selfloops could be applied to obtain optimal memoryless search algorithms on other types of graphs as well. In particular, it may be possible to prove bounds on the success probability and step complexity for memoryless search with selfloops on other types of graphs with sufficient structure, such as the hexagonal lattice used in [CPBS18].

Our second algorithm gives new perspective on the query complexity of detecting perfect matchings—a problem for which the tight asymptotic bound is not yet known. To investigate this gap, we give a new quantum algorithm based on an algebraic characterization by Tutte. To our knowledge, this is the first time this approach has been used in the quantum setting.

As a key component in our approach, we give a new quantum algorithm for 0-determinant verification, which we use as a subroutine in our algorithm for perfect matchings. Our algorithm for 0-determinant verification is defined using a span program, a construction in which the query complexity is given by the span program's witness size. Our span program for 0-determinant verification matches a previous span program by Belovs in witness size, while also having a simple structure and clean analysis.

An unusual aspect of our span program is that it incorporates randomization into the span program definition. Our span program vectors have entries sampled from a probability distribution, and we show that with constant probability, this results in a span program that is correct and has a small witness size. Our span program and Belovs' span program for rank-finding both use randomization, but there are few other such examples. Our span program is therefore a new example of how randomization can be usefully incorporated into span programs, which in turn could be a useful tool in the development of new quantum algorithms.

We note that the query complexity of our algorithm for 0-determinant verification depends on the spectrum of the input matrix. If an $n \times n$ input matrix has small eigenvalues, the query complexity of our algorithm becomes very large. As an example of this, we give an explicit matrix with binary entries for which the query complexity is exponential in n. For such matrices, this seems to contradict the fact that all entries in the input matrix could be accessed in only $O(n^2)$ queries. However, this can be accounted for by the possibility that matrix

0-determinant verification is a problem for which there exists an asymptotic separation between the quantum query complexity, which is trivially $O(n^2)$, and the quantum step complexity, which may be exponential.

Using the hard-case matrices we identify for 0-determinant verification, we show how to construct a family of hard-case graphs for our algebraic perfect matching detection algorithm. These graphs are bipartite, with biadjacency matrices that have exponentially small eigenvalues. Because of their spectral properties, these graphs seem to be fundamentally resistant to the algebraic approach. Our algorithm appears to have exponential query complexity on these inputs, even accounting for adjustable parameters in the algorithm. These graphs have not yet been considered in the context of perfect matchings, so their study may lead to new insights on the problem. In particular, it would be interesting to compare the hard-case inputs for our algorithm with hard-case inputs for alternative approaches to detecting perfect matchings. The difference between the best-known upper and lower bounds for finding and detecting perfect matchings remains a tantalizing problem for future work.

Bibliography

[AA05] Scott Aaronson and Andris Ambainis. Quantum search of spatial regions. *Theory of Computing*, 1(4):47–79, 2005. arXiv:quant-ph/0303041, doi:10.4086/toc.2005.v001a004.

[AAKV01] Dorit Aharonov, Andris Ambainis, Julia Kempe, and Umesh Vazirani. Quantum walks on graphs. In *Proceedings of the 33rd Annual ACM Symposium on Theory of Computing*, STOC'01, pages 50–59, 2001. arXiv:quant-ph/0012090, doi:10.1145/380752.380758.

[AAMP20] Frank Acasiete, F. P. Agostini, Jalil Khatibi Moqadam, and Renato Portugal. Implementation of quantum walks on IBM quantum computers. *Quantum Information Processing*, 19, December 2020. arXiv:2002.01905, doi:10.1007/s11128-020-02938-5.

[ABN+01] Andris Ambainis, Eric Bach, Ashwin Nayak, Ashvin Vishwanath, and John Watrous. One-dimensional quantum walks. STOC'01, pages 37—-49. Association for Computing Machinery (ACM), 2001. doi:10.1145/380752.380757.

[ABN+12] Andris Ambainis, Artūrs Bačkurs, Nikolajs Nahimovs, Raitis Ozols, and Alexander Rivosh. Search by quantum walks on two-dimensional grid without amplitude amplification. In *7th Conference on the Theory of Quantum Computation, Communication and Cryptography*, TQC'12, pages 87–97, 2012. arXiv:1112.3337, doi:10.1007/978-3-642-35656-8_7.

[ACNR21] Simon Apers, Shantanav Chakraborty, Leonardo Novo, and Jérémie Roland. Quadratic speedup for spatial search by continuous-time quantum walk, December 2021. `arXiv:2112.12746`.

[AGJ21] Simon Apers, András Gilyén, and Stacey Jeffery. A unified framework of quantum walk search. In *Proceedings of the 38th Symposium on Theoretical Aspects of Computer Science*, STACS'19. Leibniz International Proceedings in Informatics (LIPIcs), 2021. `arXiv:1912.04233`.

[AGJK20] Andris Ambainis, András Gilyén, Stacey Jeffery, and Mārtiņš Kokainis. Quadratic speedup for finding marked vertices by quantum walks. In *Proceedings of the 52nd Annual ACM Symposium on Theory of Computing*, pages 412–424. Association for Computing Machinery (ACM), 2020. `arXiv:1903.07493`, `doi:10.1145/3357713.3384252`.

[AKL$^+$79] Romas Aleliunas, Richard M. Karp, Richard J. Lipton, László Lovász, and Charles Rackoff. Random walks, universal traversal sequences, and the complexity of maze problems. In *Proceedings of the 20th IEEE Symposium on Foundations of Computer Science*, pages 218–223, 1979. `doi:10.1109/SFCS.1979.34`.

[AKR05] Andris Ambainis, Julia Kempe, and Alexander Rivosh. Coins make quantum walks faster. In *Proceedings of the 16th Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA'05, pages 1099–1108, 2005. `arXiv:quant-ph/0402107`.

[Amb07] Andris Ambainis. Quantum walk algorithm for element distinctness. *SIAM Journal on Computing*, 37(1):210–239, 2007. `arXiv:quant-ph/0311001`, `doi:10.1137/S0097539705447311`.

[APN15] Andris Ambainis, Renato Portugal, and Nikolajs Nahimovs. Spatial search on grids with minimum memory. *Quantum Information & Computation*, 15(13–14):1233–1247, 2015. `arXiv:1312.0172`.

[Āri16] Agnis Āriņš. Span-program-based quantum algorithms for graph bipartiteness and connectivity. In *Mathematical and Engineering Methods in Computer Science, MEMICS'15*, volume 9548 of *Lecture Notes in Computer Science*, pages 35–41, March 2016. `arXiv:1510.07825`, `doi:10.1007/978-3-319-29817-7_4`.

[AŠ06] Andris Ambainis and Robert Špalek. Quantum algorithms for matching and network flows. In *Proceedings of the 23rd Symposium on Theoretical Aspects of Computer Science*, STACS'06, pages 172–183, 2006. `arXiv:quant-ph/0508205`, `doi:10.1007/11672142_13`.

[Bel11] Aleksandrs Belovs. Span-program-based quantum algorithm for the rank problem, 2011. `arXiv:1103.0842`.

[Ben02] Paul Benioff. Space searches with a quantum robot. In *Quantum Computation and Information*, volume 305 of *AMS Contemporary Mathematics*, pages 1–12. American Mathematical Society, October 2002. `arXiv:quant-ph/0003006`.

[BHMT02] Gilles Brassard, Peter Høyer, Michele Mosca, and Alain Tapp. Quantum amplitude amplification and estimation. In *Quantum Computation and Information*, volume 305 of *AMS Contemporary Mathematics*, pages 53–74. American Mathematical Society, 2002. `arXiv:quant-ph/0005055`.

[BR12] Aleksandrs Belovs and Ben W. Reichardt. Span programs and quantum algorithms for st-connectivity and claw detection. In *Proceedings of the 20th Annual European Conference on Algorithms*, ESA'12, pages 193–204, 2012. `arXiv:1203.2603`, `doi:10.1007/978-3-642-33090-2_18`.

[BT19] Salman Beigi and Leila Taghavi. Span program for non-binary functions. *Quantum Information & Computation*, 19:760–792, August 2019. `arXiv:1805.02714`.

[CCD+03] Andrew M. Childs, Richard Cleve, Enrico Deotto, Edward H. Farhi, Samuel Gutmann, and Daniel A. Spielman. Exponential algorithmic speedup by a quantum walk. In *Proceedings of the 35th Annual ACM Symposium on Theory of Computing*, STOC'03, pages 59—68. Association for Computing Machinery (ACM), 2003. `doi:10.1145/780542.780552`.

[CFG02] Andrew M. Childs, Edward H. Farhi, and Samuel Gutmann. An example of the difference between quantum and classical random walks. *Annual Conference on Quantum Information Processing*, 1:35–43, 2002. `arXiv:quant-ph/0103020`, `doi:10.1023/A:1019609420309`.

[CG04] Andrew M. Childs and Jeffrey Goldstone. Spatial search by quantum walk. *Physical Review A: General Physics*, 70(2):022314, 2004. `arXiv:quant-ph/0306054`, `doi:10.1103/PhysRevA.70.022314`.

[Chi09] Andrew M. Childs. Universal computation by quantum walk. *Physical Review Letters*, 102(18):180501, 2009. `doi:10.1103/physrevlett.102.180501`.

[CP18] Gabriel Coutinho and Renato Portugal. Discretization of continuous-time quantum walks via the staggered model with hamiltonians. *Natural Computing*, 18:403–409, 2018. `arXiv:1701.03423`, `doi:10.1007/s11047-018-9688-8`.

[CPBS18] Bruno Chagas, Renato Portugal, Stefan Boettcher, and Etsuo Segawa. Staggered quantum walk on hexagonal lattices. *Physical Review A: General Physics*, 98:052310, November 2018. `doi:10.1103/PhysRevA.98.052310`.

[DH17] Cătălin Dohotaru and Peter Høyer. Controlled quantum amplification. In *Proceedings of the 44th International Colloquium on Automata, Languages, and Programming*, volume 80 of *ICALP'17*, pages 18:1–18:13, Dagstuhl, Germany, July 2017. Schloss Dagstuhl–Leibniz-Zentrum für Informatik. `doi:10.4230/LIPIcs.ICALP.2017.18`.

[DHHM06] Christoph Dürr, Mark Heiligman, Peter Høyer, and Mehdi Mhalla. Quantum query complexity of some graph problems. *SIAM Journal on Computing*, 35(6):1310–1328, 2006. `arXiv:quant-ph/0401091`, `doi:10.1137/050644719`.

[DK21] Vojtěch Dvořák and Ohad Klein. Probability mass of Rademacher sums beyond one standard deviation, 2021. `arXiv:2104.10005`.

[DL78] Richard A. Demillo and Richard J. Lipton. A probabilistic remark on algebraic program testing. *Information Processing Letters*, 7(4):193–195, 1978. `doi:10.1016/0020-0190(78)90067-4`.

[Dör09] Sebastian Dörn. Quantum algorithms for matching problems. *Theory of Computing Systems*, 24:613–628, October 2009. `doi:10.1007/s00224-008-9118-x`.

[DT09] Sebastian Dörn and Thomas Thierauf. The quantum query complexity of the determinant. *Information Processing Letters*, 109(6):325–328, 2009. `doi:10.1016/j.ipl.2008.11.006`.

[Edm65] Jack Edmonds. Paths, trees, and flowers. *Canadian Journal of Mathematics*, 17:449–467, 1965. `doi:10.4153/CJM-1965-045-4`.

[EV93] Avshalom C. Elitzur and Lev Vaidman. Quantum mechanical interaction-free measurements. *Foundations of Physics*, 23:987–997, May 1993. `arXiv:hep-th/9305002`, `doi:10.1007/BF00736012`.

[Fal13] Matthew D. Falk. Quantum search on the spatial grid, March 2013. `arXiv:1303.4127`.

[FG98] Edward H. Farhi and Samuel Gutmann. Quantum computation and decision trees. *Physical Review A: General Physics*, 58:915–928, August 1998. `doi:10.1103/PhysRevA.58.915`.

[Gro96] Lov K. Grover. A fast quantum mechanical algorithm for database search. In *Proceedings of the 28th Annual ACM Symposium on Theory* *of Computing*, STOC'96, pages 212–219, 1996. arXiv:quant-ph/9605043, doi:10.1145/237814.237866.

[HK73] John E. Hopcroft and Richard M. Karp. An $n^{5/2}$ algorithm for maximum matchings in bipartite graphs. *SIAM Journal on Computing*, 2(4):225–231, 1973. doi:10.1137/0202019.

[HL22] Peter Høyer and Janet Leahy. Spatial search via an interpolated memoryless walk. *Physical Review A: General Physics*, 106:022418, August 2022. arXiv:2204.09076, doi:10.1103/PhysRevA.106.022418.

[HLŠ07] Peter Høyer, Troy Lee, and Robert Špalek. Negative weights make adversaries stronger. In *Proceedings of the 39th Annual ACM Symposium on Theory of Computing*, STOC'07, pages 526–535, 2007. arXiv:quant-ph/0611054, doi:10.1145/1250790.1250867.

[Hut90] Michael F. Hutchinson. A stochastic estimator of the trace of the influence matrix for laplacian smoothing splines. *Communications in Statistics - Simulation and Computation*, 19(2):433–450, 1990. doi:10.1080/03610919008812866.

[JK17] Stacey Jeffery and Shelby Kimmel. Quantum algorithms for graph connectivity and formula evaluation. *Quantum*, 1:26, April 2017. arXiv:1704.00765, doi:10.22331/q-2017-08-17-26.

[KMOR16] Hari Krovi, Frédéric Magniez, Māris Ozols, and Jérémie Roland. Quantum walks can find a marked element on any graph. *Algorithmica*, 74(2):851–907, 2016. arXiv:1002.2419, doi:10.1007/s00453-015-9979-8.

[KPSS18] Norio Konno, Renato Portugal, Iwao Sato, and Etsuo Segawa. Partition-based discrete-time quantum walks. *Annual Conference on Quantum Information Processing*, 17:1–35, 2018. arXiv:1707.07127, doi:10.1007/s11128-017-1807-4.

[KW93] Mauricio Karchmer and Avi Wigderson. On span programs. In *Proceedings of the 8th Annual IEEE Symposium on Structure in Complexity*, SSC'93, pages 102–111, 1993. doi:10.1109/SCT.1993.336536.

[KW21] Shelby Kimmel and R. Teal Witter. A query-efficient quantum algorithm for maximum matching on general graphs. In *Workshop on Algorithms and Data Structures*, Lecture Notes in Computer Science, pages 543–555. Springer Berlin Heidelberg, 2021. arXiv:2010.02324, doi:10.1007/978-3-030-83508-8_39.

[LL16] Cedric Yen-Yu Lin and Han-Hsuan Lin. Upper bounds on quantum query complexity inspired by the Elitzur-Vaidman bomb tester. *Theory of Computing*, 12:1–35, 2016. URL: https://dl.acm.org/doi/10.5555/2833227.2833254, arXiv:1410.0932.

[Lov79] László Lovász. On determinants, matchings and random algorithms. In *Proceedings of the 2nd Symposium on Fundamentals of Computation Theory*, volume 79, pages 565–574, January 1979.

[MdOP17] Jalil Khatibi Moqadam, Marcos César de Oliveira, and Renato Portugal. Staggered quantum walks with superconducting microwave resonators. *Physical Review B: Solid State*, 95, 2017. arXiv:1609.09844, doi:10.1103/PhysRevB.95.144506.

[MS04] Marcin Mucha and Piotr Sankowski. Maximum matchings via Gaussian elimination. In *Proceedings of the 45th IEEE Symposium on Foundations of Computer Science*, pages 248–255, 2004. doi:10.1109/FOCS.2004.40.

[MSS07] Frédéric Magniez, Miklos Santha, and Mario Szegedy. Quantum algorithms for the triangle problem. *SIAM Journal on Computing*, 37(2):413–424, 2007. arXiv:quant-ph/0310134, doi:10.1137/050643684.

[MV80] Silvio Micali and Vijay V. Vazirani. An $O(\sqrt{|V|}|E|)$ algorithm for finding maximum matching in general graphs. In *Proceedings of the 21st IEEE Symposium on Foundations of Computer Science*, SFCS '80, pages 17–27, 1980. `doi:10.1109/SFCS.1980.12`.

[NC00] Michael Nielsen and Isaac L. Chuang. *Quantum Computation and Quantum Information*. Cambridge University Press, 2000.

[Ole96] Krzysztof Oleszkiewicz. On the Stein property of Rademacher sequences. *Probability and Mathematical Statistics*, 16(1):127–130, 1996.

[PBF15] Renato Portugal, Stefan Boettcher, and Stefan Falkner. One-dimensional coinless quantum walks. *Physical Review A: General Physics*, 91, May 2015. `arXiv:1408.5166`, `doi:10.1103/PhysRevA.91.052319`.

[PdOM17] Renato Portugal, Marcos César de Oliveira, and Jalil Khatibi Moqadam. Staggered quantum walks with hamiltonians. *Physical Review A: General Physics*, 95, January 2017. `arXiv:1605.02774`, `doi:10.1103/PhysRevA.95.012328`.

[PF17] Renato Portugal and Tharso D. Fernandes. Quantum search on the two-dimensional lattice using the staggered model with hamiltonians. *Physical Review A: General Physics*, 95, January 2017. `arXiv:1701.01468`, `doi:10.1103/PhysRevA.95.042341`.

[Por16a] Renato Portugal. Establishing the equivalence between Szegedy's and coined quantum walks using the staggered model. *Quantum Information Processing*, 15(4):1387–1409, April 2016. `arXiv:1509.08852`, `doi:10.1007/s11128-015-1230-7`.

[Por16b] Renato Portugal. Staggered quantum walks on graphs. *Physical Review A: General Physics*, 93, June 2016. `arXiv:1603.02210`, `doi:10.1103/PhysRevA.93.062335`.

[PRR05] Apoorva Patel, K. S. Raghunathan, and Pranaw Rungta. Quantum random walks do not need a coin toss. *Physical Review A: General Physics*, 71, March 2005. arXiv:quant-ph/0405128, doi:10.1103/PhysRevA.71.032347.

[PRR10] Apoorva Patel, K. S. Raghunathan, and Md. Aminoor Rahaman. Search on a hypercubic lattice using a quantum random walk. ii. $d = 2$. *Physical Review A: General Physics*, 82, September 2010. arXiv:1003.5564, doi:10.1103/PhysRevA.82.032331.

[PSFG16] Renato Portugal, Raqueline A. M. Santos, Tharso D. Fernandes, and Demerson N. Gonçalves. The staggered quantum walk model. *Quantum Information Processing*, 15(1):85–101, 2016. arXiv:1505.04761, doi:10.1007/s11128-015-1149-z.

[Rei09] Ben W. Reichardt. Span programs and quantum query complexity: The general adversary bound is nearly tight for every boolean function. In *Proceedings of the 50th IEEE Symposium on Foundations of Computer Science*, FOCS'09, pages 544–551, 2009. arXiv:0904.2759, doi:10.1109/FOCS.2009.55.

[Rei11] Ben W. Reichardt. Reflections for quantum query algorithms. In *Proceedings of the 22nd Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA'11, pages 560–569, 2011. arXiv:1005.1601, doi:10.1137/1.9781611973082.44.

[RŠ12] Ben W. Reichardt and Robert Špalek. Span-program-based quantum algorithm for evaluating formulas. *Theory of Computing*, 8(13):291–319, 2012. arXiv:0710.2630, doi:10.4086/toc.2012.v008a013.

[RV89] Michael O. Rabin and Vijay V. Vazirani. Maximum matchings in general graphs through randomization. *Journal of Algorithms*, 10(4):557–567, 1989. doi:10.1016/0196-6774(89)90005-9.

[San08] Miklos Santha. Quantum walk based search algorithms. In *Theory and Applications of Models of Computation*, pages 31–46. Springer Berlin Heidelberg, 2008. `arXiv:0808.0059`.

[Sch80] Jacob T. Schwartz. Fast probabilistic algorithms for verification of polynomial identities. *Journal of the ACM*, 27(4):701–717, October 1980. `doi:10.1145/322217.322225`.

[Sho97] Peter W. Shor. Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer. *SIAM Journal on Computing*, 26(5):1484–1509, 1997. `arXiv:quant-ph/9508027`, `doi:10.1137/S0097539795293172`.

[SKW03] Neil Shenvi, Julia Kempe, and Birgitta K. Whaley. Quantum random-walk search algorithm. *Physical Review A: General Physics*, 67(5):052307, 2003. `arXiv:quant-ph/0210064`, `doi:10.1103/PhysRevA.67.052307`.

[Sze04] Mario Szegedy. Quantum speed-up of Markov chain based algorithms. In *Proceedings of the 45th IEEE Symposium on Foundations of Computer Science*, FOCS'04, pages 32–41, 2004. `doi:10.1109/FOCS.2004.53`.

[Tul08] Avatar Tulsi. Faster quantum-walk algorithm for the two-dimensional spatial search. *Physical Review A: General Physics*, 78(1):012310, July 2008. `arXiv:0801.0497`, `doi:10.1103/PhysRevA.78.012310`.

[Tut47] William T. Tutte. The factorization of linear graphs. *Journal of the London Mathematical Society*, s1-22(2):107–111, 1947. `doi:10.1112/jlms/s1-22.2.107`.

[Zha05] Shengyu Zhang. On the power of Ambainis lower bounds. *Theoretical Computer Science*, 339:241–256, 2005. `arXiv:quant-ph/0311060`, `doi:10.1016/j.tcs.2005.01.019`.

[Zip79] Richard Zippel. Probabilistic algorithms for sparse polynomials. In *Symbolic and Algebraic Computation*, Lecture Notes in Computer Science, pages 216–226. Springer Berlin Heidelberg, 1979. `doi:10.1007/3-540-09519-5_73`.

Appendices

Composition with a reflection

In this appendix, we discuss techniques to characterize the spectrum of the composition of a real operator with a one-dimensional reflection. Operators with this structure appear at multiple points in Chapter 3. In this section, we present two lemmas from the quantum walk literature that we apply in our analysis of both WF_1 and WF.

Consider an arbitrary real unitary operator T acting on a space $\mathscr{H}$, and let $|s\rangle \in \mathscr{H}$ be a state with real amplitudes. Define $\mathsf{S} = \mathsf{I} - |s\rangle\langle s|$ to be the reflection of state $|s\rangle$. The goal of this section is to describe the spectrum of the operator TS.

Because T is real-valued, its eigenvalues different from ± 1 come in complex conjugate pairs. We denote the eigenvalues as $e^{\pm \imath \phi_k}$ for $k = 1, 2, \ldots m$, corresponding to the eigenvectors $|T_k^{\pm}\rangle$. We then decompose $|s\rangle$ into the eigenbasis of T as

$$|s\rangle = s_0|T_0\rangle + \sum_k s_k\Big(|T_k^+\rangle + |T_k^-\rangle\Big) + s_{-1}|T_{-1}\rangle. \tag{1}$$

Here, $|T_0\rangle$ and $|T_{-1}\rangle$ are eigenvectors of T with eigenvalues $+1$ and -1, respectively. The coefficients s_0, s_{-1}, and all s_k are chosen to be non-negative real numbers by multiplying the eigenvectors with appropriate phases. This decomposition allows us to state the following lemma, originally given by [Amb07].

Lemma 19 *Consider the (unnormalized) state* $|e_\alpha\rangle = |s\rangle + \imath|e_\alpha{}^\perp\rangle$*, where*

$$|e_\alpha{}^\perp\rangle = s_0 \cot\left(\frac{\alpha}{2}\right)|T_0\rangle + \sum_k s_k\left[\cot\left(\frac{\alpha-\phi_k}{2}\right)|T_k^+\rangle + \cot\left(\frac{\alpha+\phi_k}{2}\right)|T_k^-\rangle\right] - s_{-1}\tan\left(\frac{\alpha}{2}\right)|T_{-1}\rangle, \tag{2}$$

and $|e_\alpha{}^\perp\rangle$ *is orthogonal to* $|s\rangle$*. Then* $|e_\alpha\rangle$ *is an eigenvector of* TS *with eigenvalue* $e^{\imath\alpha}$ *exactly when* α *is a solution of the equation*

$$s_0^2 \cot\left(\frac{\alpha}{2}\right) + \sum_k s_k^2\left[\cot\left(\frac{\alpha-\phi_k}{2}\right) + \cot\left(\frac{\alpha+\phi_k}{2}\right)\right] - s_{-1}^2\tan\left(\frac{\alpha}{2}\right) = 0. \tag{3}$$

This lemma allows us to determine the eigenvectors and eigenvalues of TS by specifying a set of constraints they must satisfy. The lemma is applied in [Amb07], and with slight variations in [AKR05], [Tul08] and [DH17], to obtain bounds on the smallest eigenphase of a walk operator. We use the lemma for the same purpose, applying it to obtain a lower bound for the smallest eigenphase of WF_1 and WF in Lemmas 6 and 12. We also use a similar technique in our analysis of the eigenvector $|\zeta\rangle$ in Lemma 7, where we derive a set of constraints and use them to find properties of a and $|\psi\rangle$.

The next theorem we state describes the behaviour of the eigenphases of TS in relation to those of T. The flip-flop theorem of [DH17] describes how the eigenphases of the operators interlace, with the exact pattern of interlacing depending on the eigenspaces of T that $|s\rangle$ intersects. We limit the theorem statement to the case we apply in this paper, where $|s\rangle$ intersects the (+1)-eigenspace, the (−1)-eigenspace, and at least one other eigenspace of T.

Theorem 7 (Flip-flop theorem) *Consider any real unitary* T *and let* $|s\rangle$ *be a state with real amplitudes in the same space. Denote the positive eigenphases of* T *different from* $0, \pi$ *by* $0 < \phi_1 \le \phi_2 \le \cdots \le \phi_m < \pi$*. If* $s_0 \neq 0$*,* $s_{-1} \neq 0$ *and* $s_k \neq 0$ *for some* k*, then* TS *has* $m+1$ *two-dimensional eigenspaces, and no* (+1)*- or* (−1)*-eigenspaces which overlap* $|s\rangle$*. The positive eigenphases* α_j *of* TS *satisfy the inequality* $0 < \alpha_0 < \phi_1 \le \alpha_1 \le \cdots \le \phi_m \le \alpha_m < \pi$*.*

We apply this theorem in Lemmas 6 and 12 to obtain an upper bound on the smallest positive eigenphases of WF_1 and WF, respectively. One of the contributions of our work is to show how Theorem 7 can be used in combination with Lemma 19 to tightly bound these eigenphases. We show that this approach can be used in the case of an operator composed with a reflection, and then by applying a second reflection, to an operator composed with a two-dimensional rotation.

Decomposition of $|+\rangle$ and $|-\rangle$

To analyse the behaviour of WF_1 and WF, we specify how W acts on vectors in the non-trivial eigenspaces of F. This allows us to prove Lemma 4, which is crucial to the analysis of our walk. Recall from the definitions in equation (3.10) and equation (3.11) that $|+\rangle$ and $|-\rangle$ are orthonormal vectors that have the same span as $|g\rangle$ and $|a_{00}\rangle$. Together with $|\circlearrowleft\rangle$, they span a space that includes the two-dimensional subspace on which F acts non-trivially. In this appendix, we prove Lemma 4, which describes how $|+\rangle$ and $|-\rangle$ decompose into the invariant subspaces of W.

To simplify notation, define

$$s^+_{kl} = \frac{1}{2}(r^+_{kl} + r^-_{kl}) = \sqrt{1 + \frac{\sin \tilde{l}}{p_{kl}}}$$

$$s^-_{kl} = \frac{1}{2}(r^+_{kl} - r^-_{kl}) = \epsilon_l \sqrt{1 - \frac{\sin \tilde{l}}{p_{kl}}}$$

and

$$d^+_{kl} = \frac{1}{2}(c^+_{kl} + c^-_{kl}) = \sqrt{1 + \frac{\sin \tilde{k}}{p_{kl}}}$$

$$d^-_{kl} = \frac{1}{2}(c^+_{kl} - c^-_{kl}) = \epsilon_k \sqrt{1 - \frac{\sin \tilde{k}}{p_{kl}}} \ .$$

In the case where $k = l = 0$, we define $s^+_{00} = d^+_{00} = \sqrt{2}$ and $s^-_{00} = d^-_{00} = 0$. Note that $r^\pm_{kl} = s^+_{kl} \pm s^-_{kl}$ and $c^\pm_{kl} = d^+_{kl} \pm d^-_{kl}$.

Recall that both $|a_{00}\rangle$ and $|g\rangle$ lie in the span of the basis states $|00\rangle$, $|01\rangle$, $|10\rangle$ and $|11\rangle$. We compute the projections of these basis states onto the components of the eigenvectors of W.

$$\begin{aligned}
\langle 0|u_{kl}\rangle &= \sqrt{2}\langle 0|r_{kl}\rangle\langle 0|\phi_r^k\rangle = \frac{1}{\sqrt{2n_r}}r_{kl}^- \qquad \langle 0|v_{kl}\rangle = \sqrt{2}\langle 0|c_{kl}\rangle\langle 0|\phi_c^l\rangle = \frac{1}{\sqrt{2n_c}}c_{kl}^- \\
\langle 1|u_{kl}\rangle &= \sqrt{2}\langle 1|r_{kl}\rangle\langle 1|\phi_r^k\rangle = \frac{1}{\sqrt{2n_r}}r_{kl}^+\omega_{n_r}^k \; \langle 1|v_{kl}\rangle = \sqrt{2}\langle 1|c_{kl}\rangle\langle 1|\phi_c^l\rangle = \frac{1}{\sqrt{2n_c}}c_{kl}^+\omega_{n_c}^l
\end{aligned}$$

$$\begin{aligned}
\langle 0|u_{kl}^1\rangle &= \sqrt{2}\langle 0|r_{kl}^1\rangle\langle 0|\phi_r^k\rangle = -\frac{1}{\sqrt{2n_r}}r_{kl}^+ \qquad \langle 0|v_{kl}^1\rangle = \sqrt{2}\langle 0|c_{kl}^1\rangle\langle 0|\phi_c^l\rangle = -\frac{1}{\sqrt{2n_c}}c_{kl}^+ \\
\langle 1|u_{kl}^1\rangle &= \sqrt{2}\langle 1|r_{kl}^1\rangle\langle 1|\phi_r^k\rangle = \frac{1}{\sqrt{2n_r}}r_{kl}^-\omega_{n_r}^k \; \langle 1|v_{kl}^1\rangle = \sqrt{2}\langle 1|c_{kl}^1\rangle\langle 1|\phi_c^l\rangle = \frac{1}{\sqrt{2n_c}}c_{kl}^-\omega_{n_c}^l .
\end{aligned}$$

Now, using the property that

$$\begin{aligned}
\frac{1}{2}\Big(r_{kl}^+\omega_{n_r}^k + r_{kl}^-\Big) &= \frac{1}{2}\omega_{n_r}^{k/2}\Big(r_{kl}^+\omega_{n_r}^{k/2} + r_{kl}^-\omega_{n_r}{}^{-k/2}\Big) = \omega_{n_r}^{k/2}\Big(\cos\Big(\frac{\tilde{k}}{2}\Big)s_{kl}^+ + \imath\sin\Big(\frac{\tilde{k}}{2}\Big)s_{kl}^-\Big) \\
\frac{1}{2}\Big(r_{kl}^-\omega_{n_r}^k - r_{kl}^+\Big) &= \frac{1}{2}\omega_{n_r}^{k/2}\Big(r_{kl}^-\omega_{n_r}^{k/2} - r_{kl}^+\omega_{n_r}{}^{-k/2}\Big) = \omega_{n_r}^{k/2}\Big(-\cos\Big(\frac{\tilde{k}}{2}\Big)s_{kl}^- + \imath\sin\Big(\frac{\tilde{k}}{2}\Big)s_{kl}^+\Big),
\end{aligned}$$

we compute

$$\begin{aligned}
\frac{1}{\sqrt{2}}(\langle 0| + \langle 1|)|u_{kl}\rangle &= \frac{1}{\sqrt{n_r}}\omega_{n_r}^{k/2}\Big(\cos\Big(\frac{\tilde{k}}{2}\Big)s_{kl}^+ + \imath\sin\Big(\frac{\tilde{k}}{2}\Big)s_{kl}^-\Big) \\
\frac{1}{\sqrt{2}}(\langle 0| + \langle 1|)|u_{kl}^1\rangle &= \frac{1}{\sqrt{n_r}}\omega_{n_r}^{k/2}\Big(-\cos\Big(\frac{\tilde{k}}{2}\Big)s_{kl}^- + \imath\sin\Big(\frac{\tilde{k}}{2}\Big)s_{kl}^+\Big) \\
\frac{1}{\sqrt{2}}(\langle 0| + \langle 1|)|v_{kl}\rangle &= \frac{1}{\sqrt{n_c}}\omega_{n_c}^{l/2}\Big(\cos\Big(\frac{\tilde{l}}{2}\Big)d_{kl}^+ + \imath\sin\Big(\frac{\tilde{l}}{2}\Big)d_{kl}^-\Big) \\
\frac{1}{\sqrt{2}}(\langle 0| + \langle 1|)|v_{kl}^1\rangle &= \frac{1}{\sqrt{n_c}}\omega_{n_c}^{l/2}\Big(-\cos\Big(\frac{\tilde{l}}{2}\Big)d_{kl}^- + \imath\sin\Big(\frac{\tilde{l}}{2}\Big)d_{kl}^+\Big).
\end{aligned}$$

Excluding the case $k = l = 0$, the squared amplitudes of the projections are then

$$\begin{aligned}
\left\|\langle 0|u_{kl}\rangle\right\|^2 &= \frac{1}{n_r}\left(1-\frac{\sin\tilde{k}\cos\tilde{l}}{p_{kl}}\right) & \left\|\frac{1}{\sqrt{2}}(\langle 0|+\langle 1|)|u_{kl}\rangle\right\|^2 &= \frac{1}{n_r}\left(1+\frac{\cos\tilde{k}\sin\tilde{l}}{p_{kl}}\right)\\
\left\|\langle 0|u^1_{kl}\rangle\right\|^2 &= \frac{1}{n_r}\left(1+\frac{\sin\tilde{k}\cos\tilde{l}}{p_{kl}}\right) & \left\|\frac{1}{\sqrt{2}}(\langle 0|+\langle 1|)|u^1_{kl}\rangle\right\|^2 &= \frac{1}{n_r}\left(1-\frac{\cos\tilde{k}\sin\tilde{l}}{p_{kl}}\right)\\
\left\|\langle 0|v_{kl}\rangle\right\|^2 &= \frac{1}{n_c}\left(1-\frac{\cos\tilde{k}\sin\tilde{l}}{p_{kl}}\right) & \left\|\frac{1}{\sqrt{2}}(\langle 0|+\langle 1|)|v_{kl}\rangle\right\|^2 &= \frac{1}{n_c}\left(1+\frac{\sin\tilde{k}\cos\tilde{l}}{p_{kl}}\right)\\
\left\|\langle 0|v^1_{kl}\rangle\right\|^2 &= \frac{1}{n_c}\left(1+\frac{\cos\tilde{k}\sin\tilde{l}}{p_{kl}}\right) & \left\|\frac{1}{\sqrt{2}}(\langle 0|+\langle 1|)|v^1_{kl}\rangle\right\|^2 &= \frac{1}{n_c}\left(1-\frac{\sin\tilde{k}\cos\tilde{l}}{p_{kl}}\right).
\end{aligned}$$

Lemma 20 *For any subspace* W_{kl},

$$\langle g|\Pi_{kl}|a_{00}\rangle = \begin{cases} \frac{1}{2} & \textit{if } k=l=0 \\ 0 & \textit{otherwise} \end{cases}. \tag{4}$$

Proof Recall that $|g\rangle = |00\rangle$ and $|a_{00}\rangle = \frac{1}{2}(|0\rangle+|1\rangle)\otimes(|0\rangle+|1\rangle)$. Consider any k, l not both 0. Then

$$\begin{aligned}
&\langle g|w^{11}_{k'l}\rangle\langle w^{11}_{k'l}|a_{00}\rangle\\
&= \frac{1}{2N}(r^+_{k'l}c^+_{k'l})\left(\omega_{n_r}^{k'/2}\omega_{n_c}^{l/2}\right)\left(-\cos\left(\frac{\tilde{k}'}{2}\right)s^-_{k'l}+\imath\sin\left(\frac{\tilde{k}'}{2}\right)s^+_{k'l}\right)\left(-\cos\left(\frac{\tilde{l}}{2}\right)d^-_{k'l}+\imath\sin\left(\frac{\tilde{l}}{2}\right)d^+_{k'l}\right)\\
&= \frac{-1}{2N}(r^+_{kl}c^-_{kl})\left(\omega_{n_r}^{-k/2}\omega_{n_c}^{l/2}\right)\left(\cos\left(\frac{\tilde{k}}{2}\right)s^+_{kl}+\imath\sin\left(\frac{\tilde{k}}{2}\right)s^-_{kl}\right)\left(\cos\left(\frac{\tilde{l}}{2}\right)d^-_{kl}+\imath\sin\left(\frac{\tilde{l}}{2}\right)d^+_{kl}\right)\\
&= \frac{-1}{2N}(r^-_{kl}c^-_{kl})\left(\omega_{n_r}^{k/2}\omega_{n_c}^{l/2}\right)\left(\cos\left(\frac{\tilde{k}}{2}\right)s^+_{kl}+\imath\sin\left(\frac{\tilde{k}}{2}\right)s^-_{kl}\right)\left(\cos\left(\frac{\tilde{l}}{2}\right)d^+_{kl}+\imath\sin\left(\frac{\tilde{l}}{2}\right)d^-_{kl}\right)\\
&= \quad -\langle g|w^{00}_{kl}\rangle\langle w^{00}_{kl}|a_{00}\rangle.
\end{aligned}$$

Similarly, it can be derived that

$$\langle g|w^{11}_{kl'}\rangle\langle w^{11}_{kl'}|a_{00}\rangle = -\langle g|w^{00}_{kl}\rangle\langle w^{00}_{kl}|a_{00}\rangle.$$

Using the property that $(k')' = k$, this further implies that $\langle g|w^{00}_{k'l'}\rangle\langle w^{00}_{k'l'}|a_{00}\rangle = \langle g|w^{00}_{kl}\rangle\langle w^{00}_{kl}|a_{00}\rangle$. By definition of the invariant subspaces W_{kl} in Section 3.4.2, this shows that for any k, l not both 0, $\langle g|\Pi_{kl}|a_{00}\rangle = 0$. It follows that $\langle g|\Pi_{00}|a_{00}\rangle = \langle g|a_{00}\rangle = \frac{1}{2}$. □

Lemma 4 *The following statements hold.*

$$\Pi_{kl}|+\rangle \perp \Pi_{kl}|-\rangle \qquad \text{for all subspaces } \mathsf{W}_{kl} \tag{3.22}$$

$$\|\Pi_{kl}|+\rangle\|^2 = \frac{2}{3}\frac{\dim(\mathsf{W}_{kl})}{N} \qquad \text{for all } k,l \text{ not both } 0 \tag{3.23}$$

$$\|\Pi_{kl}|-\rangle\|^2 = 2\frac{\dim(\mathsf{W}_{kl})}{N} \qquad \text{for all } k,l \text{ not both } 0 \tag{3.24}$$

$$\|\Pi_{00}|+\rangle\|^2 = \frac{2}{3}\frac{N+2}{N} \tag{3.25}$$

$$\|\Pi_{00}|-\rangle\|^2 = \frac{4}{N}. \tag{3.26}$$

Proof Observe that for any k,l not both zero,

$$\|\langle g|w_{kl}^{00}\rangle\|^2 + \|\langle g|w_{kl'}^{11}\rangle\|^2 + \|\langle g|w_{k'l}^{11}\rangle\|^2 + \|\langle g|w_{k'l'}^{00}\rangle\|^2$$

$$= \frac{1}{N}\left(1-\frac{\sin\tilde{k}\cos\tilde{l}}{p_{kl}}\right)\left(1-\frac{\cos\tilde{k}\sin\tilde{l}}{p_{kl}}\right) + \frac{1}{N}\left(1-\frac{\sin\tilde{k}\cos\tilde{l}}{p_{kl}}\right)\left(1+\frac{\cos\tilde{k}\sin\tilde{l}}{p_{kl}}\right)$$

$$+ \frac{1}{N}\left(1+\frac{\sin\tilde{k}\cos\tilde{l}}{p_{kl}}\right)\left(1-\frac{\cos\tilde{k}\sin\tilde{l}}{p_{kl}}\right) + \frac{1}{N}\left(1+\frac{\sin\tilde{k}\cos\tilde{l}}{p_{kl}}\right)\left(1+\frac{\cos\tilde{k}\sin\tilde{l}}{p_{kl}}\right)$$

$$= \frac{4}{N} = \|\langle a_{00}|w_{kl}^{00}\rangle\|^2 + \|\langle a_{00}|w_{kl'}^{11}\rangle\|^2 + \|\langle a_{00}|w_{k'l}^{11}\rangle\|^2 + \|\langle a_{00}|w_{k'l'}^{00}\rangle\|^2.$$

Therefore, for any subspace W_{kl} with $kl \neq 00$, we have $\langle g|\Pi_{kl}|g\rangle = \langle a_{00}|\Pi_{kl}|a_{00}\rangle = \frac{\dim(\mathsf{W}_{kl})}{N}$. Next, recall that W_{00} refers to the $(+1)$-eigenspace of W. We know that both $|g\rangle$ and $|a_{00}\rangle$ are normalized, so

$$\langle g|\Pi_{00}|g\rangle = \langle a_{00}|\Pi_{00}|a_{00}\rangle = 1 - \sum_{kl\neq 00}\frac{\dim(\mathsf{W}_{kl})}{N} = \frac{N+4}{2N}.$$

Applying Lemma 20, this implies that

$$\sqrt{3}\langle -|\Pi_{kl}|+\rangle = \langle g|\Pi_{kl}|g\rangle + \langle g|\Pi_{kl}|a_{00}\rangle - \langle a_{00}|\Pi_{kl}|g\rangle - \langle a_{00}|\Pi_{kl}|a_{00}\rangle = 0,$$

for any subspace W_{kl}. This proves equation (3.22).

To prove equation (3.23), we compute

$$\langle +|\Pi_{kl}|+\rangle = \frac{1}{3}\Big[\langle g|\Pi_{kl}|g\rangle - \langle g|\Pi_{kl}|a_{00}\rangle - \langle a_{00}|\Pi_{kl}|g\rangle + \langle a_{00}|\Pi_{kl}|a_{00}\rangle\Big] = \frac{2\dim(\mathsf{W}_{kl})}{3N},$$

and similarly for equation (3.24).

Using the property that $|+\rangle$ and $|-\rangle$ are normalized, this implies that

$$\|\Pi_{00}|+\rangle\|^2 = 1 - \sum_{kl\neq 00} \frac{2\dim(\mathsf{W}_{kl})}{3N} = \frac{2(N+2)}{3N},$$

which proves equation (3.25). We can similarly compute that $\|\Pi_{00}|-\rangle\|^2 = \frac{4}{N}$, proving equation (3.26). □

Sums

In this appendix, we prove asymptotic bounds on a set of sums over the spectrum of W, as defined in equation (3.7). We assume a square grid, with $n_r = n_c = \sqrt{N}$.

Fact 1 *Suppose* $0 < \alpha < \theta_{kl}$ *for all* k, l*, and consider the sum*

$$\sum_{kl\neq 00} \dim(\mathsf{W}_{kl}) \cot\left(\frac{\theta_{kl}-\alpha}{2}\right). \tag{5}$$

1. *If* $\alpha \in \Theta(\frac{1}{\sqrt{N}})$*, then the sum has order* $\Omega(\sqrt{N}\log N)$*.*
2. *If* $\alpha \in o(\frac{1}{\sqrt{N}})$*, then the sum has order* $\Theta(\alpha N \log N)$*.*

Proof Instead of taking the sum over the subspaces W_{kl}, which partition the domain of W, we convert to a sum over k and l. Recall that each pair $0 \le k, l \le \sqrt{N}/2 - 1$ corresponds to two eigenvectors of W: $|w_{kl}^{00}\rangle$ with eigenphase θ_{kl} and $|w_{kl}^{11}\rangle$ with eigenphase $-\theta_{kl}$. Using this property, we rewrite the sum as

$$\begin{aligned}
&\sum_{kl\neq 00} \dim(\mathsf{W}_{kl}) \cot\left(\frac{\theta_{kl}-\alpha}{2}\right) \\
&= \sum_{k=0}^{\frac{\sqrt{N}}{2}-1} \sum_{\substack{l=0 \\ \text{not both } 0}}^{\frac{\sqrt{N}}{2}-1} \cot\left(\frac{\theta_{kl}-\alpha}{2}\right) - \cot\left(\frac{\theta_{kl}+\alpha}{2}\right) \\
&= 2 \sum_{k=0}^{\frac{\sqrt{N}}{2}-1} \sum_{\substack{l=0 \\ \text{not both } 0}}^{\frac{\sqrt{N}}{2}-1} \frac{\cot(\frac{\alpha}{2})\left(\cot^2(\frac{\theta_{kl}}{2})+1\right)}{\cot^2(\frac{\alpha}{2}) - \cot^2(\frac{\theta_{kl}}{2})},
\end{aligned} \tag{6}$$

where the final equality follows from angle sum identities.

Next, observe that by the definition of θ_{kl},

$$\cot^2\left(\frac{\theta_{kl}}{2}\right)+1=\frac{2}{1-\cos\theta_{kl}}=\frac{1}{1-\cos^2\tilde{k}\cos^2\tilde{l}}.$$

We therefore consider the sum

$$\sum_{k=0}^{\frac{\sqrt{N}}{2}-1}\sum_{\substack{l=0\\ \text{not both }0}}^{\frac{\sqrt{N}}{2}-1}\frac{1}{1-\cos^2\tilde{k}\cos^2\tilde{l}}.$$

For the terms where $l=0$, we get

$$\sum_{k=1}^{\frac{\sqrt{N}}{2}-1}\frac{1}{1-\cos^2\tilde{k}}\in\Theta(N),$$

and similarly for $k=0$. The remaining terms satisfy

$$\sum_{k=1}^{\frac{\sqrt{N}}{2}-1}\sum_{l=1}^{\frac{\sqrt{N}}{2}-1}\frac{1}{1-\cos^2\tilde{k}\cos^2\tilde{l}}\in\Theta(N\log N).$$

Now, consider equation (6) in the case where $\alpha\in\Theta(\frac{1}{\sqrt{N}})$. We have

$$\begin{aligned}&2\sum_{k=0}^{\frac{\sqrt{N}}{2}-1}\sum_{\substack{l=0\\ \text{not both }0}}^{\frac{\sqrt{N}}{2}-1}\frac{\cot(\frac{\alpha}{2})\Big(\cot^2(\frac{\theta_{kl}}{2})+1\Big)}{\cot^2(\frac{\alpha}{2})-\cot^2(\frac{\theta_{kl}}{2})}\\ &\geq 2\sum_{k=0}^{\frac{\sqrt{N}}{2}-1}\sum_{\substack{l=0\\ \text{not both }0}}^{\frac{\sqrt{N}}{2}-1}\frac{\cot(\frac{\alpha}{2})\Big(\cot^2(\frac{\theta_{kl}}{2})+1\Big)}{\cot^2(\frac{\alpha}{2})}\\ &=\frac{2}{\cot(\frac{\alpha}{2})}\sum_{k=0}^{\frac{\sqrt{N}}{2}-1}\sum_{\substack{l=0\\ \text{not both }0}}^{\frac{\sqrt{N}}{2}-1}\frac{1}{1-\cos^2\tilde{k}\cos^2\tilde{l}}.\end{aligned}$$

Therefore,

$$\sum_{kl\neq 00}\dim(\mathsf{W}_{kl})\cot\left(\frac{\theta_{kl}-\alpha}{2}\right)\in\Omega(\sqrt{N}\log N).$$

In the case where $\alpha \in o(\frac{1}{\sqrt{N}})$, the denominator in equation (6) is dominated by the term $\cot^2(\frac{\alpha}{2})$. Therefore, the expression has the same asymptotic order as

$$\frac{1}{\cot(\frac{\alpha}{2})} \sum_{k=0}^{\frac{\sqrt{N}}{2}-1} \sum_{\substack{l=0 \\ \text{not both } 0}}^{\frac{\sqrt{N}}{2}-1} \frac{1}{1-\cos^2\tilde{k}\cos^2\tilde{l}} \in \Theta(\alpha N \log N),$$

proving the second clause. □

Fact 2 *Suppose* $0 < \alpha < \theta_{kl}$ *for all* k, l, *and that* $\alpha \in o(\frac{1}{\sqrt{N}})$. *Then*

$$\sum_{kl \neq 00} \dim(\mathsf{W}_{kl}) \left[\cot\left(\frac{\theta_{kl}+\alpha}{2}\right) - \cot\left(\frac{\theta_{kl}-\alpha}{2}\right)\right]^2 \in \Theta(\alpha^2 N^2). \tag{7}$$

Proof

$$\sum_{kl \neq 00} \dim(\mathsf{W}_{kl}) \left[\cot\left(\frac{\theta_{kl}+\alpha}{2}\right) - \cot\left(\frac{\theta_{kl}-\alpha}{2}\right)\right]^2$$
$$= \sum_{k=0}^{\frac{\sqrt{N}}{2}-1} \sum_{\substack{l=0 \\ \text{not both } 0}}^{\frac{\sqrt{N}}{2}-1} \left[\cot\left(\frac{\theta_{kl}+\alpha}{2}\right) - \cot\left(\frac{\theta_{kl}-\alpha}{2}\right)\right]^2$$

By a similar derivation as in Fact 1, this sum has the same order as

$$\frac{1}{\cot^2(\frac{\alpha}{2})} \sum_{k=0}^{\frac{\sqrt{N}}{2}-1} \sum_{\substack{l=0 \\ \text{not both } 0}}^{\frac{\sqrt{N}}{2}-1} \left(\frac{1}{1-\cos^2\tilde{k}\cos^2\tilde{l}}\right)^2.$$

Using the Taylor expansion of cosine, for $l = 0$ we get

$$\sum_{k=0}^{\frac{\sqrt{N}}{2}-1} \left(\frac{1}{1-\cos^2\tilde{k}}\right)^2 \in \Theta(N^2),$$

and similarly for $k = 0$. Finally,

$$\sum_{k=1}^{\frac{\sqrt{N}}{2}-1} \sum_{l=1}^{\frac{\sqrt{N}}{2}-1} \left(\frac{1}{1-\cos^2\tilde{k}\cos^2\tilde{l}}\right)^2 \in \Theta(N^2).$$

Therefore,

$$\sum_{kl \neq 00} \dim(\mathsf{W}_{kl}) \left[\cot\left(\frac{\theta_{kl}+\alpha}{2}\right) - \cot\left(\frac{\theta_{kl}-\alpha}{2}\right)\right]^2 \in \Theta(\alpha^2 N^2).$$

□

Lower bound for 0-determinant verification

As a final note, we present an observation on lower bounds for the query complexity of 0-determinant verification. This is the same problem solved by the span program we present in Section 4.3.

Currently, the best published lower bound for 0-determinant verification is by Dörn and Theirauf [DT09]. They prove a bound of $\Omega(n^2)$ quantum queries for deciding whether an $n \times n$ matrix has determinant equal to some value k. This matches the trivial upper bound of $O(n^2)$ queries, obtained by simply querying all entries in the input matrix, so the query complexity bound on k-determinant verification is tight. This tight bound is stated and cited as applying to 0-determinant verification as well. However, the $\Omega(n^2)$ lower bound by Dörn and Thierauf does not apply when the problem is restricted to the case where $k = 0$. We argue this as follows.

The proof in [DT09] is based on reducing a series of problems on $n \times n$ matrices to the $SUM_{n^2}(x_1, x_2, \ldots, x_{n^2}, a)$ problem. In this sum problem, the goal is to decide whether the sum of the n^2 binary inputs is equal to the input number a. In general, the query complexity of this problem is $\Omega(n^2)$, but this bound does not apply when the problem is restricted to the case where $a = 0$. In fact, when $a = 0$, the sum problem can be solved in only $\Theta(n)$ quantum queries by using quantum search [Gro96, BHMT02].

The lower bound for k-determinant verification is proved by a reduction to $SUM_{n^2}(x_1, x_2, \ldots, x_{n^2}, k)$. Therefore, the lower bound of $\Omega(n^2)$ still holds for general k. In 0-determinant verification, where $k = 0$, it is only possible to obtain the weaker bound of $\Omega(n)$ quantum queries for the problem through this reduction. Thus, there remains a gap between the current upper and lower bounds for the 0-determinant verification problem.

As a further observation, we note that in [DT09], the lower bound on query complexity for 0-determinant verification is also used to derive a bound for the matrix rank problem. In the matrix rank problem, the goal is to decide whether the rank of an input matrix is equal to some value k. The lower bound derivation is based on case $k = n$, using the property that a matrix has full rank if and only if its determinant is nonzero. By our previous statement, deciding whether the determinant is nonzero has a lower bound of $\Omega(n)$ queries, so our observation implies that the lower bound shown in [DT09] is in fact $\Omega(n)$ for the rank problem as well.