Combinatorial Structures in Quantum Information

University College London

Joshua Lockhart

For Sarah.

Declaration

I, Joshua Lockhart confirm that the work presented in this thesis is my own. Where information has been derived from other sources, I confirm that this has been indicated in the thesis.

My appreciation to **Simone Severini** for his advice, candour and friendship throughout the PhD.
Thanks to **Otfried Gühne** for his insight on all things entanglement. Thanks to **David Roberson** and **Laura Mančinska** for their encouragement, mentorship, and hospitality. Many thanks also to **Toby Cubitt** for his open office door, and to **Carlos González-Guillén** for his supply of counter-examples to conjectures.
Thanks to **Scott Aaronson**, **Aram Harrow**, **Ashley Montanaro**, **Fernando Brandão**, **László Babai**, **Andreas Winter**, **Giannicola Scarpa**, and **Will Matthews** for helpful discussions.
To all those I have shared an office with, **Dan Dervovic**, **Dimitris Stamos**, **Giulia Luise**, **Alexander Botev**, **Nadish de Silva**, **Leonard Wossnig**, **Smudge the rabbit**, **Andrea Rochetto**, **James Watson**, **Ed Grant**, **Carlo Sparaciari**, **Thomas Galley**, **Alexandru Paraschiv** and **Octavio Zapata**: thank you for all the good times. Thanks to **Dan** in particular for going above and beyond to maintain the cookie culture in Research Office 3.14. Big thanks to Cohort One of UCL's Delivering Quantum Technologies CDT for your support and friendship.
Finally, I thank **Sarah Lockhart** for her unwavering support over these four years.

Contents

Abstract

This work is an exploration of how graphs and permutations can be applied in the context of quantum information processing. In Chapter 2 we consider problems about the permutations of the subsystems of a quantum system. Explicitly, we attempt to understand the problem of determining if two quantum states of N qubits are *isomorphic*: if one can be obtained from the other by permuting its subsystems. We show that the well known graph isomorphism problem is a special case of state isomorphism. We also show that the complement of state isomorphism, the problem of determining if two states are *not* isomorphic, can be verified by a quantum interactive proof system, and that this proof system can be made statistical zero knowledge. We also consider the complexity of isomorphism problems for stabilizer states, and mixed states.

In Chapter 3 we work with a special class of quantum states called *grid states*, in an effort to develop a toy model for mixed state entanglement. The key idea with grid states is that they can be represented by what we call a *grid-labelled graph*, literally, a graph forced to have vertices on a two dimensional grid. We show that whether or not a grid state is entangled can sometimes be determined solely from the structural properties of its corresponding grid-labelled graph. We use the grid state framework to build families of bound entangled states, suggesting that even in this restricted setting detecting entanglement is non-trivial and will require more than a single entanglement criterion.

Impact Statement

The thesis focuses on two ways in which mathematical tools more traditionally associated with classical computer science can be brought to bear on questions to do with quantum computing. Quantum computers are known to provide exponential speed up over the best known classical algorithms for a number of well known problems including integer factorisation and solving linear systems of equations.

The first section is immersed in the theory of quantum computational complexity, with the aim of bringing to it fresh ideas from classical computer science. We consider a quantum generalisation of the graph isomorphism problem: the problem of determining if two quantum states are equivalent up to rearrangements of their constituent subsystems. We show that this problem, which we call the state isomorphism problem, has a number of similarities to its classical counterpart in terms of its complexity classification. This work was presented as a poster at a number of quantum computing conferences, with highlights being QIP and TQC, suggesting that it is of interest to the broader theoretical quantum computing community.

It is not clear which aspect of quantum mechanics is responsible for quantum algorithmic speed up. One candidate phenomena is quantum entanglement, the so called “spooky” way in which quantum particles can be correlated with one another, and affect each others state despite potentially being astronomically far apart. In this latter part of the thesis, we consider a special class of quantum states that can be represented by graphs on a grid. We show that properties relating to the entanglement of such a state can be deduced by considering structural properties of its corresponding graph. The work in this direction has resulted in a publication in Physical Review A. The work is intended to be inter-disciplinary,

with the aim of fostering collaboration between the quantum information and combinatorics communities. The work was presented as a talk at Eurocomb 2017 and the British Colloquium for Theoretical Computer Science.

Complexity Class Glossary

P: Polynomial time
NP: Non-deterministic Polynomial time
PSPACE: Polynomial Space
BPP: Bounded error Probabilistic Polynomial time
BQP: Bounded error Quantum Polynomial time
QMA: Quantum Merlin Arthur
QCMA: Quantum Merlin Arthur with classical certificate
QAM: Quantum Arthur Merlin
QCAM: Quantum Arthur Merlin with classical certificate
IP: Interactive Proofs
QIP: Quantum Interactive Proofs
QSZK: Quantum Statistical Zero Knowledge
QCSZK: Quantum Statistical Zero Knowledge with classical communication
MA: Merlin Arthur
AM: Arthur Merlin

Chapter 1

Introduction

A computer that can access advanced processing power by tapping into the realm of quantum physics is an idea that could have come straight from the pages of a science fiction novel. However, at the time of writing a Canadian startup have received $9 million in funding to investigate *quantum machine learning* [1], China and the USA are engaged in what Bloomberg calls a *"Quantum Computing Arms Race"* [2], and Microsoft have just released a Visual Studio plugin for Q#, their *quantum programming language* [3]. It appears that science fiction is well on its way to becoming fact.

Despite this recent explosion of activity, this field is not a new one. The idea of a quantum computer is rooted in Feynman's 1981 talk "Simulating Physics with Computers" [4], in which he proposes that in order to provide an accurate simulation of a quantum mechanical system, a computer must operate according to quantum laws. Four years later, David Deutsch mapped out some of the theoretical underpinnings of such a computing device [5], before showing in 1992 with Richard Jozsa [6] that they could outperform classical computers at certain, rather contrived, tasks.

It wasn't until 1994, when Peter Shor [7] came up with a quantum algorithm that could efficiently find the prime factors of large numbers, that the broader scientific community started to pay attention. Here was a new kind of computing device that, if built, would allow its users to break the RSA cryptosystem [8], built on the assumption that this exact task was impractical. RSA formed, and

still forms, the backbone of secure Internet communications, so the implications of such a device being built would be catastrophic. Another key result came in 1996, when Lov Grover [9] showed that quantum computers are inherently better than classical computers at a foundational problem in information processing: searching through unordered lists of data. Suppose you have a list of N elements, and you are tasked with finding the location of a particular item on that list. In the worst case, the marked item could be the last element: this would require you to check all entries of the list, $\mathcal{O}(N)$ queries. Counter-intuitively, Grover's quantum search algorithm can find the marked element in at most $\mathcal{O}(\sqrt{N})$ queries of the list elements. This has immediate practical applications, even for NP-complete problems like finding a satisfying assignment to a boolean function of N variables. The naive classical brute force search would require you to check $\mathcal{O}(2^N)$ variable assignments in the worst case. With Grover, the worst case complexity would be $\mathcal{O}(2^{N/2})$ queries.

Building a practical quantum computing device that can realise these computational capabilities is a difficult problem. The challenge arises from the fact that the computational primitive of such a computer, the qubit, must be constructed from a physical system affected by the laws of quantum mechanics. Such systems, perhaps photons of light [10], individual ionized atoms [11], or nuclear spins of atoms "trapped" by an impurity in a semi-conducting material [12] tend to be difficult to gain experimental control over. Worse, even if sufficient control is obtained over these systems and they are prepared into a quantum mechanical state usable for information processing, sometimes they interact with their surroundings, diluting the quantum nature of their state. If they interact with their environment too much, the quantum state *decoheres* into a classical state.

In the years surrounding Shor's and Grover's algorithms the importance of protecting this *quantum information* from environmental noise became clear. To this end, a veritable menagerie of quantum error correcting codes sprang up: the Shor code [14] which encodes the state of an error corrected *logical* qubit in the collective state of nine physical qubits, the Steane [18] code which combines classical error correcting codes into a quantum code, later generalised into the CSS (Calderbank-Shor-Steane) codes and *stabilizer codes* [15]. More recently, topological codes based on Kitaev's toric code [17] have come to the fore, with Google

focusing on utilising such schemes in their experimental quantum computing hardware [19].

In the past decade, governmental and business interest in quantum computing has grown substantially. More recent discoveries in quantum algorithms that have applications in machine learning, for instance the Harrow-Hassidim-Lloyd (HHL) linear systems algorithm [20], have captured the attention of the tech giants Google [21], Microsoft [22], Alibaba [23], IBM [24], and Baidu [25]. These companies now fund their own quantum research groups, attracting both experimental and theoretical researchers in an attempt to be the first to build the first scalable quantum computing hardware. Governments on the other hand are interested in the security offered by quantum information-based communication schemes (see the outline given in the overview of the EU's quantum technologies flagship programme [26]), and of course the code-breaking capabilities offered by Shor's algorithm [27].

Quantum computer science

In the past fifty years we have visited the moon, deciphered our own genetic code, and built a communications network that carries 1.2 zettabytes per year[1]. All of this was achieved with computers that are either oblivious to quantum effects, or are designed to cancel them out. Quantum theory is the best known description of how the universe operates, so just what are we missing out on by not building computers that are affected by it? If we can build devices that utilise the quantum effects, then what are their capabilities (and limitations)? As we have seen, the early results (*c. f.* Shor, Grover, HHL) look promising, a fact which coupled with the (perhaps) near-term feasibility of implementing these devices provides scientists a wonderful opportunity to study this question, which cuts to the core of theoretical physics and computer science.

Stephen Cook's 1971 paper "The Complexity of Theorem-Proving Procedures" [29], along with Richard Karp's "Reducibility among combinatorial problems" [30] which came a year later, provided much of the bedrock on which the theory of *computational complexity* was built. Computational complexity theory takes a

[1] A zettabyte is one trillion gigabytes. See [28] for more statistics on total Internet traffic.

model of computation, for example, the Turing machine, and attempts to categorise computational problems in terms of the resources required for that model to solve them. Karp considers time as a resource, in an attempt to find out what a Turing machine can do within a number of time steps that scales as some polynomially bounded function in the length of their inputs (a concept that we henceforth refer to as the Turing machine running in *polynomial time*). He considers two classes of problem: P, those for which a Turing machine can *find* a solution in polynomial time, and NP, those for which a Turing machine can *verify* a solution in polynomial time. He also writes about what would later become known as a *Karp reduction* between computational problems. A problem A is said to be Karp reducible to a problem B (denoted A $\leq_p$ B) if any instance of A can be transformed into an instance of B in polynomial time. Thus, if one has a polynomial time algorithm for B, one can use it to solve any instance of A: just translate it to an instance of B (in polynomial time, by definition) and solve it with the algorithm for B.
Cook's paper [29] deals with problems solvable in polynomial time by nondeterministic Turing machines, a model of computation thought to be strictly more powerful than standard Turing machines. Through the lens of Karp's work, his main result can be stated in the Cook-Levin Theorem. This theorem also bears L. A. Levin's name, due to work in this direction that he conducted independently at around the same time as Cook [31].

Theorem 1.0.1 (The Cook-Levin Theorem) *All problems in* NP *are Karp reducible to the boolean satisfiability problem.*

A problem in NP that has the property described in the Cook-Levin Theorem is referred to as NP-complete, and is considered in all likelihood to be computationally intractable. Karp's paper lists twenty one such problems.
Almost fifty years later we know of many more complexity classes, but P and NP, along with the notion of a Karp reduction, are still central ideas in the theory of computational complexity. The set of NP-complete problems, essentially those that are as hard as anything belonging in NP, now touches every corner of modern scientific endeavour, encompassing problems in fields as diverse as graph theory, operations research, economics, and even computer games [32, 33, 34].

The ideas above are not tied to one model of computation. We can attack other models, provided that they are defined rigorously. Naturally, complexity theory has been focused on analysing the computational capabilities of quantum computers too. In 1997, Bernstein and Vazirani [35] built on Deutsch's work on quantum Turing machines, considering a quantum variant of P that they call BQP. Later, Watrous [36] defined QMA, intended to capture the notion of a problem being efficiently verifiable by a quantum computer (as we will see later, the probabilistic nature of quantum computing means that this class is not so much a "quantum NP", more "quantum Merlin-Arthur"). In a 2002 book on theoretical aspects of quantum computing, Kitaev, Shen and Vyalyi [37] demonstrated that the problem of determining the ground state energy of a local Hamiltonian is complete for this class QMA. This result was improved several times, culminating in Kempe, Kitaev and Regev's result [40] showing that it is QMA-complete even for Hamiltonians where all terms are 2-local. The theory of quantum computational complexity was well on its way, with a notion of efficient computation (the problems in BQP) and a suite of problems thought to be too difficult for quantum computers to solve efficiently (the QMA-complete problems). An excellent survey on these ideas is [38] by Watrous, and a sizeable list of problems known to be QMA-complete is compiled in [39].

This thesis is concerned with how ideas from combinatorics can be applied to topics related to quantum information processing. Our focus in particular will be on computational problems about quantum states. In Chapter 2 we consider the problem of determining if two quantum states are isomorphic, a problem directly inspired by the well known *graph isomorphism* problem. If one considers the problems that have received the most attention in quantum complexity theory, the influence of theoretical physics on the direction of the field becomes quite obvious: many of the problems are about Hamiltonians for instance. With the state isomorphism problem, the idea is to explore analogues between the quantum and classical complexity theories. The graph isomorphism problem is particularly thorny and difficult to handle in a complexity context, so with the state isomorphism problem I wanted to know if this thorniness could be made to manifest itself in a quantum problem. The work in Chapter 2 is based on research I conducted in collaboration with Carlos E. González Guillén [41].

In Chapter 3 we explore the problem of determining if a quantum state is *entangled* or *separable*. Quantum entanglement is a key component of quantum mechanics, and is thought to play a role in the speed up offered by quantum algorithms. However, determining if a quantum system is entangled given a formal description of its current state is known to be NP-hard (see Gurvits [124], and later work by Ioannou [125] and Gharibian [181]). In the chapter we consider how to detect entanglement in a class of quantum states called grid-states, each parametrised by a special kind of graph. The entanglement properties of these states can often be determined by considering structural properties of their corresponding graphs. Chapter 3 is based on [42] and [43], written in collaboration with Simone Severini and Otfried Gühne.

It is important to be clear on some basic notions in quantum information before we proceed with the rest of the thesis. We will now outline some definitions and basic ideas in this area. For a complete treatment of these topics, readers should consult any of a number of excellent introductions to quantum information that exist at the time of writing [44, 45, 46].

1.1 Quantum preliminaries

By convention, we represent the state of a quantum mechanical system with a complex column vector. In particular, the valid states of a quantum system with d degrees of freedom are the column vectors in $\mathbb{C}^d$ with ℓ_2-norm equal to 1. Such *state vectors* comprise the *state space* of the system, and are written in Dirac bracket notation as $|\psi\rangle$. The conjugate transpose of such a state vector $|\psi\rangle = \Big(\psi_1, \ldots, \psi_d\Big)^T$ is denoted $\langle\psi| = \Big(\psi_1^*, \ldots, \psi_d^*\Big)$. In this way the ℓ_2-norm condition on $|\psi\rangle$ is expressed $|\langle\psi|\psi\rangle|^2 = 1$.

Consider a composite system of $n \geq 2$ systems with respective state spaces $\mathbb{C}^{d_1}, \ldots \mathbb{C}^{d_n}$. The state space of such a composite system is the unit vectors of the tensor product space $\mathbb{C}^{d_1} \otimes \cdots \otimes \mathbb{C}^{d_n}$. Suppose we know the respective quantum states of each constituent system $|\psi_1\rangle, \ldots, |\psi_n\rangle$. The state of the composite system is obtained by taking the *Kronecker product* of these states. The Kronecker product of two states $|\psi\rangle = \Big(\psi_1, \ldots, \psi_q\Big)^T \in \mathbb{C}^p$ and $|\phi\rangle = \Big(\phi_1, \ldots, \phi_q\Big)^T \in \mathbb{C}^q$ is the vector

$$\begin{pmatrix} \psi_1 \cdot \begin{pmatrix} \phi_1 \\ \vdots \\ \phi_q \end{pmatrix} \\ \vdots \\ \psi_p \cdot \begin{pmatrix} \phi_1 \\ \vdots \\ \phi_q \end{pmatrix} \end{pmatrix} =: |\psi\rangle \otimes |\phi\rangle.$$

Explicitly then, the state of the composite state is denoted $|\psi_1\rangle \otimes \cdots \otimes |\psi_n\rangle$, and belongs to $\mathbb{C}^{d_1} \otimes \cdots \otimes \mathbb{C}^{d_n}$.

In this thesis we will often consider quantum systems with $d = 2$ degrees of freedom. Such systems are commonly called *qubits* (**qu**antum **bits**). From the above discussion, we know that the state space of a system of N qubits is the unit vectors in $\mathbb{C}^{2^N}$. The *computational basis states* which are used to represent classical information on a quantum computer are a special subset of the states of

N qubits. First, define the single qubit states

$$|0\rangle := \begin{pmatrix} 1 \\ 0 \end{pmatrix} \text{ and } |1\rangle := \begin{pmatrix} 0 \\ 1 \end{pmatrix},$$

and let $x = x_1 \dots x_N \in \{0,1\}^N$ be a bitstring of length N. Finally, define the corresponding computational basis state $|x\rangle := |x_1\rangle \otimes \cdots \otimes |x_N\rangle$. The reader can verify that $|x\rangle$ has all coefficients equal to 0 except the coefficient in position $\sum_{i=0}^{N} x_i \cdot 2^i$, which is equal to 1. Occasionally it will make the discourse easier to parse if we use various shorthand ways of writing the state of a composite system. For example, the expressions $|01\rangle, |0,1\rangle, |0\rangle|1\rangle$, and $|0\rangle \otimes |1\rangle$ are taken to mean the same thing.

Note that $\{0,1\}^N$ for an integer $N \geq 1$ is standard notation used to denote the set of bitstrings of length N, which we will use throughout the thesis. We will also use $\{0,1\}^*$ to denote the set of all bitstrings $\cup_{N\geq 1}\{0,1\}^N$, and the $|\cdot|$ operator to denote the length of a bitstring. That is, for all bitstrings $x_1 \dots x_n$, we have that $|x_1 \dots x_n| = n$.

Definition 1.1.1 (Pure state entanglement) *A bipartite state $|\psi_{AB}\rangle \in \mathcal{H}_{AB}$ is said to be* separable *if there exists $|\psi_A\rangle \in \mathcal{H}_A$ and $|\psi_B\rangle \in \mathcal{H}_B$ such that $|\psi_{AB}\rangle = |\psi_A\rangle \otimes |\psi_B\rangle$. If a state is not separable then it is called* entangled.

The Bell states are an important family of bipartite entangled pure states

$$|\Psi^{\pm}\rangle := \frac{1}{\sqrt{2}}\left(|00\rangle \pm |11\rangle\right);$$
$$|\Phi^{\pm}\rangle := \frac{1}{\sqrt{2}}\left(|01\rangle \pm |10\rangle\right),$$

which we will come across several times in the thesis.

A quantum computer is simply a system of N qubits that can undergo evolution that is controlled by an external user. The evolution of a quantum mechanical system is modelled by applying particular unitary matrices to the system's state vector. Consider a quantum system with state space $\mathbb{C}^d$. The quantum evolutions of such a system are described by the *unitary* matrices on $\mathbb{C}^d$: recall that a matrix U is unitary when $U^\dagger U = UU^\dagger = I$.

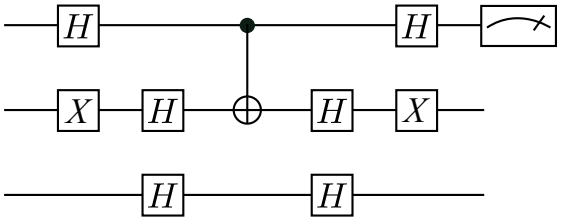


Figure 1.1: A three qubit quantum circuit.

In analogy to the notion of building a logic circuit from gates like AND, OR and NOT, in quantum computing we build quantum circuits out of simple unitary matrices called *quantum gates*. Some quantum gates that are commonly used are the controlled-NOT gate (CNOT), and the Hadamard gate. These correspond to the unitary matrices

$$\text{CNOT} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix}, \quad H = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}, \text{ and } \sigma_x = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}$$

respectively. The reader can verify that the CNOT gate is a two qubit gate, which has the following effect on two qubits in the computational basis:

$$\text{CNOT}|i,j\rangle = \begin{cases} |i,j\rangle \text{ if } i = 0; \\ |i,1-j\rangle \text{ if } i = 1, \end{cases}$$

for all $i, j \in \{0, 1\}$. The first qubit acts as the *control* qubit: if it is 1 then the second *target* qubit is acted upon by a NOT. Another set of gates we will come across repeatedly, and which may be familiar to readers with a physics background are those corresponding to the Pauli matrices

$$X = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}, Y = \begin{pmatrix} 0 & i \\ -i & 0 \end{pmatrix}, Z = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}, \text{ and } I = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}.$$

The CNOT gate is part of a wider family of quantum gates with a control qubit. In this thesis, we will make use of the controlled-Z gate, along with the controlled-

SWAP gate with

$$\text{CZ} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & -1 \end{pmatrix}, \text{ and } \quad \text{CSWAP} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix},$$

as their respective unitary matrix representations.

Quantum circuits are represented diagrammatically, with horizontal lines representing single qubits, and gates drawn over them corresponding to the qubits they act upon. Consider one such circuit diagram, illustrated in Figure 1.1. Circuits are read left to right: initially we apply a Hadamard gate on the first qubit along with a Pauli X on the second, then a Hadamard on both the first and second qubits, and so on. The connected circles represent a CNOT with the first qubit as control and the second qubit as target. At the far right the box with the "dial" on the first qubit corresponds to performing a *measurement with respect to Pauli Z*. Quantum mechanical measurements are treated in terms of Hermitian operators known as *observables*. The eigenvalues of the observable of a measurement correspond to the possible outcomes of that measurement. Let $|\psi\rangle$ be the state of a quantum system, and let M be an observable with spectrum $\{\lambda_i, |\lambda_i\rangle\}$. When the measurement is performed, the probability of the λ_i outcome occurring is equal to $\text{Tr}[|\lambda_i\rangle\langle\lambda_i|\psi\rangle\langle\psi|]$. Consider the Z measurement mentioned above. The Hermitian matrix Z has $|0\rangle$ and $|1\rangle$ as $+1$ and -1 eigenvectors respectively, which implies that for an arbitrary single qubit state $|\psi\rangle = \alpha|0\rangle + \beta|1\rangle$, the probability of outcome $+1$ is equal to $|\alpha|^2$, and the probability of outcome -1 is equal to $|\beta|^2$. Note that in this case $|\cdot|$ denotes the absolute value of a complex number.

The quantum states we have considered so far are called *pure states*. They are used when we are certain of the state of the quantum system. This is not a realistic scenario because sometimes we are not sure which state the system has evolved to: perhaps something went wrong with the experimental apparatus, or the system was affected by noise. We represent probabilistic uncertainty in quantum information with *mixed states*. Consider a system in the lab about whose true quantum state we are unsure. Suppose we know that it is in the pure state

$|\psi_1\rangle$ with probability p_1, the pure state $|\psi_2\rangle$ with probability p_2, and so on, up to pure state k with probability p_k. We represent the state of this quantum system with the matrix $\rho = \sum_{i=1}^{k} p_i |\psi_i\rangle\langle\psi_i|$, which is called the *density matrix* of the mixed state of the system we just described, which is said to have the probabilistic ensemble $\{(p_i, |\psi_i\rangle)\}_{i=1,\dots,k}$. The density matrix of a composite system with n subsystems that have respective state spaces $\mathbb{C}_1^d, \dots, \mathbb{C}^{d_n}$ is a $d_1 \dots d_k \times d_1 \dots d_k$ positive semi-definite matrix with unit trace. Mixed states are combined into composite states analogously to the classical case, taking the Kronecker product of the density matrices. Consider such a *multi-particle* density matrix $\rho_{1,\dots,n}$. For a subsystem of index $1 \leq j \leq n$ we can obtain the density matrix of the system obtained by not considering that subsystem by using the *partial trace* with respect to that subsystem. The reduced state is obtained by "tracing out" subsystem j in the following way:

$$\begin{aligned} \rho_{1,\dots,n\setminus j} &= \mathrm{Tr}_j[\rho_{1,\dots,n}] \\ &:= \sum_{i=1}^{d_j} \langle i|_j \rho_{1,\dots,n} |i\rangle_j \end{aligned}$$

In order to describe the reduced subsystems of entangled states, it is necessary to use mixed states. In general, given some multi-particle pure state $|\psi\rangle$ we can find one of its subsystem states by performing the partial trace on the density matrix $|\psi\rangle\langle\psi|$.

Definition 1.1.2 (Mixed state entanglement) *A density matrix ρ on a bipartite Hilbert space $\mathcal{H}_{AB}$ is called separable if it can be written as*

$$\rho = \sum_i p_i \rho_i \otimes \sigma_i, \tag{1.1}$$

where ρ_i and σ_i are density matrices on $\mathcal{H}_A$ and $\mathcal{H}_B$ respectively and for all $i \geq 1$, $p_i \geq 0$ such that $\sum_i p_i = 1$. As before, a density matrix that is not separable is called entangled.

The *trace distance* will be useful as a measure of similarity between two mixed states. Let ρ, σ be mixed states on the same state space $\mathbb{C}^d$. Their trace distance

is the quantity $D(\rho,\sigma) = \frac{1}{2}\|\rho-\sigma\|_1$, where $\|M\|_1 := \mathrm{Tr}[|M|]$ for a matrix M denotes its *trace norm*, equal to the sum of its singular values. Considering the trace distance between two pure states, that is, $\rho = |\psi\rangle\langle\psi|$ and $\sigma = |\phi\rangle\langle\phi|$ for pure $|\psi\rangle, |\phi\rangle \in \mathbb{C}^d$, we have that [2]

$$\begin{aligned} D(|\psi\rangle\langle\psi|, |\phi\rangle\langle\phi|) &= \frac{1}{2}\||\psi\rangle\langle\psi| - |\phi\rangle\langle\phi|\|_1 \\ &= \sqrt{1-|\langle\psi|\phi\rangle|^2} =: D(|\psi\rangle, |\phi\rangle). \end{aligned}$$

We have covered all the basic mathematical tools needed for the rest of the thesis. Let us begin now.

[2] see [47]

Chapter 2

Quantum state isomorphism

The GRAPHISOMORPHISM problem (GI) is a beautiful problem, worthy of any mathematician's attention. However, it has two relatively benign characteristics that when combined, make it a very dangerous problem indeed. GI manages to concurrently be (a) so easy to state that a class of primary school children could start working on problem instances right away, and (b) so difficult to handle using the tools of modern complexity theory that after forty years of being listed in the "Open Problems" appendix of Garey and Johnson [32], its complexity still remains unclassified. While we now have good evidence that it is not NP-hard (see Boppana, Håstad and Zachos [48], and later Schöning [49]), after all this time it still eludes a polynomial time algorithm, despite some reasonable progress on special cases. For instance, it has been known since 1974 that the problem is efficiently solvable for planar graphs [50]. In 1980 it was shown that most instances are efficiently solvable (shown using random graph theory in [51], in a result that was later improved on by Czajka and Pandurangan in 2008 [52]). Through the years the results improved. Notably, Luks showed that it is efficiently solvable for bounded degree graphs [53]. A 1983 algorithm of Babai and Luks [54] was, until relatively recently, the best known algorithm.

In November 2015, Babai claimed to have found a algorithm for GI that runs in *quasi-polynomial* [1] time. The work, a technical tour-de-force that runs to almost ninety pages, could be thought of as the culmination of the now decades-old

[1] An algorithm runs in quasi-polynomial time if there is some $c > 0$ such that for all inputs of length n, it halts in time $O\left(2^{(\log n)^c}\right)$

programme initiated in *Refs.* [53, 54] of applying algorithmic group theory to the symmetry groups of the graphs under consideration.

In early 2017, Helfgott announced that he had found a serious error in Babai's work [55]. When taken into account, this error demoted the algorithm's running time to sub-exponential[2] rather than quasi-polynomial. The error, discovered while Helfgott was preparing to give a Bourbaki seminar on Babai's result, sent shockwaves throughout the community. Five days later though, Babai announced a modified algorithm, and the original result was saved: this new revised algorithm ran in quasi-polynomial time, verified by Babai and Helfgott [58].

An outsider may look at the struggle to find a polynomial time algorithm for graph isomorphism and conclude that no such algorithm exists, that perhaps the problem is not easy after all. However, in 1987, Boppana, Håstad and Zachos (BHZ) [48] showed that if graph isomorphism were NP-hard, then the *polynomial hierarchy* would collapse. The polynomial hierarchy is a complexity theoretic construction that can be thought of as an attempt to generalise P and NP: if NP is the class of problems that have a P verifier (that is, can be verified by a polynomial time algorithm), then we can define the next level in a hierarchy of such classes by considering the class of problems that have an NP verifier, and so on. While not as paradigm-shifting as NP collapsing to P (that is, $\mathsf{P} = \mathsf{NP}$), a collapse of the hierarchy to some higher level is thought to be extremely unlikely. This link between the graph isomorphism problem and the polynomial hierarchy comes from the fact that graph *non*-isomorphism can be verified efficiently by various kinds of *interactive proof system*, an idea that we will unpack in greater depth later in the chapter.

In addition to the theoretical results that show that most instances of GI are solvable in polynomial time, there are also well known software solvers for it [56, 57]. That such software packages exist is perhaps not surprising, GI appears in biomedical [59] and cheminformatics settings [60]. This disconnect between theory and practice suggests that we as computer scientists should give some thought as to how we classify problems as efficiently solvable. Our field is built around the Cobham-Edmonds thesis [61, 62, 63], which takes the terms "solvable

[2] An algorithm runs in sub-exponential time if for all constants $c > 0$, it runs in time $O(2^{n^c})$ on all inputs of length n.

efficiently" and "solvable in polynomial time" to be effectively synonymous. One direction of this '*polynomial time* $\equiv$ *easy*' equivalence is borne out well: problems in P tend to have algorithms with runtime involving relatively small exponents (see the discussion in Section 1.2.1 of Aaronson's mammoth survey of the P vs. NP problem [64]). GI seems to cut against the equivalence hypothesis from the other direction, constituting a problem that is known to be quite easy to solve practically in the vast majority of cases, but which has so far resisted our attempts to prove that it is in P. Either we have a lot more to discover about graph algorithms, about what it means for a problem to be efficiently solvable, or both.

In short, it appears that the graph isomorphism problem highlights a gap in our knowledge about algorithms, or a deficiency in the mathematical tools we use to analyse the computational complexity of problems. Regardless, it has provided theorists with a worthy adversary over the years. In this chapter I formulate a problem about quantum states that I call STATEISOMORPHISM (SI) in the hope of understanding if there are problems about quantum systems that have similar properties to GI. The chapter is spent building up evidence that this problem can be thought of as a quantum version of GI, by demonstrating a number of similarities between the two problems. The work presented here is based on my publication [41]. A number of the definitions and proofs presented in this chapter are very similar or equivalent to the way they appear in the publication itself.

SI is roughly the following problem: given a pair of N qubit quantum states $|\psi\rangle$ and $|\phi\rangle$, can the subsystems of $|\psi\rangle$ be permuted in such a way that the resulting state $|\psi'\rangle$ is close to $|\phi\rangle$ under some suitable distance measure?

In this chapter we show that SI has a number of similar properties to GI: we show that the problem is in QCMA, and is thus efficiently verifiable. We also show that it has an efficient quantum interactive proof system, and that furthermore, such a proof system can be made into a proof system that is statistical zero knowledge. Immediately, these results give evidence against the problem being QCMA-complete: if it was, then all problems in QCMA would have quantum statistical zero knowledge proof systems. In the classical world, it is thought to be unlikely that all problems in NP have (classical) statistical zero knowledge

proof systems, since this would collapse the polynomial hierarchy (see Aaronson [84]). Because of this, it seems unlikely that all problems in QCMA have quantum statistical zero knowledge proof systems. We also consider isomorphism problems for different kinds of quantum states: *stabilizer states* and *mixed states*. We show that these different kinds of states can make the problem easier or harder. In particular, stabilizer states have similar complexity to standard pure states but all interactive proof systems can be done with classical communication only. Considering isomorphism of mixed states however is considerably more difficult, we are able to show that this problem is QSZK-hard. An upper bound for this problem is elusive at this point, as we will see later, it is difficult to even check if two mixed states are similar, never mind isomorphic. This makes even *verifying* that two mixed states are isomorphic under a permutation a difficult task.
In the next section we will explore more formally the mathematical concepts that we will require for a discussion of the analysis of SI. In particular, we cover GI and its mathematical formulation, classical complexity theory including the classes P and NP, and similar notions in probabilistic computation captured by the classes BPP, MA and AM.

2.1 Graph isomorphism

A graph encodes a relationship between a set of objects. The objects being modelled, be they humans, research papers, or banks, are represented by the graph's vertices. If a pair of these objects are related (perhaps by a friendship, a citation, or a loan) then we represent this by linking them together with an edge. The graph allows us to consider an abstract *collective* relationship between a set of objects. A pair of graphs made up of different constituent objects may represent the same abstract collective relationship. Since we care about the relationship between vertices rather than what the vertices themselves represent, we are often interested when this is the case. A pair of graphs that represent the same abstract collective relationship are called *isomorphic*. A pair of isomorphic graphs can be transformed into one another via a transformation called an *isomorphism*. Graph theory is mostly concerned with the study of properties of graphs that are

invariant under isomorphism. In Figure 2.1 we illustrate a pair of isomorphic graphs.

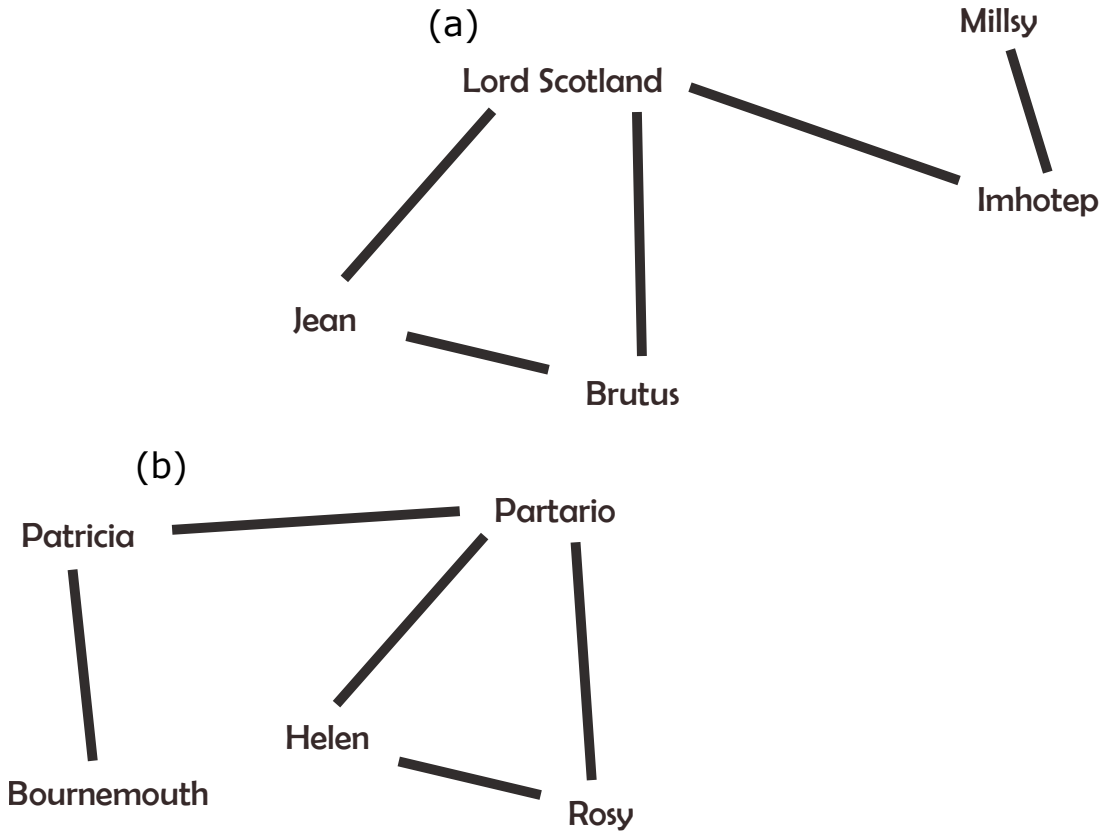


Figure 2.1: Two graphs that are isomorphic. Despite describing the friendship structure of different sets of people, that friendship structure is the same in both graphs: it consists of a clique of three people, with a 'tail' of two people not known to the rest of the clique.

Formally, a graph G is a set of *vertices* V, and a set of unordered pairs of vertices $E \subseteq \binom{V}{2}$ called *edges*. Consider two such graphs $G = (V, E)$ and $H = (W, F)$. If there exists a function $f : V \to W$ that satisfies the condition

$$\{i, j\} \in E \text{ if and only if } \{f(i), f(j)\} \in F, \tag{2.1}$$

then we say that G and H are *isomorphic*. The function f is often called an *isomorphism* between G and H.

In order to define the problem of determining if two graphs are isomorphic, we will use the *decision problem* framework. A decision problem is a set of bitstrings

$D \subseteq \{0,1\}^*$ that correspond to what we will refer to as the YES instances of a problem. Note that $\{0,1\}^*$ denotes the set of all bitstrings, this is notation that we will use quite a lot in the subsequent discussion. As an explicit example of specifying a problem in this framework, consider the following

$$\textsc{EvenHamming} := \{x \text{ has even Hamming weight } : x \in \{0,1\}^*\},$$

which specifies the problem of determining if a bitstring has an even number of 1's. For example, the bitstrings 11, 110, 001111 belong in A, while 1110 does not. We say that an algorithm solves a decision problem A if, for all x, the algorithm outputs 1 if $x \in$ A and 0 otherwise.
It is no longer common to define decision problems using "set-builder" notation like we use above. Instead, problems are usually specified in terms of inputs and the expected outputs of an algorithm that solves it. The graph isomorphism problem is specified in this format like so.

Problem 2.1.1 GRAPHISOMORPHISM
Input: A pair of graphs G and H.
Output: YES *if and only if G and H are isomorphic.*

When we specify problems in this informal way, we are not truly specifying a decision problem. For structured data like graphs, to do so would require a great deal of "book-keeping", for example, we would need to agree on a reasonable representation of a graph as a bitstring. Normally decision problems are not specified to that level of detail, because details about representation are not what we are interested in. Often it is reasonable to just assume that a representation has been agreed on beforehand, and that all data will be given to our algorithm in that format. A cleaner way of specifying computational problems is in terms of *promise problems*. The promise problem framework allows us to assume things about the problem under consideration in a formal way (literally, we make promises about the problem instances that an algorithm is expected to solve). We will revisit these ideas later in Section 2.3.3.
To proceed with our discussion of GI, we will need to define some of the complexity classes we have discussed so far. The class P (**P**olynomial time) is the class of

decision problems that can be solved by a Turing machine [108] in a number of time steps that scales as some polynomial function of the length of the problem instance. Such problems are considered efficiently solvable.

Definition 2.1.2 (P) *Let* A *be a decision problem. We say that* A *is in the class* P *if there exists a polynomial time Turing machine M such that for all $x \in \{0,1\}^*$*

- *if $x \in$* A *then M accepts x;*
- *if $x \notin$* A *then M rejects x.*

The next class, NP (**N**on-deterministic **P**olynomial time), consists of problems whose solutions can be *verified* efficiently. It is useful to consider such problems as having a "P verifier": there is an efficient algorithm that, when given the problem instance and a short bitstring known as a *certificate*, can verify in polynomial time that the problem instance satisfies the constraints of the decision problem. The class is defined below, where the operator '$\circ$' denotes concatenation of bitstrings. Formally, a function $f : \mathbb{N} \to \mathbb{N}$ is polynomially bounded if there exists $c > 0$ such that $f \in \mathcal{O}(n^c)$. We occasionally use poly to denote the set of polynomially bounded functions, or poly(n) to mean some generic function that is polynomially bounded in argument n. Likewise, we occasionally use $\exp(n)$ to mean some generic function that scales exponentially in n.

Definition 2.1.3 (NP) *Let* A *be a decision problem. We say that* A *is in the class* NP *if there exists a polynomial time Turing machine M and a polynomially-bounded function $p : \mathbb{N} \to \mathbb{N}$ such that for all $x \in \{0,1\}^*$,*

- *if $x \in$* A *then there exists a bitstring c of length at most $p(|x|)$ such that M accepts $x \circ c$;*
- *if $x \notin$* A *then for all bitstrings of length at most $p(|x|)$, M rejects $x \circ c$.*

If two graphs *are* isomorphic then it is always possible to describe the isomorphism f between G and H by specifying the action of f on each vertex of G with the cardinality $|V|$ set of tuples $\{(v, f(v)) \ : \ v \in V(G)\}$. In order to verify that such an f does in fact constitute an isomorphism between G and H, all it takes

is to verify the isomorphism condition, Eq. 2.1, for each of G's edges, of which there are at most $|V| \cdot (|V| - 1)$. This reasoning can be used to show that GI is in NP.

Formally, a decision problem A is *hard* for a complexity class C (and referred to as C-hard) if for all problems B $\in$ C, there is a polynomial time *Karp reduction* from B to A. A Karp reduction from a problem B to a problem A is a function $f : \{0,1\}^* \to \{0,1\}^*$ with the property that for all bitstrings $x \in \{0,1\}^*$, we have that $x \in$ B if and only if $f(x) \in$ A. If A is C-hard and A $\in$ C then we say that A is C-*complete*.

Boppana, Håstad and Zachos (BHZ) give complexity theoretic evidence against GI being NP-hard: they show that if this was the case, then the polynomial hierarchy would collapse. The mechanics behind this result provides us with much of the motivation for the results we prove in this chapter, so we will spend some time going through them. The crux of BHZ comes from the fact that GI's *complement* problem, GRAPHNONISOMORPHISM (GNI) is efficiently verifiable using an *Arthur-Merlin interactive proof system*. Let us now discuss the notion of verification using proof systems.

2.2 Probabilistic and interactive proofs

While it is easy to certify that two graphs G and H are isomorphic, it seems almost impossible to efficiently verify that two graphs are *not* isomorphic. Of course, we could check all bijections $f : V(G) \to V(H)$ to see if any of them map G to H. If you try all such bijections, and none of them constitute such a mapping then you know the graphs are not isomorphic. Unfortunately since the number of bijections between G and H scales as an exponential function of the number of vertices, this isn't efficient. It seems unlikely that GNI is in NP.

However, the notion of "verifiability" captured by the class NP is rather strict: a P verifier t be must be convinced by the certificate c that x is a YES instance. Since P is a deterministic class, what we are talking about here is verification without doubt. In the real world we rarely require such certainty. We are often happy to be presented with *predictions* about the truth of a statement, provided

the predictor gives us an idea of how sure they are about the prediction. In complexity theory, we model this notion of being "pretty sure" about something being true by using *probabilistic Turing machines*. A probabilistic Turing machine is the same as a standard Turing machine, but it has access to a source of random bits that it can make use of during the computation. Essentially, the model is just a Turing machine with the ability to flip a fair coin and make decisions based on the outcome.

A clean way of formalising probabilistic Turing machines is to consider a standard Turing machine with access to an additional read-only tape called the *random tape*, which is populated with a sequence of bits at the start of computation. At each step of a computation, the machine can either operate as a standard Turing machine, or serially read the next bit of its random tape. Such a Turing machine M is given two bitstring inputs, x which is placed on its standard work tape, and r which is placed on its random tape. We denote the state of M's post-halting work tape after being given x and r as input by $M_r(x)$. Then M accepts input x with respect to random tape r if $M_r(x) = 1$, and rejects if $M_r(x) = 0$. The acceptance probability of such a Turing machine is proportional to the number of random tape initialisations that cause it to accept. As before, we say that such a Turing machine runs in polynomial time if for all inputs, M halts within a number of time steps that scales as a polynomially bounded function of the length of that input. The class BPP (**B**ounded-error **P**robabilistic **P**olynomial time) is the set of problems that can be solved in polynomial time by a probabilistic Turing machine.

Definition 2.2.1 (BPP**)** *Let* A *be a decision problem. We say that* A *is in the class* BPP *if there exists a polynomial time probabilistic Turing machine M and a polynomially bounded function $p : \mathbb{N} \to \mathbb{N}$ such that for all $x \in \{0,1\}^*$,*

- *if $x \in$* A *then*

$$\frac{1}{2^{p(|x|)}} \left|\{r \in \{0,1\}^{p(|x|)} \ : \ M_r(x) = 1\}\right| \geq 2/3;$$

- *if $x \notin$* A *then*

$$\frac{1}{2^{p(|x|)}} \left|\{r \in \{0,1\}^{p(|x|)} \ : \ M_r(x) = 1\}\right| \leq 1/3.$$

Just like how we defined NP as the class of problems that have P verifiers, it is meaningful to define complexity classes in terms of BPP verifiers. The following, known as MA (**M**erlin-**A**rthur) is one such class.

Definition 2.2.2 (MA**)** *Let* A *be a decision problem. We say that* A *is in the class* MA *if there exists a polynomial time probabilistic Turing machine M and polynomially bounded functions $p, q : \mathbb{N} \to \mathbb{N}$ such that for all $x \in \{0,1\}^*$*

- *if $x \in$ A there exists $c \in \{0,1\}^{q(|x|)}$ such that*

$$\frac{1}{2^{p(|x|+|c|)}} \left|\{r \in \{0,1\}^{p(|x|+|c|)} \ : \ M_r(x \circ c) = 1\}\right| \geq 2/3;$$

- *if $x \notin$ A then for all $c \in \{0,1\}^{q(|x|)}$ we have that*

$$\frac{1}{2^{p(|x|+|c|)}} \left|\{r \in \{0,1\}^{p(|x|+|c|)} \ : \ M_r(x \circ c) = 1\}\right| \leq 1/3.$$

The reason for the strange name is because what is actually happening can be thought of as an interaction between two parties: the computationally unbounded *prover* (the wizard, Merlin), and a probabilistic *verifier* restricted to operate in polynomial time (the somewhat magically-limited King Arthur). Merlin sends information in the form of a certificate c to Arthur in an attempt to convince him with bounded error that a particular assertion is true. We can think of NP in this Prover-Verifier setting too, the only difference being that Arthur is required to output his decision without making mistakes. This argument can be made rigorous to show that MA is a more general class than NP: in other words, that NP $\subseteq$ MA.

A dialogue is often more convincing than a monologue, so it seems unrealistic that the prover should only get one shot at convincing the verifier of something. We can generalise the notion of efficient verification even further by considering verification protocols that consist of multiple rounds of interaction between the parties. Arthur-Merlin proof systems can be generalised to multiple rounds, but first we will consider the class IP, which captures the notion of **I**nteractive **P**roof systems.

2.2.1 Proofs with private coins

Interactive proof systems involve a prover and a verifier, who take it in turns to work over several rounds. When a party has finished working in one round, they send a message to the other party, who then starts working again, and so on. At the end, the verifier outputs a bit signifying its decision. In [65] Goldwasser, Micali and Rackoff define such proof systems in terms of a special kind of Turing machine they call an *interactive* Turing machine. Such a Turing machine has a *work tape*, a *random tape*, and two *communication tapes*: one that is read-only, and the other which is write-only. The work tape and random tape are the same as in probabilistic Turing machines, where the random tape supplies the Turing machine with a source of "coin flips" on which to base its decisions. The communication tapes allow two interactive Turing machines to send messages to one another, as we will now see.

Formally, an *interactive protocol* is a pair of interactive Turing machines (P, V) such that the read-only communication tape of P is the write-only communication tape of V, and the read-only communication tape of V is the write-only communication tape of P. In this way, P can send messages to V that V can read but not modify. Likewise, V can do the same for P. Furthermore, the Turing machine P is computationally unbounded, but V is restricted to run in polynomial time. A k-message interaction (for k even) of (P, V) on an input $x \in \{0,1\}^*$ starts with V running with x as its input and halting. The value on V's write-only tape is message $\#1$. Then P runs and halts. The value on P's write only tape after halting is message $\#2$. The parties run in this way, running while it's their turn, halting while the other is computing, until k messages have been exchanged. In the last round, the verifier halts with a 1 or a 0 on its tape denoting acceptance and rejection of x respectively, as has been the convention thus far. Naturally, the verifier must go last, so for odd k, the prover goes first. The class $\mathsf{IP}(k)$, defined below, is the class of problems that have k message interactive proof systems.

Definition 2.2.3 ($\mathsf{IP}(k)$**)** *Let* A *be a decision problem. We say that* A *belongs to* $\mathsf{IP}(k)$ *for some* $k \geq 1$ *if there exists a* k*-message interactive protocol* (P, V) *such that for all* $x \in \{0,1\}^*$

- *if* $x \in A$ *then after the* k*-message interaction between* P *and* V*,*

$$\Pr[V \text{ is in accepting state}] \geq 2/3;$$

- *if* $x \notin A$ *then after any* k*-message interaction between* V *and any interactive Turing machine* P'

$$\Pr[V \text{ is in accepting state}] \leq 1/3.$$

Returning to our discussion of verifying that two graphs are not isomorphic, it is possible to show that GNI has a 2 message interactive proof system. Let us attempt to gain some intuition as to why this is the case. Consider two graphs with the same vertex set, $G_1 = (V, E)$ and $G_2 = (V, F)$ (if they had a different number of vertices then they couldn't possibly be isomorphic). Denote by $\mathfrak{S}_{|V|}$ the group of permutations of the vertices of the graphs. Consider the following interactive protocol.

1. *Verifier*: Uniformly at random, select an index $i \in \{1, 2\}$ and a permutation $\sigma \in \mathfrak{S}_{|V|}$. Send the permuted graph $G' := \sigma(G_i)$ to the prover.
2. *Prover*: Determine i from G', and send back your best guess j.
3. *Verifier*: ACCEPT if and only if $i = j$.

On step 2, all the prover sees is a graph G', they don't receive the permutation σ or the index i. The claim is that this protocol will allow the verifier to determine if G_1 is *not* isomorphic to G_2, with some reasonable probability better than random guessing.

Consider the case where the graphs are not isomorphic. By definition, for all permutations of the vertices $\sigma \in \mathfrak{S}_{|V|}$ we have that $\sigma(G_1) \neq G_2$. In this case, the prover can try all permutations of the vertices of G'. If one of the permutations yields G_1, then they know that G_1 is the pre-permutation graph. Likewise,

if they came across G_2 during their search through permutations, they know that the original graph must have been G_2. Since the prover is computationally unbounded, they are perfectly within their rights to use such an algorithm (remember, we are interested in how to convince the verifier that the graphs are non-isomorphic, NOT in solving the non-isomorphism problem efficiently).
In the case where the graphs *are* isomorphic to one another then there exists a permutation σ such that $\sigma(G_1) = G_2$. There also exists a permutation π with the property that $\pi(G_2) = G_1$ (of course $\pi = \sigma^{-1}$). There is now insurmountable ambiguity for the prover, since G' could have come from either one of the original graphs. The prover can do no better than to randomly guess the answer.
Determining non-isomorphism for a pair of graphs can be done by playing the above game with the prover enough times to determine if they can genuinely tell the permuted graphs apart from one another, or if they are just guessing randomly. More rigorously, we need to show *completeness* of the protocol: for all non-isomorphic graphs, the prover can convince the verifier that they are non-isomorphic, and *soundness*: no malicious prover (P' in definition 2.2.3) can convince the verifier that two isomorphic graphs are non-isomorphic. We have shown completeness already: the prover can always search through all permutations of G' until they find the graph that it came from. Soundness follows from the fact that the prover, despite being computationally unbounded, can do no better than randomly guess in the case of isomorphism. In this case, the prover's wins and losses in the game will resemble tosses of a fair coin, and we know that this can be efficiently detected by the verifier using elementary statistical techniques. Later, we will show that similar protocols can be employed in the quantum case to verify that two quantum states are non-isomorphic: the STATENONISOMORPHISM problem.

2.2.2 Proofs with public coins

The protocol we described in the previous section is referred to as a *private coin* interactive proof system, since it relies on the fact that the verifier is allowed to withhold from the prover the result of its coin tosses that generate $i \in \{1, 2\}$ and the permutation σ. The above protocol would not work without the ability

to hold this information back from the prover. Remarkably there also exists a *public-coin* proof system for GNI, where all of the verifier's coin tosses are known to the prover.

To proceed we will place a restriction on our two graphs $G_1 = (V, E)$ and $G_2 = (V, F)$, we will assume that they have no non-trivial automorphisms. That is, we assume that neither of the graphs are isomorphic to themselves: $\sigma(G_1) \neq G_1$ and $\sigma(G_2) \neq G_2$ for all $\sigma \in \mathfrak{S}_{|V|}$ with the exception of the identity permutation. We make this restriction to make the proof sketches that follow easier to understand, but it is worth noting that the result holds for graphs with automorphisms too.

Consider the set of graphs that can be obtained from G_i via permutations

$$S_i := \{\sigma(G_i) \ : \ \sigma \in \mathfrak{S}_{|V|}\}.$$

Since G_1 and G_2 have no non-trivial automorphisms, we have that $|G_1| = |G_2| = n!$, where $n = |V|$. Furthermore, if $G_1 \cong G_2$ then the set G_1 is identical to the set G_2, and so in this case we would have $|S_1 \cup S_2| = n!$. Conversely, if $G_1 \ncong G_2$ then for all permutations $\sigma \in \mathfrak{S}_n$, we have by definition that $\sigma(G_1) \neq G_2$. Even stronger, for all pairs of permutations $\sigma, \pi \in \mathfrak{S}_n$, we have that $\sigma(G_1) \neq \pi(G_2)$. So in the case where the graphs are not isomorphic we have that $S_1 \cap S_2 = \emptyset$ and so $|S_1 \cup S_2| = 2n!$.

Summarising, if $G_1 \cong G_2$ then $|S_1 \cup S_2| = n!$, but if $G_1 \ncong G_2$ then $|S_1 \cup S_2| = 2n!$. Determining non-isomorphism becomes a question about the size of the set $S_1 \cup S_2$.

Goldwasser and Sipser [100] demonstrate what has come to be known as a *set lower bound protocol*: they show that there is a public coin interactive proof system for certifying the size of $S_1 \cup S_2$. This protocol can be implemented in the Arthur-Merlin framework to show that GNI is in the class AM, the class of public coin proof systems where Arthur goes first. In an AM proof system, Arthur and Merlin as usual want to determine if some x belongs to a decision problem A or not. Arthur goes first, sending Merlin a randomly generated bitstring s. Merlin then responds with a certificate c, and Arthur uses both s and Merlin's response c to inform his decision as to whether x belongs to A. Crucially, none of Arthur's randomness can be hidden from the prover, Arthur must send all of his random

string s to Merlin, and s must be the only random data he generates. This is captured formally in the following.

Definition 2.2.4 (AM) *Let* A *be a decision problem. We say that* A *is in the class* AM *if there exists a polynomial time probabilistic Turing machine M, polynomially bounded functions $p, q : \mathbb{N} \to \mathbb{N}$, and a collection of bitstrings $\{c_r \ : \ r \in \{0,1\}^{p(|x|)}\}$ each of length $q(|x|)$ such that for all $x \in \{0,1\}^*$*

- *if $x \in$ A then*

$$\frac{1}{2^{p(|x|)}} \left|\{r \in \{0,1\}^{p(|x|)} \ : \ M_r(x \circ c_r)\}\right| \geq 2/3;$$

- *if $x \notin$ A then*

$$\frac{1}{2^{p(|x|)}} \left|\{r \in \{0,1\}^{p(|x|)} \ : \ M_r(x \circ c_r)\}\right| \leq 1/3.$$

It is possible to extend the Arthur-Merlin proof systems to arbitrarily many rounds of interaction, defining the class AM$[k]$ for $k \geq 3$. However, increasing the number of rounds of interaction has no effect since it is possible to prove that AM$[k] =$ AM (see Arora-Barak [108]).

While the restriction to public coins makes AM seem like a particularly weak class, a remarkable fact is that it contains GRAPHNONISOMORPHISM. The proof of this relies on the Goldwasser-Sipser set lower bound protocol [100], which makes use of *pairwise independent hash functions* which are defined in the following way, where we take $x \in_R X$ to mean that x is drawn uniformly at random from a finite set X.

Definition 2.2.5 (Pairwise independent hash functions) *A family of functions $\{f_i : A \to B \ : \ i \in I\}$ is called* pairwise independent *if for all $x, x' \in A$ such that $x \neq x'$, and for all $y, y' \in B$, we have that*

$$\Pr_{i \in_R I}[h_i(x) = y \wedge h_i(x') = y'] = |B|^{-2}$$

For brevity, we will just show the AM protocol for GNI without proving it correct. This is a classic result, so it is proved in a number of places. Interested readers

may consult the discussion in the Arora-Barak textbook [108], or the lecture notes of Jonathan Katz for a neat proof [81].

In what follows, we take $\mathcal{H}_{a,b}$ to be a family of pairwise independent hash functions from $\{0,1\}^a$ to $\{0,1\}^b$. The below protocol depends on suitably chosen a and b. Omitting details, we just state these appropriate sizes as l and m. The set S in the protocol is any finite set, for example the set $S_1 \cup S_2$ referred to previously.

1. *Arthur:* Uniformly at random, select $h_i \in \mathcal{H}_{l,m}$.
2. *Merlin:* Find $G \in S$ such that $h_i(G) = 0^m$. Send G to Arthur, along with a certificate proving that $G \in S$.
3. *Arthur:* Accept if $G \in S$ and $h_i(G) = 0^m$.

The key idea is that the hash function family provides a constraint on Merlin that he must fulfil using data *only* from the set S. Note that Merlin cannot cheat and pick some G not in S that satisfies the constraint, since by construction, Arthur can verify membership in S in polynomial time. Intuitively, if the set S is small then it will be more difficult for Merlin to find a bitstring that satisfies it, and his winning probability is lower. If the set S is big then there are more opportunities for him to find a bitstring that satisfies it, so he can win the game with higher probability. The meat in the proof of correctness is that the difference between $n!$ and $2n!$ is large enough to make these two cases distinguishable when the game is repeated only polynomially many times. It is possible to prove that there is a constant separation in the probability of Merlin being able to find a graph G in S that will satisfy the hash function constraint in each case.

This protocol is sufficient to show that GNI is in the class AM. Another way of stating this result is to say that GI belongs to coAM, where coAM denotes the *complement* of the complexity class AM. More generally, let C be a complexity class. A problem A is in coC if and only if its complement problem $\overline{A}$ is in C.

It is this result that GI $\in$ coAM combined with the fact that GI $\in$ NP, that the BHZ polynomial hierarchy collapse result hinges on. Explicitly, let A $\in$ NP $\cap$ coAM. If A is NP-complete then by definition of completeness, all problems in NP have a polynomial time Karp reduction to A. Since A $\in$ NP $\cap$ coAM this would imply

that NP $\subseteq$ coAM, equivalently, that coNP $\subseteq$ AM, which they show via a technical lemma implies that the polynomial hierarchy collapses.

We have drawn up an outline of some results about the GRAPHISOMORPHISM problem, focusing on how isomorphic graphs can be certified as isomorphic, and also how non-isomorphism can be certified if we consider more general forms of verification than P verifiers. The fact that GRAPHISOMORPHISM is in NP, but that its complement can be efficiently verified in the interactive proof setting can be thought of as a defining characteristic of the problem. It is not clear if there are other problems in NP that have this property, meaning that GI stands out as a potential candidate for what Ladner [68] proves must exist if P $\neq$ NP: an NP-*intermediate* problem, not in P, not NP-complete, but in NP all the same.

Having focused on efficient verification of GNI, we now work on proving that STATENONISOMORPHISM (SNI) has similar characteristics in the quantum setting. These results give some evidence that STATEISOMORPHISM can be thought of as a "quantum GRAPHISOMORPHISM" which may turn out to be intermediate for the quantum equivalent of the class NP.

In the next section we will formally define STATEISOMORPHISM, before exploring the quantum complexity classes that correspond to the classical classes we have covered so far.

2.3 Isomorphisms of strings and states

Our starting point is a classical problem known as STRINGISOMORPHISM: given two bitstrings and a permutation group, does there exist a permutation in the group that maps one string to the other? Babai's quasi-polynomial time algorithm for GRAPHISOMORPHISM [77] is in fact an algorithm for STRINGISOMORPHISM, since it turns out that GRAPHISOMORPHISM is a special case of STRINGISOMORPHISM. We focus on STRINGISOMORPHISM because it is quite obvious how to make a quantum generalisation: instead of considering bitstrings, define the problem in terms of multi-qubit quantum states. Let us now define these ideas rigorously.

2.3.1 Permutations and STRINGISOMORPHISM

Let Ω be a finite set. A bijection $\sigma : \Omega \to \Omega$ is called a *permutation* of the set Ω. The set of all permutations of a finite set Ω forms a group under composition. This group is called the *symmetric group*, and we denote it by $\mathfrak{S}(\Omega)$. For $x \in \Omega$ and $\sigma \in \mathfrak{S}(\Omega)$, we denote the image of x under σ by $\sigma(x)$.
A *string* $\mathfrak{s} : \Omega \to \Sigma$ is an assignment of *letters* from a finite set Σ called an *alphabet* to the elements of a finite *index set* Ω. Let $\mathfrak{s} : \Omega \to \Sigma$ be a string. The letters of $\mathfrak{s}$ are indexed by elements of the index set Ω. The letter corresponding to $i \in \Omega$ is thus denoted by $\mathfrak{s}_i$. Let $\sigma \in \mathfrak{S}(\Omega)$ be a permutation. Then the action of σ on $\mathfrak{s}$ is denoted by $\sigma(\mathfrak{s})$, and is a string such that for all $i \in \Omega$, $\sigma(\mathfrak{s})_i = \mathfrak{s}_{\sigma(i)}$. Often it is useful to deal with permutations of strings indexed by the natural numbers. Hence, we denote the symmetric group $\mathfrak{S}([n])$ by $\mathfrak{S}_n$, where $[n] := \{1, \ldots, n\}$. In what follows we denote the fact that a group G is a subgroup of a group H by $G \leq H$.

Problem 2.3.1 STRINGISOMORPHISM
Input: Finite sets Ω, Σ, a permutation group $G \leq \mathfrak{S}(\Omega)$ specified by a set of generators, and strings $\mathfrak{s}, \mathfrak{t} : \Omega \to \Sigma$.
Output: YES if and only if there exists $\sigma \in G$ such that $\sigma(\mathfrak{s}) = \mathfrak{t}$.

It is clear that STRINGISOMORPHISM is at least as hard as GRAPHISOMORPHISM: there is a polynomial time Karp reduction from GRAPHISOMORPHISM, obtained by "flattening" the adjacency matrices of the graphs in question into bitstrings. The set of string permutations that correspond to graph isomorphisms form a proper subgroup of the full symmetric group (see explicit details in [69]). Now that we have this definition, it is easy to obtain the equivalent problem about quantum states.

2.3.2 Permutations of quantum states and STATEISOMORPHISM

Let $\sigma \in \mathfrak{S}_n$ be a permutation. Then the following is an n qubit unitary operator that "implements" the permutation in terms of the qubit subsystems (recall the Dirac bracket notation outlined in section 1.1)

$$P_\sigma := \sum_{x \in \{0,1\}^*} |\sigma(x)\rangle\langle x|. \tag{2.2}$$

In what follows, let $\mathcal{Q}_{m,n}$ for $m \geq n$ denote the set of all quantum circuits with m input qubits and n output qubits. In particular, $Q_{n,n}$ is the set of all pure state quantum circuits on n qubits. Then, for $m > n$, $\mathcal{Q}_{m,n}$ is the set of all mixed state circuits that can be obtained by discarding the last $m - n$ output qubits of the circuits in $\mathcal{Q}_{m,m}$.
When we specify a circuit with a subscript label, such as $Q_\psi \in \mathcal{Q}_{m,n}$, we do so to easily refer to the state of the output qubits when the circuit is applied to the state $|0^m\rangle$. In particular, when $m = n$ this is the pure state $|\psi\rangle \in \mathbb{C}^{2^n}$, and the mixed state ψ acting on $\mathbb{C}^{2^n}$ otherwise.

Problem 2.3.2 STATEISOMORPHISM (SI)
Input: Efficient descriptions of quantum circuits Q_{ψ_0} and Q_{ψ_1} in $\mathcal{Q}_{n,n}$, a set of permutations $\{\tau_1, \ldots, \tau_k\}$ generating some permutation group $\langle \tau_1, \ldots, \tau_k \rangle =: G \leq \mathfrak{S}_n$, and a function $\epsilon : \mathbb{N} \to [0, 1]$ such that $\epsilon(n) \geq 1/\text{poly}(n)$ for all n.

YES: There exists a permutation $\sigma \in G$ such that $|\langle \psi_1 | P_\sigma | \psi_0 \rangle| = 1$.

NO: For all permutations $\sigma \in G$, $|\langle \psi_1 | P_\sigma | \psi_0 \rangle| \leq \epsilon(n)$.

The astute reader will have noticed that we have defined the above problem in a different way to all the classical problems we have considered already. We have split the 'question' of the problem in terms of a YES case and a NO case. This is what we referred to earlier as a *promise problem*. Quantum complexity classes are often defined in terms of these promise problems, rather than decision problems. Formally, a promise problem is a pair of sets of bitstrings $A_{\text{YES}}, A_{\text{NO}} \subseteq \{0, 1\}^*$ such that $A_{\text{YES}} \cap A_{\text{NO}} = \emptyset$. These sets respectively correspond to the YES and

NO instances of the problem, and so the intersection must be empty or else the definition wouldn't make sense from this perspective. An algorithm is said to decide a promise problem if for any $x \in A_{\text{YES}} \cup A_{\text{NO}}$, it can decide which of the two sets it belongs to. The promise problem framework is useful because bitstrings that are not in A_{YES} or A_{NO} need not be considered by the algorithm. In contrast, in the decision problem framework, it is necessary to make an implicit assumption that the bitstrings received by the algorithm encode a meaningful instance of the problem. For example, implicit in many graph decision problems is that the bitstring instance encodes a graph in some meaningful way. In the promise problem framework we are allowed to make such an assumption explicitly: we don't consider nonsense bitstrings at all, only those that represent graphs.

Let us now define the two variants of SI we also consider: STABILIZERSTATEISOMORPHISM, and MIXEDSTATEISOMORPHISM. To get to the definition of the first problem, we need to define the *stabilizer states*. The Gottesman-Knill theorem [78] states that any quantum circuit made up of CNOT, Hadamard and phase gates along with single qubit measurements can be simulated in polynomial time by a classical algorithm. Such circuits are called *stabilizer circuits*, and any n-qubit quantum state $|\psi\rangle$ such that $|\psi\rangle = Q|0\rangle^{\otimes n}$ for a stabilizer circuit Q is referred to as a *stabilizer state*.

Let $|\psi\rangle$ be an n-qubit state. A unitary U is said to be a *stabilizer* of $|\psi\rangle$ if $U|\psi\rangle = \pm 1|\psi\rangle$. The set of stabilizers of a state $|\psi\rangle$ forms a finite group under composition called the *stabilizer group* of $|\psi\rangle$, denoted $\text{Stab}(|\psi\rangle)$.

The Pauli matrices form a finite group $\mathcal{P}$ called the *single qubit Pauli group* under composition. The n-qubit Pauli group $\mathcal{P}_n$ is the group with elements $\{(\pm 1)U_1 \otimes \cdots \otimes (\pm 1)U_n \ : \ U_j \in \mathcal{P}\} \cup \{(\pm i)U_1 \otimes \cdots \otimes (\pm i)U_n \ : \ U_j \in \mathcal{P}\}$.

It is well known (*c.f.* [94] Theorem 1) that an n-qubit stabilizer state $|\psi\rangle$ is uniquely determined by the finite group $S(|\psi\rangle) := \text{Stab}(|\psi\rangle) \cap \mathcal{P}_n$, of size 2^n. Hence, $|\psi\rangle$ is determined by the $n = \log(2^n)$ elements of $\mathcal{P}_n$ that generate $S(|\psi\rangle)$. These elements each take $2n$ bits to specify the Pauli matrices in the tensor product, and an extra bit to specify the overall ± 1 phase. As we will see later, this fact, along with a result about preparing such a description efficiently, means that given a polynomial number of copies of a stabilizer state $|\psi\rangle$, we can produce an efficient classical description of that state by means of the generators of $S(|\psi\rangle)$.

We can now state the first problem variant.

Problem 2.3.3 STABILIZERSTATEISOMORPHISM (SSI)
Input: Efficient descriptions of quantum circuits Q_{ψ_0} and Q_{ψ_1} in $\mathcal{Q}_{n,n}$ such that $|\psi_0\rangle$ and $|\psi_1\rangle$ are stabilizer states, a set of permutations $\{\tau_1, \dots \tau_k\}$ generating some permutation group $\langle \tau_1, \dots, \tau_k \rangle =: G \leq \mathfrak{S}_n$, and $\epsilon : \mathbb{N} \to [0, 1]$ such that $\epsilon(n) \geq 1/\text{poly}(n)$ for all n.

YES: There exists a permutation $\sigma \in G$ such that $|\langle \psi_1 | P_\sigma | \psi_0 \rangle| = 1$.

NO: For all permutations $\sigma \in G$, $|\langle \psi_1 | P_\sigma | \psi_0 \rangle| \leq \epsilon(n)$.

The last problem is where we consider isomorphism of mixed states. Note that in this definition, we consider quantum circuits in $\mathcal{Q}_{2n,n}$, that is, those that take $2n$ qubits to n qubits. We do this in order to represent n qubit mixed states, which are obtained by an implicit partial trace in the quantum circuit.

Problem 2.3.4 $(\epsilon, 1-\epsilon)$-MIXEDSTATEISOMORPHISM (MSI)
Input: Efficient descriptions of quantum circuits Q_{ρ_0} and Q_{ρ_1} in $\mathcal{Q}_{2n,n}$, a set of permutations $\{\tau_1, \dots, \tau_k\}$ generating some permutation group $\langle \tau_1, \dots, \tau_k \rangle =: G \leq \mathfrak{S}_n$, and $\epsilon : \mathbb{N} \to [0, 1]$.

YES: There exists a permutation $\sigma \in G$ such that $D(P_\sigma \rho_0 P_\sigma^\dagger, \rho_1) \leq \epsilon(n)$.

NO: For all permutations $\sigma \in G$, $D(P_\sigma \rho_0 P_\sigma^\dagger, \rho_1) \geq 1 - \epsilon(n)$.

We also consider the above problems where the permutation group specified is equal to the symmetric group $G = \mathfrak{S}_n$. We denote these problems with the prefix $\mathfrak{S}_n$, for example, $\mathfrak{S}_n$-SI. It is clear that $\text{SSI} \leq_p \text{SI} \leq_p \text{MSI}$.

2.3.3 Quantum complexity theory

In this section we turn our attention to complexity classes that correspond to quantum computation. Since this is the focus of the chapter, we will define the classes with more rigour than we did for the classical classes in the previous section.

A good starting point is to formalise what it means for a problem to be solvable efficiently by a quantum computer. While quantum generalisations of the Turing machine have been considered [35], certainly at the time of writing, the lingua-franca of quantum algorithms is that of circuits made up of unitary gates. Just what does it mean for such a quantum circuit to be efficient? Do we count the number of gates it consists of? Or perhaps consider the depth of the circuit? Remember that the composition of two unitary gates is also unitary, so if we wanted to we could simplify the circuit by collapsing some gates into one. With considerations like this, the notion of what constitutes an "efficient quantum circuit diagram" could become quite subtle.

We can get rid of these issues entirely by considering quantum circuits that are generated by efficient classical Turing machines. Such an idea may seem strange at first because of the use of classical Turing machines. However, such such a definition makes sense from a practical point of view. After all, we'd need to program our quantum hardware to perform the algorithm, so it makes sense that we would need a means of quickly producing a classical description of it. Likewise, it wouldn't make sense if we claimed to have an efficient quantum algorithm that solves a problem, but there was no way of efficiently producing a description of a circuit that implemented it. We refer to a set of quantum circuits $Q_{\mathbb{N}} := \{Q_n \ : \ n \in \mathbb{N}\}$ as a *family* of circuits indexed by the natural numbers. We say that such a family $Q_{\mathbb{N}}$ is *uniformly generated* in polynomial time if there is a Turing machine M and some polynomially bounded function $p : \mathbb{N} \to \mathbb{N}$ such that when any $n \in \mathbb{N}$ is given as input to M it halts after at most $p(n)$ time steps with a description of circuit Q_n on its tape. Note that the description is left undefined, we assume that some reasonable means of specifying quantum circuits is agreed upon beforehand. Such means of specification of course do exist, so we don't care about the details. In this chapter we also use families of circuits indexed by the set of bitstrings, that is, families of the form $\{Q_x \ : \ x \in \{0,1\}^*\}$. In this case, everything is defined in the same way, but the generating Turing machine must halt in time polynomial in the length of the bitstring x given as input.

Now that we have covered the basic notions involved in quantum complexity theory, let us now provide some definitions. We use the definitions in [38, 105] as our guide. First of all, we consider the class BQP (**B**ounded Error **Q**uantum

Polynomial time), which is interpreted as the class of problems that can be solved efficiently by a quantum computer. We say that a circuit Q accepts a quantum state $|\psi\rangle$ if, when we measure the first qubit of $Q|\psi\rangle$ in the computational basis the measurement outcome is 1. Otherwise, we say that the circuit has rejected $|\psi\rangle$.

Definition 2.3.5 (BQP**)** *Let* $\mathrm{A} = (\mathrm{A}_{YES}, \mathrm{A}_{NO})$ *be a promise problem. We say that* A *is in the class* BQP *if there exists a polynomial time generated uniform family of quantum circuits* $\{Q_i \ : \ i \in \mathbb{N}\}$ *such that for all* $x \in \mathrm{A}_{YES} \cup \mathrm{A}_{NO}$

- *if* $x \in A_{YES}$ *then*

$$\Pr[Q_{|x|} \text{ accepts } |x\rangle] \geq 2/3;$$

- *if* $x \in A_{NO}$ *then*

$$\Pr[Q_{|x|} \text{ accepts } |x\rangle] \leq 1/3.$$

Now that we know what it means for a promise problem to be solvable efficiently by a quantum computer, we will look at the notion of verification using a quantum computer. Since the notion of efficient quantum computation is a bounded error class, it makes sense to turn to a quantum extension of Merlin-Arthur games. The next class QMA (**Q**uantum **M**erlin **A**rthur) captures this notion [36].

Definition 2.3.6 (QMA**)** *Let* $\mathrm{A} = (\mathrm{A}_{\mathrm{YES}}, \mathrm{A}_{\mathrm{NO}})$ *be a promise problem. We say that* A *is in* QMA *if there exists a polynomial time generated uniform family of quantum circuits* $\{V_x \ : \ x \in \{0,1\}^*\}$ *and polynomially bounded* $p : \mathbb{N} \to \mathbb{N}$ *such that*

- *for all* $x \in \mathrm{A}_{\mathrm{YES}}$ *there exists* $|\psi\rangle \in \mathbb{C}^{2^{p(|x|)}}$ *such that*

$$\Pr[V_x \text{ accepts } |\psi\rangle] \geq 2/3;$$

- *for all* $x \in \mathrm{A}_{\mathrm{NO}}$ *and for all* $|\psi\rangle \in \mathbb{C}^{2^{p(|x|)}}$*,*

$$\Pr[V_x \text{ accepts } |\psi\rangle] \leq 1/3.$$

Verifying problems using a quantum computer has an interesting subtlety not present in the classical case. In QMA, the verification is performed using quantum certificates: the state $|\psi\rangle$ in the definition above. What would happen if we were to restrict the certificates to classical bitstrings instead? Can Merlin convince Arthur of more facts if he is allowed to send a quantum state, rather than a classical state? The class QCMA (**Q**uantum **C**lassical **M**erlin **A**rthur) captures the notion of using BQP verifiers but with classical certificates [70].

Definition 2.3.7 (QCMA**)** *Let* $\mathrm{A} = (\mathrm{A}_{\mathrm{YES}}, \mathrm{A}_{\mathrm{NO}})$ *be a promise problem. We say that* A *is in* QCMA *if there exists a polynomial time generated uniform family of quantum circuits* $\{V_x \ : \ x \in \{0,1\}^*\}$ *and a polynomially bounded function* $p : \mathbb{N} \to \mathbb{N}$ *such that*

- *for all* $x \in \mathrm{A}_{\mathrm{YES}}$ *there exists* $c \in \{0,1\}^{p(|x|)}$ *such that*

$$\Pr[V_x \textit{ accepts } |c\rangle] \geq 2/3;$$

- *for all* $x \in \mathrm{A}_{\mathrm{NO}}$ *and for all* $c \in \{0,1\}^{p(|x|)}$*,*

$$\Pr[V_x \textit{ accepts } |c\rangle] \leq 1/3.$$

Of course, computational basis states are still quantum states and therefore count as a special case of quantum certificate. Hence, it is possible to prove that QCMA $\subseteq$ QMA. While an oracle separation between the classes was demonstrated by Aaronson and Kuperberg [82] (later revisited and improved upon by Fefferman and Kimmel in [83]), it is still not known if this inclusion is proper.

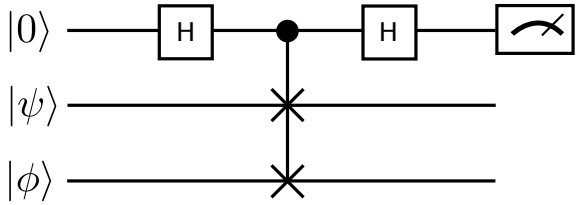


Figure 2.2: The SWAP test circuit.

Let us now show that SI can be verified efficiently using classical certificates. In doing so we make use of a quantum circuit known as the *SWAP test* [76], illus-

trated in Figure 2.2 where the three qubit gate in the middle is a controlled-SWAP gate. This circuit takes as input pure states $|\psi\rangle, |\phi\rangle$ and accepts with probability $(1 + |\langle\psi|\phi\rangle|^2)/2$. Note that the circuit accepts with probability 1 if $|\psi\rangle = e^{i\tau}|\phi\rangle$ for some $\tau \in [-2\pi, 2\pi]$, but only accepts with probability $1/2$ if they are orthogonal.

Theorem 2.3.8 STATEISOMORPHISM $\in$ QCMA.

Proof. In the case of a YES instance, by definition there exists $\sigma \in G$ such that $|\langle\psi_1|P_\sigma|\psi_0\rangle| = 1$. The latter equality can be verified by performing a SWAP test on the states $P_\sigma|\psi_0\rangle$ and $|\psi_1\rangle$, which by definition will accept with probability equal to 1. Since the states $|\psi_0\rangle$ and $|\psi_1\rangle$ are given as efficient classical descriptions of quantum circuits that will prepare them, this verification can be performed in quantum polynomial time. Furthermore, there exists an efficient classical description of the permutation σ in terms of the generators of the group specified in the input, each of which can be described via their permutation matrices. The unitary P_σ can be performed efficiently via a series of SWAP gates given the description of σ.

Determining membership/non-membership of some permutation $\sigma \in \mathfrak{S}_n$ in the permutation group $G \leq \mathfrak{S}_n$ specified by the set of generators $\{\tau_1, \ldots, \tau_k\}$ can be verified in classical polynomial time by utilizing standard techniques from computational group theory. In particular, since we are considering permutation groups we can use the Schreier-Sims algorithm to obtain a base and a strong generating set for G in polynomial time from $\{\tau_1, \ldots, \tau_k\}$. These objects can then be used to efficiently verify membership in G [73, 74, 75].

In the case that the states are not isomorphic, we have by definition that for all permutations $\sigma \in G$, $|\langle\psi_1|P_\sigma|\psi_0\rangle| \leq \epsilon(n)$, which can again be verified by using the SWAP test, which will accept the states with probability at most $1/2 + \epsilon(n)$. Since $\epsilon(n) = 1/\mathsf{poly}(n)$, via standard amplification techniques [38] this acceptance probability can be reduced below $1/3$ by performing the above process at most a polynomial number of times. ■

It is not clear if MIXEDSTATEISOMORPHISM (MSI) is in QCMA, or even in QMA. While the isomorphism σ can still be specified efficiently classically, it is not known if there exists an efficient quantum circuit for testing if two mixed states are close in trace distance. In fact, this problem is known as the STATEDISTINGUISHABILITY problem, and is known to be QSZK-complete [98].

There exists a polynomial time Karp reduction from GRAPHISOMORPHISM to STABILIZERSTATEISOMORPHISM (SSI), indeed it is identical to the reduction from GRAPHISOMORPHISM to STRINGISOMORPHISM. SSI is in turn trivially reducible to the isomorphism problems for pure and mixed states respectively. These problems are therefore at least as hard as GRAPHISOMORPHISM. Interestingly however, there also exists a reduction from GRAPHISOMORPHISM to a restricted form of SI where the permutation group is equal to the full symmetric group $\mathfrak{S}_n$, which we call $\mathfrak{S}_n$-STATEISOMORPHISM). In order to demonstrate this, we require a family of stabilizer states referred to as *graph states* [89]. Let $G = (V, E)$ be an n-vertex graph. For each vertex $v \in V$, define the observable $K^{(v)} := X^{(v)} \prod_{w \in N(v)} Z^{(w)}$ where $N(v)$ is the neighborhood[3] of v, and $X^{(j)}$ denotes the n-qubit operator consisting of Pauli X applied to the j^{th} qubit and identity on the rest (defined analogously for $Z^{(j)}$). The graph state $|G\rangle$ is defined to be the state stabilized by the set $S_G := \{K^{(v)} \ : \ v \in V\}$, that is, $K^{(v)}|G\rangle = |G\rangle$ for all $v \in V$. Since the stabilizers of a graph state $|G\rangle$ are all elements of the $|V|$ qubit Pauli group, graph states are stabilizer states, and the following theorem provides an upper bound on the overlap of non-equal graph states (see also [95], Theorem 8).

Theorem 2.3.9 (Aaronson-Gottesmann [94].) *Let $|\psi\rangle, |\phi\rangle$ be non-orthogonal stabiliser states, and let s be the minimum, taken over all sets of generators $\{P_1, \ldots, P_n\}$ for $S(|\psi\rangle)$ and $\{Q_1, \ldots, Q_n\}$ for $S(|\phi\rangle)$, of the number of values of i where $P_i \neq Q_i$. Then $|\langle\psi|\phi\rangle| = 2^{-s/2}$.*

[3]The neighbourhood of a vertex is the set of vertices it is connected to: $N(v) := \{w \in V(G) \ : \ \{v, w\} \in E(G)\}$

We can now describe the reduction.

Theorem 2.3.10 GRAPHISOMORPHISM $\leq_p \mathfrak{S}_n$-STATEISOMORPHISM.

Proof. Consider two n-vertex graphs G and H. If $G = H$ then clearly $|\langle G|H\rangle|^2 = 1$ since $|G\rangle$ and $|H\rangle$ are the same state up to a global phase. Suppose $G \neq H$. Then necessarily $s > 0$, so by Theorem 2.3.9 we have that $|\langle G|H\rangle|^2 \leq \frac{1}{2}$. Consider a permutation $\sigma \in \mathfrak{S}_n$. Then for each $v \in V$, $K^{(\sigma(v))} = P_\sigma K^{(v)} P_\sigma^T$, so $|\langle \sigma(G)|P_\sigma|G\rangle|^2 = 1$. Explicitly, if $G \cong H$ then there exists a permutation of the vertices σ such that $\sigma(G) = H$ and so $|\langle \sigma(G)|H\rangle|^2 = |\langle G|P_\sigma^T|H\rangle|^2 = 1$. If $G \ncong H$ then for all σ, $\langle G|P_\sigma^T|H\rangle|^2 \leq \frac{1}{2}$.

To complete the reduction we must show that for any graph $G = (V, E)$, a description of a quantum circuit that prepares $|G\rangle$ can be produced efficiently classically. This is trivial, an alternate definition of graph states [89] gives us that $|G\rangle = \Pi_{\{i,j\}\in E} CZ_{ij}|+\rangle^{\otimes|V|}$, where CZ_{ij} is the controlled-Z operator with qubit i as control and j as target. ■

Therefore, the STATEISOMORPHISM problem where no restriction is placed on the permutations is at least as hard as GRAPHISOMORPHISM. Since the states involved in the reduction are stabilizer states, we have actually proved the stronger result that GI $\leq_p \mathfrak{S}_n$-SSI. The complexity of $\mathfrak{S}_n$-STATEISOMORPHISM stands in stark contrast to the complexity of the corresponding classical problem, which is trivially in P: two bitstrings are isomorphic under $\mathfrak{S}_n$ if and only if they have the same Hamming weight, which is easily determined.

As we have described, QMA and QCMA can be thought of as quantum generalisations of the class MA. Quantum generalisations of AM$[k]$ can also be defined, let us do so now. In what follows we define a QAM *verification procedure* to be a tuple (V, m, s) where

$$V = \{V_{x,y} \ : \ x \in \{0,1\}^*, y \in \{0,1\}^{s(|x|)}\}$$

is a uniform family of polynomial time generated quantum circuits, and $m, s : \mathbb{N} \to \mathbb{N}$ are polynomially bounded functions. Each circuit in V acts on $m(|x|)$ qubits sent by Merlin and $k(|x|)$ qubits which correspond to Arthur's workspace. As usual, for all x, y, we say that $V_{x,y}$ accepts (*resp.* rejects) a state $|\psi\rangle \in \mathcal{C}^{2^{m(|x|)}}$

if, upon measuring the first qubit of the state

$$V_{x,y}|\psi\rangle|0\rangle^{\otimes k(|x|)}$$

in the computational basis, the outcome is '1' (*resp.* '0'). The class can be defined like so [105].

Definition 2.3.11 (QAM**)** *A promise problem* $\mathsf{A} = (\mathrm{A}_{\mathrm{YES}}, \mathrm{A}_{\mathrm{NO}})$ *is in* QAM *if there exists a* QAM *verification procedure* (V, m, s) *such that*

- *for all* $x \in \mathrm{A}_{\mathrm{YES}}$*, there exists a collection of* $m(|x|)$*-qubit quantum states* $\{|\psi_y\rangle\}$ *such that*

$$\frac{1}{2^{s(|x|)}} \sum_{y \in \{0,1\}^{s(|x|)}} \Pr[V_{x,y} \text{ accepts } |\psi_y\rangle] \geq 2/3;$$

- *for all* $x \in \mathrm{A}_{\mathrm{NO}}$*, and for all collections of* $m(|x|)$*-qubit quantum states* $\{|\psi_y\rangle\}$*, it holds that*

$$\frac{1}{2^{s(|x|)}} \sum_{y \in \{0,1\}^{s(|x|)}} \Pr[V_{x,y} \text{ accepts } |\psi_y\rangle] \leq 1/3.$$

The class QCAM is defined in the same way but with the states $\{|\psi_y\rangle\}$ restricted to computational basis states.

To complete this section, we will show that STABILIZERSTATENONISOMORPHISM (SSNI) is in QCAM. This follows from the following theorem, which shows that stabilizer states can be described efficiently classically.

Theorem 2.3.12 (Montanaro [92], corollary of Theorem 1) *There exists a quantum algorithm with the following properties:*

- *Given access to $\mathcal{O}(n)$ copies of an n-qubit stabilizer state $|\psi\rangle$, the algorithm outputs a bitstring describing a set of n-qubit Pauli operators $s_1, \ldots, s_n \in \mathcal{P}_n$ such that $\langle s_1, \ldots, s_n \rangle = S(|\psi\rangle)$;*
- *the algorithm halts after $\mathcal{O}(n^3)$ classical time steps;*
- *all collective measurements are performed over at most two copies of the state $|\psi\rangle$;*
- *the algorithm succeeds with probability $1 - 1/\exp(n)$.*

We now prove the result.

Theorem 2.3.13 STABILIZERSTATENONISOMORPHISM *is in* QCAM.

Proof. For a stabilizer state $|\psi\rangle$, denote by $s_\psi^{(1)}, \ldots, s_\psi^{(n)} \in \{\pm I, \pm X, \pm Y, \pm Z\}^n$ the classical strings that describe the stabilizer generators of $|\psi\rangle$ that we can obtain efficiently using the algorithm of Theorem 2.3.12. We denote by s_ψ the length $2n$ string that is obtained by concatenating these stabilizer strings, that is $s_\psi = s_\psi^{(1)} \ldots s_\psi^{(n)}$. Then for any permutation $\sigma \in \mathfrak{S}_n$, we take $\sigma(s_\psi) = s_\psi^{(\sigma(1))}, \ldots, s_\psi^{(\sigma(n))}$. For a permutation group $G \leq \mathfrak{S}_n$, consider the set

$$S_G := \bigcup_{j \in \{0,1\}, \sigma \in G} \left\{ \left(\sigma\left(s_{\psi_j}\right), \pi \right) \ : \ \pi \in G \wedge \pi\left(\sigma\left(s_{\psi_j}\right)\right) = \sigma\left(s_{\psi_j}\right) \right\}.$$

If there exists σ such that $|\langle \psi_1 | P_\sigma | \psi_0 \rangle| = 1$ then $\sigma(s_{\psi_0}) = s_{\psi_1}$, and so in this case $|S_G| = |G|$. If for all $\sigma \in G$ we have that $|\langle \psi_1 | P_\sigma | \psi_0 \rangle| \leq 1 - \epsilon(n)$ then likewise for all $\sigma \in G$, $\sigma(s_{\psi_0}) \neq s_{\psi_1}$ and therefore $|S_G| = 2|G|$. If we can show that membership in S_G can be efficiently verified by Arthur then since all elements of the set are classical bitstrings, we can apply the Goldwasser-Sipser set lower bound protocol [100] to determine isomorphism of the states. To convince Arthur

with high probability that $(\sigma(s_{\psi_j}), \pi) \in S_G$, Merlin sends the permutation σ and the index $j \in \{0, 1\}$. Arthur can then obtain the string s_{ψ_j} with probability greater than $1 - 1/\exp(n)$ using Montanaro's algorithm of Theorem 2.3.12 applied to $U_{\psi_j}|0\rangle$. He can then verify in polynomial time that the string he received is equal to $\sigma(s_{\psi_j})$, that π is an automorphism of $\sigma(s_{\psi_j})$, and that the permutation σ is in the group G. ■

2.3.4 Quantum interactive proofs and zero knowledge

As we have seen previously, an interactive proof system consists of a *verifier* and a *prover*. The computationally unbounded prover attempts to convince the computationally limited verifier that a particular statement is true. A quantum interactive proof system [103] is where the verifier is equipped with a quantum computer, and quantum information can be transferred between verifier and prover. Again, our formal definitions will follow those of Watrous [98, 38].

A *quantum verifier* is a polynomial time computable function V, where for each $x \in \{0, 1\}^*$, $V(x)$ is an efficient classical description of a sequence of quantum circuits $V(x)_1, \ldots, V(x)_{k(|x|)}$. Each circuit in the sequence acts on $v(|x|)$ qubits that make up the verifier's private workspace, and a buffer of $c(|x|)$ communication qubits that both verifier and prover have read/write access to.

A *quantum prover* is a function P where for each $x \in \{0, 1\}^*$, $P(x)$ is a sequence of quantum circuits $P(x)_1, \ldots, P(x)_{l(|x|)}$. Each circuit in the sequence acts on $p(|x|)$ qubits that make up the prover's private workspace, and the $c(|x|)$ communication qubits that are shared with each verifier circuit. Recall that the prover is defined to be computationally unbounded, so no restrictions are placed on the circuits $P(x)$. For instance, unlike the circuits corresponding to the verifier, they don't need to be polynomial time generated. We say that a verifier V and a prover P are *compatible* if all their circuits act on the same number of communication qubits, and if for all $x \in \{0, 1\}^*$, $k(|x|) = \lfloor m(|x|)/2 + 1 \rfloor$ and $l(|x|) = \lfloor m(|x|)/2 + 1/2 \rfloor$, for some $m(|x|)$ which is taken to be the number of messages exchanged between the prover and verifier. We say that (P, V) are a compatible m-message prover-verifier

pair. Given such a pair (P, V), we define the quantum circuit

$$(P(x), V(x)) := \begin{cases} V(x)_1 \cdot P(x)_1 \ldots P(x)_{m(|x|)/2} \cdot V(x)_{m(|x|)/2+1} & \text{if } m(|x|) \text{ is even,} \\ P(x)_1 \cdot V(x)_1 \ldots P(x)_{(m(|x|)+1)/2} \cdot V(x)_{(m(|x|)+1)/2} & \text{if } m(|x|) \text{ is odd.} \end{cases}$$

Let $q(|x|) = p(|x|) + c(|x|) + v(|x|)$. We say that (P, V) accepts an input $x \in \{0,1\}^*$ if the result of measuring the verifier's first workspace qubit of the state

$$(P(x), V(x))|0\rangle^{\otimes q(|x|)}$$

in the computational basis is '1', and that it rejects the input if the measurement result is '0'.

Definition 2.3.14 ($\mathsf{QIP}(k)$**)** *Let* $\mathrm{M} = (\mathrm{M}_{\mathrm{YES}}, \mathrm{M}_{\mathrm{NO}})$ *be a promise problem, and let* $k \in \mathbb{N}$*. Then* $M \in \mathsf{QIP}(k)$ *if and only if there exists a* k*-message verifier* V *such that*

- *if* $x \in \mathrm{M}_{\mathrm{YES}}$ *then*

$$\max_P \left(\Pr[(P, V) \textit{ accepts } x]\right) \geq 2/3,$$

- *if* $x \in \mathrm{M}_{\mathrm{NO}}$ *then*

$$\max_P \left(\Pr[(P, V) \textit{ accepts } x]\right) \leq 1/3,$$

where the maximisation is performed over all compatible k*-message provers* P*. We say that the pair* (P, V) *is an interactive proof system for* M*.*

We can now prove that SNI has a two message quantum interactive proof system. We make use of the following result.

Lemma 2.3.15 (Harrow-Lin-Montanaro [107], Lemma 12) *Given access to a sequence of unitaries $U_1, \ldots, U_n$, along with their inverses $U_1^\dagger, \ldots, U_n^\dagger$ and controlled implementations c-U_1,...,c-U_n, as well as the ability to produce copies of a state $|\psi\rangle$ promised that one of the following cases holds:*

1. *For some i, $U_i|\psi\rangle = |\psi\rangle$;*
2. *For all i, $|\langle\psi|U_i|\psi\rangle| \leq 1 - \delta$.*

Then there exists a quantum algorithm which distinguishes between these cases using $\mathcal{O}(\log n/\delta)$ copies of $|\psi\rangle$, succeeding with probability at least $2/3$.

Theorem 2.3.16 STATENONISOMORPHISM *is in* QIP(2).

Proof. We will prove that the following constitutes a two message quantum interactive proof system for SNI.

1. (*Verifier*) Uniformly at random, select $\sigma \in G$ and $j \in \{0, 1\}$. Send the state $|\Psi\rangle^{\otimes k}$ to the prover, where $k = \mathcal{O}(\log(|G|)/(1 - \epsilon(n)))$ and $|\Psi\rangle = P_\sigma|\psi_j\rangle$.
2. (*Prover*) Determine j from $|\Phi\rangle^{\otimes k}$. Send your best guess $j' \in \{0, 1\}$ to the verifier.
3. (*Verifier*) Accept if and only if $j' = j$.

Obtaining a uniformly random element from G as in step 1 can be achieved efficiently if the verifier is in possession of a base and a strong generating set for G. These can be obtained in polynomial time from any generating set of G by using the Schreier-Sims algorithm [73, 74, 75]. For a permutation $\pi \in G$, we define the $2n$ qubit circuit $U_\pi^{(j)} = \text{SWAP} \cdot (P_{\pi^{-1}} \otimes P_\pi)$, where the SWAP acts like $\text{SWAP}|\psi_0\rangle|\psi_1\rangle = |\psi_1\rangle|\psi_0\rangle$. Now consider the sets of quantum circuits $C_G^{(j)} = \{U_\pi^{(j)} : \pi \in G\}$ for $j \in \{0, 1\}$, each of cardinality $|G|$. Since each circuit in $C_G^{(0)} \cup C_G^{(1)}$ is made up two permutations and a SWAP gate, each of their inverses can easily be obtained. Additionally, the controlled versions of these gates can be implemented via standard techniques.

Consider first the YES case. The $k = \mathcal{O}(\log(|G|)/(1-\epsilon(n)))$ copies of $|\Psi\rangle$ enables the prover to determine j with success probability at least $2/3$ in the following manner.

1. Uniformly at random, select $j' \in \{0,1\}$.
2. Prepare k copies of the state $|\Psi\rangle|\psi_{j'}\rangle$
3. Use the HLM algorithm with the state $|\Psi\rangle|\psi_{j'}\rangle$ and the set of circuits $C_G^{(j')}$ as input. If the algorithm reports case 1 then output j', otherwise output $j' \oplus 1$.

Let us check that the HLM algorithm will work for our purposes. In the case that the prover's guess is correct and $j' = j$, we have that $|\Psi\rangle|\psi_{j'}\rangle = (P_\sigma \otimes I)|\psi_j\rangle|\psi_j\rangle$, and so

$$\begin{aligned} U_\sigma(P_\sigma \otimes I)|\psi_j\rangle|\psi_j\rangle &= \text{SWAP} \cdot (P_{\sigma^{-1}} \otimes P_\sigma) \cdot (P_\sigma \otimes I)|\psi_j\rangle|\psi_j\rangle \\ &= \text{SWAP} \cdot (I \otimes P_\sigma)|\psi_j\rangle|\psi_j\rangle \\ &= |\Psi\rangle|\psi_j\rangle. \end{aligned}$$

This corresponds to case 1 of Lemma 2.3.15. If the prover's guess is incorrect ($j' \neq j$) then for all $\pi \in G$

$$\begin{aligned} |\langle\Psi|\langle\psi_{j'}|U_\pi|\Psi\rangle|\psi_{j'}\rangle| &= |\langle\Psi|\langle\psi_{j'}|\text{SWAP} \cdot (P_{\pi^{-1}} \otimes P_\pi)(P_\sigma \otimes I)|\psi_j\rangle|\psi_{j'}\rangle| \\ &= |\langle\Psi|\langle\psi_{j'}|(P_\pi \otimes P_{\pi^{-1}\cdot\sigma})|\psi_{j'}\rangle|\psi_j\rangle| \\ &\leq |\langle\psi_j|P_\sigma^\dagger P_\pi|\psi_{j'}\rangle| \cdot |\langle\psi_{j'}|P_{\pi^{-1}\cdot\sigma}|\psi_j\rangle| \\ &\leq \epsilon(n)^2, \end{aligned}$$

with the last inequality following from the fact that we are in the YES case: for all $\sigma \in G$, we have that $|\langle\psi_2|P_\sigma|\psi_1\rangle| \leq \epsilon(n)$. This corresponds to case 2 of Lemma 2.3.15. Therefore, the HLM algorithm allows the prover to determine if their guess was correct or not, with success probability at least $2/3$.

Consider now the NO case, where we have that for some $\sigma \in G$, $|\langle\psi_1|P_\sigma|\psi_2\rangle| = 1$. To determine j correctly, a cheating prover must be able to distinguish the mixed states $\rho_j = \frac{1}{|G|}\sum_{\pi\in G}\left(P_\pi|\psi_j\rangle\langle\psi_j|P_\pi^\dagger\right)^{\otimes k}$ correctly for $j \in \{1,2\}$, when given k

copies. However,

$$\begin{aligned}
\|\rho_1 - \rho_2\|_1 &= \frac{1}{|G|} \left\| \sum_{\pi \in G} P_\pi^{\otimes k} \left(|\psi_1\rangle\langle\psi_1|\right)^{\otimes k} P_\pi^{\dagger \otimes k} - \sum_{\pi \in G} P_\pi^{\otimes k} \left(|\psi_2\rangle\langle\psi_2|\right)^{\otimes k} P_\pi^{\dagger \otimes k} \right\|_1 \\
&= \frac{1}{|G|} \left\| \sum_{\pi \in G} P_\pi^{\otimes k} P_\sigma^{\otimes k} \left(|\psi_1\rangle\langle\psi_1|\right)^{\otimes k} P_\sigma^{\dagger \otimes k} P_\pi^{\dagger \otimes k} - \sum_{\pi \in G} P_\pi^{\otimes k} \left(|\psi_2\rangle\langle\psi_2|\right)^{\otimes k} P_\pi^{\dagger \otimes k} \right\|_1 \\
&= \frac{1}{|G|} \left\| \sum_{\pi \in G} P_\pi^{\otimes k} \left(|\psi_2\rangle\langle\psi_2|\right)^{\otimes k} P_\pi^{\dagger \otimes k} - \sum_{\pi \in G} P_\pi^{\otimes k} \left(|\psi_2\rangle\langle\psi_2|\right)^{\otimes k} P_\pi^{\dagger \otimes k} \right\|_1 \\
&= 0,
\end{aligned}$$

so they are indistinguishable. Note that the fact that the prover has been given k copies does not help, as the overlap is still 0. In this case, the probability that the prover can guess j correctly is therefore equal to $1/2$. ■

2.3.5 Statistical zero knowledge

Some interactive proof systems can be performed in such a way that the verifier learns nothing about the problem instance, except for whether it is a YES instance or a NO instance. Intuitively, this captures the idea that in some protocols, the prover doesn't need to withhold any information from the verifier throughout the protocol. It can be shown that the two message interactive proof system for GRAPHNONISOMORPHISM is such a protocol. There are a number of definitions of zero knowledge, but we concern ourselves only with *statistical zero knowledge*. Formally, a protocol is statistical zero knowledge if the entire interaction with the prover can be simulated by the verifier, and this simulated interaction with the prover is statistically indistinguishable from the true interaction. Contrast this with computational zero knowledge, a weaker notion where the simulated interaction need only be computationally indistinguishable from the true interaction. Classical zero knowledge and its application to cryptography is given a comprehensive treatment in [109].

Quantum interactive proof systems can also exhibit the zero knowledge property [98]. We will now define the notion of *quantum statistical zero-knowledge*.

Consider the function

$$\text{view}_{V,P}(x,j) := \text{Tr}_P[(P(x),V(x))_j|0^{q(|x|)}\rangle\langle 0^{q(|x|)}|(P(x),V(x))_j^\dagger],$$

where $(P(x),V(x))_j$ is the circuit obtained from running $(P(x),V(x))$ up to the j^{th} message. For some index set X, we say that a set of density operators $\{\rho_x \ : \ x \in X\}$ is *polynomial time preparable* if there exists a polynomial time uniformly generated family of quantum circuits $\{Q_x \ : \ x \in X\}$, each with a designated set of output qubits, such that for all $x \in X$, the state of the output qubits after running Q_x on a canonical initial state $|0\rangle^{\otimes n}$ is equal to ρ_x. The following class, **Q**uantum **S**tatistical **Z**ero-**K**nowledge (QSZK) captures the notion of having a statistical zero knowledge proof system in the quantum setting [98]. The definition makes use of *negligible functions*. A function $\delta : \mathbb{N} \to \mathbb{R}$ is negligible if for every positive integer c there is some N_c such that for all $x > N_c$, $\delta(x)$ is bounded from above by x^{-c}.

Definition 2.3.17 (QSZK**)** *Let $M = (\text{M}_{\text{YES}}, \text{M}_{\text{NO}})$ be a promise problem, and let $k : \mathbb{N} \to \mathbb{N}$. Then $M \in \text{QSZK}(k)$ if and only if $M \in \text{QIP}(k)$ with quantum interactive proof system (P,V) such that there exists a polynomial time preparable set of density operators $\{\rho_{x,i}\}$ and a negligible function $\delta : \mathbb{N} \to [0,1]$ such that for all $x \in \{0,1\}^*$, if $x \in \text{M}_{\text{YES}}$ then*

$$D(\rho_{x,i}, \text{view}_{P,V}(x,i)) \leq \delta(|x|).$$

The above class assumes an honest verifier, so technically we have defined the class **H**onest **V**erifier **Q**uantum **S**tatistical **Z**ero-**K**nowledge (HVQSZK). However, it is known that such proof systems are equivalent to the more general case [98], so we need only consider honest verifiers.

We can use a standard amplification argument to modify the above protocol so that it has negligible completeness error, which means that it can be made statistical zero knowledge. We prove this now.

Theorem 2.3.18 STATENONISOMORPHISM *is in* QSZK.

Proof. We first show that the protocol above can be modified to have exponentially small completeness error. This allows us to show that the view of the

protocol can be simulated with error scaling as a negligible function of the input length.

First, the verifier sends the prover $k' = O(n \log(|G|)/(1 - \epsilon(n)))$ copies of the state $|\Psi\rangle$. The prover can then use HLM n times to guess j, responding with the value of j that appears in $n/2$ or more of the trials. Let $X_1, \ldots, X_n \in \{\text{'T'}, \text{'F'}\}$ be independent random variables corresponding to whether or not the prover guessed correctly on the i^{th} repetition. By Lemma 2.3.15, we have that $\Pr[X_i = \text{'T'}] \geq 2/3$ and so

$$\begin{aligned}
\Pr\left[\text{Prover guesses correctly}\right] &= 1 - \Pr\left[\frac{1}{n}\sum_{i=1}^{n} X_i < 1/2\right] \\
&= 1 - \Pr\left[\frac{1}{n}\sum_{i=1}^{n} X_i - 2/3 < -1/6\right] \\
&\geq 1 - 2^{-\Omega(n)}
\end{aligned}$$

via the Chernoff bound: for $p, q \in [0, 1]$, we have that

$$\Pr\left[\sum_{i=1}^{n}(X_i - p)/n < -q\right] < e^{-q^2 n/2p(1-p)}.$$

Clearly, sending k' copies of $|\Psi\rangle$ rather than k gives no advantage to the prover, the trace distance between the mixed states ρ_0 and ρ_1 is still 0 in the NO case.

What remains is to show that the protocol is statistical zero knowledge. This is easily obtained, and follows by similar reasoning to the protocol in [98]: the view of the verifier after the first step can be obtained by the simulator by selecting σ and j then preparing k' copies of the state $|\Psi\rangle$. The view of the verifier after the prover's response can be obtained by tracing out the message qubits and supplying the verifier with the value j. Since the completeness error is exponentially small, the trace distance between the simulated view and the actual view is a negligible function. ■

We know that SI belongs in QCMA, since permutations are classical objects and can be described with computational basis state certificates. However, the protocol requires quantum communication: the communicating parties need to

send more than permutations to one another, they must send quantum states. It is not enough to just be able to discuss the *solution* to the problem as in the QCMA protocol. In these more complicated protocols it is necessary to discuss the problem instance itself, which is of course defined in terms of quantum states. It is not clear if a similar protocol exists that uses classical communication only. In the next theorem we show that such a protocol *does* exist for STABILIZERSTATENONISOMORPHISM, since stabilizer states can be described efficiently classically. In order to do so we make use of Theorem 2.3.12 again.

Theorem 2.3.19 STABILIZERSTATENONISOMORPHISM *is in* QCSZK.

Proof. It suffices to show that the state $|\Psi\rangle$ in the protocol above can be communicated to the prover using classical communication only. We know from Theorem 2.3.12 that a classical description can be obtained efficiently from $\mathcal{O}(n)$ copies of $|\Psi\rangle$. These copies can be prepared efficiently, since they are specified in the problem instance by quantum circuits that prepare them. ■

We now prove that MIXEDSTATEISOMORPHISM (MSI) is QSZK-hard. We actually prove the following stronger result.

Theorem 2.3.20 $(\epsilon, 1-\epsilon)$-$\mathfrak{S}_n$-MIXEDSTATEISOMORPHISM *is* QSZK*-hard for all* $\epsilon(n) = 1/\exp(n)$.

We prove this by reduction from the following problem (α, β)-PRODUCTSTATE, which as shown in [91] is QSZK-hard even when $\alpha = \epsilon, \beta = 1 - \epsilon$ and ϵ goes exponentially small in n.

Problem 2.3.21 (α, β)-PRODUCTSTATE
Input: Efficient description of a quantum circuit Q_ρ in $\mathcal{Q}_{0,n}$.

> *YES:* There exists an n-partite product state $\theta_1 \otimes \cdots \otimes \theta_n$ such that $D(\rho, \theta_1 \otimes \cdots \otimes \theta_n) \leq \alpha$
>
> *NO:* For all n-partite product states $\theta_1 \otimes \cdots \otimes \theta_n$, $D(\rho, \theta_1 \otimes \cdots \otimes \theta_n) \geq \beta$.

We make use of the following lemma. For an n-partite mixed state ρ, let ρ_i denote the state of the i^{th} subsystem, obtained by tracing out the other subsystems.

Lemma 2.3.22 (Gutoski et al. [91], Lemma 7.4) *Let ρ be an n qubit state. If there exists a product state $\theta_1 \otimes \cdots \otimes \theta_n$ such that $\|\rho - \theta_1 \otimes \cdots \otimes \theta_n\|_1 \leq \alpha$, then $\|\rho - \rho_1 \otimes \cdots \otimes \rho_n\|_1 \leq (n+1)\alpha$.*

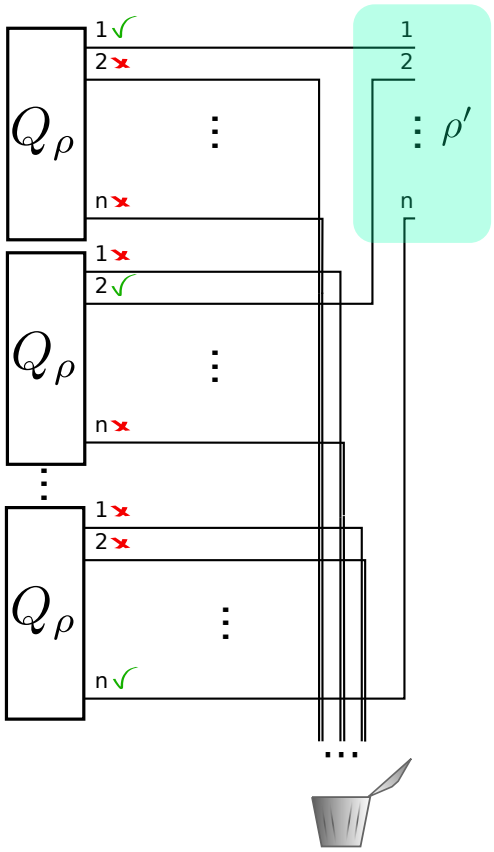


Figure 2.3: Constructing the state $\rho' = \rho_1 \otimes \cdots \otimes \rho_n$ from n copies of the input circuit Q_ρ.

Proof of Theorem 2.3.20. We now must show that every instance of (α, β)-PRODUCTSTATE can be converted to an instance of (α', β')-$\mathfrak{S}_n$-MSI. In particular, consider an instance ρ of (α, β)-PRODUCTSTATE. Our reduction takes this to an instance (ρ, ρ') of $((n+1)\alpha, \beta)$-$\mathfrak{S}_n$-MSI, where $\rho' = \rho_1 \otimes \cdots \otimes \rho_n$ can be prepared in the following way from n copies of the state ρ. Denote these n copies as $\rho^{(1)}, \ldots, \rho^{(n)}$. The i^{th} qubit line of ρ' is the i^{th} qubit line of $\rho^{(i)}$, all unused qubit lines are discarded (illustrated in Figure 2.3).
Let ρ be an n-partite state. If ρ is product then $D(\rho, \rho_1 \otimes \cdots \otimes \rho_n) \leq (n+1)\alpha/2$ and so (ρ, ρ') corresponds to a YES instance of $((n+1)\alpha, \beta)$-$\mathfrak{S}_n$-MSI. If ρ is a

NO instance of (α, β)-PRODUCTSTATE then $D(\rho, \theta) \geq \beta$ for all product states θ. This means that $D(\rho, P_\sigma \rho_1 \otimes \cdots \otimes \rho_n P_\sigma) \geq \beta$ for all $\sigma \in \mathfrak{S}_n$ since all such states are product. ■

In this section we have shown that STATEISOMORPHISM is in QSZK, and so is unlikely to be QCMA-complete unless all problems in QCMA have quantum statistical zero knowledge proof systems. We have also shown that STABILIZER-STATEISOMORPHISM has a quantum statistical zero knowledge proof system that uses classical communication only, and that MIXEDSTATEISOMORPHISM is QSZK-hard.

2.4 Summary

In this chapter we have considered a quantum generalisation of GRAPHISOMORPHISM (GI), the problem of determining if two quantum states can be "rearranged" into one another. We spent time gathering evidence that this problem, STATEISOMORPHISM (SI), exhibits similar properties to GI.

We showed that GI is a special case of SI (Theorem 2.3.10: GI $\leq_p$ SI), and that SI can be verified efficiently by a quantum computer (Theorem 2.3.8: SI $\in$ QCMA). GI is unique in that it is in NP, but its complement GRAPHNONISOMORPHISM can be efficiently verified by an Arthur-Merlin proof system (GI $\in$ coAM). This fact has formed the main motivation for the results we obtained in this chapter about SI. We showed that STATENONISOMORPHISM can be verified by a two message quantum interactive proof system (Theorem 2.3.16: SNI $\in$ QIP(2)), and that this proof system can be made quantum statistical zero knowledge (Theorem 2.3.18: SNI $\in$ QSZK). We were unable to prove that SNI $\in$ QAM. In Chapter 4 we discuss some of the reasons why this may be a difficult result to obtain.

We also explored how the types of states under consideration changes the complexity of determining if two states are isomorphic. We showed that restricting the problem to the stabilizer states considering STABILIZERSTATENONISOMORPHISM (SSNI) makes it easier. We showed that in this case, the aforementioned interactive protocols can be modified to make all communication between parties

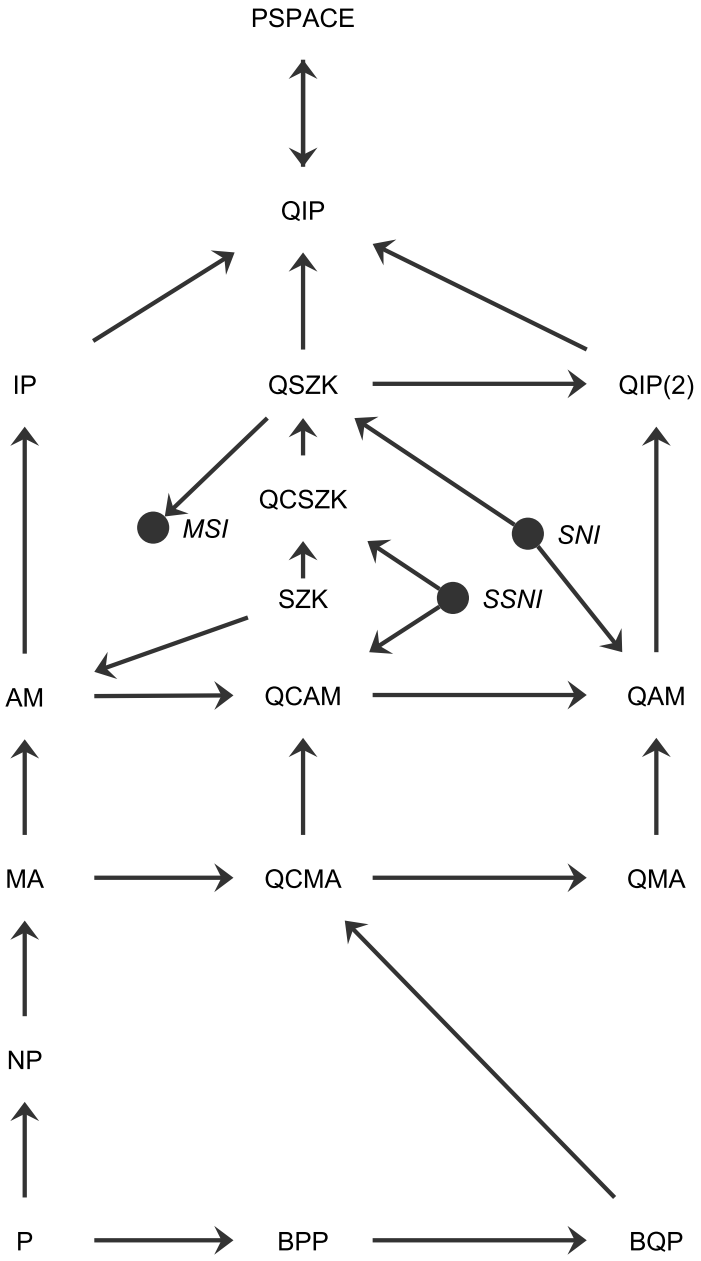


Figure 2.4: A graphical representation of the relationship between the complexity classes considered in this chapter.

classical (Theorem 2.3.19: $\mathrm{SSNI} \in \mathrm{QCSZK}$). The fact that stabilizer states can be "discussed" classically by the interacting parties means that SSNI can be verified with a quantum Arthur-Merlin proof system that uses classical communication

only (Theorem 2.3.13: $\mathsf{SSNI} \in \mathsf{QCAM}$). In contrast, isomorphism problems about mixed states are more challenging, highlighted by Theorem 2.3.20 where we show that MIXEDSTATEISOMORPHISM is QSZK-hard.

In Figure 2.4 we summarise the finding of the chapter. The figure shows the relationship between the myriad complexity classes we have come across, indicating where the quantum isomorphism problems fit. A directed edge from one complexity class to another indicates that the former is a subset of the latter. A directed edge from a problem to a class indicates that the problem is included in the class, while a directed edge from a class to a problem indicates that the problem is hard for the class.

In Chapter 4 we reflect on these results in more detail. Now we move away from complexity theory, focusing on quantum entanglement.

Chapter 3

Grid states

In the standard mathematical formalism of quantum mechanics, we consider the states of multi-particle systems by combining the states of their constituent subsystems with the Kronecker product. Hence, tensor product Hilbert spaces have a fundamental role in describing the behaviour of physical phenomena. Inherent to this tensor product space setting is that some quantum states can be decomposed as a Kronecker product of their constituent subsystems, and others cannot. Remarkably, this mathematical quirk gives us one of physics' most well known phenomena: quantum entanglement. In quantum information processing, it is now clear that entanglement has ubiquitous applications in cryptographic protocols, and computation [110]. See the work of Jozsa and Linden [112] and Vidal [113] for evidence that quantum speed up depends on the presence of a sufficiently large amount of entanglement.

In quantum information theory, the role of entanglement is more clear cut, see for example its role in zero-error quantum information theory [114, 115, 116], and superactivation of channels [117, 118, 119]. Entanglement is an inherently quantum feature embodying non-classical correlations between the constituent parts of a physical system [120].

Determining if a quantum system is entangled or separable is highly non-trivial: no necessary and sufficient entanglement criterion has been found that will efficiently determine this from the density matrix of the system. In fact, the problem of determining if a density matrix is separable is NP-hard [124], which suggests,

rather disappointingly, that no "silver-bullet" entanglement criteria exists that works efficiently in all cases. Due to its central position and myriad technological applications, quantum information theorists and mathematicians have developed a rich set of tools in relation to the problem. The toolbox includes hierarchies of positive semi-definite programs, results from operator theory and functional analysis, ideas from linear algebra and matrix theory, *etc.* [125, 110]. While pure state entanglement is relatively well understood, the study of the mixed state case has revealed numerous subtleties.

In this chapter we come at the mixed state entanglement problem from a new direction, considering a simple class of mixed states of two qudits that we call *grid states*, and attempting to classify which of them are separable and which are entangled. The unique aspect of grid-states is that they can be represented visually with an object we refer to as a *grid-labelled graph*. The chapter will be devoted to unravelling how the structural properties of grid-labelled graphs correspond to separability or entanglement in their underlying grid states. The work is based on two publications, [42] and [43]. Some of the definitions and proofs in this chapter are very similar or equivalent to the way they appear in the publications themselves.

The next section will give a brief tour of some of the highlights of the chapter. Since much of the work is visual, we will attempt to give a pictorial overview, leaving most of the formal definitions for subsequent sections.

3.1 Chapter Overview

A grid state is defined in terms of what could be called "Bell-ish" states, of the form

$$|i,j;k,l\rangle := \frac{1}{\sqrt{2}}\left(|i,j\rangle - |k,l\rangle\right)$$

for $i,j,k,l > 0$[1].
Formally, given a set of Bell-ish states $E = \{|i,j;k,l\rangle\}$, we refer to the state

$$\frac{1}{2|E|}\sum_{|e\rangle\in E}|e\rangle\langle e|$$

as the grid state over E. The set E can be represented by drawing lines between points on a grid, as illustrated in Figure 3.1.

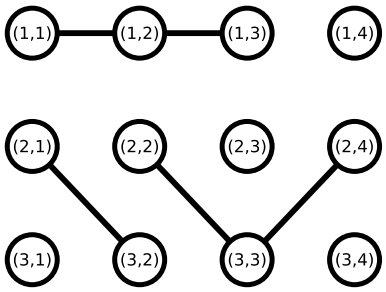


Figure 3.1: We can represent the Bell-ish state $\frac{1}{\sqrt{2}}\left(|i,j\rangle - |k,l\rangle\right)$ by drawing a line between points (i,j) and (k,l) on a suitably sized Cartesian grid. A set of such states $E = \{|1,1;1,2\rangle, |1,2;1,3\rangle, |2,1;3,2\rangle, |2,2;3,3\rangle, |3,3;2,4\rangle\}$ can thus be represented as above.

Since a grid state is parametrised by a set of Bell-ish states, the construction in Figure 3.1 is a kind of graphical representation of a grid state. We refer to this graphical representation as the grid state's *grid-labelled graph*. To summarise then, Bell-ish states are pure states of two qudits, grid states are uniform mixtures of these states which can be represented by grid-labelled graphs. The grid states thus

[1]The states $|i,j\rangle$ are of course computational basis states. Conventionally these are indexed starting from 0, but in this chapter we start from 1. This will make the indices in the equations and the pictorial representations of the states that come later much easier to read.

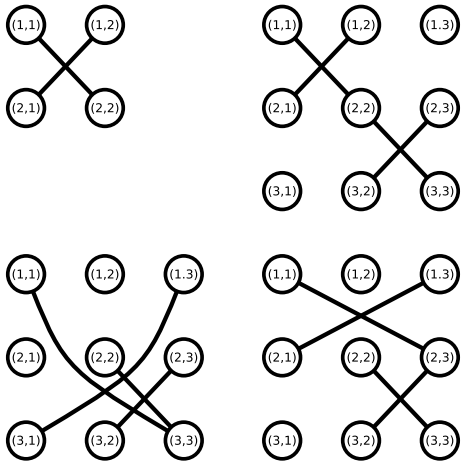


Figure 3.2: Separable graphs, constructed from the 'X' shape graph.

form a discrete family of bipartite mixed states, the entanglement properties of which we will attempt to understand through the properties of their grid-labelled graphs.

The main idea of this chapter is that the pictorial representation offered by the grid-labelled graph allows us to spot generalisable patterns in the states we consider. For example, it is well known that the uniform mixture of the Bell states $\frac{1}{\sqrt{2}}(|1,1\rangle - |2,2\rangle)$ and $\frac{1}{\sqrt{2}}(|1,2\rangle - |2,1\rangle)$ is separable. The grid state that corresponds to such a mixture is in the shape of an 'X' (see Figure 3.2 top-left). In fact, we will use the notion of *local isomorphism* that we develop in Section 3.3.2 to show that such an 'X' graph is always separable, regardless of its position on the grid. We say that pair of grid-labelled graphs are locally isomorphic to one another if they can be obtained by swapping rows and columns, as illustrated in Figure 3.3. We show in this section that for two locally isomorphic grid-labelled graphs, one is separable if and only if the other is.

Returning to our example of the 'X'-shaped graphs from earlier: any such graph is separable simply because the edges can be transported to the top left again by means of row and column swaps. Another fact we prove early on is that the union of two separable graphs is always separable. We therefore know just by examining their structure that all graphs in Figure 3.2 are separable: they can all be built from taking unions of 'X' graphs and performing row or column swaps.

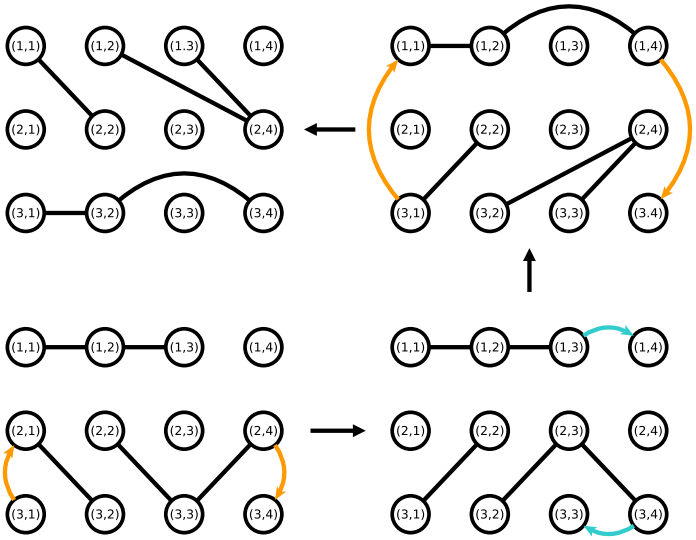


Figure 3.3: Swapping rows and columns to obtain new grid-labelled graphs.

With this reasoning we are able to prove that a graph that has a decomposition into 'X' graphs corresponds to a separable state, regardless of grid dimension. A key theme of the work in this chapter is the question of whether there exists more such "graphical rules" for determining if a grid state is separable.

We also consider applying various entanglement criteria to the grid state density matrices, and show that they can be recast in terms of graph structure. As we survey later, some work has already been done on applying the Peres-Horodecki criterion [126, 127] to Laplacian matrices of graphs, and we show in Section 3.4 that these results can be brought to bear on grid states. In terms of the grid-labelled graph picture, these results involve performing the *partial transpose* operation on the graph (literally, replace each edge $\{(i,j),(k,l)\}$ with the edge $\{(i,l),(k,j)\}$) then checking if any vertex degrees change. If this happens, then the corresponding grid state is entangled.

In Section 3.6 we demonstrate an infinite family of grid states that are entangled, but not detectable by the matrix realignment criterion [151]. The grid state perspective may be useful in generating more such pathological states to push the limits of existing entanglement criteria.

Later in Section 3.7 we apply the *range criterion* [153] to grid states, which

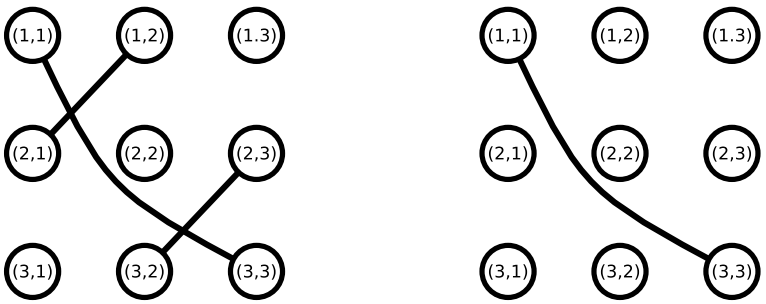


Figure 3.4: An example of surgery. It is possible to use the techniques developed in Section 3.7 that if a product vector $|\alpha\rangle|\beta\rangle$ is in the range of the left hand graph, then it must also be in the range of the right hand graph. We say that the graph on the right is obtainable by surgery on vertex $(2, 2)$ of the left hand graph.

requires us to count the number of product states in the range of the density matrix under consideration. We will see that performing an operation called *surgery* on the grid-labelled graph under consideration will allow us to simplify the graph, but maintain the number of product states in its range (illustrated in Figure 3.4.)

We apply this graphical range criterion to construct two infinite families of *bound entangled* states, some examples are illustrated in Figure 3.5. Bound entangled states are a peculiar class of entangled mixed states for which the entanglement is not *distillable* into pure entanglement: their useful ingredient stays "locked up" inside and is not directly usable quantum information processing tasks. Families of such states have been discussed in the literature up until now, see for example the pyramid and tile states of Bennett et al. [176]. However, the bound entangled states we construct seem to be the only families of such states that are parametrised by discrete objects, in our case graphs. The fact that there exist bound entangled grid states gives credibility to our endeavours, suggesting that grid states are worth considering as a combinatorial "toy-model" of mixed state entanglement.

We have given a brief round-up of what is to come in this chapter. Let us review some existing literature that relates graph theory and quantum entanglement, before getting started with the main technical work.

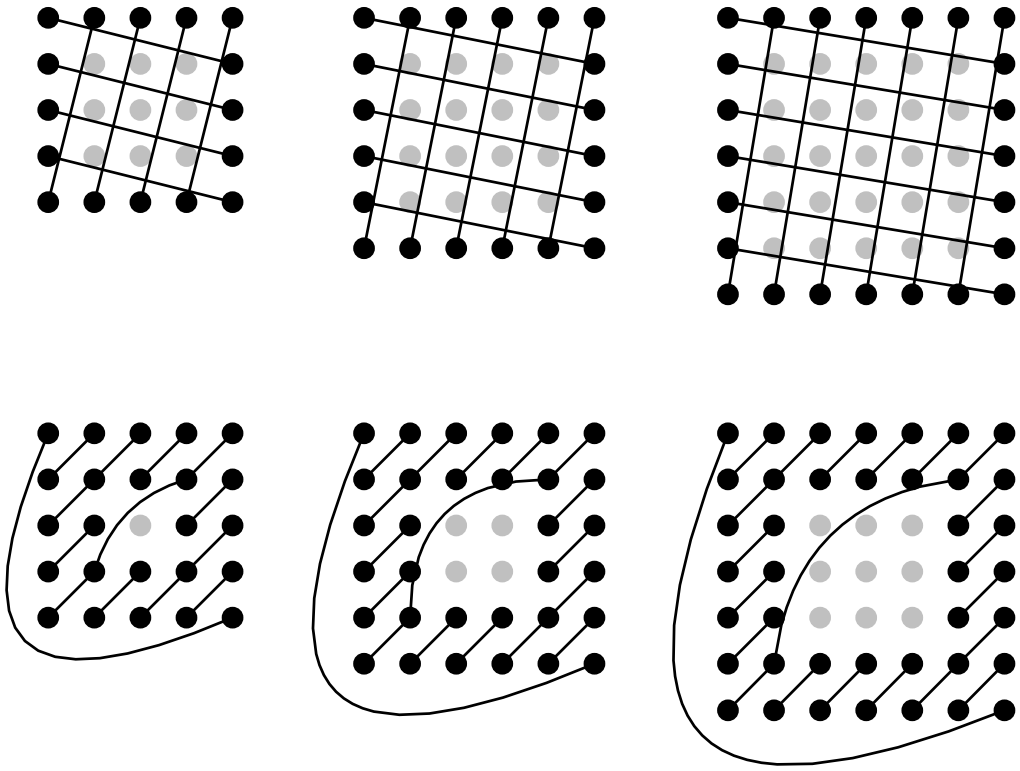


Figure 3.5: Three examples from two different families of grid-labelled graphs that correspond to bound entangled grid states.

3.2 Graphs and quantum entanglement

This chapter revolves around the *combinatorial Laplacian matrix* of a graph (Laplacian for short). The Laplacian combines the degree matrix and adjacency matrix of a graph, and allows mathematicians to apply linear algebraic and matrix theoretic techniques to the study of graphs. For example, the second smallest eigenvalue of the Laplacian is directly related to how connected its graph is (indeed, this eigenvalue is often called the *algebraic connectivity* of a graph [121]). Let $G = (V, E)$ be a simple undirected graph, with a finite vertex set $V = \{1, \ldots, m\}$. The *adjacency matrix* of G is the $m \times m$ matrix $A(G)$ with entries

$$[A(G)]_{ij} := \begin{cases} 1 & \text{if } \{i, j\} \in E \\ 0 & \text{otherwise.} \end{cases}$$

The *degree* of a vertex $v \in V$ is the number of vertices *incident* to v. Formally $d(v) := |\{w \in V \ : \ w \neq v, \{v, w\} \in E\}|$. The degree matrix of G is the diagonal matrix made up of the degrees of the vertices of G: $D(G) := \text{diag}(d(1), \ldots, d(m))$. The Laplacian of G is the $m \times m$ matrix $L(G) := D(G) - A(G)$.
For all graphs G, the corresponding Laplacian matrix $L(G)$ is positive semi-definite, and so the unit trace $\rho(G) := L(G)/\text{Tr}(L(G))$ can be interpreted as the density matrix of a quantum state. In [122], Braunstein, Ghosh and Severini (BGS) consider such density matrices. They explore a number of ideas in this direction relating to entanglement and von Neumann entropy, and the one we focus on is the following.

For which graphs is $\rho(G)$ separable?

In an attempt to answer this, BGS apply the *Peres-Horodecki entanglement criterion*. Roughly stated, Peres-Horodecki tells us that if a density matrix is separable, then its transpose with respect to one of its constituent Hilbert spaces (the *partial transpose*), must also be a density matrix. We can use this fact to determine if a density matrix is entangled: take its partial transpose, and check for negative eigenvalues.
In a follow up paper [123] these ideas are fleshed out further. They show that the partial transpose of a Laplacian manifests itself as a manipulation of the edges of its graph. They then demonstrate that if this edge manipulation causes a change in the vertex degrees of the graph, then the Laplacian density matrix is entangled. This Peres-Horodecki criterion for graphs (the *degree criterion*) then amounts to the following: 1. perform the partial transpose, 2. if a vertex has a different degree, then the state is entangled. In these two works, Braunstein et al. have transformed questions about separability of a quantum mechanical state into a purely combinatorial question. In some cases, they can show that a quantum state is entangled just by manipulating the edges of a graph. This will form the foundation of the ideas developed in this chapter, so let us now examine their work in more detail.
Consider a bipartite density matrix ρ_{AB} with state space $\mathbb{C}^m \otimes \mathbb{C}^n$. The interpretation of the matrix ρ_{AB} as being the density matrix of a bipartite system means

that we consider it as an $m \times m$ block matrix ρ,

$$\rho_{AB} = \begin{pmatrix} M_{11} & M_{12} & \dots & M_{1m} \\ M_{21} & \ddots & & \\ \vdots & & & \\ M_{m1} & & & M_{mm} \end{pmatrix}$$

where each block M_{ij} is $n \times n$. The partial transpose is performed with respect to a subsystem. The partial transpose of ρ_{AB} with respect to A is

$$\rho_{AB}^{\Gamma_A} = \begin{pmatrix} M_{11} & M_{21} & \dots & M_{m1} \\ M_{12} & \ddots & & \\ \vdots & & & \\ M_{1m} & & & M_{mm} \end{pmatrix},$$

and the partial transpose with respect to B is

$$\rho_{AB}^{\Gamma_B} = \begin{pmatrix} M_{11}^T & M_{12}^T & \dots & M_{1m}^T \\ M_{21}^T & \ddots & & \\ \vdots & & & \\ M_{m1}^T & & & M_{mm}^T \end{pmatrix}.$$

We define these notions formally later, but for now it suffices to understand that either we transpose the density matrix with respect to its blocks, or we transpose each block. The Peres-Horodecki criterion states that if either one of the partial transpositions of a bipartite density matrix results in a matrix with a negative eigenvalue (such a matrix will no longer be positive semi-definite and has no interpretation as a state of a quantum system) then the density matrix corresponds to an entangled state.

In order to proceed with characterising the entanglement of a graph $G = (V, E)$, it is necessary to imbue $\rho(G)$ with a state space structure. The way Braunstein et al. achieve this is to assume that $\rho(G)$ is a bipartite density matrix with state space $\mathbb{C}^m \otimes \mathbb{C}^n$, choosing m and n such that $m \times n = |V|$. With this assumption,

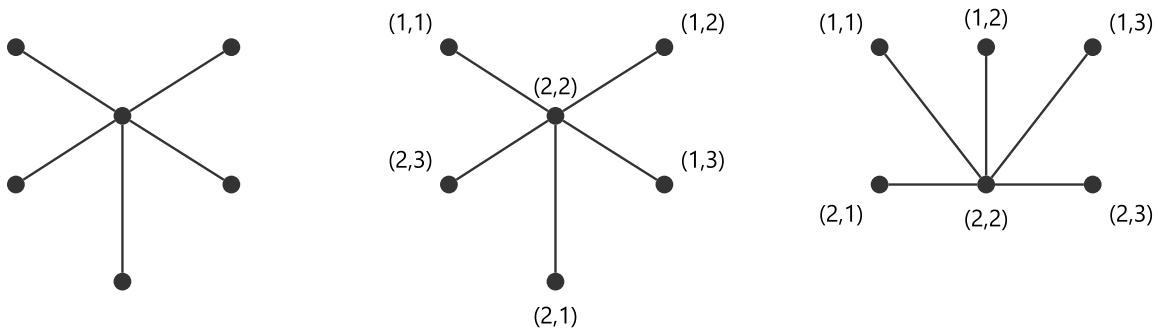


Figure 3.6: From left to right: a graph, a tuple labelling of the graph, redrawn labelled graph.

$\rho(G)$ can be interpreted as an $m \times m$ block matrix with respect to the standard bipartite computational basis $\{|i\rangle \otimes |j\rangle \ : \ (i,j) \in [m] \times [n]\}$, where each block is $n \times n$. To reflect this structure, the vertices of the graph are labelled with tuples $(i,j) \in [m] \times [n]$. See Figure 3.6 for an example of this process: a graph is fixed, suitable subsystem dimensions are decided upon (in the case illustrated in the figure, $m = 2, n = 3$), and the vertices are labelled accordingly.

After the subsystem dimensions are fixed in this way, Braunstein et al. [123] show that the partial transpose of a graph can be interpreted as a manipulation of the edges. Explicitly, they show that the partial transpose with respect to subsystem A (resp. subsystem B) on the Laplacian of the labelled graphs is to map each edge between a pair of labelled vertices (i,j) and (k,l) to an edge between (i,k) and (j,l) (resp. to an edge between (k,j) and (i,l)). They prove that if such a mapping causes one of the vertices to change degree, then $\rho(G)$ is not positive under partial transpose, and so the density matrix is entangled. We illustrate this operation being performed in Figure 3.7. Examining the right hand graph, we can see that most vertices have changed degree. Applying this graphical Peres-Horodecki criterion, we know that the density matrix under consideration is entangled.

This idea of a graphical Peres-Horodecki criterion being used to "diagnose" entanglement in a graph just by manipulating its edges and checking degrees is compelling. Perhaps other entanglement criteria can be cast in this framework, and we can progress towards a combinatorial theory of entanglement, throwing

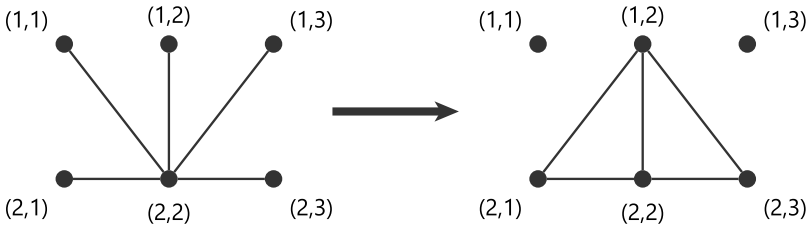


Figure 3.7: The graph obtained by performing the 'graph partial transpose' operation on the far right graph in Figure 3.6.

off the chains of density matrices and eigenvalues. Not so fast! Unfortunately, the question of whether a graph is separable or entangled doesn't make sense. Entanglement is a physical phenomena that is present in systems comprised of two or more quantum particles, so in order to talk about a graph being entangled it is necessary to decide on a subsystem structure. This leads to ambiguity. For example, consider a graph G with 20 vertices. In the BGS framework, the Laplacian $L(G)$ can be interpreted as the density matrix of a state in $\mathbb{C}^{10} \otimes \mathbb{C}^2$, $\mathbb{C}^5 \otimes \mathbb{C}^4$ (or even $\mathbb{C}^5 \otimes \mathbb{C}^2 \otimes \mathbb{C}^2$ if we are open to the possibility of graphs living on a three-dimensional grid). Just which quantum state are we talking about here? The central thesis in these works, that structural graph theory can be used to reason about entanglement in quantum states, doesn't go very far when this subsystem ambiguity is taken into account. Whether a density matrix is entangled or separable is tied up intrinsically in how we divide it into subsystems. Even worse, two isomorphic graphs can correspond to both separable and entangled states respectively. Consider the two labellings of the graph $K_2 \uplus K_1 \uplus K_1$, illustrated in Figure 3.8. The Laplacian of the graph on the left is separable, whereas the Laplacian of the graph on the right is entangled.

Despite these issues, these two papers [123, 122] do contain some exciting ideas. There are several subsequent works that build on their foundations, but inherit the problems we have discussed. The work in this chapter is focused on how to modify the foundations of this theory of "combinatorial entanglement" so that it results in more physically meaningful notions. It will therefore be useful to take a tour of this literature.

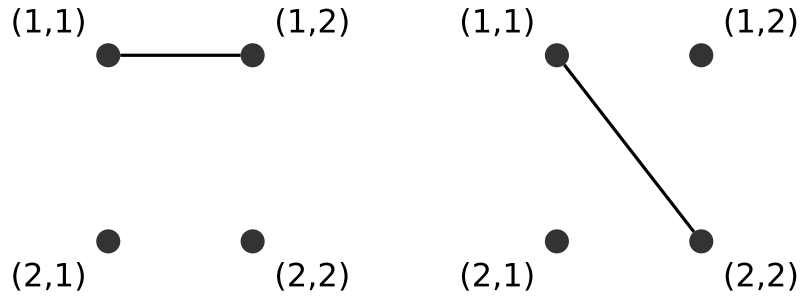


Figure 3.8: These graphs are isomorphic, and yet the graph on the left corresponds to an entangled state and the graph on the right to a separable one.

3.2.1 Subsequent work

A major contributor to the literature that builds on the work of Braunstein et al. is C. W. Wu, who has up to now written a string of single author papers tackling questions in this direction. In [132] they consider what will later come to be known as the degree conjecture: that the graphical Peres-Horodecki criterion (the *degree criterion*) elucidated in [122] and [123] is a necessary and sufficient entanglement criterion for all graph Laplacians. Wu proves that the degree criterion is necessary and sufficient for graph Laplacians considered as density matrices on $\mathbb{C}^2 \otimes \mathbb{C}^q$, for all $q \geq 2$. This stands in remarkable contrast to when we apply the Peres-Horodecki criterion to arbitrary density matrices, for which it is necessary and sufficient for $\mathbb{C}^p \otimes \mathbb{C}^q$ for $p \cdot q \leq 6$ only. Explicitly, there exists density matrices in $\mathbb{C}^p \otimes \mathbb{C}^q$ for $p \cdot q > 6$ that remain positive semi-definite under partial transpose, but are entangled. Such states are referred to as *bound entangled* states, and will be our focus later in the chapter.
In [134], Hildebrand, Mancini and Severini show that the degree conjecture is in fact false. They manage to find a 9×9 Laplacian matrix M that is positive under partial transpose (PPT), but is entangled in $\mathbb{C}^3 \otimes \mathbb{C}^3$. Explicitly, M is the 9×9 matrix

$$M = \begin{pmatrix} I_4 & 0 & -I_4 \\ 0 & 0 & 0 \\ -I_4 & 0 & I_4 \end{pmatrix},$$

which corresponds to the graph illustrated in Figure 3.9. We will come across

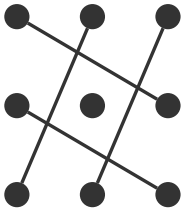


Figure 3.9: The graph that serves as the counter-example to the degree criterion, from [134].

this state again in Section 3.5.

Hildebrand et al. conclude the paper by providing an additional proof that the degree criterion is necessary and sufficient in $\mathbb{C}^2 \otimes \mathbb{C}^q$ for $q \geq 2$. It is worth reiterating here that the labelling problem arises in this work. Despite the fact that Hildebrand et al. have found a bound entangled graph, not all labellings correspond to states that are PPT. In this way, it is not so much that the graph as a structural object corresponds to a bound entangled state, more that for at least one labelling of that structure, the corresponding Laplacian is a bound entangled density matrix.

The case of multi-particle entanglement of Laplacians is considered in a number of papers. In [140], Wu considers separability in the bipartite case before extending analysis to arbitrary multi-particle systems. The labelling problem arises in this setting too: while some graphs are entangled for all vertex labellings, most have the property where they are separable for some labellings and entangled for others. Wu explicitly draws attention to this, even stating in the introduction that the paper considers separability of Laplacians, rather than graphs. In another paper, [135], Wang and Wang consider tripartite separability, and explicitly build a pair of isomorphic graphs G and H such that $\rho(G)$ is separable and $\rho(H)$ is entangled. Wu returns to multi-particle entanglement in [136], this time trying to give more thought towards labellings and separability. In this work, they define three classes of graphs: class S, the graphs with Laplacians that are separable under all vertex labellings; class SE, those graphs with Laplacians separable under some labellings and entangled under the rest; and class E, the graphs with Laplacians that are entangled under all vertex labellings. Wu proves that the class S is made up of the complete graphs K_n, along with $K_{2,2}$ and its complement. Note that

in [140], Wu has already implicitly explored the class E, and has shown certain families of complete bipartite graphs to be multi-particle entangled under all vertex labellings.
In [137], Wu comes back to separability of Laplacians from an enumerative combinatorics perspective, attempting to count the number of separable graphs as a function of the number of vertices. In Appendix A.2 we consider similar questions. In [133], Dutta et al. consider separability in the bipartite setting. They introduce the notions of partially symmetric graphs and degree symmetric graphs. This work is not particularly clear, because the definitions depend on the graphs being laid out on a grid, and yet they consider the traditional definition of a graph which has no fixed embedding on a grid. As we will see later in the chapter, these concepts come out clearly when we take the grid-labelling as part of the graph itself, removing ambiguity. Rahiminia and Amini [138, 144] consider graphs with *entangled edges*. As will become clear later, such edges correspond to diagonal edges on a grid-labelled graph. They show that a graph with one entangled edge is always entangled. Adhikari et al. [139] consider a more general approach for connecting quantum information with graph theory, by considering weighted graphs that are allowed to have self-loops. While they don't consider questions relating to separability, they show that various kinds of quantum states can be parametrised by a matrix they call the signed Laplacian matrix, and attempt to characterise which signed Laplacians correspond to pure and mixed states. Hassan and Joag also consider graphs with real edge weights in [141]. They consider recasting a number of quantum information theoretic concepts in this weighted graph framework, covering von Neumann entropy, separability and even attempt to view adding and removing edges as a quantum operation using the Kraus operator framework. They define a modified form of the graph tensor product. Hassan and Joag return to this framework in [142], where they claim to show that the degree criterion is necessary and sufficient for *pure* multi-particle quantum states. In [143], Xie, Zhao and Wang apply the matrix realignment criterion to graph Laplacians, and use this to prove entanglement results for some explicit families of graphs such as the star graphs, and the nearest point graphs. They also consider the tensor product of Laplacians, showing that the tensor product of graphs is separable. In [145] Zhao and Fan consider a matrix called the

signless Laplacian as a density matrix, and also work with edge weighted, and vertex weighted graphs. In an interesting series of papers, Dutta, Adhikari and Banerjee take the graph Laplacian state framework to new areas: quantifying quantum discord [146, 147], representing local unitaries as graph operations [149], and defining a quantum generalisation of Seidel switching [148] to generate co-spectral density matrices. Finally, in an exercise in spectral graph theory Li, Chen, and Yang [150] consider which edges can be added to a graph via LOCC, by considering the von Neumann entropy of the corresponding density matrices.

3.2.2 Discussion of literature

Attacking the problem of determining if a density matrix is separable or entangled using new mathematical tools is a worthwhile venture: there are certain difficult problems in entanglement theory that might yield with the aid of a new perspective. In particular, classifying the computational complexity of the separability problem might be easier if we were able to recast this problem as a graph theoretic one: there are a huge number of graph theoretic NP-complete problems that may be useful for a reduction. Such a reduction would likely be considerably simpler than the geometric techniques employed in the proof of NP-hardness by Gurvits [124].

The fact that the graph Laplacian is positive semi-definite, and therefore fits into the mathematical framework of quantum information, makes it a good starting point for a "graph theoretic entanglement theory". The fact that there is already an established literature [131, 128] on algebraic and spectral graph theory makes this topic very tempting to work on, as evidenced by the number of papers written in the few years since the original Braunstein et al. papers [122, 123]. As I've touched upon, this work gradually diverges from physically meaningful questions due to what I feel is its weak foundation: assigning too much significance to conventional graph theory. In particular, these papers tend to suffer from one or both of two problems that I call respectively, the labelling problem, and the subsystems problem. Consider a graph G, with normalised Laplacian $\rho(G)$. If we want to consider the normalised Laplacian $\rho(G)$ as the density matrix of a quantum state then as we have shown, we need to fix two things: the labelling

of the vertices, and the segmentation of the state space. If these two things are not fixed beforehand, then we are not talking about a quantum state at all, we are talking about a matrix. As has become clear, it doesn't make sense for us to talk about entanglement if we have not decided where the split between the subsystems is. The existing literature displays what is for me a pathological fixation upon the simple graph G, even after it is shown that the same graph G can correspond to multiple different quantum states, some of which are separable, and others are entangled. Clearly there is no physical meaning behind any of the graph structure if isomorphic graphs with the same structure can have completely different entanglement properties.

The answer I propose is to stop considering conventional graphs as the starting point of the theory. Instead, we will impose that our graphs have vertices that form a two dimensional grid, and are labelled accordingly. Traditional notions of isomorphism are not relevant to these *grid-labelled graphs*, we will only concern ourselves with a restricted form of isomorphism that is physically motivated (see Section 3.3.2). As we will see, using grid-labelled graphs as our foundation means that there is no longer any ambiguity about subsystem structures or basis labellings.

3.3 Preliminaries

The core object in this work is a family of quantum states called *grid states*. Each grid state corresponds to a combinatorial structure called a *grid-labelled graph*. As we shall see, whether or not a grid-state is entangled can sometimes be determined from the structure of its corresponding grid-labelled graph.

The first result we prove is that the density matrix of a grid-state is exactly equal to the normalised Laplacian matrix of its grid-labelled graph. Make no mistake, we are considering the same kind of states as considered in the Laplacian entanglement literature, and this allows us to make use of some results from this literature. However, considering the grid-labelling as "baked-in" to the graph itself means that the questions we consider have a firmer grounding in physical reality. This allows us to go further than the existing literature. An explicit exam-

ple of how this grid-labelling restriction helps us is when we come to consider physically meaningful notions of isomorphism between grid-labelled graphs. It turns out that swapping the vertices in a row or column of a grid-labelled graph corresponds to a local operation on the density matrix, a kind of physical manipulation that preserves separability. We call grid-labelled graphs that can be obtained from one another in this fashion *locally isomorphic*.

3.3.1 Grid-labelled graphs and grid-states

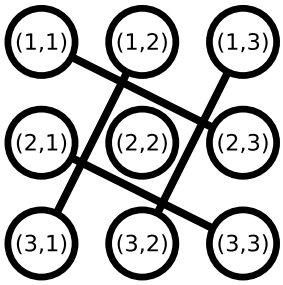


Figure 3.10: A 3×3 grid-labelled graph with four edges.

An $m \times n$ grid-labelled graph is a tuple (V, E), where the elements of $V \subseteq [m] \times [n]$ are called *vertices*, and the elements of set $E \subseteq \binom{V}{2}$ are called *edges*. Grid-labelled graphs can be drawn on the plane, as illustrated in Figure 3.10. Essentially, a grid-labelled graph is just a graph with a grid structure imposed on its vertices. Let $G = (V, E)$ be an $m \times n$ grid-labelled graph. We refer to the quantum state

$$\rho(G) := \frac{1}{|E|} \sum_{\{(i,j),(k,l)\} \in E} \left(\frac{|i,j\rangle - |k,l\rangle}{\sqrt{2}} \right) \left(\frac{\langle i,j| - \langle k,l|}{\sqrt{2}} \right)$$

as the *grid state* of G.

For readability we will now make some definitions. Let $e = (i,j),(k,l) \in E$ be an edge of G. Then its *edge state* is the bipartite pure state $|e\rangle = \frac{|i,j\rangle - |k,l\rangle}{\sqrt{2}}$, and G's grid state is the uniform mixture over its edge states $\rho(G) = \frac{1}{|E|} \sum_{e \in E} |e\rangle\langle e|$.

We can consider the Laplacian of a grid-labelled graph by imposing an ordering on its vertices. In what follows we consider the row-wise ordering $r((i,j)) \mapsto mi + j$. The following lemma allows us to connect grid-states to the literature discussed

in the previous section.

Lemma 3.3.1 *Let $G = (V, E)$ be a grid-labelled graph. Then $\rho(G) = L(G)/2|E|$.*

Proof. Consider G as defined in the statement of the lemma. Then by definition

$$\begin{aligned}
\rho(G) &= \frac{1}{|E|}\sum_{e\in E}|e\rangle\langle e| \\
&= \frac{1}{|E|}\sum_{\{(i,j),(k,l)\}\in E}\left(\frac{|ij\rangle - |kl\rangle}{\sqrt{2}}\right)\left(\frac{\langle ij| - \langle kl|}{\sqrt{2}}\right) \\
&= \frac{1}{2|E|}\left(\sum_{(i,j)\in V}|ij\rangle\langle ij| - \sum_{\{(i,j),(k,l)\}\in E}|ij\rangle\langle kl| + |kl\rangle\langle ij|\right) \\
&= \frac{1}{2|E|}(D(G) - A(G)).
\end{aligned}$$

■

An issue with the work of Braunstein et al. [122, 123] is that separability is not invariant under isomorphism. That is, there are pairs of graphs G_1, G_2 for which $\rho(G_1)$ is separable and $\rho(G_2)$ is entangled, but we have that $G_1 \cong G_2$. In the next section we will consider a form of isomorphism that preserves separability: *local isomorphism*.

3.3.2 Local isomorphism

It is known that if a quantum mechanical system undergoes a unitary evolution that can be decomposed into the Kronecker product of unitaries, each acting on a separate part, then the amount of entanglement between the particles cannot increase [110]. Such unitaries are known as *local*. Let us now attempt to put these notions on a mathematically firmer footing.
Let U be a unitary operator that acts on an n-partite Hilbert space $\mathbb{C}^{d_1}\otimes\cdots\otimes\mathbb{C}^{d_n}$. If $U = U_1\otimes\cdots\otimes U_n$, where U_i acts on $\mathbb{C}^{d_i}$ for $1 \leq i \leq n$ then we say that the unitary U is *local*. Consider the following lemma.

Lemma 3.3.2 *Let ρ be an n-partite density matrix with state space $\mathcal{H}$, and let U be a local unitary on $\mathcal{H}$. Then the state $U\rho U^\dagger$ is separable if and only if ρ is separable.*

Proof. If ρ is separable then by definition there exists $\{\rho_1^{(i)}, \ldots, \rho_n^{(i)}\}$ and $\{p^{(i)}\}$ such that $\rho = \sum_i p^{(i)} \rho_1^{(i)} \otimes \cdots \otimes \rho_n^{(i)}$, and so

$$U\rho U^\dagger = \sum_i p^{(i)} U_1 \rho_1^{(i)} U_1^\dagger \otimes \cdots \otimes U_n \rho_n^{(i)} U_n^\dagger,$$

which is again by definition separable. Conversely, if the state $U\rho U^\dagger$ is separable then by definition it has some decomposition

$$U\rho U^\dagger = \sum_i q^{(i)} U_1 \sigma_1^{(i)} U_1^\dagger \otimes \cdots \otimes U_n \sigma_n^{(i)} U_n^\dagger,$$

and so $\rho = U^\dagger U \rho U^\dagger U$ which implies that $\rho = \sum_i q^{(i)} \sigma_1^{(i)} \otimes \cdots \otimes \sigma_n^{(i)}$ ■

The above lemma shows that local unitaries can be useful when trying to determine if a density matrix is entangled or separable. Suppose we want to know if a particular density matrix is separable. We could transform it into another state by means of a local unitary. If this state is somehow more amenable to analysis then we have made progress: we know that the original state is separable if and only if the easier state is. For example, consider the uniform mixture of two Bell pairs $\rho_{\text{Bell}} = \frac{1}{2}\left(|\Psi^-\rangle\langle\Psi^-| + |\Phi^-\rangle\langle\Phi^-|\right)$. Perhaps surprisingly, this state is separable, but we would require some more sophisticated techniques[2] if we wanted to easily see this just by looking at its density matrix. However, the local unitary $H \otimes H$

[2]We will revisit this in a later section on the Peres-Horodecki criterion.

transforms the density matrix into a state which is more obviously separable:

$$
\begin{aligned}
&(H\otimes H)\frac{1}{2}\left(|\Psi^-\rangle\langle\Psi^-| + |\Phi^-\rangle\langle\Phi^-|\right)(H\otimes H)^\dagger \\
&\qquad = \frac{1}{16}\begin{pmatrix}1&1&1&1\\1&-1&1&-1\\1&1&-1&-1\\1&-1&-1&1\end{pmatrix}\cdot\begin{pmatrix}1&0&0&-1\\0&1&-1&0\\0&-1&1&0\\-1&0&0&1\end{pmatrix}\cdot\begin{pmatrix}1&1&1&1\\1&-1&1&-1\\1&1&-1&-1\\1&-1&-1&1\end{pmatrix} \\
&\qquad = \frac{1}{2}\begin{pmatrix}0&0&0&0\\0&1&0&0\\0&0&1&0\\0&0&0&0\end{pmatrix} \\
&\qquad = \frac{1}{2}\left(|01\rangle\langle 01| + |10\rangle\langle 10|\right) = \frac{1}{2}\left(|0\rangle\langle 0|\otimes|1\rangle\langle 1| + |1\rangle\langle 1|\otimes|0\rangle\langle 0|\right).
\end{aligned}
$$

We can use these ideas to define a form of isomorphism that preserves separability of grid-labelled graphs.

Definition 3.3.3 (Local isomorphism) *A pair of $m\times n$ grid-labelled graphs G and H are said to be* locally isomorphic *if there exists an $m\times m$ permutation matrix P and an $n\times n$ permutation matrix Q such that $\rho(G) = (P\otimes Q)\cdot\rho(G)\cdot(P\otimes Q)$.*

Local isomorphisms have a neat interpretation in terms of permutations of the rows and columns of a grid-labelled graph: the permutation P permutes the row labels, and Q permutes the column labels.
In Figure 3.11 we illustrate a pair of locally isomorphic grid-labelled graphs. In what follows we will use the symbol '$\cong$' to denote local isomorphism. The next lemma follows from the definition of local isomorphism, and as a corollary of Lemma 3.3.2.

Lemma 3.3.4 *Let G and H be a pair of locally isomorphic grid-labelled graphs. Then $\rho(G)$ is separable if and only if $\rho(H)$ is separable.*

Lemma 3.3.4 assures us that local isomorphism is what we are looking for: a method of modifying graphs that preserves separability. From now on, we will

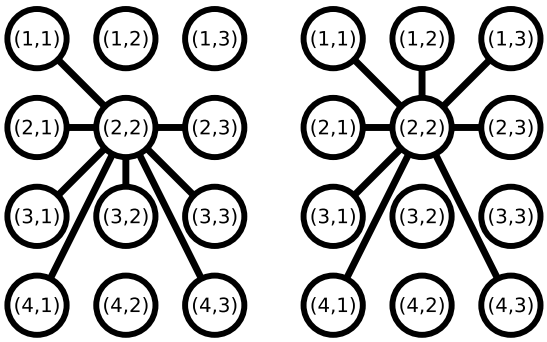


Figure 3.11: A pair of locally isomorphic graphs. The graph on the right can be obtained from the graph on the left by swapping rows 1 and 3.

denote by $\mathcal{S}$ the set of grid-labelled graphs G such that $\rho(G)$ is separable. For example, if G is separable we say that $G \in \mathcal{S}$.
The computational problem of determining if two grid-labelled graphs are locally isomorphic is at least as hard as the "conventional" graph isomorphism problem. Let us define these notions in more formality and prove this assertion. Two conventional graphs G and H are isomorphic if there is an $n \times n$ permutation matrix P such that $P \cdot L(G) \cdot P^T = L(H)$. Recall the following decision problem from Chapter 2.

Problem 3.3.5 GRAPHISOMORPHISM
Input: Graphs G and H.
Question: Are G and H isomorphic?

The corresponding problem about grid-labelled graphs and local isomorphism is defined in the following way.

Problem 3.3.6 LOCALISOMORPHISM
Input: Grid-labelled graphs G and H.
Question: Are G and H locally isomorphic?

We have the following.

Theorem 3.3.7 *There exists a polynomial time Karp reduction from* GRAPHISOMORPHISM *to* LOCALISOMORPHISM.

Proof. Let $G = (V, E)$ be a simple graph G with n vertices. Let G' be the $1 \times n$ grid-labelled graph with edge set $E' = \{\{(1, v), (1, w)\} \ : \ \{v, w\} \in E\}$. It is clear that $L(G) = L(G')$.
Consider G and H as defined in the statement of the theorem. If G is isomorphic to H then there exists a permutation matrix P such that $P \cdot L(G) \cdot P^T = L(H)$. Likewise, $(1 \otimes P)L(G')(1 \otimes P) = L(H')$, where 1 is the unit scalar.
Since the unit scalar is the only 1×1 permutation matrix, the converse follows for the same reason as above. ■
In the next section we will broaden our discussion of local operations, and show that some grid-labelled graphs can be constructed via local unitaries and classical communication, starting with some canonical separable state.

3.3.3 Local operations and classical communication

Intuitively, a unitary acting on a system of multiple parts is local if it can be carried out with no interaction between each of the parts. We can think of the different parts of the evolution being split up and "out-sourced" to a number of separate parties, each carrying out their assigned part of the unitary evolution on the part of the state they have been given. If entanglement requires interaction to generate, how can it be generated in such a setting?
A more general setting involves classical communication: the parties are allowed to communicate via a classical channel. It is possible to prove that they would still not be able to generate entanglement in this setting, referred to in the literature as LOCC (**L**ocal **O**perations and **C**lassical **C**ommunication). We rely on this fact in what follows, and direct the interested reader to [111, 110] for a more in depth discussion of what can be achieved with LOCC.
As we have seen earlier on in the discussion, a grid state is just a uniform mixture of a set of states parametrised by the edges of its grid-labelled graph. In fact, the orientation of the edges determines whether or not that particular component of the mixed state is separable or entangled. For instance, the horizontal and vertical edges always correspond to product states, and so if we have a grid-labelled graph that consists exclusively of such edges then we can expect its corresponding density matrix to be separable as a mixture of product states is

always separable. Let us explore this in more detail with the following definition.

Definition 3.3.8 *An edge $\{(i,j),(k,l)\}$ of a grid-labelled graph is said to be*

- *horizontal if $i = k$ and $j \neq l$,*
- *vertical if $j = l$ and $i \neq k$,*
- *diagonal if $i \neq k$ and $j \neq l$.*

It turns out that if a grid-labelled graph consists exclusively of non-diagonal edges then two parties can construct the corresponding density matrix from the pure state $|1\rangle|1\rangle$ using local unitaries and shared randomness only. We will use a particular family of local unitaries to prove this, which are constructed in the following lemma.

Lemma 3.3.9 *Consider a bipartite Hilbert space $\mathcal{H}_{AB} \cong \mathbb{C}^m \otimes \mathbb{C}^n$ for $2 \leq m, n < \infty$. For all $1 \leq i, k \leq m$ and $1 \leq j, l \leq n$ with $i \neq k$ and $j \neq l$ there exists a unitary matrix $U_{(i,j),(k,l)}$ on $\mathcal{H}_{AB}$ such that*

$$U_{(i,j),(k,l)}|1\rangle_A|1\rangle_B \mapsto \frac{1}{\sqrt{2}}\left(|i\rangle_A|j\rangle_B - |k\rangle_A|l\rangle_B\right).$$

Furthermore, $U_{(i,j),(k,l)}$ is a local unitary if

- *$i = k$ and $j \neq l$, or*
- *if $i \neq k$ and $j = l$.*

Proof. It is trivial to prove that there always exists a unitary matrix $U_{(i,j),(k,l)}$ mapping $|1,1\rangle$ to a state $\frac{1}{\sqrt{2}}(|i,j\rangle - |k,l\rangle)$, so we must only prove the latter half, regarding such a unitary being local if the above conditions are met. Consider the matrix

$$V_h = I_A \otimes (H_B \oplus I),$$

where I_A is the identity matrix of size $a \times a$ acting on $\mathcal{H}_A$,

$$H_B = \frac{1}{\sqrt{2}}\begin{pmatrix} 1 & 1 \\ -1 & 1 \end{pmatrix},$$

and I is an identity matrix of size $(b-2)\times(b-2)$. Here, "$\oplus$" denotes matrix direct sum. The matrix V is real-orthogonal, because H_B is a Hadamard matrix. We have that $V_H|1,1\rangle = \frac{1}{\sqrt{2}}(|1,1\rangle - |1,2\rangle)$. This state can be mapped to any horizontal edge state $\frac{1}{\sqrt{2}}(|i,j\rangle - |i,k\rangle)$ by applying two local isomorphisms, a row swap to bring it from row 1 to row i, followed by two column swaps to bring the incident vertices to the correct horizontal position.
Likewise, consider the matrix

$$V_v = (H_A \oplus I) \otimes I_B.$$

We have that $V_v|1,1\rangle = \frac{1}{\sqrt{2}}(|1,1\rangle - |2,1\rangle)$, which can then be mapped to any vertical edge in the same way, by means of local isomorphisms. ■
With the use of this lemma, we can now prove the result we want, elucidated in the following theorem.

Theorem 3.3.10 *Let G be a grid-labelled graph with m edges. If all edges of G are horizontal or vertical then $\rho(G)$ can be obtained from the state $\rho_0 = |1\rangle_A|1\rangle_B\langle 1|_A\langle 1|_B$ by application of local unitaries and $\log(m)$ bits of classical communication.*

Proof. We prove this statement by demonstrating a protocol by which two parties can turn the density matrix ρ_0 into $\rho(G)$, for grid-labelled graphs G with no diagonal edges. The protocol uses only classical communication and local unitaries acting on ρ_0.
With every edge $\{(i,j),(k,l)\}$, we can associate the unitary matrix $U_{(i,j),(k,l)}$. If all edges are horizontal or vertical, then by Lemma 3.3.9 all m of these matrices are implementable by applying local unitaries on ρ_0. Both Alice and Bob know what each of these matrices are, and which local unitaries they need to perform on their respective Hilbert space.

The protocol is as follows:

1. Alice: uniformly at random, select $\{(i,j),(k,l)\} \in E(G)$ and perform half on $|1\rangle_A$. Send $\{(i,j),(k,l)\}$ to Bob, and erase classical memory of $\{(i,j),(k,l)\}$.

2. Bob: perform other half of $U_{(i,j),(k,l)}$ on $|1\rangle_B$. Erase classical memory of $\{(i,j),(k,l)\}$.

The protocol uses only local unitaries. At the end of the protocol, the final state is equal to

$$\begin{aligned}\rho &= \frac{1}{m} \sum_{\{(i,j),(k,l)\}\in E(G)} U_{(i,j),(k,l)}\rho_0 {U_{(i,j),(k,l)}}^\dagger \\ &= \frac{1}{2m} \sum_{\{(i,j),(k,l)\}\in E(G)} (|i\rangle_A|j\rangle_B - |k\rangle_A|l\rangle_B)(\langle i|_A\langle j|_B - \langle k|_A\langle l|_B) \\ &= \rho(G),\end{aligned}$$

Alice needs to communicate $\log m$ bits to communicate the edge $\{(i,j),(k,l)\}$ to Bob. ■

We can conclude this section with the following corollary, which shows how the adjacency structure of a grid-labelled graph influences physical properties of the corresponding quantum state. Recall that $\mathcal{S}$ is the set of separable graphs.

Corollary 3.3.11 *If a grid-labelled graph G has only horizontal and vertical edges then $G \in \mathcal{S}$.*

Proof. This follows from Theorem 3.3.10 and from Lemma 3.3.4. If LOCC is sufficient to obtain $\rho(G)$ from the separable state ρ_0, provided the edges of G are either horizontal or vertical, then for any such grid-labelled graph, $G \in \mathcal{S}$. ■

In the next section we will talk about the first technique for determining if a grid-labelled graph is separable or entangled: the *degree criterion*.

3.4 The Degree Criterion

We have referred to the Peres-Horodecki criterion a number of times so far. Let us finally give a rigorous overview of this entanglement criterion, starting with a definition of its crucial component, the *partial transpose* of a density matrix.

Definition 3.4.1 (Partial transpose [110]) *Let ρ_{AB} be a density operator acting on a bipartite Hilbert space $\mathcal{H}_{AB}$, and let $\{|i,j\rangle\} \subset \mathcal{H}_{AB}$ denote a fixed product basis of $\mathcal{H}_{AB}$. Then the* partial transpose *with respect to subsystem A of ρ_{AB} is the density operator $\rho_{AB}^{\Gamma_A}$ defined such that*

$$\langle i,j|\rho_{AB}^{\Gamma_A}|k,l\rangle := \langle k,j|\rho_{AB}|i,l\rangle$$

for all i,j,k,l. Respectively, the partial transpose with respect to subsystem B of ρ_{AB} is defined

$$\langle i,j|\rho_{AB}^{\Gamma_B}|k,l\rangle := \langle i,l|\rho_{AB}|k,j\rangle.$$

Armed with this definition, we can state the Peres-Horodecki criterion.

Theorem 3.4.2 (Peres-Horodecki criterion [126, 127]) *Let ρ_{AB} be a bipartite density operator. If ρ_{AB} is separable, then $\rho_{AB}^{\Gamma_A}$ (resp. $\rho_{AB}^{\Gamma_B}$) is positive semi-definite.*

Horodecki *et al.* [127] prove that this criterion is necessary and sufficient for separability for density operators acting on bipartite Hilbert spaces of dimension $d \leq 6$, a result which is summarised in the following theorem.

Theorem 3.4.3 (Horodecki *et al.* [127]) *Let ρ_{AB} be a density operator acting on $\mathbb{C}^2 \otimes \mathbb{C}^2$ or $\mathbb{C}^2 \otimes \mathbb{C}^3$. Then ρ_{AB} is separable if and only if $\rho_{AB}^{\Gamma_A}$ (resp. $\rho_{AB}^{\Gamma_B}$) is positive semi-definite.*

This entanglement criterion is the one applied by Braunstein et al. in [122, 123], and which translates into the graphical entanglement criterion which became known as the degree criterion. The degree criterion can of course be applied to grid-labelled graphs, since the underlying state is a Laplacian matrix. First, let us define the partial transpose of a grid-labelled graph.

Definition 3.4.4 (Partial transpose of a graph) *The* partial transpose *of a grid-labelled graph $G = (V, E)$ is the grid-labelled graph $\Gamma(G) = (V, \Gamma(E))$ where an edge $\{(i, j), (k, l)\} \in \Gamma(E)$ if and only if the edge $\{(i, l), (k, j)\} \in E$.*

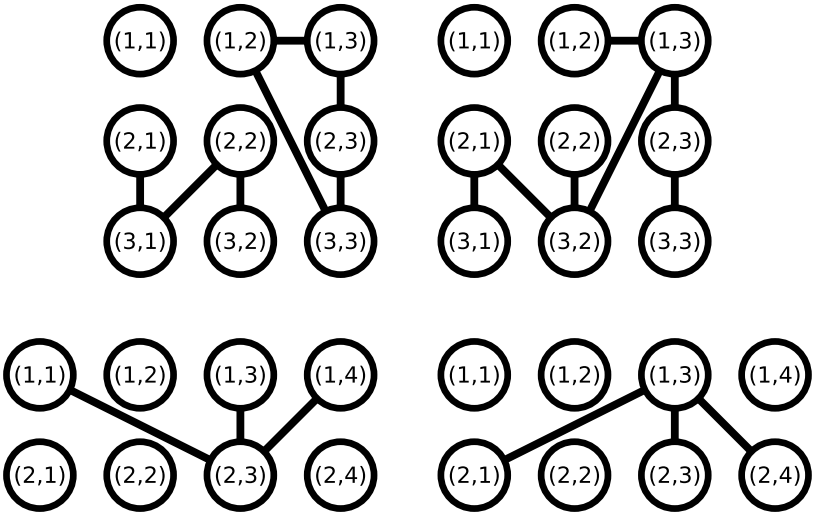


Figure 3.12: Two grid-labelled graphs side by side with their partial transpositions.

Example 3.4.5 In Figure 3.12, we show two grid-labelled graphs alongside their partial transpositions. Note that the partial transpose operation does not affect horizontal and vertical edges, but transposes each diagonal edge.

Let us now state the degree criterion. The proof of the following theorem follows the proof of Theorem 2 in [123]. Recall that $D(G)$ is the degree matrix of G.

Theorem 3.4.6 (Degree criterion) *If $G \in \mathcal{S}$ then $D(G) = D(\Gamma(G))$.*

Proof. Let G be a grid-labelled graph. Let $A(G)$ be the adjacency matrix of G, and let $L(G)$ be the Laplacian matrix of G. For an $a \cdot b \times a \cdot b$ matrix M, let the matrix $\Gamma(M) := M^{\Gamma_A}$ be its partial transpose. Without loss of generality, we consider only partial transpositions with respect to the A subsystem, and do not consider

normalisation factors. Clearly,

$$\begin{aligned}\Gamma(L(G)) &= D(G) - A(\Gamma(G)) \\ &= D(G) - D(\Gamma(G)) + L(\Gamma(G)).\end{aligned}$$

Thus, $\Gamma(L(G)) = L(\Gamma(G)) + \Delta$, where $\Delta := D(G) - D(\Gamma(G))$. The $a \cdot b \times a \cdot b$ matrix Δ is of course real and diagonal, with trace

$$\text{Tr}(\Delta) = \text{Tr}(D(G)) - \text{Tr}(D(\Gamma(G))) = 0.$$

This implies that if Δ is not equal to the zero matrix then there must be one or more negative entries on its diagonal. Let $\delta_i := \Delta_{i,i}$ be one such non-zero entry. Let $|\psi_0\rangle = \sum_{j=1}^{a \cdot b} |j\rangle$ be the all-ones vector, and $|\phi\rangle := k|i\rangle$ for $k \in \mathbb{R}$.
Then for $|\chi\rangle := |\psi_0\rangle + |\phi\rangle$,

$$\begin{aligned}\langle\chi|(L(G) + \Delta)|\chi\rangle &= \langle\chi|L(G)|\chi\rangle + \langle\chi|\Delta|\chi\rangle \\ &= \langle\psi_0|L(G)|\psi_0\rangle + \langle\psi_0|L(G)|\phi\rangle + \langle\phi|L(G)|\psi_0\rangle + \langle\phi|L(G)|\phi\rangle \\ &+ \langle\psi_0|\Delta|\psi_0\rangle + \langle\psi_0|\Delta|\phi\rangle + \langle\phi|\Delta|\psi_0\rangle + \langle\phi|\Delta|\phi\rangle.\end{aligned}$$

Since $|\psi_0\rangle$ is a eigenvector of the (symmetric) matrix $L(G)$ with eigenvalue 0, and that $\langle\psi_0|\Delta|\psi_0\rangle = \text{Tr}(\Delta) = 0$, it follows that

$$\langle\chi|(L(G) + \Delta)|\chi\rangle = \langle\phi|L(G)|\phi\rangle + \langle\psi_0|\Delta|\phi\rangle + \langle\phi|\Delta|\psi_0\rangle + \langle\phi|\Delta|\phi\rangle.$$

Finally, we have
$\langle\phi|L(G)|\phi\rangle = k^2[L(G)]_{i,i} = k^2 d(i)$, where $d(i)$ is the degree of vertex i, $\langle\phi|\Delta|\phi\rangle = \delta_i k^2$, and $\langle\psi_0|\Delta|\phi\rangle = \langle\phi|\Delta|\psi_0\rangle = \delta_i k$. Hence,

$$\langle\chi|(L(G) + \Delta)|\chi\rangle = k^2(d(i) + \delta_i) + 2\delta_i k. \tag{3.1}$$

Since $\delta_i < 0$ by definition, a positive k can be chosen small enough to make Equation 3.1 negative. We therefore have that $L(G) + \Delta \not\geq 0$. If $G \in \mathcal{S}$ then by Theorem 3.4.2, $\Gamma(L(G)) \geq 0$. Since $\Gamma(L(G)) = L(\Gamma(G)) + \Delta$, then this can only be true if $\Delta = 0$, which means that $D(G) = D(\Gamma(G))$. ■

The degree criterion is equivalent to applying the PPT criterion to the Laplacian of the graph. However, the degree criterion turns out to be necessary and sufficient for $2 \times b$ grid-labelled graphs, for arbitrary b. Contrast this result with Theorem 3.4.3, which states that Peres-Horedecki is necessary and sufficient for states in $\mathbb{C}^2 \otimes \mathbb{C}^3$ and $\mathbb{C}^3 \otimes \mathbb{C}^2$.

Theorem 3.4.7 *For a $2 \times b$ grid-labelled graph G with $b \geq 2$,*

$$\Delta(G) = \Delta(\Gamma(G))$$

if and only if $G \in \mathcal{S}$.

To prove this we require some technical lemmas. A *decomposition* of a (conventional) graph G is a set of subgraphs $\{H_1, H_2, ..., H_k\}$ that partition the edges of G: $\bigcup_{i=1}^{k} H_i = G$ and for all $i \neq j$, $E(H_i) \cap E(H_j) = \emptyset$. Notice that isolated vertices do not contribute to a decomposition and so each H_i can always be seen as a *spanning subgraph* (this is a subgraph that contains all the vertices). It will be useful to explicitly define some of these graph theoretic concepts in the context of grid-labelled graphs. Let G and H be grid-labelled graphs. We say that G is a *subgraph* of H, denoted $G \subseteq H$ if for all edges $e \in E(G)$, we have that $e \in E(H)$. On the other hand, if G and H have no edges in common $E(G) \cap E(H) = \emptyset$ then we say that they are *edge disjoint*, and the union of G and H is the grid-labelled graph $G \cup H$ with edge set $E(G) \cup E(H)$. A set of edge disjoint graphs $G_1, \ldots, G_n \subseteq G$ is called a *decomposition* of G if $\bigcup_{i=1}^{n} G_i = G$.
Lemma 3.4.8 is easy to prove, but valuable since it allows us to break up a grid-labelled graph into its **H**orizontal, **V**ertical and **D**iagonal components, which we refer to as its *HVD decomposition*.

Lemma 3.4.8 (HVD decomposition) *Given a grid-labelled graph G with m edges, its density matrix can be written in the form*

$$\rho(G) = \frac{|E(H)|}{m}\rho(H) + \frac{|E(V)|}{m}\rho(V) + \frac{|E(D)|}{m}\rho(D),$$

where the graphs H, V, D contain all of the horizontal, vertical, and diagonal edges of G respectively. These graphs form a unique decomposition of G, which we will call the HVD decomposition.

Proof. For any grid-labelled graph G with m edges,

$$\begin{aligned}
\rho(G) &= \frac{1}{2m}L(G) \\
&= \frac{1}{2m}(L(H) + L(V) + L(D)) \\
&= \frac{1}{2m}(2\,|E(H)|\,\rho(H) + 2\,|E(V)|\,\rho(V) + 2\,|E(D)|\,\rho(D)) \\
&= \frac{|E(H)|}{m}\rho(H) + \frac{|E(V)|}{m}\rho(V) + \frac{|E(D)|}{m}\rho(D).
\end{aligned}$$

■

Lemma 3.4.9 *Let G be a grid-labelled graph. Let H, V and D be the components of the HVD decomposition of G. Then, G satisfies the degree criterion if and only if D satisfies the degree criterion.*

Proof. By definition H and V contain only horizontal and vertical edges. These edges remain invariant under the partial transpose operation, and hence can be disregarded when considering the degree criterion. ■

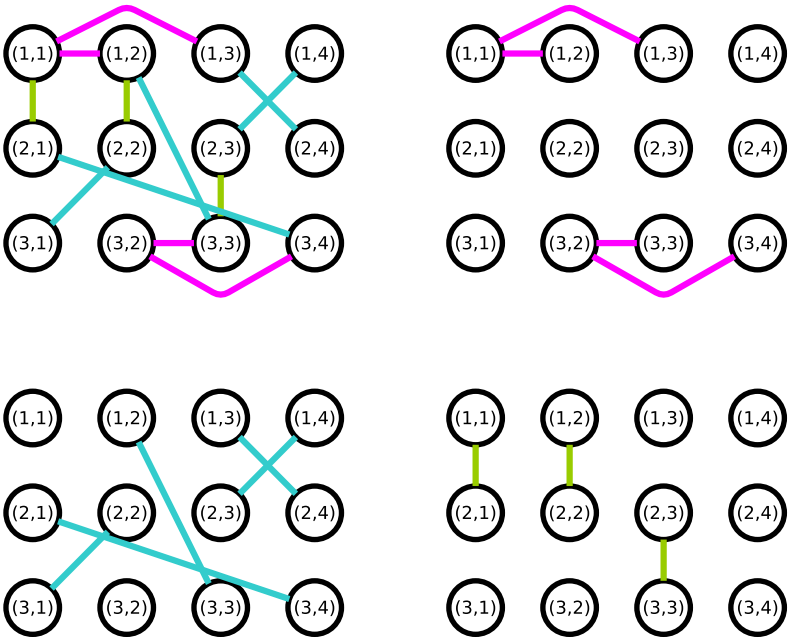


Figure 3.13: Moving clockwise from top-left, we illustrate a grid-labelled graph and its **H**orizontal, **V**ertical and **D**iagonal components, constituting its HVD decomposition.

In Figure 3.13 we illustrate a graph with its HVD decomposition. The last component needed to prove Theorem 3.4.7 is the following theorem of Ando [152], which we state without proof.

Theorem 3.4.10 (Ando [152]: Theorem 4.9) *Any positive semi-definite matrix of the form*

$$\begin{pmatrix} C & A \\ A^\dagger & C \end{pmatrix},$$

where A and C are $b \times b$ complex matrices, is separable in $\mathbb{C}^2 \otimes \mathbb{C}^b$.

Proof of Theorem 3.4.7. We need to show that if we have some $2 \times b$ grid-labelled graph G that satisfies the degree criterion then $G \in \mathcal{S}$. We know from Lemma 3.4.9 that G satisfies the degree criterion if and only if D satisfies the degree criterion, where D is from the HVD decomposition of G and contains all diagonal edges of G. Hence, we need to show that if the degree criterion holds

then $D \in \mathcal{S}$. Up to normalisation, the structure of this density matrix is

$$\rho(D) = \begin{pmatrix} \Delta^{(1)} & A \\ A^T & \Delta^{(2)} \end{pmatrix},$$

where $\Delta^{(1)}$ and $\Delta^{(2)}$ are diagonal matrices encoding the degrees of the vertices in rows 1 and 2 respectively, $\Delta^{(i)}_{j,j} = d((i,j))$, and the matrix A encodes the diagonal edges of G. This can be seen from the fact that the adjacency matrix of any grid-labelled graph G with diagonal edges *only* is a 2×2 symmetric block matrix with block diagonals equal to the zero matrix: diagonal edges are incident to vertices in different rows by definition. Performing the partial transpose operation on the edges of D yields a graph with density matrix

$$\rho(\Gamma(D)) = \begin{pmatrix} \Delta^{(2)} & A^T \\ A & \Delta^{(1)} \end{pmatrix}.$$

By assumption, the degree criterion holds, so $\text{diag}(\rho(D)) = \text{diag}(\rho(\Gamma(D)))$, and we have that $\Delta^{(1)} = \Delta^{(2)}$. Since density matrices are by definition positive semi-definite, we know from Theorem 3.4.10 that $\rho(D)$ is separable. ■

In proving this theorem we have uncovered the following corollary, which comes from the requirement that the matrices $\Delta^{(1)}$ and $\Delta^{(2)}$ are equal for $2 \times b$ grid-labelled graphs that satisfy the degree criterion.

Corollary 3.4.11 *Let G be a $2 \times b$ grid-labelled graph. Then $G \in \mathcal{S}$ if and only if $d((1,j)) = d((2,j))$ for $1 \leq j \leq b$ after discarding all horizontal and vertical edges of G.*

That is, we can always check if a $2 \times b$ grid-labelled graph is separable or entangled by throwing out horizontal and vertical edges before checking each of the "column-pairs" of vertices: if there is a degree mismatch, then the state is entangled.

It will be useful for the work we do later to introduce a generalisation of local isomorphism, *LE-isomorphism*. The idea is that adding extra empty rows or columns of vertices to a grid-labelled graph does not affect separability of its grid state.

3.4.1 Extensions and LE-isomorphism

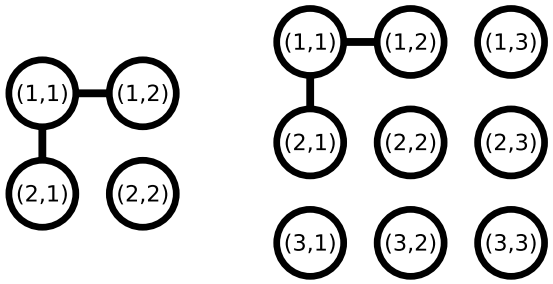


Figure 3.14: The grid-labelled graph on the right is an extension of that on the left.

It seems intuitively reasonable that if we take some grid-labelled graph and extend its grid without changing the edges, then the resulting state should have the same entanglement properties. Adding isolated vertices won't have any effect on the underlying state, since we are just thinking of the same state living in a higher dimensional Hilbert space. This idea is formalised in the following definition, that of an *extension*.

Definition 3.4.12 (Grid-labelled graph extension) *Let G be an $a\times b$ grid-labelled graph, and let H be a $c \times d$ grid-labelled graph such that $c \geq a$ and $d \geq b$. We say that H is an* extension *of G if H has the same edge set as G.*

Example 3.4.13 In Figure 3.14, the graph on the right is an extension of the graph on the left, because it can be obtained by increasing the dimensions of the grid.

In the same way, if two graphs turn out to be locally isomorphic to one another if they are placed on a grid the same size, then it seems likely that they should have the same entanglement properties. Such graphs we call *LE-isomorphic* (**L**ocal+**E**xtendible+isomorphic).

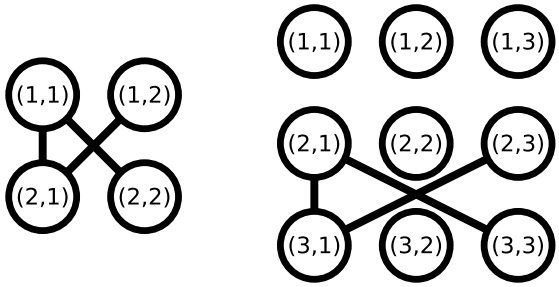


Figure 3.15: Two grid-labelled graphs that are LE-isomorphic.

Definition 3.4.14 (LE-isomorphism) *Grid-labelled graphs G and H are said to be LE-isomorphic if at least one of the following conditions is true*

- *G has an extension G' such that $G' \cong H$,*
- *H has an extension H' such that $H' \cong G$.*

Note that from the way we have defined it, LE-isomorphism is reflexive. When two grid-labelled graphs are LE-isomorphic we denote this with the '$\simeq$' symbol.

Example 3.4.15 The two grid-labelled graphs in Figure 3.15 are LE-isomorphic. The graph on the right can be obtained from the graph on the left by increasing the grid size to 3×3 and swapping columns 2 and 3, then swapping rows 2 and 3 before swapping rows 1 and 2.

Proposition 3.4.16 *If two grid-labelled graphs are locally isomorphic then they are LE-isomorphic. The converse is not necessarily true.*

Proof. If two grid-labelled graphs G and H are locally isomorphic, then their grid dimensions are identical. Trivially, the grid-labelled graphs are extensions of one another, and are hence LE-isomorphic.
The converse is not true because $G \simeq H$ does not imply that their grid dimensions are equal. This property is required by the definition of local isomorphism. ■

Proposition 3.4.17 *Let G, H be two grid-labelled graphs with $G \simeq H$. Then $G \in \mathcal{S}$ if and only if $H \in \mathcal{S}$.*

Proof. If G and H have identical grid dimensions then they are locally isomorphic and the result follows from Lemma 3.3.4. It suffices to prove that if a graph A is an extension of a graph B then $A \in \mathcal{S}$ if and only if $B \in \mathcal{S}$. This follows trivially from the fact that $\rho(A)$ and $\rho(B)$ are the same matrix, but $\rho(A)$ acts on a higher dimensional Hilbert space than $\rho(B)$ ■
In the next section we will consider decompositions of grid-labelled graphs in greater depth.

3.4.2 Decompositions

It makes sense that a mixture of two separable states will not be entangled. We can express this in terms of decompositions of grid-labelled graphs, and show that if a grid-labelled graph can be decomposed into separable graphs then it too is separable.

Theorem 3.4.18 (Separable Decompositions) *If there exists a decomposition X of G such that for all $H \in X$ we have $H \in \mathcal{S}$, then $G \in \mathcal{S}$.*

Proof. We know that

$$\begin{aligned}\rho(G) &= \frac{1}{2|E(G)|}L(G) \\ &= \frac{1}{2|E(G)|}\sum_{H\in X} L(H) \\ &= \frac{1}{2|E(G)|}\sum_{H\in X} 2|E(H)|\cdot\rho(H) \\ &= \sum_{H\in X}\frac{|E(H)|}{|E(G)|}\rho(H).\end{aligned}$$

If for all $H \in X$, $H \in \mathcal{S}$ then a convex combination of the form given by Equation (1.1) can be formed by setting $p_i = |E(H)|/|E(G)|$. Therefore, $G \in \mathcal{S}$. ■

The *partially-symmetric* graphs are discussed in [133]. A similar notion can be applied to the grid-labelled context, which we call "pair-symmetry", because diagonal edges are grouped together into symmetric pairs.

Definition 3.4.19 (Pair-symmetric grid-labelled graphs) *A grid-labelled graph is said to be* pair-symmetric *if each of its diagonal edges* $\{(i,j),(k,l)\}$ *have a* counterpart *edge* $\{(k,j),(i,l)\}$. *An edge and its counterpart are referred to as a* counterpart pair.

All pair-symmetric grid-labelled graphs are separable, which is quite obvious when we consider that such graphs are just mixtures of graphs that are locally isomorphic to the 'cross' graph illustrated in Figure 3.16. This cross graph is the uniform mixture of Bell pairs that we have come across a number of times so far in this chapter.

Proposition 3.4.20 *Every pair-symmetric grid-labelled graph is separable.*

Proof. Let G be a pair-symmetric grid-labelled graph with k counterpart pairs. Let H_i for $1 \leq i \leq k$ be the subgraph of G containing only the edges of the i^{th} counterpart pair. Let H_{k+1} be the subgraph of G containing the remaining edges of G, that is, those edges not part of the list of k counterpart pairs. Clearly, the

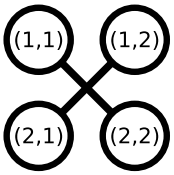


Figure 3.16: A 2×2 graph with a cross.

set $X = \{H_1, H_2, ..., H_{k+1}\}$ forms a decomposition of G. To prove the proposition it suffices to show that for each $H \in X, H \in \mathcal{S}$. It is obvious that for $1 \leq i \leq k$, $H_i \simeq S$, where S is shown in Figure 3.16. This graph is invariant under partial transpose, and since it is 2×2 we know it is separable, since the degree criterion is necessary and sufficient in these grid dimensions. Hence, for all $1 \leq i \leq k, S \simeq H_i$, and so $H_i \in \mathcal{S}$.

The last component of the decomposition X, H_{k+1}, has only horizontal and vertical edges (since by definition, all diagonal edges in a pair-symmetric graph are

involved in a counterpart pair). Hence from Corollary 3.3.11, $H_{k+1} \in \mathcal{S}$, and so X is a separable decomposition of G. The proposition thus follows via Theorem 3.4.18. ■

We will now explore another family of grid-labelled graphs, which we call *stratified*. It turns out that satisfaction of the degree criterion is a necessary and sufficient condition for separability of stratified grid-labelled graphs. Furthermore, such grid-labelled graphs exist in all bipartite dimensions.

Definition 3.4.21 (Stratified grid-labelled graphs) *A grid-labelled graph is called* row *(resp.* column*)* stratified *if for all of its diagonal edges* $\{(i,j),(k,l)\}$*,* $|i-k|=1$ *(resp.* $|j-l|=1$*).*

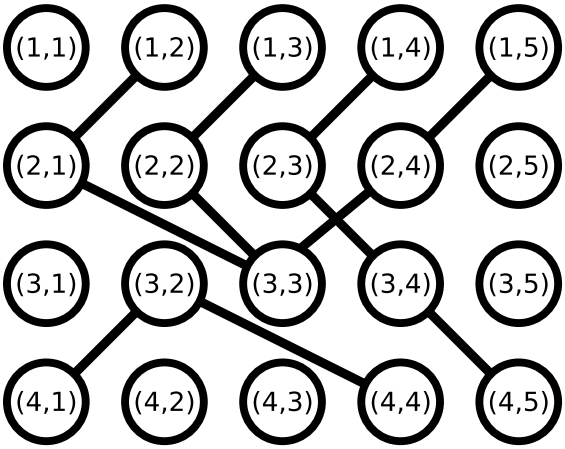


Figure 3.17: A row stratified graph

Example 3.4.22 Figure 3.17 illustrates a row stratified grid-labelled graph. The endpoints of all diagonal edges are restricted to nearest neighbour rows of vertices, hence the name "stratified".

The following lemma is obvious.

Lemma 3.4.23 (Strata decomposition) *Let* G *be a row (resp. column) stratified grid-labelled graph. Let* D *be the diagonal part of the HVD decomposition of* G*. Then,* D *has a decomposition* $S=\{S_i\}$ *for* $1 \leq i < a$*, where* S_i *is the subgraph of* D *containing the edges* $\{(p,q),(r,s)\} \in E(D)$ *with* $p=i$ *and* $r=i+1$ *(resp.* $q=i$ *and* $s=i+1$*). We call* S *the* strata decomposition *of* G*.*

This can be used to prove the following.

Lemma 3.4.24 *Let G be a stratified grid-labelled graph. If G satisfies the degree criterion then each element of its strata decomposition satisfies the degree criterion.*

Proof. We prove the lemma for row stratified graphs. The same argument holds for column stratified graphs if all references to "rows" are replaced by "columns". In what follows, we refer to graphs that satisfy the degree criterion as "DC", and those that do not as "non-DC".
By definition, the elements of a strata decomposition of a graph G are LE-isomorphic to $2 \times b$ graphs – that is, we are only interested in the vertices in the 2 "occupied rows" of the strata. By Corollary 3.4.11 if a $2 \times b$ grid-labelled graph is non-DC then the degrees of the vertices in the upper row do not match those of the lower row. Therefore, if a single element S_i of the strata decomposition of G (in this case, corresponding to the subgraph induced by vertices in row i and $i+1$ of G) is non-DC, then the degree of at least 1 vertex in these rows will change after partial transpose. If this happens, then of course G is non-DC.
To finish the proof of this lemma, we must show that if two or more elements of the strata decomposition of G are non-DC then G is non-DC. This must be considered because conceivably the degree changes in 2 or more non-DC strata could cancel out in some way, leaving $D(G)$ equal to $D(\Gamma(G))$.
The only way two strata degree changes could cancel out would be if two non-DC strata shared a row of G. That is, if the strata S_i and S_{i+1} were non-DC. However, if this were the case then rows i and $i+2$ would have a partial transpose degree change. By the same argument, no additional non-DC strata can be selected either side of these to cancel out the degree changes. The lemma follows. ■

Theorem 3.4.25 *A stratified grid-labelled graph G is separable if and only if it satisfies the degree criterion.*

Proof. The degree criterion has been proved to be necessary for separability, hence we must only prove sufficiency.
From Lemma 3.4.23, for any stratified grid-labelled graph G there exists a decomposition $S = \{S_i\}$ for $1 \leq i < b$. Each of the $a \times b$ graphs S_i are LE-isomorphic to a $2 \times b$ graph S_i': we can discard each isolated vertex. If the degree criterion

holds for G, then by Lemma 3.4.24 it holds for each S_i'. Hence, by sufficiency of the degree criterion for separability for grid-labelled graphs with $a = 2$, each S_i' is separable, and the strata decomposition S is a separable decomposition of G. Separability of G follows from Theorem 3.4.18. ■

Note that the work of Wu [132] explores separability of matrices with line-sum symmetric blocks, and proves results of the same flavour to what we do in this section. Perhaps there is a link between grid-labelled graphs that are locally isomorphic to stratified grid-labelled graphs and combinatorial Laplacian matrices with line-sum symmetric blocks.

3.5 3×3 graphs that satisfy the degree criterion

We would like to understand the structural properties of the grid-labelled graphs that correspond to entangled states. As we've seen in previous sections, the $2 \times n$ entangled graphs are fully classified by the degree criterion. In this section we go into uncharted territory and consider the 3×3 grid-labelled graphs that satisfy the degree criterion, showing that not all such graphs are separable.

It turns out that all such graphs in 3×3 can be constructed from a set of four graphs that we refer to as the *building block graphs*, described below.

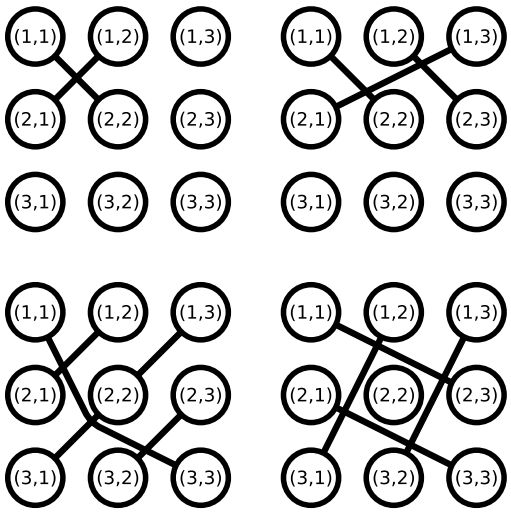


Figure 3.18: Clockwise from top-left, the graphs B_2, B_3, B_4, and B_5.

Definition 3.5.1 (Building-blocks) *The following grid-labelled graphs are illustrated in Figure 3.18.*

- *The* criss-cross B_2 *has two edges:* $\{(1,1),(2,2)\}$ *and* $\{(1,2),(2,1)\}$.
- *The* tally B_3 *has three edges* $\{(1,1),(2,2)\}$, $\{(1,2),(2,3)\}$ *and* $\{(2,1),(1,3)\}$.
- *The* cross-hatch B_4 *has four edges:* $\{(1,1),(2,3)\}$, $\{(2,1),(3,3)\}$, $\{(1,2),(3,1)\}$ *and* $\{(1,3),(3,2)\}$.
- *The* skew-mesh B_5 *has five edges:* $\{(1,1),(3,3)\}$, $\{(1,2),(2,1)\}$ *and* $\{(1,3),(2,2)\}$, $\{(2,2),(3,1)\}$, $\{(2,3),(3,2)\}$.

Let us obtain these graphs from first principles. Along the way, we will define some mathematical objects that will aid us in classifying more complicated examples. It was established early in on this chapter that when considering the graphs that satisfy the degree criterion, we need only consider graphs with diagonal edges. Horizontal and vertical edges remain invariant under partial transpose, and so can be discarded. Note however that it is not clear that we can discard such edges when considering separability. We have no guarantee that for instance, adding a horizontal edge to an entangled state wouldn't cause it to become separable.

It will be useful to further classify diagonal edges into *uphill* and *downhill* edges. Consider an edge $\{(i,j),(k,l)\}$. We say that the edge is uphill if $\text{sgn}(i-k) \neq \text{sgn}(j-l)$, and downhill if $\text{sgn}(i-k) = \text{sgn}(j-l)$. Intuitively, uphill edges are those which travel from bottom-left vertices to top-right vertices, and downhill edges are those which travel from top-left to bottom-right. The grid-labelled graph in Figure 3.19 has 3 downhill edges and 2 uphill edges. In order for a grid-labelled graph to satisfy the degree criterion, the degrees of its vertices must not change after the partial transpose operation. Each edge of the graph contributes 1 to the degrees of its two incident vertices, v_1 and v_2. If the edge is diagonal, then after the partial transpose operation, this edge will contribute 1 to the degrees of two different vertices, $w_1 \neq v_1$ and $w_2 \neq v_2$.

Consider a graph G with a single diagonal edge $\{(1,1),(2,2)\}$. This graph does not satisfy the degree criterion, indeed, without much thought we can convince

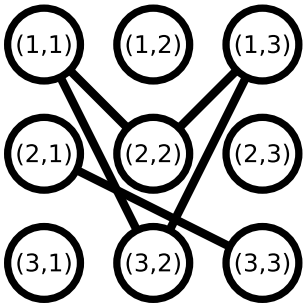


Figure 3.19: A grid-labelled graph with 2 uphill and 3 downhill diagonal edges.

ourselves that there is no grid-labelled graph with a single diagonal edge that satisfies it. Let us consider how we can add a single edge to make this graph satisfy the degree criterion. As we have seen before, horizontal and vertical edges are invariant under partial transpose, so the edge we add must be diagonal. The edge added to G must be added in such a way that the degrees remain the same before and after partial transpose. In order for this to be the case, the new edge must be placed such that after partial transpose, the vertices $(1, 1)$ and $(2, 2)$ have degree 1. Also, the vertices to which the edge moves to after partial transpose, $(1, 2)$ and $(2, 1)$, must have degree 1 *before* partial transpose. The only edge that can be added such that the graph will have these two properties is easily seen to be the edge $\{(1, 2), (2, 1)\}$. With this simple reasoning, we have proved that a grid-labelled graph with two diagonal edges satisfies the degree criterion if and only if it is LE-isomorphic to B_2. Note that this holds for all grid dimensions, although with larger numbers of vertices we will not be able to guarantee that this is true so we will stay in 3×3.

This reasoning about pre- and post-transpose degree properties can be generalised to grid-labelled graphs with more edges via the concept of *edge contributions*, which we will explore in the next section.

3.5.1 Edge contributions

Definition 3.5.2 (Edge contribution matrix) *Let G be an $a \times b$ grid-labelled graph with edge set E. For each diagonal edge $\{(i,j),(k,l)\} \in E$, define its* $a \times b$ edge contribution matrix $A_{(i,j),(k,l)}$ *such that*

$$[A_{(i,j),(k,l)}]_{pq} := \begin{cases} +1 & \text{if } p = i, q = j \text{ or } p = k, q = l; \\ -1 & \text{if } p = i, q = l \text{ or } p = k, q = j; \\ 0 & \text{otherwise.} \end{cases}$$

Definition 3.5.3 (Graph contribution matrix) *Let G be a grid-labelled graph with diagonal edge set $D \subseteq E(G)$. The* contribution matrix *of G is defined*

$$C(G) = \sum_{\{(i,j),(k,l)\} \in D} A_{(i,j),(k,l)}. \tag{3.2}$$

The contribution matrix of a grid-labelled graph encodes the pre- and post-transpose degree contributions of all its edges. If the partial transpose changes the degree of a vertex, then this means that there is some edge whose pre- and post-transpose degree contribution do not match and cancel out. Hence there is a non-zero component somewhere in the contribution matrix of the graph.

Lemma 3.5.4 *Let G be a grid-labelled graph with contribution matrix $C(G)$. Then G satisfies the degree criterion if and only if $C(G)$ is equal to the $a \times b$ matrix with all entries equal to 0.*

Proof. Let G be a grid-labelled graph. Without loss of generality we may assume that it contains only diagonal edges. From the definition of the contribution matrix of a graph it is clear that for each vertex $(i,j) \in V(G)$ with degree d, there will be d edge contribution matrices in the sum in Equation (3.2) that have $+1$ for their ij^{th} element.
If G satisfies the degree criterion then for each vertex $(i,j) \in V(G)$ with degree d there will be d edges $\{(i,l),(k,j)\} \in E(G)$ such that after partial transpose, an endpoint of each edge will be the vertex (i,j). By definition, the edge contribution

matrices of these edges will have -1 in their ij^{th} entry. Hence, the ij^{th} entry of $C(G)$ is zero. This is true for all entries of $C(G)$.
For the other direction, if $C(G)$ is equal to the zero matrix then for each element of the edge contribution matrices in the sum in Equation (3.2) there will be an equal number of positive and negative entries. This means that the degrees of the vertices of the graph are the same before and after the partial transpose, by definition of the graph contribution matrix. ■

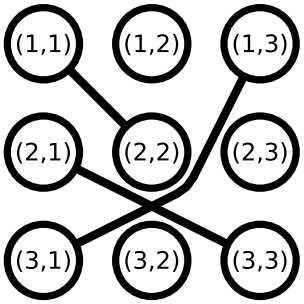


Figure 3.20: A grid-labelled graph that does not satisfy the degree criterion.

Example 3.5.5 The contribution matrix of a graph G with three edges, $\{(1,1),(2,2)\}$, $\{(3,1),(1,3)\}$ and $\{(2,1),(3,3)\}$, as illustrated in Figure 3.20, is

$$\begin{aligned} C(G) &= A_{(1,1),(2,2)} + A_{(3,1),(1,3)} + A_{(2,1),(3,3)} \\ &= \begin{pmatrix} +1 & -1 & 0 \\ -1 & +1 & 0 \\ 0 & 0 & 0 \end{pmatrix} + \begin{pmatrix} -1 & 0 & +1 \\ 0 & 0 & 0 \\ +1 & 0 & -1 \end{pmatrix} + \begin{pmatrix} 0 & 0 & 0 \\ +1 & 0 & -1 \\ -1 & 0 & +1 \end{pmatrix} \\ &= \begin{pmatrix} 0 & -1 & +1 \\ 0 & +1 & -1 \\ 0 & 0 & 0 \end{pmatrix}. \end{aligned}$$

It is clear from the fact that the contribution matrix $C(G)$ is non-zero that the graph G does not satisfy the degree criterion. However, it is easily checked that adding the edge $\{(1,3),(2,2)\}$ causes the contribution matrix to become equal to the zero matrix.

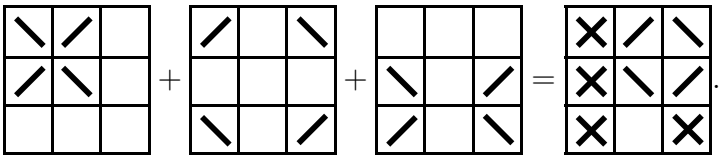
Let us now discuss a method of pictorially representing the contribution matrix

of a graph: *contribution tables*. We will use this pictorial representation heavily in the next few sections, where we try to pick out graphs that satisfy the degree criterion.

Definition 3.5.6 (Contribution table) *Let C be an $a \times b$ edge contribution matrix of dimension $a \times b$. The* contribution table *of C is an $a \times b$ grid, whose cells are populated with diagonal dashes running from top-left to bottom-right (*down *dashes), or bottom-left to top-right multi-particle (*up *dashes).*
The dash placement on the table corresponding to C is as follows. If $C_{ij} = +1$ then place a down dash in the corresponding grid square. If $C_{ij} = -1$ then place an up dash in the corresponding grid square.

We call dashes in the same grid square but in different directions *complementary*. Two complementary dashes are called a *cross* (because the drawing looks like an 'X'). If a cell contains a dash that can not be uniquely paired with a complementary dash, then that dash is called *unmatched*. The *addition* of two contribution tables is the contribution table with all dashes from both tables. The contribution table of a graph is the addition of the contribution tables of each of its edges.

Example 3.5.7 The contribution table of the grid-labelled graph G defined in Example 3.5.5 and illustrated in Figure 3.20 is found to be equal to



The contribution table of G has four crosses and four unmatched dashes.

The next lemma follows directly from Lemma 3.5.4.

Lemma 3.5.8 *Let G be a grid-labelled graph. Then G satisfies the degree criterion if and only if its contribution table contains no unmatched dashes.*

Let us now put the contribution table framework to work.

3.5.2 3×3 degree criterion with 2 diagonal edges

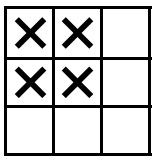
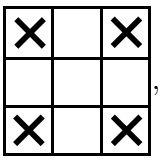
It turns out that the only way to arrange two diagonal edges such that the result satisfies the degree criterion is to arrange them so that they cross over: the criss-cross graph. We prove this in the 3×3 case with the following lemma, which will come in handy later because a lot of the analysis to come will require us to reason about the number of crosses in a contribution table.

Lemma 3.5.9 *Let G be a grid-labelled graph with m edges. If G satisfies the degree criterion, then its contribution table will contain exactly $2m$ crosses.*

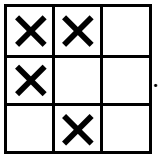
Proof. By definition, each edge in a grid-labelled graph contributes 4 dashes to the contribution table, so there are $4m$ dashes on the table. If a grid-labelled graph satisfies the degree criterion, then we know that each dash must be matched. A 'match' consists of two dashes, meaning there are $4m/2 = 2m$ crosses on the table. ■

Lemma 3.5.10 *Let G be a 3×3 grid-labelled graph with 2 diagonal edges. Then G satisfies the degree criterion if and only if it is locally isomorphic to B_2.*

Proof. From Lemma 3.5.9, it follows that any graph with 2 edges that satisfies the degree criterion must have a table with 4 crosses, for example,


, and

, *etc.*

There are of course several additional tables that have 4 crosses. However, not all of these are associated with a grid-labelled graph. For example, it is clear that there is no 2 edge graph that will have the table


.

The only valid tables are those with four crosses in a rectangular placement, as illustrated in the first example. It is clear that the only grid-labelled graphs which lead to such tables are locally isomorphic to B_2. ■
In proving Lemma 3.5.10 we have seen that there exist contribution tables that do not correspond to graphs. If a contribution table does not correspond to a graph then we call it *invalid*. The following lemma will be useful.

Lemma 3.5.11 *If a* 3×3 *contribution table has a column or row with only* 1 *non-empty cell then it is invalid.*

Proof. By definition, the sum of any number of edge contribution matrices will never produce a matrix with a column or row with only 1 non-zero entry. Hence, no contribution table will have a column or row with only 1 non-empty cell. ■

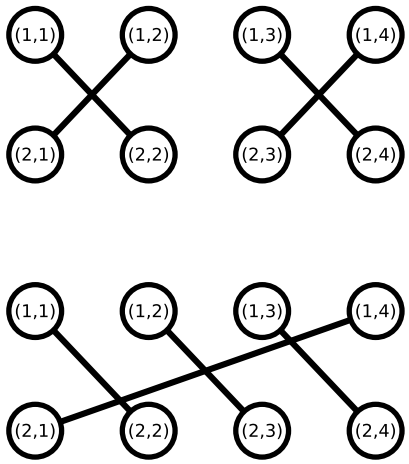


Figure 3.21: Two grid-labelled graphs that have the same contribution table. Note that they both satisfy the degree criterion.

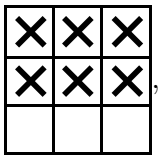
Analogously to the case for grid-labelled graphs, we say that two contribution tables that can be obtained from one another by permuting rows and columns are locally isomorphic. Two locally isomorphic graphs will have locally isomorphic contribution tables. However, the converse is not the case: consider the pair of graphs in Figure 3.21. Both of these have the same contribution table, since both satisfy the degree criterion. However, they are not locally isomorphic graphs. It is clear that for a valid contribution table, there is a set of graphs that have that

contribution table. If two valid contribution tables are locally isomorphic, then the set of corresponding graphs for one can be obtained from the set for the other by performing the corresponding local isomorphisms to the grid-labelled graphs. Using these observations, we can proceed with our characterisation of the 3×3 grid-labelled graphs that satisfy the degree criterion. In what follows we will consider all valid contribution tables for a fixed number of edges, then we will work out the grid-labelled graphs that can correspond to such contribution tables. It does not make sense to consider pairs of contribution tables which can be obtained from one another by rotation since we are only interested in the "topological" properties of the graphs that satisfy the degree criterion.

3.5.3 3×3 degree criterion with 3 diagonal edges

Lemma 3.5.12 *A 3×3 grid-labelled graph G with 3 diagonal edges satisfies the degree criterion if and only if it is locally isomorphic to B_3 or a rotation of B_3.*

Proof. From Lemma 3.5.9 we know that any 3×3 grid-labelled graph G with 3 diagonal edges which satisfies the degree criterion has a contribution table with 6 crosses. It can be verified by direct inspection that all valid 6 cross contribution tables are locally isomorphic to rotations of

 ,

which can only be the contribution table of a grid-labelled graph locally isomorphic to B_3. ■

Notice that our reasoning so far about grid-labelled graphs with 2 or 3 edges that satisfy the degree criterion is valid for arbitrary grid-labelled graphs, not just 3×3. Indeed, this observation along with Lemmas 3.5.10 and 3.5.12 can be used to prove the following.

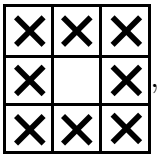
Theorem 3.5.13 *A grid-labelled graph G with 2 (resp. 3) diagonal edges is separable if and only if $G \simeq B_2$ (resp. $G \simeq B_3$).*

Proof. We know from Lemmas 3.5.10 and 3.5.12 that a 3×3 grid-labelled graph with 2 (resp. 3) diagonal edges satisfies the degree criterion if and only if it is locally isomorphic to B_2 (resp. a rotation of B_3). The reasoning in the proofs of these lemmas is independent of grid dimension. It is clear from the structure of B_2 and B_3 that all grid-labelled graphs that are LE-isomorphic to rotations of these grid-labelled graphs are row or column stratified. From Theorem 3.4.25, such grid-labelled graphs are separable. ■

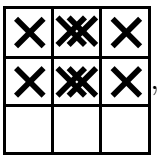
3.5.4 3×3 degree criterion with 4 and 5 diagonal edges

Lemma 3.5.14 *A 3×3 grid-labelled graph with 4 diagonal edges satisfies the degree criterion if and only if it is locally isomorphic to B_4 or the union of two grid-labelled graphs locally isomorphic to B_2.*

Proof. From Lemma 3.5.9, we know we must only consider tables with 8 crosses. We leave to the reader to verify that the only ways of placing 8 crosses on a table such that the table is valid correspond to placements that are locally isomorphic to

 , (3.3)

or locally isomorphic to rotations of either

 , (3.4)

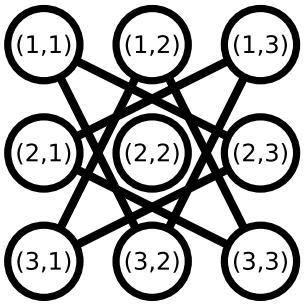
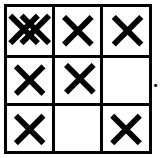


Figure 3.22: Edges that can be removed from the contribution table in Equation (3.3) that lead to a valid contribution table.

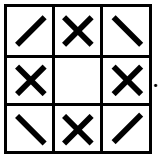
or



(3.5)

Let us now reason about which edges a grid-labelled graph must have to attain the tables (3.3), (3.4), and (3.5), starting with (3.3).

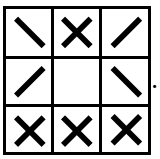
Without loss of generality we may pick any edge supported by the table as a starting point. However, selecting some edges will lead us to a dead-end, unable to proceed further. For an example of an edge leading to a dead-end, let us choose $\{(1,1),(3,3)\}$, removing its associated dashes from the table. This leaves us with



Removing any edge from this table will cause it to have at least 1 row or column with an isolated cross. From Lemma 3.5.11, we know that this will mean the table is invalid.

Indeed, the only single diagonal edges that can be removed from the table in (3.3) that do not lead to a dead-end are those belonging to the graph illustrated in Figure 3.22. Removal of any one of these 8 edges results in a table locally

isomorphic to a rotation of



We leave it to the reader to check that proceeding in this way to add edges to the graph to clear the contribution table will yield the graph B_4. Likewise, each rotation of the table will yield a graph that is locally isomorphic to B_4.
All there is left to prove is that the graphs compatible with tables (3.4) and (3.5) are exclusively made up of the union of two graphs locally isomorphic to B_2. Indeed, this is self-evident. The only compatible grid-labelled graphs are those with their edges arranged into two criss-crosses. ■
Before continuing to the 5 edge case, it will be useful to have the following technical lemmas. The first concerns adding single edges to a graph that satisfies the degree criterion.

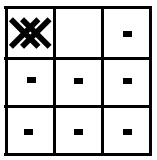
Lemma 3.5.15 *Let G be a grid-labelled graph that satisfies the degree criterion, and let G' be a grid-labelled graph obtained by adding a single diagonal edge to G. Then G' does not satisfy the degree criterion.*

Proof. Since G satisfies the degree criterion, when considering how to add edges to it to preserve the degree criterion it is logically equivalent to considering adding edges to the empty graph. We know that a graph with a single diagonal edge never satisfies the degree criterion, which proves the lemma. ■
The next lemma will make the 5 edge case easier to prove.

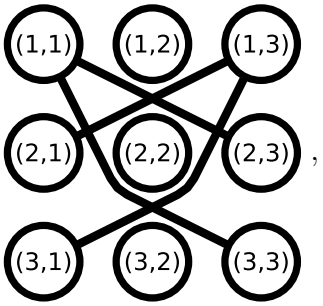
Lemma 3.5.16 *Let G be a 3×3 grid-labelled graph with 5 edges that satisfies the degree criterion. If G has 6 vertices with degree 1, and 2 vertices with degree 2, then G is equal to the union of two graphs locally isomorphic to B_2 and B_3.*

Proof. We will consider a number of possible edge contribution tables. Each will have 6 cells containing a single cross (corresponding to the 6 vertices with degree 1), 2 cells containing a double cross (corresponding to the 2 vertices with degree 2), and a single empty cell. As before, we will start with a particular edge

contribution table, and attempt to clear it by adding edges to an empty 3×3 grid-labelled graph. If we can not clear the table by adding edges, we know that the table is invalid. In order to clear a single cross from the table, 2 edges must be added. In order to clear a double cross from the table, it is clear that 4 edges must be added. A double cross can be in the same row (resp. column) as the empty cell, or on a different row (resp. column).
Let us reason about both cases now. Without loss of generality, assume 1 of the double cross cells is on the same row as the empty cell. Then the table is locally isomorphic to the following,

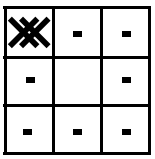


where the black dots denote cells whose contents are irrelevant to the proof. It is obvious that the empty cell restricts the choice of edges that can be selected to remove the double cell. To remove each dash from the double cell under consideration, the four edges in the following graph must be selected,

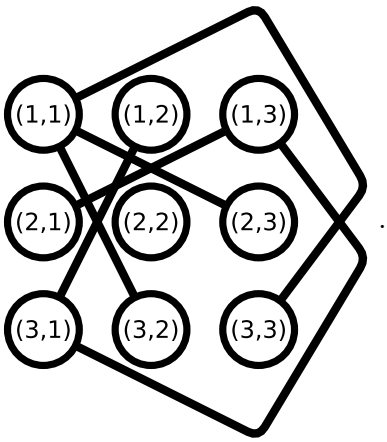


because these are the only edges that are compatible with such a contribution table. Selecting these four edges will produce a grid-labelled graph that is locally isomorphic to the union of 2 grid-labelled graphs locally isomorphic to B_2. We know by Lemma 3.5.15 that a single edge cannot be added to such a graph to produce a grid-labelled graph that satisfies the degree criterion. Such tables are therefore invalid. When a double cross cell is not on the same row as the empty

cell then the table is locally isomorphic to the following:



To remove all the dashes from the double cross cell, 4 of the 6 edges from the following grid-labelled graph must be selected:

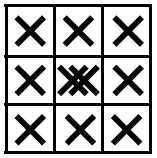


It is impossible to choose 4 edges from this grid-labelled graph without introducing a subgraph locally isomorphic to B_2. Any remaining edges added to this grid-labelled graph in such a way that it would still satisfy the degree criterion would need to be from a grid-labelled graph that itself satisfied it. By Lemma 3.5.12, the only three edge graphs that have this property are locally isomorphic to B_3. The lemma then holds. ■

Lemma 3.5.17 *A 3×3 grid-labelled graph G with 5 diagonal edges satisfies the degree criterion if and only if it is locally isomorphic to B_5 or has a decomposition into two graphs locally isomorphic to rotations of B_2 and B_3 respectively.*

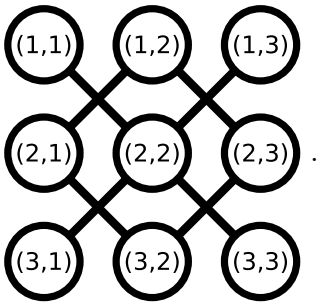
Proof. By Lemma 3.5.9 we must consider all tables with 10 crosses. There are a limited number of ways of placing 10 crosses on a 3×3 table that lead to the table being valid. In order to fit all the crosses on the table, there must be some cells of the table that have double crosses. In Lemma 3.5.16 we have seen that we don't need to consider the tables with 2 double crosses. It is obvious that tables

with more than 2 double crosses are invalid, so, we must only consider the tables with a single double cross. We can reason about these tables by considering the example



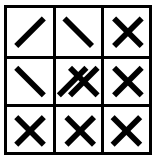
, (3.6)

to which all others of this kind are locally isomorphic. To eliminate the double cross in the middle of the table, 4 edges from the graph



. (3.7)

must be selected. It is clear that the 4 edges must be chosen such that there is no subgraph locally isomorphic to B_2. If there was a such a subgraph, then the remaining 3 edges would need to form a graph locally isomorphic to B_3 in order for the degree criterion to be satisfied, and we have already considered such a case. Consider the removal of the downhill edge $\{(1,1),(2,2)\}$. This leads to the table



.

Clearly the remaining three edges we must choose from Table (3.7) must be chosen to be downhill, as there is no other way of selecting edges without introducing isolated crosses on rows or columns. If instead we chose our first edge to be *uphill*, by similar reasoning the remaining three must also be chosen to be uphill.

We can then observe that removal of 4 uphill or downhill edges from the board (3.6) in the way described leaves a single edge in the direction that completes the grid-labelled graph G_5. Hence, the lemma is proved. ■
The next lemma is proved in Appendix A.3.

Lemma 3.5.18 *Let G be a grid-labelled graph with $6 \leq e \leq 9$ diagonal edges. If G satisfies the degree criterion, then it has a decomposition $\{X_1, \ldots, X_n\}$ where for all $1 \leq i \leq n$, X_i is locally isomorphic to a rotation of a building-block graph.*

Lemma 3.5.19 *Let G be a grid-labelled graph with $e \geq 6$ diagonal edges. If G satisfies the degree criterion, then it has a decomposition $\{X_1, \ldots, X_n\}$ where for all $1 \leq i \leq n$, X_i is locally isomorphic to a rotation of a building-block.*

Proof. A 3×3 grid-labelled graph can have up to 18 diagonal edges. If a 3×3 grid-labelled graph has 10 or more diagonal edges, then it is obvious that it must contain a subgraph locally isomorphic to B_2. The remainder of the cases, graphs with $6 \leq e \leq 9$ diagonal edges, are covered in Lemma 3.5.18 which is proved in Appendix A.3. The lemma then follows by induction. ■
We can finally outline a catalogue of the 3×3 grid-labelled graphs that satisfy the degree criterion.

Theorem 3.5.20 *A 3×3 grid-labelled graph satisfies the degree criterion if and only if it has a decomposition $\{X_1, \ldots, X_n\}$ where each X_i is locally isomorphic to a rotation of a building-block.*

Proof. It is obvious that a grid-labelled graph obtained as a union of building-block graphs will satisfy the degree criterion, because each building-block graph satisfies it.
Let us now prove the other direction. Let G be a grid-labelled graph that satisfies the degree criterion. From Lemmas 3.5.10, 3.5.12, 3.5.14 and 3.5.17 we know that if G has $2 \leq m \leq 5$ diagonal edges then it is locally isomorphic to a building-block B_m, or has a decomposition into grid-labelled graphs that are locally isomorphic to building-blocks, or rotations of building blocks.
Finally, from Lemma 3.5.19, we know that this is also true for graphs with 6 or more diagonal edges. This concludes the proof. ■

We have characterised all of the 3×3 grid-labelled graphs that satisfy the degree criterion. We have found that such grid-labelled graphs satisfy the degree criterion if and only if they are built out of a small set of "building-block graphs". It is reasonable to assume that the same is true for the grid-labelled graphs on a larger grid. A natural question is how the size of the building-block set grows as a function of grid dimension.

In some cases, we can apply these results to characterising not just graphs that satisfy the degree criterion, but graphs that are *separable* in 3×3 and higher. In particular, Theorem 3.5.13 characterises all separable graphs with 2 or 3 diagonal edges (note that all graphs with a single diagonal edge are entangled: they never satisfy the degree criterion).

In arbitrary dimensions a, b there are of course a finite number of graphs with fixed number of edges k that satisfy the degree criterion. Finding all of the building blocks for $k = 4$ and $k = 5$ case would be an interesting extension of the work in this section.

In Appendix A.2 we consider enumerating entangled grid-labelled graphs in some more detail. A fascinating question that we have not yet considered would be to count how many graphs satisfy the degree criterion as a function of k, when fixed grid dimensions are not taken into account. The reasoning to be used in this case would be exactly equivalent to what we have done in this section, but on an infinite grid and with infinite sized contribution tables.

3.5.5 Bound entanglement

As we will verify in sections to come, not all of the 3×3 building-blocks correspond to separable states. It is possible to verify via the *matrix realignment criterion* [151], and the *range criterion* that both B_4 and B_5 correspond to entangled states. Recall that the Peres-Horodecki criterion is not necessary and sufficient in all bipartite dimensions. Likewise, the degree criterion is not sufficient for 3×3 grid-labelled graphs, and so here we have two graphs B_4 and B_5 with density matrices positive under partial transpose, but that are entangled. Such states are known as *bound entangled* states. The term seems to originate in a 1998 work [156] of Michał, Paweł, and Ryszard Horodecki, where they show that if a mixed

state is *distillable* then it must not be positive under partial transpose. It is possible to transform some number of mixed entangled states into a smaller number of arbitrarily pure entangled pairs via a process called *entanglement distillation* [157]. The Horodeckis [156] showed that in order for this distillation process to work, the state must *not* satisfy the PPT criterion. They showed that this implied that some entangled states were not distillable, the entanglement was "bound-up" in an inaccessible form in the state itself and couldn't be extracted into a form usable in quantum protocols *e.g.* teleportation. [3]

The problem of characterising these bound entangled states has attracted some attention from theorists. Whole families of bound entangled states, mostly in $\mathbb{C}^3 \otimes \mathbb{C}^3$ have been discovered, such as the *pyramid* and *tile* states of Bennett et al. [176], revisited some time later in [175] to generalise them to higher dimensions; the *checkerboard* states [171] which stem from an early example of a bound state discovered by Bruß and Peres [172]; and the so called *Smolin state* [164]. The existence of bound entangled states has been experimentally verified, with bound entanglement being prepared in the polarization of photons [158, 161], and also in the continuous variable setting [159, 160]. On the more theoretical side, certain characterisations of bound entanglement in terms of the rank of the density matrix have been attempted [178, 179, 167, 169, 170], with [167] in particular showing that there are no bipartite bound entangled states of rank 2.

While bound entanglement cannot be distilled, it has been shown that they can be used in so called *activation* protocols [162] to improve fidelity of some quantum communication protocols, and also can achieve teleportation with better than classical fidelity [180]. It was shown by Masanes [163] that all bound entangled states can in some way be used as a non-classical resource. It has also been shown that bound entanglement has applications in a cryptographic context for the generation of secure private keys [165, 177].

In the previous paragraphs we alluded to the matrix realignment and range entanglement criteria, which are often used to show that a particular state is bound entangled. In the next two sections we will apply these criteria to grid

[3] Contrast this with *free* states, the term used in [156] for entangled states that are *not* bound which hasn't caught on.

states, showing that they can be recast in terms of graph structure.

3.6 Matrix Realignment Criterion

The matrix realignment criterion is defined in terms of the Ky Fan norm of the *realigned* density matrix of the state under consideration. Let us define these concepts.
Let M be an $m \times n$ matrix. Then its *vectorization* is the $(m \cdot n)$-dimensional row vector

$$\text{vec}(M) := \begin{pmatrix} [M]_{11} & [M]_{21} & \dots & [M]_{m1} & [M]_{12} & \dots & [M]_{mn} \end{pmatrix}.$$

The vectorization of the blocks of a matrix are used to realign it:

Definition 3.6.1 (Realigned matrix) *Let M be an $m \times m$ block matrix with $n \times n$ blocks M_{ij}. The* realignment *of M with respect to n is the $m^2 \times n^2$ matrix*

$$R_n(M) := \begin{pmatrix} \text{vec}(M_{11}) \\ \vdots \\ \text{vec}(M_{m1}) \\ \vdots \\ \text{vec}(M_{1m}) \\ \vdots \\ \text{vec}(M_{mm}) \end{pmatrix}.$$

Definition 3.6.2 (Ky Fan Norm) *The Ky Fan norm of a matrix M is the quantity*

$$\|M\|_K = \sum_i s_i(M),$$

where s_i is the i^{th} singular value of M.

The following theorem describes the entanglement criterion.

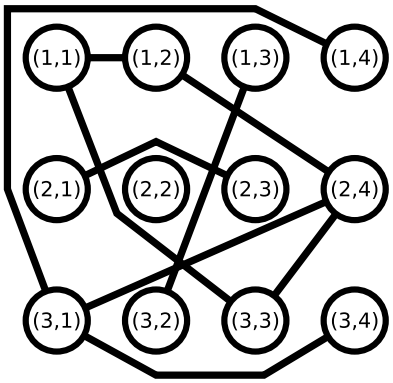


Figure 3.23: A grid-labelled graph, the Laplacian of which we will realign.

Theorem 3.6.3 (Matrix realignment criterion [151]) *Let ρ be the density matrix of a bipartite quantum state with state space $\mathbb{C}^m \otimes \mathbb{C}^n$. If $\|R_n(\rho)\|_K > 1$, then ρ is entangled.*

3.6.1 Realignment of combinatorial Laplacian matrices

The density matrices we consider in this work are real matrices. It is well known that for any real matrix M,

$$s_i(M) = \sqrt{\lambda_i(M^T \cdot M)} = \sqrt{\lambda_i(M \cdot M^T)},$$

where $\lambda_i(M)$ denotes the i^{th} eigenvalue of a matrix M. The fact that $M^T \cdot M$ (resp. $M \cdot M^T$) is real and symmetric implies that $s_i(M) \in \mathbb{R}$. Thus, to apply the matrix realignment criterion to the bipartite quantum state ρ acting on $\mathbb{C}^m \otimes \mathbb{C}^n$ we will require the eigenvalues of the matrices $R_n(\rho)^T \cdot R_n(\rho)$ and $R_n(\rho) \cdot R_n(\rho)^T$. The non-zero eigenvalues of both matrices are identical. Therefore, without loss of generality, we need only consider the eigenvalues of $R_n(\rho) \cdot R_n(\rho)^T$. In particular, we will determine the form of $R_n(\rho) \cdot R_n(\rho)^T$ when ρ is the density matrix of a grid-labelled graph.

Xie *et al.* [143] briefly study the matrix realignment criterion for combinatorial Laplacian matrices and present a structural entanglement criterion. In what fol-

lows we go further, finding a general form for the realigned Laplacian, and using it to find an algebraic entanglement criteria for grid-labelled graphs.
The best way to understand the structure of a realigned Laplacian matrix is to calculate one explicitly. Consider the graph illustrated in Figure 3.23. With some work we can write out its Laplacian matrix

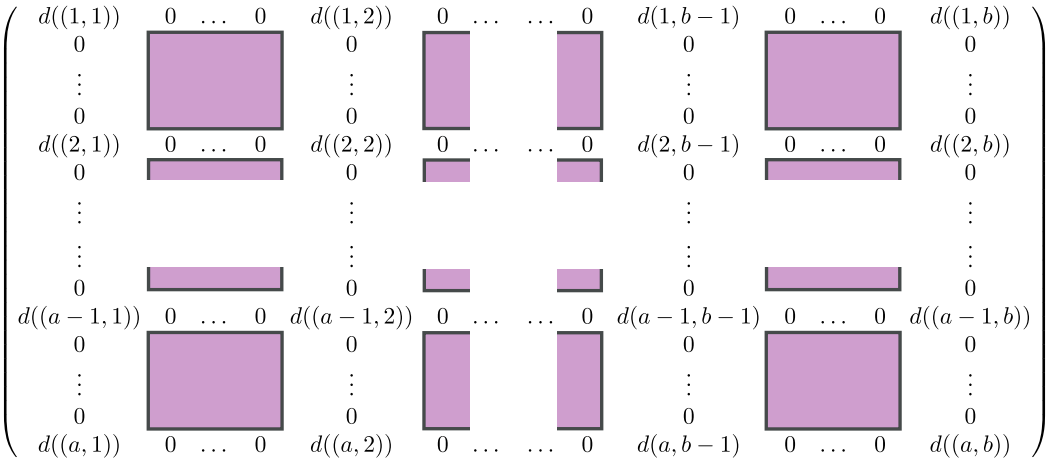
$$L = \begin{pmatrix} 2 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 \\ -1 & 2 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & -1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & -1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 & 0 & 0 & 0 & 3 & -1 & 0 & -1 & 0 \\ 0 & 0 & 0 & -1 & 0 & 0 & 0 & -1 & 3 & 0 & 0 & -1 \\ 0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ -1 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & 2 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & 1 \end{pmatrix}.$$

Realigning according to Definition 3.6.1 we obtain

$$R_4(L) = \begin{pmatrix} \mathbf{2} & -1 & 0 & 0 & -1 & \mathbf{2} & 0 & 0 & 0 & 0 & \mathbf{1} & 0 & 0 & 0 & 0 & \mathbf{1} \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & -1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 \\ \mathbf{1} & 0 & -1 & 0 & 0 & \mathbf{0} & 0 & 0 & -1 & 0 & \mathbf{1} & 0 & 0 & 0 & 0 & \mathbf{3} \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & -1 & 0 \\ 0 & 0 & 0 & -1 & 0 & 0 & -1 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & 0 \\ \mathbf{3} & 0 & 0 & -1 & 0 & \mathbf{1} & 0 & 0 & 0 & 0 & \mathbf{2} & 0 & -1 & 0 & 0 & \mathbf{1} \end{pmatrix}.$$

In the above, certain matrix elements are highlighted in bold. These are the entries that are on the diagonal of L, which represent the vertex degrees. From inspection of the above, it is clear that the realigned Laplacian matrix provides

a "topographic" view of the grid-labelled graph with vertex degrees stored in a grid layout, and with edge information between them. Note for example the edge between vertices $(1,1)$ and $(1,2)$, which manifests itself in the realigned Laplacian as -1 entries at positions $1,2$ and $1,5$.
We leave it to the reader to convince themselves that this topographic structure, in which the matrix directly represents how the graph is oriented on the plane, always emerges when the Laplacian is realigned. We use this topographic structure to our advantage. In particular, consider a grid-labelled graph with no horizontal or vertical edges. It is easy to see that the realigned Laplacian matrix for such a graph will take the following form,



where the filled blocks denote the parts of the matrix where the diagonal edge information is. Let G be an $a \times b$ grid-labelled graph with no horizontal or vertical edges. We use the matrix structure to find a general form of the matrix $R_b(L(G)) \cdot R_b(L(G))^T$. We have that

$$\begin{aligned} R_b(L(G)) &= R_b\left(D(G) - A(G)\right) \\ &= R_b\left(D(G)\right) - R_b\left(A(G)\right). \end{aligned}$$

We know from the structure of the realigned matrices that for G with no horizontal or vertical edges, $R_b(D(G)) \cdot R_b(A(G))^T = R_b(A(G)) \cdot R_b(D(G))^T = 0$, and

so

$$\begin{aligned} R_b(L(G)) \cdot R_b(L(G))^T &= (R_b(D(G)) - R_b(A(G))) \cdot (R_b(D(G)) - R_b(A(G)))^T \\ &= R_b(D(G)) \cdot R_b(D(G))^T + R_b(A(G)) \cdot R_b(A(G))^T. \end{aligned}$$

The first term is easily obtained by direct computation

$$R_b(D(G)) \cdot R_b(D(G))^T =$$
$$\sum_{j=1}^{b} \begin{pmatrix} d((1,j))^2 & 0 & \dots & 0 & d((1,j))d((2,j)) & 0 & \dots & \dots & 0 & d((1,j))d((b,j)) \\ 0 & & & & 0 & & & & & 0 \\ \vdots & & & & \vdots & & & & & \vdots \\ 0 & & & & 0 & & & & & 0 \\ d((2,j))d((1,j)) & 0 & \dots & 0 & d((2,j))^2 & 0 & \dots & \dots & 0 & d((2,j))d((b,j)) \\ 0 & & & & 0 & & & & & 0 \\ \vdots & & & & \vdots & & & & & \vdots \\ \vdots & & & & \vdots & & & & & \vdots \\ 0 & & & & 0 & & & & & 0 \\ d((a,j))d((1,j)) & 0 & \dots & 0 & d((a,j))d((2,j)) & 0 & \dots & \dots & 0 & d((a,j))d((b,j)) \end{pmatrix}.$$

Now we will find a general form for the realigned adjacency matrix. In order to do so, we will make use of the following definitions.

Definition 3.6.4 (Row subgraphs) *Let G be an $a \times b$ grid-labelled graph. Then $R(G,i,j)$ is a* row subgraph *of G: the $2 \times b$ grid-labelled graph with edge set*

$$\{\{(c(i),p),(c(j),q)\} \ : \ \{(i,p),(j,q)\} \in E(G)\},$$

where $c(i) = 1$ and $c(j) = 2$.

Equivalently, $R(G,i,j)$ is the graph with vertex set populated with all vertices from rows i and j of G, and edge set populated by all edges strictly between those rows, and endpoints in different rows. Note that for any grid-labelled graph, all

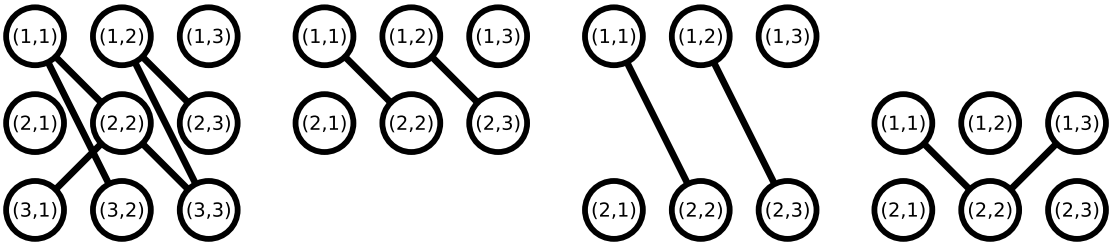


Figure 3.24: From left to right, a grid-labelled graph G, and row subgraphs $R(G,1,2), R(G,1,3), R(G,3,2) \subseteq G$.

edges of any row subgraph $R_{i,i}$ are horizontal, and also for any $R(G,i,j)$,

$$R(G,i,j) = \Gamma(R(G,i,j)).$$

In Figure 3.24 we illustrate several row subgraphs.

Definition 3.6.5 (Intersection) *Let G and H be grid-labelled graphs. Their* intersection *is the grid-labelled graph $G \cap H$ with edge set $E(G) \cap E(H)$.*

The following lemma will aid us in what is to come.

Lemma 3.6.6 *Let G be a grid-labelled graph with adjacency matrix A. Then,*

$$\left[R_b(A) \cdot R_b(A)^T\right]_{ij} = r(i \bmod a, \lceil i/a \rceil; j \bmod a, \lceil j/a \rceil),$$

where

$$r(i,j;k,l) := |E(R(G,i,j) \cap R(G,k,l))|.$$

Proof. By definition,

$$R_b(A(G)) \cdot R_b(A(G))^T = \begin{pmatrix} \text{vec}(A_{11}) \\ \text{vec}(A_{21}) \\ \vdots \\ \text{vec}(A_{aa}) \end{pmatrix} \cdot \left(\text{vec}(A_{11})^T \ \text{vec}(A_{21})^T \ \ldots \ \text{vec}(A_{aa})^T\right),$$

where the $n \times n$ blocks A_{ij} of A have the form

$$A_{ij} = \begin{pmatrix} e(i,1;j;1) & e(i,1;j,2) & \dots & e(i,1;j,b) \\ e(i,2;j;1) & e(i,2;j,2) & \dots & e(i,2;j,b) \\ \vdots & \vdots & \ddots & \vdots \\ e(i,a;j;1) & e(i,a;j,2) & \dots & e(i,a;j,b) \end{pmatrix},$$

where the function $e(p,q;r,s)$ is equal to 1 if there is an edge $\{(p,q),(r,s)\} \in E(G)$, and is equal to 0 otherwise. Hence, the matrix A_{ij} encodes the edges from row i to row j. Clearly

$$\begin{aligned} \text{vec}(A_{ij}) \cdot \text{vec}(A_{kl})^T &= \sum_{p,q=1}^{b} e(i,p;j,q)e(k,p;l,q) \\ &= |E(R(G,i,j) \cap R(G,k,l))| =: r(i,j;k,l). \end{aligned}$$

Therefore,

$$\begin{aligned} \left[R_b(A(G) \cdot R_b(A(G)^T\right]_{ij} &= \text{vec}(A_{i \bmod a, \lceil i/a \rceil}) \cdot \text{vec}(A_{j \bmod a, \lceil j/a \rceil})^T \\ &= r(i \bmod a, \lceil i/a \rceil; j \bmod a, \lceil j/a \rceil). \end{aligned}$$

■

Since G has no horizontal or vertical edges, we have seen from the discussion previously about the topographic structure of these matrices that $R_b(L(G)) = R_b(D(G)) \cdot R_b(D(G))^T + R_b(A(G)) \cdot R_b(A(G))^T$.

For a graph with no horizontal or vertical edges, we can make use of the structure of the matrices $R_b(D(G)) \cdot R_b(D(G))^T$ and $R_b(A(G)) \cdot R_b(A(G))^T$ to simplify further. It is clear from looking at our exploration of the structure of these matrices above that they have disjoint supports. In this case then the matrix $R_b(L(G))$ is permutation similar to a matrix $\mathcal{D}(G) \oplus \mathcal{A}(G)$, where $\mathcal{D}(G)$ and $\mathcal{A}(G)$ are the *degree structure* and *adjacency structure* matrices of G respectively, which we now define. Recall that $d((i,j))$ is the degree of the vertex $(i,j) \in V(G)$.

Definition 3.6.7 (Degree structure matrix) *Let G be an $a \times b$ grid-labelled graph. Then the* degree structure matrix *of G is the $a \times a$ matrix with entries*

$$[\mathcal{D}(G)]_{i,j} := \sum_{p=1}^{b} d((i,p))d((j,p)).$$

Definition 3.6.8 (Adjacency structure matrix) *Let G be an $a \times b$ grid-labelled graph with all edges diagonal. Then the* adjacency structure matrix *of G is the $a(a-1) \times a(a-1)$ matrix*

$$\mathcal{A}(G) := \begin{pmatrix} r(1,2;1,2) & r(1,2;1,3) & \dots & r(1,2;2,1) & \dots & r(1,2;a,a-1) \\ r(1,3;1,2) & r(1,3;1,3) & \dots & r(1,3;2,1) & \dots & r(1,3;a,a-1) \\ \vdots & \vdots & & \vdots & & \vdots \\ r(2,1;1,2) & r(2,1;1,3) & \dots & r(2,1;2,1) & \dots & r(2,1;a,a-1) \\ \vdots & \vdots & & \vdots & & \vdots \\ r(a,a-1;1,2) & r(a,a-1;1,3) & \dots & r(a,a-1;2,1) & \dots & r(a,a-1;a,a-1) \end{pmatrix},$$

where we take $r(i,j;k,l)$ to be equal to $|E(R(i,j)) \cap E(R(k,l))|$.

It is obvious that the degree and adjacency structure matrices of any $a \times b$ grid-labelled graph G with no horizontal or vertical edges can be obtained from $R_b(D(G)) \cdot R_b(D(G))^T$ and $R_b(A(G)) \cdot R_b(A(G))^T$ respectively, by applying suitable permutations and discarding empty rows and columns. We are concerned only with the eigenvalues of these matrices, so empty rows and columns are not important.
We have thus proved the following theorem, where the factor $1/2e$ is for normalisation.

Theorem 3.6.9 *Let G be a grid-labelled graph with e edges, all diagonal. Let $\Lambda := \mathrm{sp}(\mathcal{A}(G))$ and $\Theta := \mathrm{sp}(\mathcal{D}(G))$ be the set of non-zero eigenvalues of the adjacency structure and degree structure matrices of G. Then*

$$\|R_b(\rho(G))\|_K = \frac{1}{2e}\left(\sum_{\lambda\in\Lambda}\sqrt{\lambda} + \sum_{\theta\in\Theta}\sqrt{\theta}\right).$$

In the next section we will use this theorem to demonstrate an infinite family of entangled quantum states that are not detected by the matrix realignment criterion.

3.6.2 Failure of the matrix realignment criterion

In order to proceed, we require the following two definitions.

Definition 3.6.10 (Row orthogonality) *Let G be a grid-labelled graph. We say that G is* row orthogonal *if for all $i, j \in [m]$ and $k \in [n]$,*

$$d((i,k))d((j,k)) \neq 0$$

if and only if $i = j$.

Equivalently, we say that two rows of vertices are orthogonal if each column (two vertices aligned vertically) has at least one isolated vertex. A grid-labelled graph is then row orthogonal if all of its vertex rows are pairwise orthogonal.

Definition 3.6.11 (Singleton edge) *Let G be a grid-labelled graph. An edge $\{v_1, v_2\} \in E(G)$ is described as a* singleton *if*

$$d(v_1) = d(v_2) = 1.$$

We can now prove the following theorem.

Theorem 3.6.12 *Let G be a $2 \times b$ row orthogonal grid-labelled graph with e edges, such that all edges are singleton and diagonal. If $e \geq 4$ then $\|R_b(\rho(G))\|_K \leq 1$.*

Proof. Since G has $a = 2$ rows,

$$\mathcal{A}(G) = \begin{pmatrix} r(1,2;1,2) & r(1,2;2,1) \\ r(2,1;1,2) & r(2,1;2,1) \end{pmatrix},$$

and

$$\mathcal{D}(G) = \sum_{j=1}^{b} \begin{pmatrix} d((1,j))^2 & d((1,j))d((2,j)) \\ d((2,j))d((1,j)) & d((2,j))^2 \end{pmatrix}.$$

Since G is row orthogonal,

$$r(1,2;2,1) = r(2,1;1,2) = 0,$$

and

$$\sum_{j=1}^{b} d((1,j))d((2,j)) = \sum_{j=1}^{b} d((2,j))d((1,j)) = 0,$$

so both $\mathcal{A}(G)$ and $\mathcal{D}(G_l^{2,b})$ are diagonal. Hence,

$$\begin{aligned}\|R_b(\rho(G))\|_K &= \frac{\sqrt{r(1,2;1,2)} + \sqrt{r(2,1;2,1)} + \sqrt{\sum_{j=1}^{b} d((1,j))^2} + \sqrt{\sum_{j=1}^{b} d((2,j))^2}}{2e} \\ &= \frac{2\sqrt{e} + \sqrt{\sum_{j=1}^{b} d((1,j))^2} + \sqrt{\sum_{j=1}^{b} d((2,j))^2}}{2e}.\end{aligned}$$

Since all edges are singleton, all degrees are either 0 or 1. This means that

$$\begin{aligned}\|\rho(G)\|_K &= \frac{4\sqrt{e}}{2e} \\ &= \frac{2}{\sqrt{e}},\end{aligned}$$

from which the theorem follows. ■

The fact that such grid-labelled graphs are row orthogonal means that they do not satisfy the degree criterion. Since the degree criterion is necessary and sufficient for separability for grid-labelled graphs with $a = 2$ (Theorem 3.4.7), these grid-labelled graphs describe quantum states that are entangled. An example of a family of grid-labelled graphs with the properties described in Theorem 3.6.12 are the $2 \times 2k$ graphs with edge sets

$$E_k := \bigcup_{i=1}^{k} \Big\{ \{(1, 2i-1), (2, 2i)\} \Big\}$$

for $k \geq 4$. For example, the $k = 4$ case corresponds to the quantum state with

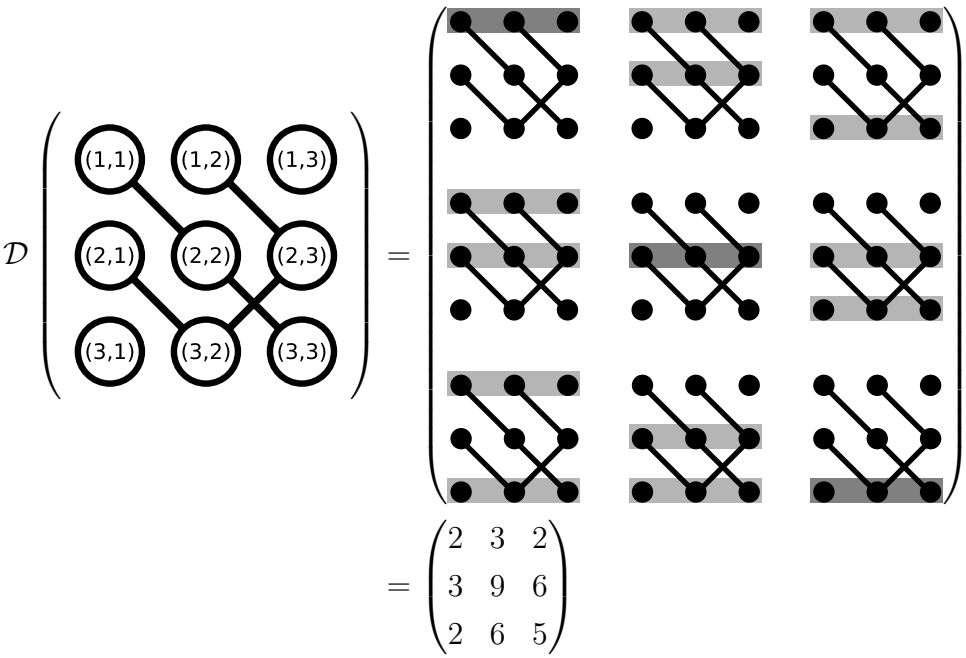
density matrix

$$\rho = \frac{1}{4}\big(|1,1;2,2\rangle\langle 1,1;2,2| + |1,3;2,4\rangle\langle 1,3;2,4| + |1,5;2,6\rangle\langle 1,5;2,6| + |1,7;2,8\rangle\langle 1,7;2,8|\big)$$

acting on $\mathbb{C}^2 \otimes \mathbb{C}^8$.

In this section we have applied the matrix realignment criterion to grid-labelled graphs. We showed that entanglement in grid-labelled graphs can be detected using the eigenvalues of their adjacency and degree structure matrices. We used this result to construct a family of entangled quantum states that are not detected as entangled by the matrix realignment criterion. Let us conclude by explicitly calculating the adjacency and degree structure matrices for an example grid-labelled graph, in the hope that it will be illuminating for the reader.

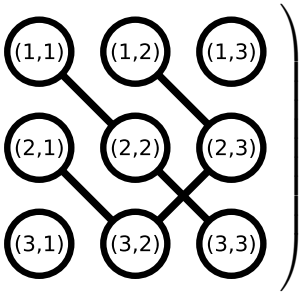
Example 3.6.13 Let us calculate the degree and adjacency structure matrices for an example graph.

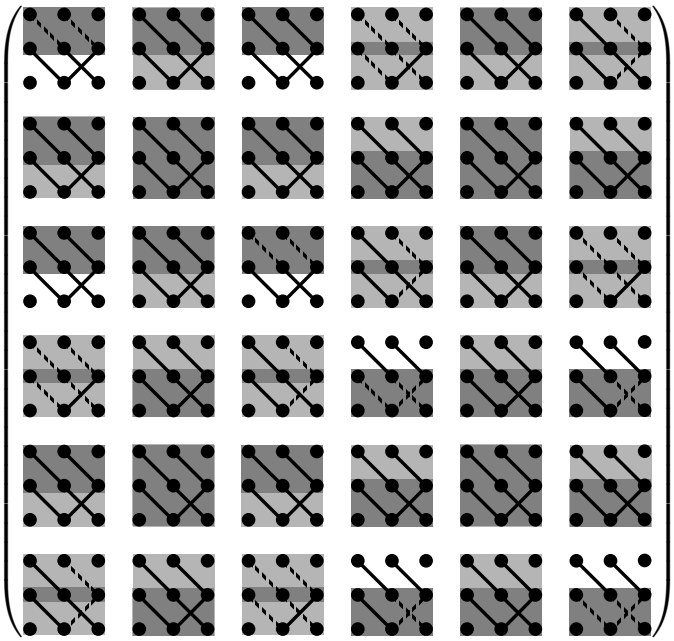


$$= \begin{pmatrix} 2 & 3 & 2 \\ 3 & 9 & 6 \\ 2 & 6 & 5 \end{pmatrix}$$

As illustrated above, finding the entries of the degree structure matrix of a $m \times n$ grid-labelled graph amounts to taking the pairwise inner product of the m vectors

of dimension n which have as their entries the degrees of the vertices in that row. It can be seen that the entries of the adjacency structure matrix are in some sense equal to the "overlap" of each row subgraph, as we now illustrate pictorially:

$\mathcal{A}$ 

$=$ 

$$= \begin{pmatrix} 2 & 0 & 0 & 2 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 2 & 1 & 0 & 2 \\ 2 & 0 & 1 & 3 & 0 & 2 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 2 & 2 & 0 & 3 \end{pmatrix},$$

where shaded regions denote the row subgraphs being compared. For example, the entry at index $(6, 3)$ compares the row subgraphs $R_{1,2}$ and $R_{2,3}$, which have 2 edges in their intersection. The entry at index $(3, 1)$ compares $R_{1,2}$ and $R_{2,1}$, which have 0 edges in their intersection.

3.6.3 Applying the matrix realignment criterion to B_4 and B_5.

Let us now apply the matrix realignment criterion to the building block B_4. Some calculation yields the following

$$\mathcal{D}(B_4) = \begin{pmatrix} 3 & 2 & 3 \\ 2 & 2 & 2 \\ 3 & 2 & 3 \end{pmatrix}$$

which has eigenvalues $2(2 \pm \sqrt{3})$, and

$$\mathcal{A}(B_4) = \begin{pmatrix} 1 & 0 & 0 & 1 & 0 & 0 \\ 0 & 2 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 2 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix}$$

which has 4 rows with only diagonal non-zero entries, and so examining these rows will have two of its eigenvalues equal to 1, and another two of its eigenvalues equal to 2. Hence, applying Theorem 3.6.9 we have that $\|\rho(B_4)\|_K \geq (2 + 2\sqrt{2} + \sqrt{2(2+\sqrt{3})} + \sqrt{2(2-\sqrt{3})})/8 \approx 8.3/8 > 1$ and so via the matrix realignment criterion the state is entangled.

The graph B_5 is not as convenient to handle as B_4, since $\mathcal{A}(B_5)$ doesn't have the

almost diagonal form of $\mathcal{A}(B_4)$. The realigned Laplacian matrix is equal to

$$R_n(L(B_5)) = \frac{1}{10}\begin{pmatrix} 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & -1 & 0 & 0 & 0 & -1 & 0 \\ 0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 & 0 & -1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 2 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & -1 & 0 & 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 \\ 0 & -1 & 0 & 0 & 0 & -1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 \end{pmatrix}.$$

The singular values of this matrix as reported by Mathematica are

$$\left\{ \frac{\sqrt{2(3 \pm 2\sqrt{2})}}{10}, \frac{1}{5}, \frac{1}{5}, \frac{1}{10}, \frac{1}{10} \right\}.$$

The sum of these is exactly 1, so the matrix realignment criterion is inconclusive. Confusingly, the *IsSeparable* method in the QETLAB [154] library reports that the state is entangled, and that it came to this conclusion by applying the matrix realignment criterion. Computing the singular values in the NumPy linalg [155] library yields 9 non-zero singular values, 3 of which are very small. Since the realigned Laplacian is rank 6 (see that there are exactly 6 linearly independent rows: rows 1 through 5, and row 7), these 3 small singular values should be considered to be numerical errors, and be taken to be 0. It is therefore clear that QETLAB is erroneously reporting that the matrix realignment criteria points to the state being entangled because it is including these three small non-zero entries in the sum from the definition of the Ky Fan norm.
While it is not possible to use the matrix realignment criterion to prove it, the state $\rho(B_5)$ does turn out to be entangled. This can be shown using the range criterion as we see in the next section.

3.7 Range criterion

In this section we will explore how another entanglement criterion, the range criterion, can be applied to grid states. First, let us state the criterion in its most general form. Recall the following from basic linear algebra.

Definition 3.7.1 (Range and kernel of a matrix) *Let M be a matrix acting on a finite dimensional Hilbert space $\mathbb{C}^d$. We say that a vector $\vec{v} \in \mathbb{C}^d$ is in the range of M if there is a non-zero vector $\vec{w}$ such that $M\vec{w} = \vec{v}$. We say that $\vec{v}$ is in the kernel of M if $M\vec{v} = \vec{0}$.*

The set of vectors in the range (resp. kernel) of a matrix M forms a vector space, which we denote by $R(M)$ (resp. $K(M)$).

Theorem 3.7.2 (Rank-nullity) *Let M be a matrix acting on $\mathbb{C}^n$. Then* $\dim(R(M))+\dim(K(M)) = n$.

The following encompasses the range criterion.

Theorem 3.7.3 (Horodecki '97) *Let ρ be a density matrix acting on a multi-particle Hilbert space $\mathbb{C}^{d_1} \otimes \cdots \otimes \mathbb{C}^{d_n}$. If ρ is separable then there exists a set of product vectors $P = \{|\psi_1^{(i)}\rangle \otimes \cdots \otimes |\psi_n^{(i)}\rangle\}_i$ such that $R(\rho) = \text{span}_{\mathbb{C}} P$.*

The following corollary will be useful for our purposes.

Corollary 3.7.4 *Let ρ be a density matrix acting on $\mathbb{C}^a \otimes \mathbb{C}^b$. If for all product states $|\psi\rangle|\phi\rangle \in \mathbb{C}^a \otimes \mathbb{C}^b$ we have that $|\psi\rangle|\phi\rangle \notin R(\rho)$ then ρ is entangled.*

It is possible to use the range criterion to show that $\rho(B_5)$ is entangled. The proof is long and uses only elementary linear algebra, so we relegate it to Appendix A.1. Let us now dig into the details of how the criterion can be applied to grid states.

3.7.1 Row and column surgery

In order to apply the range criterion to a grid state, we will need to reason about its range. The range of a graph Laplacian can be complicated, so we will make use of a technique called *surgery* to simplify things.

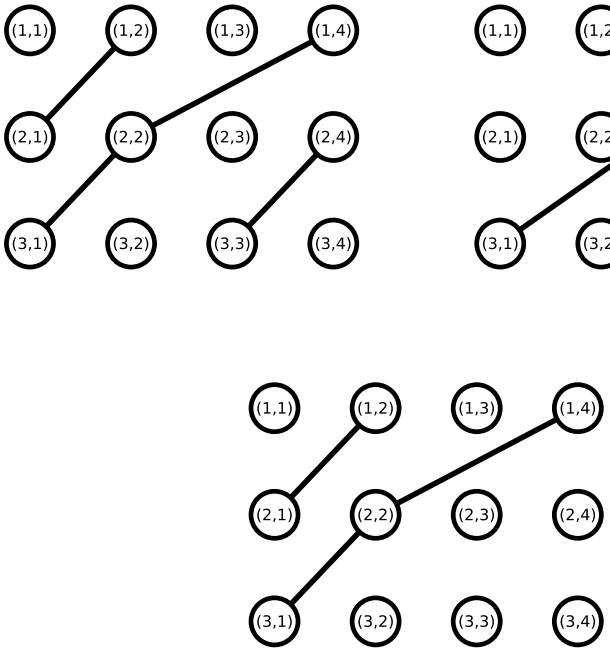


Figure 3.25: Consider vertex $(2, 3)$ of the top-left graph. Performing row surgery yields the graph on the top-right, and performing column surgery yields the bottom graph.

Definition 3.7.5 (Row/Column Surgery) *Let G be a grid-labelled graph with an isolated vertex (i, j). Then $G^R_{i,j}$ (resp. $G^C_{i,j}$) is the graph obtained by performing row (resp. column) surgery which has two steps:*

- *CUT: for all edges $e \in E(G)$, if e is incident to a vertex in row i (resp. column j) then remove it.*
- *STITCH: add the minimum possible number of edges between vertices not in row i (resp. row j) such that the following property holds: for all vertices $v, w \in V(G)$ not in row i (resp. column j), if there was a path between them in G then there is a path between them in $G^R_{i,j}$ (resp. $G^C_{i,j}$).*

In Figure 3.25 we show how surgery is performed on a graph. Note that the CUT step of row surgery on $(2, 3)$ causes vertices $(3, 1)$ and $(1, 4)$ to no longer be in the same connected component. This is fixed by the STITCH step, where an edge is added between them. The following proposition shows how we will use this graph simplification procedure to study the range of a grid-labelled graph.

Proposition 3.7.6 *Let G be a grid-labelled graph with isolated vertex $(i,j) \in V(G)$. For all product vectors $|\alpha\rangle|\beta\rangle \in \mathbb{C}^m \otimes \mathbb{C}^n$, if $|\alpha\rangle|\beta\rangle \in R[\rho(G)]$ then $|\alpha\rangle|\beta\rangle \in R[\rho(G^R_{(i,j)})]$ or $|\alpha\rangle|\beta\rangle \in R[\rho(G^C_{(i,j)})]$.*

Before proving this result, we need two lemmata. The first provides a characterization of the range of a grid state. For its formulation, we denote by $C(G)$ the set of connected components of a graph. Here, also disconnected vertices are considered to constitute a connected component. We also associate with every grid-labelled graph G the state $|G\rangle = \sum_{(i,j)\in V(S)} |ij\rangle$[4]. This construction can also be applied to a single connected component $S \in C(G)$.

Lemma 3.7.7 *Let G be an $m \times n$ grid-labelled graph, and let $C(G)$ denote the set of its connected components. Then $|\psi\rangle \in R(\rho(G))$ if and only if $|\psi\rangle \perp |S\rangle$ for all $S \in C(G)$. This implies that for any $m \times n$ vertex grid-labelled graph G, the dimension of the kernel of $\rho(G)$ is equal to the number of connected components $|C(G)|$. Therefore, the rank of $\rho(G)$ is equal to $m \cdot n - |C(G)|$.*

Proof. For all graphs G, $\rho(G)$ is Hermitian so $|\psi\rangle \in R(\rho(G))$ if and only if $|\psi\rangle \perp K[\rho(G)]$. For any connected component $S \in C(G)$ with k vertices, $\rho(S)|S\rangle = 0$, so $|S\rangle \in K[\rho(S)]$. Since S is connected, it has a spanning tree T with $k-1$ edges. The edges of T correspond to a set of linearly independent vectors $(|ij\rangle - |kl\rangle)/\sqrt{2}$ in the range of $R[\rho(S)]$, so $\dim(K[\rho(S)]) = k - (k-1) = 1$. Therefore, $K[\rho(S)] = \text{span}_{\mathbb{C}}(|S\rangle)$.
The density operator $\rho(G)$ can be decomposed in terms of $C(G)$,

$$\begin{aligned}
\rho(G) &= \frac{1}{2|E|} \sum_{\{(i,j),(k,l)\}\in E(G)} (|ij\rangle - |kl\rangle)(\langle ij| - \langle kl|) \\
&= \frac{1}{2|E|} \sum_{S\in C(G)} 2|E(S)|\rho(S) \\
&= \sum_{S\in C(G)} \frac{|E(S)|}{|E(G)|}\rho(S). \\
&= \sum_{S\in C(G)} \frac{|E(S)|}{|E(G)|} \sum_{(i,j),(k,l)\in E(S)} (|ij\rangle - |kl\rangle)(\langle ij| - \langle kl|).
\end{aligned}$$

[4]Note that this is not the same as the graph states we consider in Chapter 2.

By definition the components S have no edges or vertices in common, so $|\psi\rangle \perp K[\rho(G)]$ if and only if $|\psi\rangle \perp K[\rho(S)] = \text{span}_{\mathbb{C}}(|S\rangle)$ for all $S \in C(G)$. ■
To proceed further, we will need to define the vectors

$$|G_{i,*}\rangle := \sum_{\substack{(k,l)\in V(G)\\ k\neq i}} |kl\rangle \tag{3.8}$$

and

$$|G_{*,j}\rangle := \sum_{\substack{(k,l)\in V(G)\\ l\neq j}} |kl\rangle, \tag{3.9}$$

for any grid-labelled graph G. Then we have:

Lemma 3.7.8 *Let G be a grid-labelled graph with $m \times n$ vertices. If a state $|\psi\rangle$ is orthogonal to all states in*

- $\{|S_{i,*}\rangle \ : \ S \in C(G)\}$ *and* $\{|i,1\rangle, \ldots, |i,n\rangle\}$ *then* $|\psi\rangle \in R[\rho(G^R_{(i,j)})]$*;*
- $\{|S_{*,j}\rangle \ : \ S \in C(G)\}$ *and* $\{|1,j\rangle, \ldots, |m,j\rangle\}$ *then* $|\psi\rangle \in R[\rho(G^C_{(i,j)})]$.

Proof. It is clear that $G^R_{(i,j)}$ can be obtained by performing surgery on each connected component of G. For such a component $S \in C(G)$, we have that $K[\rho(S)] = \sum_{(k,l)\in V(S)} |kl\rangle$. Performing the CUT step of surgery on row i of S removes all edges to vertices in that row, which introduces new isolated vertices. The STITCH step then ensures that the remnants of the graph remain connected. Therefore, if a state $|\psi\rangle$ is orthogonal to $\sum_{(k,l)\in V(S)} |kl\rangle$ for $k \neq i$, and is orthogonal to $\{|i,q\rangle\}$ for all of the new isolated vertices (i,q), then it is in the range of $\rho(S^R_{(i,j)})$ by Lemma 3.7.7. It is clear that if $|\psi\rangle$ is orthogonal to each of the states $|S_{i,*}\rangle$ for $S \in C(G)$, as well as all the isolated vertex states $|i,1\rangle, \ldots, |i,m\rangle$ introduced by performing CUT on each component then it is in the range of $\rho(G^R_{(i,j)})$ by Lemma 3.7.7. By similar reasoning, the same is true for the graph obtained by column surgery. ■
We may now prove Proposition 3.7.6.
Proof of Proposition 3.7.6. Since (i,j) is isolated then $|i,j\rangle \in K[\rho(G)]$. Therefore, if $|\alpha\rangle|\beta\rangle \in R(\rho(G))$ then either $|\alpha\rangle \perp |i\rangle$ or $|\beta\rangle \perp |j\rangle$. Suppose the former

is the case. Then clearly $|\alpha\rangle|\beta\rangle$ is orthogonal to all $|i,1\rangle,\ldots,|i,m\rangle$. Further, we know that for all $S \in C(G)$, $|\alpha\rangle|\beta\rangle$ is orthogonal to $|S\rangle$, and so must be orthogonal to $|S_{i,*}\rangle$. Therefore, by Lemma 3.7.8 it must be in the range of $\rho(G^R_{(i,j)})$. If we instead assume that $|\beta\rangle \perp |j\rangle$ then by similar reasoning, $|\alpha\rangle|\beta\rangle \in R[\rho(G^C_{(i,j)})]$. ■
We can use some consequences of Proposition 3.7.6 to come up with a "graphical range criterion" and discover some bound entangled grid-states.

3.7.2 Bound entangled graphs via the range criterion

We warm up by proving something we already know, that B_4 is entangled. We do this by showing that there are no product vectors in the range of $\rho(B_4)$, which is achieved by considering the result of performing successive surgeries on the graph and applying Proposition 3.7.6.

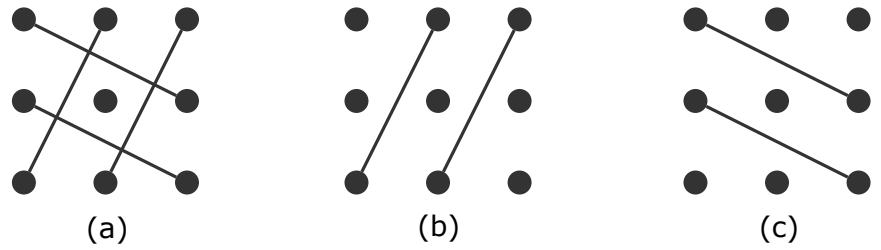


Figure 3.26: (a) The cross-hatch graph B_4 (b) B_4 after performing row surgery on isolated vertex $(2,2)$ (c) B_4 after performing column surgery on isolated vertex $(2,2)$.

From examining the illustration of graph B_4 (Figure 3.26 (a)), we see that it has a single isolated vertex, $(2,2)$. Suppose that there is a product vector $|\psi\rangle|\phi\rangle =: |\psi_p\rangle$ in the range of $\rho(B_4)$. Then we know from Proposition 3.7.6 that this product vector is in the range of the graph obtained by performing row surgery on this vertex (illustrated in Figure 3.26 (b)) or of the graph obtained by performing column surgery (illustrated in Figure 3.26 (c)). In order to understand where the vector $|\psi_p\rangle$ could fit, we now must understand the ranges of these two new graphs. We do this by performing more surgeries. When those surgeries are done, we consider surgeries on the new graphs, and so on.
Let us consider the first graph above, obtained by performing row surgery on B_4. We now have a number of isolated vertices that we could perform surgery on:

$(1,2), (1,3), (2,2), (3,1)$, and $(3,2)$. For simplicity, let's see what happens when we try performing surgery on the same vertex again, $(2,2)$. Performing row surgery on this vertex will eliminate both edges from the graph, leaving us with the 3×3 empty graph. However, performing column surgery doesn't remove any edges, leaving the graph unchanged. In this case, applying Proposition 3.7.6 is useless, all that we have been able to discover is that if our product vector $|\psi_p\rangle$ is in the range of $\rho(B_4')$, then it is either in the range of an empty graph, or in the range of $\rho(B_4')$. Of course, this is a tautology. Performing surgeries on the other isolated vertices in column 2 has the same effect, we don't learn anything new about the range by performing surgery there. It's clear in general that we can only learn something new by performing surgery on isolated vertices that are not part of a full row or column of isolated vertices. We call isolated vertices that are not part of an isolated row/column *viable vertices*.

The graph we are considering has 2 viable vertices: $(1,3)$ and $(3,1)$. We need only consider one of these due to symmetry, it's clear from the structure here that the surgeries will yield the same graphs. Column surgery on $(1,3)$ yields the empty graph, row surgery yields a graph with a single edge $\{(2,1),(3,3)\}$ From Proposition 3.7.6, we know that if our product vector is in the range of the graph in Figure 3.26 (c), then it is in the range of either the empty graph, or the single edge graph above. It's clear furthermore that any surgery on a viable vertex of this single edge graph will remove the edge, resulting in the empty graph. Now we are making progress: if we can show that any sequence of surgeries starting from vertex $(2,2)$ in graph B_4 will terminate in an empty graph then we have shown that there are no product vectors in the range of $\rho(B_4)$. Proposition 3.7.6 tells us that the product vector $|\psi_p\rangle$ must be in the range of one of these empty graphs, but this is not possible since the empty graph corresponds to the zero matrix, which has no non-trivial product vectors in its range.

In Figure 3.27 we summarise the above reasoning with a picture. We illustrate all sequences of surgeries starting from $(2,2)$ of B_4 (the starting point B_4 is marked with a star on the figure). Note that we cut off part of the tree for brevity, this part can be obtained easily from symmetry. From examining this "surgery tree" it is clear that all surgeries will terminate in empty graphs, implying that there are no product vectors in $R(\rho(B_4))$.

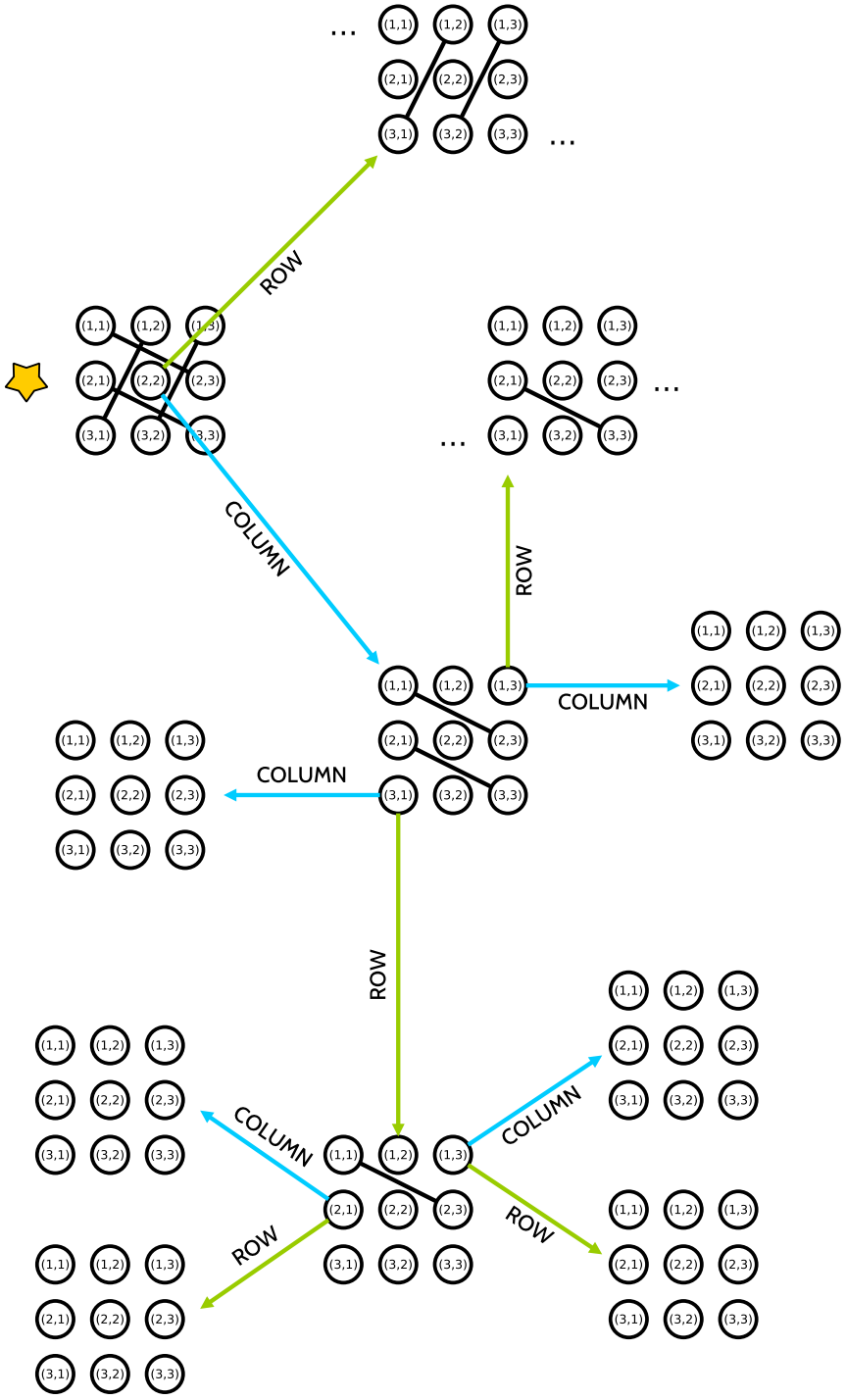


Figure 3.27: A series of surgeries being performed on B_4. We start at the starred graph.

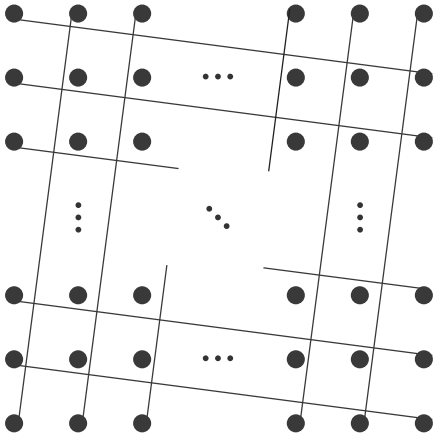


Figure 3.28: The cross-hatch graph B_4 in arbitrary grid dimensions.

We therefore have the following.

Lemma 3.7.9 *The grid state $\rho(B_4)$ is entangled.*

The cross-hatch pattern of B_4 can be extended to arbitrary grid dimensions, as illustrated in Figure 3.28. With some thought we can show that this generalised cross-hatch graph is entangled in all grid dimensions, not just 3×3.

Theorem 3.7.10 *The cross-hatch graph in $m \times n$ is entangled for all $m, n \geq 3$.*

Proof. Examining Figure 3.28, it is clear that the vertices $(i, j) \in \{2, \ldots, m-1\} \times \{2, \ldots, n-1\}$ are always viable for arbitrary $m, n \geq 3$. Performing the CUT step of row surgery on one of these viable vertices (i, j) removes two edges: $\{(i-1, 1), (i, n)\}$ and $\{(i, 1), (i+1, n)\}$. Likewise, performing the CUT step of column surgery on (i, j) removes $\{(1, j-1), (m, j)\}$ and $\{(1, j), (m, j+1)\}$. Since all of the edges in the graph are singleton (see Definition 3.6.11), removing these edges will not require any application of the STITCH step in the surgery: there are no connected components that will become disconnected by the CUT step. We are left with a proper subgraph of the graph we started with. This subgraph will always have viable vertices, and so another surgery can be performed, which removes two more edges either running vertically or horizontally. It is clear that any non-empty subgraph of the original cross-hatch graph has at least two viable vertices, so surgery can always be performed. Furthermore, each such subgraph can have surgery performed on it without needing to perform the

STITCH step and add new edges. Therefore, each surgery always produces a proper subgraph of the original cross-hatch graph. Therefore, all sequences of surgeries will terminate in the empty graph, which implies that there are no product vectors in the range of the original cross-hatch graph. ■

In proving the above, we have started to get a taste of some techniques we can apply in our search for bound entangled grid states. First of all, we have identified a special type of grid-labelled graph: those for which all sequences of surgeries starting on any of their viable vertices terminate in the empty graph. While not all entangled grid states have this property (consider the 2×3 graph with edges $\{(1,1),(2,2)\}$ and $\{(2,2),(1,3)\}$: row surgery on $(2,1)$ unavoidably introduces a horizontal edge $\{(1,1),(1,3)\}$ that is impossible to remove by subsequent surgeries), it is clear that graphs with this property have no product states in their range, and are therefore entangled. We call these graphs *clearable*.

The cross-hatch graph has another useful property that helps us in showing that it is clearable: for all sequences of surgeries, at no point will a STITCH step need to be applied. This means that all we need to do to prove that it is clearable is to show that all subgraphs have viable vertices. In general, if a graph has these two properties then we know that at all stages we will be able to perform both kinds of surgery (since surgery will always produce a subgraph via removal of edges), and the number of edges will always decrease. This means that all sequences of surgeries eventually end up with the empty graph and so the original graph is clearable. An example of a family of graphs with the "no-STITCH" property are the graphs where all edges are singleton. This is because all connected components have either 1 or 2 vertices since they either correspond to an isolated vertex or they correspond to a singleton edge. If a 2 vertex connected component becomes disconnected by a CUT, then there is no need to add an edge to reconnect the vertices since one of them must be on the row or column that the surgery is being performed with respect to. We can prove the following about such singleton graphs.

Lemma 3.7.11 *Let G be a singleton graph. If for all non-empty subgraphs $H \subseteq G$, H has a viable vertex, then G is clearable.*

Proof. Trivially, $G \subseteq G$ so it has a non-empty set of viable vertices W. For any

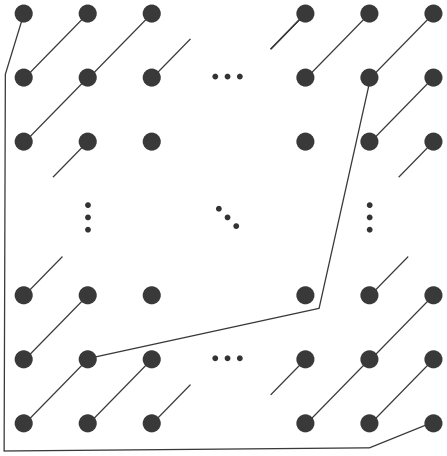


Figure 3.29: The building block graph B_5, the "skew mesh" graph, in arbitrary dimensions.

$w \in W$, performing a surgery will remove at least one edge from the graph. Since G is singleton, none of its components will become disconnected in a way that requires edges to be added in the STITCH step. Therefore, for all $w \in W$ and for both kinds of surgery, we obtain a proper subgraph $H_w \subset G$, and furthermore, this subgraph is singleton since edges have only been removed, not added. By assumption, H_w must have at least one viable vertex. The same reasoning holds for this new subgraph, and so the process can be iterated until there are no more edges. Therefore, all sequences of surgeries will result in the empty graph, and so G is clearable. ■

Let us now consider a different type of bound entangled graph, the generalisation of B_5, as illustrated in Figure 3.29. We are not able to apply Lemma 3.7.11 here since such graphs are not singleton. However, we can strengthen Lemma 3.7.11 into something more useful by considering which edges could potentially be added by any STITCH step. Consider a connected component of a graph $C \subseteq G$. If C became disconnected due to the edges of a particular vertex $v \in C$ being removed, then we would need to apply the STITCH step: new edges would need to be added in order to reconnect the relevant parts of C again. There is no reason to add an edge that is incident to a vertex that is not part of C, since by definition the STITCH step adds the minimum number of edges. Therefore, when we consider the set of all edges that could potentially be added in a STITCH step

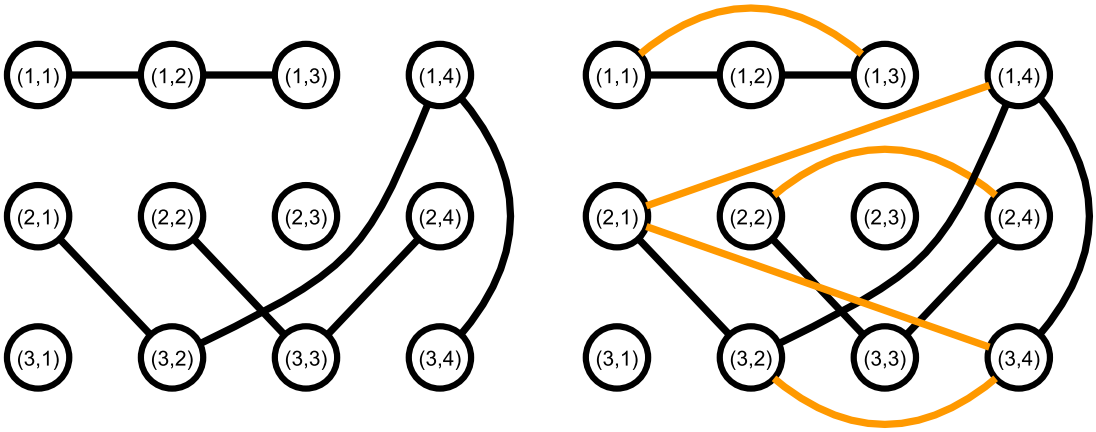


Figure 3.30: Left is a grid-labelled graph, to the right is its completion with new edges highlighted in orange.

to "repair" C, we need only consider all possible edges between the vertices of C. This leads us to our next definition.

Definition 3.7.12 *Let G be a grid-labelled graph, and let $C \subseteq G$ be a connected component of G with more than one vertex. The completion of G with respect to C is the graph obtained by adding edges between all pairs of vertices of C. The completion of G is the graph obtained by completing all of its non-trivial connected components.*

Example 3.7.13 *In Figure 3.30 we illustrate a graph and its completion.*

We now have a bit of a handle on which edges could potentially be added to a graph during a sequence of surgeries. If we know that all possible graphs that can come out of a surgery will have viable vertices then we know that for no sequence of surgeries will we get to a graph with no viable vertices where we cannot proceed. With some thought, it is possible to show that for some kinds of graphs, surgery will always reduce the number of edges. Let us formalise these notions in the following.

Lemma 3.7.14 *Let G be a grid-labelled graph, and let G' be its completion. If for all non-empty subgraphs $H \subseteq G'$, H has at least one viable vertex then G is clearable.*

Proof. We first show that for all $H \subseteq G'$ with viable vertex set W_H, we have for all $w \in W_H$

(a) $H_w^R \subseteq G'$ and $H_w^C \subseteq G'$,

(b) $|E(H_w^R)| < |E(H)|$ and $|E(H_w^C)| < |E(H)|$.

That is, for any subgraph of the completion of G, performing surgery of either kind on any of its viable vertices leads to a graph which is also a subgraph of its completion, and that has strictly fewer edges than H.

Let G be as defined, and let $H \subseteq G'$ be non-empty. By assumption, the set W_H of viable vertices of H is non-empty. Consider performing row surgery on one such vertex $w \in W_H$. Since w is viable, the CUT step removes at least 1 edge from the graph. If the removal of these edges causes a connected component $C \subseteq G$ to become disconnected, then the STITCH step of the surgery comes into operation. The disconnected component is reconnected by adding one or more edges between vertices in $V(C)$. By definition, such edges belong to the completion of G. Hence, all edges in H_w^R are also edges of G' and so $H_w^R \subseteq G'$. The same is true for H_w^C, and so we have proved (a).

We can prove (b) by showing that the CUT step removes more edges from H than the STITCH step adds. Consider performing row (resp. column) surgery on a viable vertex, and let v be another vertex on that same row (resp. column), with e edges. The CUT step removes these e edges, splitting the connected component of v into sub-components $\{C_1, \dots C_k\} \cup \{v\}$ that are not connected to one another. Note that the number of such sub-components is upper bounded $k \leq e$: suppose removing the e edges of v caused the graph to split into $k' > e$ disconnected components. Then v would have had to be connected to k' vertices, a contradiction. The STITCH step requires that the union over $\{C_1, \dots, C_k\}$ be connected, which can always be achieved by adding $k - 1$ edges (recall that each component C_i is a connected graph by definition). Therefore, for each such vertex v, the CUT step has removed $d(v)$ edges and the STITCH step needs to add at most $d(v) - 1$ edges.

Since $G \subseteq G'$, we have by assumption that G has a non-empty set of viable vertices. From the above reasoning we know that for any of these vertices, performing either kind of surgery leads to a subgraph of G' with fewer edges, which

again by assumption has a non-empty set of viable vertices. Therefore, any sequence of surgeries starting from a viable vertex of G will iteratively decrease the number of edges in the graph until there are no edges left. ■
We can put what we just proved to good use in the proof of the following theorem.

Theorem 3.7.15 *The skew-mesh graph in $m \times n$ is bound entangled for all $m, n > 3$.*

Proof. By examining the structure of the graph from Figure 3.29, it is clear that the completion has the property that all subgraphs have viable vertices. Therefore, by Lemma 3.7.14, the graph is clearable for all grid sizes, which means that there are no product vectors in the range of the corresponding grid states. Therefore, all the states are entangled by application of the range criterion. By inspection, the graphs all satisfy the degree criterion, so they correspond to bound entangled states. ■
We have shown that the generalisations of B_4 and B_5 to higher grid dimensions also correspond to bound entangled states. This is encouraging for our "visual" approach to studying entanglement, it has enabled us to pick out generalisable patterns and use them to build families of bound entangled states.

3.8 Summary

In this chapter we have spent a great deal of time working with a family of quantum states we call *grid states*. We did this because each grid state can be represented by a special kind of graph we call a *grid-labelled graph*, that has as its vertices the points on a subsection of the Cartesian grid. The fact that these states have such a discrete mathematical representation means that we can understand certain things about them just by reasoning about and manipulating their grid-labelled graphs. The main idea that we have explored is that the entanglement properties of grid states can sometimes be understood using these considerations and manipulations of grid-labelled graphs. This is exciting because grid states correspond to *mixed states*, a class of quantum mechanical states that is difficult to understand the entanglement properties of. Perhaps our graphical techniques

can be put to good use in the future to provide a fresh perspective on mixed state entanglement. The main thrust of this chapter has been to develop the grid state setting as a toy model of mixed state entanglement, with a combinatorial handle attached – the grid-labelled graph – that may draw attention from discrete mathematicians not working in the field of quantum information. In a nutshell then, we hope to draw in fresh ideas and perspectives from other areas of mathematics in the hope that they will have something to say about the perplexing problem of mixed state entanglement. We do this by developing a theory of combinatorial entanglement which cuts out the "traditional" tools and settings from quantum information, to lower the barrier of entry to this area, with the hope that some questions about entanglement can possibly be recast as purely combinatorial questions.

Explicitly then, in this chapter we have considered how mixed state entanglement can be explored combinatorially via grid-labelled graphs. We surveyed existing literature that approaches entanglement from a graph theoretic perspective, and identified some issues in the foundations on which it is built. In particular, these papers focus on determining structural properties of traditional graphs that relate to entanglement. The issues arise because in order to relate graphs to entanglement, some assumptions need to be made about the subsystem structure and vertex labels. In particular, the major issue with the framework is that separability is not invariant under isomorphism.

We spent the chapter exploring how a minor modification to the starting point of this theory of "combinatorial entanglement" allows it to proceed in a more natural and physically better motivated direction. We consider a combinatorial object called a *grid-labelled graph*, a graph-like object restricted so that vertices are always on a two dimensional grid. Thinking about graphs in this way allow us to define a physically motivated notion of isomorphism, which we refer to as *local isomorphism*. We showed early on that two locally isomorphic grid labelled graphs have the same entanglement properties, in particular that one is separable if and only if the other is.

We've talked about how the Peres-Horodecki (Section 3.4), matrix realignment (Section 3.6) and range (Section 3.7) entanglement criteria can be applied to grid-labelled graphs, and their corresponding states (grid states). We've used

these "graphical entanglement criteria" to construct families of bound entangled states (Section 3.7.2), as well as entangled states that are not detected by the matrix realignment criterion (Section 3.6.2). We've also catalogued the 3×3 grid-labelled graphs that correspond to states that are positive under partial transposition.

In the next chapter, we will summarise all of the work in the thesis. In the latter half of this summary we will spend some time developing some ideas as to how this theory of combinatorial entanglement could be taken further.

Chapter 4

Discussion

This thesis has developed a two-threaded discussion of how combinatorial objects and concepts can inform topics in quantum information processing. In Chapter 2 we brought the graph isomorphism problem into the quantum realm with the STATEISOMORPHISM problem. This first thread starts with an observation that in quantum computational complexity, many of the problems considered are focused around quantum Hamiltonians. This is in stark contrast with the wide variety of computational problems that are studied in classical complexity theory. Of course, this disparity comes from the fact that quantum computers are a relatively recent object of study, and quantum complexity theory as a field even more so. The classical "problem collectors" have had a considerable head-start on their quantum counterparts. Nevertheless, this chapter was conducted with the idea that our collection of quantum problems could be boosted quite considerably if some ideas from the realm of graph theory could be generalised to the quantum realm.

We went about this by considering the problem of determining if two quantum states can be obtained from one another by rearranging their subsystems with permutations from a particular group, the STATEISOMORPHISM (SI) problem. We have seen that this problem is a generalisation of GRAPHISOMORPHISM (GI), explicitly deriving a Karp reduction from GI to SI in Theorem 2.3.10. Remarkably, GI can be reduced to a restricted form of SI, where the permutation group is the full symmetric group. We showed that in analogy to how GI can

be efficiently verified with a classical computer, instances of SI can be verified efficiently when equipped with a quantum verifier (Theorem 2.3.8: SI is in QCMA).

While the complement of the graph isomorphism problem GRAPHNONISOMORPHISM (GNI) is not thought to be in NP, it can be verified efficiently in some "weaker" settings in terms of probabilistic verifiers. Explicitly, GNI has a two message interactive proof system (Ref. [86]: GNI is in IP(2)), which can be made statistical zero knowledge (GNI is in SZK, see discussion in [85]), as well as public-coin (GNI is in AM). We were able to prove that the complement of the state isomorphism problem, STATENONISOMORPHISM (SNI), has similar properties. We think that it is unlikely that SNI belongs to QCMA for similar reasons to how GNI is unlikely to be in NP: there doesn't seem to be a better way of being certain that two states are not isomorphic than just comparing all permutations. However, similarly to GNI, we have been able to show that SNI has a two round quantum interactive proof system (Theorem 2.3.16: SNI is in QIP(2)), and that this interactive proof system can be made statistical zero knowledge (Theorem 2.3.18: SNI is in QSZK). Since QSZK is closed under complement [98], a corollary of this result is that SI is in QSZK.

Results pertaining to public-coin quantum interactive proof systems have been more challenging to obtain. While GNI is in AM, we haven't been able to prove the analogous quantum result that SNI is in QAM. As a consolation prize, we were able to prove that if the states under consideration can be efficiently described classically then the problem can be verified with a public coin quantum interactive proof system. Stabilizer states have this property, and we show in Theorem 2.3.13 that STABILIZERSTATENONISOMORPHISM (SSNI) is in QCAM. While this result shows that stabilizer states are relatively easy to handle in an interactive setting, the other end of the spectrum is mixed states, which seem to be substantially more difficult to work with. This is evidenced by Theorem 2.3.20, that MIXEDSTATEISOMORPHISM is QSZK-hard. Remarkably, even determining if two mixed states are close together in trace distance is QSZK-complete [98]. This latter result means that it's not even clear if MSI is in QCMA, verifying isomorphism would require us to solve this QSZK-complete problem.

It is not clear if SNI actually belongs to QAM, or we have been trying in vain to

achieve a result that doesn't hold. It could possibly be proved by adapting the set lower bound techniques used in Goldwasser-Sipser [100] to the quantum setting. Explicitly, these techniques would need to be generalised to the case where the sets are sets of quantum states rather than bitstrings. Let's take a brief detour to look at some ideas in this direction.

Recall that the set lower bound proof of GNI $\in$ AM (discussed in Section 2.2.2) involves the set $S(G) := \{\sigma(G) \ : \ \sigma \in \mathfrak{S}_{|V|}\}$ for a graph $G = (V, E)$. If G is isomorphic to a graph $H = (V, F)$ then assuming G and H have no non-trivial automorphisms, the union of these two 'isomorphism classes' $S(G) \cup S(H)$ has cardinality $n!$. Compare this to the case when they are not isomorphic, where the set must be of cardinality $2n!$. At first glance, it seems obvious that this notion can be used in the case where we are considering isomorphism of two n qubit quantum states $|\psi\rangle$ and $|\phi\rangle$. For instance, we could define the set $S(|\psi\rangle) := \{P_\sigma|\psi\rangle \ : \ \sigma \in \mathfrak{S}_n\}$, then consider how big the set $S(|\psi\rangle) \cup S(|\phi\rangle)$ is. An issue arises here stemming from the question of what it means for a set of quantum states to be "big". A tentative definition could be something along the following lines.

Definition 4.0.1 (ϵ-cardinality of a set of vectors) *Let $S = \{|\psi_1\rangle, \ldots, |\psi_k\rangle\} \subseteq \mathbb{C}^d$ be a set of vectors. For $0 \leq \epsilon \leq 1$, an* ϵ-subset *of S is a set of vectors $\{|\psi_{i_1}\rangle, \ldots, |\psi_{i_l}\} \subseteq S$ such that for all pairs $1 \leq m, n \leq l$ we have that $|\langle\psi_{i_m}|\psi_{i_n}\rangle| \leq \epsilon$. The* ϵ-cardinality *of S is the size of its largest ϵ-subset.*

Suppose we want to know if $|\psi\rangle$ and $|\phi\rangle$ are isomorphic or not. Explicitly, we want to distinguish between the case where there exists σ such that $|\langle\psi|P_\sigma|\phi\rangle| = 1$, and the case where for all σ, we have that $|\langle\psi|P_\sigma|\phi\rangle| \leq \epsilon$ for some ϵ. A nice assumption to make would be that the states $|\psi\rangle$ and $|\phi\rangle$ have no non-trivial automorphisms. That is, for all non-identity permutations σ, we have that $|\langle\psi|P_\sigma|\psi\rangle| \leq \epsilon$ and $|\langle\phi|P_\sigma|\phi\rangle| \leq \epsilon$. If it is reasonable to make this non-automorphism assumption about the states, then isomorphism of $|\psi\rangle$ and $|\phi\rangle$ would imply that the size of the largest ϵ-subset of $S(|\psi\rangle) \cup S(|\phi\rangle)$ is $n!$ and so the ϵ-cardinality is $n!$. Conversely, if the states were not isomorphic then there would be an ϵ-subset of $S(|\psi\rangle) \cup S(|\phi\rangle)$ size $2n!$.

Thus, verifying SNI becomes a question of lower bounding the ϵ-cardinality of $S(|\psi\rangle) \cup S(|\phi\rangle)$. Unfortunately, things get more difficult from this point onwards, as it is not clear how to do so! It might be possible to define some notion of quantum pairwise independent hash functions. Indeed, a weaker notion in this direction is 2-universal families of hash functions, of which unitary 2-designs are known to be quantum examples of [87]. This could be a fruitful approach in this direction.

A result that comes early on in the chapter, where we show that GI is a special case of SI (Theorem 2.3.10), immediately makes the problem more interesting: if one can find an efficient quantum algorithm that solves SI, then one can use it to efficiently solve GI. For a long time, researchers have suspected that GI has an efficient quantum algorithm, since it has similar complexity theoretic properties to the problem of factoring integers, FACTORING. Of course, FACTORING is the problem solved efficiently by Shor's algorithm. It was thought that since both FACTORING and GI are unlikely to be NP-complete as well as being difficult to put in P that they share some sort of structure, and that there must therefore exist some efficient quantum algorithm for GI too[1]. However, after receiving a great deal of attention in the early 2000's [66, 67], no efficient quantum algorithm has been found. While we don't consider it directly in this work, this algorithmic consideration was part of my original motivation for considering SI: perhaps focusing on permuting quantum states will lead to some breakthrough. At the least, we have learned more about graphs by considering how to determine if they are isomorphic (consider the deep links between abstract and linear algebra and graph theory, for example, considering automorphism groups [128], and graph spectra [130, 129]). Perhaps considering isomorphism of quantum states will lead to an injection of new mathematical ideas to quantum information in the same way.

Reading our discussion so far, it seems reasonable to ponder exactly what the quantum analogue of a graph is. The fact that graph and state isomorphism have such similar properties from a complexity point of view suggests that we might as well think of pure states as being a "quantum graph", and try to recast some more

[1]It turns out that FACTORING is the only other problem listed in Garey and Johnson that remains unclassified alongside GI.

classical problems in terms of pure states. One concrete example is a quantum generalisation of the NP-complete problem SUBGRAPHISOMORPHISM, regarding the *subsystems* of a quantum state. Explicitly, consider a problem where we are given two states, $|\psi_0\rangle$ on m qubits, and $|\psi_1\rangle$ on $n \leq m$ qubits. We want to know if there is some way of rearranging $|\psi_0\rangle$ such that its first n qubits are close in trace distance to $|\psi_1\rangle$.

Problem 4.0.2 SUBSTATEISOMORPHISM
Input: Efficient descriptions of quantum circuits Q_{ψ_0} in $\mathcal{Q}_{m,m}$, and Q_{ψ_1} in $\mathcal{Q}_{n,n}$ with $n \leq m$, and a function $\epsilon : \mathbb{N} \to [0, 1]$ such that $\epsilon(n) \geq 1/\text{poly}(n)$ for all n.

YES: There exists a permutation σ such that

$$D\left(\text{Tr}_{n,\ldots,m}\left[|\psi_0\rangle\langle\psi_0|\right], |\psi_1\rangle\langle\psi_1|\right) = 0.$$

NO: For all permutations σ

$$D\left(\text{Tr}_{n,\ldots,m}\left[|\psi_0\rangle\langle\psi_0|\right], |\psi_1\rangle\langle\psi_1|\right) \geq 1 - \epsilon(n).$$

This problem has the same feel as the QMA-complete problem QUANTUMCIRCUITSAT: given a quantum circuit U, find a pure state input $|\psi\rangle$ that makes it accept. We can cast this as a problem about subsystems of "post-execution" state $U|\psi\rangle$: in the case of acceptance, the first qubit will be close to $|1\rangle$. It's not clear if there is a reduction from this problem to SUBSTATEISOMORPHISM however, since there is no room in the definition to accommodate the unitary U. Furthermore, it may even be a challenge to put the problem in QMA: while the permutation can be specified efficiently, in full generality the state $\text{Tr}_{n,\ldots,m}\left[|\psi_0\rangle\langle\psi_0|\right]$ is mixed. Perhaps it is possible to modify the SWAP test to determine distance between this mixed state and the pure state $|\psi_1\rangle\langle\psi_1|$.
In general it seems like an interesting direction to attempt to find a QMA- or QCMA-complete problem that is about pure quantum states.
Moving away from questions about pure states, the second thread to the thesis starts in Chapter 3, where we consider a combinatorial theory of mixed state entanglement. The problem of determining if a mixed quantum state is entangled

or separable has attracted a great deal of attention over the years due to its link to foundations of quantum mechanics, and quantum technologies. It turns out to be a difficult and subtle problem, and the literature surrounding it employs a wide variety of mathematical techniques: hierarchies of positive semi-definite programs, results from operator theory and functional analysis, ideas from linear algebra and matrix theory, *etc.* [125, 110]. The focus of this chapter has been to come at this problem from a different angle. Instead of considering the separability problem for arbitrary density matrices, we consider a special family: the grid states. As we have outlined in great detail in the discourse, each grid state corresponds to a combinatorial object called a grid-labelled graph. From this graphical representation, we can sometimes determine if the underlying grid state is entangled or separable.

We have shown that several well known entanglement criteria, when applied to grid states, can be reinterpreted as structural criteria placed on their grid-labelled graphs. For example, in Section 3.4 we explore how the Peres-Horodecki criterion, recast as the degree criterion by Braunstein et al. [123, 122], can be applied to grid states. Determining if certain grid-labelled graphs are entangled in this way involves replacing each diagonal edge with its mirror image (the *partial transpose* of the graph), and checking if the vertex degrees are modified by this procedure. In Section 3.6 we show that the matrix realignment criterion can be applied to grid states by finding the eigenvalues of a pair of matrices that take into account degree and adjacency structure. The last criterion is the range criterion, which involves determining how many product vectors belong in the range of the grid state density matrix. In Section 3.7 we show how a procedure called graph surgery allows us to remove edges from a grid-labelled graph in such a way that the range remains the same. By repeatedly applying surgeries, sometimes we can simplify the graph under consideration to something trivial, and show that there are not enough product vectors in the range for the original grid state to be entangled.

While we have used traditional linear algebraic techniques in the derivation of the proofs involved in developing these structural entanglement criteria, there is minimal technical work required in applying them to grid-labelled graphs. In Section 3.7 for instance, we were able to apply the graphical range criterion we

developed along with some very basic structural lemmas to prove that the cross-hatch graphs and the skew-mesh graphs correspond to bound entangled states in all grid-dimensions. Once the groundwork has been laid, the proof that these families of states are entangled involves manipulating the grid-labelled graphs according to the recipe prescribed by the surgery, and observing that all sequences of surgeries lead to an empty graph. By considering the possible sequences of surgeries of a grid-labelled graph in this way, one can reason about entanglement in these grid states purely combinatorially. To me, this seems like a victory, because of the possibility of engaging researchers from outside of the quantum information community. Perhaps a combinatorial perspective on entanglement will allow new progress on old problems.

A concrete example of an issue that a combinatorial perspective may help with is in clearing up some loose ends in the complexity classification of the separability problem. In 2003, Gurvits [124] showed that the problem of determining if a given density matrix is separable is NP-hard. However, as pointed out in a comprehensive discussion by Ioannou [125] on the matter, the Gurvits proof seems to go through because some entangled density matrices can be exponentially close to being separable. In these cases, the difficulty essentially comes from determining which of the two cases holds, which of course requires exponential precision. Later, Gharibian [181] is able to improve the NP-hardness proof in a way that does not require this exponential precision.

Another issue with the NP-hardness of separability testing comes from the type of reduction used. The most common way of showing that a problem is NP-hard is to show that there exists a Karp reduction from some other NP-hard problem (*e.g.* 3-SAT) to that problem. A Karp reduction shows that every instance of the existing, "source" NP-hard problem can be transformed into an instance of the "target" problem, and that the transformation can be achieved in time that scales as a polynomial in the problem instance. However, the type of reduction used in the Gurvits proof is a weaker form of reduction called a *Turing reduction*. Loosely, when a problem is Turing reducible to some target problem then that problem is solvable in polynomial time if a solution to the target problem is known. While this weaker notion seems to make more practical sense, complexity theorists still consider Karp reductions as the gold standard of reductions, for better or for

worse. Ioannou [125] suggests that separability testing may be the first natural example of a problem that is NP-hard with respect to Turing reductions, but not Karp reductions. Note that even the improved NP-hardness result of Gharibian [181] still involves a Turing reduction.

I think that some progress could be made in this area using the grid state formalism because it translates the problem of determining if a state is separable into the language of complexity theory: discrete mathematical structures. The separability problem that we discuss in traditional quantum information, and as defined and studied in a complexity theoretic sense by by Gurvits, Ioannou and Gharibian is an inherently continuous problem. The objects are complex matrices, and this unavoidably introduces issues of precision. What if a state is *almost* separable? While this doesn't make much sense from a physical perspective, it certainly has caused issues in the complexity analysis as Ioannou has pointed out. On the other hand, a grid-labelled graph either corresponds to a separable state, or an entangled state. There is no ambiguity, and no need for precision considerations, because the object is discrete.

The problem of determining if a grid-labelled graph is separable is a genuine special case of the more general separability problem for arbitrary density matrices, since the grid states are just a special kind of density matrix. If the problem about grid-labelled graphs is NP-hard, then trivially, the more general problem about arbitrary states is NP-hard as well. Perhaps it is possible to reduce one of the many and varied NP-hard graph problems to the grid-state entanglement problem, thus clearing up the issues with the Gurvits proof we have outlined above.

Another area that may benefit from this combinatorial perspective is in the exploration and cataloguing of the bound entangled states. The bound entangled states are a mysterious class of states that contain entanglement, but in a form that is "locked away", not distillable into a pure form to be used in quantum protocols such as teleportation [182]. The neat graphical representation of the states we have considered in this thesis has been beneficial in identifying whole families of such states. To clarify, the bound entangled states we came across in Section 3.5 when we attempted to enumerate all of the 3×3 states that satisfy the degree criterion have very identifiable structures, reflected by the names we

chose to call them cross-hatch, and skew-mesh. The former can be generalised in the obvious way (a big cross-hatch!), and the latter with some work can be generalised too. These generalisable structures are made much more visible with the grid-labelled graph framework, which allowed us to immediately see how they would work in higher dimensions. A potential direction in this theme is to attempt a full characterisation of the PPT states in higher dimensions than 3×3. In this way, perhaps a catalogue of the bound entangled grid states is possible.

To cast these questions in language more familiar to quantum information theorists, when we talk about grid states, what we are really talking about is uniform mixtures of what I will now refer to as "Bell-ish" states: pure states of the form $(|i,j\rangle - |k,l\rangle)/\sqrt{2}$. The problem of entanglement in grid states is equivalent to the following

When is a uniform mixture of Bell-ish states separable?

As referred to previously, we can apply basic techniques from quantum information to show that the uniform mixture of $(|0,0\rangle - |1,1\rangle)\sqrt{2}$ and $(|0,1\rangle - |1,0\rangle)\sqrt{2}$ is separable. The grid-labelled graph framework gives us a means of visualising the mixtures we are considering: each edge represents a Bell-ish state, the full graph represents the uniform mixture of its edges. When we draw the grid-labelled graph corresponding to the uniform mixture of these two states, we obtain the criss-cross state. Applying the partial transpose, we see that the graph remains unchanged, and so the corresponding mixed state is separable. In fact, it's not just the criss-cross on a 2×2 grid: the criss-cross between any four vertices is locally isomorphic to a 2×2 criss-cross. We have identified a rule: two Bell-ish states that overlap in a criss-cross correspond to a uniform mixture that is separable. The main exercise in Chapter 3 has been an attempt to run with this idea of finding more general rules.

In Section 3.5 when we attempt to catalogue the 3×3 graphs that satisfy the degree criterion, we do not consider horizontal or vertical edges. Since the partial transpose leaves such edges unaffected, when considering the graphs that satisfy it, it does not make sense to include them. This motivates a simplification of the more general Bell-ish entanglement problem considered above: what if we just consider graphs that have *diagonal* Bell-ish states only? Remarkably, this doesn't

seem to make things easier: all of the building block graphs in Section 3.5 are built up using diagonal edges only, and two of these (B_4 and B_5) are bound entangled. In Section 3.7 we are able to build quite substantial families of bound entangled states in larger grid dimensions. In fact, it seems quite challenging to build a grid-labelled graph out of diagonal edges in such a way that the state is *separable*. It is possible, with minor set up, to state a formal conjecture based on this feeling that mixing diagonal Bell-ish states seems to always produce entangled states in all but a few cases.

The only families of separable diagonal-only graphs in 3×3 and above that we have been able to find in this research are those that are built out of generalisations of cross-hatch states and crosses. Consider the diagonal-graphs of size $2 \times q$, where each pair of vertices in each column has equal degree. Such graphs encompass the criss-cross graph, all unions of criss-crosses, any graph that looks like a tally mark, and so on (see Corollary 3.4.11). I conclude this thesis by conjecturing that the only separable diagonal-only grid-labelled graphs are those obtained by taking the union of these kinds of graphs. An example of such a decomposable graph is illustrated in Figure 4.1.

Conjecture 1 *A diagonal-only grid-labelled graph is separable if and only if it has a decomposition $X_1, \ldots, X_k$ such that for all $1 \leq i \leq k$, X_i has either two rows or two columns, and satisfies the degree criterion.*

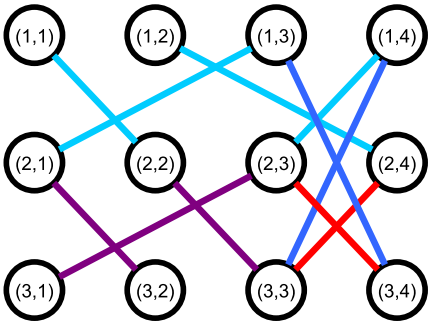


Figure 4.1: A grid-labelled graph, decomposed into $2 \times q$ or $q \times 2$ graphs that each satisfy the degree criterion.

Bibliography

[1] Xanadu – Overview on Crunchbase.com, https://www.crunchbase.com/organization/xanadu-2 (Retrieved 14/06/2018).

[2] S. Decker, C. Yasiejko, "Forget the Trade War. China Wants to Win Computing Arms Race." https://www.bloomberg.com/news/articles/2018-04-08/forget-the-trade-war-china-wants-to-win-the-computing-arms-race (Retrieved 14/06/2018).

[3] Microsoft Quantum Development Kit – Microsoft Visual Studio Marketplace, https://marketplace.visualstudio.com/items?itemName=quantum.DevKit (Retrieved 14/06/2018).

[4] R. P. Feynman, "Simulating physics with computers", *International Journal of Theoretical Physics* 21, 6-7, pp. 467-488 (1982).

[5] D. Deutsch, "Quantum theory, the Church-Turing principle and the universal quantum computer", *Proceedings of the Royal Society of London A* 400, pp. 97-117 (1985).

[6] D. Deutsch, R. Jozsa. "Rapid solutions of problems by quantum computation". *Proceedings of the Royal Society of London A.* 439 553. (1992)

[7] P. W. Shor, "Polynomial-Time Algorithms for Prime Factorization and Discrete Logarithms on a Quantum Computer", *SIAM Journal on Scientific and Statistical Computing* 26 1484 (1997).

[8] R. L. Rivest, A. Shamir, L. Adleman, "A method for obtaining digital signatures and public-key cryptosystems.", *Communications of the ACM* 21, 2, pp. 120-126 (1978).

[9] L. K. Grover, "A fast quantum mechanical algorithm for database search", *Proceedings of the 28th Annual ACM Symposium on the Theory of Computing (STOC)*, pp. 212-219 (1996).

[10] G. J. Milburn, "Photons as qubits", *Physica Scripta* 2009, T137, 014003 (2009).

[11] H. Haeffner, C.F. Roos, R. Blatt, "Quantum computing with trapped ions", *Physics Rep.* 469, pp. 155-203 (2008).

[12] B. E. Kane, "A silicon-based nuclear spin quantum computer", Nature 393, pp. 133-137 (1998).

[13] D. Gottesman, "An Introduction to Quantum Error Correction and Fault-Tolerant Quantum Computation", *arXiv:0904.2557 [quant-ph] (preprint)* (2009).

[14] P. W. Shor, "Scheme for reducing decoherence in quantum memory", *Physical Review A* 52, pp. 2493-2496 (1995).

[15] D. Gottesman, "Stabilizer codes and quantum error correction", *Caltech Ph.D. thesis*, quant-ph/9705052 (1997).

[16] H. Bombin, "Topological Codes", book chapter in *Quantum Error Correction*, edited by Daniel A. Lidar and Todd A. Brun, Cambridge University Press, New York (2013).

[17] A. Yu. Kitaev, "Fault-tolerant quantum computation by anyons", *Annals Physics* 303 pp. 2-30 (2003).

[18] A. Steane, "Multiple-Particle Inference and Quantum Error Correction", *Proc. Roy. Soc. Lond.* A. 452 pp. 2551-2577 (1996).

[19] "A Preview of Bristlecone, Google's New Quantum Processor" *Google AI Blog*, `https://ai.googleblog.com/2018/03/a-preview-of-bristlecone-googles-new.html` (Retrieved 09/06/2018).

[20] A. W. Harrow, A. Hassidim, S. Lloyd, "Quantum algorithm for solving linear systems of equations", *Physical Review Letters* 15, 103, pp. 150502 (2009).

[21] M. Mohseni, P. Read, H. Neven, S. Boixo, V. Denchev, R. Babbush, A. Fowler, V. Smelyanskiy, J. Martinis, "Commercialize quantum technologies in five years", *Nature* 543 pp. 171-174 (2017).

[22] R. Cellan-Jones, "Microsoft gambles on a quantum leap in computing", *BBC News* https://www.bbc.co.uk/news/technology-43580972 (Retrieved 09/06/2018).

[23] R. Chirgwin, "Alibaba fires up a cloudy quantum computer", *The Register* https://www.theregister.co.uk/2018/03/05/alibaba_quantum_cloud_computer/ (Retrieved 09/06/2018)

[24] N. Summers, "This is what a 50-qubit quantum computer looks like", *Engadget* https://www.engadget.com/2018/01/09/this-is-what-a-50-qubit-quantum-computer-looks-like/ (Retrieved 09/06/2018).

[25] M. Borak, After Alibaba, "Baidu leaps into quantum computing", *Technode*, https://technode.com/2018/03/08/baidu-quantum-computing/ (retrieved 09/06/2018)

[26] M. F. Riedel, D. Binosi, R. Thew, T. Calarco, "The European quantum technologies flagship programme", *Quantum Science and Technology*, 2, 3 (2017).

[27] S. Rich, B. Gellman, "NSA seeks to build quantum computer that could crack most types of encryption", *The Washington Post* Jan 2 2014, Retrieved from https://www.washingtonpost.com on 09/06/2018.

[28] "The Zettabyte Era: Trends and Analysis", *Cisco Whitepaper*, https://www.cisco.com/c/en/us/solutions/collateral/service-provider/visual-networking-index-vni/vni-hyperconnectivity-wp.html (Retrieved 09/06/2018).

[29] S. A. Cook, "The Complexity of Theorem-Proving Procedures", *Proceedings of the Third Annual ACM Symposium on the Theory of Computing (STOC)* (1971).

[30] R. M. Karp, "Reducibility among combinatorial problems", *Complexity of computer computations*, pp. 85-103 (1972).

[31] L. A. Levin, "Universal Sequential Search Problems", *Probl. Peredachi Inf.*, 9, 3, pp. 115-116 (1973); *Problems Inform. Transmission*, 9, 3, pp. 265-266 (1973).

[32] M. R. Garey, D. S. Johnson, "Computers and Intractability; A Guide to the Theory of NP-Completeness", W. H. Freeman & Co. New York, NY, USA (1990)

[33] "A compendium of NP optimization problems", edited by P. Crescenzi and V. Kann, `http://www.csc.kth.se/~viggo/wwwcompendium/` (Retrieved 14/06/2018).

[34] Greg Aloupis, Erik D. Demaine, Alan Guo, Giovanni Viglietta, "Classic Nintendo Games are (Computationally) Hard", *arXiv:1203.1895 (preprint)* (2012).

[35] E. Bernstein, U. Vazirani, "Quantum complexity theory", *SIAM Journal of Comput.* 26, 5, pp. 1411–1473 (1997).

[36] J. Watrous, "Succinct quantum proofs for properties of finite groups", *Proceedings of IEEE FOCS'2000*, pp. 537-546, (2000).

[37] A. Yu. Kitaev, A. Shen, M. N. Vyalyi, "Classical and Quantum Computation", *American Mathematical Society*, Boston, MA, USA (2002)

[38] J. Watrous, "Quantum Computational Complexity" *arXiv:0804.3401 (preprint)* (2008).

[39] A. D. Bookatz, "QMA-complete problems", *Quantum Information and Computation*, 14, 5-6 (2014).

[40] J. Kempe, A. Kitaev, O. Regev, "The Complexity of the Local Hamiltonian Problem", *SIAM Journal of Computing*, Vol. 35(5), pp. 1070-1097 (2006).

[41] J. Lockhart, C. E. González Guillén, "Quantum State Isomorphism" *arXiv:1709.09622 [quant-ph] (preprint)* (2017).

[42] J. Lockhart, S. Severini, "Combinatorial Entanglement", *arXiv:1605.03564 (preprint)* (2016).

[43] J. Lockhart, O. Gühne, S. Severini, "Entanglement properties of quantum grid states", *Physical Review A* 97, 062340 (2018).

[44] M. A. Nielsen, I. L. Chuang, "Quantum Computation and Quantum Information: 10th Anniversary Edition" *Cambridge University Press* (2011).

[45] N. David Mermin, "Quantum Computer Science: An Introduction", *Cambridge University Press* (2007).

[46] J. Watrous, "The Theory of Quantum Information", *Cambridge University Press* (2018).

[47] Ashley Montanaro, Ronald de Wolf, "A Survey of Quantum Property Testing", *Theory of Computing Graduate Surveys* 7 (2016).

[48] R. B. Boppana, J. Håstad, S. Zachos, "Does co-NP have short interactive proofs?", *Information Processing Letters* 25(2) pp. 126-132 (1987).

[49] U. Schöning, "Graph isomorphism is in the low hierarchy", *Journal of Computer and System Sciences*, 37, pp. 312-323 (1988).

[50] J. E. Hopcroft, J. K. Wong, "Linear time algorithm for isomorphism of planar graphs (Preliminary Report)", *Proceedings of the sixth annual ACM Symposium on Theory of Computing (STOC)*, pp. 172-184 (1974).

[51] L. Babai, P. Erdös, S. M. Selkow. "Random graph isomorphism", *SIAM Journal on Computing* 9, 3 pp. 628-635, (1980).

[52] T. Czajka and G. Pandurangan, "Improved random graph isomorphism", *Journal of Discrete Algorithms* 6(1) pp. 85-92 (2008).

[53] E. M. Luks, "Isomorphism of graphs of bounded valence can be tested in polynomial time", *Journal of Computer and System Science* 25(1) pp. 42–65 (1982).

[54] L. Babai and E. M. Luks, "Canonical labeling of graphs", *Proceedings of the fifteenth annual ACM symposium on Theory of computing*, pp. 171-183, (1983).

[55] Harald Helfgott, "Graph isomorphism in sub-exponential time", *The value of the variable (Personal Blog)* `https://valuevar.wordpress.com/2017/01/04/graph-isomorphism-in-subexponential-time/` (Retrieved 29/05/2018).

[56] B. D. McKay, "Practical Graph Isomorphism" *Congr. Numer.* 30, pp. 45-87, (1981).

[57] B.D. McKay, A. Piperno, "Practical Graph Isomorphism, II" *Journal of Symbolic Computation* 60, pp. 94-112 (2014).

[58] From Babai's personal home page `http://people.cs.uchicago.edu/~laci/update.html`.

[59] K. Baxter, J. Glasgow, "Protein structure determination: combining inexact graph matching and deformable templates", *Proc. Vision Interface* 2000 pp. 179-186 (2000).

[60] P. J. Hansen, P. C. Jurs, "Chemical applications of graph theory. Part I. Fundamentals and topological indices", *Journal of Chem. Educ.* 65, 7, pp. 574 (1988).

[61] A. Cobham, "The intrinsic computational difficulty of functions." *Proc. 1964 Congress for Logic, Methodology, and the Philosophy of Science.* (1964).

[62] J. Edmonds, "Paths, Trees, and Flowers", *Canad. Journal of Mathematics* 17 pp. 449-467 (1965).

[63] O. Goldreich , "Computational complexity: a conceptual perspective", *Cambridge University Press*, p. 128 (2008).

[64] S. Aaronson, "P $\stackrel{?}{=}$ NP", hosted `https://www.scottaaronson.com/papers/pnp.pdf` (Retrieved 11/05/2018).

[65] S. Goldwasser, S. Micali, C. Rackoff, "The knowledge complexity of interactive proof systems", *Proceedings of the seventeenth annual ACM symposium on Theory of computing (STOC)* pp. 291-304, (1985).

[66] C. Lomont, "The Hidden Subgroup Problem - Review and Open Problems", *arXiv:quant-ph/0411037 (preprint)* (2004)

[67] A. M. Childs, W. van Dam, "Quantum algorithms for algebraic problems", *Rev. Mod. Physics* 82 pp. 1-52 (2010).

[68] R. Ladner, "On the Structure of Polynomial Time Reducibility", *Journal of the ACM* 22, 1 pp. 155-171 (1975).

[69] R. J. Lipton, K. W. Regan, "Permutation Problems With Strings", *Blog post on Gödel's Lost Letter and* P = NP, `https://rjlipton.wordpress.com/2015/12/07/permutation-problems-with-strings/` (2015) (Retrieved 01/07/2018).

[70] D. Aharonov, T. Naveh, "Quantum NP – A Survey", *arXiv:quant-ph/0210077 (preprint)* (2002).

[71] L. Babai, "Monte-Carlo algorithms in graph isomorphism testing", *Université de Montréal Technical Report*, DMS:79-10 pp. 42 (1979).

[72] "Group-Theoretic Algorithms and Graph Isomorphism", *Lecture Notes in Computer Science* 136, Editor: C. M. Hoffmann (1982).

[73] C. C. Sims, "Computation with permutation groups", *Proceedings of the Second ACM Symposium on Symbolic and Algebraic Manipulation* pp. 23-28 (1971).

[74] M. Furst, J. Hopcroft, E. Luks, "Polynomial-time algorithms for permutation groups", *21st Annual Symposium on Foundations of Computer Science* (1980).

[75] E. M. Luks, "Permutation groups and polynomial-time computation". *Groups and Computation, DIMACS Series in Discrete Mathematics and Theoretical Computer Science* 11 pp. 139-175 (1993).

[76] H. Buhrman, R. Cleve, J. Watrous, R. de Wolf. "Quantum fingerprinting", *Physical Review Letters*, 87, 16, 167902 (2001).

[77] L. Babai, "Graph Isomorphism in Quasipolynomial Time" *arXiv:1512.03547 (preprint)* (2015).

[78] D. Gottesman, *talk at International Conference on Group Theoretic Methods in Physics*, *arXiv:quant-ph/9807006 (preprint)* (1998).

[79] R. Jain, Z. Ji, S. Upadhyay, J. Watrous, "QIP=PSPACE", *arXiv:0907.4737 (preprint)* (2009).

[80] M. Nielsen, I. Chuang, "Quantum Computation and Quantum Information", 10^{th} ed., *Cambridge University Press* (2011).

[81] J. Katz, Lecture notes on Complexity Theory, `http://www.cs.umd.edu/~jkatz/complexity/f11/lecture17.pdf` (Retrieved 01/06/2018).

[82] S. Aaronson, G. Kuperberg, "Quantum Versus Classical Proofs and Advice", *arXiv:quant-ph/0604056 (preprint)* (2006).

[83] B. Fefferman, S. Kimmel, "Quantum vs. Classical Proofs and Subset Verification", *arXiv:1510.06750 (preprint)* (2015).

[84] S. Aaronson, "NP-complete Problems and Physical Reality", *ACM SIGACT News*, March 2005.

[85] L. Trevisan, "Interactive and probabilistic proof-checking" *Annals of Pure and Applied Logic* 104 pp. 325-342 (2000).

[86] O. Goldreich, S. Micali, A. Wigderson, "Proofs that Yield Nothing But Their Validity, or, All Languages in NP Have Zero-Knowledge Proof Systems", *Journal of the Association of Computing Machinery* 38, 1, pp. 691-729 (1991).

[87] "Near-linear constructions of exact unitary 2-designs", *Quantum Information and Computation*, 16, 9 & 10, pp. 721-756 (2016).

[88] A. W. Harrow, "The Church of the Symmetric Subspace" *arXiv:1308.6595 (preprint)* (2013).

[89] M. Hein, W. Dür, J. Eisert, R. Raussendorf, M. Van den Nest, H.-J. Briegel "Entanglement in Graph States and its Applications", *Proceedings of the International School of Physics "Enrico Fermi": Quantum Computers, Algorithms and Chaos* pp. 115-218 (2006).

[90] S. Aaronson, "BQP and the Polynomial Hierarchy", *arXiv:0910.4698 (preprint)* (2009).

[91] G. Gutoski, P. Hayden, K. Milner, M. M. Wilde, "Quantum interactive proofs and the complexity of separability testing" *Theory of Computing*, 11(3), pp. 59-103 (2015).

[92] A. Montanaro, "Learning stabilizer states by Bell sampling", *arXiv:1707.04012 (preprint)* (2017).

[93] D. Gottesman, "Stabilizer Codes and Quantum Error Correction", *arXiv:quant-ph/9705052 (PhD thesis)* (1997).

[94] S. Aaronson, D. Gottesmann, "Improved simulation of stabilizer circuits" *Physical Review A* 70, 052328 (2004).

[95] H. J. Garcia, I. L. Markov, A. W. Cross, "Efficient Inner-product Algorithm for Stabilizer States", *arXiv:1210.6646 (preprint)* (2012).

[96] T Yamakami, "Quantum NP and a Quantum Hierarchy", *Proceedings of the 2nd IFIP International Conference on Theoretical Computer Science*, pp. 323-336 (2002).

[97] S. Gharibian, J. Kempe, "Hardness of approximation for quantum problems", *Quantum Information and Computation* 14 (5 and 6) pp. 517-540 (2014).

[98] J. Watrous, "Quantum statistical zero-knowledge", *arXiv:quant-ph/0202111 (preprint)* (2002).

[99] S. Aaronson, "Quantum versus classical proofs and advice", *arXiv:quant-ph/0604056 (preprint)* (2006).

[100] S. Goldwasser, M. Sipser, "Private coins versus public coins in interactive proof systems", *Proceedings of the Eighteenth Annual ACM Symposium on Theory of Computing (STOC)* pp. 59-68 (1986).

[101] H. Kobayashi, F. Le Gall, H. Nishimura, "Generalized Quantum Arthur-Merlin Games", *Proceedings of the 30th Conference on Computational Complexity (CCC2015)*, pp. 488-511 (2015).

[102] C. W. Helstrom, "Quantum Detection and Estimation Theory", *Academic Press*, New York, (1976).

[103] J. Watrous, "PSPACE has constant-round quantum interactive proof systems", *Proceedings of the 40th IEEE Conference on Foundations of Computer Science (FOCS)*, pp. 112-119 (1999).

[104] R. Jain, S. Upadhyay, J. Watrous, "Two-message quantum interactive proofs are in PSPACE" *Proceedings of the 50th IEEE Conference on Foundations of Computer Science (FOCS)*, pp. 534–543 (2009).

[105] C. Marriott, J. Watrous, "Quantum Arthur-Merlin Games", *arXiv:cs/0506068 (preprint)* (2005).

[106] A. Kitaev and J. Watrous. "Parallelization, amplification, and exponential time simulation of quantum interactive proof systems", *Proceedings of the 32nd ACM Symposium on Theory of Computing (STOC)*, pp. 608-617 (2000).

[107] A. Harrow, C. Y. Lin, A. Montanaro, "Sequential measurements, disturbance and property testing", *arXiv:1607.03236 (preprint)* (2016).

[108] S. Arora, B. Barak, "Computational Complexity: A Modern Approach" *Cambridge University Press* (2009).

[109] O. Goldreich, "Probabilistic proof systems", *Technical Report TR94-008, Electronic Colloquium on Computational Complexity* (1994).

[110] R. Horodecki, P. Horodecki, M. Horodecki, K. Horodecki, Quantum Entanglement, *Rev. Mod. Physics* 81 (2009).

[111] E. Chitambar, D. Leung, L. Mancinska, M. Ozols, A. Winter, "Everything You Always Wanted to Know About LOCC (But Were Afraid to Ask)", *Communications Mathematical Physics*, 328, 1, pp. 303-326 (2014).

[112] R. Jozsa, N. Linden, "On the role of entanglement in quantum-computational speed-up", *P. Roy. Soc. A-Mathematical Phy.* 459, 2036 (2003).

[113] G. Vidal, "Efficient classical simulation of slightly entangled quantum computations", *Physical Review Letters* 91, 147902 (2003).

[114] R. Duan, S. Severini, A. Winter, "Zero-error communication via quantum channels, non-commutative graphs and a quantum Lovasz theta function", *IEEE T. Inform. Theory* 59, 2 (2013).

[115] S. Beigi, P. W. Shor, "On the Complexity of Computing Zero-Error and Holevo Capacity of Quantum Channels", *arXiv:quant-ph/0709.2090 [] (preprint)* (2007).

[116] D. Leung, L. Mančinska, W. Matthews, M. Ozols, A. Roy, "Entanglement can increase asymptotic rates of zero-error classical communication over classical channels", *Communications Mathematical Physics* 311, 1 (2012).

[117] T. S. Cubitt, G. Smith, "An extreme form of superactivation for quantum zero-error capacities", *IEEE T. Inform. Theory* 58, 3 (2012).

[118] T. S. Cubitt, J. Chen, A. W. Harrow, "Superactivation of the Asymptotic Zero-Error Classical Capacity of a Quantum Channel", *IEEE T. Inform. Theory* 57, 12 (2011).

[119] R. Duan, "Super-activation of zero-error capacity of noisy quantum channels", *arXiv:0906.2527 (preprint)* (2009).

[120] J.F. Clauser, M.A. Horne, A. Shimony, R.A. Holt, "Proposed experiment to test local hidden-variable theories", *Physical Review Letters* 23, 15 (1969).

[121] Eric. W. Weisstein, "Algebraic Connectivity." From MathWorld–A Wolfram Web Resource. `http://mathworld.wolfram.com/AlgebraicConnectivity.html`

[122] S. L. Braunstein, S. Ghosh, S. Severini, "The laplacian of a graph as a density matrix: a basic combinatorial approach to separability of mixed states", *Ann. Combinatorics* 10, 3 (2006).

[123] S. L. Braunstein, S. Ghosh, T. Mansour, S. Severini, R. C. Wilson, "Some families of density matrices for which separability is easily tested", *Physical Review A* 73, 012320 (2006).

[124] L. Gurvits, "Classical deterministic complexity of Edmonds' problem and Quantum Entanglement", *arXiv:quant-ph/0303.055v1 (preprint)* (2003).

[125] L. M. Ioannou, "Computational complexity of the quantum separability problem", *Quantum Information and Computation* 7, 4 (2007).

[126] A. Peres, "Separability Criterion for Density Matrices", *Physical Review Letters* 77 (1996).

[127] M. Horodecki, P. Horodecki, R. Horodecki, "Separability of mixed states: necessary and sufficient conditions", *Physics Letters A* 223, 1-2 (1996).

[128] C. Godsil, G. F. Royle, "Algebraic Graph Theory", Volume 207 of Graduate Texts in Mathematics, Springer New York (2001).

[129] W. H. Haemers, E. R. van Dam, "Which graphs are determined by their spectrum?", *Linear Algebra and its Applications* 373, 241-272 (2003).

[130] A.E. Brouwer, W.H. Haemers, "Spectra of graphs", Springer, New York, etc. (2012).

[131] F. R. K. Chung, "Spectral Graph Theory", American Mathematical Society (1997)

[132] C. W. Wu, "Conditions for separability in generalized Laplacian matrices and diagonally dominant matrices as density matrices", *Physics Letters A* 351, 1-2 (2006).

[133] S. Dutta, B. Adhikari, S. Banerjee, R. Srikanth, "Bipartite separability and non-local quantum operations on graphs", *Physical Review A* 94, 012306 (2016).

[134] R. Hildebrand, S. Mancini, S. Severini, "Combinatorial laplacians and positivity under partial transpose", *Mathematical Struct. Comp. Sci.* Volume 18, Special Issue 1 (2008).

[135] Z. Wang, Z. Wang, "The tripartite separability of density matrices of graphs", *Electronic Journal of Combinatorics* 14, 2 (2007).

[136] C. W. Wu, "On graphs whose Laplacian matrix's multipartite separability is invariant under graph isomorphism", *Discrete Mathematics* 310, 21 (2010).

[137] C. W. Wu, "Graphs whose normalized Laplacian matrices are separable as density matrices in quantum mechanics", *Discrete Mathematics* 339, 4 (2016).

[138] H. Rahiminia, M. Amini, "Entangled Graphs", *arXiv:cs/0609156 (preprint)* (2006).

[139] B. Adhikari, S. Banerjee, S. Adhikari, A. Kumar, "Laplacian matrices of weighted digraphs represented as quantum states", *Quantum Inf. Process.* 16, 79 (2017).

[140] C. W. Wu, "Multipartite separability of Laplacian matrices of graphs", *Electronic Journal of Combinatorics* 16, 1 (2009).

[141] A. S. M. Hassan, P. S. Joag, "A combinatorial approach to multipartite quantum systems: basic formulation", *Journal of Physics A-Mathematical Theor,* 40, 33 (2007).

[142] A. S. M. Hassan, P. S. Joag, "On the degree conjecture for separability of multipartite quantum states", *Journal of Mathematical Physics* 49, 012105 (2008).

[143] C. Xie, H. Zhao, Z. Wang, "Separability of density matrices of graphs for multipartite systems", *Electronic Journal of Combinatorics* 20, 4 (2013).

[144] H. Rahiminia, M. Amini, "On separability of graphs with some entangled edges", *Quantum Inf. Process.* 8, 6 (2008).

[145] Z. Hui, F. Jiao, "Separability of Generalized Graph Product States", *Chinese Physics Letters* 30, 9 (2013).

[146] S. B. Adhikari, S. Banerjee, "Quantum discord of states arising from graphs", *Quantum Inf. Process.* 16, 183 (2017).

[147] S. Dutta, B. Adhikari, S. Banerjee, "Zero discord quantum states arising from weighted digraphs",*arXiv:1705.00808 [quant-ph] (preprint)* (2017).

[148] S. Dutta, B. Adhikari, S. Banerjee, "Seidel switching for weighted multi-digraphs and its quantum perspective", *arXiv:1608.07830 [math.CO] (preprint)* (2016).

[149] S. Dutta, B. Adhikari, S. Banerjee, "A graph theoretical approach to states and unitary operations", *Quantum Inf. Process.* 15, 5 (2016).

[150] J. Li, X. Chen, Y. Yang, "Quantum state representation based on combinatorial Laplacian matrix of star-relevant graph", *Quantum Information Processing* 14, 12 (2015).

[151] K. Chen, L. Wu, "A matrix realignment method for recognizing entanglement", *Quantum Information and Computation*, Vol. 3, No. 3 193-202 (2003).

[152] T. Ando, "Cones and norms in the tensor product of matrix spaces", *Linear Algebra Appl.* 379 (2004).

[153] P. Horodecki, "Separability Criterion and Inseparable Mixed States with Positive Partial Transposition", *Physics Letters A* 232, (1997).

[154] Nathaniel Johnston, QETLAB: A MATLAB toolbox for quantum entanglement, version 0.9. `http://www.qetlab.com`, (2016).

[155] T. E. Oliphant, "A guide to NumPy", *USA: Trelgol Publishing*, (2006).

[156] M. Horodecki, P. Horodecki, R. Horodecki, "Mixed-state entanglement and distillation: is there a "bound" entanglement in nature?" *Physical Review Letters* 80 pp. 5239 - 5242 (1998).

[157] C. H. Bennett, G. Brassard, S. Popescu, B. Schumacher, J. A. Smolin, W. K. Wootters, "Purification of Noisy Entanglement and Faithful Teleportation via Noisy Channels", *Physical Review Letters* 76 pp. 722 - 725 (1996).

[158] J. Lavoie, R. Kaltenbaek, M. Piani, K. J. Resch, "Experimental bound entanglement in a four-photon state", *Physical Review Letters* 105 130501 (2010)

[159] J. DiGuglielmo, A. Samblowski, B. Hage, C. Pineda, J. Eisert, R. Schnabel, "Experimental Unconditional Preparation and Detection of a Continuous Bound Entangled State of Light", *Physical Review Letters* 107, 240503 (2011).

[160] F. E. S. Steinhoff, M. C. de Oliveira, J. Sperling, W. Vogel, "Bipartite bound entanglement in continuous variables through degaussification", *Physical Review A* 89, 032313 (2014).

[161] E. Amselem, M. Sadiq, and M. Bourennane, "Experimental bound entanglement through a Pauli channel", *Sci. Rep.* 3, 1966 (2013).

[162] P. Horodecki, M. Horodecki, R. Horodecki, "Bound entanglement can be activated", *Physical Review Letters* 82 pp. 1056 - 1059 (1999).

[163] Ll. Masanes, "All entangled states are useful for information processing", *Physical Review Letters* 96, 150501 (2006).

[164] J. A. Smolin, "Four-party unlockable bound entangled state", *Physical Review A*, 63 032306 (2001).

[165] K. Horodecki, M. Horodecki, P. Horodecki, J. Oppenheim, "Secure key from bound entanglement", *Physical Review Letters* 94, 160502 (2005)

[166] P Hyllus, C. Moura Alves, D. Bruß, C. Macchiavello, "Generation and detection of bound entanglement", *Physical Review A* 70, 032316 (2004).

[167] P. Horodecki, J. A. Smolin, B. M. Terhal, A. V. Thapliyal, "Rank two bipartite bound entangled states do not exist", *Theoretical Computer Science* 292 pp 589-596 (2003).

[168] L. Pankowski, M. Piani, M. Horodecki, P. Horodecki, "A Few Steps More Towards NPT Bound Entanglement", *IEEE Transactions on Information Theory* 56 8 (2010).

[169] L. Chen, D. Ž. Đoković, "Distillability and PPT entanglement of low-rank quantum states", *Journal of Physics A: Mathematical Theor.* 44 285303 (2011).

[170] L. Chen, D. Ž. Đoković, "Corrigendum: Distillability and PPT entanglement of low-rank quantum states", *Journal of Physics A: Mathematical and Theoretical*, 45, 059501 (2012).

[171] D. Ž. Đoković, "The checkerboard family of entangled states of two qutrits", *Cent. Eur. Journal of Physics* 9(1) pp. 65 - 70 (2011)

[172] D. Bruß, A. Peres, "Construction of quantum states with bound entanglement", *Physical Review A* 61, 030301(R) (2000).

[173] K.-C. Ha, S.-H. Kye, "Construction of $3 \otimes 3$ entangled edge states with positive partial transposes", *Journal of Physics A: Mathematical and General* 38 9039 (2005).

[174] M. Lewenstein, B. Kraus, J. I. Cirac, P. Horodecki, "Optimization of entanglement witnesses", *Physical Review A*62 052310 (2000).

[175] D. P. DiVincenzo, T. Mor, P. W. Shor, J. A. Smolin, B. M. Terhal, "Unextendible Product Bases, Uncompletable Product Bases and Bound Entan-

glement", *Communications in Mathematical Physics*, 238, 3, pp. 379 - 410 (2003).

[176] C. H. Bennett, D. P. DiVincenzo, T. Mor, P. W. Shor, J. A. Smolin, and B. M. Terhal, "Unextendible Product Bases and Bound Entanglement", *Physical Review Letters* 82, 5385 (1999).

[177] M. Ozols, G. Smith, J. A. Smolin, "Bound Entangled States with a Private Key and their Classical Counterpart", *Physical Review Letters* 112, 110502 (2014).

[178] L. Clarisse, "Construction of bound entangled edge states with special ranks", *Physics Letters A* 359 pp. 603-607 (2006).

[179] W. C. Kim, Comment on: "Construction of bound entangled edge states with special ranks" [Physics Letters A 359 (2006) 603], *Physics Letters A* 372 pp. 2336 - 2338 (2008).

[180] N. Linden, S. Popescu, "Bound Entanglement and Teleportation", *Physical Review A* 59, 137 (1999).

[181] S. Gharibian, "Strong NP-Hardness of the Quantum Separability Problem", *Quantum Information and Computation* 10, No. 3&4, pp. 343-360 (2010)

[182] C. H. Bennett, G. Brassard, C. Crépeau, R. Jozsa, A. Peres, W. K. Wootters, "Teleporting an Unknown Quantum State via Dual Classical and Einstein–Podolsky–Rosen Channels", *Physical Review Letters* 70, pp. 1895-1899 (1993).

Appendix A

Grid States Miscellany

A.1 Entanglement of B_5

Let us use the range criterion to prove that $\rho(B_5)$ is entangled. This follows as a corollary of the following, the proof of which I credit to Otfried Gühne.

Proposition A.1.1 *The range of $\rho(B_5)$ is of dimension 5, and contains exactly 2 product vectors.*

Proof. We consider the range of the matrix

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

The matrix ρ is rank 5 (size of largest set of linearly independent rows is 5: see rows 1 through 5), so it has 5-dimensional range. We can write $\rho = \sum_{i=1}^{5} |\psi_i\rangle\langle\psi_i|$,

where

$$
\begin{aligned}
|\psi_1\rangle &:= |00\rangle - |11\rangle, \\
|\psi_2\rangle &:= |01\rangle - |12\rangle, \\
|\psi_3\rangle &:= |02\rangle - |20\rangle, \\
|\psi_4\rangle &:= |10\rangle - |21\rangle, \\
|\psi_5\rangle &:= |11\rangle - |22\rangle.
\end{aligned}
$$

It is easily verified that each of the following vectors are in the kernel of ρ

$$
\begin{aligned}
|\phi_1\rangle &:= |01\rangle + |12\rangle, \\
|\phi_2\rangle &:= |02\rangle + |20\rangle, \\
|\phi_3\rangle &:= |10\rangle + |21\rangle, \\
|\phi_4\rangle &:= |00\rangle + |11\rangle + |22\rangle.
\end{aligned}
$$

These vectors are mutually orthogonal, and since by rank-nullity the kernel is 4-dimensional, we know they span the kernel of ρ. Consider a product vector $|\alpha\rangle|\beta\rangle = (\alpha_0|0\rangle + \alpha_1|1\rangle + \alpha_2|2\rangle)\,(\beta_0|0\rangle + \beta_1|1\rangle + \beta_2|2\rangle)$ in the range of ρ. Since ρ is Hermitian, $|\alpha\rangle|\beta\rangle$ must be orthogonal to the kernel, which implies that

$$
\begin{aligned}
\alpha_0\beta_1 + \alpha_1\beta_2 &= 0, \\
\alpha_0\beta_2 + \alpha_2\beta_0 &= 0, \\
\alpha_1\beta_0 + \alpha_2\beta_1 &= 0, \\
\alpha_0\beta_0 + \alpha_1\beta_1 + \alpha_2\beta_2 &= 0.
\end{aligned}
$$

That is, for some $\alpha = (\alpha_0\ \alpha_1\ \alpha_2)$ we have that any $\beta^* = (\beta_0^*\ \beta_1^*\ \beta_2^*)$ must be orthogonal to each of

$$
\begin{pmatrix} \alpha_0 \\ \alpha_1 \\ \alpha_2 \end{pmatrix}, \begin{pmatrix} 0 \\ \alpha_0 \\ \alpha_1 \end{pmatrix}, \begin{pmatrix} \alpha_2 \\ 0 \\ \alpha_0 \end{pmatrix} \text{ and } \begin{pmatrix} \alpha_1 \\ \alpha_2 \\ 0 \end{pmatrix}.
$$

If such a β^* existed then these four vectors must only span a 2-dimensional subspace of $\mathbb{C}^3$ (there of course can only be three mutually orthogonal vectors in $\mathbb{C}^3$). Hence, any triple of these vectors must not be linearly independent, that is, any triple must contain a vector linearly dependent on one of the others. Considering one such triple, for this to be true then

$$\det \begin{pmatrix} 0 & \alpha_2 & \alpha_1 \\ \alpha_1 & 0 & \alpha_2 \\ \alpha_2 & \alpha_0 & 0 \end{pmatrix} = 0$$

implying that $(\alpha_0^2 + \alpha_2^2)\,\alpha_1 = 0$. We now consider cases.

CASE $\alpha_1 = 0$: this implies that

$$\det \begin{pmatrix} \alpha_0 & 0 & \alpha_2 \\ 0 & \alpha_0 & 0 \\ \alpha_2 & 0 & \alpha_0 \end{pmatrix} = 0$$

which implies that $\alpha_0 \left(\alpha_0^2 - \alpha_2^2\right) = 0$.

SUBCASE $\alpha_0 = 0$: this would imply that

$$\det \begin{pmatrix} 0 & \alpha_2 & 0 \\ 0 & 0 & \alpha_2 \\ \alpha_2 & 0 & 0 \end{pmatrix} = 0$$

implying that $\alpha_2 = 0$ and therefore $\alpha = 0$.

SUBCASE $\alpha_0 \neq 0$: without loss of generality we can normalise α such that $\alpha_0 = 1$, which implies that $\alpha_2^2 = 1$ which means that $\alpha_2 = \pm 1$. We have two possibilities now for orthogonal α, β pairs:

$$\alpha^A = \begin{pmatrix} 1 \\ 0 \\ 1 \end{pmatrix} \text{ and } \beta^A = \begin{pmatrix} 1 \\ 0 \\ -1 \end{pmatrix}$$

or

$$\alpha^B = \begin{pmatrix} 1 \\ 0 \\ -1 \end{pmatrix} \text{ and } \beta^B = \begin{pmatrix} 1 \\ 0 \\ 1 \end{pmatrix}.$$

CASE $\alpha_1 \neq 0$: this implies that $\alpha_0^2 + \alpha_2^2 = 0$.

SUBCASE $\alpha_0 = 0$: this cannot be true since then $\alpha_2 = 0$ and so

$$\det \begin{pmatrix} 0 & 0 & \alpha_1 \\ \alpha_1 & 0 & 0 \\ 0 & \alpha_1 & 0 \end{pmatrix} = 0$$

which means that $\alpha_1 = 0$.

SUBCASE $\alpha_0 \neq 0$: again, without loss of generality we can normalise so that $\alpha_0 = 1$ which implies that $\alpha_2 = \pm i$, meaning that

$$\det \begin{pmatrix} 1 & 0 & \pm i \\ \alpha_1 & 1 & 0 \\ \pm i & \alpha_1 & 1 \end{pmatrix} = 0.$$

This implies that $1 \pm i\alpha_1^2 - 1 = 0$, so $\alpha_1^2 = 0$ and we obtain a contradiction.

We conclude that there are only two product vectors in the range of ρ: $\alpha^A \otimes \beta^A$ and $\alpha^B \otimes \beta^B$. ■

A.2 Counting graphs that satisfy the degree criterion

In Section 3.5 we showed that the 3×3 grid-labelled graphs satisfy the degree criterion if and only if they are constructed from a small number of building-block graphs. Now we will outline a more general framework for enumeration of grid-labelled graphs of type $a \times b$, with k edges, that satisfy the degree criterion. In what follows, the quantity $P_k(a, b)$ is defined to be the number of graphs of type $a \times b$ with k edges that satisfy the degree criterion. Let $D_k(a, b)$ be the number of graphs of type $a \times b$ with k edges, all diagonal, that satisfy the degree criterion.

Definition A.2.1 (Rook's graph) *The* rook's graph *is the grid-labelled graph with an edge between every pair of vertices in the same row or column. It is well known that the $a \times b$ rook's graph has*

$$r(a,b) := \frac{a \cdot b(a+b)}{2} - a \cdot b$$

edges.

Lemma A.2.2 *For any $a, b, k \in \mathbb{N}$,*

$$P_k(a,b) = \binom{r(a,b)}{k} + \sum_{i=2}^{k} D_i(a,b) \cdot \binom{r(a,b)}{k-i}.$$

Proof. Let $G(a,b,d,h)$ be equal to the number of $a \times b$ grid-labelled graphs with d diagonal edges and h non-diagonal (horizontal or vertical) edges that satisfy the degree criterion. Clearly,

$$P_k(a,b) = \sum_{i=0}^{k} G(a,b,i,k-i).$$

Since the degree criterion is not affected by horizontal or vertical edges, and there are at most $r(a,b)$ horizontal and vertical edges in a $a \times b$ grid-labelled graph, it is clear that for any $i, j \geq 0$,

$$G(a,b,i,j) = D_i(a,b) \cdot \binom{r(a,b)}{j}.$$

Since for any a, b, $D_0(a,b) = 1$ and $D_1(a,b) = 0$, the lemma holds. ■
We obtain the first few values of $D_k(a,b)$ in the next statement.

Proposition A.2.3 *For any $a, b \geq 2$, the following are true:*

- $D_2(a,b) = \binom{a}{2} \cdot \binom{b}{2}$;
- $D_3(2,b) = 2 \cdot \binom{b}{3}$;
- $D_4(2,b) = 3 \cdot \binom{b}{3} + 9 \cdot \binom{b}{4}$.

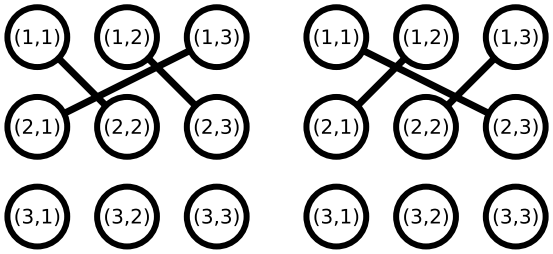


Figure A.1: The reflections of B_3 that are relevant for separability in $2 \times b$ grid-labelled graphs.

Proof. By Lemma 3.5.10, a 3×3 grid-labelled graph G with 2 diagonal edges satisfies the degree criterion if and only if it is locally isomorphic to B_2. Indeed, for such a graph in arbitrary grid dimensions $a \times b$ it must be the case that it satisfies the degree criterion if and only if it is LE-isomorphic to B_2. Therefore, we must count the number of $a \times b$ graphs with this property. For $a = 2$ this is of course $\binom{b}{2}$. Increasing a gives an additional $\binom{a}{2}$-dimensional degree of freedom, so

$$\begin{aligned} D_2(a,b) &= \binom{a}{2} \cdot D_2(2,b) \\ &= \binom{a}{2} \cdot \binom{b}{2}. \end{aligned}$$

By Lemma 3.5.12, we know that any 3×3 graph with 3 diagonal edges satisfies the degree criterion if and only if it is locally isomorphic to a rotation of B_3. Similarly to the 2 edge case, this generalises to arbitrary $a \times b$ grid-labelled graphs with 3 edges. In other words,an $a \times b$ graph with three edges satisfies the degree criterion if and only if it is LE-isomorphic to a rotation of B_3. Here, we only consider the $2 \times b$. Unlike B_2, the graph B_3 is not invariant under reflections. Indeed, there are two graphs that we must consider when counting the 3 edge degree criterion in $2 \times b$ grid-labelled graphs, which are illustrated in Figure A.1. By similar reasoning to the 2 edge case, $D_3(2,b) = 2 \cdot \binom{b}{3}$. In order to obtain

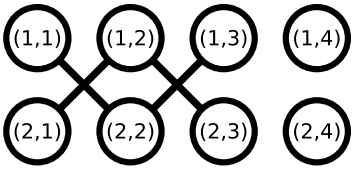


Figure A.2: An example of a 0, 1 and 2 degree graph of grid size 2×4 and with 4 edges.

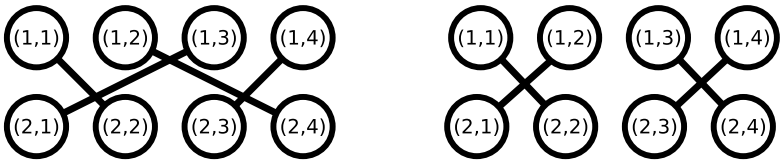


Figure A.3: Two examples of the 0 and 1 degree grid-labelled graphs with 4 edges on grid size 2×4. There is a bijection between such graphs and the derangements of a set of 4 elements.

$D_4(2, b)$, we will take a different approach. We know from Corollary 3.4.11 that a $2 \times b$ grid-labelled graph satisfies the degree criterion if and only if the degrees of the vertices in the top row are equal to the vertices in the bottom row. We leave it to the reader to verify that there are only two types of four edge grid-labelled graphs of type $2 \times b$ with this condition: those with all vertices of degree $0, 1$ or 2, and those with all vertices of degree 0 or 1. The former kind are all locally isomorphic to the grid-labelled graph illustrated in Figure A.2. Hence there are $3 \cdot \binom{b}{3}$ grid-labelled graphs of type $(2, b)$ with this pattern of degrees, because there are three columns of non-zero degree vertices: two with degree 1 vertices, one with degree 2. Therefore, there are $\binom{b}{3}$ placements of these columns, and 3 orderings within each placement.

We can enumerate the latter kind of grid-labelled graph by noticing that they are precisely the *derangements* of a set of 4 elements (consider the illustration in Figure A.3. Let us recall that a derangement of a finite set is a permutation without fixed points. The number of derangements on a set of n elements is counted by the *sub-factorial* function, defined

$$!n := (n-1)(!(n-1)+!(n-2)).$$

Hence, in this case, there are $!4 \cdot \binom{b}{4} = 9 \cdot \binom{b}{4}$ grid-labelled graphs under consid-

eration with this edge pattern. Therefore,

$$D_4(2,b) = 3 \cdot \binom{b}{3} + 9 \cdot \binom{b}{4}.$$

■

From what we have proved we are able to see the following.

Theorem A.2.4 (Entanglement of random grid-labelled graphs)

- *Let G be a random $a \times b$ grid-labelled graph with 2 edges. Then asymptotically almost surely (a.a.s), $G \notin \mathcal{S}$.*
- *Let G be a random $2 \times b$ grid-labelled graph with 3 (resp. 4) edges. Then a.a.s, $G \notin \mathcal{S}$.*

Proof. We know from Lemma 3.4.9 that we only need to consider the diagonal edges of a grid-labelled graph to test if it satisfies the degree criterion. Let us compare the growth of $D_k(a,b)$ as a function of grid dimension with that of $G_k^D(a,b)$, the total number of grid-labelled graphs with k edges, all diagonal.
Let us first obtain a general form for $G_k^D(a,b)$. Let $Q(a,b)$ be the number of diagonal edges in the $a \times b$ complete grid-labelled graph $K_{a\cdot b}$. Then

$$G_k^D(a,b) = \binom{Q(a,b)}{k}.$$

Clearly

$$\begin{aligned} Q(a,b) &= |E(K_{a\cdot b})| - r(a,b) \\ &= \frac{a\cdot b(a\cdot b-1)}{2} - \frac{a\cdot b(a+b)}{2} + a\cdot b \\ &= \frac{a^2\cdot b^2 - a\cdot b - a^2\cdot b - a\cdot b^2}{2} + a\cdot b \\ &= \frac{(a^2-a)(b^2-b)}{2} \\ &= 2\binom{a}{2}\cdot\binom{b}{2}, \end{aligned}$$

Hence,

$$G_k^D(a,b) = \binom{2 \cdot \binom{a}{2} \cdot \binom{b}{2}}{k},$$

and so $G_2^D(a,b) \sim (a^4 \cdot b^4)$. We know from Proposition A.2.3 that $D_2(a,b) = \binom{a}{2} \cdot \binom{b}{2} \sim (a^2 \cdot b^2)$. Therefore, given a random grid-labelled graph with grid dimensions $a, b \geq 2$ and with $k = 2$ edges, a.a.s. it will not satisfy the degree criterion and is therefore not separable.
Setting $a = 2$, we find that

$$\begin{aligned} G_3^D(2,b) &= \binom{2 \cdot \binom{b}{2}}{3} \\ &\sim (b^2)^3 \\ &= b^6, \end{aligned}$$

and

$$\begin{aligned} G_4^D(2,b) &= \binom{2\binom{b}{2}}{4} \\ &\sim (b^2)^4 \\ &= b^8. \end{aligned}$$

In comparison,

$$D_3(2,b) = 2 \cdot \binom{b}{3} \sim b^3,$$

and

$$D_4(2,b) = 3 \cdot \binom{b}{3} + 9 \cdot \binom{b}{4} \sim b^4.$$

We can conclude that for a random grid-labelled graph with grid dimensions $a = 2, b \geq 2$ and $k = 3, 4$ edges, a.a.s. it will not satisfy the degree criterion and is therefore not separable. ■
In order to find a general form for $D_k(a,b)$, more sophisticated techniques from

enumerative combinatorics will need to be employed. On the basis of what we have been able to prove so far however, it is safe to conjecture that for any grid-labelled graph G, a.a.s. $G \notin \mathcal{S}$.

A.3 Proof of $6, 7, 8, 9$ edge lemma

Let us now prove Lemma 3.5.18.

Proof. Let G be a 3×3 grid-labelled graph. We wish to prove that if G has $6 \leq m \leq 9$ diagonal edges, then it has a decomposition into building-block graphs.

That this is true is most easily seen by direct inspection. Through exhaustive computer search, all graphs with a set number of diagonal edges can be enumerated. There are a large number of graphs to be checked, especially in the 9 edge case. To make the lists smaller and easier to parse, we remove subgraphs locally isomorphic to the cross graph. Then, we remove all duplicate graphs from the list. In Figures A.4, A.5, A.6 and A.7 we show all 3×3 graphs that satisfy the degree criterion with $6, 7, 8$ and 9 diagonal edges respectively. It can be verified by examining these figures that each of the grid-labelled graphs have decompositions into grid-labelled graphs locally isomorphic to building-blocks or rotations of building-blocks. ■

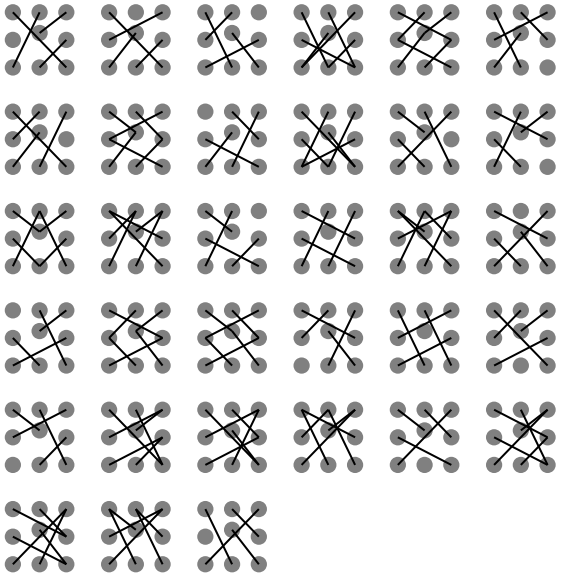


Figure A.4: All 6 edge grid-labelled graphs that satisfy the degree criterion up to local isomorphism, with cross subgraphs removed.

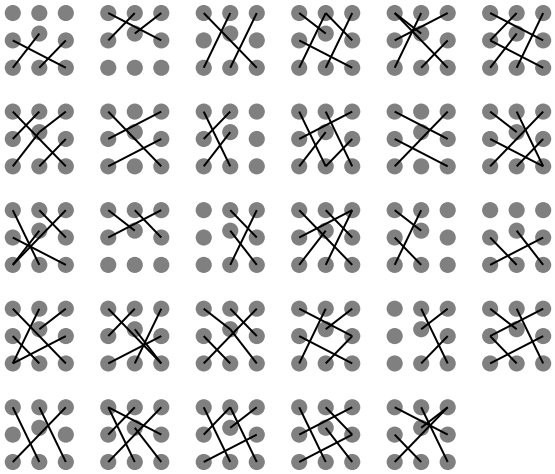


Figure A.5: All 7 edge grid-labelled graphs that satisfy the degree criterion up to local isomorphism, with cross subgraphs removed.

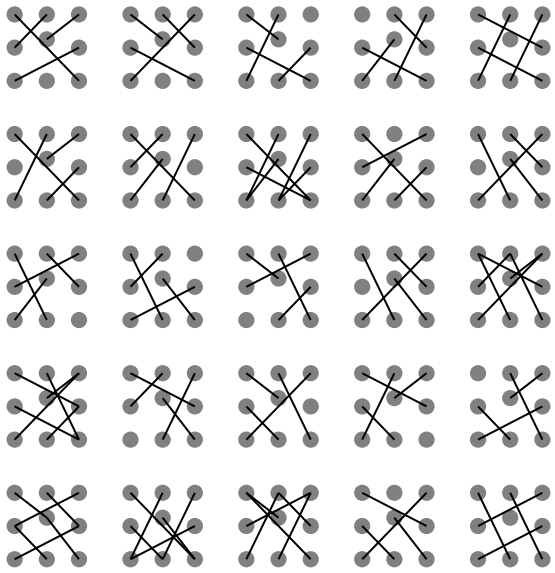


Figure A.6: All 8 edge grid-labelled graphs that satisfy the degree criterion up to local isomorphism, with cross subgraphs removed.

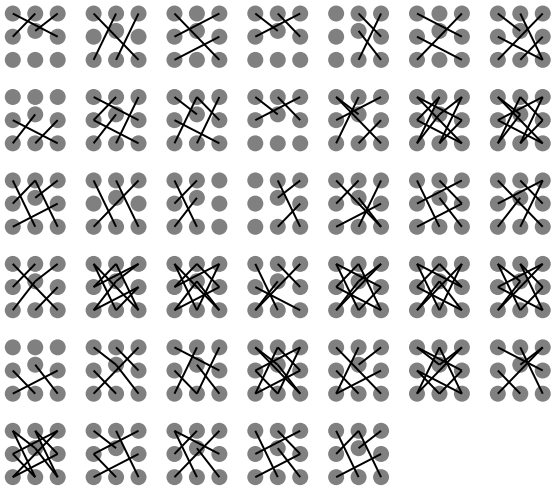


Figure A.7: All 9 edge grid-labelled graphs that satisfy the degree criterion up to local isomorphism, with cross subgraphs removed.