

UNIVERSIDADE FEDERAL DE MINAS GERAIS
Instituto de Ciências Exatas
Programa de Pós-Graduação em Ciência da Computação

Pedro Vinícius Ferreira Baptista

TRANSFERÊNCIA DE ESTADO QUASE PERFEITA

Belo Horizonte
2021

Pedro Vinícius Ferreira Baptista

TRANSFERÊNCIA DE ESTADO QUASE PERFEITA

Versão Final

Dissertação apresentada ao Programa de Pós-Graduação em Ciência da Computação do Instituto de Ciências Exatas da Universidade Federal de Minas Gerais como requisito parcial para a obtenção do grau de Mestre em Ciência da Computação.

Orientador: Gabriel de Moraes Coutinho

Belo Horizonte
2021

Pedro Vinícius Ferreira Baptista

PRETTY GOOD STATE TRANSFER

Final Version

Thesis presented to the Graduate Program in Computer Science of the Federal University of Minas Gerais in partial fulfillment of the requirements for the degree of Master in Computer Science.

Advisor: Gabriel de Morais Coutinho

Belo Horizonte
2021

Baptista, Pedro Vinícius Ferreira.

B222p Pretty good state transfer [manuscrito] / Pedro Vinícius
Ferreira Baptista — 2021.
116 f. il.

Orientador: Gabriel de Moraes Coutinho.
Dissertação (mestrado) - Universidade Federal de Minas
Gerais, Instituto de Ciências Exatas, Departamento de Ciência
da Computação
Referências: f. 114-116

1. Computação – Teses. 2. Passeios quânticos – Teses. 3.
Teoria dos grafos – Teses. 4. Computação quântica – Teses.
I. Coutinho, Gabriel de Moraes. III. Universidade Federal de
Minas Gerais, Instituto de Ciências Exatas, Departamento de
Ciência da Computação. IV. Título.

CDU 519.6*51 (043)



UNIVERSIDADE FEDERAL DE MINAS GERAIS
INSTITUTO DE CIÊNCIAS EXATAS
PROGRAMA DE PÓS-GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO


FOLHA DE APROVAÇÃO


Pretty Good State Transfer

PEDRO VINÍCIUS FERREIRA BAPTISTA

Dissertação defendida e aprovada pela banca examinadora constituída pelos Senhores:


PROF. GABRIEL DE MORAIS COUTINHO - Orientador
Departamento de Ciência da Computação - UFMG


PROF. MARIO SÉRGIO FERREIRA ALVIM JÚNIOR
Departamento de Ciência da Computação - UFMG


PROF. CARLOS FELIPE LARDIZABAL RODRIGUES
Instituto de Matemática e Estatística - UFRGS


PROF. RAPHAEL CAMPOS DRUMOND
Departamento de Matemática - UFMG

Belo Horizonte, 10 de Novembro de 2021.

Resumo

Passeios quânticos em tempo contínuo é uma das áreas de pesquisa em teoria algébrica de grafos e computação quântica. Uma das suas sub-áreas de pesquisa é a de transferência de estados entre vértices de um grafo. Transferências de estados são importantes, pois permitem avaliar em quais casos uma rede com comunicação feita através de estados modelados por um grafo permitem que esses estados sejam transmitidos com o máximo possível de probabilidade de maneira eficiente.

Em geral, trabalhos sobre transferência de estados lidam com transferências perfeitas ou quase perfeitas entre dois vértices. Transferências perfeitas de estado possuem caracterizações para as matrizes de Adjacência e Laplaciana. Além disso, foi mostrado ser possível verificar transferências perfeitas de estado em um grafo com um algoritmo polinomial.

Em relação a transferências quase perfeitas, embora existam caracterizações para sua ocorrência, tais caracterizações demandam um certo trabalho para sua verificação em grafos e nenhum algoritmo exato é conhecido para validar sua existência.

Alguns artigos mostram que, devido às restrições que as transferências perfeitas impõem nos autoespaços do grafo, tais transferências são relativamente raras em classes comuns de grafos. Portanto, é natural tentar verificar a ocorrência de transferências quase perfeitas de estados.

Nessa dissertação, apresenta-se o primeiro algoritmo exato para conferir a ocorrência de transferências quase perfeita de estados em grafos. Além disso, aplicou-se resultados conhecidos nas matrizes de adjacência e Laplaciana de transferências perfeita e quase perfeita de estados na matriz Normalizada Laplaciana, considerando sua relação com passeios clássicos em grafos.

Palavras-chave: Passeios Quânticos, Transferências de Estado, Computação Quântica, Teoria Algébrica de Grafos.

Abstract

Continuous-time quantum walks is a recent area of research, both in quantum computing and algebraic graph theory. It has applications in quantum search algorithms, state transfer and, more recently, in creating a set of universal quantum gates. In terms of state transfer, its main motivation is to research in which cases there can be transfer of states in a network comprised of vertices of a graph and how much time must we wait for a specific probability.

One of the main topics of research in state transfer in quantum walks is about two types of transfer: perfect state transfer and pretty good state transfer. In so far, the former has been thoroughly researched with characterizations for its occurrence on the Adjacency and Laplacian matrices. Furthermore, it was also shown that we can verify its occurrence in a graph using a classical algorithm in polynomial time.

As for the pretty good state transfer, some results are known in terms of its occurrence for some classes of graphs and also for a characterization of its occurrence. The main problem is that the characterizations we know demand some work to verify it for a graph, and no exact algorithm was known to do it.

Furthermore, for some common classes of graphs, it was shown that perfect state transfer is rare. This is mostly due to the restrictions it imposes on the eigenspaces of the graph. Therefore, since we cannot have state transfer with 1 probability, it is natural to check for state transfer with probability close to 1 and at what time cost it demands.

In this master's thesis, we present the first exact algorithm for verifying pretty good state transfer in graphs. Another path of research was to try to replicate some known results of state transfer, perfect and pretty good state transfers, for the adjacency and the Laplacian matrix in the Normalized Laplacian. The motivation for that arises in the connection of the Normalized Laplacian with the Classical Random Walk.

Palavras-chave: Quantum Walks, State Transfer, Quantum Computing, Algebraic Graph Theory.

List of Figures

2.1	Graph and Adjacency Matrix of P_6	46
2.2	Approximate roots of $P_+(x)P_-(x)$	47
2.3	An exact formula for roots of $P_+(x)P_-(x)$	47
3.1	Graph $P_2 \square P_3$	71
3.2	Adjacency Matrix of $P_2 \square P_3$	71

List of Tables

3.1	Table of convergents of $(\sqrt{5} + 1)/2$ modulo 4	69
3.2	Table of convergents of $\sqrt{3}$ modulo 4	70
3.3	Table of convergents of $\sqrt{2} + 1 \pmod{2}$	72
3.4	Convergents of $\sqrt{2} + 1 \pmod{3}$	73

Contents

Resumo	6
Abstract	7
List of Figures	8
List of Tables	9
1 Introduction	12
1.1 Motivation	12
1.2 Background	14
1.3 Contributions	21
2 Deciding Pretty Good State Transfer	23
2.1 Conditions to test Pretty Good State Transfer	23
2.2 Computing the Splitting Field	26
2.3 Solving Diophantine Linear Systems	35
2.4 Algorithm for Deciding Pretty Good State Transfer	41
2.5 Computing the Average Mixing Matrix	51
3 Continued Fractions	55
3.1 Why use Continued Fractions?	55
3.2 How to Compute Continued Fractions?	60
3.3 Continued Fractions and State Transfer	67
4 Normalized Laplacian	75
4.1 Other Matrices	75
4.2 Strong Cospectrality	77
4.3 Perfect State Transfer on The Normalized Laplacian	81
4.4 Perfect State Transfer in Trees	93

4.5	Pretty Good State Transfer in Paths	101
5	Future Work	111
	Bibliography	114

Chapter 1

Introduction

1.1 Motivation

The development of computational systems has been one of the most fundamental reasons for the exponential growth of technology in the last decades. This development has been based not only on better algorithms, but also on the continued capability of decreasing the size of transistors, which are the basic units that enable the computation of modern computers.

This decreasing in the size of the transistors is famously known as Moore's Law, which was an observation made by Gordon Moore that the size of the transistors were decreasing in half in about one and half years continuously and the amount of transistors in a chip doubled in the same rate.

Recently, this possibility has diminished as the transistors are getting smaller and their size is getting closer to its physical limit. Because of it, there has been a shift in focus on research not only to try to find other materials that could further this limit, but also in other models of computation that do not depend on transistors and could still give us more computational power.

One of such models is quantum computing. One of the first to think of quantum systems used for computation was Richard Feynman in his famous paper *Simulating Physics with computers*, Feynman [1982]. There he discusses quantum physics phenomena in which a simulation in a classical computer would require an exponentially large complexity. His proposed solution was the development of computers that would use quantum systems within their processing units.

As much as this paper drew some attention to the idea of quantum computers, the real shift in focus to quantum computing appeared in the paper of Shor [1994] when he showed that factoring integers, which is a problem thought to be difficult in classical

models of computation, if not *NP*-complete for classical computers, can be solved in polynomial time in quantum computers. This result, together with Grover's algorithm, Grover [1996], and others that came after it showed that quantum computers, if possible to construct, could be very useful to solve or, at least, speed up the solution to a large variety of modern problems.

Another area in which quantum computing can be applied is cryptography. Quantum computing opens new paths for investigation, both for using quantum channels as a way of communication, but also for creating new methods of cryptography using quantum objects.

One extra reason for the interest in the area was because of the risk that Shor's algorithm could do to methods using mathematical tools like the RSA, that relied on the widely believed assumption that it is hard to factor integers into its prime factors in classical algorithms.

The new ways that could be explored to make cryptography, together with the possibility that one of its most used methods could be weak against quantum algorithms, made it important to find novel ways to make communication secure. One of the first papers to propose a new idea on quantum communication as a way of cryptography is in Bennett and Brassard [1984].

Relating more specifically to the work in this thesis, we highlight the work done in quantum walks and its applications. First, in Farhi and Gutmann [1997], it was shown that continuous-time quantum walks could be used for walking on a tree, similarly as the classical random walk. Not only that, but, for some cases, classical random walks was exponentially slower than quantum walks. Even though there were faster classical algorithms, this showed the potential of quantum walks.

Continuing the work on quantum walks, in Childs and Goldstone [2003] it was shown that quantum walks could be used for a searching a vertex that contained a loop in a graph. Christandl et al. [2004] showed cases in which continuous-time quantum walks could be used for transferring states between vertices of a graph with fidelity 1, i.e., perfect state transfer. In Godsil [2010], Godsil showed necessary conditions on the eigenvalues of the graph for perfect state transfer to occur.

More recently, in Herrman and Humble [2019] and Herrman and Wong [2021], it was shown how it is possible to create a universal set of gates for quantum computation by performing quantum walks on dynamic graphs, meaning that we can add/remove edges from the graph between quantum walks. In these papers, it is also shown how perfect state transfer can be used not only for creating a universal set of quantum gates, but also for reducing the amount of graphs and the time necessary for these quantum walks.

Now, one might ask why is pretty good state transfer relevant. One reason is that if we want to create quantum systems to perform operations for us, we may need some sort of communication in a network that can have qubits as nodes. Or, as we talked above, quantum walks can be used for creating universal sets of quantum gates.

In both scenarios, the quantum walk can be defined by a Hamiltonian that evolves in time, as we will show in Section 1.2. At some point, we need to transfer some unknown quantum state. Of course, we want it to have a high probability of it being close to perfect state transfer.

However, as it is stated in Godsil et al. [2012], for the Heisenberg XY Hamiltonian a chain of qubits has perfect state transfer if and only if it has 2 or 3 qubits. The same paper shows us that for a chain of n qubits pretty good state transfer happens if and only if $n = p - 1$ or $n = 2p - 1$ or $n = 2^m - 1$ for any prime p and any positive integer m .

Furthermore, in Coutinho and Liu [2015], the authors state that for the Laplacian matrix model, no perfect state transfer happens for any tree on more than two vertices and as paths are also trees the result holds also for paths on more than two vertices. Meanwhile, in Banchi et al. [2017], it is shown that pretty good state transfer happens between the extreme vertices of paths for the Laplacian matrix if and only if n is a power of 2.

So these are some of the results that show that if we want to create even a simple network, in this case a chain, unless we are willing to accept a state transfer close to, but not equal to, perfect we are limited on the choices we may have. For this reason, investigating pretty good state transfer can lead us to show not only on which networks we may get an arbitrarily close to 1 probability if we accept to wait for it, but also could lead us to show how much time we should expect to wait for a given deviation from perfect state transfer.

1.2 Background

In classical computing, we have bits as the basic unit of information. A bit can be in either the 0 or 1 state. For quantum computing, we have qubits as this basic unit of information. We can write the quantum counterparts of the states as 1–dimensional subspaces represented by vectors in \mathbb{C}^2 as the following

$$|0\rangle = \begin{pmatrix} 1 \\ 0 \end{pmatrix}, \quad |1\rangle = \begin{pmatrix} 0 \\ 1 \end{pmatrix}.$$

Above we used Dirac's notation, where $|0\rangle = e_1$ and $\langle 0| = e_1^T$, and e_1 is the first vector of the standard basis of \mathbb{C}^2 with 1 in the first entry and 0 elsewhere. However, with quantum computing, the qubit has more possible states. A state of a quantum system is a 1-dimensional subspace of \mathbb{C}^{2^n} , where n is the number of qubits of our system. For example, for a single qubit we can view it as any vector that lies in the complex sphere of dimension 2 and of distance 1 to the origin.

The time-evolution of a closed quantum system is postulated to be determined by a unitary mapping on \mathbb{C}^m for some integer m . So, in quantum computing, any computation can be described as a unitary matrix U applied to a given initial state $|\phi\rangle$, that is, $U|\phi\rangle$.

The Hamiltonian is an operator that defines the energy of a quantum system. Schrödinger's equation uses it to define how the quantum system evolves according to time, as follows

$$i\hbar \frac{d}{dt} |\Psi(t)\rangle = \hat{H} |\Psi(t)\rangle, \quad (1.1)$$

where \hat{H} is the Hamiltonian of the system, $|\Psi(t)\rangle$ is the state vector of the quantum system in time t and \hbar is Planck's constant.

The solution to Schrödinger's equation is a transition operator $U(t)$ of the form

$$U(t) = \exp(-it\hat{H}), \quad (1.2)$$

where \hat{H} is a Hermitian matrix. Therefore, because \hat{H} is a Hermitian matrix, it can be diagonalized. Let $\theta_0, \theta_1, \dots, \theta_d$ be its distinct eigenvalues with respective E_0, E_1, \dots, E_d eigenprojectors. In [Hoffman and Kunze, 1971, Theorem 9], for instance, it is described the Spectral Decomposition Theorem for normal operators. Therefore, as $U(t)$ is unitary, a consequence of this theorem is that it can be written as

$$U(t) = \exp(-it\hat{H}) = \sum_{r=0}^d \exp(-it\theta_r) E_r. \quad (1.3)$$

This operator defines a continuous time quantum walk and \hat{H} is the Hamiltonian of the walk. In this thesis, we are interested in the case where \hat{H} is a block matrix and one of these blocks is a graph related matrix. The standard choice is that this matrix is the adjacency matrix of the graph. However, it could be the Laplacian, the Normalized Laplacian or any other representation of the graph.

For instance, in Banchi et al. [2017], we have the Heisenberg (XYZ) Hamiltonian

defined by

$$H_{XYZ} = \frac{1}{2} \sum_{i \neq j} A_{ij} (X_i X_j + Y_i Y_j + Z_i Z_j), \quad (1.4)$$

where X_i represents the operator that applies the Pauli matrix X on the qubit at position i , but is elsewhere the identity operator. Analogously for Y_i, Z_i , considering the Pauli matrices Y and Z , respectively. A_{ij} is the (i, j) -entry of the adjacency matrix of the graph we are modeling. Similarly, it is defined the XY Hamiltonian as follows

$$H_{XY} = \frac{1}{2} \sum_{i \neq j} A_{ij} (X_i X_j + Y_i Y_j). \quad (1.5)$$

If we define H_{XY}^1 and H_{XYZ}^1 to be the XY -Hamiltonian and XYZ -Hamiltonian, respectively, restricted on the action on the single-particle subspace defined by the product of $X_i |0\rangle^{\otimes n}$. One can show that this action can be represented as follows

$$\begin{aligned} H_{XYZ}^1 &= |E(G)|I - 2L, \\ H_{XY}^1 &= 2A, \end{aligned}$$

for a graph G with adjacency matrix A , Laplacian matrix L , edge set $E(G)$ and I is the identity matrix of appropriate dimension. So, up to a shift and scaling, we can use A and L as Hamiltonians in this subspace, as it is explained in Banchi et al. [2017].

If we say nothing of the Hamiltonian matrix at hand, we assume that it is the adjacency matrix $A(G)$ of the graph G . If there is no confusion, we denote the adjacency matrix only as A . Now, with that description of quantum walks, we can define two types of state transfer: perfect state transfer, when we can transfer the state between vertices with 100% probability, and pretty good state transfer, when we can get as close to a perfect state transfer as we want, but not necessarily reach it.

Definition 1.1 (Pretty good state transfer). Let G be a graph with adjacency matrix A , vertices a and b . If, for all $0 < \epsilon \leq 1$, there is a positive time t such that

$$|e_b^T U(t) e_a| = |(\exp(-itA))_{a,b}| > 1 - \epsilon, \quad (1.6)$$

then we say that pretty good state transfer occurs between a and b .

Definition 1.2 (Perfect state transfer). Let G , A , a and b be as above. If we allow $\epsilon \geq 0$ and turn the above inequality into an equality when $\epsilon = 0$, then we say that perfect state transfer occurs between a and b .

This formulation does not give a practical way of testing if the phenomena occur.

In order for us to get a better characterization of perfect state transfer, we need more basic definitions. To simplify notation, we order the vertices v_1, v_2, \dots, v_n in any way and each has a characteristic vector that represents it, so that e_i represents the vertex v_i . So let us give a few definitions first. They give some conditions that the eigenspaces can have relative to a pair of vertices.

Definition 1.3 (Cospectral vertices). We say that two vertices a and b of a graph $A(G)$ are cospectral if $A(G \setminus a)$ and $A(G \setminus b)$ have the same spectrum, i.e., if they have the same eigenvalues with respective multiplicities. Here, $A(G \setminus a)$ is the adjacency matrix A of the graph G removing the vertex a .

Definition 1.4 (Parallel vertices). We say that two vertices a and b of a graph $A(G)$ are parallel if, for all r , the projections onto eigenspaces $E_r e_a$ and $E_r e_b$ are parallel.

Definition 1.5 (Strongly cospectral vertices). We say that two vertices a and b of a graph $A(G)$ are strongly cospectral if they are cospectral and parallel.

We call S_a the eigenvalue support of a vertex a , where θ_r , an eigenvalue of A , is in S_a if and only if $E_r e_a \neq 0$. When there is no confusion over which vertex we are dealing with, we use S for it. With that defined, we can give a better characterization of perfect state transfer that is presented in Coutinho [2014].

Theorem 1.6 (Characterization of perfect state transfer). *Let a and b vertices in a graph G and assume that $S = \{\theta_0, \theta_1, \dots, \theta_k\}$ is the eigenvalue support of a , and θ_0 is the largest eigenvalue. There is perfect state transfer between vertices a and b if and only if the following conditions hold:*

1. *Vertices a and b are strongly cospectral.*
2. *The eigenvalues in S are either integers or quadratic integers, and if the latter is true, there are integers $a, b_0, b_1, \dots, b_k, \Delta$ where Δ is square-free and positive, such that*

$$\theta_r = \frac{1}{2}(a + b_r \sqrt{\Delta}).$$

3. *Let g to be as follows*

$$g = \gcd \left(\frac{\theta_0 - \theta_r}{\sqrt{\Delta}} \right)_{r=0, \dots, k}.$$

Then

- $(E_r)_{ab} > 0$ if and only if $(\theta_0 - \theta_r)/(g\sqrt{\Delta})$ is even,
- $(E_r)_{ab} < 0$ if and only if $(\theta_0 - \theta_r)/(g\sqrt{\Delta})$ is odd.

If the conditions hold, then the minimum time we have perfect state transfer between a and b is $\tau = \pi/g\sqrt{\Delta}$. Other times in which it occurs are odd multiples of τ .

This characterization was first written as it is defined in Coutinho [2014] and as it was noted in Coutinho and Godsil [2017] this theorem can be used to construct a polynomial time algorithm to test if perfect state transfer happens. Not only that, if the graph has perfect state transfer it gives us the precise time, or rather, the minimal time and its multiples such that it occurs.

Below we show how one might test this based on the same paper: Coutinho and Godsil [2017]. In the paper, it is shown not only how to test, but also shows that we are able to do a procedure that test if perfect state transfer happens in polynomial time on the size of the graph.

- First, we need to check if two given vertices of the graph are strongly cospectral. Let $\phi(A)$ be the characteristic polynomial of $A(G)$. If $S \subseteq V(G)$, then let $\phi_S(A)$ be the characteristic polynomial of $A(G \setminus S)$. It is easy to see that two vertices a and b of G are cospectral if and only if $\phi_a(A) = \phi_b(A)$.
- Now, to test if they are parallel, we use [Coutinho and Godsil, 2017, Lemma 2.4] that shows that we only need to test if the poles of $\phi_{ab}(A)/\phi(A)$ are simple. For that, we need $g(x) = \gcd(\phi_{ab}(A), \phi(A))$, which we can get by performing the Euclidean algorithm. So, we need $f(x) = \phi(X)/g(x)$ to have no repeated roots. However, that is the same as $f(x)$ and $f'(x)$ having no common divisor of degree > 0 .
- For condition 2, we take $f(x) = \phi(A)/\gcd(\phi(A), \phi(A \setminus a))$. As it is shown in the paper, the roots of this polynomial are simple and are exactly the eigenvalue support of a .
- Now, we need to check if the roots are in the form defined in condition 2 of Theorem 1.6. They must be either all integers or all quadratic integers of the form defined. Let a_i be the coefficient of x^i in $f(x)$ and $\deg(f(x))$ be its degree.
 - First, we test if they are all integers. For this, we can just factor the polynomial over the integers. If all the factors are linear, then they are all integers, and we can get the roots by checking the zero-degree coefficient.

- Now, if they are not integers, but they can still be of the second form, we need to test if all roots are quadratic integers. We use the same idea as above, we factor the polynomial over the integers and see if all the factors are quadratic factors (besides a possible zero root).
- With that, we get all possible quadratic integer roots of $f(x)$. Then, we need to check if we have $\deg(f(x))$ of them and if they are all of the form defined in condition 2 of Theorem 1.6. To see this, we use the coefficients of the polynomial factors to compute the square-free $\Delta > 1$ of each polynomial factor and check if the square-free part Δ of them are equal. If so, then we test the next condition. Otherwise, there is no perfect state transfer.
- Now we only need the final condition. We already have the square-free delta. So, we only need to check the following. Let g be the gcd of all the differences between θ_0 and all eigenvalues in the support. Compute one eigenvector, v_r , which can be done with Gaussian elimination, for each eigenvalue in the support and check if $(v_r)_a = (v_r)_b$, then $(\theta_0 - \theta_r)/(g\sqrt{\Delta})$ must be even. Otherwise, it must be odd.
- If they pass all these conditions, then perfect state transfer must happen between vertices a and b at the minimal time $\pi/(g\sqrt{\Delta})$.

This whole procedure is an adaptation of the one suggested in Coutinho and Godsil [2017] and there we have a proof that it is polynomial time. In the end, we can check if two vertices of a graph have perfect state transfer between them and at which minimal time it does happen, if it does.

Now, perfect state transfer is a stronger version of pretty good state transfer, meaning that if we have perfect state transfer we indeed have pretty good state transfer, so we may want to derive similar characterizations and algorithms to test and show at which time τ we have pretty good state transfer. In order for it to happen between vertices a and b we need

$$U(\tau)e_a \approx \lambda e_b, \tag{1.7}$$

where $|\lambda| = 1$. Here, we are using \approx to mean that the expression on the left can get arbitrarily close to the expression on the right, for different choices of τ . See definition 1.1. Now, if we use Equation 1.3 and multiply the equation above by E_r , we have that pretty good state transfer is equivalent to this equation below for all r

$$e^{i\tau\theta_r} E_r e_a \approx \lambda E_r e_b. \tag{1.8}$$

Now, as $E_r e_a$ is a real vector, for all r such that $E_r e_a \neq 0$, we have that $e^{i\tau\theta_r} \approx \lambda = e^{i\delta}$ and so, $\tau\theta_r \approx \delta + m_r\pi$, where m_r is even. That exact observation, together with Kronecker's theorem shown below, was key to provide a characterization of pretty good state transfer.

Theorem 1.7 (Kronecker's Theorem, see reference Levitan et al. [1982], Chapter 3). *Let $\theta_1, \theta_2, \dots, \theta_n$ and $\lambda_1, \lambda_2, \dots, \lambda_n$ be arbitrary real numbers. For the system of inequalities*

$$|\theta_k t - \lambda_k| < \epsilon \pmod{2\pi} \text{ with } k = 1, 2, \dots, n \quad (1.9)$$

to have consistent real solutions for any arbitrarily small positive number ϵ , it is necessary and sufficient that every time the relation $\ell_1\theta_1 + \ell_2\theta_2 + \dots + \ell_n\theta_n = 0$ holds, where $\ell_1, \ell_2, \dots, \ell_n$ are integers, we have the congruence

$$\ell_1\lambda_1 + \ell_2\lambda_2 + \dots + \ell_n\lambda_n \equiv 0 \pmod{2\pi}. \quad (1.10)$$

So, we make the theorem's θ_k to be the eigenvalues of the graph, $t = \tau$, and the theorem's λ_k to be equal $\delta + m_r\pi$ for some δ such that $U(t)e_a \approx \lambda e_b$ and $\lambda = e^{i\delta}$. In Banchi et al. [2017], they used these observations to prove the following characterization of pretty good state transfer. The same can be also found in a similar form for paths in Coutinho et al. [2017].

Theorem 1.8 ([Banchi et al., 2017, Theorem 2]). *Let a and b be vertices of a graph G represented by a symmetric matrix A . Then pretty good state transfer happens between a and b if and only if both conditions below are satisfied.*

1. *Vertices a and b are strongly cospectral. In this case, let $\theta_0, \theta_1, \dots, \theta_d$ be eigenvalues in their support, and for $r = 0, 1, \dots, d$, let σ_r be defined as 0 if the projections onto E_r are equal, and 1 if they have opposite signs.*
2. *For all sets of integers $\ell_0, \ell_1, \dots, \ell_d$ such that*

$$\sum_{r=0}^d \ell_r \theta_r = 0 \text{ and } \sum_{r=0}^d \ell_r \sigma_r \text{ is odd,} \quad (1.11)$$

then

$$\sum_{r=0}^d \ell_r \neq 0. \quad (1.12)$$

For the Laplacian matrix, we have that one of the eigenvalues, say $\theta_0 = 0$ for the eigenvector $\mathbb{1}$, therefore $\sigma_0 = 0$. This was observed in [Banchi et al., 2017, Corollary 5]

and allowed them to change the constraints. Because now it is always possible to make $\sum_{r=0}^d \ell_r = 0$, so we only need to show that if $\sum_{r=1}^d \ell_r \theta_r = 0$, then $\sum_{r=1}^d \ell_r \sigma_r$ is even.

So, one application of this theorem together with the corollary above is to show whether pretty good state transfer happens on paths for the Laplacian matrix. In Banchi et al. [2017], they showed first that the eigenvalues of these matrices are of the form

$$\lambda_r = 2 - (\zeta_{2n}^r + \zeta_{2n}^{2n-r}), \quad (1.13)$$

where $\zeta_{2n} = e^{i\pi/2n}$. This allowed them to write the linear combination of the eigenvalues as

$$\sum_{r=1}^{n-1} \ell_r (-2 + (\zeta_{2n}^r + \zeta_{2n}^{2n-r})) = 0. \quad (1.14)$$

Assuming that $\ell_0 = \sum_{r=1}^{n-1} \ell_r$ this equation can be transformed into the following polynomial $L(x)$

$$L(x) = 2\ell_0 + \sum_{r=1}^{n-1} \ell_r x^r + \sum_{r=n+1}^{2n-1} \ell_{2n-r} x^r. \quad (1.15)$$

So, if the above polynomial is equal to zero when applied at ζ_{2n} , then it must be divisible by the cyclotomic polynomial $\Phi_{2n}(x)$ and that division, if exact, imposes conditions on the coefficients ℓ_r . This fact, together with the fact that

$$\sum_{r=0}^{n-1} (-1)^r E_r = R, \quad (1.16)$$

where R is the anti-diagonal matrix and E_r are eigenprojectors of the spectrum of the Laplacian of P_n , allows them to show that pretty good state transfer happens between the extreme vertices of P_n for the Laplacian matrix if and only if n is a power of 2.

1.3 Contributions

This master thesis will develop in the following manner. In Chapter 2, we will show one exact algorithm that can be used for deciding whether pretty good state transfer occurs between two vertices. Moreover, we will show that the same tools can be used to compute other quantum objects related to quantum walks.

From Section 2.1 to Section 2.3, we show the tools already known in the literature that we used in the algorithm. In Section 2.4, we show the algorithm. In Section 2.5,

we show another application of the tools we used for the algorithm, now to compute the average mixing matrix.

The algorithm we will show can have time exponential on the size of the graph. For this reason, in Chapter 3, we will show another path that could be further developed and investigated to decide pretty good state transfer using continued fraction as approximations of the eigenvalues. This idea came from one example found in Godsil [2015]. We show known algorithms to compute real roots of polynomials that could be used to compute the eigenvalues in terms of continued fractions and how we can use them. We do not find a characterization for state transfer in terms of them, but we show some examples that can be used to elucidate how state transfer could be studied in this framework.

In Chapter 4, we will show some results both for perfect and pretty good state transfer for the Normalized Laplacian Matrix. Some tools used in this chapter gave us ideas for our algorithm for pretty good state transfer. We manage to show results previously known for the adjacency matrix and the Laplacian matrix, but now in the context of the Normalized Laplacian.

In Section 4.2, we show a relation between cospectral vertices in the Normalized Laplacian and classical random walks. In Section 4.3, we show a characterization for perfect state transfer in the Normalized Laplacian. In Section 4.4, we use this characterization to show conditions for perfect state transfer in trees.

Finally, in Section 4.5, we show a characterization of pretty good state transfer in paths with respect to the Normalized Laplacian.

Chapter 2

Deciding Pretty Good State Transfer

2.1 Conditions to test Pretty Good State Transfer

In Chapter 1, we presented what the literature knows about perfect state transfer and pretty good state transfer. We saw that the former has a useful characterization, meaning that from it we can extract an exact polynomial algorithm to determine whether it happens between two vertices in a graph and at what time it happens.

As for the latter, we also saw that there is a characterization to it. However, differently from the former, an exact algorithm has not been found yet. One of the goals for this thesis is to produce an algorithm that determine if pretty good state transfer happens between two vertices in a given graph.

Now that we have a characterization for pretty good state transfer, we can think of ways of determining if pretty good state transfer happens or not. The first observation we might make is that the first condition of Theorem 1.8 is the same as the one in Theorem 1.6. Since the occurrence of perfect state transfer can be verified classically in polynomial time, we consider this condition to be dealt with.

Now, the second condition does not give us much in terms of how to determine whether it happens or not. One way we may see this is, in order for us to test this condition, we need a convenient way of expressing the eigenvalues of the matrix in the support of the vertices so that we can test all integer linear combinations of them to see if the condition holds. One difficulty of doing that is because, in general, the graph does not need to have integer nor rational eigenvalues, so testing linear combination of irrational values can be prone to precision errors.

There were two main ways we initially proposed to investigate this. The first was to get the approximate eigenvalues via numerical methods and try to “guess” linear integer relations between them. After that, use a linear system to get the general solution and test this condition.

The main problem in doing that is how to guess these linear integer relations, and how much can we trust this guess. The more naïve way is to get each root approximated at d decimal places. Then, make a column vector out of each one, where each entry is a decimal place, and then, use matrix operations to guess linear relations between them. This is prone to precision and arithmetic errors, so we proceeded to find other alternatives.

There are algorithms that produce integer linear relations using approximations of real values. For example, PSLQ in Ferguson and Bailey [1992] shows an algorithm that can extract linear integer relations. One line of investigation that could be pursued is to see how reliable and how general are the solutions that it produces.

Since we needed all possible solutions to the linear equations defined in Theorem 1.8, and due to the possible problems with precision errors, we decided not to continue in this path.

The second line of investigation we initially proposed to pursue would be to find new characterizations or conditions for pretty good state transfer to occur. For instance, Eisenberg et al. [2019] found a non-trivial sufficient condition. First, they define how we can get the minimal polynomial of the eigenvalues in the support.

Definition 2.1 (Minimal polynomial relative to a vector). If M is a symmetric matrix with entries in a field \mathbb{F} , we define $p(x)$ to be the minimal polynomial relative to a vector z if $p(x)$ is the smallest degree polynomial such that $p(M)z = 0$. It is also required that $p(x)$ is monic.

Definition 2.2 (Minimal polynomials of the support). For vertices a and b , let P_+ be the minimal polynomial of M relative to $e_a + e_b$ and P_- be the minimal polynomial of M relative to $e_a - e_b$.

With these definitions, they give the characterization of pretty good state transfer below. Note how this characterization is essentially the same as in Theorem 1.8, but now using these polynomials P_+ and P_- .

Lemma 2.3 (Lemma 2.10 of Eisenberg et al. [2019]). *Let a, b be vertices of a graph represented by the symmetric matrix M . Then pretty good state transfer from a to b occurs if the following two conditions are satisfied:*

1. The vertices a and b are strongly cospectral.
2. Let $\{\lambda_i\}$ be the roots of P_+ and $\{\mu_j\}$ be the roots of P_- . Then for any choice of integers l_i, m_j such that

$$\sum_i l_i \lambda_i + \sum_j m_j \mu_j = 0 \quad (2.1)$$

$$\sum_i l_i + \sum_j m_j = 0, \quad (2.2)$$

we have

$$\sum_j m_j \text{ is even.} \quad (2.3)$$

Finally, with that characterization, they were able to give the following sufficient condition for pretty good state transfer to occur.

Theorem 2.4 ([Eisenberg et al., 2019, Theorem 2.11]). *Let M be a symmetric matrix representing a graph G with strongly cospectral vertices $a, b \in V(G)$. Assume also that P_+ and P_- are irreducible polynomials. Then if*

$$\frac{\text{Tr}(P_+)}{\deg(P_+)} \neq \frac{\text{Tr}(P_-)}{\deg(P_-)}, \quad (2.4)$$

where Tr denotes the trace, i.e., sum of the roots of a polynomial, then there is pretty good state transfer from a to b .

This proposition gives us a simple sufficient condition to check if there is pretty good state transfer in a graph. One point to note in this theorem is that it is assumed that both P_+ and P_- are irreducible. This may not be true in general. At the same time, just removing this condition is not enough to make an if and only if theorem.

The work of van Bommel [2020] shows this by the following example.

Example 2.5 ([van Bommel, 2020, Example 4.1]). For the graph P_8 , the end-vertices are strongly cospectral, but pretty good state transfer does not occur between them. At the same time, $P_+ = (x-1)(x^3-3x-1)$ and $P_- = (x+1)(x^3-3x+1)$ are reducible polynomials such that the condition in Theorem 2.4 holds.

In his paper, he proceeds to show conditions similar to the one above that rule out pretty good state transfer. For example, he shows the following theorem, which removes the condition for P_+ and P_- to be irreducible.

Theorem 2.6 ([van Bommel, 2020, Theorem 4.2]). *Let G be a graph, let a and b be strongly cospectral vertices of G , and suppose P_+ and P_- have (possibly trivial) factors f_+ and f_- of odd degree. Then if*

$$\frac{\text{Tr}(f_+)}{\deg(f_+)} = \frac{\text{Tr}(f_-)}{\deg(f_-)}, \quad (2.5)$$

then there is no pretty good state transfer from a to b .

In the same paper, he shows other conditions that rule out pretty good state transfer, but none is applicable for all graphs. In summary, there are no known simple necessary and sufficient condition to test if pretty good state transfer happens or not.

2.2 Computing the Splitting Field

We now proceed to propose a new way of approaching the problem of testing if condition 2 of Theorem 1.8 holds. First, we show this through one example. We already know that (see Godsil et al. [2012]) there is pretty good state transfer in the end-nodes of a path on 4 vertices with respect to the adjacency matrix. So, we show how one could compute it. Its eigenvalues are

$$\theta_1 = \frac{1}{2}(\sqrt{5} + 1), \quad \theta_2 = \frac{1}{2}(\sqrt{5} - 1), \quad \theta_3 = \frac{1}{2}(-\sqrt{5} + 1), \quad \theta_4 = \frac{1}{2}(-\sqrt{5} - 1).$$

Moreover, following Lemma 2.3, θ_1, θ_3 are roots of P_+ and θ_2, θ_4 are roots of P_- . Therefore, we need to verify if for all integers ℓ_1, ℓ_2 and m_1, m_2 such that

$$\ell_1\theta_1 + \ell_2\theta_3 + m_1\theta_2 + m_2\theta_4 = 0 \quad (2.6)$$

$$\ell_1 + \ell_2 + m_1 + m_2 = 0, \quad (2.7)$$

then $m_1 + m_2$ is even. Note that the first equation can be reorganized as

$$\frac{1}{2}(\sqrt{5}(\ell_1 - \ell_2 + m_1 - m_2) + (\ell_1 + \ell_2 - m_1 - m_2)) = 0. \quad (2.8)$$

This, in turn, can be rearranged in two equations

$$\ell_1 - \ell_2 + m_1 - m_2 = 0,$$

$$\ell_1 + \ell_2 - m_1 - m_2 = 0.$$

By adding the first equation to the second, we get that $\ell_1 = m_2$. By subtracting, we get that $\ell_2 = m_1$. Using these equalities into Equation 2.7 we get that $m_1 = -m_2$.

Hence, the sum $m_1 + m_2$ is always equal to zero and therefore is always even. Thus, pretty good state transfer happens between the end-nodes.

In Section 1.2, we showed that in order for us to characterize pretty good state transfer in paths for the Laplacian matrix, the idea was to write the eigenvalues as powers of a root of unity.

Here, again, we wrote the eigenvalues as powers of $\sqrt{5}$, for instance, $\theta_1 = \frac{1}{2}(\sqrt{5}^1 + \sqrt{5}^0)$. With this representation at hand, we could create a new condition for the integer coefficients that we needed to determine.

The problem here lies on the fact that we could only do that by knowing exactly the eigenvalues and how to express them. Since the eigenvalues in general could be any algebraic numbers, this is not trivial. Therefore, our work here is to create an algorithmic way to both represent the eigenvalues exactly and with that representation create a system that can be efficiently solved to determine whether pretty good state transfer happens.

So we will use an alternative representation of the eigenvalues of the matrix representing the graph. This is what we discuss below, but let us define how can we extend the field of rationals to create a larger field that contains the eigenvalues.

Definition 2.7 (Algebraic field extension). We say that $\mathbb{Q} \subset \mathbb{Q}(\alpha)$ is an algebraic field extension of \mathbb{Q} if $\mathbb{Q}(\alpha)$ is a field and $\alpha \in \mathbb{Q}(\alpha)$. Moreover, $\alpha \notin \mathbb{Q}$ and there is a rational polynomial $p(x)$ such that $p(\alpha) = 0$.

Definition 2.8 (Splitting field). For a monic polynomial $p(x) \in \mathbb{Z}[x]$ with distinct roots $\theta_1, \theta_2, \dots, \theta_n$ over \mathbb{C} , we say that $\mathcal{F} = \mathbb{Q}(\theta_1, \theta_2, \dots, \theta_n)$ is the splitting field of $p(x)$.

The splitting field of a polynomial $p(x)$ is the smallest field extension of \mathbb{Q} such that $p(x)$ factors completely.

One of the fundamental theorems of Galois Theory, the Primitive Element Theorem, see for instance [Cox, 2012, Theorem 5.4.1], is that for monic integer polynomials their splitting field \mathcal{F} is a simple field extension, meaning, $\mathcal{F} = \mathbb{Q}(\alpha)$ for some $\alpha \in \mathbb{C}$.

This theorem will be key in our work on finding an algorithmic way of deciding whether pretty good state transfer occurs. The idea is the following. We consider the monic polynomial of the eigenvalue support, say $f(x)$, compute the primitive element of its splitting field $\mathcal{F} = \mathbb{Q}(\alpha)$ and then we factor $f(x)$ over \mathcal{F} . This gives us all factors of $f(x)$ as linear factors, and the constant coefficient of each factor is defined as a linear combination of the powers of α .

Now, we take these coefficients and, together with the characterization of pretty good state transfer in Theorem 1.8, define a linear system, compute its general solution

and see if it is possible to extract some condition on the coefficients of the general solution that determine if condition 2 of Theorem 1.8 is satisfied or not.

There are known algorithms that compute the splitting field. In Landau [1985], one such algorithm is shown to be polynomial in the size of the splitting field and the polynomial that we want to split. We make use of this algorithm and others in this paper to make our algorithm to test if pretty good state transfer happens or not. Some results that we discuss below are also present in Trager [1976]. There he shows some algorithms that can be used to compute splitting fields of polynomials. We focus however on Landau's paper, as it shows that it all can be done in polynomial time of the polynomials given and some algebraic extension of the rationals.

Let us first explain the idea behind the main algorithm that we use for computing the splitting field. First, the paper defines in [Landau, 1985, Section 5] an algorithm for the factorization of a polynomial $f(x)$ over a simple extension. This algorithm is a key part of the algorithm to compute the splitting field. Moreover, it is also shown that this factorization is polynomial in the size of $f(x)$ and the minimal polynomial of the extension. The factorization algorithm is also shown in Geddes et al. [1992].

First, we need a concept that the author uses for the algorithm of factorization and for computing the splitting field: the norm of a polynomial. For a polynomial $f(x) \in \mathbb{Q}(\alpha)[x]$, we can view it as a polynomial in the two variables x and α and write it as $f_\alpha(x)$.

Definition 2.9 (Algebraic conjugates). Let $\alpha \in \mathbb{C}$ be algebraic and $g(x)$ be its minimal polynomial over $\mathbb{Q}[x]$ with degree n . Then, we say that $\alpha_2, \alpha_3, \dots, \alpha_n$, the other distinct roots of $g(x)$ over \mathbb{C} , are the algebraic conjugates, or conjugates for short, of α .

Definition 2.10 (Norm of polynomials). Let $f(x) \in \mathbb{Q}(\alpha)[x]$. Let also $\alpha_2, \alpha_3, \dots, \alpha_n$ be the conjugates of $\alpha = \alpha_1$ over \mathbb{Q} . Then, the norm $N(f(x)) \in \mathbb{Q}(x)$ of $f(x)$ is defined to be

$$N(f(x)) = \prod_i f_{\alpha_i}(x). \quad (2.9)$$

Thus, we can note that $f(x)$ divides $N(f(x))$. Note that $N(f(x))$ is indeed rational, as it is explained in Landau [1985].

In order for us to compute the norm of the polynomial in $\mathbb{Q}(\alpha)$, we would need to find all conjugates of α and then compute the formula we defined above. For a more practical way, the author uses another object that we define below.

Definition 2.11 (Sylvester matrix). Let $p(x) = \sum_{i=0}^m p_i x^i$ and $q(x) = \sum_{j=0}^n q_j x^j$, then the Sylvester matrix of the polynomials, denoted by $S_{p(x),q(x)}$, is a $(n+m) \times (n+m)$ matrix that has:

- $(p_m, p_{m-1}, \dots, p_0, 0, \dots, 0)$ as first column,
- $(0, p_m, p_{m-1}, \dots, p_0, 0, \dots, 0)$ as second column and this “shift” in the rows repeats for the first n columns,
- $(q_n, q_{n-1}, \dots, q_0, 0, \dots, 0)$ as the $n+1$ -th column,
- $(0, q_n, q_{n-1}, \dots, q_0, 0, \dots, 0)$ as the $n+2$ -th column and, again, this shift repeats now for the final m columns.

Definition 2.12 (Resultant of polynomials). Let $p(x), q(x)$ be polynomials as before, then the resultant of $p(x)$ and $q(x)$ according to the variable x is

$$\text{Res}_x(p(x), q(x)) = \det(S_{p(x),q(x)}). \quad (2.10)$$

Another way we can define the resultant that is also present in Landau [1985] is the following. Let $\mu_1, \mu_2, \dots, \mu_m$ be roots of $p(x)$. Then,

$$\text{Res}_x(p(x), q(x)) = p_m^n \prod_i^m q(\mu_i). \quad (2.11)$$

The equation above allows us to compute the norm of polynomials using the formula defined in Equation 2.9. Let $g(x)$ with degree m be the minimal polynomial of α and α_i be the conjugates of α . Let also $f(x)$ with degree n be the polynomial we want to compute the norm, then the following equalities hold

$$\begin{aligned} N(f(x)) &= \prod_i f_{\alpha_i}(x) \\ &= \frac{g_m^n \prod_i f(\alpha_i, x)}{g_m^n} \\ &= \frac{\text{Res}_t(g(t), f(t, x))}{g_m^n}. \end{aligned} \quad (2.12)$$

If $g(x)$ is monic, that is, $g_m = 1$, then $N(f(x)) = \text{Res}_t(g(t), f(t, x))$. Knowing that, we now know how to compute the norm of the polynomial. The key result for the factorization algorithm presented in the paper is the following.

Theorem 2.13 ([Landau, 1985, Theorem 1.5]). *Let $f(x) \in \mathbb{Q}(\alpha)[x]$ be such that $N(f(x))$ is square-free. Then if $N(f(x)) = \prod_i G_i(x)$ is a factorization into irreducible*

polynomials in $\mathbb{Q}[x]$, then $f(x) = \prod_i \gcd(f(x), G_i(x))$ is a factorization into irreducible polynomials in $\mathbb{Q}(\alpha)[x]$.

So, if we want to factor a polynomial $f(x)$ in $\mathbb{Q}(\alpha)[x]$ over this ring, as long as its norm is square-free, we just need to take the gcd of the irreducible polynomials of its norm. The algorithm on Landau's paper spends a few steps into assuring that the original polynomial, not only its norm, is also square-free. We need not worry about this, as the polynomials $P_+(x)$ and $P_-(x)$ are square-free, as it is shown in [Kempton et al., 2020, Lemma 2.5]. So we omit some details of the full factorization algorithm that deal with repeated roots of the initial polynomial we want to factor.

One step we still have to deal with is that as much as the initial polynomial is square-free, its norm could not be. So we need to transform $f(x)$ in such a way that its norm is square-free, and we can get it back after it. For that, Landau's paper gives us the following lemma.

Lemma 2.14 ([Landau, 1985, Lemma 1.6]). *Let $f(x) \in \mathbb{Q}(\alpha)[x]$ be a square-free polynomial of degree n , where $[\mathbb{Q}(\alpha) : \mathbb{Q}] = m$. Then, there are at most $(nm)^2/2$ integers s such that $N(f(x - s\alpha))$ is not square-free.*

This shows us that we can compute the norm of $f(x - s\alpha)$ for different values of s until we get a $N(f(x - s\alpha))$ that is square-free, and that the number of attempts should be at most polynomial on the size of $f(x)$ and the extension. Thus, we can define the following pseudocode.

Algorithm 2.15 Compute square-free norm (sqrfnorm)

Input: $f(x) \in \mathbb{Q}(\alpha)[x]$ and $g(x)$ minimal polynomial of α over \mathbb{Q}

Output: s such that $p(x) = N(f(x - s\alpha))$ is square-free and $p(x)$

$s = \text{nextInteger}()$

$p(x) = \text{norm}(t, f(x - st), g(t))$

while $\text{isSquareFree}(p(x)) == \text{False}$ **do**

$s = \text{nextInteger}()$

$p(x) = \text{norm}(t, f(x - st), g(t))$

end while

return $(s, p(x))$

One important thing to notice is the function *nextInteger*. It just returns a different integer every time. One way of doing this is returning the previous integer +1, and it can always start at some default initial integer at the beginning of the algorithm.

In a more practical way, as we know by Lemma 2.14, the number of integers that make the norm not square-free is polynomial in the size of the extension and the polynomial we are trying to compute. So, if you have an integer sample set large enough you could pick randomly, and you should expect to find a suitable integer in a few tries.

Also, by our previous explanation, we can compute the norm of $f(x - st)$ by simply computing the resultant. So we write the *norm* function receiving the variable that the resultant is applied over and the two polynomials that it will be applied to. It is easy to see that it can be computed in polynomial time, as it is the determinant of the Sylvester matrix of the two polynomials.

Now that we know that by successive transformations of $f(x)$ into $f(x - st)$, we can find some integer s such that the norm $N(f(x - st))$ is square-free, we can apply Theorem 2.13. We know from this theorem that if we factor $N(f(x - st))$ into irreducible polynomials $G_i(x)$ over \mathbb{Q} , then $\gcd(G_i(x), f(x - st))$ is an irreducible polynomial factor of $f(x - st)$. So, we just change the variables and take $\gcd(G_i(x + st), f(x))$ to get an irreducible polynomial factor of $f(x)$.

This algorithm can be represented by the following pseudocode. A similar version for polynomials with repeated roots is contained in Landau [1985]. Before that, we give two more definitions we need.

Definition 2.16 (Algebraic integer). We say that α is an algebraic integer if it is a root of an integer monic polynomial.

Definition 2.17 (Ring of algebraic integers). Let $K = \mathbb{Q}(\alpha)$ be an algebraic field extension. We define O_K to be the ring formed by the set of algebraic integers in K .

Algorithm 2.18 Factorization over $\mathbb{Q}(\alpha)$

Input: $g(t) \in \mathbb{Z}[t]$, monic, irreducible.
 $f(x, t) \in \mathbb{Q}[x, t]$ square-free polynomial with coefficients in O_K , $K = \mathbb{Q}[t]/(g(t))$
Output: $f_i(x)$ the irreducible factors of $f(x, t)$ over $O_K[x]$
 $s, p(x) = \text{sqrtnorm}(f(x, t), g(t))$
Factor $p(x) = \prod_i G_i(x)$ over \mathbb{Q}
for each factor $G_i(x)$ **do**
 make $f_i(x) = \gcd_{\mathbb{Q}[t]/g(t)}(G_i(x + st), f(x, t))$
end for
return $\{f_i(x)\}$ \triangleright We view $f_i(x)$ as a polynomial in x with coefficients over K

The function *sqrtnorm* is the algorithm we defined in 2.15. It is also important to notice that the gcd is taken over $\mathbb{Q}[t]/g(t)$. We omit how to do this as it is not the focus

of our paper, but Landau's paper explains how one may do this computation and also shows that it can be done in polynomial time on the polynomials and the extension we are computing the gcd over. We also need to factor $p(x)$ in $\mathbb{Q}[x]$, but that can also be done efficiently, and that is also explained in Landau's paper.

Now that we know how to factorize a polynomial in some extension, we can proceed to the splitting field algorithm. For that, the paper gives us the following theorem.

Theorem 2.19 ([Landau, 1985, Theorem 1.4]). *Let $f(x) \in \mathbb{Q}(\alpha)[x]$ be irreducible. Then $N(f(x))$ is a power of an irreducible polynomial in $\mathbb{Q}[x]$.*

The results above show us that if we have a polynomial $f(x)$ we only need a polynomial amount of tests to find a s such that $N(f(x - s\alpha))$ is square-free and if $f(x - s\alpha)$ is irreducible, its norm also is an irreducible polynomial. Moreover, the following lemma gives us that if $f(x)$ is an irreducible polynomial, everything else follows.

Lemma 2.20 (Irreducibility of Polynomials). *Let $h(x)$ be an irreducible polynomial in $\mathbb{Q}(\beta)[X]$, and let $\{\alpha_1, \alpha_2, \dots, \alpha_n\}$ be the roots of $h(x)$. Then, $h(x - s\beta)$ is an irreducible polynomial in $\mathbb{Q}(\beta)[X]$, for $s \in \mathbb{Z}$.*

Proof. We will prove the counter-positive of this assertion. Suppose that $h(x - s\beta)$ is reducible over $\mathbb{Q}(\beta)[X]$. Then, there are polynomials $p(x), q(x) \in \mathbb{Q}(\beta)[X]$, with $\min(\deg(p(x)), \deg(q(x))) \geq 1$, such that $h(x - s\beta) = p(x)q(x)$.

But now, consider $p(x + s\beta), q(x + s\beta) \in \mathbb{Q}(\beta)[X]$. They have exactly all the roots of $h(x)$, so $h(x) = p(x + s\beta)q(x + s\beta)$. \square

Now, we know that if we have a factor $h(x)$ of our $f(x)$ that is irreducible over some extension $\mathbb{Q}(\alpha)[x]$, we can find in a polynomial amount of attempts a s such that $g(x) = N(h(x - s\alpha))$ is square-free. This fact gives us that $g(x)$ is the minimal polynomial of $\beta + s\alpha$ for $h(\beta) = 0$ over $\mathbb{Q}[x]$ and thus, we will show in the following lemma that $\mathbb{Q}(\alpha, \beta) = \mathbb{Q}(\beta + s\alpha)$. The proof is an adaptation of the one from Cox [2012] for the Primitive Element Theorem.

Lemma 2.21. *If $h(x) \in \mathbb{Q}[x]$, with β as a root, is irreducible over some extension $\mathbb{Q}(\alpha)[x]$ with $u(x)$ being the minimal polynomial of α and $g(x) = N(h(x - s\alpha))$ is square-free for some integer s , then $\mathbb{Q}(\alpha, \beta) = \mathbb{Q}(\beta + s\alpha)$.*

Proof. By Theorem 2.19 we know that $g(x)$ is a power of an irreducible polynomial and being square-free means that it is an irreducible polynomial with no repeated roots.

Moreover, by the definition of the norm, $h(x - s\alpha)|g(x)$. This means that $\beta + s\alpha$ is a root of $g(x)$ and, thus, $g(x)$ is the minimal polynomial of $\beta + s\alpha$. Now, we only need to show that either α or β is in $\mathbb{Q}(\beta + s\alpha)$ as clearly $\mathbb{Q}(\beta + s\alpha) \subset \mathbb{Q}(\beta, \alpha)$.

If β_i are the roots of $h(x)$ and α_j are the roots of $u(x)$, then, by the definition of the norm, $\beta_i + s\alpha_j$ are roots of $g(x)$. As it has no repeated roots, we have that $s \neq \frac{\beta_i - \beta_l}{\alpha_k - \alpha_j}$ for $k \neq j$. More specifically, if $\alpha = \alpha_1$ and $\beta = \alpha_1$, then $\beta + s\alpha \neq \beta_i + s\alpha_j$ for i free, but $j \neq 1$.

Note that $u(x)$ and $h(\beta + s\alpha - sx)$ are in $\mathbb{Q}(\beta + s\alpha)[x]$ and vanish at α . Now, take the $\gcd(u(x), h(\beta + s\alpha - sx)) = q(x) \in \mathbb{Q}(\beta + s\alpha)[x]$. Then, for some $A(x)$ and $B(x)$, we have the following

$$A(x)u(x) + B(x)h(\beta + s\alpha - sx) = q(x). \quad (2.13)$$

Now, if we apply α to the equation above, we can see that the left-hand side is equal to 0. So the $q(x) \neq c$ for some constant $c \neq 0$.

So $q(x)$ has degree greater than 0. If $\deg(q(x)) > 1$, then, as $q(x)|u(x)$, some α_j with $j > 1$ is a root of $u(x)$. But then α_j must also be a root of $h(\beta + s\alpha - sx)$, which means that $\beta + s\alpha - s\alpha_j = \beta_i$. This is a contradiction.

So, $q(x) = x - \alpha \in \mathbb{Q}(\beta + s\alpha)[x]$, which means that $\alpha \in \mathbb{Q}(\beta + s\alpha)$. Furthermore, as $s \in \mathbb{Z}$, $\beta \in \mathbb{Q}(\beta + s\alpha)$. \square

We can show a simple example of the procedure of the lemma above to find the minimal polynomial of a larger extension. Suppose that we wish to factorize $h(x) = x^2 - 3$, the minimal polynomial of $\beta = \sqrt{3}$ over $\mathbb{Q}[x]$, and we already have the minimal polynomial $g(x) = x^2 - 2$ of $\alpha = \sqrt{2}$. Then, we need to compute the $N(f(x))$ over $\mathbb{Q}(\alpha)$, where $f(x) = h(x - s\alpha)$ for some integer s .

We could test different values of s until we get a square-free norm, as by Lemma 2.14 we know that we only need to test a polynomial on $\deg(h(x))$ and $\deg(g(x))$ amount of values of s . But one can easily check that, for $s = 1$, we have the following

$$\frac{-(\sqrt{2} + \sqrt{3})^3 + 11(\sqrt{2} + \sqrt{3})}{2} = \sqrt{3}, \quad (2.14)$$

$$\frac{(\sqrt{2} + \sqrt{3})^3 - 9(\sqrt{2} + \sqrt{3})}{2} = \sqrt{2}. \quad (2.15)$$

So, we compute $N(h(x - \alpha))$. By Equation 2.12, we define both $g(t) = t^2 - 2$ and $f(t, x) = (x - t)^2 + 3$ to compute $Res_t(g(t), f(t, x))$. Now, first we show the Sylvester

matrix for this resultant

$$S_{g(t),f(t,x)} = \begin{bmatrix} 1 & 0 & 1 & 0 \\ -2x & 1 & 0 & 1 \\ -3+x^2 & -2x & -2 & 0 \\ 0 & -3+x^2 & 0 & -2 \end{bmatrix}. \quad (2.16)$$

Finally, the resultant, as $g(x)$ is monic, is just the determinant of the Sylvester matrix

$$\text{Res}_t(g(t), f(t, x)) = x^4 - 10x^2 + 1, \quad (2.17)$$

which can be easily checked to be the minimal polynomial of $(\sqrt{2} + \sqrt{3})$ over $\mathbb{Q}[x]$. Therefore, if we try to factor $h(x)$ over $\mathbb{Q}(t)$, where $t = \sqrt{2} + \sqrt{3}$, using the expressions we showed in Equation 2.14 we can see that we get $h(x, t) = (x - (-t^3 + 11t)/2)(x - (t^3 - 11t)/2)$.

We can see that in this whole procedure we never used the values of α , only its minimal polynomial, and used instead a new variable in the polynomial we wanted to compute the norm. So this whole computation is exact. The same for the factorization algorithm that uses the resultant.

So, with all these results above and after this explanation on why to use and compute the norm above, we can finally explain the result that we needed from Landau's paper. It is the following corollary.

Corollary 2.22 ([Landau, 1985, Corollary 6]). *Let $f(x)$ be a polynomial in $\mathbb{Z}[x]$. Let F be the splitting field of $f(x)$. Then, it can be determined in time polynomial in $[F : \mathbb{Q}]$ and $\log|f(x)|$.*

In the corollary above, by determine the splitting field, Landau means that we can compute the minimal polynomial of the primitive element of the splitting field and also factor $f(x)$ over $F[x]$.

We can view the general idea of the algorithm in the pseudocode below. *Factor* is a function that receives two polynomials $f(x)$ and $g(t)$. If $g(t) = 0$, then it factors the $f(x)$ over \mathbb{Q} . Otherwise, it calls the algorithm 2.18 that factorizes the $f(x)$ over $\mathbb{Q}[t]/g(t)$. It is important to remember that while it assumes that $f(x) \in \mathbb{Q}[x, t]$, polynomials in $\mathbb{Q}[x]$ are in $\mathbb{Q}[x, t]$ as well.

Another thing to notice is that the Algorithm 2.18 requires that $g(t)$ is a monic integer irreducible polynomial. The first time it is clear that it is the case as it is one of the factors of $f(x)$. But to see that it is still the case after we make $g(t)$

the minimal polynomial of a larger extension, note that we are making extensions over algebraic integers and the linear combination of an algebraic integer is also an algebraic integer. So, even though, *sqrtnorm* could return a rational polynomial, by construction it returns an integer polynomial.

Algorithm 2.23 Compute Splitting Field and Factor Polynomial

Input: $f(x) \in \mathbb{Q}[x]$
Output: factors of $f(x)$ over its splitting field $\mathbb{Q}[\alpha]$ and minimal polynomial of α
 $factors = \text{factor}(f(x), 0)$ \triangleright factor $f(x)$ over \mathbb{Q}
 $g(x) = NULL$
while hasNonLinearFactors(factors) **do**
 $h(x) = \text{getNonLinearFactor}(factors)$
 if $g(x) == NULL$ **then**
 $g(x) = h(x)$
 else
 $s, g(x) = \text{sqrtnorm}(h(x), g(x))$
 end if
 $factors = \text{factor}(f(x), g(x))$ \triangleright factor $f(x)$ over $\mathbb{Q}(t)/g(t)$
end while
return (factors, $g(x)$)

2.3 Solving Diophantine Linear Systems

Before we move into deciding pretty good state transfer, we need to show how to solve Diophantine linear systems. In Theorem 1.8, we showed the characterization of pretty good state transfer in terms of a system of linear equations over the eigenvalue support of the matrix. Moreover, the coefficients of the linear system are assumed to be integers. Here, we show how one may approach the problem of solving Diophantine linear systems of equations.

In general, a system of m linear equations on n variables can be expressed as

$$\sum_{j=1}^n a_{ij}x_j = b_i, \text{ where } 1 \leq i \leq m. \quad (2.18)$$

We can also view this linear system in matrix form as

$$Ax = b, \quad (2.19)$$

where A is a matrix with entries a_{ij} , b and x are vectors with entries b_i , x_j respectively. If $b = 0$, then we say that the system is homogeneous. If our system has the property

that the entries of A and b are integers, and we want to find only solutions x with integer entries, we say that the system is a Diophantine system of linear equations.

Diophantine equations are a particular kind of system of equations in which we ask for solutions over the integers. They are famous because not only they arise naturally in various areas, but because some types of Diophantine equations are known for being difficult to solve or having no solution at all, like the famous equation in Fermat's Last Theorem.

One of the main ideas to solve a general linear system is to transform a system of linear equations into row reduced echelon form. In the case of Diophantine linear systems we still use this technique, but we need to extend the usual row reduced definition. Before we show how to solve it, we define what we require for a matrix to be in row reduced echelon form. This definition was taken from Storjohann [2000].

Definition 2.24 (Row echelon and reduced row echelon forms). A m by n upper triangular matrix A with entries a_{ij} is said to be in row echelon form if:

1. Let r be the rank of A , then only the first r rows of A are non-zero.
2. For $1 \leq i \leq r$, let a_{ij_i} be the first non-zero entry of the row i . Then $j_1 < j_2 < \dots < j_r$. We call $[j_1, j_2, \dots, j_r]$ the rank profile of A .

Furthermore, if besides these two conditions A also follows the two conditions below, we say that A is in row reduced echelon form:

1. $a_{i,j_i} > 0$ for $1 \leq i \leq r$.
2. For $1 \leq k < i \leq r$, $a_{ij_i} > a_{kj_i} \geq 0$.

Definition 2.25 (Unimodular matrix). A matrix U is said to be unimodular if it is a square integer matrix with determinant equal to ± 1 .

In general, the row reduced echelon form also asks for the first non-zero entries of each row to be the identity of the field we are dealing with. As we want all the entries to be integers, we cannot always ask for this. Also, if we were computing the matrix over a field, in the last condition of the definition of the row reduced echelon form, we normally would ask for the $a_{kj_i} = 0$.

If the matrix A is a square non-singular integer matrix in row reduced echelon form, we say that A is in Hermite normal form. Some authors also use this name for general matrices in row reduced echelon form, but this is not standard, so we use it

only for the original case. There are more ways to extend the Hermite normal form to matrices with arbitrary rank and dimensions, but these may not be unique.

This is different from this definition of row reduced echelon form, as it is a canonical form of row equivalence. The row echelon form is not unique, but the rank profile defined by it is unique. The positions a_{ij_i} for $1 \leq i \leq r$ are called pivot positions. These positions are unique even for the row echelon form over the rationals as the only constraint we need to add is that the pivots are equal to 1, but this scaling of the rows do not change the rank profile. A discussion on this can be found in [Storjohann, 2000, section 1.4] and also at Anton and Rorres [2013].

One important feature of this row reduced echelon form is that if H is the row reduced echelon form of a matrix A , then there is a unimodular matrix U such that $UA = H$. The rows of the unimodular matrix define the standard row operations we apply on H : multiplying a row for ± 1 , adding a multiple of a row into another and switching two rows.

Note that we cannot divide or multiply a row by an integer as the matrix U would not be unimodular, but if we use the notion of row reduction and echelon form over the rationals we are allowed to perform these operations. The difference is that now U would not be required to be unimodular.

Given all these definitions, we can use it to solve systems of Diophantine linear equations. Let A be a m by n integer matrix with rank r and a system defined as $Ax = b$, b an integer vector. One simple way of doing this is to take an initial system A , transform it into row reduced echelon form, and then compute a solution if the system has one. This is easier to do as the final matrix is upper triangular. Note that the row operations applied in A must also be applied at b .

This is enough if one wants a solution to the system, but if we want all the possible solutions, this is not the best way of doing it. The idea for this part was taken from [Storjohann, 2000, section 5.3], but we explain it below. He improves the idea given in Blankinship [1966].

If we want all possible solutions to the system $Ax = b$, we need one solution for the system, say x_p . We call it a particular solution. We also need a n by $n - r$ matrix Y such that $AY = 0$. The columns of Y are linear independent vectors, that define all the solutions to the homogeneous system $Ax = 0$. More specifically, any integer linear combination of the columns of Y is a solution to the homogeneous Diophantine system. The pair (x_p, Y) forms the general solution to the system. If there is only one particular solution for the system, Y is a zero matrix.

In order for us to get all the solutions, just computing the row reduced echelon form of A is not enough, we need to tweak a little this procedure. First, create a matrix

B as follows

$$B = \left(\begin{array}{c|c} -b^T & 1 \\ \hline A^T & 0 \end{array} \right). \quad (2.20)$$

Then, compute its row echelon form T . We will say from now on that the rank of $B = r$, where $r = \text{rank}(A) + 1$. Now, from the discussion we had above one can see that there is an unimodular matrix U such that

$$UB = \left(\begin{array}{c|ccc} d_1 & * & \dots & * \\ d_2 & * & \dots & * \\ \vdots & & & \vdots \\ \hline d_r & & & x_p^T \\ \hline 0 & & & Y^T \end{array} \right) \left(\begin{array}{c|c} -b^T & 1 \\ \hline A^T & 0 \end{array} \right) = T = \left(\begin{array}{ccccc} * & * & \dots & * & d_1 \\ 0 & * & \dots & * & d_2 \\ \vdots & & & * & \vdots \\ 0 & 0 & \dots & * & d_r \\ \hline 0 & 0 & \dots & 0 & 0 \\ \vdots & & & & \vdots \\ 0 & 0 & \dots & 0 & 0 \end{array} \right). \quad (2.21)$$

Now, if $j_r = m + 1$, it is easy to see that Y and x_p as in above form the general solution pair for $Ax = d_r b$, but we want an integer solution for $d_r = 1$. That is a problem that is dealt with in the following lemma.

Lemma 2.26 (Blankinship [1966]). *For the system $Ax = b$, let B be as in Equation 2.20 and let U be an unimodular matrix such that UB is in row echelon form. Then, the system has a rational solution if and only if $j_r = m + 1$. Moreover, the system has an integer solution if and only if $|d_r| = 1$. If $|d_r| = 1$, then the general solution is a pair $(d_r x_p, Y)$, where they are as in Equation 2.21 and if T has full row rank, then $Y = 0$.*

Proof. If $j_r = m + 1$, clearly, the system has a rational solution, just look at the r -th row of U and divide x_p by d_r . For the other direction, say the system has a rational solution. Take A^T and transform it into row echelon form. Let U' be the unimodular matrix such that $U'A^T$ is in row echelon form. Let also x' be a rational solution to $Ax' = b$. Now, create a matrix U'' as in below

$$U'' = \left(\begin{array}{c|c} 0 & U'_1 \\ \vdots & \\ \hline 1 & x' \\ \hline 0 & U'_2 \\ \vdots & \end{array} \right), \quad (2.22)$$

where U'_1 is the matrix that has the first $r - 1$ rows of U' and U'_2 has the rest of the rows. Now, $U''B$ is in row echelon form over the rationals with rank profile such that $j_r = m + 1$, and as the rank profile is unique over the rationals and integers, this shows that if $Ax = b$ has a rational solution, then $j_r = m + 1$.

Now, suppose that $j_r = m + 1$, meaning the system has a solution over the rationals. We want to show that $|d_r| = 1$ if and only if the system has an integer solution. If $|d_r| = 1$, clearly, the system has an integer solution. Just take $d_r x_p$ as by Equation 2.21.

If the system has an integer solution, then let x' be this integer solution. Thus, there is an integer linear combination of the $(n - r)$ last rows of U defined by a vector v , such that

$$\begin{pmatrix} d_r \\ x_p \end{pmatrix}^T - v^T \begin{pmatrix} 0 & Y^T \end{pmatrix} = d_r \begin{pmatrix} 1 \\ x' \end{pmatrix}^T. \quad (2.23)$$

Now, this transformation of $(d_r \ x_p)^T$ into $d_r(1 \ x')$ in the matrix U can be expressed as an unimodular matrix U' . But now the gcd of the entries of the r -th row of $U'U$ is equal to d_r . This means that the determinant of $U'U$ is a multiple of d_r . Because $U'U$ is unimodular, it follows that $|d_r| = 1$.

Everything else follows. If T has full rank, we can define that $Y = 0$, as the system has at most one solution. \square

This is known in the literature as one of the ways to solve Diophantine linear systems. We only need now to show how to do it efficiently. Algorithms for computing normal forms and row reduced echelon forms over rationals are standard, as they are used in various branches of mathematics.

One problem that arises when dealing with row echelon form over the integers is that the entries of the intermediate matrices that appear during the computation can grow too much in size. This is referred to as the intermediate expression swelling. One solution to this problem is to deal with the intermediate expressions in a residue number system.

One algorithm that solves this problem of intermediate swell, computes the row reduced echelon form and apply it to solving linear Diophantine systems is in the dissertation Storjohann [2000]. There he computes the row reduced echelon form as we defined. He calls it the Hermite normal form.

The author divides the computation of the U we defined in two parts, the first part computes a matrix E that has the first r rows of U such that EA has the non-zero rows of the Hermite normal form. He calls them the Hermite basis. Then, he computes

the matrix M that has the row null space of A . Therefore, our U can be written as follows

$$U = \begin{pmatrix} E \\ M \end{pmatrix}. \quad (2.24)$$

The algorithm presented in the dissertation computes in some steps, from the original matrix A , a square non-singular integer matrix A' such that that A' can be written as

$$A' = \begin{pmatrix} B & 0 \\ D & I_{n-r} \end{pmatrix}, \quad (2.25)$$

where I_{n-r} is the $n-r$ dimensional identity. So, when we refer to B , we are referring to this B that comes from A to form the matrix A' . Now, we write the two propositions of his dissertation that solve efficiently the problem.

Lemma 2.27 ([Storjohann, 2000, Proposition 6.3]). *Let $A \in \mathbb{Z}^{n \times m}$ have rank r . A matrix $E \in \mathbb{Z}^{r \times n}$ such that EA equals the Hermite basis of A can be recovered in $O(nr^{\theta-1} \log \beta + nr \log(r) B(\log(\beta)))$ word operations where $\beta = (\sqrt{r} \|A\|)^r$. At most $r + \lceil \log_2 \beta \rceil$ columns of E will be nonzero and $\|E\| \leq \beta$.*

Lemma 2.28 ([Storjohann, 2000, Proposition 6.6]). *Let $A \in \mathbb{Z}^{n \times *}$ have column space bounded by m and rank \bar{r} bounded by r . Let $\beta = (\sqrt{r} \|A\|)^r$. A nullspace $M \in \mathbb{Z}^{n-\bar{r} \times n}$ for A which satisfies $\|M\| \leq r\beta^2$ can be recovered in $O(nmr^{\theta-2}(\log 2n/r)(\log \beta) + nm(\log n) B(\log \beta))$ words operations.*

In both lemmas, $B(k)$ in the complexity of both algorithms is a function of the form $B(k) = O(k(\log k)^2(\log \log k))$ and θ is a parameter such that two n by n matrices in a ring can be multiplied in $O(n^\theta)$ using operations $(+, -, \times)$ of the commutative ring the entries of the matrices are in. In our case, we can assume that $\theta = 3$.

These two lemmas give what we need for computing the general solution to a Diophantine linear system of equations efficiently. We take A and b and construct the matrix B as in Equation 2.20 and use the two algorithms of the two lemmas above to compute the matrix U such that UA is in row echelon form. Then, using Lemma 2.26 we check if $j_r = m + 1$ in UA and if $|d_r| = 1$. If both validations are true, then we take Y and x_p from U as we defined in Equation 2.21.

Corollary 2.29. *For a Diophantine linear system of equations defined by A and b such that $Ax = b$, if it is consistent, we can compute a pair $(d_r x_p, Y)$ that defines the general solution of the system in polynomial time.*

So with that result, we can compute the general solutions to systems of linear Diophantine equations in polynomial time.

2.4 Algorithm for Deciding Pretty Good State Transfer

In Section 2.2, we showed how to factor $f(x)$ over its splitting field and get the minimal polynomial $g(x)$ that defines the splitting field using the results of Landau [1985]. Now, we will make use of this result to determine if pretty good state transfer happens or not between two vertices in a given graph.

If we make $f(x) = P_+(x)P_-(x)$, using what we showed in the last section, we can factor it as follows

$$f(x) = P_+(x)P_-(x) = \prod_{i=0}^{\deg(P_+(x))-1} (x - p_i(\alpha)) \prod_{j=0}^{\deg(P_-(x))-1} (x - g_j(\alpha)), \quad (2.26)$$

where if λ_i, μ_j are the roots of $P_+(x)$ and $P_-(x)$ respectively, then $p_i(\alpha) = \lambda_i$ and $g_j(\alpha) = \mu_j$, for α the primitive element of the splitting field of $f(x)$.

Note that the algorithm receives one polynomial and splits it over its splitting field. To get the polynomials p_i and g_j separately, simply compute the splitting field of $P_+(x)P_-(x)$ and then, use the minimal polynomial of the splitting field to factor P_+ and P_- separately using Landau's factorization algorithm.

In Section 2.2, we assumed in some algorithms that the polynomial we want to factor is square-free, for this we need that $P_+(x)P_-(x)$ to be square-free. As A is symmetric, this is showed to be solved by the following two lemmas.

Lemma 2.30 ([Eisenberg et al., 2019, Lemma 2.5]). *Given a symmetric matrix M and cospectral vertices $u, v \in V(M)$, the characteristic polynomial of M decomposes as*

$$\phi_M = P_+(x)P_-(x)P_0(x), \quad (2.27)$$

where P_+ and P_- have no multiple roots, and there is an orthonormal basis of eigenvectors such that:

1. for each root λ of P_+ , the basis contains a unique eigenvector φ with eigenvalue λ and $\varphi_u = \varphi_v \neq 0$,
2. for each root λ of P_- , the basis contains a unique eigenvector φ with eigenvalue λ and $\varphi_u = -\varphi_v \neq 0$,

3. for each root λ of P_0 , with multiplicity k , the basis contains exactly k eigenvectors with eigenvalue λ , all of which vanish on both u and v .

Lemma 2.31 ([Eisenberg et al., 2019, lemma 2.8]). *The following are equivalent:*

1. Vertices u and v are strongly cospectral.
2. Vertices u and v are cospectral, and P_+ and P_- do not have any common roots.
3. $E_\lambda e_u = \pm E_\lambda e_v$ for all λ .

This ensures that, as strong cospectrality is one of the conditions of pretty good state transfer, by testing first this condition, the square-free requirement follows. One thing we still need to show is how to get the polynomials P_+ and P_- and preferably in polynomial time.

In the explanation of the PST algorithm in Section 1.2, we showed how to get the polynomial of the eigenvalue support with no repeated roots. This is not enough as in this case we will be getting $P_+(x)P_-(x)$, but not both separately, so we need another way.

By the definition 2.2, we know that $P_+(x)$ and $P_-(x)$ are the minimum polynomials of M relative to $e_u + e_v$ and $e_u - e_v$, respectively. With this definition at hand, we can show that we can compute them in polynomial time.

Lemma 2.32. *Let $P_+(x)$ and $P_-(x)$ be the polynomials as defined in definition 2.2 for the adjacency matrix of a connected graph G with respect to the vertices u, v , and assume $|V(G)| = n$. Then, we compute the polynomials in polynomial time in n .*

Proof. We will show that we can do it for $P_+(x)$. For $P_-(x)$ the procedure is analogous.

Iteratively, keep checking if $A^r(e_u + e_v)$ is in the span of $\{A^k(e_u + e_v)\}_{k=0}^{r-1}$ until it finally is. When this happens, find the linear combination of the rows that makes it so. Note that P_+ is monic, because $P_+ | \phi_M$ as in Lemma 2.30.

□

With these polynomials in hand, we can make the following modification to the characterization in Lemma 2.3.

Lemma 2.33 (Characterization of pretty good state transfer). *Let u, v be vertices of a graph G represented by the integer symmetric matrix M . Let $f(x) = P_+(x)P_-(x)$ and $\mathbb{Q}(\alpha)$ be the splitting field of $f(x)$ and $d(x) \in \mathbb{Q}[x]$ the minimal polynomial of α . Let $p_i(x), g_j(x) \in \mathbb{Q}[x]$ be polynomials such that, for λ_i, μ_j roots of $P_+(x)$ and $P_-(x)$, respectively, $p_i(\alpha) = \lambda_i$ and $g_j(\alpha) = \mu_j$. Assume w.l.o.g. that $\deg(p_i), \deg(g_j) <$*

$\deg(d)$. Then pretty good state transfer from u to v occurs if and only if the following two conditions are satisfied:

1. The vertices u and v are strongly cospectral.
2. For any choice of integers l_i, m_j such that

$$\sum_i l_i p_i(x) + \sum_j m_j g_j(x) = 0, \quad (2.28)$$

and

$$\sum_i l_i + \sum_j m_j = 0, \quad (2.29)$$

we have

$$\sum_j m_j \text{ is even.} \quad (2.30)$$

Proof. We only need to show that the second condition, in particular, the first equation of the second condition, is the same as the one in Theorem 2.3. First, we make the following simple substitution

$$\begin{aligned} 0 &= \sum_i l_i \lambda_i + \sum_j m_j \mu_j \\ &= \sum_i l_i p_i(\alpha) + \sum_j m_j q_j(\alpha) \\ &= \sum_i l_i p_i(x) + \sum_j m_j g_j(x). \end{aligned}$$

The last equality is a simple observation that as the degrees of $p_i(x)$ and $g_j(x)$ are lower than D , then their linear combination is a polynomial with degree lower than D . As we assumed that $d(x)$ is already the minimal polynomial, this linear combination must be the zero polynomial. \square

Now, with this characterization, we can finally propose the following algorithm for deciding pretty good state transfer.

Theorem 2.34 (Algorithm for pretty good state transfer). *Suppose we have $u, v \in V(G)$ vertices represented by the integer symmetric matrix M . We can decide if pretty good state transfer happens between them in polynomial time in n and the degree of the*

splitting field of $P_+(x)P_-(x)$. The complexity to get the coefficients for $p_i(x)$ and $g_j(x)$ as before, is also polynomial in the same parameters.

Proof. Let λ_i be the roots of P_+ , μ_j be the roots of P_- , $D_+ = \deg(P_+)$ and $D_- = \deg(P_-)$. Let α be such that $\mathbb{Q}[\alpha] \cong \mathbb{Q}[\lambda_1, \dots, \lambda_n, \mu_1, \dots, \mu_m]$ and that we have $d(x) \in \mathbb{Q}[x]$ its minimal polynomial with $\deg(d(x)) = D$. We also have $p_i(x), g_j(x) \in \mathbb{Q}[x]$ polynomials such that $p_i(\alpha) = \lambda_i$ and $g_j(\alpha) = \mu_j$. Again, we can assume that $\deg(p_i), \deg(g_j) < D$.

For the first condition of the lemma above, we know how to test it by the explanation we gave for the algorithm of perfect state transfer in 1.2. So we will concentrate on the second condition.

Let $p_i(x) = \sum_{k=0}^{D-1} a_{ik}x^k$ and $g_j(x) = \sum_{k=0}^{D-1} b_{jk}x^k$. If the degree of any of the polynomials is smaller than $D - 1$, we make the larger than the degree coefficients equal to zero. In practice, we just need the coefficients for both polynomials up to the largest degree between the two, for now we can assume that it is in the worst case $D - 1$.

Hence, by the same observation as in the previous lemma, as the degrees of the p_i 's and g_j 's are lower than D and $d(x)$ is the minimal polynomial of α we get the first equation as

$$\sum_i l_i p_i(x) + \sum_j m_j g_j(x) = 0, \quad (2.31)$$

which we can turn into the following set of equations

$$\sum_i l_i a_{ik} + \sum_j m_j b_{jk} = 0 \text{ for } 0 \leq k < D. \quad (2.32)$$

We add to these equations, the next equation of the lemma

$$\sum_i l_i + \sum_j m_j = 0. \quad (2.33)$$

We turn this system into a system of Diophantine equations as follows: for each equation $0 \leq k < D$ of the first D equations, take the least common multiple of the denominators of the a_{ik} and b_{jk} , say d_k . Now multiply each equation by its d_k , such that they will now be of the form

$$\sum_i l_i d_k a_{ik} + \sum_j m_j d_k b_{jk} = 0 \text{ for } 0 \leq k < D. \quad (2.34)$$

This modification does not exclude any integer solutions, and now it is an integer linear system of the form $Ax = b$, where $b = 0$. So it is homogeneous. By applying Corollary 2.29, we know that we can compute its general solution in polynomial time and get a result of the form (x_p, Y) , where $Ax_p = b$ and $AY = 0$.

Since our system is homogeneous, it always has the trivial solution, so we do not need to worry about the system being inconsistent over the integers. Moreover, all solutions are integer linear combinations of the columns of Y . So for any integer vector v , with appropriate dimension, $AYv = 0$. If the only solution of the system is the trivial solution, we can assume that Y is the zero column vector of appropriate size.

If r' is the number of columns of Y , then we can write the m_j as follows

$$m_j = \sum_{k=1}^{r'} t_k Y_{D_++j,k}, \quad (2.35)$$

where t_k is a free variable and $Y_{D_++j,k}$ is the entry corresponding to k -th column and the row of the coefficients corresponding to m_j . Now, if we sum all the m_j , we get an expression as follows

$$\sum_j m_j = \sum_{k=1}^{r'} t_k \left(\sum_j Y_{D_++j,k} \right) = \sum_{k=1}^{r'} t_k \beta_k, \quad (2.36)$$

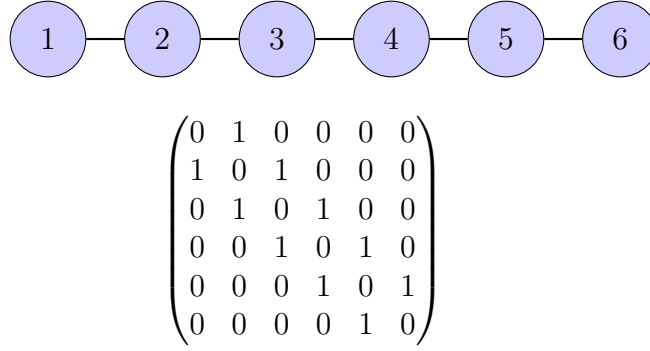
where the β_k 's are integers. Now, as the t_k 's can be any integers, this expression can be odd if and only if any of the β_k is odd. Thus, we can check if pretty good state transfer happens between two vertices by simply checking if any of the β_k is odd.

As every step here can be done in polynomial time, we can do this in polynomial time. \square

The size of the splitting field will determine the complexity, as the polynomials in α that define the roots of $P_+(x)$ and $P_-(x)$ can have degree equal to the size of the minimal polynomial of the splitting field minus 1. This can be very large compared to the size of polynomials P_+ and P_- we have at the beginning. In fact, it can be as large as $n!$. So, this method can be exponential in n in the worst case.

We will show how the whole algorithm works through one example. Consider the graph P_6 below, the path on 6 vertices, with respect to the adjacency matrix, also shown below. We talked before in Section 1.1 that for this graph and matrix, there is pretty good state transfer between its end-nodes.

We assume that we already verified that the extreme vertices are strongly cospectral. We proceed to verify the second condition of Lemma 2.33 using what is shown in Lemma 2.34.

Figure 2.1: Graph and Adjacency Matrix of P_6

First, we need to compute the polynomials $P_+(x)$ and $P_-(x)$ as in Lemma 2.32. So, for $P_+(x)$, apply $e_1 + e_6$ to the powers A^k , k starting at zero, until the collection of the resulting vectors are linear dependent. In this case, we get that

$$A^0(e_1 + e_6) = \begin{pmatrix} 1 \\ 0 \\ 0 \\ 0 \\ 0 \\ 1 \end{pmatrix}, A^1(e_1 + e_6) = \begin{pmatrix} 0 \\ 1 \\ 0 \\ 0 \\ 1 \\ 0 \end{pmatrix}, A^2(e_1 + e_6) = \begin{pmatrix} 1 \\ 0 \\ 1 \\ 1 \\ 0 \\ 1 \end{pmatrix}, A^3(e_1 + e_6) = \begin{pmatrix} 0 \\ 2 \\ 1 \\ 1 \\ 2 \\ 0 \end{pmatrix}.$$

One can see that

$$\begin{pmatrix} 1 \\ 0 \\ 0 \\ 0 \\ 0 \\ 1 \end{pmatrix} - 2 \begin{pmatrix} 0 \\ 1 \\ 0 \\ 0 \\ 1 \\ 0 \end{pmatrix} - \begin{pmatrix} 1 \\ 0 \\ 1 \\ 1 \\ 0 \\ 1 \end{pmatrix} + \begin{pmatrix} 0 \\ 2 \\ 1 \\ 1 \\ 2 \\ 0 \end{pmatrix} = 0.$$

And therefore,

$$P_+(x) = 1 - 2x - x^2 + x^3.$$

If the linear combination results in a polynomial that is not monic, just divide the polynomial by the gcd of its coefficients. Similarly, for $P_-(x)$, now applying to the vector $e_1 - e_6$, we get

$$P_-(x) = -1 - 2x + x^2 + x^3.$$

Now, if we were to compute the roots of the eigenvalues, we would get something like this below for its approximations and an exact form.

Solutions
$x \approx -1.2470$
$x \approx 0.44504$
$x \approx 1.8019$
$x \approx -1.8019$
$x \approx -0.44504$

Figure 2.2: Approximate roots of $P_+(x)P_-(x)$

$$\begin{aligned}
 x &= \frac{1}{3} - \frac{7^{2/3}(1+i\sqrt{3})}{3 \cdot 2^{2/3} \sqrt[3]{-1+3i\sqrt{3}}} - \frac{1}{6}(1-i\sqrt{3}) \sqrt[3]{\frac{7}{2}(-1+3i\sqrt{3})} \\
 x &= \frac{1}{3} - \frac{7^{2/3}(1-i\sqrt{3})}{3 \cdot 2^{2/3} \sqrt[3]{-1+3i\sqrt{3}}} - \frac{1}{6}(1+i\sqrt{3}) \sqrt[3]{\frac{7}{2}(-1+3i\sqrt{3})} \\
 x &= \frac{1}{3} \left(1 + \frac{7^{2/3}}{\sqrt[3]{\frac{1}{2}(-1+3i\sqrt{3})}} + \sqrt[3]{\frac{7}{2}(-1+3i\sqrt{3})} \right) \\
 x &= -\frac{1}{3} - \frac{7^{2/3}(1+i\sqrt{3})}{3 \cdot 2^{2/3} \sqrt[3]{1+3i\sqrt{3}}} - \frac{1}{6}(1-i\sqrt{3}) \sqrt[3]{\frac{7}{2}(1+3i\sqrt{3})} \\
 x &= -\frac{1}{3} - \frac{7^{2/3}(1-i\sqrt{3})}{3 \cdot 2^{2/3} \sqrt[3]{1+3i\sqrt{3}}} - \frac{1}{6}(1+i\sqrt{3}) \sqrt[3]{\frac{7}{2}(1+3i\sqrt{3})}
 \end{aligned}$$

Figure 2.3: An exact formula for roots of $P_+(x)P_-(x)$

So, as they are not rational, we can see that it could be hard to verify the condition manually or even algorithmically using the exact expressions or dealing with approximations. The next step is to factor these polynomials over their splitting field. So, we follow Algorithm 2.23 passing $P_+(x)P_-(x)$ as $f(x)$.

The first step is to factor $f(x)$ over the rationals. The result gives us $P_+(x)$ and $P_-(x)$ as irreducible factors. This could not be the case, as we already discussed in Section 2.1. Regardless of that, we pick any of the irreducible non-linear factors as the minimal polynomial of our first algebraic field extension. In our example, we take $P_+(x)$ and make $g(x) = P_+(x)$. Thus, we factor $f(x)$ again, now over $\mathbb{Q}(\alpha)$, where α is some root of $g(x)$.

We want to factor $f(x)$ as in Algorithm 2.18. In order to do so, we pass $g(x)$ replacing x by t and $f(x)$ as $f(x, t)$. Note that $f(x)$ can still be seen as a polynomial in x and t , even though its degree in t is zero.

The first step to factor $f(x)$ is to find an integer s such that the $N(f(x - st))$ over $\mathbb{Q}(\alpha)$ is square-free. We discussed some ways this can be done in Section 2.2. For our example, one can check that for $s = 2$, we get

$$\begin{aligned}
 f(x - 2t) &= t^6(64) + t^5(-192x) + t^4(240x^2 - 80) + t^3(-160x^3 + 160x) \\
 &+ t^2(60x^4 - 120x^2 + 24) + t(-12x^5 + 40x^3 - 24x) + (x^6 - 5x^4 + 6x^2 - 1).
 \end{aligned}$$

Moreover, this is the Sylvester matrix of $f(x - 2t)$ and $g(t)$, where the columns relative to the coefficients of $f(x - 2t)$ repeat 3 times, since $\deg(g(t)) = 3$ and the columns relative to the coefficients of $g(t)$ repeat 6 times, since $\deg(f(x - 2t))$ over the

variable t is 6,

$$S_{f(x-2t),g(t)} = \begin{pmatrix} 64 & 0 & 0 & 1 & 0 & \dots & 0 \\ -192x & 64 & 0 & -1 & 1 & & 0 \\ 80(3x^2 - 1) & -192x & 64 & -2 & -1 & & 0 \\ 160(-x^3 + x) & \vdots & \vdots & 1 & -2 & & 0 \\ 12(5x^4 - 10x^2 + 2) & & & 0 & 1 & & 0 \\ 4(-3x^5 + 10x^3 - 6x) & & & 0 & 0 & & 1 \\ f(x) & & & 0 & 0 & & -1 \\ 0 & f(x) & & 0 & 0 & & -2 \\ 0 & 0 & f(x) & 0 & 0 & \dots & 1 \end{pmatrix},$$

and its determinant, consecutively $N(f(x - 2t))$, is

$$\begin{aligned} L(x) = \det(S_{f(x-2t),g(t)}) &= x^{18} - 12x^{17} - 3x^{16} + 528x^{15} - 1347x^{14} - 7700x^{13} \\ &+ 32256x^{12} + 36800x^{11} - 291156x^{10} + 56256x^9 + 1176437x^8 - 825968x^7 \\ &- 2236577x^6 + 1836996x^5 + 2173953x^4 - 1375672x^3 - 1062006x^2 + 225576x + 132327, \end{aligned}$$

which is square-free. The next step is to factor $L(x)$ over \mathbb{Q} , doing so results in the factors below

$$\begin{aligned} L(x) &= (x^3 - 3x^2 - 18x + 27) (x^3 - 3x^2 - 4x - 1) (x^3 - 3x^2 - 4x + 13) \\ &\quad (x^3 - x^2 - 16x - 13) (x^3 - x^2 - 16x + 29) (x^3 - x^2 - 2x + 1). \end{aligned}$$

Now, by Theorem 2.13, for each irreducible factor $\ell_i(x)$ of $L(x)$, $\gcd_{\mathbb{Q}(\alpha)}(\ell_i(x), f(x - 2t))$ is an irreducible factor of $f(x - 2t)$, where the gcd is taken over $\mathbb{Q}(\alpha)$. Therefore, both $\ell_i(x)$ and $f(x - 2t)$ are seen as polynomials in x with coefficients over $\mathbb{Q}(\alpha)$, where t , in this case, represents α .

As we want the factors of $f(x)$, we compute for each irreducible factor ℓ_i of $L(x)$, $\gcd_{\mathbb{Q}(\alpha)}(\ell_i(x + 2t), f(x))$ and get the factors of $f(x)$ as below

$$f(x) = (x - t)(x + t)(x - t^2 + 2)(x + t^2 - 2)(x - t^2 + t + 1)(x + t^2 - t - 1). \quad (2.37)$$

In our example, the polynomial factored completely after one extension of the rationals. If it did not, we proceeded as in the loop of Algorithm 2.23, until we generate a $g(t)$ that defines the splitting field of our polynomial and factor $f(x)$ into linear factors over x . Since all factors are linear, we proceed.

The next procedure is to determine what are the factors of $P_+(x)$ and $P_-(x)$. In Section 2.4, we talk about one of the methods to do so. That is, to use the Algorithm

2.18 to factor separately $P_+(x)$ and $P_-(x)$ over the final $g(t)$.

Another possible solution is, for each linear factor of the form $(x - f_i(t))$, compute $P_+(f_i(t)) \bmod g(t)$. If it is equivalent to 0, then $f_i(t)$ represents a root of $P_+(x)$, otherwise it is a root of $P_-(x)$. For instance, $(x - t)$ is one of the factors of $f(x)$. Since in our case $g(x) = P_+(x)$, one can see that

$$P_+(t) \equiv 0 \pmod{g(t)}.$$

Doing so for each factor, we arrive at

$$\begin{aligned} P_+(x) &= (x - (t))(x - (t^2 - t - 1))(x - (-t^2 + 2)), \\ P_-(x) &= (x - (-t))(x - (-t^2 + t + 1))(x - (t^2 - 2)). \end{aligned}$$

One property of this case is that the roots of $P_+(x)$ are exactly minus the roots of $P_-(x)$. This is not the general case, so we cannot always assume this. After that, we can finally write the system we have to solve. The system we have to solve in the second condition of Lemma 2.33 becomes

$$\begin{aligned} \ell_1(t) + \ell_2(t^2 - t - 1) + \ell_3(-t^2 + 2) \\ + m_1(-t) + m_2(-t^2 + t + 1) + m_3(t^2 - 2) &= 0, \\ \ell_1 + \ell_2 + \ell_3 + m_1 + m_2 + m_3 &= 0. \end{aligned}$$

Moreover, by Lemma 2.34, we can rewrite this system as in below

$$\begin{aligned} t^2(\ell_2 - \ell_3 - m_2 + m_3) &= 0, \\ t(\ell_1 - \ell_2 - m_1 + m_2) &= 0, \\ -\ell_2 + 2\ell_3 + m_2 - 2m_3 &= 0, \\ \ell_1 + \ell_2 + \ell_3 + m_1 + m_2 + m_3 &= 0. \end{aligned}$$

In our example, all the coefficients of the ℓ_i 's and m_j 's are integers, but it could be that some of them are rational numbers. If it were the case that some of them are rationals, then just multiplying each equation by the *lcm* (the least common multiple) of the denominators turns the system into a Diophantine linear system, which is what

we needed. In matrix form, our system becomes

$$Ax = \begin{pmatrix} 0 & 1 & -1 & 0 & -1 & 1 \\ 1 & -1 & 0 & -1 & 1 & 0 \\ 0 & -1 & 2 & 0 & 1 & -2 \\ 1 & 1 & 1 & 1 & 1 & 1 \end{pmatrix} \begin{pmatrix} \ell_1 \\ \ell_2 \\ \ell_3 \\ m_1 \\ m_2 \\ m_3 \end{pmatrix} = 0 = b.$$

Following that, we write A as the matrix B as in Equation 2.20. Given the matrix B , all we have to do is to transform it into row echelon form by a series of integer operations into its rows as explained in Section 2.3, such that we get an unimodular matrix U whose rows have the operations we have done in B . Below we have matrices U , B and R' , such that $UB = R'$ and R' is in a row echelon form of B .

$$\left(\begin{array}{c|cccccc} 0 & 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 1 & 0 & 0 \\ \hline 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & -1 & 1 & 0 & -1 & 1 & 0 \\ \hline 0 & -1 & 0 & 1 & -1 & 0 & 1 \end{array} \right) \left(\begin{array}{cccc|c} 0 & 0 & 0 & 0 & 1 \\ \hline 0 & 1 & 0 & 1 & 0 \\ 1 & -1 & -1 & 1 & 0 \\ -1 & 0 & 2 & 1 & 0 \\ \hline 0 & -1 & 0 & 1 & 0 \\ -1 & 1 & 1 & 1 & 0 \\ \hline 1 & 0 & -2 & 1 & 0 \end{array} \right) = \left(\begin{array}{cccc|c} 1 & 0 & -1 & 2 & 0 \\ 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & 1 & 3 & 0 \\ \hline 0 & 0 & 0 & 2 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ \hline 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{array} \right).$$

Using the language we showed in Section 2.3, we separate all the matrices into blocks that can be written as

$$\left(\begin{array}{c|c} d & E^T \\ \hline d_r & x_p \\ \hline 0 & Y^t \end{array} \right) \left(\begin{array}{c|c} -b^T & 1 \\ \hline A^T & 0 \end{array} \right) = \left(\begin{array}{c|c} R^T & d \\ \hline 0 & d_r \\ \hline 0 & 0 \end{array} \right),$$

where $d_r x_p$ is a particular solution to our system, since it is homogeneous, $x_p = 0$ is a particular solution, and the columns of Y generate all integers solutions to our system as in Lemma 2.26. Finally, we can verify the second condition of Lemma 2.34

by computing the sum of m_i 's parameterized by the columns of Y as in below

$$\begin{pmatrix} \ell_1 \\ \ell_2 \\ \ell_3 \\ m_1 \\ m_2 \\ m_3 \end{pmatrix} = \begin{pmatrix} -1 & -1 \\ 1 & 0 \\ 0 & 1 \\ -1 & -1 \\ 1 & 0 \\ 0 & 1 \end{pmatrix} \begin{pmatrix} z_1 \\ z_2 \end{pmatrix}.$$

All the integers coefficients of the second condition must be of this form. Therefore,

$$m_1 = -z_1 - z_2, \quad m_2 = z_1, \quad m_3 = z_2.$$

Hence,

$$\sum m_i = (-z_1 - z_2) + (z_1) + (z_2) = 0,$$

which is even, thus, pretty good state transfer happens between the end-nodes of P_6 . We already knew that, but now we could verify it without using approximations of the eigenvalues or needing to know how to express them beforehand. All steps follow the exact algorithm we showed in this Chapter.

2.5 Computing the Average Mixing Matrix

When we perform a quantum walk, we saw that it can be determined by a unitary matrix $U(t)$ that is equal to $\exp(-itH)$ for some Hamiltonian H . But, as much as this unitary matrix defines the evolution of the system for a given time t , in reality we cannot access this matrix as we can only measure the state at a given time and after this measurement the whole quantum state collapses into the state we measured.

If one repeats this process enough times, one can get an approximation of the probability distribution over the possible states for the measurement. That is why we define the Mixing Matrix $M_G(t)$ of a graph, it captures this probability distribution for a time evolution t .

Definition 2.35 (Mixing matrix). The mixing matrix $M_G(t)$ belonging to the graph G is the following

$$M_G(t) = U(t) \circ U^\dagger(t) = U(t) \circ U(-t), \quad (2.38)$$

where \circ is the Schur product (the entry-wise product of the matrices).

As $U(t)$ is unitary, it follows that each row and column sum to 1, so each row and column form a probability density over the vertices. Now, one might want to know if by performing measurements at any given time, what is the probability distribution they might expect to get. That is why we define the average mixing matrix \hat{M}_G as follows

$$\hat{M}_G = \lim_{T \rightarrow \infty} \frac{1}{T} \int_0^T M_G(t) dt. \quad (2.39)$$

As we are dealing with the adjacency matrix, we can get to a simple expression for the average mixing matrix.

Theorem 2.36 (Godsil [2015]). *Let $A = \sum_r \theta_r E_r$ be the spectral decomposition of the adjacency matrix of G a graph, then*

$$\hat{M}_G = \sum_r E_r \circ E_r. \quad (2.40)$$

Proof. By definition of the mixing matrix and some simple manipulations, we get that

$$\begin{aligned} M_G &= U(t) \circ U(-t) \\ &= \sum_{i,j} e^{it(\theta_i - \theta_j)} E_i \circ E_j \\ &= \sum_i E_i \circ E_i + \sum_{i < j} (e^{it(\theta_i - \theta_j)} + e^{-it(\theta_i - \theta_j)}) E_i \circ E_j \\ &= \sum_i E_i \circ E_i + \sum_{i < j} 2 \cos(t(\theta_i - \theta_j)) E_i \circ E_j. \end{aligned}$$

Now, as

$$\lim_{T \rightarrow \infty} \frac{1}{T} \int_0^T \cos(\gamma t) dt = 0, \quad (2.41)$$

together with the above expression and the definition of the average mixing matrix, the theorem follows. \square

This theorem and some results of this section are contained in Godsil [2015]. This theorem shows us that if we can find the eigenprojectors of A , we can compute the average mixing matrix. At first, one might try to do it numerically. But this is prone to errors. With our work in Section 2.2, we can get the exact average mixing matrix.

First, we need to show how one can construct polynomials to get the eigenprojectors. This is well known in the literature.

Definition 2.37 (Lagrange polynomials). Let $\theta_1, \theta_2, \dots, \theta_m$ be the distinct eigenvalues of A . The Lagrange basis polynomials $\ell_i(x)$ over the eigenvalues can be defined as follows

$$\ell_i(x) = \prod_{j \neq i} \frac{(x - \theta_j)}{(\theta_i - \theta_j)}. \quad (2.42)$$

The idea is quite simple. Use the Lagrange basis polynomials over the eigenvalues of the adjacency matrix to get the eigenprojectors, as follows

$$\ell_i(A) = \sum_r \ell_i(\theta_r) E_r = E_i. \quad (2.43)$$

Given that, we repeat the steps we did in the previous Section. We get the minimal polynomial of A , that is just the square-free part of the characteristic polynomial of A as it is diagonalizable, compute its splitting field, say $\mathbb{Q}(\alpha)$, factor it over its splitting field and get polynomials $p_i(t)$, such that $p_i(\alpha) = \theta_r$.

As we know, we do not have the α from the splitting field algorithm. So, initially we only have the polynomials, but we will show that this will not be necessary for us to compute the average mixing matrix. So, we proceed as follows, let us define the Lagrange basis polynomials over $\mathbb{Q}(\alpha)$

$$\ell_i(x, t) = \prod_{j \neq i} \frac{(x - p_j(t))}{(p_i(t) - p_j(t))}. \quad (2.44)$$

If one wishes to avoid the coefficients and degree in t to grow too much, it is a good idea to compute these polynomials over $\mathbb{Q}(\alpha)$, using modular arithmetic over the minimal polynomial of α , say $g(t)$. Considering that we have these polynomials, we can compute the average mixing matrix as follows

$$\hat{M}_G = \sum_r \ell_r(A, t) \circ \ell_r(A, t).$$

Each term of the expression above can be seen as the Schur product of two matrices of polynomial fractions over the variable t . This might not seem too helpful at first, but the following lemma, shown in Godsil [2015], shows us that this is exactly what we wanted.

Lemma 2.38 (Godsil [2015]). *The entries of the average mixing matrix of a graph are rational.*

We can assume that the gcd of each entry is 1, otherwise, just divide both the numerator and denominator by their gcd. This fact, together with the lemma above, shows us that the polynomials fractions in each entry of \hat{M}_G , when applied at α are in fact a rational number.

So, suppose that each entry ij is of the form $p_{ij}(t)/q_{ij}(t)$. Find the inverse of $q_{ij}(t)$ modulo $g(t)$. This can be done by using the extended Euclidean algorithm to find polynomials $r(t), s(t), u(t)$ such that

$$r(t)q_{ij}(t) + s(t)g(t) = \gcd(q_{ij}(t), g(t)) = u(t). \quad (2.45)$$

Then, multiply both $p_{ij}(t)$ and $q_{ij}(t)$ by $r(t)/u(t)$ and compute their values modulo $g(t)$. Since $r(t)/u(t)$ is the inverse of $q_{ij}(t)$, it must be that the result of $q_{ij}(t)r(t)/u(t)$ modular $g(t)$ is a rational number (if the gcd is done so that the result is normalized, the result must be equal to 1).

As we multiplied both the numerator and denominator, the new polynomial must have the same value when applied to α . Therefore, the result of $r(t)p_{ij}(t)/u(t)$ modulo $g(t)$ must also be a rational number, and we get what we wanted.

If we proceed in this way for all the entries of the matrix, we have the average mixing matrix. All these procedures are polynomial on the size of the splitting field and the size of the minimal polynomial of A .

Chapter 3

Continued Fractions

3.1 Why use Continued Fractions?

In Chapter 2 we saw an algorithm to decide if pretty good state transfer happens between two vertices a and b in a graph. This algorithm is exact, and it is polynomial on the size of the characteristic polynomial of the graph and the degree of the Splitting Field of the eigenvalues in the support of either vertex.

As much as this algorithm helps us, due to the degree of the Splitting Field potentially being exponential on the size of the polynomial, we might want to check the occurrence of pretty good state transfer by other means, using approximations of the eigenvalues. Moreover, it would be good to find a way of using these approximations to determine a time t that, using some parameter that determines how close our approximations are to the eigenvalues, the absolute value of $U(t)_{a,b}$ approaches 1, i.e., perfect state transfer.

For example, if all our approximations are such that the absolute value of their difference from their respective eigenvalues is at most ϵ with $0 < \epsilon \leq 1$ small enough, we might want to find a time t such that

$$|\exp(itA)_{a,b}| \geq 1 - \epsilon, \quad (3.1)$$

which is precisely our definition for pretty good state transfer.

We propose to use continued fractions as a form to give a structured approximation to the eigenvalues. From this representation we would want to derive properties that would allow us to assert if pretty good state transfer happens and at what time. The idea came from one example found in Godsil [2015].

Our goal here is to at least introduce this topic of research, show how it can be

used, and give some more examples. So let us define a few things first. The study of continued fraction and its applications is extensive on its own. We will show only some initial definitions and concepts related to it. Most of what we show can be found in Khinchin [1964].

Definition 3.1 (Simple continued fraction). Let r be a real number. We say that the list $[a_0; a_1, a_2, \dots]$, where $a_0 \in \mathbb{Z}$ and $a_i \in \mathbb{N}$ for $i \geq 1$, is a simple continued fraction representation for r if

$$r = a_0 + \frac{1}{a_1 + \frac{1}{a_2 + \dots}}. \quad (3.2)$$

Definition 3.2 (Finite/infinite continued fractions). We say that the continued fraction is finite if the list of integers that defines it is finite. Otherwise, we say that is infinite. We call the numbers a_i from the continued fractions by elements.

Definition 3.3 (Convergents of a continued fraction). The rational number

$$\frac{p_i}{q_i} = [a_0; a_1, a_2, \dots, a_i] \quad (3.3)$$

is called the i -th convergent.

Definition 3.4 (Periodic continued fractions). We say that the infinite continued fraction $r = [a_0, a_1, \dots]$ is periodic if there are positive integers k_0 and h such that for all $k \geq k_0$

$$a_{k+h} = a_k. \quad (3.4)$$

We denote periodic continued fractions as follows

$$r = [a_0, a_1, \dots, a_{k_0-1}, \overline{a_{k_0}, a_{k_0+1}, \dots, a_{k_0+h-1}}]. \quad (3.5)$$

Definition 3.5 (Best rational approximation of a real number). For a real number r , we say that a/b , where $b > 0$, is a best rational approximation of r if there is no other rational number closer to r with a denominator smaller or equal than b .

There is a generalized continued fraction for complex numbers or even other more general definitions for real numbers, but we will focus on simple continued fractions. So from here on when we say continued fractions we mean simple continued fractions.

Continued fractions have numerous relevant properties. For example, if $a, b \in \mathbb{Z}$ are coprime there are infinitely many ways of describing a/b , multiplying both a and b by any integer different from zero. However, in continued fractions, there are only two ways. For example, if $q \in \mathbb{Q}$, then there are a_0, \dots, a_n so that $q = [a_0; a_1, a_2, \dots, a_n + 1] = [a_0; a_1, a_2, \dots, a_n, 1]$. If we impose the condition that the last integer in the list cannot be 1, then the continued fraction is always unique.

Moreover, all finite continued fractions define a rational number. Infinite continued fractions define all irrational numbers ([Khinchin, 1964, Theorem 14]). Also, with the above restriction on the last element of the finite continued fractions, the continued fraction of a real number is unique.

One more interesting feature of continued fractions is that we can use them to get each one of the best rational approximations of a real number. In Khinchin [1964], there are two definitions of the best rational approximations. The first says that for a real number r , a/b is a best rational approximation of the first kind if for all fractions $c/d \neq a/b$, with $0 < d \leq b$, then

$$\left| r - \frac{a}{b} \right| < \left| r - \frac{c}{d} \right|. \quad (3.6)$$

Now, a/b is a best rational approximation of the second kind for r if for all fractions $c/d \neq a/b$ with $0 < d \leq b$, then

$$|br - a| < |dr - c|. \quad (3.7)$$

In Khinchin [1964], it is shown that best rational approximations of the second kind are also of the first kind. Moreover, we have the following theorem.

Theorem 3.6 ([Khinchin, 1964, Theorem 16]). *Every best approximation of the second kind is a convergent.*

Therefore, this theorem shows us that the convergents defined by the continued fractions not only are good approximations to a given real number, but they are the best in a given measure.

All these points may be enough to understand the interest in them as approximations of real numbers. But now we need to show how do we obtain the continued fractions of a given number. As the continued fraction of irrational numbers are infinite we eventually want the procedure to stop, so below is an algorithm that stops at the k -th element of a continued fraction. A similar procedure to the one below, but with no condition to stop, is shown in [Khinchin, 1964, Theorem 14].

Algorithm 3.7 Compute the Continued Fraction of a Real Number

Input: $r \in \mathbb{R}$ and $0 \leq k \in \mathbb{N}$ **Output:** The continued fraction of r up to the index k : a_0, a_1, \dots, a_k

$$a_0 = \lfloor r \rfloor$$

$$i = 1$$

while $i \leq k$ **do**

$$r = r - a_{i-1}$$

$$r = r^{-1}$$

$$a_i = \lfloor r \rfloor$$

$$i = i + 1$$

end while**return** (a_0, a_1, \dots, a_k)

For example, for the real number $r = \sqrt{2}$, we have the following steps: first, compute $\lfloor \sqrt{2} \rfloor = 1$, so $a_0 = 1$. After that, make

$$r = \frac{1}{\sqrt{2} - 1} = \sqrt{2} + 1.$$

Again, compute $\lfloor r \rfloor = 2$, so $a_1 = 2$. Now, make

$$r = \frac{1}{\sqrt{2} + 1 - 2} = \frac{1}{\sqrt{2} - 1} = \sqrt{2} + 1. \quad (3.8)$$

One can check that this pattern will repeat. Therefore, as we defined before, $\sqrt{2}$ is represented by $[1, \bar{2}]$. Periodic continued fractions are a very distinct type of continued fractions because they define the elements of the form $a + b\sqrt{D}$, where $a, b \in \mathbb{Q}$ and $D \in \mathbb{N}$ square-free. These numbers are called quadratic irrationals. This can be seen in the following theorem.

Theorem 3.8 ([Khinchin, 1964, Theorem 28]). *Every periodic continued fraction represents a quadratic irrational number, and every quadratic irrational number is represented by a periodic continued fraction.*

Quadratic irrational numbers are of particular interest for state transfer as they appear in characterizations of perfect state transfer like in Theorem 1.6 and Section 4.3. Furthermore, perfect state transfer between two vertices, as shown in Corollary 4.14, imply that the vertices are periodic. We define what it means in the next chapter for vertices to be periodic, but it suffices to say that it is a concept related to perfect state transfer. This is another reason why we may want to investigate their relation in terms of pretty good state transfer.

Now, one question we might ask is if the size of the periodic part of the periodic continued fraction is always the same when we fix the quadratic extension, meaning, if the D in $a + b\sqrt{D}$ is fixed. Or even if by multiplying a quadratic irrational number, we multiply the elements of the periodic part. That can be seen to be not true by using the example of $\sqrt{2}$ above and $2\sqrt{2}$ below.

Again, $r = 2\sqrt{2}$, then $a_0 = \lfloor 2\sqrt{2} \rfloor = 2$. Now,

$$r = \frac{1}{2\sqrt{2} - 2} = \frac{1}{2(\sqrt{2} - 1)} = \frac{\sqrt{2} + 1}{2}.$$

So, $a_1 = 1$ and

$$r = \frac{2}{\sqrt{2} - 1} = 2(\sqrt{2} + 1),$$

therefore, $a_2 = 4$ and

$$r = \frac{1}{2(\sqrt{2} + 1) - 4} = \frac{1}{2(\sqrt{2} - 1)}.$$

We can conclude from the calculations above that $r = 2\sqrt{2} = [2, \overline{1, 4}]$. Now, the size of the period doubled, so another question could be if the size of the period grows with the b multiplying \sqrt{D} and this can be seen to be not true always with the examples:

$$\begin{aligned}\sqrt{2} &= [1; \overline{2}], \\ 2\sqrt{2} &= [2; \overline{1, 4}], \\ 3\sqrt{2} &= [4; \overline{4, 8}], \\ 4\sqrt{2} &= [5; \overline{1, 1, 1, 10}], \\ 8\sqrt{2} &= [11; \overline{3, 5, 3, 22}].\end{aligned}$$

So, even when we fix a in $a + b\sqrt{2}$, there is not much we can say about the pattern of the periodic part. Besides that, not much is known for continued fractions in terms of its patterns for higher extension fields. Therefore, one line of investigation to pursue is to see if for graphs that present pretty good state transfer, there is any special property that helps us decide if it happens or not.

One final thing to note is that there is another formula that relates the convergents p_i/q_i of a given continued fractions to the two prior convergents, p_{i-1}/q_{i-1} and p_{i-2}/q_{i-2} and the last element of a_i . For this formula to be more general we define that $p_{-1} = 1$ and $q_{-1} = 0$ and, of course, $p_0 = a_0$ and $q_0 = 1$.

Theorem 3.9 ([Khinchin, 1964, Theorem 1]). *For $k \geq 1$, the convergents p_i and q_i can also be described as:*

$$p_k = a_k p_{k-1} + p_{k-2},$$

$$q_k = a_k q_{k-1} + q_{k-2}.$$

This simplifies the computation of the convergents since if we obtain the last element we do not need to compute the whole expression again to find the next convergents, we just need to have in hand the last two computed convergents.

3.2 How to Compute Continued Fractions?

In Section 3.1, we showed some of what is known in the literature about continued fractions, some of its interesting properties and how to compute them given a real number. In practice, we usually do not have the real number we are dealing with, otherwise we might just use it. What we may have is an approximation of the number or some representation of it.

In our case, what we have is a polynomial that contains the eigenvalues of the support. Moreover, regardless of the graph matrix we are dealing with, we assume that it is a symmetric integer-weighted matrix. Therefore, all of its eigenvalues are real and continued fractions are a suitable representation for them. We can also assume that the roots are simple and that they are not rational, since we can easily compute the rational roots and leave the remainder roots for this algorithm.

Here, we will show Akritas [1980] and Akritas and King [1983] method to compute continued fractions of polynomials with real roots. The procedure follows the same idea as the one for computing the continued fractions given in Algorithm 3.7.

First, we will show this procedure only for the positive roots. For the negative roots, just note that if $f(x)$ is our polynomial, then $f(-x)$ makes the negative roots positive and vice-versa. The representation of continued fractions for a negative number from a positive number can be easily computed.

The algorithm is divided into two similar parts, but with different purposes. Initially we have one polynomial that may have more than one positive real root, so we want to compute the continued fractions elements and at the same time to isolate roots. This procedure would be such that we reach a point where for each positive root we have one polynomial that comes from the initial one that has exactly that one root, or some transformation of it, as a positive root.

After this part, we want to continue the process of computing the continued fractions for each polynomial up to some approximation that we want to achieve.

The idea of the algorithm is based on two operations on polynomials that follow from the algorithm of continued fractions. The first one is to shift all the roots by a given amount, say b . This is achieved by taking $f(x)$ and replacing x by $x + b$. If b is positive, we shift all the roots to the left by b in the x axis.

This gives us two things, first, all positive roots that have absolute value lower than b are now negative in the new polynomial. Moreover, all roots that are between b and $b + 1$ are now between 0 and 1.

The second operation is, if $d = \deg(f(x))$, to replace x in $f(x)$ by $1/x$ and after it, multiply the resulting function by x^d . This procedure gives us that every root of $f(x)$ that is between 0 and 1 will be between 1 and ∞ in the new polynomial we created with this operation. Moreover, all the roots that are larger than 1, will now be between 0 and 1.

It is a simple exercise to check that if there is only one root between b and $b + 1$, then after replacing x by $x + b$, then doing this second operation on the resulting polynomial and finally, replacing x by $x + 1$, the resulting polynomial will have only a single positive real root.

Furthermore, it can be checked that if r is a root of $f(x)$ with continued fraction $[a_0, a_1, \dots]$, then applying successively, starting with a_0 , the substitution of x by $1/x + a_i$, then multiplying it by x^d , where d is the degree of the previous polynomial, eventually for some i , we will have only the remainder continued fraction expansion of r as a positive root.

Now, the first problem appears, as some of these operations depend exactly on the a_i 's we are trying to compute. Because we are trying to compute the continued fractions of all roots, we will show that this is not a problem. The solution is to find a tight lower bound on the positive roots of $f(x)$. This lower bound must be exactly the floor of the positive root of $f(x)$ with the smallest absolute value, i.e., it's the a_i .

So, after finding this a_i , the algorithm will replace x by $1/x + a_i$, multiply the resulting function by x^d for the correct d , call itself recursively for $f(x+1)$ and continue computing the continued fractions with its current $f(x)$. This will form a recursion tree such that the current node divides into two nodes, one that computes the continued fractions for roots between a_i and $a_i + 1$ and the other for roots after $a_i + 1$.

The recursion stops when a node detects that it has a polynomial with one positive real root. Now, it can continue to compute the continued fractions for this single root up to the approximation that is needed. This is done by simply continuing the procedure of computing a tight lower bound a_i on this single root, replacing x by $1/x + a_i$,

multiplying it by x^d for the current degree d , and repeating.

Of course, it is possible to stop the procedure before isolating the roots if we get the approximation that we wanted, but it is reasonable to assume that it is not the case, otherwise our continued fractions by that point could not distinguish sufficiently two or more roots.

Given that the general procedure was defined, we need only to explain some details of the algorithm. Mainly, how to get a tight lower bound, how to check if our current approximation is enough and how to check if we have only a single positive root.

First, let us show how to check if we have only a single positive root. For this, there is a rule by Descartes known as Descartes's rule of signs that states that: for a polynomial $f(x) = \sum_{i=0}^k a_i x^i$ with real coefficients, if after removing the a_i which are equal to zero, we order the coefficients a_i in descending order by the variable exponent, the number of positive real roots (counting repeated roots) of $f(x)$ is either equal to the amount of sign variations between the consecutive ordered coefficients or is less than it by an even amount.

For example, $f(x) = (x+1)^2(x-1) = x^3 + x^2 - x - 1$ has the following sequence of signs $(+, +, -, -)$, so it has one sign variation (between the second and third coefficient). Therefore, as it can only have an even number less than that of positive real roots, it must have a single positive real root. This condition is exactly what we wanted, that is, we just need to check if the current polynomial has a single sign variation.

Another example with coefficients equal to zero is $x^3 - 1$, which has a single real root. Its sign sequence, after removing zero coefficients, is $(+, -)$ and therefore its only real root is positive.

This method is very useful to find the number of positive and negative (by replacing x by $-x$) real roots, because if the polynomial has no complex roots, as the polynomials we are dealing with, the number of sign variations is exactly the number of positive real roots. The even difference occurs only when there are complex roots which come in pairs for real polynomials.

Now, we will show some methods known in the literature on how to compute the lower bound on the positive roots. One simple idea is to successively replace x by $x+1$ and then check if the sign variation has changed. If yes, then the number of substitutions we have done is the lower bound a_i . Otherwise, repeat the process. It is easy to see that this method works, but it can be very inefficient, since the a_i that we want to reach could be quite large.

Another approach is to find a function that computes a lower bound b , replace x

by $x + b + 1$ and check if the sign variation changed. If not, replace x by $x + b$ and repeat the process for the new polynomial. If the function finds good lower bounds, the number of repetitions of this procedure tends to be low enough to be worthwhile.

In the paper Akritas and King [1983], they use the following theorem for it.

Theorem 3.10 ([Akritas and King, 1983, Theorem 2.2]). *Let*

$$p(x) = \sum_{i=0}^n a_i x^i$$

be an integral-coefficient, monic polynomial of positive degree n , and let λ be the number of its negative coefficients. Then

$$b = \max_{\substack{1 \leq k \leq n \\ a_{n-k} < 0}} |\lambda a_{n-k}|^{1/k} \quad (3.9)$$

is an upper bound on the values of the positive roots of $p(x)$.

This theorem gives us an upper bound on the positive roots of the polynomial, therefore, if b is the upper bound for $f(1/x)$, then it is a lower bound for $f(x)$. According to Akritas and King [1983], this works for the polynomials before isolating the roots.

After the root isolation, the authors say that we can use another function that uses this property of having one sign variation.

Corollary 3.11 ([Akritas and King, 1983, Corollary 2.1]). *Let*

$$p(x) = a_n x^n + a_{n-1} x^{n-1} + \dots + a_{n-r+1} x^{n-r+1} - a_{n-r} x^{n-r} - \dots a_1 x - a_0$$

be an integral-coefficient polynomial of degree n , with only one sign variation in the sequence of its coefficients. Then an upper bound on its (only one) positive root is given by

$$b = \frac{\max_{0 \leq j \leq r} (|a_j|)}{\sum_{i=r+1}^n a_i} + 1. \quad (3.10)$$

The point of it being an upper bound is dealt with as before.

Now, the problem of how good is this upper bound, meaning, how many times we need to apply before we found the actual tight upper bound is not discussed much in Akritas and King [1983].

In Sharma [2008], it is shown what is the complexity of this function and an improved function for computing lower bounds is presented. In the paper, it is also shown how much the growth of the coefficients of the polynomial due to the shift by

the lower bound affect the complexity of this function and the whole procedure. As this is not our focus, we leave this discussion here. What matter for us is that it is possible to compute this lower bound and the general idea behind it.

Now, the last point we still have to tackle is how to check if our current approximation is good enough. This is a subject that is one of the main areas of research on continued fractions since, as we already pointed out in Theorem 3.6, every best rational approximation of the second kind is a convergent.

One of the expressions that show how well the convergents approximate the real number they represent can be calculated on the following sequence of theorems and corollaries.

Theorem 3.12 ([Khinchin, 1964, Theorem 2]). *For all $k \geq 0$,*

$$q_k p_{k-1} - q_{k-1} p_k = (-1)^k. \quad (3.11)$$

Proof. As we defined $p_{-1} = 1$ and $q_{-1} = 0$. We know that $q_0 = 1$ and $p_0 = a_0$. Then

$$q_0 p_{-1} - q_{-1} p_0 = 1. \quad (3.12)$$

Now, assume that, for all $i \leq k$, the formula is valid. So, use the equations in Theorem 3.9. Multiply the first by q_{k-1} , the second by p_{k-1} and then, subtracting the first from the second gives us

$$q_k p_{k-1} - q_{k-1} p_k = (a_k p_{k-1} q_{k-1} - a_k p_{k-1} q_{k-1}) - (q_{k-1} p_{k-2} - q_{k-2} p_{k-1}). \quad (3.13)$$

Applying the formula recursively results in what we wanted. \square

Before we proceed, we need one more definition.

Definition 3.13 (Mediant of two fractions). The mediant of two fractions a/b and c/d , with positive denominators, is the fraction

$$\frac{a+b}{c+d}. \quad (3.14)$$

It is a simple exercise to show the following lemma about mediants.

Lemma 3.14 (Khinchin [1964]). *The mediant of two fractions always lie between them in value.*

Now, with that in mind, consider the following sequence

$$\frac{p_{k-2}}{q_{k-2}}, \frac{p_{k-2} + p_{k-1}}{q_{k-2} + q_{k-1}}, \frac{p_{k-2} + 2p_{k-1}}{q_{k-2} + 2q_{k-1}}, \dots, \frac{p_{k-2} + a_{k-1}p_{k-1}}{q_{k-2} + a_{k-1}q_{k-1}} = \frac{p_k}{q_k}. \quad (3.15)$$

First, we can note that each element from the sequence is a mediant between the one before it and p_{k-1}/q_{k-1} . Therefore, it lies between them. These fractions are called intermediate fractions.

Now, by Theorem 3.12, if $k - 1$ is even, then $p_{k-1}/q_{k-1} < p_{k-2}/q_{k-2}$ and, if $k - 1$ is odd, then $p_{k-1}/q_{k-1} > p_{k-2}/q_{k-2}$. So, in this sequence, if k is even, it forms an increasing sequence. Otherwise, the sequence is decreasing. The definition of mediants and the discussion above can be used to show that.

Theorem 3.15 ([Khinchin, 1964, Theorem 8]). *The value of an infinite continued fraction is greater than any of its even-order convergents and is less than any of its odd-order convergents.*

All of this can be used to show how well the convergents approximate the infinite continued fraction.

Theorem 3.16 ([Khinchin, 1964, Theorem 9]). *The value α of the convergent infinite continued fraction for arbitrary k satisfies the inequality*

$$\left| \alpha - \frac{p_k}{q_k} \right| < \frac{1}{q_k q_{k+1}}. \quad (3.16)$$

Proof. By Theorem 3.15 we know that α is between any two consecutive convergents, therefore

$$\left| \alpha - \frac{p_k}{q_k} \right| < \left| \alpha - \frac{p_k}{q_k} \right| + \left| \alpha - \frac{p_{k+1}}{q_{k+1}} \right| = \left| \frac{p_{k+1}}{q_{k+1}} - \frac{p_k}{q_k} \right|. \quad (3.17)$$

Using Theorem 3.12, we arrive at what we wanted

$$\left| \alpha - \frac{p_k}{q_k} \right| < \left| \frac{p_{k+1}}{q_{k+1}} - \frac{p_k}{q_k} \right| = \frac{1}{q_k q_{k+1}}. \quad (3.18)$$

□

From the formula to compute the convergents in Theorem 3.9 and the fact that $q_0, q_1 \geq 0$ and $a_i \geq 1$, for $i \geq 1$, we have that $q_i \leq q_{i+1}$. Therefore, one simple corollary is the one below.

Corollary 3.17. *For any $k \geq 0$, the inequality holds*

$$\left| \alpha - \frac{p_k}{q_k} \right| < \frac{1}{q_k^2}. \quad (3.19)$$

Now, to show this bound is significant, we have the following theorem.

Theorem 3.18 ([Khinchin, 1964, Theorem 12]). *For arbitrary $k \geq 2$,*

$$q_k \geq 2^{\frac{k-1}{2}}. \quad (3.20)$$

Proof. This follows directly from Theorem 3.9, the fact that $q_{i-1} \leq q_i$ and $a_i \geq 1$ for $i \geq 1$, as

$$q_k = a_k q_{k-1} + q_{k-2} \geq 2q_{k-2}. \quad (3.21)$$

Applying this successively gives us what we want. \square

So, the theorem and the corollary above combined show that the denominator of the current convergent computed can be used to check if the current approximates well enough the roots and, moreover, that the denominator grows rather rapidly.

One question that can be made is if this is the best that can be showed for the approximations of the convergents. The answer is no. As this is a somewhat extensive area, we do not show it here. It can be shown that there are tighter approximations. For a more in depth look into these approximations, we refer to [Khinchin, 1964, Chapter 2].

One important feature to consider in terms of the computation of the continued fractions is that, even when we have only p_k/q_k , and we still do not know the value of q_{k+1} , the first inequality already shows us that the next convergent is in a way determining the approximation. Even more, as $q_{k+1} = a_{k+1}q_k + q_{k-1}$, we can show that

$$\left| \alpha - \frac{p_k}{q_k} \right| < \frac{1}{q_k q_{k+1}} < \frac{1}{a_{k+1} q_k^2}. \quad (3.22)$$

So the elements also control how well the convergents approximate the number. This implicates the complexity of the algorithm in two ways. If in some way we know that the elements a_i are bounded above, this means that the approximation cannot be much tighter than the one in Corollary 3.17, meaning, we cannot find a c too small such that

$$\left| \alpha - \frac{p_k}{q_k} \right| < \frac{c}{q_k^2} \quad (3.23)$$

always holds. This is unfortunate, as having a a_i bounded above could be interesting for the algorithm we showed, since it dictates the growth of the coefficients of $f(x)$ when we replace x by $x + a_i$ and how much we may need to work to find the tight lower bound of the positive roots.

On the other hand, having the a_i 's unbounded could mean that our coefficients of $f(x + b)$ grow too much and the process to find the lower bound gets worse. This comes with the benefit that our approximations get better. This last fact on the approximations is summarized on the theorem below.

Theorem 3.19 ([Khinchin, 1964, Theorem 23]). *For every irrational number α with bounded elements, and for sufficiently small c , the inequality*

$$\left| \alpha - \frac{p}{q} \right| < \frac{c}{q^2} \quad (3.24)$$

has no solution for integers p and q ($q > 0$). On the other hand, for every α with an unbounded sequence of elements and arbitrary $c > 0$, the inequality above has an infinite set of such solutions.

3.3 Continued Fractions and State Transfer

In the previous sections we showed what is known in the literature about continued fractions, how to compute them and some of its properties that are interesting to us. Now, we want to show some possible results for State Transfers by using continued fractions.

We already know from Godsil et al. [2012] that perfect state transfer in the adjacency matrix only happens between the end-nodes in paths of 2 and 3 vertices, whilst if $n = p - 1$ or $n = 2p - 1$ or $2^m - 1$, for p prime and m positive integer, pretty good state transfer happens between the end-nodes.

We want to show a possible connection between pretty good state transfer on a graph and perfect state transfer with the convergents of the eigenvalues in the support. We do this by some examples.

So, our first example, taken from Godsil [2015], is for $p = 5$. Now, for the adjacency matrix of P_4 , we have that its eigenvalues are of the form

$$\theta_1 = \frac{1}{2}(\sqrt{5} + 1), \quad \theta_2 = \frac{1}{2}(\sqrt{5} - 1), \quad \theta_3 = \frac{1}{2}(-\sqrt{5} + 1), \quad \theta_4 = \frac{1}{2}(-\sqrt{5} - 1).$$

Furthermore, it can be computed that its eigenprojectors E_i are such that

$$E_1 - E_2 + E_3 - E_4 = \begin{pmatrix} 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 \end{pmatrix}. \quad (3.25)$$

Denote by R the anti-diagonal matrix above. Let p and $q > 0$ be integers such that

$$\frac{p}{q} \approx \theta_1. \quad (3.26)$$

This means that $q\theta_1 \approx p$ and with that we can infer for the others eigenvalues that

$$q\theta_2 = q(\theta_1 - 1) \approx p - q, \quad q\theta_3 = q(-\theta_1 + 1) \approx -p + q, \quad q\theta_4 = -q\theta_1 \approx -p.$$

Because they differ only by a sign and/or a constant, it is easy to see that the approximation factor is the same for all of them.

Now, what do we mean by pretty good state transfer implying perfect state transfer on the convergents? By Theorem 1.6, we know that the vertices that we are dealing with (the end-nodes) must be strongly cospectral. They are. We know that they must be all integers or of the same quadratic extension. They are. But if they are quadratic integers, they also must be of the form

$$\theta_r = \frac{a + b_r\sqrt{D}}{2}, \quad (3.27)$$

for a, b_r, D to be integers and D square-free. They do not have the form required. Furthermore, they must have some parity condition on the eigenspaces given the eigenvalues.

So, what we do is relax the second condition to apply it on the convergents of the eigenvalues, such that we can check the parity condition. For example, for P_4 as we wrote above, say that $p = 987$, $q = 610$ and make $\tau = q\pi/2 = 610\pi/2 = 305\pi$, then, using the expressions we showed above we have that $\tau\theta_r$ can be approximated as follows

$$\tau\theta_1 \approx 987\pi/2, \quad \tau\theta_2 \approx 377\pi/2, \quad \tau\theta_3 \approx -377\pi/2, \quad \tau\theta_4 \approx -987\pi/2.$$

These values modulo 2π are congruent to

$$3\pi/2, \quad \pi/2, \quad 3\pi/2, \quad \pi/2,$$

which means that they approximate the pattern of plus and minus' that we wanted in Equation 3.25. In other words, where the sign of E_i is plus, $e^{i\tau\theta_i} \approx e^{i3\pi/2}$ and, where

the sign of E_i is minus, $e^{i\tau\theta_i} \approx e^{i\pi/2} = -e^{i3\pi/2}$. Thus, we can see that

$$U(305\pi) \approx -iR,$$

which, according to Godsil [2015], is accurate to as much as five decimal places.

One question that may be asked is why we picked 987 and 610. The answer is that they are part of the Fibonacci sequence, which is exactly the sequence of the convergents of $\theta_1 = [1; \bar{1}]$. Meaning, if f_i and f_{i+1} are terms in the Fibonacci sequence, then, for some index k ,

$$\frac{p_k}{q_k} = \frac{f_{i+1}}{f_i} \quad (3.28)$$

is a convergent of θ_1 . So, as we want to repeat that same pattern of $3\pi/2$, suppose that $f_0 = 1$ and $f_1 = 1$ and therefore, $p_0/q_0 = 1$ and $p_1/q_1 = 2$. Then we have the following table.

i	0	1	2	3	4	5	6	7	8	9
$p_i \bmod 4$	1	2	3	1	0	1	1	2	3	1
$q_i \bmod 4$	1	1	2	3	1	0	1	1	2	3

Table 3.1: Table of convergents of $(\sqrt{5} + 1)/2$ modulo 4

Therefore, for $m \geq 0$,

$$f_{6m+2} \equiv 2 \pmod{4}, \quad f_{6m+3} \equiv 3 \pmod{4}, \quad (3.29)$$

hence, we can conclude that there is an infinite set of indices such that the convergents of the eigenvalues allows us to achieve a ‘‘perfect state transfer’’. As we commented before, as the convergents grow fast, this approximation also grows fast, which comes at the cost of a long time for achieving this.

Another example can be seen for P_5 . It has pretty good state transfer as $5 = 2p - 1$, with $p = 3$ prime. By similar reasons its end-nodes are strongly cospectral and the pattern of the eigenprojectors is the same (alternating plus and minus). It has the following eigenvalues

$$\theta_1 = \sqrt{3}, \theta_2 = 1, \theta_3 = 0, \theta_4 = -1, \theta_5 = -\sqrt{3}.$$

So, similarly, we make

$$p/q \approx \theta_1, \quad (3.30)$$

thus,

$$q\theta_2 = q, \quad q\theta_3 = 0, \quad q\theta_4 = -q, \quad q\theta_5 \approx -p.$$

Now, as $q\theta_3 = 0$, we need that $\tau\theta_i$ to be equal to 0 modulo 2π if i is odd and π if i is even. The continued fractions of $\sqrt{3} = [1; \overline{1, 2}]$, with convergents

$$\frac{1}{1}, \frac{2}{1}, \frac{5}{3}, \frac{7}{4}, \frac{19}{11}, \frac{26}{15}, \frac{71}{41}, \frac{97}{56}, \frac{265}{153} \dots$$

If we compute the values of p_i/q_i modulo 4, we have the following table.

i	0	1	2	3	4	5	6	7	8	9
$p_i \pmod 4$	1	2	1	3	3	2	3	1	1	2
$q_i \pmod 4$	1	1	3	0	3	3	1	0	1	1

Table 3.2: Table of convergents of $\sqrt{3}$ modulo 4

This shows us that if we set $\tau = q\pi$, then at $i = 4m + 1$, for $m \geq 0$, the $p_i \pmod 4$ are equal to 2 and the $q_i \pmod 4$ are equal to 1 or 3, which means that we will have the following approximations for $\tau\theta_r$ modulo 2π

$$0, \quad \pi, \quad 0, \quad \pi, \quad 0,$$

which is exactly what we wanted.

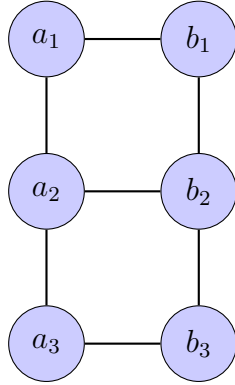
Now, both examples we showed are paths that have pretty good state transfer. Our next example also uses paths, but it is a little more intricate.

The Cartesian product of two graphs G and H is a graph $G \square H$, such that $V(G \square H) = V(G) \times V(H)$ and $((a, b), (c, d)) \in E(G \square H)$ if either the first or the second coordinate in each pair are equal, and the other coordinate have two adjacent vertices in its respective graph.

Let a, b be the vertices of P_2 and $(1, 2, 3)$ be the vertices of P_3 , 2 being the vertex in the middle. Then $P_2 \square P_3$ can be represented as below.

This graph as shown in theorem below by Pal and Bhattacharjya [2017] has pretty good state transfer both between a_1 and a_3 , but also between a_1 and b_1 at different times.

Theorem 3.20 ([Pal and Bhattacharjya, 2017, Theorem 4.2]). *Let G_1 and G_2 be two graphs, so that G_1 is periodic at a vertex at τ and G_2 exhibits perfect state transfer at η . If τ and η are independent over the rational numbers, then $G_1 \square G_2$ admits pretty good state transfer.*

Figure 3.1: Graph $P_2 \square P_3$

$$\begin{pmatrix} 0 & 1 & 0 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 & 1 & 0 \end{pmatrix}$$

Figure 3.2: Adjacency Matrix of $P_2 \square P_3$

In [Pal and Bhattacharjya, 2017, Example 4.1] it is shown why this theorem applies for P_2 and P_3 , but we already know that both of them admit perfect state transfer between its end-nodes. Therefore, using Theorem 4.14, one just need to check if the times in which the perfect state transfer happens in both graphs are independent over the rationals, where the time is defined as in Theorem 1.6.

Knowing all the above, we proceed similarly, first presenting the eigenvalues

$$\theta_1 = \sqrt{2} + 1, \theta_2 = 1, \theta_3 = \sqrt{2} - 1, \theta_4 = 1 - \sqrt{2}, \theta_5 = -1, \theta_6 = -\sqrt{2} - 1,$$

with respective eigenvectors

$$\begin{pmatrix} 1 \\ \sqrt{2} \\ 1 \\ 1 \\ \sqrt{2} \\ 1 \end{pmatrix}, \begin{pmatrix} -1 \\ 0 \\ 1 \\ -1 \\ 0 \\ 1 \end{pmatrix}, \begin{pmatrix} -1 \\ -\sqrt{2} \\ -1 \\ 1 \\ \sqrt{2} \\ 1 \end{pmatrix}, \begin{pmatrix} 1 \\ -\sqrt{2} \\ 1 \\ 1 \\ -\sqrt{2} \\ 1 \end{pmatrix}, \begin{pmatrix} 1 \\ 0 \\ -1 \\ -1 \\ 0 \\ 1 \end{pmatrix}, \begin{pmatrix} -1 \\ \sqrt{2} \\ -1 \\ 1 \\ -\sqrt{2} \\ 1 \end{pmatrix}.$$

Note that all the eigenvalues are in the support of a_1, a_3, b_1 and b_3 . First, we consider the pretty good state transfer between a_1 and a_3 . Now, since we want to have a similar parity condition as perfect state transfer, we just need to look for their respective entries in the eigenvectors and check their signs.

If they are equal, they must have the positive parity, in terms of perfect state transfer condition, $E_r e_a = E_r e_b$, otherwise they have negative parity and $E_r e_a = -E_r e_b$. In this case, we look at the first and third entries of each eigenvector. Therefore, the signs of the eigenprojectors, in that order, are the following $+, -, +, +, -, +$.

As for pretty good state transfer between a_1 and b_1 , we look at the first and fourth entries, so the signs of the eigenvectors are $+, +, -, +, -, -$.

Again, we want to approximate the eigenvectors by their continued fraction. So, suppose that

$$p/q \approx \theta_1 = \sqrt{2} + 1. \quad (3.31)$$

$\sqrt{2} + 1$ has the following continued fraction $[2; \bar{2}]$. These are its first convergents

$$\frac{2}{1}, \frac{5}{2}, \frac{12}{5}, \frac{29}{12}, \frac{70}{29}, \frac{169}{70}, \frac{408}{169}, \frac{985}{408}, \frac{2378}{985}, \frac{5741}{2378}, \dots$$

Now, using our approximation for θ_1 , we can do the following

$$\begin{aligned} q\theta_1 &\approx p, & q\theta_2 &= q, & q\theta_3 &= q(\theta_1 - 2) \approx p - 2q, \\ q\theta_4 &= q(-\theta_3) \approx -p + 2q, & q\theta_5 &= -q, & q\theta_6 &= -q\theta_1 \approx -p. \end{aligned}$$

Now, for the vertices a_1 and a_3 , using the sign sequence of the parity condition that we had before, we want a τ such that $\theta_1, \theta_3, \theta_4, \theta_6$ have the same sign. Making the table with convergents, but modulo 2 instead of 4, we get the following.

i	0	1	2	3
$p_i \pmod 2$	0	1	0	1
$q_i \pmod 2$	1	0	1	0

Table 3.3: Table of convergents of $\sqrt{2} + 1 \pmod 2$

Thus, if we make $\tau = q\pi$, we have the following approximations of $\tau\theta_r$ for the eigenvalues in the positive parity respectively

$$\tau\theta_1 \approx p\pi, \quad \tau\theta_3 \approx (p - 2q)\pi, \quad \tau\theta_4 \approx -(p - 2q)\pi, \quad \tau\theta_6 \approx -p\pi.$$

The other two, $\tau\theta_2$ and $\tau\theta_5$, should be

$$q\pi \text{ and } -q\pi,$$

respectively. So it is easy to see that every convergent allows for $\tau\theta_r$, modulo 2π to be exactly as intended, with the first four being equivalent to 2π when the last two are equivalent to π , and vice-versa.

All of these cases, together with our assumptions and knowledge about state transfers, made us propose the following conjecture: if G is a graph with pretty good state transfer between two vertices, then G admits “perfect state transfer” for some time $\tau = a\pi/b$, $a, b \in \mathbb{Z}$, for some infinite sequence, not necessarily consecutive, of the convergents instead of the eigenvalues.

This relaxed form of perfect state transfer would happen by using the convergents to check the parity condition works, just as we showed in the previous examples.

This would seem to be a reasonable conjecture, since the converse seems to be quite trivial. So, we continue for the other case of $P_2 \square P_3$, which shows us that this conjecture is not true.

For the second case, we want to show a similar result, but now for vertices a_1 and b_1 . We still made the same approximations, since the eigenvalues are the same. But now, the set of eigenvalues that must have the same sign, using our previous defined sign sequence, is the following: $\theta_1, \theta_2, \theta_4$.

If we were to make as before $\tau = q\pi/n$ for some integer n , we would have the following approximations of $\tau\theta_i$ for the first set of eigenvalues

$$p\pi/n, \quad q\pi/n, \quad (-p + 2q)\pi/n.$$

The first option would be that n is even. In this case, simply $p \equiv q \not\equiv 0 \pmod n$ would suffice. But then, looking at the fact that p and q always have different parities as in table 3.3 imply that there is some integer a such that

$$\begin{aligned} p \equiv q \equiv a \not\equiv 0 \pmod n \\ p - np' = a \\ q - nq' = a, \end{aligned}$$

where $p', q' \in \mathbb{Z}$. Suppose w.l.o.g. that p is the even and q is the odd one for a given convergent p/q . Then, since n is even, the left-hand side of the second equation is even and, so, a must be even. But, in the third equation, the left-hand side is odd and, thus, a must be odd. So, for an even n , this is not possible.

Now, for n odd, this would mean that we want a n such that $p \equiv q \pmod n$ for any integer n . But that is not exactly what we want. Looking at the table below, we see the convergents of $\sqrt{2} + 1$, but now $\pmod 3$.

convergents	1	2	3	4	5	6	7	8	9
p mod 3	2	2	0	2	1	1	0	1	2
q mod 3	1	2	2	0	2	1	1	0	1

Table 3.4: Convergents of $\sqrt{2} + 1 \pmod 3$

So, we see that for the convergent $169/70$, we have that $p \equiv q \pmod 3$, but

$$169 \equiv 56 * 3 + 1 \equiv 1 \equiv 23 * 3 + 1 \equiv 70 \pmod 3. \tag{3.32}$$

But then, we have

$$p\pi/n = 169\pi/3 = 56\pi + \pi/3 \equiv \pi/3 \pmod{2\pi}, \quad (3.33)$$

and

$$q\pi/n = 70\pi/3 = 22\pi + 4\pi/3 \equiv 4\pi/3 \pmod{2\pi}. \quad (3.34)$$

Therefore, they have opposite signs, which is not what we wanted as they have the same sign in our sign sequence. So, if n is odd we need not only that $p \equiv q \pmod{n}$, but also, that $\lfloor p/n \rfloor \equiv \lfloor q/n \rfloor \pmod{2}$. This in turn means that we want that

$$p \equiv q \pmod{(2n)}. \quad (3.35)$$

As we already showed above, when n was assumed to be even, there is no solution in this case. Therefore, our conjecture does not work.

This does not invalidate the use of continued fractions in approximating the eigenvalues, or even in studying conditions that they may imply for state transfers. It just means that we might need to work more in finding more useful examples to get insights into how their convergents may or not affect state transfers.

Chapter 4

Normalized Laplacian

4.1 Other Matrices

In the previous chapters, we assumed that the matrix we are dealing with is the adjacency matrix of the graph. While some results that we presented, cited or proved are more general, others are unique to the adjacency matrix or integer weighted adjacency matrices.

In this chapter, we will study another matrix that can be used for representing a graph: the Normalized Laplacian. We will show a characterization for perfect state transfer similar to the one in Theorem 1.6. Following that, we will use this characterization to show some occurrences of perfect state transfer. Furthermore, we will show a characterization of pretty good state transfer between the end-vertices of paths, similar to the one showed in Section 1.2 for the Laplacian matrix.

First, let us define what is the Normalized Laplacian of a graph.

Definition 4.1 (Degree matrix). We call by D , the degree matrix of the graph G , an $n \times n$ diagonal matrix with entry $D_{u,u} = \deg(u)$, where $\deg(u)$ is the degree of the vertex u .

Definition 4.2 (Laplacian matrix). A matrix L is called the Laplacian matrix of a graph G if $L = D - A$, where D is the degree matrix of G and A is the adjacency matrix of the same graph.

Definition 4.3 (Normalized Laplacian). A matrix \mathcal{L} is called the Normalized Laplacian of a graph G if $\mathcal{L} = D^{-1/2} L D^{-1/2}$, where L is the Laplacian matrix of G , D is the degree matrix of G and $D^{-1/2}$ is the diagonal matrix that we get by taking the inverse of the square root of each non-zero diagonal entry of D and the zero entries stay the same.

So one major difference between the Normalized Laplacian matrix of a graph and both the adjacency matrix and the Laplacian is that while the last two have only integer entries, the former can have rational or even irrational entries. We also define a useful notation for us.

Definition 4.4 (Density matrix of a vertex). If e_a is the initial state for a vertex a , then $D_a = e_a e_a^T$ is its density matrix. Moreover, we denote by $D_a(t) = U(t)D_a U(-t)$ its time evolution density matrix.

The density matrix notation is defined here as it is a convenient way of representing a state. As this chapter is about the Normalized Laplacian, we use $U(t)$ to denote $\exp(-it\mathcal{L})$, where \mathcal{L} is the Normalized Laplacian.

This changes nothing on results that assume only that the matrix is symmetric. But if we assume that entries are integers or rational, the results could not be the same. For perfect state transfer, the result we showed in Theorem 1.6 depends on the fact that the matrix we are dealing with is integer valued.

In the Definition 1.3, we defined what are cospectral vertices. For this chapter we will need another equivalent definition of cospectral vertices, so we use the theorem below taken from Godsil [2015]. We do not prove it here, as it is not the focus of this dissertation.

Theorem 4.5 (Godsil [2015]). *Given G , $A = \sum_r \theta_r E_r$ its adjacency matrix and vertices a and b . Let also $G \setminus \{a\}$ be the graph that we get by removing the vertex a from G and $W_{a,a}(G, t) = \sum_{k=0}^{\infty} (A^k)_{a,a} t^k$, then the following are equivalent:*

1. $\phi(G \setminus \{a\}, t) = \phi(G \setminus \{b\}, t)$.
2. $(E_r)_{a,a} = (E_r)_{b,b}$ for all r .
3. $(A^k)_{a,a} = (A^k)_{b,b}$ for all integers k .
4. $W_{a,a}(G, t) = W_{b,b}(G, t)$.

This theorem is shown for the adjacency matrix, but we can show it for any symmetric real matrix that represents the graph. The main difference is that the walk generating function $W(G, t)$ for the adjacency matrix has a combinatorial interpretation, as its (a, b) entry has the generating function for the number of walks of fixed lengths from a to b .

So, for the Normalized Laplacian, we need to show another result. In Section 4.2, we show how we get the strong cospectrality condition for perfect state transfer,

and we also show a combinatorial interpretation of cospectrality over the Normalized Laplacian.

In Section 4.3, we will show a characterization for perfect state transfer to occur between two vertices. This work is a continuation of the work done in the undergrad thesis, Baptista [2019]. Following that, in Section 4.4, we use this characterization together with other results to show some conditions for trees to present perfect state transfer.

Finally, in Section 4.5, we will show a characterization of pretty good state transfer in paths on the Normalized Laplacian. This work is also present in the undergrad thesis, Baptista [2019].

4.2 Strong Cospectrality

In Section 4.1, we showed Theorem 4.5 from Godsil [2015] that gives equivalent definitions for two vertices to be cospectral. There it is shown for the adjacency matrix, but the results follow for any symmetric real matrix. One of such definitions is that if a and b are cospectral, then $A_{a,a}^k = A_{b,b}^k$ for all $k \in \mathbb{N}$.

In Kempton et al. [2020], it is shown that cospectrality between vertices implies a condition on the automorphisms of the isospectral reductions over these vertices, which shows connections of cospectrality with different concepts of graph theory.

For the adjacency matrix A this definition has a combinatorial interpretation, because for $i, j \in V(G)$, $A_{i,j}^k$ is equal to the number of different walks of size k between i and j . So, two vertices a and b being cospectral means that for any integer k the number of closed walks, i.e., walks that start and end at the same vertex, of size k is the same for a and b .

For weighted adjacency matrices, we could still view this definition as a “count” of weighted closed walks, but the interpretation unless for specific weights and applications lose its meaning. So, as the Normalized Laplacian can be viewed as a weighted adjacency matrix with loops, we want to give another interpretation for the cospectrality between two vertices.

First, we need to show one relation of the Normalized Laplacian and classical random walks on graphs. Classical random walks on graphs is a procedure similar to quantum walks, in which each vertex gets a probability distribution over its edges (the non-edges get zero probability). Then, starting at a vertex, at each iteration we jump from the current vertex to another vertex taken from a random sampling of the edges of the graph defined by the probability distribution of the current vertex.

Random walks have multiple applications, some different definitions/formulations and are a field of research on their own, and we do not delve into more details as they are not our focus.

One kind of random walk can be defined by the matrix $D^{-1}A$, where A is the adjacency matrix and D is the diagonal degree matrix, as we defined before. This matrix is stochastic, meaning its rows form uniform probability distributions over the edges of a given graph. This, together with the procedure defined before, gives us a Markov chain that represents the random walk.

Hereon, we assume that the graph has no isolated vertices. Now, one way of defining the Normalized Laplacian matrix \mathcal{L} is through the equation $\mathcal{L} = D^{-1/2}LD^{-1/2}$. Now, by conjugating \mathcal{L} by $D^{-1/2}$, we get that

$$D^{-1/2}\mathcal{L}D^{1/2} = D^{-1}L, \quad (4.1)$$

this can be rearranged and, using the definition of the Laplacian matrix, we have that

$$D^{-1}A = D^{-1/2}(I - \mathcal{L})D^{1/2}. \quad (4.2)$$

This relation between the classical random walk and the Normalized Laplacian is one of the reasons on why we might want to study quantum walks on the latter. Another reason is that some effort has been made recently in constructing weighted graphs such that we could tweak those weights to get the state transfer we want. One possible weight to chose from are the weights defined by the Normalized Laplacian.

Now, there are multiple questions we might ask for the classical random walks. One question, in particular, is if we start at a vertex a , what is the chance that we stay at that vertex after k steps of the random walk. For this question, we look at the entry $(D^{-1}A)_{a,a}^k$.

The lemma below shows that for two cospectral vertices in the Normalized Laplacian, whatever this probability of staying at the same vertex is, it is the same for both of them.

Lemma 4.6. *Let a and b be cospectral vertices for a graph G with no isolated vertices with respect to the Normalized Laplacian. Then, considering the classical random walk modeled by the Markov Chain defined by $D^{-1}A$, the probability that we start at a and end at a after k iterations is the same that we start at b and end at b after k steps, for any $k \geq 0$.*

Proof. By Theorem 4.5, we know that if a and b are cospectral over the Normalized Laplacian, then $\mathcal{L}_{a,a}^k = \mathcal{L}_{b,b}^k$. Now, as \mathcal{L} is a symmetric matrix, this means that for any

polynomial $p(x)$,

$$p(\mathcal{L})_{a,a} = p(\mathcal{L})_{b,b}. \quad (4.3)$$

Now, we know that $D^{-1}A = D^{-1/2}(I - \mathcal{L})D^{1/2}$. This means that

$$\begin{aligned} (D^{-1}A)_{a,a}^k &= (D^{-1/2}(I - \mathcal{L})D^{1/2})_{a,a}^k \\ &= e_a^T D^{-1/2} (I - \mathcal{L})^k D^{1/2} e_a \\ &= (I - \mathcal{L})_{a,a}^k. \end{aligned}$$

Now, the conclusion follows from the fact that $(I - \mathcal{L})^k$ is a polynomial in \mathcal{L} , and thus $(I - \mathcal{L})_{a,a}^k = (I - \mathcal{L})_{b,b}^k$. \square

Now, before we show our results for the characterization of the Normalized Laplacian, we need to show what is the notion of strong cospectrality and how we prove that it is a necessary condition for graphs with perfect state transfer. The proofs contained below in this section are taken from Godsil [2015].

The results presented in this section follow from three inequalities derived from $|U(t)_{a,b}|$. Despite this chapter being focused on the Normalized Laplacian, in this particular section, the results presented work for any matrix A that is real and symmetric. So, for the Laplacian or any weighted adjacency matrix, the results also work.

Let $U(t) = \exp(itA)$ and $A = \sum \theta_r E_r$. Let also σ_r be the sign of $(E_r)_{a,b}$. Meaning, $\sigma_r = 1$ if $(E_r)_{a,b} > 0$ and -1 if $(E_r)_{a,b} < 0$. Otherwise, make $\sigma_r = (E_r)_{a,b} = 0$. The following lemmas define the three inequalities.

Lemma 4.7. *We have $|U(t)_{a,b}| \leq \sum_r |(E_r)_{a,b}|$. Equality holds if and only if there is a complex number γ such that $e^{it\theta_r} = \sigma_r \gamma$ whenever $(E_r)_{a,b} \neq 0$.*

Proof. By definition and the triangle inequality, the following sequence of inequalities hold

$$\begin{aligned} |U(t)_{a,b}| &= \left| \sum_r e^{it\theta_r} (E_r)_{a,b} \right| \\ &\leq \sum_r |e^{it\theta_r} (E_r)_{a,b}| \\ &\leq \sum_r |(E_r)_{a,b}|. \end{aligned}$$

By the triangle inequality, the equality holds if and only if the condition stated holds. \square

In the Definition 1.4, we defined what it means for two vertices to be parallel. There we defined it over the adjacency matrix, but for any symmetric real matrix given its spectral decomposition the definition applies.

Lemma 4.8. *For vertices a and b , we have that $\sum_r |(E_r)_{a,b}| \leq \sum_r \sqrt{(E_r)_{a,a}} \sqrt{(E_r)_{b,b}}$. Equality is true if and only if a and b are parallel.*

Proof. By Cauchy-Schwarz, for each r , the following is true

$$((E_r)_{a,b})^2 = \langle E_r e_a, E_r e_b \rangle^2 \leq \langle E_r e_a, E_r e_a \rangle \langle E_r e_b, E_r e_b \rangle = (E_r)_{a,a} (E_r)_{b,b}. \quad (4.4)$$

So, applying this inequality to the sum, we have what we wanted. By the Cauchy-Schwarz inequality, the inequality is tight if and only if $E_r e_a$ and $E_r e_b$ are parallel. \square

Lemma 4.9. *For vertices a and b , we have that $\sum_{r=0}^d \sqrt{(E_r)_{a,a}} \sqrt{(E_r)_{b,b}} \leq \sqrt{\sum_{r=0}^d (E_r)_{a,a} \sum_{r=0}^d (E_r)_{b,b}}$. Equality is true if and only if a and b are cospectral.*

Proof. Create a vector v_a with entries v_{a_i} such that $v_{a_i} = \sqrt{(E_r)_{a,a}}$ for $i \in 0, 1, \dots, d$. Similarly, create a vector v_b for entries $(E_r)_{b,b}$. Now, by Cauchy-Schwarz, we have the following

$$\langle v_a, v_b \rangle^2 = \left(\sum_{r=0}^d \sqrt{(E_r)_{a,a} (E_r)_{b,b}} \right)^2 \leq \langle v_a, v_a \rangle \langle v_b, v_b \rangle = \sum_{r=0}^d (E_r)_{a,a} \sum_{r=0}^d (E_r)_{b,b}. \quad (4.5)$$

Now, again by Cauchy-Schwarz, the equality is true if and only if v_a and v_b are parallel. As the entries of both vectors are positive and the sum of the $(E_r)_{a,a}$ and $(E_r)_{b,b}$ sum to one, then $v_a = v_b$. Thus, by Theorem 4.5, a and b are cospectral. \square

We know from the spectral decomposition that $\sum_r E_r = I$, where I is the identity matrix. So, these three lemmas show us the following sequence of inequalities

$$\begin{aligned} |U(t)_{a,b}| &\leq \sum_r |(E_r)_{a,b}| \\ &\leq \sum_r \sqrt{(E_r)_{a,a}} \sqrt{(E_r)_{b,b}} \\ &\leq \sqrt{\sum_{r=0}^d (E_r)_{a,a} \sum_{r=0}^d (E_r)_{b,b}} \\ &= 1. \end{aligned}$$

A graph has perfect state transfer if and only if the three inequalities are tight. This means, by the statement of the lemmas, that a and b are cospectral and parallel.

According to our definition of strongly cospectral vertices in Definition 1.5 this means that a and b are strongly cospectral.

Another definition for strongly cospectral vertices is the following.

Definition 4.10 (Strongly cospectral vertices 2nd def). Two vertices a and b of the graph G represented by a symmetric real matrix $A = \sum_r \theta_r E_r$ are said to be strongly cospectral if for each r , $E_r e_a = \pm E_r e_b$.

We can prove that this definition is equivalent to the one above by the following lemma.

Lemma 4.11. *Two vertices a and b for a graph G represented by a symmetric real matrix $A = \sum_r \theta_r E_r$ are strongly cospectral if and only if they are cospectral and parallel.*

Proof. We know from our definition above of strong cospectrality that if a and b are strongly cospectral, then $(E_r)e_a = \pm(E_r)e_b$. Thus, they are parallel. Moreover, $(E_r)_{a,a} = e_a^T E_r E_r e_a = e_b^T E_r E_r e_b = (E_r)_{b,b}$, hence they are cospectral.

Now, assume that a and b are cospectral and parallel. This means that for each r , there is $\gamma_r \in \mathbb{C}$, such that $(E_r)e_a = \gamma_r(E_r)e_b$. Now, we know that E_r is symmetric and, as the vertices are cospectral, multiplying the above equation by e_a^T on the left we get

$$(E_r)_{a,a} = \gamma_r(E_r)_{a,b} = \gamma_r(E_r)_{b,a} = (E_r)_{b,b}. \quad (4.6)$$

Doing the same, but now multiplying it by e_b^T on the left, we get

$$(E_r)_{b,a} = \gamma_r(E_r)_{b,b}. \quad (4.7)$$

This means that $\gamma_r^2 = 1$. Thus, $\gamma_r \in \{-1, 1\}$. So, for all r , $E_r e_a = \pm E_r e_b$ and the vertices are strongly cospectral. \square

4.3 Perfect State Transfer on The Normalized Laplacian

In this section, we want to show a characterization for perfect state transfer to occur on a graph between two vertices with respect to the Normalized Laplacian. First, we need some results and definitions. We still use, when the result we want to show is

more general, A as a real symmetric matrix. The first one is about periodic vertices, which can be thought of as a vertex that presents perfect state transfer with itself.

Definition 4.12 (Periodic vertices). Let G be a graph and u be a vertex. We say that u is periodic according to a real symmetric matrix A if for some time $\tau > 0$

$$|\exp(i\tau A)_{u,u}| = 1. \quad (4.8)$$

Using the lemma below, taken from Godsil [2015], we can see that perfect state transfer between u and v implies periodicity in u .

Lemma 4.13. *If we have perfect state transfer between u and v at time τ for a real symmetric matrix A , then we have perfect state transfer between v and u at the same time.*

Proof. Suppose that we have perfect state transfer between u and v at time τ for the matrix A , therefore we have that

$$U(\tau)e_u = \exp(i\tau A)e_u = \lambda e_v, \quad (4.9)$$

for $|\lambda| = 1$. Now multiply both sides by $\frac{U(-\tau)}{\lambda}$, we get

$$\lambda^{-1}e_u = U(-\tau)e_v. \quad (4.10)$$

Now, take the conjugate of both sides and, as $U(t)$ is unitary, we have

$$\lambda e_u = \overline{\lambda^{-1}}e_u = \overline{U(-\tau)}e_v = U(\tau)e_v. \quad (4.11)$$

□

From that, it follows the corollary from Godsil [2015].

Corollary 4.14. *If perfect state transfer happens between u and v at time τ for a real symmetric matrix A , then u is periodic according to the same matrix at time 2τ .*

Proof. By the lemma above, the sequence of equalities below follows

$$U(2\tau)e_u = U(\tau)(U(\tau)e_u) = \lambda U(\tau)e_v = \lambda^2 e_u. \quad (4.12)$$

As $\lambda = e^{i\gamma}$, for some γ , then $|\lambda^2| = 1$. □

This corollary allows us to show a restriction on the possible values of the eigenvalues of the matrices of graphs that have perfect state transfer between two vertices.

Definition 4.15 (Ratio condition). For any $\theta_i, \theta_j, \theta_k, \theta_l$, with $\theta_k \neq \theta_l$, eigenvalues in the support of a vertex u , we say that the ratio condition holds if

$$\frac{\theta_i - \theta_j}{\theta_k - \theta_l} \in \mathbb{Q}. \quad (4.13)$$

This definition allows us to show the following lemma from Godsil [2015].

Lemma 4.16. *A graph G is periodic at a vertex u if and only if the ratio condition holds at the vertex u .*

Proof. Let $A = \sum_r \theta_r E_r$ be the spectral decomposition of the matrix we are performing the quantum walk. Periodicity at a vertex u implies that at a time t the following holds

$$U(t)e_u = \lambda e_u, \quad (4.14)$$

where $\lambda \in \mathbb{C}$ and $|\lambda| = 1$. Now, if we take the left side and multiply it by its conjugate-transpose on the right, we get

$$\sum_{r,s} e^{it(\theta_r - \theta_s)} E_r e_u e_u^T E_s = (U(t)e_u)(U(t)e_u)^* = e_u e_u^T. \quad (4.15)$$

Now, as the E_i 's are orthogonal to each other, if we multiply on the left by E_r and on the right by E_s , for any r, s we have

$$e^{it(\theta_r - \theta_s)} E_r e_u e_u^T E_s = E_r e_u e_u^T E_s, \quad (4.16)$$

which gives us that $e^{it(\theta_r - \theta_s)} = 1$. Now, this can only be true if for all θ_r, θ_s in the eigenvalue support of u , there is an integer $m_{r,s}$ such that

$$t(\theta_r - \theta_s) = 2m_{r,s}\pi. \quad (4.17)$$

Now, if we take any of these equations with any r and s and divide by any of the same equations for indexes k and l , but now, $\theta_k \neq \theta_l$, then the ration condition is true as we get

$$\frac{\theta_r - \theta_l}{\theta_k - \theta_l} = \frac{m_{r,s}}{m_{k,l}} \in \mathbb{Q}. \quad (4.18)$$

On the other hand, assume that the ratio condition holds. Note that for any r, s and $\theta_k \neq \theta_l$, we have that for some $m_{k,l} \in \mathbb{Z}$

$$\frac{m_{k,l}(\theta_r - \theta_s)}{(\theta_k - \theta_l)} \in \mathbb{Z}. \quad (4.19)$$

So, if we take $t = 2m_{k,l}\pi/(\theta_k - \theta_l)$, since $\sum_r E_r = I$, we have that the graph is periodic at a vertex u as the following holds

$$D_u(t) = \sum_{r,s} e^{it(\theta_r - \theta_s)} E_r D_u E_s = \sum_r E_r \left(\sum_s e_u e_u^T E_s \right) = D_u. \quad (4.20)$$

□

Now, we need one more auxiliary result from Algebraic Number Theory before we start showing our results.

Lemma 4.17. *Let $\mathbb{Q} \subset \mathbb{L}$ be a finite extension and $S \subset \mathbb{L}$ a set of elements that is closed under the action of the Galois Group of the extension. Then for the $\alpha_i \in S$ the following is true*

$$p = \prod_{i \neq j} (\alpha_i - \alpha_j) \in \mathbb{Q}. \quad (4.21)$$

Proof. This arises naturally from the fact that S is closed under the action of the Galois Group of the extension. We know from Galois Theory that if an element from the extension is fixed by all automorphisms of the Galois Group, then it must be an element from the base field. In this case, a rational number. This can be seen in [Cox, 2012, Theorem 7.1.1].

Now, take an automorphism $\psi \in \text{Gal}(\mathbb{L}/\mathbb{Q})$ and apply on both sides of the equation. First, we know that the automorphism is additive, so $\psi(\alpha_i + \alpha_j) = \psi(\alpha_i) + \psi(\alpha_j)$. Also, note that each clause $c_{ij} = (\alpha_i - \alpha_j)$ is just mapped to itself or another clause as the set is closed over the action of the Galois Group. Finally, two clauses cannot be mapped to the same clause for a fixed automorphism, as the automorphism belongs to a group. So, we have that

$$\psi(p) = \psi\left(\prod_{i \neq j} (\alpha_i - \alpha_j)\right) = \prod_{i \neq j} (\alpha_i - \alpha_j) = p \in \mathbb{Q}. \quad (4.22)$$

□

We still need one result before we get to what we wanted. This can be also found in Godsil [2015]. The eccentricity of a set of vertices is a measure of the largest distance of a set to the remainder of the graph.

Definition 4.18 (Eccentricity of a set). Let G be a graph. The eccentricity of a set $T \subset V(G)$ is the smallest integer r such that any vertex of G is at distance at most r of any vertex in T .

Given the definition above, we can show the following lemma. This lemma is shown for the Normalized Laplacian, but it can be shown that it works for other real symmetric matrices. The restriction being that these matrices must be “adjacency matrices”. We mean by that the off-diagonal entries must be non-zero on the entries corresponding to adjacent vertices and zero otherwise. Moreover, we need that the off-diagonal entries respect some restrictions, for example, that all non-zero entries have the same sign.

Lemma 4.19. *Let G be a graph, \mathcal{L} its Normalized Laplacian, v a vertex of G and S its eigenvalue support. If v has eccentricity r , then $r + 1 \leq |S|$.*

Proof. Let e_v be the characteristic vector of v . As v has eccentricity r , then $\mathcal{L}^k e_v$, for $k \in \{0, 1, 2, \dots, r\}$, is a set of linear independent vectors, because the non-zero support reaches a new vertex for each power.

Now, let $\mathcal{L} = \sum_r \theta_r E_r$ be the spectral decomposition of \mathcal{L} . As we know, any polynomial $p(x)$ applied to \mathcal{L} can be described as

$$p(\mathcal{L}) = \sum_r p(\theta_r) E_r. \quad (4.23)$$

This means that the $\mathcal{L}^k e_v$'s are linear combinations of the $\{E_r e_v\}$. Moreover, by Section 2.5, we know that the E_r 's are polynomials in \mathcal{L} . So, the $E_r e_v$'s are linear combinations of the $\mathcal{L}^k e_v$'s. This means that

$$\text{span}\{\mathcal{L}^k e_v\} = \text{span}\{E_r e_v\}. \quad (4.24)$$

Furthermore, as the E_r are orthogonal, the $E_r e_v$ are also orthogonal, so

$$\dim(\text{span}\{\mathcal{L}^k e_v\}) = |\{\theta_r : E_r e_v \neq 0\}|. \quad (4.25)$$

As we showed that the $\mathcal{L}^k e_v$'s are linear independent for $0 \leq k \leq r$, then $r + 1 \leq |S|$. □

Definition 4.20 (Algebraic number). We say that θ is an algebraic number if it is the root of a rational polynomial.

Definition 4.21 (Quadratic irrational/integer). Let Δ be a square-free integer. We say that θ is a quadratic irrational if it is the root of a second degree polynomial with rational coefficients. If the coefficients are all integers and the polynomial is monic we say that θ is a quadratic integer.

The results below are similar to the ones in Godsil [2015], but there they are presented for integer matrices. Here we show them to the Normalized Laplacian, which is typically not rational.

Theorem 4.22. *Let $S = \{\theta_0, \theta_1, \dots, \theta_d\}$ be a set of real algebraic numbers, closed under taking algebraic conjugates, and with $d \geq 3$. Then, for all r, s, k, l with $\theta_k \neq \theta_l$, we have*

$$\frac{\theta_r - \theta_s}{\theta_k - \theta_l} \in \mathbb{Q} \quad (4.26)$$

if and only if either condition holds:

1. *The elements in S are rational.*
2. *The elements in S are quadratic irrational. Moreover, there is a square-free integer $\Delta > 1$ and rational numbers a, b_0, b_1, \dots, b_d so that*

$$\theta_r = \frac{1}{2}(a + b_0\sqrt{\Delta}). \quad (4.27)$$

Proof. Clearly, if the elements of S are all of either form, then the ratio condition, Equation 4.26, holds. For the converse, if there are at least two elements of S that are rational, say θ_0 and θ_1 with $\theta_0 \neq \theta_1$, we know that for $\theta_r \in S$ the following holds by the ratio condition

$$\frac{\theta_s - \theta_1}{\theta_1 - \theta_0} \in \mathbb{Q}. \quad (4.28)$$

So, all the numbers in S are rational and the first condition holds. Now, assume that at most one element in S is rational. Now, by Equation 4.18, we know that there is a rational $a_{r,s}$ such that $\forall \theta_r, \theta_s \in S$, the equality holds

$$(\theta_r - \theta_s) = a_{r,s}(\theta_1 - \theta_0). \quad (4.29)$$

Now, say that $|S| = \delta$, from the equation above, one can get the following

$$\prod_{r \neq s} (\theta_r - \theta_s) = (\theta_1 - \theta_0)^{\delta^2 - \delta} \prod_{r \neq s} a_{r,s}. \quad (4.30)$$

By Lemma 4.17, we know that the left-hand side is a rational number and the product of the $a_{r,s}$ is also rational, so $(\theta_1 - \theta_0)^{\delta^2 - \delta}$ must be a rational number.

So, let m be the smallest positive integer such that $(\theta_1 - \theta_0)^m$ is a rational number. That means that there are m distinct conjugates of $(\theta_1 - \theta_0)$ of the form $\beta e^{2\pi i k/m}$ for

$k \in \{0, 1, 2, \dots, m-1\}$, where β is the positive m -th root of a rational. As we assumed that S is a set of real numbers, $m \leq 2$.

So, $(\theta_1 - \theta_0)$ is either a rational or a rational multiple of the square-root of a square-free rational Δ with no common factors between numerator and denominator. In the former case, we can still view it as the latter, as in this case $\Delta = 1$. As

$$(\theta_r - \theta_s)^2 = a_{r,s}^2 (\theta_1 - \theta_0)^2 \quad (4.31)$$

we can see that $(\theta_r - \theta_s)^2$ is also rational. So, $(\theta_r - \theta_s)$ is also a rational multiple of a square-free rational, say $\Delta_{r,s}$. Moreover, for all $\theta_k \neq \theta_l$,

$$(\theta_k - \theta_l)(\theta_r - \theta_s) = a_{k,l} a_{r,s} (\theta_1 - \theta_0)^2. \quad (4.32)$$

So, as the right-hand side is a rational, we can conclude that the square-free rational part is the same for all r and s , meaning $\Delta_{r,s} = \Delta, \forall r, s$.

Therefore, there are rational numbers m_r such that, for each r ,

$$\theta_r = \theta_0 + m_r \sqrt{\Delta}. \quad (4.33)$$

One thing more we can note is that if

$$\sqrt{\Delta} = \sqrt{\frac{a}{b}}, \quad (4.34)$$

where $\gcd(a, b) = 1$, $a, b \in \mathbb{Z}$, $b \neq 0$ and both square-free. Then,

$$\sqrt{\Delta} = \frac{\sqrt{ab}}{b}. \quad (4.35)$$

So, we can assume from now on that Δ is a square-free integer and group the $1/b$ into the m_r .

If we sum over S , we get that

$$\sum_r \theta_r = |S| \theta_0 + \sqrt{\Delta} \left(\sum_r m_r \right) \in \mathbb{Q}. \quad (4.36)$$

This can be verified as the left-hand side is fixed by all the automorphisms of the extension containing the elements of S , and as we know it, this means that it is a rational. Therefore, $\theta_0 \in \mathbb{Q}(\sqrt{\Delta})$ and thus, all the elements in S are elements of the same extension.

Moreover, the fact that the rational part is the same follows from Equation 4.33. \square

This theorem shows us that a set of real algebraic numbers that respect the ratio condition can have only a strict format. We will show that, for the Normalized Laplacian, we can narrow it down even further. First, let us show a few characteristics known for the Normalized Laplacian. These results are standard for the Normalized Laplacian. One reference for them is Chung [1997].

Definition 4.23 (Positive semidefinite matrix). We say that a symmetric matrix M is positive semidefinite if all its eigenvalues are non-negative.

There is a whole plethora of characterizations and applications of positive semidefinite matrices, but this is out of our scope, so we focus on this definition and one equivalent definition we give below. These results are standard in studies of positive semidefinite matrices. A symmetric matrix M is called positive semidefinite if there is a matrix B such that

$$M = BB^T. \quad (4.37)$$

Now, the Laplacian matrix is known to be positive semidefinite. It is a simple exercise to check that for any orientation of the edges of the graph the Laplacian matrix $L = BB^T$, where B is the vertex-edge incidence matrix of this orientation of the graph.

This means that the Laplacian matrix is positive semidefinite. Now, by our definition of the Normalized Laplacian we can write it as follows

$$\mathcal{L} = D^{-1/2}LD^{-1/2} = D^{-1/2}BB^TD^{-1/2} = (D^{-1/2}B)(D^{-1/2}B)^T. \quad (4.38)$$

Thus, the Normalized Laplacian is also positive semidefinite. Now, let $\mathbf{1}$ be the all ones vector. Assume also that the graphs we are dealing with have no isolated vertices. One thing we can notice about the eigenvalues of any graph on the Normalized Laplacian is that because we know that $\mathbf{1}$ is an eigenvector for the eigenvalue 0 on the Laplacian matrix, then the following is true

$$\mathcal{L}D^{1/2}\mathbf{1} = D^{-1/2}LD^{-1/2}D^{1/2}\mathbf{1} = D^{-1/2}L\mathbf{1} = 0. \quad (4.39)$$

As the Normalized Laplacian is positive semidefinite, from now on we order eigenvalues of \mathcal{L} such that $0 = \theta_0 \leq \theta_1 \leq \dots \leq \theta_{n-1}$. In Chung [1997], we have the following two lemmas that show properties of the eigenvalues of the Normalized Laplacian. We do not prove this results as they are not our focus, but a complete proof of them can be found in the reference we provide.

Lemma 4.24 ([Chung, 1997, Lemma 1.7]). *For a graph G of n vertices, Normalized Laplacian matrix \mathcal{L} and eigenvalues $\theta_0 \leq \theta_1 \leq \dots \leq \dots \theta_{n-1}$, we have:*

1.

$$\sum_i \theta_i \leq n \quad (4.40)$$

with equality holding if and only if G has no isolated vertices.

2. For $n \geq 2$,

$$\theta_1 \leq \frac{n}{n-1} \quad (4.41)$$

with equality if and only if G is the complete graph on n vertices. Also, for a graph G without isolated vertices, we have

$$\theta_{n-1} \geq \frac{n}{n-1} \quad (4.42)$$

3. For a graph which is not a complete graph, we have $\theta_1 \leq 1$.

4. If G is connected, then $\theta_1 > 0$. If $\theta_i = 0$ and $\theta_{i+1} \neq 0$, then G has exactly $i + 1$ connected components.

5. For all $i \leq n - 1$, we have

$$\theta_i \leq 2. \quad (4.43)$$

with $\theta_{n-1} = 2$ if and only if a connected component of G is bipartite and nontrivial.

6. *The spectrum of a graph is the union of the spectra of its connected components.*

Therefore, if we assume that the graph is connected, then θ_0 is simple. Moreover, with no isolated vertices, $D^{1/2}\mathbf{1}$ is a positive eigenvector, meaning all its entries are positive, then $\theta_0 = 0$ is in the eigenvalue support of all vertices.

Let *den* be a function that returns the denominators of a rational number. If the number is an integer, then it returns 1. Let also *lcm* be a function that returns the least common multiple of a set of integers. From this explanation above and Theorem 4.22 we can get the following theorem.

Theorem 4.25. *Suppose G is a connected graph with at least two vertices, \mathcal{L} its Normalized Laplacian and let $S = \{\theta_0, \theta_1, \dots, \theta_d\}$ be the eigenvalue support of the vertex v , with $\theta_0 < \theta_1 < \dots < \theta_d$. Then G is periodic at v if and only if the eigenvalues in S are rational. Moreover, if the conditions hold, let*

$$m = \text{lcm}\{\text{den}\{\theta_r\}_{\theta_r \in S}\} \quad (4.44)$$

and

$$g = \text{gcd}\{m\theta_r\}_{\theta_r \in S}. \quad (4.45)$$

Then the smallest positive t , such that $D_v(t) = D_v$ is

$$t = \frac{2m\pi}{g} \quad (4.46)$$

and any other periodicity at v occurs at integer multiples of t .

Proof. If the eigenvalues are rational, then certainly the ratio condition is true and by Lemma 4.16 the vertex is periodic.

Now, we want to show that if the vertex v is periodic, then by Lemma 4.16 the ratio condition holds. But first we need to show that all eigenvalues of \mathcal{L} are algebraic numbers. So, we will show that \mathcal{L} is similar to a rational matrix.

Consider the matrix, $D^{-1/2}$, the diagonal matrix that has the inverse square-root of the degrees of the vertices in the diagonal entries. As the graph is connected, this is a valid definition. Its inverse is $D^{1/2}$, the diagonal matrix that has the square-root of the degrees of the vertices in its diagonal. Now, conjugate \mathcal{L} by these matrices, by the definition of the Normalized Laplacian, we have that

$$D^{-1/2}\mathcal{L}D^{1/2} = D^{-1}L. \quad (4.47)$$

So, as the entries of $D^{-1}L$ are rational, its eigenvalues, and by similarity the eigenvalues of \mathcal{L} , are algebraic numbers as they are roots of a polynomial with rational coefficients, the characteristic polynomial of $D^{-1}L$. They are also real, since \mathcal{L} is symmetric. Moreover, \mathcal{L} have all algebraic conjugates of its eigenvalues. Finally, we still need to show that S , the eigenvalue support of v , is closed under taking conjugates.

Now, let u be an eigenvector of \mathcal{L} for the eigenvalue θ_r . Let also θ_s be an algebraic conjugate of θ_r and $\psi \in \text{Gal}(\mathbb{L}/\mathbb{Q})$ be an automorphism for the Galois extension that has both eigenvalues. It is straightforward to show that $D^{-1/2}\psi u$ is an eigenvector of

$D^{-1}L$ for the eigenvalue θ_r

$$D^{-1}LD^{-1/2}u = D^{-1/2}\mathcal{L}D^{1/2}D^{-1/2}u = D^{-1/2}\mathcal{L}u = \theta_r D^{-1/2}u. \quad (4.48)$$

Now, we show that the automorphism that sends θ_r to θ_s sends $D^{-1/2}u$ to some eigenvector of θ_s .

$$\psi(\theta_r)\psi(D^{-1/2}u) = \psi(D^{-1}LD^{-1/2}u) = \psi(D^{-1}L)\psi(D^{-1/2}u) = D^{-1}L\psi(D^{-1/2}u). \quad (4.49)$$

So, $E_r e_v = 0$ if and only if $E_s e_v = 0$. So, the eigenvalue support is closed under taking conjugates. Now, as G is connected and has at least two vertices, we know that $|S| \geq 2$ by Lemma 4.19. If $|S| = 2$, then either the eigenvalues in S are rational or are roots of a second degree rational polynomial. Until this point, either condition is possible, but we will show that this is not the case for \mathcal{L} .

So, assume that $|S| > 2$. As the eigenvalues in S are real algebraic numbers closed under taking conjugates, we can apply Theorem 4.22. Therefore, for $|S| \geq 2$, they are either all rational or all quadratic irrationals with the same rational part, meaning, $\theta_r = \frac{1}{2}(a + b_r\sqrt{\Delta})$, for a, b_r rational and $\Delta > 1$ square-free integer.

Now, for $|S| \geq 2$, assume they are all quadratic irrationals of the form above. By our previous discussion, θ_0 is in S and $\theta_0 = 0$. Therefore, $a = 0$. Moreover, as the Normalized Laplacian is positive semidefinite all its eigenvalues are non-negative. But if $\theta_r = \frac{1}{2}b_r\sqrt{\Delta}$ is an eigenvalue, then, as S is closed under taking conjugates $-\frac{1}{2}b_r\sqrt{\Delta}$ is also an eigenvalue. Meaning, $-\theta_r$ must also be an eigenvalue, which is a contradiction. Thus, the eigenvalues in S are all rational.

Now, if a vertex v is periodic at a time t , we can write t as

$$t = \tau \frac{2\pi m}{g}, \quad (4.50)$$

where τ is some real number. Now, in the proof of Lemma 4.16, we showed that the periodicity at time t implies that there are integers $m_{r,s}$ such that $t(\theta_r - \theta_s) = 2m_{r,s}\pi$ for $\theta_r, \theta_s \in S$. By fixing $\theta_s = \theta_0$, we can rewrite this equation as

$$t = \frac{2m_{r,0}\pi}{\theta_r}, \quad (4.51)$$

for $\theta_r \neq 0$. Now, using both equations we defined for t , we can see that

$$\tau \frac{m\theta_r}{g} \in \mathbb{Z}. \quad (4.52)$$

Now, from the definitions we gave for m and g , we can see that τ must be an

integer. □

With all these results, we can finally give a characterization of perfect state transfer to the Normalized Laplacian.

Theorem 4.26. *Let a and b be vertices in a connected graph G , $U(t) = \exp(it\mathcal{L})$ express the unitary time evolution of our system, \mathcal{L} being the Normalized Laplacian and let $S = \{\theta_0, \theta_1, \dots, \theta_d\}$ be the eigenvalue support of a , with $0 = \theta_0 < \theta_1 < \dots < \theta_d$. Then, there is perfect state transfer between a and b if and only if the following conditions hold:*

1. *Vertices a and b are strongly cospectral.*
2. *The eigenvalues in S are rational.*
3. *Let*

$$m = \text{lcm}\{\text{den}\{\theta_r\}_{\theta_r \in S}\} \quad (4.53)$$

and

$$g = \text{gcd}\{m\theta_r\}_{\theta_r \in S}. \quad (4.54)$$

For all $0 \leq r \leq d$, the following holds:

- $(E_r)_{a,b} > 0$ if and only if $m\theta_r/g$ is even,
- $(E_r)_{a,b} < 0$ if and only if $m\theta_r/g$ is odd.

If the conditions hold, then the minimum positive time we have perfect state transfer between a and b is $\tau = m\pi/g$, and any other time it occurs is an odd multiple of τ .

Proof. By Corollary 4.14 we have that perfect state transfer between a and b implies periodicity. Periodicity by Theorem 4.25 implies condition 2, and Section 4.2 shows that perfect state transfer implies strong cospectrality.

So now, we assume conditions 1 and 2 and show that perfect state transfer is an equivalent condition to the third condition. First, note that by the Definition 4.10 of strong cospectrality the first condition implies that S is the same for a and b .

Now, let $\tau = tm\pi/g$ for any real t such that for some $|\lambda| = 1$ we have

$$U(\tau)e_a = \lambda e_b. \quad (4.55)$$

This happens if and only if, for all θ_r , upon multiplying both sides by E_r on the right, we get $e^{i\tau\theta_r} E_r e_a = \lambda E_r e_b$. As G is connected, we know that $\theta_0 = 0$ is a simple eigenvalue for an eigenvector $D^{1/2}\mathbb{1}$. Therefore, E_0 is a non-negative matrix.

Thus, the condition is equivalent to $e^{i\tau\theta_0} = 1 = \lambda$ and

$$e^{i\tau\theta_r} = \pm 1, \quad (4.56)$$

where the sign is determined by the sign of $E_r e_a = \pm E_r e_b$. Equivalently, this sign can be determined by multiplying both sides by e_a^T on the left and, as $(E_r)_{a,a} > 0$, the sign of $(E_r)_{a,b}$ determines the sign of the equation above. Let now m_0, m_1, \dots, m_d be real numbers such that for all θ_r

$$\tau\theta_r = \frac{t\pi m\theta_r}{g} = m_r\pi. \quad (4.57)$$

This means that perfect state transfer is equivalent to m_r being an even integer if $(E_r)_{a,b} > 0$ and an odd integer if $(E_r)_{a,b} < 0$. As t is a constant and the expression holds for all θ_r , it must be an odd integer that makes no difference on the signs and whether perfect state transfer happens or not.

Thus, perfect state transfer is equivalent to $m\theta_r/g$ being even if $(E_r)_{a,b} > 0$ and $m\theta_r/g$ being odd if $(E_r)_{a,b} < 0$, which is precisely the third condition. \square

4.4 Perfect State Transfer in Trees

Trees are minimally connected graphs, thus good choices if one wants to build a network and minimize resources. So, it is not surprising that research has been done in state transfer in trees.

In Coutinho and Liu [2015], it was shown that no tree on three or more vertices admits perfect state transfer with regards to the Laplacian matrix. So, considering how the Normalized Laplacian can be defined in terms of the Laplacian matrix, we try to see if something similar happens for the Normalized Laplacian. We proceed similarly as in Coutinho and Liu [2015] and Godsil [2015].

We start by showing some useful spectrum properties of trees over the Normalized Laplacian. We already know by Lemma 4.24 that, for any graph, 0 is an eigenvalue, and the eigenvalues are between 0 and 2. Moreover, if the graph is a tree, hence connected

and bipartite, 0 and 2 are eigenvalues. Now, we can also show that if it is a tree, 2 is also a simple eigenvalue. This can also be seen by the following lemma.

Lemma 4.27 ([Chung, 1997, Lemma 1.8]). *The following statements are equivalent:*

1. G is bipartite.
2. G has i connected components and $\theta_{n-j} = 2$ for $1 \leq j \leq i$.
3. For each θ_i , the value $2 - \theta_i$ is also an eigenvalue of G .

Proof. By Lemma 4.24, we know that the eigenvalues of G are formed by the union of the eigenvalues of each connected component of G . Also, we know that if G is connected 0 is a simple eigenvalue. So we only need to show the third condition for a connected graph, and the rest follows.

Suppose that G is a connected bipartite graph such that \mathcal{L} can be written as

$$\mathcal{L} = \begin{pmatrix} I & C \\ C^T & I \end{pmatrix}. \quad (4.58)$$

Now, let θ be an eigenvalue of \mathcal{L} for an eigenvector v . So, if G is bipartite, its vertices can be separated into two sets A and B such that both sets form an independent vertex set. By indexing v by the vertex set, we can define a vector u as follows

$$u_i = \begin{cases} v_i & \text{if } i \in A \\ -v_i & \text{otherwise.} \end{cases} \quad (4.59)$$

Let v_A be the vector with only entries in A and v_B similarly. If I is an identity matrix with proper dimension, we can see that

$$\mathcal{L}v = \begin{pmatrix} I & C \\ C^T & I \end{pmatrix} \begin{pmatrix} v_A \\ v_B \end{pmatrix} = \begin{pmatrix} v_A + Cv_B \\ v_B + C^T v_A \end{pmatrix} = \theta \begin{pmatrix} v_A \\ v_B \end{pmatrix}. \quad (4.60)$$

From that we see that $(\theta - 1)v_A = Cv_B$ and $(\theta - 1)v_B = C^T v_A$. Using this definition, we can show that u is an eigenvector of \mathcal{L} for the eigenvalue $2 - \theta$ by the following sequence of equalities

$$\begin{aligned}
\mathcal{L}u &= \begin{pmatrix} I & C \\ C^T & I \end{pmatrix} \begin{pmatrix} v_A \\ -v_B \end{pmatrix} \\
&= \begin{pmatrix} v_A - Cv_B \\ -v_B + C^T v_A \end{pmatrix} \\
&= \begin{pmatrix} v_A - (\theta - 1)v_A \\ -v_B + (\theta - 1)v_B \end{pmatrix} \\
&= (2 - \theta) \begin{pmatrix} v_A \\ -v_B \end{pmatrix}.
\end{aligned}$$

□

Before we show our results, let us define the concept of positive and negative eigenvalue support.

Definition 4.28 (Positive and negative eigenvalue support). Let a and b be strongly cospectral vertices and S its eigenvalue support. Then we say that S^+ is the positive eigenvalue support if for each $\theta_r \in S$, such that $E_r e_a = E_r e_b$, then $\theta_r \in S^+$. Similarly, S^- is the negative eigenvalue support if for each $\theta_r \in S$, such that $E_r e_a = -E_r e_b$, then $\theta_r \in S^-$.

One thing we can notice is that $S^+ \cup S^- = S$ and $S^+ \cap S^- = \emptyset$. Also, if perfect state transfer happens between a and b , by Theorem 4.26, $\theta_r \in S^+$ if and only if $m\theta_r/g$ is even for m and g as defined in the theorem. Similarly, $\theta_r \in S^-$ if and only if $m\theta_r/g$ is odd.

Also, we can show that for two strongly cospectral vertices, neither S^- and S^+ are empty.

Lemma 4.29. *Let a and b be strongly cospectral vertices, a and b different. Then, if S^+, S^- are their positive and negative eigenvalue support, respectively, then $1 \leq \min(|S^-|, |S^+|)$.*

Proof. Let $\mathcal{L} = \sum_r \theta_r E_r$ be the spectral decomposition of \mathcal{L} . We know that $\sum_r E_r = I$, where I is the identity.

Now we will show that the negative eigenvalue support is not empty. We know that $\theta_r \in S^-$ if and only if $E_r e_a = -E_r e_b$. Now, if $S^- = \emptyset$, we can write the following

$$e_a = \sum_r E_r e_a = \sum_{\theta_r \in S^+} E_r e_a = \sum_{\theta_r \in S^+} E_r e_b = \sum_r E_r e_b = e_b, \quad (4.61)$$

where the sum over the $\theta_r \in S^+$ is taken over the eigenprojectors of the eigenvalues in the positive support. This is obviously false. So, $1 \leq |S^-|$. Analogously, we can show that $1 \leq |S^+|$. \square

The Lemma 4.27 gives us what we needed for the following lemma.

Lemma 4.30. *Let G be a tree on two or more vertices. Let c and d be vertices that have perfect state transfer between them and S^-, S^+ be their negative and positive eigenvalue support, respectively. If c and d are on the same bipartite class, then $|S^-| = 1$. Otherwise, c and d are on different bipartite classes and if $\theta \in S^-$, then $\theta = 2/k$ for some odd positive integer k .*

Proof. We know that \mathcal{L} is similar to the matrix $D^{-1}L$. Moreover, we also know that if G has no isolated vertices, then $\mathcal{L}z = \theta z$ if and only if $D^{-1}LD^{-1/2}z = \theta D^{-1/2}z$.

We can see that $D^{-1}L$ is a rational matrix, so if θ is a rational eigenvalue of $D^{-1}L$, we can always find a rational eigenvector z such that $D^{-1}Lz = \theta z$. Moreover, as any multiple of an eigenvector is an eigenvector, we can assume that z is an integer vector with gcd of its entries equal to 1. From Theorem 4.26 we know that all eigenvalues in S are rational, so we deal only with rational eigenvalues here.

We can also note that D being a non-negative matrix changes nothing on the signs of the entries of z , so we can analyze $Lz = \theta Dz$ instead, as we prefer to deal with integers.

Now, let $a, b \in \mathbb{Z}$ such that $\theta = a/b$, so $0 \leq a$ and $1 \leq b$. If $a = 0$, then we can assume that $b = 1$. Let also \deg be a function that returns the degree of a vertex. Let w be a leaf and v its unique neighbor, then we can see that

$$e_w^T Lz = \deg(w)e_w^T z - e_v^T z = e_w^T z - e_v^T z = \frac{a}{b}e_w^T z = \theta e_w^T z. \quad (4.62)$$

The left-hand side is the sum of two integers, therefore, as $\gcd(a, b) = 1$, $b \mid e_w^T z$. This means that

$$e_w^T z - e_v^T z \equiv \frac{a}{b}e_w^T z \equiv 0 \pmod{a}. \quad (4.63)$$

Therefore, $e_w^T z \equiv e_v^T z \pmod{a}$. Now, since G is a tree we can proceed recursively, from the leafs to their unique neighbors, at each step considering vertices that have exactly one neighbor not considered before. We conclude that all entries of z are

equivalent \pmod{a} . Now, if there is perfect state transfer between c and d , then for all $\theta_r \in S^-$, its eigenvectors v_{θ_r} in \mathcal{L} will be such that

$$e_c^T v_{\theta_r} = -e_d^T v_{\theta_r}. \quad (4.64)$$

Because $0 \in S^+$ is a simple eigenvalue with eigenvector $D^{1/2}\mathbf{1}$, then $\deg(c) = \deg(d)$. Moreover, since $D^{-1/2}$ is a non-negative matrix, it changes nothing on the signs of the entries of the v_{θ_r} , therefore if $a/b \in S^-$ it means that for some $v_{\theta_r} = D^{1/2}z$ and we get

$$e_c^T z \equiv -e_d^T z \pmod{a}. \quad (4.65)$$

As we assumed that the gcd of the entries is 1, this can only happen if $a = 1$ or $a = 2$. Now, as G is bipartite and connected, either c and d are part of the same bipartite class or they are from different classes. Assume first that they are from the same bipartite class. From Lemma 4.27, we can see that if $\theta_r = a/b \in S^-$, and as $D^{-1/2}$ does not change the sign, then

$$2 - \theta_r = \frac{2b - a}{b} \in S^-. \quad (4.66)$$

But, again, this means that $2b - a = 1$ or $2b - a = 2$. If $a = 1$, then $b = 1$ or $3/2$. Since $b \in \mathbb{Z}$, then $a/b = 1$. Now, if $a = 2$, then $b = 3/2$ or $b = 2$, which means that $a/b = 1$. Therefore, if a and b are in the same bipartite class, a/b can only be equal to 1 and from Lemma 4.29 we get that $|S^-| = 1$.

Suppose now, that c and d are in different bipartite classes. This means that $2 - \theta_r \in S^+$. Therefore, as we assumed that perfect state transfer happens between c and d , then for m and g as defined in Theorem 4.26

$$\frac{m}{gb}a \text{ is odd} \quad \text{and} \quad \frac{m}{gb}(2b - a) \text{ is even.}$$

Now, if $a = 1$, $m/(gb)$ must be odd, which implies that $2b - a$ is even. But, as a is odd, this is not possible.

The only other possible solution is that $a = 2$, in that case, since we assumed that $\gcd(a, b) = 1$, then b is odd. \square

Before our next lemma, we define what we mean when we say that two vertices are twins.

Definition 4.31 (Twin vertices). Let a and b be vertices. Let also $N(a)$ and $N(b)$ be the set of neighbors of a and b , respectively. Then a and b are said to be twins if either

$N(a) = N(b)$ or $a \cup N(a) = b \cup N(b)$.

So, twins can be adjacent to each other or not, but they must share every other neighbor. It was shown in Coutinho and Liu [2015] that if a and b are strongly cospectral vertices for the Laplacian and $|S^-| = 1$, then a and b are twins. We can show the same result for the Normalized Laplacian below.

Lemma 4.32. *Let a and b strongly cospectral vertices of a connected graph according to the Normalized Laplacian matrix. Let also S^+, S^- be their positive and negative eigenvalue support. If $|S^-| = 1$, then a and b are twins.*

Proof. Let $\mathcal{L} = \sum_r \theta_r E_r$ be the spectral decomposition of the Normalized Laplacian. Define the following vectors

$$z^+ = \sum_{\theta_r \in S^+} E_r e_a, \quad z^- = \sum_{\theta_r \in S^-} E_r e_a.$$

As a and b are strongly cospectral, we can see that

$$z^+ + z^- = \sum_{\theta_r \in S^+} E_r e_a + \sum_{\theta_r \in S^-} E_r e_a = \sum_r E_r e_a = e_a,$$

and also,

$$z^+ - z^- = \sum_{\theta_r \in S^+} E_r e_a + \sum_{\theta_r \in S^-} (-E_r e_a) = \sum_{\theta_r \in S^+} E_r e_b + \sum_{\theta_r \in S^-} E_r e_b = \sum_r E_r e_b = e_b.$$

Therefore, we conclude that $z^+ = \frac{1}{2}(e_a + e_b)$ and $z^- = \frac{1}{2}(e_a - e_b)$. We can also see that for each $E_r e_a$

$$\mathcal{L} E_r e_a = \theta_r E_r e_a.$$

Therefore, if $|S^-| = 1$, we can see that z^- is an eigenvector of \mathcal{L} for some eigenvalue $\theta \neq 0$, as $0 \in S^+$. Now, define δ to be the number of common neighbors of a and b , A the set of neighbors of a , B the set of neighbors of b and $d(\cdot)$ a function that maps the vertices to its degrees. We showed in the previous lemma that if a and b are strongly cospectral, then they have the same degree. Therefore, if a is a neighbor of b

$$\begin{aligned}
\mathcal{L}(e_a - e_b) &= \left(e_a - \sum_{v \in A} \frac{e_v}{\sqrt{d(a)d(v)}} \right) - \left(e_b - \sum_{v \in B} \frac{e_v}{\sqrt{d(b)d(v)}} \right) \\
&= e_a - e_b - \sum_{v \in A-B} \frac{e_v}{\sqrt{d(a)d(v)}} + \sum_{v \in B-A} \frac{e_v}{\sqrt{d(b)d(v)}} \\
&= \left(1 + \frac{1}{\sqrt{d(a)d(b)}} \right) (e_a - e_b) + \sum_{v \in (A-B) \setminus b} \frac{e_v}{\sqrt{d(a)d(v)}} + \sum_{v \in (B-A) \setminus a} \frac{e_v}{\sqrt{d(b)d(v)}} \\
&= \theta(e_a - e_b).
\end{aligned}$$

The last equality implies that a and b must share all of their neighbors. Now, if a and b are not adjacent, similarly

$$\begin{aligned}
\mathcal{L}(e_a - e_b) &= \left(e_a - \sum_{v \in A} \frac{e_v}{\sqrt{d(a)d(v)}} \right) - \left(e_b - \sum_{v \in B} \frac{e_v}{\sqrt{d(b)d(v)}} \right) \\
&= e_a - e_b - \sum_{v \in A-B} \frac{e_v}{\sqrt{d(a)d(v)}} + \sum_{v \in B-A} \frac{e_v}{\sqrt{d(b)d(v)}} \\
&= \theta(e_a - e_b).
\end{aligned}$$

Again we have the same implication in the last equality. This means that they must be twins. \square

We know that P_2 has perfect state transfer between its only two vertices as $\mathcal{L} = L$ in this case and in the book Godsil [2015] and in Coutinho and Liu [2015] it is shown that for the Laplacian matrix perfect state transfer happens in this case. This, together with the previous lemma, leads us to the following corollary.

Corollary 4.33. *Let G be a tree with two or more vertices. Assume that a and b are vertices in different bipartite classes of the tree, such that perfect state transfer happens between them. Let also S^- be their negative eigenvalue support. If 2 is the only eigenvalue in S^- , then $G = P_2$.*

Proof. From Lemma 4.32 we know that if $|S^-| = 1$, then a and b are twins. Since a and b are in different bipartite classes, they must be connected. They can not have another neighbor, otherwise there would be a cycle in G . Therefore, $G = P_2$. \square

Now, from Lemma 4.30 we know that if perfect state transfer happens in a tree between a and b and they are in different bipartite classes, all the eigenvalues in S^- are of the form $2/k$, for k an odd positive integer.

This corollary above shows that if we cannot find non-integer eigenvalues of this form in the graph, then, unless the graph is P_2 , there is no perfect state transfer between

vertices in different bipartite classes. This observation simplified the search significantly. Calculation done in SAGE, using CoCalc, showed no trees with eigenvalues of this form for trees with up to 16 vertices.

Now, for P_3 , one can check that $\mathcal{L}(P_3)$ has eigenvalues

$$0, \quad 1, \quad 2 \tag{4.67}$$

with the respective eigenprojectors

$$\frac{1}{4} \begin{pmatrix} 1 & \sqrt{2} & 1 \\ \sqrt{2} & 2 & \sqrt{2} \\ 1 & \sqrt{2} & 1 \end{pmatrix}, \frac{1}{2} \begin{pmatrix} 1 & 0 & -1 \\ 0 & 0 & 0 \\ -1 & 0 & 1 \end{pmatrix}, \frac{1}{4} \begin{pmatrix} 1 & -\sqrt{2} & 1 \\ -\sqrt{2} & 2 & -\sqrt{2} \\ 1 & -\sqrt{2} & 1 \end{pmatrix}.$$

Using Theorem 4.26, we can easily check that the end-vertices, in this case corresponding to the first and third row/columns, are strongly cospectral. Moreover, the support S of both vertices is equal to $\{0, 1, 2\}$, and they are all rational.

Finally, we can easily check that, following the definitions of the characterization of the theorem, for P_3 , $m = 1$ and $g = 1$. Moreover, $S^- = \{1\}$ and $S^+ = \{0, 2\}$. Therefore, the third condition is true and so P_3 according to the Normalized Laplacian also has perfect state transfer between its end nodes.

It is also immediate to see that the vertex in the middle is not strongly cospectral to any of the other two vertices, so perfect state transfer happens only between the end-vertices. Furthermore, as shown in the Theorem 4.26, the minimal positive time that it happens is π .

One might ask if there are more paths or trees that exhibit perfect state transfer between any pair of its vertices for the Normalized Laplacian. In Alvir et al. [2014], it is shown that if the graph is a path on more than 3 vertices, this is not possible.

These results reduced significantly the possible cases for perfect state transfer in trees. There are still trees that are not fully covered in these cases, so it is still an open question whether there are more trees that allow for perfect state transfer. Considering that this is also an open question for trees with respect to the adjacency matrix, it is an interesting research topic for the future to fully characterize perfect state transfer in trees with respect to the Normalized Laplacian, as the tools necessary for it could also be helpful to deal with the adjacency matrix problems.

4.5 Pretty Good State Transfer in Paths

In this Chapter, we provided a new characterization of perfect state transfer for the Normalized Laplacian, as the known results assumed that the matrix is an integer matrix. However, for pretty good state transfer, Theorem 1.8 assumes only that the matrix is symmetric. So it still holds for the Normalized Laplacian.

When we look for graphs that present state transfer, one of the first graphs to be considered are paths. If we think about how to construct a computational circuit, paths are a particular type of graph that can be useful if not for the circuit itself, then for connecting circuits to transfer states between them.

Paths and state transfers have been a constant topic of research in quantum walks. Typically, one would want for paths to have perfect state transfer between the end-nodes, as this would mean that one could successfully transfer states between them and therefore transfer the state over circuits connected by them.

Unfortunately, this is not always the case. In the paper Christandl et al. [2004], it was shown that perfect state transfer over the adjacency matrix for paths of n vertices can only occur for $n \leq 3$. In Coutinho and Liu [2015], it was shown that, for the Laplacian matrix, no tree on more than two vertices admits perfect state transfer. This means that only the path P_2 admits perfect state transfer for the Laplacian matrix.

Moreover, Alvir et al. [2014] showed that, for the Normalized Laplacian of paths, perfect state transfer between the end-nodes does not happen for $n \geq 4$. We also showed in Section 4.4 some conditions for perfect state transfer in the Normalized Laplacian in trees.

All these restrictions show that, at least for paths on the models we cited, perfect state transfer is restricted to only a few small graphs. Due to these restrictions, the next step is to look for pretty good state transfer over paths. In Coutinho et al. [2017], it was shown that pretty good state transfer happens between the end-nodes of paths on n vertices for the Laplacian matrix if and only if n is a power of 2. We adapt these results and proofs now applied to the context of the Normalized Laplacian.

For a path on n vertices over the Normalized Laplacian, it is shown in [Chung, 1997, Example 1.4] that the eigenvalues of the graph are of the form

$$\lambda_r = 1 - \cos \frac{\pi r}{n-1}, \text{ for } 0 \leq r \leq n-1. \quad (4.68)$$

Now, as for the characterization of both perfect and pretty good state transfer requires only the eigenvalues in the support, we still need to show which of these eigenvalues are in the support of either the end-vertices. The lemma below, taken from

Godsil [2015], show that all of them are.

Lemma 4.34 (Godsil [2015]). *Let P be a weighted path of length d , with vertex set $\{0, \dots, d-1\}$. Let E_0, \dots, E_{d-1} be the spectral idempotents for $A(P)$. If the end-vertices of P are cospectral, then P is symmetric. Furthermore, if R is the automorphism that swaps the end-vertices of P , then R is a polynomial in $A(P)$ and $R = \sum_{k=0}^{d-1} (-1)^k E_k$.*

This lemma assumes that P is a weighted path, but it allows loops. So the Normalized Laplacian is a possible candidate for $A(P)$. Noticing that the Normalized Laplacian is symmetric and mirror-symmetric, i.e., symmetric over the anti-diagonal, it is a simple exercise, using Theorem 4.5, to show that $\mathcal{L}_{a,a}^k = \mathcal{L}_{b,b}^k$ for a, b the end-vertices and k any non-negative integer.

Therefore, applying the lemma for the Normalized Laplacian, as R is a polynomial in $A(P)$ it commutes with A , so we can see that for any r , by multiplying R on the right by E_r , we get that

$$(-1)^r E_r e_a = (R E_r) e_a = (E_r R) e_a = E_r e_b. \quad (4.69)$$

Moreover, we can see by the discussion of the eigenvectors of tridiagonal matrices in [Godsil, 2015, Section Eigenvectors, Chapter Orthogonal Polynomials] that all the eigenvalues of \mathcal{L} for paths are in the support of either end-nodes. This together with our characterization for perfect state transfer on Theorem 4.26 would show that perfect state transfer does not happen for $n \geq 4$ as we only have to show that not all eigenvalues are rational. Now, we are going to investigate pretty good state transfer between the end-nodes of paths with respect to the Normalized Laplacian.

We know that the end-nodes are cospectral, but we still need to show that they are strongly cospectral. Using Equation 4.68 for the eigenvalues of Paths in the Normalized Laplacian and the following corollary, we show just that.

Corollary 4.35 (Godsil [2015]). *If the eigenvalues of G , a graph, are simple, then cospectral vertices are strongly cospectral.*

First, we need to define what are of roots of unity and cyclotomic polynomials. We define just the necessary concepts. A more in depth look of these definitions can be seen in Cox [2012].

Definition 4.36 (Roots of unity). We call a number α to be a n -th root of unity if α is a root of the integer polynomial $x^n - 1$. In particular, we define ζ_n as $\zeta_n = e^{\frac{2\pi i}{n}}$ and all n -th roots of unity are of the form ζ_n^k for $0 \leq k \leq n-1$.

Definition 4.37 (Primitive roots of unity). We say that ζ_n is a primitive n -th root of unity if the smallest positive integer k such that $\zeta_n^k = 1$ means that $k = n$.

Definition 4.38 (Cyclotomic polynomial). We say that $\Phi_n(x)$ is the n -th cyclotomic polynomial if it is the monic integer polynomial that is the minimal polynomial over \mathbb{Q} of the primitive n -th roots of unity. We can also define it as

$$\Phi_n(x) = \prod_{\substack{1 \leq k \leq n \\ \gcd(k, n) = 1}} (x - \zeta_n^k). \quad (4.70)$$

One thing we can note is that as $\Phi_n(x)$ is the minimal polynomial of the n -th primitive roots of unity, then all polynomials that have ζ_n (or any other n -th primitive root of unity) as a root are divisible by $\Phi_n(x)$. We know the formula for the eigenvalues of the Normalized Laplacian of paths, showed in Equation 4.68. We can also write them as follows

$$\begin{aligned} \lambda_r &= 1 - \cos \frac{\pi r}{n-1} \\ &= 1 - \frac{2 \cos \frac{\pi r}{n-1}}{2} \\ &= 1 - \frac{\zeta_{2(n-1)}^r + \zeta_{2(n-1)}^{-r}}{2} \\ &= 1 - \frac{\zeta_{2(n-1)}^r + \zeta_{2(n-1)}^{2(n-1)-r}}{2}. \end{aligned}$$

In Theorem 1.8 the third condition requires that for all integers ℓ_i such that $\sum_r \ell_r \theta_r = 0$ and $\sum_r \sigma_r \ell_r$ is odd, then $\sum_r \ell_r \neq 0$. Now, for the Laplacian, as $\theta_0 = 0$ and $\sigma_0 = 0$, it was shown that we can simplify this condition to the following.

Corollary 4.39 ([Banchi et al., 2017, Corollary 5]). *Assume M is the Laplacian matrix of a graph with strongly cospectral vertices a and b . Say $\theta_0 = 0$, and $\sigma_0 = 0$. Say the other eigenvalues in their support are $\theta_1, \dots, \theta_d$, and have $\sigma_1, \dots, \sigma_d$ defined as in Theorem 1.8. Then pretty good state transfer occurs between a and b if and only if whenever there are integers ℓ_1, \dots, ℓ_d such that*

$$\sum_{r=1}^d \ell_r \theta_r = 0, \quad (4.71)$$

then

$$\sum_{r=1}^d \sigma_r \ell_r \text{ is even.} \quad (4.72)$$

Moreover, in this case, the complex phase with which pretty good state transfer occurs will be equal to 1.

This corollary also shows that we can define ℓ_0 to be an arbitrary integer. So, we can use it to help us in our proofs.

For the Normalized Laplacian, the same condition is true, meaning $\theta_0 = 0$ is an eigenvalue in the positive support for all vertices. Therefore, we can also use this corollary to show whether pretty good state transfer happens for any vertex.

Here, as it was shown in Banchi et al. [2017], we define $\ell_0 = -\sum_{r=1}^d \ell_r$. This condition applied to the eigenvalues of the Normalized Laplacian gives us that

$$0 = \sum_{r=1}^{n-1} \ell_r \lambda_r = \sum_{r=1}^{n-1} \ell_r \left(1 - \frac{\zeta_{2(n-1)}^r + \zeta_{2(n-1)}^{2(n-1)-r}}{2} \right), \quad (4.73)$$

which implies that

$$0 = \sum_{r=1}^{n-1} \ell_r (-2 + \zeta_{2(n-1)}^r + \zeta_{2(n-1)}^{2(n-1)-r}). \quad (4.74)$$

Now, reorganizing this equation and making $\zeta_{2(n-1)}$ to be the variable x , we can see that it defines the following polynomial $L(x)$

$$L(x) = 2\ell_0 + \sum_{r=1}^{n-1} \ell_r x^r + \sum_{r=n-1}^{2(n-1)-1} \ell_{2(n-1)-r} x^r. \quad (4.75)$$

This allows us to assert that whatever the integer coefficients ℓ_r may be if $\sum_{r=1}^{n-1} \ell_r \lambda_r = 0$, then $\Phi_{2(n-1)} \mid L(x)$.

Therefore, divide $L(x)$ by the cyclotomic polynomial and use the condition that the division must be exact to see what the remainder of this division gives us in terms of restrictions on the integer coefficients ℓ_r . This procedure gives us the following theorem.

Theorem 4.40 (Baptista [2019]). *Pretty good state transfer occurs in the Normalized Laplacian between the end-nodes of a path if and only if the path has $2^k + 1$ vertices with $k \in \mathbb{N}$.*

Proof. As we said, our initial procedure will be given the number of vertices of the path n of a given form, we will divide $L(x)$ by $\Phi_{2(n-1)}$ and, as we will assume that the division will be exact, we should get some polynomial $Q(x)$, such that $L(x) = \Phi_{2(n-1)}Q(x)$ and this will give us a condition on the coefficients ℓ_i .

Let $c_i(p(x))$ be the function that returns the i -th coefficient of the polynomial $p(x)$, i.e., the coefficient of the i -th power of x , if $\deg(p(x)) < i$, then $c_i(p(x)) = 0$.

For, $n = 2^k + 1, k \in \mathbf{N}$, then $\Phi_{2(n-1)} = 1 + x^{n-1}$, we can see that $Q(x) = \sum_{j=0}^{n-2} q_j x^j$ for $q_j \in \mathbb{Z}$. Now, for $\Phi(x)Q(x) = L(x)$, we can see that $c_{n-2}(Q(x)) = \ell_1$. After that, we must have that

$$\ell_2 = c_{n-3}(Q(x))c_{n-1}(\Phi_{2(n-1)}) + c_{n-2}(Q(x))c_{n-2}(\Phi_{2(n-1)}) = q_{n-3}.$$

Similarly, we continue this procedure to get all the coefficients of $Q(x)$, which is equal to

$$2\ell_{n-1} + \sum_{r=1}^{n-2} x^r \ell_{n-1-r}. \quad (4.76)$$

One can readily check, by comparing the product $\Phi_{2(n-1)}Q(x)$ to $L(x)$, that the division is exact if and only if $\ell_{n-1} = \ell_0$ and $\ell_r = \ell_{n-1-r}$ for $r \in \{1, \dots, n-2\}$. So whenever the equation holds, $\sum \ell_{\text{odd}}$ is even and pretty good state transfer occurs.

If $n = p + 1$, where p is an odd prime number, then $\Phi_{2(n-1)} = 1 - x + x^2 \dots + x^{n-2}$. Now, $Q(x) = \sum_{j=0}^{n-1} q_j x^j$ and we can again infer that $q_{n-1} = \ell_1$. Now, for the next coefficient we have

$$\begin{aligned} \ell_2 &= c_{n-2}(Q(x))c_{n-2}(\Phi_{2(n-1)}) + c_{n-1}(Q(x))c_{n-3}(\Phi_{2(n-1)}) \\ &= q_{n-2} - q_{n-1} \\ &= q_{n-2} - \ell_1. \end{aligned}$$

This means that $q_{n-2} = \ell_2 + \ell_1$. Doing the same for q_{n-3} , we get $\ell_2 + \ell_3$. Doing this repeatedly, we see that this pattern continues until we get that

$$\begin{aligned} 2\ell_{n-1} &= \sum_{r=0}^{n-2} c_r(\Phi_{2(n-1)}(x))c_{n-1-r}(Q(x)) \\ &= \sum_{r=0}^{n-2} (-1)^r q_{n-1-r} \\ &= \ell_1 + \sum_{r=1}^{n-3} (-1)^r (\ell_{r+1} + \ell_r) + q_1 \\ &= -\ell_{n-2} + q_1. \end{aligned}$$

We have that $q_1 = 2\ell_{n-1} + \ell_{n-2}$. Finally,

$$\begin{aligned}
\ell_{n-2} &= \sum_{r=0}^{n-2} c_r(\Phi_{2(n-1)}(x))c_{n-2-r}(Q(x)) \\
&= \sum_{r=0}^{n-2} (-1)^r q_{n-2-r} \\
&= \sum_{r=0}^{n-3} (-1)^r (\ell_{r+2} + \ell_{r+1}) - (2\ell_{n-1} + \ell_{n-2}) + q_0 \\
&= -\ell_1 - 2\ell_{n-1} + q_0.
\end{aligned}$$

So, $q_0 = -\ell_1 + \ell_{n-2} + 2\ell_{n-1}$ and thus, $Q(x) =$

$$-\ell_1 + \ell_{n-2} + 2\ell_{n-1} + \ell_{n-1}x + \sum_{r=1}^{n-2} (\ell_{n-1-r} + \ell_{n-r})x^r + \ell_1 x^{n-1}. \quad (4.77)$$

Now, this means that for $L(x) = \Phi_{2(n-1)}(x)Q(x)$ we need that first $2\ell_0 = -\ell_1 + \ell_{n-2} + 2\ell_{n-1}$. Then,

$$\ell_1 = -(-\ell_1 + \ell_{n-2} + 2\ell_{n-1}) + (2\ell_{n-1} + \ell_{n-2}) = \ell_1.$$

$$\ell_2 = (-\ell_1 + \ell_{n-2} + 2\ell_{n-1}) - (2\ell_{n-1} + \ell_{n-2}) + (\ell_{n-2} + \ell_{n-3}) = -\ell_1 + \ell_{n-2} + \ell_{n-3}.$$

$$\ell_3 = -(-\ell_1 + \ell_{n-2} + 2\ell_{n-1}) + (2\ell_{n-1} + \ell_{n-2}) - (\ell_{n-2} + \ell_{n-3}) + (\ell_{n-3} + \ell_{n-4}) = \ell_1 - \ell_{n-2} + \ell_{n-4}.$$

The pattern continues, such that we have that for ℓ_i with $1 \leq i \leq n-2$

$$\ell_i = (-1)^i (\ell_{n-2} - \ell_1) + \ell_{n-1-i}. \quad (4.78)$$

This together with the fact that we defined that $\ell_0 = -\sum_{r=1}^{n-1} \ell_r$ allows us to define a general solution for the coefficients. Let $k = (p-1)/2$, $m = k+1$ and $q = (p \bmod 4)$, so in order for the division to be exact we must have that the ℓ 's can be described as

$$\ell_0 = (-1)^m \left(\sum_{r=m}^{n-2} 2\ell_r + q\ell_{n-1} \right) \quad (4.79)$$

and

$$\ell_i = (-1)^{m+i} \left(\sum_{r=m}^{n-1} 4\ell_r \right) + \ell_{n-1-i}, \quad \forall i \in \{1 \dots k\} \quad (4.80)$$

and the rest of the ℓ 's are free variables.

Now to show that this is a valid solution first, note that the first half of the

equations for ℓ_i with $2 \leq i \leq n-3$ are equal to the second half up to a sign change. Also, we do not need to check the equations $\ell_1 = \ell_1$ and $\ell_{n-2} = \ell_{n-2}$.

For the equation on the ℓ'_i s, as we talked above we can assume that $i < n/2$, by taking (4.80), equating it to (4.78) and replacing ℓ_1 for its expression in (4.80) we can see that the following holds

$$\begin{aligned}
(-1)^{m+i} \left(\sum_{r=m}^{n-1} 4\ell_r \right) + \ell_{n-1-i} &= \ell_i \\
&= (-1)^i (\ell_{n-2} - \ell_1) + \ell_{n-1-i} \\
&= (-1)^i (\ell_{n-2} - ((-1)^{m+1} \left(\sum_{r=m}^{n-1} 4\ell_r \right) + \ell_{n-2})) + \ell_{n-1-i} \\
&= (-1)^i ((-1)^m \left(\sum_{r=m}^{n-1} 4\ell_r \right)) + \ell_{n-1-i}.
\end{aligned}$$

We also have the following equation for ℓ_0 and we can use the same technique using now the formula in (4.79)

$$\begin{aligned}
2(-1)^m \left(\sum_{r=m}^{n-2} 2\ell_r + q\ell_{n-1} \right) &= 2\ell_0 \\
&= -\ell_1 + \ell_{n-2} + 2\ell_{n-1} \\
&= -((-1)^{m+1} \left(\sum_{r=m}^{n-1} 4\ell_r \right) + \ell_{n-2}) + \ell_{n-2} + 2\ell_{n-1} \\
&= (-1)^m \left(\sum_{r=m}^{n-2} 4\ell_r \right) + (2 + 4(-1)^m) \ell_{n-1}.
\end{aligned}$$

Now, if $q = (1 \equiv p \pmod{4})$, then $m \equiv k+1 \equiv (p+1)/2 \equiv 1 \pmod{4}$ and the equation is true. If $q = (p \equiv 3 \pmod{4})$, then $m \equiv k+1 \equiv (p+1)/2 \equiv 2 \pmod{4}$ and the equation is also true. Finally, we defined that $\ell_0 = -\sum_{r=1}^{n-1} \ell_r$ and we can also check that

$$\begin{aligned}
(-1)^m \left(\sum_{r=m}^{n-2} 2\ell_r + q\ell_{n-1} \right) &= \ell_0 \\
&= -\sum_{r=1}^{n-1} \ell_r \\
&= -\sum_{i=1}^{m-1} ((-1)^{m+i} \left(\sum_{r=m}^{n-1} 4\ell_r \right) + \ell_{n-1-i}) - \sum_{i=m}^{n-1} \ell_i.
\end{aligned}$$

By the same analysis as before if $q = 1$, we have that $m-1$ is even, the $4\ell_r$

cancels and therefore the sequence above turns into

$$\begin{aligned}
(-1)^m \left(\sum_{r=m}^{n-2} 2\ell_r + q\ell_{n-1} \right) &= - \left(\sum_{r=m}^{n-2} 2\ell_r + \ell_{n-1} \right) \\
&= - \sum_{i=1}^{m-1} \left((-1)^{m+i} \left(\sum_{r=m}^{n-1} 4\ell_r \right) + \ell_{n-1-i} \right) - \sum_{i=m}^{n-1} \ell_i \\
&= - \sum_{i=1}^{m-1} (\ell_{n-1-i}) - \sum_{i=m}^{n-1} \ell_i \\
&= - \sum_{i=m}^{n-2} \ell_i - \sum_{i=m}^{n-1} \ell_i
\end{aligned}$$

and if $q = 3$, $m - 1$ must be odd and so, we now have

$$\begin{aligned}
(-1)^m \left(\sum_{r=m}^{n-2} 2\ell_r + q\ell_{n-1} \right) &= \left(\sum_{r=m}^{n-2} 2\ell_r + 3\ell_{n-1} \right) \\
&= - \sum_{i=1}^{m-1} \left((-1)^{m+i} \left(\sum_{r=m}^{n-1} 4\ell_r \right) + \ell_{n-1-i} \right) - \sum_{i=m}^{n-1} \ell_i \\
&= \sum_{r=m}^{n-1} 4\ell_r - \sum_{i=1}^{m-1} \ell_{n-1-i} - \sum_{i=m}^{n-1} \ell_i \\
&= \sum_{r=m}^{n-1} 4\ell_r - \sum_{i=m}^{n-2} \ell_i - \sum_{i=m}^{n-1} \ell_i.
\end{aligned}$$

In both cases we see that the equation is valid. Therefore, this is a valid solution. Now, with that description of all ℓ 's possible, as n is always even and using a similar analysis as above for the values of q and m , we get the following sequence of equalities

$$\begin{aligned}
\sum \ell_{\text{odd}} &= \sum_{\substack{i=1 \\ i \text{ odd}}}^{m-1} \left((-1)^{m+i} \left(\sum_{r=m}^{n-1} 4\ell_r \right) + \ell_{n-1-i} \right) + \sum_{\substack{i=m \\ i \text{ odd}}}^{n-1} \ell_i \\
&= \sum_{\substack{i=1 \\ i \text{ odd}}}^{m-1} -(-1)^m \left(\sum_{r=m}^{n-1} 4\ell_r \right) + \sum_{\substack{i=1 \\ i \text{ odd}}}^{m-1} \ell_{n-1-i} + \sum_{\substack{i=m \\ i \text{ odd}}}^{n-1} \ell_i \\
&= \sum_{\substack{i=1 \\ i \text{ odd}}}^{m-1} -(-1)^m \left(\sum_{r=m}^{n-1} 4\ell_r \right) + \sum_{\substack{i=m \\ i \text{ even}}}^{n-2} \ell_i + \sum_{\substack{i=m \\ i \text{ odd}}}^{n-1} \ell_i \\
&= -p(-1)^m \sum_{r=m}^p \ell_r
\end{aligned}$$

and making the same analysis on the possible values of m , pretty good state transfer

does not occur.

We already showed for $n - 1$ equal to a prime and a composite number that is a power of two. So we only need to show for composite numbers that have an odd factor. For this case, we will change our tactics as it would be difficult to check this division for all kinds of cyclotomic polynomials as not all of them behave nicely.

Make $n = mk + 1$ where k is an odd integer greater than 1 and $m \geq 2$. We have that

$$\sum_{r=0}^{k-1} (-\zeta_{2k})^r = 1 + 2 \sum_{r=1}^{(k-1)/2} (-1)^r \cos \frac{\pi r}{k} = 0. \quad (4.81)$$

If we set $q \in \{1, 2\}$ and we multiply the equation above by $\cos \frac{q\pi}{n-1}$ we get

$$\cos \frac{q\pi}{n-1} + 2 \sum_{r=1}^{(k-1)/2} (-1)^r \cos \frac{q\pi}{n-1} \cos \frac{\pi r}{k} = 0. \quad (4.82)$$

After a few manipulations, we arrive at

$$\cos \frac{q\pi}{n-1} + \sum_{r=1}^{(k-1)/2} (-1)^r \left(\cos \frac{\pi}{n-1} (q + rm) + \cos \frac{\pi}{n-1} (q - rm) \right) = 0. \quad (4.83)$$

We can then subtract (4.83) for $q = 1$ from $q = 2$ and after some algebraic manipulations we can arrive at the following

$$\begin{aligned} & \left(\left(1 - \cos \frac{\pi}{n-1} \right) - \left(1 - \cos \frac{2\pi}{n-1} \right) \right) \\ & + \sum_{r=1}^{(k-1)/2} (-1)^r \left(\left(1 - \cos \frac{\pi}{n-1} (1 + rm) \right) - \left(1 - \cos \frac{\pi}{n-1} (2 + rm) \right) \right) \\ & + \sum_{r=1}^{(k-1)/2} (-1)^r \left(\left(1 - \cos \frac{\pi}{n-1} (1 - rm) \right) - \left(1 - \cos \frac{\pi}{n-1} (2 - rm) \right) \right) = 0. \end{aligned} \quad (4.84)$$

We can conclude that

$$(\lambda_1 - \lambda_2) + \sum_{r=1}^{(k-1)/2} (-1)^r \left((\lambda_{rm+1} - \lambda_{rm+2}) + (\lambda_{rm-1} - \lambda_{rm-2}) \right) = 0. \quad (4.85)$$

We can finally see that (4.85) is an integer linear combination of the eigenvalues that is equal to 0. The coefficients are all equal to ± 1 . Furthermore, eigenvalues of odd indexes in the summation all come in pairs. Therefore, considering the first odd indexed eigenvalue in the first pair, the sum of the coefficients of the odd indexed

eigenvalues must be odd. As we need only one integer linear combination of the odd indexed eigenvalues in which the sum of the coefficients of the odd indexed eigenvalues is odd, we can conclude that pretty good state transfer does not occur. \square

From this theorem, we know that the paths that present pretty good state transfer between its end-nodes have a specific form for its number of vertices. Since perfect state transfer implies pretty good state transfer, we provide a new proof to the result in Alvir et al. [2014].

Corollary 4.41. *Perfect state transfer occurs in the Normalized Laplacian between the end-nodes of a path if and only if the path has 2 or 3 vertices.*

Proof. We know that perfect state transfer implies pretty good state transfer. Therefore, from Theorem 4.40 we know that the number of vertices must be of the form $2^k + 1$, for $k \in \mathbb{N}$.

Now, if $k \geq 1$, then the end-nodes must be in the same bipartite class. It follows from Lemma 4.30, that $|S^-| = 1$. As we commented before, the end-nodes are strongly cospectral and then, from Lemma 4.34 the signs of the eigenspaces alternate, thus k can only be 0 or 1.

Moreover, we showed in Section 4.4 that for P_2 and P_3 , in the Normalized Laplacian, there is perfect state transfer between the end-nodes. \square

Chapter 5

Future Work

In this master's thesis, we worked on state transfer on continuous time quantum walks, with a focus on pretty good state transfer. In Chapter 1 we introduced the main concepts needed in the following chapters, we showed some history of quantum computing and some applications and motivations for the study of state transfer in graphs.

In Chapter 2, we showed an algorithm for deciding pretty good state transfer. The main tool in that algorithm was Landau's paper, Landau [1985], on factorization of integer polynomials over algebraic extensions. We used this paper to write the eigenvalues as polynomials on a primitive element of the extension of their splitting field and showed that from these polynomials it is possible to decide pretty good state transfer exactly by computing the general solution to a Diophantine Linear system.

Moreover, we used the same tools to show that it is possible to compute exactly the average mixing matrix of a quantum walk. One line of investigation that we leave is to check if there are any other quantum objects that can be computed with the same tools.

One problem with our algorithm is that it runs in polynomial time on the degree of the splitting field. This is a problem since the size of the splitting field can be exponential on the size of our initial characteristic polynomial. So, one path of investigation that is of particular interest is to improve the complexity of our algorithm.

We know, for instance, that for perfect state transfer the eigenvalues are either all integers or quadratic integers for the adjacency matrix. This means that the degree of the splitting field of the eigenvalues is at most 2. For paths, over the examples we showed, they all can be written in terms of polynomials over primitive roots of unity. This means that the degree of the splitting field is at most polynomial in n . Therefore, it may be possible that we could limit if not for all graphs, at least for some types of graphs, the size of the splitting field.

Since we are dealing with rational polynomials of graphs that have only real roots, it may be possible to determine a limit to the size of the splitting field. Another idea on the same topic is to see if the size of the splitting field of graphs that have pretty good state transfer can be upper bounded.

The importance of upper bounding the size of the splitting field is twofold. First, because it could allow for other characterizations of pretty good state transfer. Secondly, because it could be used to halt the algorithm in case the splitting field computed at a given moment is too large, this could allow for the algorithm to have a lower worst case complexity.

One final idea is to find other means of representing the eigenvalues that could be used to decide if pretty good state transfer occurs in a faster way. On this topic, we showed in Chapter 3 how continued fractions could be used to represent the eigenvalues of the graph. We showed how it could be computed and how in some examples it can be used to determine pretty good state transfer and even the time that we would have to wait for some approximations.

Unfortunately, we could not find any conditions that could be used to determine if pretty good state transfer happens or not for all graphs. Since this is only an initial work on continued fractions for state transfer, there is still some possibility of finding conditions that could allow us to say something about state transfer in general.

Finally, in Chapter 4 we showed a series of results that were known for the adjacency matrix and the Laplacian Matrix, but now applied to the Normalized Laplacian.

First, we showed that there is a connection of two vertices being cospectral in the Normalized Laplacian in terms of the probabilities of staying in the same vertex during a random walk. This gives us another motivation to study quantum walks over the Normalized Laplacian. One work that could be done is to see if strong cospectrality gives us more connections to classical random walks.

Following that, in Section 4.3, we showed a characterization for perfect state transfer over the Normalized Laplacian. This could be used to see how much state transfer differs in a matrix with non-rational weights in terms of the adjacency matrix and the Laplacian matrix and find more examples of state transfer in graphs.

In Section 4.4, we showed some conditions that perfect state transfer in trees imposes on the support of the vertices. The idea was to show a similar result as in Coutinho and Liu [2015], but due to the differences in the eigenvalues, we could not complete our intended task.

One possible idea for the future is to see if there are any more restrictions that we could find for the eigenvalues. We know that for the Laplacian matrix, there is no perfect state transfer for trees on more than 2 vertices. So, considering that the

Normalized Laplacian is a weighted Laplacian, it may still be possible to replicate the result.

Finally, in Section 4.5, following the result shown in Banchi et al. [2017], we showed that pretty good state transfer occurs between the end-nodes of a path if and only if the path has $2^k + 1$ vertices.

This is a similar result obtained in the paper for the Laplacian matrix. There, the number of vertices must be equal to a power of two. This shows a connection, at least in paths, of state transfer between the Normalized Laplacian and the Laplacian.

Therefore, one path of investigation is to see if there are any examples where state transfer differ between the Normalized Laplacian and the Laplacian. Or, in other terms, is it always the case that when some state transfer happens in the Normalized Laplacian that we can create a similar graph, maybe by just removing a vertex, that has the same state transfer in the Laplacian?

Bibliography

- Akritis, A. G. (1980). The fastest exact algorithms for the isolation of the real roots of a polynomial equation. *Computing*, 24:299–313. ISSN 0010485X.
- Akritis, A. G. and King, H. N. (1983). Exact algorithms for polynomial real root approximation using continued fractions. *Computing*, 30:63–76. ISSN 0010485X.
- Alvir, R., Dever, S., Lovitz, B., Myer, J., Tamon, C., Xu, Y., and Zhan, H. (2014). Perfect state transfer in Laplacian quantum walk. *Journal of Algebraic Combinatorics*, 43:801–826.
- Anton, H. and Rorres, C. (2013). *Elementary Linear Algebra: Applications Version, 11th Edition*. John Wiley & Sons Incorporated. ISBN 9781118879160.
- Banchi, L., Coutinho, G., Godsil, C., and Severini, S. (2017). Pretty good state transfer in qubit chains-the Heisenberg Hamiltonian. *Journal of Mathematical Physics*, 58. ISSN 00222488.
- Baptista, P. (2019). Quantum walks in the Normalized Laplacian. Undergrad Thesis, Advised by Gabriel Coutinho, Unpublished Manuscript.
- Bennett, C. H. and Brassard, G. (1984). Quantum cryptography: Public key distribution and coin tossing. *Proceedings of the International Conference on Computers, Systems & Signal Processing*, pages 175–179.
- Blankinship, W. A. (1966). Algorithm 288: Solution of simultaneous linear diophantine equations [f4]. *Communications of the ACM*, 9:514. ISSN 15577317.
- Childs, A. M. and Goldstone, J. (2003). Spatial search by quantum walk. *Physical Review A - Atomic, Molecular, and Optical Physics*, 70.
- Christandl, M., Datta, N., Ekert, A., and Landahl, A. J. (2004). Perfect state transfer in quantum spin networks. *Physical Review Letters*, 92.

- Chung, F. R. K. (1997). *Spectra Graph Theory*. American Mathematical Society Providence, Rhode Island.
- Coutinho, G. (2014). Quantum state transfer in graphs. PhD Thesis, Waterloo 2014.
- Coutinho, G. and Godsil, C. (2017). Perfect state transfer is poly-time. *Quantum Information and Computation*, 17:495–502. ISSN 15337146.
- Coutinho, G., Guo, K., and van Bommel, C. M. (2017). Pretty good state transfer between internal nodes of paths. *Quantum Information and Computation*, 17:825–830. ISSN 15337146.
- Coutinho, G. and Liu, H. (2015). No Laplacian perfect state transfer in trees. *SIAM Journal on Discrete Mathematics*, 29:2179–2188. ISSN 08954801.
- Cox, D. A. (2012). *Galois Theory*, volume 106. John Wiley & Sons.
- Eisenberg, O., Kempton, M., and Lippner, G. (2019). Pretty good quantum state transfer in asymmetric graphs via potential. *Discrete Mathematics*, 342(10):2821--2833.
- Farhi, E. and Gutmann, S. (1997). Quantum computation and decision trees. *Physical Review A - Atomic, Molecular, and Optical Physics*, 58:915–928.
- Ferguson, H. R. P. and Bailey, D. H. (1992). A polynomial time, numerically stable integer relation algorithm. *RNR Technical Report*, pages RNR–91–032.
- Feynman, R. P. (1982). Simulating physics with computers. *International Journal of Theoretical Physics*, 21:467–488. ISSN 00207748.
- Geddes, K. O., Czapor, S. R., and Labahn, G. (1992). *Algorithms for Computer Algebra*. Springer Science & Business Media.
- Godsil, C. (2010). When can perfect state transfer occur? *Electronic Journal of Linear Algebra*, 23:877–890.
- Godsil, C. (2015). *Graph Spectra and Quantum Walks*. Unpublished monograph.
- Godsil, C., Kirkland, S., Severini, S., and Smith, J. (2012). Number-theoretic nature of communication in quantum spin systems. *Phys. Rev. Lett.*, 109:050502.
- Grover, L. K. (1996). A fast quantum mechanical algorithm for database search. *Proceedings of the Annual ACM Symposium on Theory of Computing*, Part F129452:212–219.

- Herrman, R. and Humble, T. (2019). Continuous-time quantum walks on dynamic graphs. *Physical Review A*, 100(1):012306.
- Herrman, R. and Wong, T. G. (2021). Simplifying continuous-time quantum walks on dynamic graphs. *arXiv preprint arXiv:2106.06015*.
- Hoffman, K. and Kunze, R. (1971). *Linear Algebra*. Featured Titles for Linear Algebra (Advanced) Series. Prentice-Hall. ISBN 9780135367971.
- Kempton, M., Sinkovic, J., Smith, D., and Webb, B. (2020). Characterizing cospectral vertices via isospectral reduction. *Linear Algebra and its Applications*, 594:226–248.
- Khinchin, A. (1964). *Continued Fractions*. Phoenix books. University of Chicago Press. ISBN 9780226447490.
- Landau, S. (1985). Factoring polynomials over algebraic number fields. *SIAM Journal on Computing*, 14:184–195. ISSN 00975397.
- Levitan, B. M., Jikov, V. V., and Zhikov, V. (1982). *Almost periodic functions and differential equations*. CUP Archive.
- Pal, H. and Bhattacharjya, B. (2017). Pretty good state transfer on some neps. *Discrete Mathematics*, 340:746–752. ISSN 0012-365X.
- Sharma, V. (2008). Complexity of real root isolation using continued fractions. *Theoretical Computer Science*, 409:292–310. ISSN 03043975.
- Shor, P. W. (1994). Algorithms for quantum computation: Discrete logarithms and factoring. *Proceedings 35th Annual Symposium on Foundations of Computer Science*, pages 124–134.
- Storjohann, A. (2000). Algorithms for matrix canonical forms. PhD Thesis, Swiss Federal Institute of Technology – ETH 2000.
- Trager, B. M. (1976). Algebraic factoring and rational function integration. *Proceedings of the third ACM symposium on Symbolic and algebraic computation - SYMSAC '76*, pages 219–226.
- van Bommel, C. M. (2020). Pretty good state transfer and minimal polynomials. *arXiv preprint arXiv:2010.06779*.