Euler Tours on the Grid

Sophia Jones

Master of Science

Computer Science

School of Informatics

University of Edinburgh

2018

Abstract

The problem of exactly counting the number of Euler tours on 4-regular graphs is known to be #P-complete [12]. Tetali and Vempala [28] attempted to prove that the number of Euler tours of 4-regular graphs could be approximated in polynomial time, using the Markov chain Monte Carlo Method [18], wherein they devised a Markov chain that used Kotzig moves [21]. While their proof was ultimately flawed, it drew attention to the question of whether the chain *could* be rapidly mixing.

In this dissertation, we present a proof that for a special case - 2 row, 4-regular toroidal grids - the chain mixes in polynomial time.

Acknowledgements

Thank you to my supervisor, Dr Mary Cryan, for her support, guidance, and feedback throughout the project, without which I would not be presenting a project I am so proud of.

Thank you to my wonderful pals Julius, Caro, Julia, Immy, and Gergő, and to my ace team of proof readers: Lewis, Alex, and my Ma. Thank you to my family for being my cheerleaders, and thank you to Alex for being the most amazing person ever, and for helping me write my acknowledgements.

Declaration

I declare that this thesis was composed by myself, that the work contained herein is my own except where explicitly stated otherwise in the text, and that this work has not been submitted for any other degree or professional qualification except as specified.

(*Sophia Jones*)

Table of Contents

Chapter 1

Introduction

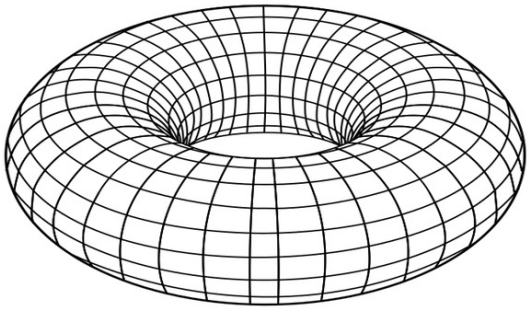

Figure 1.1: Square grid on a torus.

We begin by describing the motivations for this project, detailing the project's objectives and contributions. Finally, we give an outline of the structure of the dissertation.

1.1 Motivation

In this project we study Euler tours: a single cycle in a graph that traverses every edge once. We know by the "BEST"-theorem (de **B**ruijn, van Aardenne-**E**hrenfest, **S**mith, and **T**utte) that exactly counting the number of Euler tours on a directed graph is a simple problem to solve [32][29]. The theorem gives an expression for the number of Euler tours of a directed graph that can be calculated in polynomial time using the Matrix-Tree theorem [4]. However Brightwell and Winkler showed that for the case of undirected graphs, counting the number of Euler tours has shown to be a much harder problem [5]. While there are exceptions, such as the case of bounded treewidth graphs [6][7], for most classes of graphs it is highly unlikely that an exact counting

polynomial time algorithm will exist. In this project we focus our attention on Euler tours over a specific class of graphs - 4-regular grids on a torus. These are a special case of the general 4-regular graph, which have shown to be difficult to exactly count [12]. We focus our attention on *approximately* counting Euler tours on grids.

The task of approximately counting can, for certain problems, be reduced to the task of sampling uniformly at random from the set of solutions to the problem [18]. This is where we focus the efforts of this dissertation.

Tetali and Vempala [28] attempted to show that randomly sampling from the set of all Euler tours of a given 4-regular graph could be achieved in polynomial time. In their proof attempt, they devised a Markov chain $\mathfrak{M}$ which could be used with the Markov chain Monte Carlo method to sample from the set of all Euler tours of the given 4-regular graph. A crucial feature of the method is that the Markov chain must reach its equilibrium in polynomial time (Markov chains that possess this feature are called rapidly mixing). However, their proof was flawed and has so far resisted attempts to fix it. Rather than attempting to fix the proof ourselves, our objective was to show that there exist special cases - the $2 \times n$ grid and $3 \times n$ grid - on which their Markov chain *is* rapidly mixing. This would provide evidence that there may indeed exist a polynomial time algorithm for approximating the number of Euler tours on the general 4-regular toroidal grid. If we were unsuccessful in this pursuit, we had hoped to find evidence to show that their Markov chain was *not* rapidly mixing for these special cases, thus providing evidence that it may not be in general.

The approach we used employed "Kotzig moves" [21] to transition between Euler tours, and built upon the work by Creed [9] for exactly counting the number of Euler tours of the $2 \times n$ and $3 \times n$ grid.

1.2 Contributions

The main contribution of the project is a proof that the Markov chain $\mathfrak{M}$ is rapidly mixing for the $2 \times n$ grid. Furthermore, by approaching the problem in a novel way, we have provide new insight into how the structure of the toroidal grid may be exploited. Our secondary contributions are as follows:

- An introduction to the background material necessary to understand the work of this project;
- an explanation, with illustrative example, of the Markov chain Monte Carlo

Method;

- a review of the relevant literature, with annotations on how the the work has been utilised as part of our proof;
- a discussion of our method and its limitations.

1.3 Outline of Dissertation

The dissertation begins with a chapter detailing the necessary background material relevant to the project, including an introduction to graph theory, Markov chains, and an in-depth explanation of the tools that are used in the project. Chapter 3 introduces the Markov chain Monte Carlo method, with an illustrative example and an introduction to a technique that can be used for bounding the mixing time of the Markov chain. Chapter 4 gives a detailed explanation and discussion of Tetali and Vempala's work towards bounding the mixing time of $\mathfrak{M}$ for 4-regular graphs, with an explanation of its failings. We also explain Creed's transfer matrix method and his use of classes to exactly count the number of Euler tours on the $2 \times n$ and $3 \times n$ grids, with a description of how his work is utilised in our proof. Chapter 5 presents the main work of this dissertation: it begins by showing why we can use $\mathfrak{M}$ to approximately count the number of Euler tours on the $2 \times n$ grid, then defines some new notation to aid in the explanation of our approach, and gives an outline of our method. Finally, Section 5.2 gives a detailed description of our method and a proof that it works as stated, and Section 5.3 proves that the Markov chain is rapidly mixing. We conclude with Chapter 6, where we address our objectives and discuss the limitations of the project.

Chapter 2

Background

The purpose of this chapter is to familiarise the reader with the background material relating to the main methods and results of the project. Basic definitions will be introduced, and annotated with how they will be used in the project.[1]

2.1 Graph Theory

We begin with some graph theoretic definitions.

Definition 2.1. *A simple undirected graph $G = (V, E)$ is a non-empty set of vertices V and a set of edges $E \subseteq V \times V$ where an edge is an unordered pair of distinct vertices.*

Definition 2.2. *An Euler Tour is a cycle of a graph that traverses every edge exactly once. We write $ET(G)$ for the set of all Euler tours of a graph G.*

Definition 2.3. *The degree of some vertex v in graph G is the number of edges incident with v. If every vertex in G has degree n, then G is n-regular.*

Definition 2.4 (Creed [9])**.** *An $m \times n$ toroidal grid $G_{m,n}$ is a 4 regular graph with vertex set $\{(i, j) \ : \ 0 \leq i < m, \ 0 \leq j < n\}$ and edge set $\{[(i, j), (i', j)] : \ i' = i \pm 1 \ (mod \ m)\} \cup \{[(i, j), (i, j')] : \ j' = j \pm 1 \ (mod \ n)\}$*

Remark. *Suppose v_k is a vertex in the grid $G_{m,n}$. Let the notation v_k denote that vertex v is located in column k of the grid.*

In this project, we consider a special case of toroidal grids: $G_{2,n}$, and focus on counting the total number of Euler tours for $G_{2,n}$. Note that when we count the number

[1] The sections Graph theory (Section 2.1), Markov chains (Section 2.4) and Transition Systems (Section 2.5) have been adapted from the material in the "Background" chapter of my project proposal [19].

of Euler tours, certain tours are considered to be equivalent and so will not be included in the count.

Remark. *If Euler tour X is a rotation and/or reverse of some Euler tour Y then X and Y are equivalent, and will only be counted once.*

This notion of equivalence will be explained further when we introduce the transition system representation of an Euler tour in Section 2.5.

Rather than finding an algorithm to exactly count the number of Euler tours, we instead focus on *approximate* counting. The reason behind this choice is that we do not expect there to exist a polynomial time algorithm for exactly counting the number of Euler tours for any 4-regular graphs, of which toroidal grids are a special case. This is a result due to Ge and Stefankovic [12], who showed that the problem of counting the number of Euler Tours on 4-regular graphs is complete for the class #*P*, which will be defined in Section 2.2.

2.2 Complexity Class #*P*

The two most commonly known complexity classes are *P* and *NP*. Informally, the class *P* contains problems that can be computed by a Turing machine in polynomial time, while a problem belongs to *NP* if a certificate for the problem can be verified in polynomial time [2].

Definition 2.5 (Cook [8])**.** *A language L is in the class NP if and only if there exists a polynomial time Turing machine M (called the verifier) and a polynomial p such that*

1. *if $x \in L$ then there exists some y such that $|y| \leq p(|x|)$ and $M(x,y) = 1$; (completeness)*
2. *if $x \notin L$, then for all y such that $|y| \leq p(|x|)$, it holds that $M(x,y) = 0$. (soundness)*

The class #*P* classifies problems where we wish to determine the *number* of certificates for a certain problem, rather than simply whether or not there is a certificate.

Definition 2.6 (Valiant [30])**.** *A function $f : \{0,1\}^* \to \mathbb{N}$ belongs to the complexity class #P if there exists some polynomial $p : \mathbb{N} \to \mathbb{N}$ and a polynomial time Turing machine M such that for every $x \in \{0,1\}^*$, it holds that*

$$f(x) = |\{y \in \{0,1\}^{p(|x|)} : M(x,y) = 1\}|.$$

Since the question "is there a certificate for a certain problem P?" can be interpreted as "are there zero certificates for P?", the class NP clearly belongs to $\#P$ [2]. The class was first introduced in Valiant's paper on calculating the permanent of a matrix [31]. In a follow up paper by Valiant [30], many counting problems were shown to belong to $\#P$.

Roughly speaking, a problem is *complete* for a complexity class if it belongs to that class, and is at least as hard as every other problem in the class. In the case of the class $\#P$, the notion of being "at least as hard" is formalised in the following definition.

Definition 2.7 (Valiant [30])**.** *A function f is $\#P$-hard if any function $g \in \#P$ can be computed in polynomial time using a Turing machine M with f as an oracle.*

A function f is $\#P$-complete if it is $\#P$-hard and $f \in \#P$. The most natural example of a $\#P$-complete problem is $\#SAT$, the problem of counting the number of satisfying assignments to some CNF formula [14].

The problem of simply deciding whether some CNF formula has a satisfying assignment is known as SAT, and is complete for the class NP. Suppose there is a polynomial time algorithm to solve $\#SAT$. Then we could decide SAT by outputting "yes" (1) if the number of satisfying assignments is non-zero, and output "no" (0) otherwise. Then we could decide SAT in polynomial time, so $SAT \in P$ and since SAT is NP-complete, we could deduce that $P = NP$. Therefore it is very unlikely that any of the $\#P$-hard problems can be solved in polynomial time.

As noted earlier, the problem of counting the number of Euler tours on 4-regular graphs is also $\#P$ complete [12], and so it is unlikely that there exists a polynomial time algorithm for exactly counting the number of Euler tours on a 4-regular graph.

Note that interestingly, it is true for many $\#P$-complete problems that the corresponding decision problem can be solved in polynomial time. The problem of determining whether a graph contains an Euler tour (and is therefore Eulerian) can be solved by a simple breadth first search algorithm, due to the following theorem.

Theorem 1 (Euler)**.** *A connected graph is Eulerian if and only if its vertices all have even degree.*

A breadth first search algorithm can be performed in polynomial time $O(m+n)$, where m is the number of edges and n is the number of vertices. It is worth noting here that since the toroidal grids we are considering are 4-regular, each vertex will indeed have even degree, and so toroidal grids are Eulerian.

Since the problem of exactly counting the number of Euler tours on 4-regular graphs is #*P*-complete, it is highly unlikely that there will exist a polynomial time algorithm to compute this. We consider instead the problem of *approximate* counting in polynomial time. To discuss approximation algorithms, we require a definition of what constitutes a good approximation.

2.3 FPRAS

The definition of a fully polynomial randomised approximation scheme (FPRAS) captures the notion that the approximation should be achieved in polynomial time, and should be a "good" approximation.

Definition 2.8 (Karp and Luby [20])**.** *Suppose we have some function* $f : \Sigma^* \to \mathbb{N}$ *mapping problem instances to natural numbers. For example,* Σ^* *could be* $G_{m,n}$ *grids where* f *maps each grid to its total number of Euler tours.*

A randomised algorithm approximating f *is a* ***fully polynomial randomised approximation scheme (FPRAS)*** *if it takes as input* $x \in \Sigma^n$ *and* $\varepsilon > 0$ *and outputs a random variable* Y *satisfying*

$$Pr((1-\varepsilon)f(x) \leq Y \leq (1+\varepsilon)f(x)) \geq \frac{3}{4} \tag{2.1}$$

and furthermore, the algorithm runs in time polynomial in n *and* ε^{-1}.

Note that the $\frac{3}{4}$ in this definition is arbitrary, and could be any $\frac{1}{2} < c < 1$. Jerrum,Valiant and Vazirani [18] showed that c could be amplified to $1-\delta$ where $\delta > 0$ by performing trials and taking the median of the results.

In the same paper, they also proved a result that is fundamental to the work in this dissertation. Jerrum, Valiant and Vazirani [18] showed that for certain problems, the existence of a FPRAS to approximate the number of solutions to a problem is equivalent to randomly sampling from the set of all possible solutions of the problem in polynomial time.

By "certain problems", we mean problems which exhibit the **self-reducibility** property, which we will explain using the example of *SAT*. *SAT* is complete for the complexity class *NP*, which has an alternative definition to the one given above.

Definition 2.9 (Arora and Barak [2])**.** $NP = \cup_{c\in\mathbb{N}} NTime[n^c]$ *where* $NTime[n^c]$ *is the set of all languages that can be computed by a non-deterministic Turing machine* M *within time* n^c.

Suppose that we had a "black box" which could tell us on demand if a CNF-formula ϕ was satisfiable. Call this a *SAT*-oracle. Then we could use this to find a satisfying assignment for ϕ.

First ask the oracle if ϕ is satisfiable. If so, identify the first variable x_1 and set its value as $x_1 = 1$ or $x_1 = 0$. In both cases, the effect of setting x_1 can be written out again as a (slightly different) CNF formula (note that these CNF formulas are disjoint).Then we ask the oracle if either of these new CNF formulas are satisfiable. Continue in this way through all the variables until a satisfying assignment for ϕ is found. This notion that *SAT* can be solved by using a *SAT*-oracle to solve smaller instances of the problem is the self-reducibility property [14].

In Section 5.1.1, we will show that the problem of counting Euler tours is indeed self-reducible. In order to show that we can approximately count the number of Euler tours on $G_{2,n}$ in polynomial time, we must therefore show that we can sample from $G_{2,n}$ in polynomial time. The reduction of sampling to counting is achieved via a *telescoping product*, like others found in the literature.

To randomly sample from a set of all Euler tours on grids, a technique called the Markov Chain Monte Carlo (MCMC) technique can be used. It will be explained in detail in Chapter 3. To describe MCMC, first some definitions of Markov chains will be necessary.

2.4 Markov Chains

Let the state space Ω be the collection of all possible states of the combinatorial structures of interest (for us, these are Euler tours of the given graph). A Markov chain allows us to model how we can transition between these states. Suppose that the probability of transitioning from some state $X \in \Omega$ to another state $Y \in \Omega$ is conditional *only* on X, and not the chain of events that led to X. Then the process modelling the transitions is *memoryless*.

A Markov chain is a memoryless stochastic model describing the state space Ω. This is formalised by the following definition.

Definition 2.10 (Cryan [10])**.** *A discrete-time stochastic process on the state space* Ω *is a Markov chain if*

$$Pr[X_t = a_t | X_{t-1} = a_{t-1}, ..., X_0 = a_0] = Pr[X_t = a_t | X_{t-1} = a_{t-1}].$$

for all $a_0, ..., a_k \in \Omega$.

Definition 2.11 (Cryan [10])**.** *A stationary distribution for a Markov Chain $\mathfrak{M}$ with state space Ω is a probability distribution over Ω, denoted π, such that $\pi = \pi \cdot \mathfrak{M}$.*

Definition 2.12 (Cryan [10])**.** *A Markov chain $\mathfrak{M}$ is irreducible if for all $X, Y \in \Omega$ there is some $n \in \mathbb{N}$ such that $\mathfrak{M}^n[X,Y] > 0$, i.e. all states are accessible to all other states.*

Definition 2.13 (Cryan [10])**.** *A Markov chain $\mathfrak{M}$ is aperiodic if for every $Y \in \Omega$, the gcd of $\{k : \mathfrak{M}^k[Y,Y] > 0\}$ is 1.*

If a finite Markov chain is both irreducible and aperiodic, then it is ergodic. Ergodicity is a useful property because an ergodic Markov Chain has a unique stationary distribution. Certain Markov Chains have the uniform distribution as their stationary distribution. This property will be useful to us later in the dissertation.

Definition 2.14. *A Markov Chain is symmetric if $P(X,Y) = P(Y,X) \quad \forall X, Y \in \Omega$.*

Lemma 2.1. *An ergodic and symmetric Markov chain has the uniform stationary distribution.*

Proof. We wish to show that the uniform distribution (the vector with $\frac{1}{|\Omega|}$ in every entry) is the stationary distribution. Let π be the vector with $\frac{1}{|\Omega|}$ in every entry. Then we need to show that π satisfies $\pi \cdot \mathfrak{M} = \pi$. Consider $\pi \cdot \mathfrak{M}$ for some state X.

$$\pi(X)\mathfrak{M}(X) = \sum_Y \pi(Y)P(Y,X) \tag{2.2}$$

$$= \sum_Y \pi(Y)P(X,Y) \tag{2.3}$$

$$= \sum_Y \frac{1}{|\Omega|}P(X,Y) \tag{2.4}$$

$$= \frac{1}{|\Omega|} \cdot 1 = \pi(X). \tag{2.5}$$

Where Equation 2.3 follows from $\mathfrak{M}$ being symmetric [24]. □

Later in the dissertation we will make use of this lemma when we want to show that a Markov chain has the uniform stationary distribution. We will use our definitions relating to Markov chains to describe the Markov chain Monte Carlo method in Chapter 3. Finally, we will describe some concepts that will aid in our discussion of Euler tours.

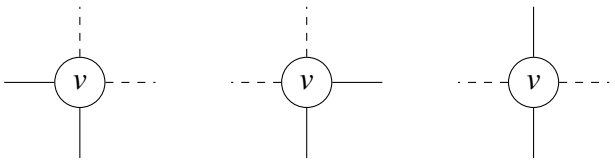

Figure 2.1: Three possibilities for the transition system at a degree 4 vertex v

2.5 Transition Systems

Transition systems provide an alternative way of thinking about a tour of a graph.

Definition 2.15 (Creed [9])**.** *Suppose we have some graph G with vertex set V and edge set E. A transition system at some vertex $v \in V$ is a pairing of the edges incident with v. A transition system on the full graph G is a function that maps each $v \in V$ to a transition system on v. We label the union of the transition systems on the vertices of G as TS_G.*

Consider a transition system on a 4-regular grid. For any single vertex, there are 3 possible ways for its incident edges to be paired. These are displayed in Figure 2.1. Since the grid is 4-regular, the vertices all have even degree. Therefore if a path uses an edge e to enter some vertex v, the transition system at v must pair e with some other edge incident with v, and so v cannot be the end point of any path. This implies that the union of the transition systems on all the vertices TS_G may only form disjoint cycles, and not paths.

Therefore a transition system on a 4-regular grid will partition it into a set of disjoint cycles. If the transition system only creates one cycle, then this cycle is an Euler tour of the graph. However, there will be graph transition systems which correspond to multi-cycle decompositions.

Throughout the paper, we will use the notation $e(v)e'$ to indicate that a trail which traverses edge e to reach vertex v will next traverse edge e'. This will be illustrated by an example tour on the grid $G_{2,3}$ shown in Figure 2.2. One potential Euler tour of $G_{2,3}$ is:

$$1(v)4(y)6(z)7(w)8(z)9(x)11(u)12(x)2(y)3(v)5(w)10(u). \tag{2.6}$$

Note that this condition on entering v along edge 1 places a direction on the Euler tour of $G_{2,3}$; however when counting the number of Euler tours on a grid, we will count an Euler tour and its reverse only once.

Now we provide a definition that allows us to move from one TS_G to another TS_G. To make this clearer, first we develop our transition system notation.

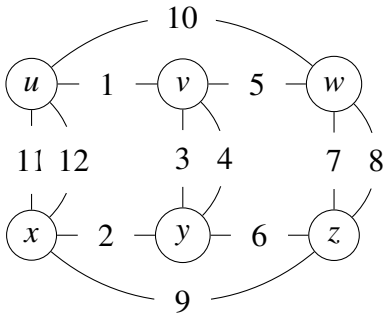

Figure 2.2: Toroidal 2×3 grid

Remark. *Let $TS_G(X)$ denote the union of the transition systems on the vertices of G according to Euler tour X of G.*

Remark. *Let $TS_G(X,v)$ denote the transition system at vertex v according to the Euler tour X of graph G.*

We now present some helpful results due to Kotzig, who considered Euler tours in 1968, defined natural transformations on Euler tours, and went on to show these transformations would connect the entire state space.

Definition 2.16 (Kotzig [21])**.** *Suppose we have some Euler tour X on a graph G. Suppose further that $TS_G(X,v)$ is arranged with $e(v)e'$ and $f(v)f'$ (and so vertex v has degree at least 4). Suppose that the expanded form of X is $\ldots e(v)e'\ldots f(v)f'\ldots$*

Let S be the segment of X starting at e' and ending at f. The Euler tour X' is obtained by a **κ-transformation** *at v if X' is obtained from X by "changing the direction of travel" for each edge of S. Then X' takes the form $\ldots e(v)f\ldots e'(v)f'\ldots$*

Note that after a κ-transformation is performed, the transition systems of all vertices besides v will have stayed the same (besides their "direction"). However $TS_G(X',v)$ now pairs e with f.

Similar to [21], we will also need to define allowed and prohibited transitions.

Definition 2.17 (Kotzig [21])**.** *Suppose we have two Euler tours X and Y on a graph G, where vertex v is respectively given transition systems $TS_G(X,v)$ and $TS_G(Y,v)$ in the tours. Suppose a transition system on the whole graph TS_G is formed so that all vertices except v are given their transition systems as allocated in X, while v is given the transition system $TS_G(Y,v)$. If the resulting TS_G is an Euler tour, then $TS_G(Y,v)$ is an* **allowed** *transition for X at v, denoted $TS_G(X) \Rightarrow_v TS_G(Y)$. However if the resulting*

*TS_G is not an Euler tour, then $TS_G(Y,v)$ is a **prohibited** transition for X at v, denoted $TS_G(X) \not\Rightarrow {}_vTS_G(Y)$.*

Recall that for any vertex on a 4-regular grid G, there are exactly 3 possible transition systems, shown in Figure 2.1. Also recall that for any Euler tour X, every vertex is allocated one transition system, in such a way that the union of the transition systems of all the vertices forms an Euler tour.

Remark. *Suppose in Euler tour X vertex v has transition system $TS_G(X,v)$. Of the two alternative transition systems at v, one transition will be allowed and one transition will be prohibited.*

We illustrate allowed and prohibited transitions with an example, taken from my IPP [19]. Consider again the Euler tour of $G_{2,3}$ shown in Equation 2.6 and repeated below. Label this Euler tour X.

$$1(v)4(y)6(z)7(w)8(z)9(x)11(u)12(x)2(y)3(v)5(w)10(u) \tag{2.7}$$

Consider another Euler tour of $G_{2,3}$ shown in Equation 2.8 which we label Y:

$$1(v)5(w)8(z)7(w)10(u)11(x)9(z)6(y)4(v)3(y)2(x)12(u). \tag{2.8}$$

Suppose we want to keep the transition systems of all the vertices besides v the same as they are in X, but give v its transition system in Y (pairing 1 with 5 and 4 with 3). This would be a prohibited transition at v, since the resulting transition system would partition the grid into two disjoint cycles:

$$1(v)5(w)10(u); \quad 4(v)3(y)2(x)12(u)11(x)9(z)8(w)7(z)6(y).$$

Suppose instead we wanted to keep the transition systems of all the vertices besides u the same as they are in X, but give u its transition system in Y (pairing 11 with 10 and 12 with 1). This transition is allowed, since the resulting transition system is an Euler tour:

$$1(v)4(y)6(z)7(w)8(z)9(x)11(u)10(w)5(v)3(y)2(x)12(u). \tag{2.9}$$

Then we can write $TS_{G_{2,3}}(X) \not\Rightarrow {}_vTS_{G_{2,3}}(Y)$ and $TS_{G_{2,3}}(X) \Rightarrow_u TS_{G_{2,3}}(Y)$.

Although there does not exist a κ-transformation that allows us to move $TS_{G_{2,3}}(X,v)$ to $TS_{G_{2,3}}(Y,v)$ while keeping the transition systems of the other vertices the same, we *can* change the transition system of some vertex $h(\neq v)$ first, which would change $TS_{G_{2,3}}$ in such a way that the transition of v in Y is now available.

For example, consider the Euler tour in Equation 2.9. All vertices besides u have their transition systems in X, while u has its transition system in Y. Label this Euler tour X'. Notice that for X', if we keep the transition systems of all vertices except v the same and give v its transition system in Y, then this will also result in an Euler tour:

$$1(v)5(w)10(u)11(x)9(z)8(w)7(z)6(y)4(v)3(y)2(x)12(u). \tag{2.10}$$

So although we had $TS_{G_{2,3}}(X) \not\Rightarrow_v TS_{G_{2,3}}(Y)$, we have $TS_{G_{2,3}}(X') \Rightarrow_v TS_{G_{2,3}}(Y)$. We call u a **helper vertex** for v, since performing a κ-transformation at u made the transition we wanted at v allowed.

Not all vertices could have been used as a helper vertex for v. In order to swap the allowed and prohibited transitions at v, the helper vertex h must **interleave** with v, as defined below.

Definition 2.18. *Two vertices v and h interleave in a tour X if visits to vertex v are alternating with visits to vertex h in the expanded representation of X.*

This knowledge that interleaving vertices u and v can be used to swap the allowed and prohibited transitions at each vertex will be very useful, particularly in describing how we can perform κ-transformations to change one Euler tour into another. We present a theorem (without proof) due to Kotzig about moving from one Euler tour X to another Y.

Theorem 2 (Kotzig [21])**.** *For any two Euler tours X and Y of a 4-regular Eulerian graph G, there exists a finite sequence of κ-transformations that transform X into Y.*

In fact, Kotzig and Abraham [1] were able to prove a stronger result, that Theorem 2 holds true for all Eulerian graphs, not necessarily 4-regular.

Later in this dissertation, we will show how the structure of $G_{2,n}$ can be exploited to design a sequence of κ-transformations that change some Euler tour X on $G_{2,n}$ to another Euler tour Y in such a way that will help us prove that the number of Euler tours of $G_{2,n}$ can be approximated in polynomial time. As mentioned previously, the problem of approximately counting the number of Euler tours of $G_{2,n}$ in polynomial time can be reduced to the problem of randomly sampling Euler tours of $G_{2,n}$ in polynomial time. In Chapter 3, we will describe a method for random sampling called the Markov Chain Monte Carlo method.

Chapter 3

Markov Chain Monte Carlo Method

3.1 Introduction

The Markov chain Monte Carlo method (MCMC) provides a way to randomly sample from a set Ω using probability distribution π. For this project, the set Ω of interest is the set of all Euler tours of $G_{2,n}$.

Recall that a Markov chain $\mathfrak{M}$ is ergodic if and only if its probability distribution converges to a unique stationary distribution π regardless of the initial state. The Markov chain Monte Carlo method requires us to design a Markov chain $\mathfrak{M}$ with state space Ω and unique stationary distribution π. Thus $\mathfrak{M}$ needs to be ergodic. Furthermore, for the purpose of approximately counting the size of Ω, we need to sample from Ω uniformly at random (Jerrum, Valiant and Vazirani [18]). Therefore $\mathfrak{M}$ needs to be designed to have a uniform stationary distribution.

With this construction, we then simulate $\mathfrak{M}$ until the probability distribution is arbitrarily close to π, and output the current state X of the chain. Note that by following these steps, X will have been sampled at random from Ω according to π, as desired.

We need to consider how many steps in the chain will be necessary until the probability distribution over the output states is arbitrarily close to π. Let T be the number of steps necessary. The size of T is important since we want to ensure that our algorithm runs in time polynomial in the size of the problem instance. In our Euler tours on $G_{2,n}$ example, we need to ensure that the algorithm runs in time polynomial in the number of vertices of $G_{2,n}$. Since the size of the state space is often exponential in the size of the problem instance, this requires that T is significantly smaller than the size of the state space (Jerrum, [15]). This is true in the case of the number of Euler tours on $G_{2,n}$, where the size of the state space is exponential in the number of vertices, as Creed

demonstrated [9].

The number of steps T required to become within ε of variation distance to the uniform distribution is referred to as the mixing time of the Markov Chain [17]. If the algorithm does indeed run in polynomial time, then we describe the chain as **rapidly mixing**. Conversely, we call a Markov chain **torpidly mixing** if its mixing time is not polynomial with respect to the problem instance [17]. In Section 3.2 we will present a method for bounding the mixing time called the canonical paths method, and in Section 3.3 we will present a property of a Markov chain called conductance. As in the survey paper by Jerrum and Sinclair [17], we now work through an example to illustrate the Markov chain Monte Carlo Method.

3.1.1 #Knapsack

The #Knapsack problem is a simple counting problem that is complete for the class $\#P$.

We will formalise the problem in the following way. Let $a = (a_1, a_2, ...a_n) \in \mathbb{N}$ be a vector with the size of item i in entry i, and let $b \in \mathbb{N}$ be the size of our knapsack. Then we wish to find the number of different vectors $x \in \{0,1\}^n$ satisfying $\sum_{i=1}^{n} a_i x_i \leq b$. Equivalently, we are estimating the size of the boolean n-dimensional hypercube truncated at the hyperplane $a \cdot x = b$.

#Knapsack: *Consider a knapsack of a fixed size $b \in \mathbb{N}$, and a collection of n items, item i having size a_i ($a_i \in \mathbb{N}$ for all $i \in \{1,...,n\}$). How many different ways are there to pack the knapsack? (a packing being a collection of items whose total size is at most b)*

Note that if $\sum_{i=1}^{n} a_i \leq b$, then the problem becomes trivial: there are 2^n solutions. So for the remainder of this section we assume otherwise. To be able to use the MCMC method with this problem, we would need to define a Markov Chain $\mathfrak{M}_{Knap}$ [17].

Definition 3.1. *Let $\mathfrak{M}_{Knap}$ be the Markov chain which has all vectors x satisfying $\sum_{i=1}^{n} a_i x_i \leq b$ as its state space Ω, with the following rules for transferring from some state x to the next state y:*

- *with probability $\frac{1}{2}$ let $y = x$;*
- *with probability $\frac{1}{2}$, select i u.a.r. where $1 \leq i \leq n$. Define $y' = (x_1, ..., x_{i-1}, 1 - x_i, x_{i+1}, ..., x_n)$. If y' satisfies $a \cdot y' \leq b$ then set $y = y'$. Else set $y = x$.*

We will now show that the chain is ergodic. For irreducibility, note that every pair of states x and y communicate with each other via the zero vector. This is because if a non-zero vector x satisfied the inequality $\sum_{i=1}^{n} a_i x_i \leq b$, then so would the vector x', which is equal to x in every position except at some position i where $x_i = 1$ and $x'_i = 0$. Using this technique we can form a sequence of vectors $x, x', \ldots$ which remove one non-zero entry at a time until the zero vector is obtained, and each vector in the sequence will satisfy the knapsack inequality. We could also re-add any non-zero entries to the zero vector one by one to obtain vector y which we also know to satisfy the knapsack inequality. Therefore every pair of states x and y do indeed communicate with each other.

Furthermore, the Markov chain is aperiodic as it contains self loops. Therefore the chain is ergodic, and so has a unique stationary distribution. For two vectors x and y in the state space, we can see from the design of the chain that $P(x,y) = P(y,x)$. Since x and y are both in the state space, $P(x,y)$ and $P(y,x)$ are both $\frac{1}{2n}$. Therefore by Lemma 2.1, the unique stationary distribution is the uniform distribution.

Hence the MCMC method can be used with our Markov chain $\mathfrak{M}_{Knap}$. Morris and Sinclair [23] found that the mixing time for this chain is polynomial in n, where n is the length of the vector a, or equivalently the dimension of the hypercube. Their proof of rapid mixing is quite difficult, using balanced almost-uniform permutations, and would not be appropriate to include here.

However we will show how, for this example, we can use random samples to estimate the number of satisfying solutions there are to the inequality $\sum_{i=1}^{n} a_i x_i \leq b$ using a technique called telescoping, hence designing a FPRAS.

3.1.1.1 Telescoping

Jerrum and Sinclair [17] show that telescoping is a technique commonly used to estimate the size of the state space using random sampling. The technique relies on a partitioning sequence of the original state space, so that in each step in the sequence, the state space reduces in size in a polynomial way. In the case of the knapsack problem, this will be achieved by keeping the item sizes the same, but reducing the size of the bound (and dropping an item from time-to-time).

First, Jerrum and Sinclair [17] create a sequence of bounds $b_0, b_1, \ldots, b_n$ for the size of the knapsack. We find the sequence of bounds in the following way. Assume w.l.o.g.

that $a_1 \leq a_2 \leq ... \leq a_n$, and define $b_0 = 0$, and b_i as:

$$b_i = min[b, \sum_{j=1}^{i} a_j] \quad for\ 1 \leq i \leq n. \tag{3.1}$$

This definition ensures that $b_n = b$. Since each of these b_i bounds will have its own set of vectors satisfying the knapsack bound, each will have its own state space. We make this clear with the notation $\Omega(b_i)$ where $\Omega(b_i)$ is the set of solutions to the knapsack bound for b_i, and $|\Omega(b_i)|$ is the number of such solutions. When $b = 0$, the only satisfying assignment to x is the zero vector, so $|\Omega(b_0)| = 1$.

We decompose $|\Omega(b)|$ in the following way:

$$|\Omega| = |\Omega(b_n)| = \frac{|\Omega(b_n)|}{|\Omega(b_{n-1})|} \times \frac{|\Omega(b_{n-1})|}{|\Omega(b_{n-2})|} \times ... \times \frac{|\Omega(b_1)|}{|\Omega(b_0)|} \times |\Omega(b_0)|. \tag{3.2}$$

We show that

$$|\Omega(b_{i-1})| \leq |\Omega(b_i)| \leq (n+1)|\Omega(b_{i-1})| \quad \forall 1 \leq i \leq n. \tag{3.3}$$

The first inequality holds because b_i has increased from b_{i-1} by at most the size of the next heaviest item a_i, and at least 0, by construction. Notice that any vector x in $\Omega(b_i)$ may be converted into a vector belonging to $\Omega(b_{i-1})$ by switching the rightmost non-zero entry to 0. This entry could have been located in at most n positions (since x is a $1 \times n$ vector), giving the second inequality [17].

Using MCMC, we can draw a solution to the knapsack problem at random for a knapsack of size b_i, and determine whether it is also a solution for the knapsack of size b_{i-1} (i.e. whether it also lies in $\Omega(b_{i-1})$). Repeating this process a number of times and taking the average success rate gives an approximation to $|\Omega(b_{i-1})|/|\Omega(b_i)|$.

They then show that approximating each of the individual ratios in Equation 3.2 can be achieved in polynomial time. Then since there are n ratios, we can approximate $|\Omega|$ in polynomial time.

Let ρ_i be the ratio $|\Omega(b_{i-1})|/|\Omega(b_i)|$. Due to Equation 3.3, we can obtain the bound on ρ_i shown in Equation 3.4

$$\rho_i = \frac{|\Omega(b_{i-1})|}{|\Omega(b_i)|} \geq \frac{1}{(n+1)}. \tag{3.4}$$

Using MCMC to approximate each ρ_i, our expectation for a single trial will be ρ_i and its variance $\rho_i(1-\rho_i)$.

Jerrum and Sinclair show that we can achieve a good approximation (in the sense of a FPRAS as defined in Section 2.3) for $|\Omega(b)|$ after performing $t = 17\varepsilon^{-2}n^2$ trials for each ρ_i.

Let $\overline{X}_i$ be the sample mean for each ρ_i. We will analyse the ratio of the variance of the sample mean to its expectation squared to show that $17\varepsilon^{-2}n^2$ trials are sufficient.

$$\frac{Var[\overline{X}_i]}{\rho_i} = \frac{1-\rho_i}{t\cdot\rho_i} \leq \frac{(1-\frac{1}{n+1})}{t\frac{1}{(n+1)}} = \frac{n}{t} = \frac{\varepsilon^2}{17n} \qquad [17] \tag{3.5}$$

The second term in Equation 3.5 is due to performing t trials, while the third term is due to equation 3.4.

Suppose we performed trials for each of the ratios in Equation 3.2, thus obtaining the random variable $Z = \overline{X}_n \cdot \ldots \cdot \overline{X}_1$ which is our approximation for $|\Omega|$. Each $\overline{X}_i$ is independent, so the expectation of Z is:

$$\mathbb{E}(Z) = \rho_n \cdot \ldots \cdot \rho_1 = |\Omega(b)|^{-1}. \tag{3.6}$$

Expanding on the equations in [17], the ratio of the variance of Z to its expectation squared is:

$$\frac{Var(Z)}{(\mathbb{E}(Z))^2} = \frac{\mathbb{E}[(\overline{X}_1\ldots\overline{X}_n)^2] - (\mathbb{E}[\overline{X}_1\ldots\overline{X}_n])^2}{(\mathbb{E}[\overline{X}_1\ldots\overline{X}_n])^2} \tag{3.7}$$

$$= \frac{\mathbb{E}[\overline{X}_1^2\ldots\overline{X}_n^2]}{(\mathbb{E}[\overline{X}_1\ldots\overline{X}_n])^2} - 1 \tag{3.8}$$

$$= \frac{\mathbb{E}[\overline{X}_1^2]\ldots\mathbb{E}[\overline{X}_n^2]}{(\mathbb{E}[\overline{X}_1\ldots\overline{X}_n])^2} - 1 \tag{3.9}$$

$$= \frac{\prod_{i=1}^{n}(Var[\overline{X}_i] + (\mathbb{E}[\overline{X}_i])^2)}{\prod_{i=1}^{n}(\mathbb{E}[\overline{X}_i]^2)} - 1 \tag{3.10}$$

$$= \prod_{i=1}^{n}\left(1 + \frac{Var[\overline{X}_i]}{\rho_i^2}\right) - 1 \tag{3.11}$$

$$\leq \left(1 + \frac{\varepsilon^2}{17n}\right)^n - 1 \tag{3.12}$$

$$\leq \frac{\varepsilon^2}{16} \tag{3.13}$$

where Equation 3.12 follows from Equation 3.5, and Equation 3.13 can be seen from the binomial expansion of Equation 3.12.

Now the final step is to show that Z is indeed an approximation in the FPRAS sense, using an extension of Chebyshev's inequality.

Proposition 3.1 (Chebyshev's Inequality)**.** *Let Z be a random variable with expectation $\mathbb{E}[Z]$. Then Chebyshev's inequality states that*

$$Pr(\lambda_1 \leq Z \leq \lambda_2) \geq 1 - \frac{4\cdot Var(Z)}{(\lambda_2-\lambda_1)^2}. \tag{3.14}$$

where $\lambda_1 + \lambda_2 = 2\mathbb{E}[Z]$ [27].

Take λ_1 and λ_2 to be $\lambda_1 = (1-\frac{\varepsilon}{2})|\Omega(b)|^{-1}$ and $\lambda_2 = (1+\frac{\varepsilon}{2})|\Omega(b)|^{-1}$. Then $\lambda_1+\lambda_2 = 2|\Omega(b)|^{-1} = 2\mathbb{E}[Z]$, and so it is possible to use Chebyshev's inequality to find that:

$$Pr\big((1-\frac{\varepsilon}{2})|\Omega(b)|^{-1} \leq Z \leq (1+\frac{\varepsilon}{2})|\Omega(b)|^{-1}\big) \geq 1 - \frac{4\cdot Var(Z)}{\varepsilon^2|\Omega(b)|^{-2}} \tag{3.15}$$

$$= 1 - \frac{4\cdot Var(Z)}{\varepsilon^2\mathbb{E}[Z]^2} \tag{3.16}$$

$$\geq 1 - \frac{4\varepsilon^2}{16\varepsilon^2} \tag{3.17}$$

$$= \frac{3}{4}. \tag{3.18}$$

as in [17]. We took $17\varepsilon^{-2}n^2$ trials for each ratio, so the algorithm is polynomial in ε^{-1} and n. Hence by Equation 3.18 the random variable $Z = |\Omega(b)|^{-1}$ can be approximated in polynomial time, and therefore we can derive an approximation for the state space in polynomial time. Combining this with the result of [23] that the Markov chain defined in Definition 3.1 is rapidly mixing, we do indeed have a FPRAS for the number of solutions to the knapsack problem.

In the next section we will introduce a method called canonical paths which can be used to bound the mixing time T of some Markov chain $\mathfrak{M}$.

3.2 Canonical Paths

The canonical paths method was first introduced by Jerrum and Sinclair [16], where they also proved the theorems to show it bounds the mixing time.

The canonical paths method interprets $\mathfrak{M}$ as an undirected **state space graph** $G_{\mathfrak{M}}$ with vertex set Ω, and an edge between two states X and Y if it is possible to transition between them in one step (i.e. $Q(X,Y) = \pi(X)P(X,Y) > 0$) . On this graph we wish to find a "good" set of paths between each pair of vertices, so that no edge of the graph is overused. In this way, constructing such a set of paths is a commodity flow problem, where we must use our knowledge of the Markov chain to design our set of paths appropriately. Note that this is an abstraction, and rarely is $\mathfrak{M}$ worked with as a state space graph directly.

In order to use this method, the Markov Chain must be designed to have the property that at any state X, the loop probability $P(X,X) \geq \frac{1}{2}$. Furthermore, the chain must be reversible, as defined below.

Definition 3.2. *Let $\pi(X)$ be the probability of state $X \in \Omega$ in the stationary distribution of some Markov Chain $\mathfrak{M}$, and $P(X,Y)$ be the probability of reaching state Y from state*

X after one step of the chain. Then $\mathfrak{M}$ is reversible if it satisfies the ***detailed balance condition:***

$$Q(X,Y) = \pi(X)P(X,Y) = \pi(Y)P(Y,X) \quad \forall X,Y \in \Omega. \tag{3.19}$$

For any two states X and Y, there may be many different paths from X to Y in the state space graph. The method requires us to use our knowledge of the Markov chain to design a **canonical path** $\gamma_{X,Y}$ from X to Y (or for the multi-commodity flow generalisation, a distribution over $X \to Y$ paths) that makes the set of all canonical paths $\Gamma = \{\gamma_{X,Y} : X,Y \in \Omega\}$ "good". When designing our canonical paths $\gamma_{X,Y}$, our focus is to consider all the canonical paths collectively, and to avoid creating a set Γ which induces "hot spots": edges of the state space graph used by disproportionately many of the $\gamma_{X,Y}$ paths. We then measure how evenly the edges in $G_{\mathfrak{M}}$ have been used with ρ:

$$\rho(\Gamma) = \max_e \frac{1}{Q(e)} \sum_{\gamma_{X,Y} \ni e} \pi(X)\pi(Y)|\gamma_{X,Y}| \tag{3.20}$$

where $|\gamma_{X,Y}|$ is the length of the path $\gamma_{X,Y}$. Note that it is usually straightforward to bound the length of the path $\gamma_{X,Y}$. For example in the case of Euler tours of $G_{2,n}$, we will be able to construct paths of length $O(n)$.
The following result due to Sinclair [25] uses ρ to bound the mixing time of the chain.

Proposition 3.2. *Let $\mathfrak{M}$ be a finite, reversible, ergodic Markov chain where the probability of a self loop $P(X,X) \geq \frac{1}{2}$ for every $X \in \Omega$. Then the mixing time τ_X of $\mathfrak{M}$ starting at initial state X satisfies*

$$\tau_X(\varepsilon) \leq \rho(ln(\pi(X)^{-1}) + ln(\varepsilon^{-1})) \quad \textit{for any } X \in \Omega. \tag{3.21}$$

Now we will consider an artificial problem: a degenerate version of #Knapsack constructed to provide a simple illustration of how the canonical paths method can be used.

3.2.1 Degenerate #Knapsack

Based on the survey by Jerrum and Sinclair [17] we present a simplified form of #Knapsack, where the size of the knapsack is fixed to be at least as big as the combined size of all items, so the bag can fit every item. This can be viewed as estimating the size of the full hypercube without truncation, an artificial problem, since the size of the

full hypercube in n dimensions is well known to be 2^n. This example problem will be analysed to demonstrate the canonical paths technique introduced in Section 3.2.
Again take a to be the vector of item sizes, and fix the value of b so that $\sum_i a_i \leq b$. Consider two possible solutions x and y to the inequalities $a \cdot x \leq b$ and $a \cdot y \leq b$ respectively. Since we made b sufficiently large, our state space is $\{0,1\}^n$, and so $x = (x_1, x_2, ..., x_n)$ and $y = (y_1, y_2, ..., y_n)$ could be any $0-1$ vector of length n.
To design the canonical path from x to y, start at x_1 and fix the bit x_1 to match y_1. Proceed to fix the bits from x_i to y_i in the order they appear in the vector, so edge i in the path fixes bit i to take the value y_i. Since x and y are $1 \times n$ vectors, the path will have length n (though some of the intermediate vectors may be identical to the prior one, with dummy transitions). The edge in position i on the path takes the form:

$$((y_1, y_2, ..., y_{i-1}, x_i, x_{i+1}, ..., x_n), (y_1, y_2, ..., y_{i-1}, y_i, x_{i+1}, ..., x_n)) \tag{3.22}$$

and changes bit i from x_i to y_i. The eventuality that $x_i = y_i$ is resolved by the presence of self loops at every vertex on the state space graph. Now to analyse ρ for their collection of canonical paths, we consider some arbitrary edge e of the state space graph:

$$e = (w, w') = ((w_1, w_2, ..., w_i, ..., w_n), (w_1, w_2, ..., w'_i, ..., w_n)). \tag{3.23}$$

In order to bound the mixing time, a complementary vector will need to be constructed. This takes the form of a $1 \times n$ vector $u = (u_1, u_2, ..., u_n)$ which is constructed using x, y and e. The design of u will be such that with knowledge only of e and the complementary vector u, we will be able to reconstruct x and y.
Define $\Gamma(w, w')$ to be the set of all paths $\gamma_{x,y}$ that use the edge (w, w') of the state space graph. We define a mapping $\eta_e : \Gamma(w, w') \to \Omega$ to map each of the end points of the elements of $\Gamma(w, w')$ to a unique element u of the state space Ω.

$$\eta_e(x, y) = (x_1, x_2, ..., x_{i-1}, w_i, y_{i+1} ..., y_n) = (u_1, u_2, ..., u_n) = u. \tag{3.24}$$

As required, with knowledge of $e = (w, w')$ (shown in Equation 3.23) and u we can entirely retrieve back the vectors x and y:

$$x = (x_1, x_2, ..., x_n) = (u_1, u_2, ..., u_{i-1}, w_i, w_{i+1}, ..., w_n) \tag{3.25}$$

$$y = (y_1, y_2, ..., y_n) = (w_1, w_2, ..., w_{i-1} w'_i, u_{i+1}, ..., u_n). \tag{3.26}$$

Hence the map is injective, and so each element of Ω cannot be mapped to by more than one x, y pair. Consider for a fixed edge e, the set of all $\eta_e(x, y)$ mapped to by all

possible states x and y that use e. Since the map is injective, this must be smaller than the whole state space Ω, and so

$$\sum_{\gamma_{x,y} \ni e} \pi(\eta_e(x,y)) \leq \sum_{x \in \Omega} \pi(x) = 1. \tag{3.27}$$

The final equality is due to π being a probability distribution over the state space. Finally, note that since π is the uniform distribution, $\forall a, b \in \Omega,\ \pi(a) = \pi(b)$. This information can be used to find a bound on the maximum edge loading using equation 3.20.

$$\begin{aligned}
\frac{1}{Q(e)} \sum_{\gamma_{x,y} \ni e} \pi(x)\pi(y)|\gamma_{x,y}| &= \frac{1}{Q(e)} \sum_{\gamma_{x,y} \ni e} \pi(w)\pi(\eta_e(x,y))|\gamma_{x,y}| && (3.28)\\
&= \frac{1}{\pi(w)P(w,w')} \sum_{\gamma_{x,y} \ni e} \pi(w)\pi(\eta_e(x,y))|\gamma_{x,y}| && (3.29)\\
&= \frac{n}{P(w,w')} \sum_{\gamma_{x,y} \ni e} \pi(\eta_e(x,y)) && (3.30)\\
&\leq \frac{n}{P(w,w')} && (3.31)\\
&\leq 2n^2. && (3.32)
\end{aligned}$$

Note that the final inequality holds because the first rule of the Markov chain makes the probability of the transition being a self loop $\frac{1}{2}$. Then, assuming the transition is not a self loop, the probability of bit i being flipped is $\frac{1}{n}$. Thus $P(w,w') \leq \frac{1}{2n}$. Since the choice of e was arbitrary, it follows that $\rho \leq 2n^2$.

Therefore, using the canonical paths method we have found a bound on the mixing time which is polynomial in the length of the vectors (or the number of items we can place in our knapsack). We will revisit the canonical paths method in Chapter 5 when we utilise it to show that the number of Euler tours on $G_{2,n}$ can be approximately counted in polynomial time.

Finally, we present the closely related *conductance* of a Markov chain.

3.3 Conductance

Consider again the state space graph for our chain. Intuitively, to have a good bound on the mixing time of the chain we would like to have a state space graph on which we can move around freely, without getting stuck in any area. This is what conductance measures; a high conductance implies the chain may rapidly reach its stationary distribution.

Consider some subset of the vertices S of the state space graph, and partition the graph into vertex sets S and $\Omega\backslash S$. We measure the conditional probability of "escaping" from S in one step of the chain, given that the initial state was in S, and the conductance is assigned the lowest value over all possible subsets S. We formalise this in Equation 3.33, where Φ denotes conductance:

$$\Phi(\mathfrak{M}) = \min_{S\subset\Omega:\ 0<\pi(S)\leq\frac{1}{2}} \frac{Q(S,\Omega\backslash S)}{\pi(S)} \tag{3.33}$$

where $Q(S,\Omega\backslash S)$ is the sum over all $Q(x,y)$ for $\{x,y\} \in E$ satisfying $x \in S$, $y \in \Omega\backslash S$ [17].

Using this bound on conductance, Sinclair [25] established the following bound on the mixing time of the chain.

Proposition 3.3 (Sinclair [25])**.** *Let $\mathfrak{M}$ be a reversible, finite, ergodic Markov chain with loop probabilities $P(X,X) \geq \frac{1}{2}$. For any choice of initial state X, the mixing time of $\mathfrak{M}$ satisfies the following bound:*

$$\tau_X(\varepsilon) \leq 2\Phi^{-2}(ln(\pi(X)^{-1}) + ln(\varepsilon^{-1})).$$

Conductance is a property of a Markov chain, and in order to find a good upper bound on the mixing time of the chain, we require a good lower bound on its conductance. In Section 4.2, we will present work by Tetali and Vempala which explicitly makes use of conductance in an attempt to bound the mixing time of a chain.

Chapter 4

Previous work

4.1 Introduction

In this chapter we discuss the work that we have built upon in order to construct our bound for the number of Euler tours of $G_{2,n}$, focusing primarily on the papers by Creed [9] and Tetali and Vempala [28]. The former gave an exact counting algorithm for $G_{2,n}$ and also $G_{3,n}$ (although the algorithm for $G_{3,n}$ was found to have mistakes, later corrected by Astefanoaei [3] and Marinov [22]). The exact counting algorithm ran in time polynomial in n for $G_{2,n}$ and $G_{3,n}$, although generalisations to $G_{m,n}$ would be exponential in m using this method [9].
Tetali and Vempala attempted to show that the problem of approximating the number of Euler tours on a 4-regular graph can be achieved in polynomial time. They made use of the MCMC method and canonical paths to form their bound, however their work was found to contain errors and has not been fixed. The errors will be discussed in Section 4.2.2, and an overview of their proof is discussed in Section 4.2.

4.2 Tetali and Vempala's Approach

In order to find an approximation algorithm for the number of Euler tours on the 4-regular grid, Tetali and Vempala used the idea of Jerrum, Valiant and Vazirani [18] to sample from the state space Ω of all Euler tours of a certain 4-regular graph. To do so they constructed the Markov chain $\mathfrak{M}$ shown in Definition 4.1 below, aiming to use this with MCMC to sample from Ω in polynomial time. Thus their proof focused on attempting to show that $\mathfrak{M}$ is rapidly mixing on the class of 4-regular graphs. The method they attempted to use to construct this proof is discussed in Section 4.2.1.

4.2.1 Method

Tetali and Vempala attempted to show that for a given connected 4-regular graph G, the following Markov chain $\mathfrak{M}$ is rapidly mixing.

Definition 4.1. *Let the Kotzig Chain $\mathfrak{M}$ be a Markov chain with state space Ω: the set of all Euler tours of G. Start at an arbitrary Euler tour X, and pick some vertex v u.a.r. Then:*

- *with probability $\frac{1}{2}$ perform the unique legal κ-transformation at v (as defined in Section 2.5);*
- *with probability $\frac{1}{2}$ do nothing.*

In order to use $\mathfrak{M}$ to sample uniformly at random from Ω using the MCMC method, it must be the case that $\mathfrak{M}$ has the uniform distribution as its unique stationary distribution. By Theorem 2 on page 14, for each Euler tour X, we can reach any other Euler tour Y via a finite sequence of κ-transformations. Therefore all states in $\mathfrak{M}$ are accessible from each other, and so $\mathfrak{M}$ is irreducible. Furthermore, due to the second rule of the Markov chain, $\mathfrak{M}$ contains a self loop at every state, and so $\mathfrak{M}$ is aperiodic. Therefore we may conclude that $\mathfrak{M}$ is ergodic, and so has a unique stationary distribution.

To show that the unique stationary distribution is the uniform distribution, we will need to show that $\mathfrak{M}$ is symmetric. Note that if we obtain Euler tour X' by transforming vertex v in X to its available transition, then the transition $TS(X,v)$ will be available for X'. Therefore $P(X,X')$ and $P(X',X)$ are both equal to the probability of picking vertex v to transform, which is $\frac{1}{2n}$. Hence $\mathfrak{M}$ is symmetric and so the unique stationary distribution of $\mathfrak{M}$ is the uniform distribution, as required.

To attempt to show that $\mathfrak{M}$ is rapidly mixing, Tetali and Vempala used the canonical paths method. They considered the state space graph $G_{\mathfrak{M}}$ as the graph whose vertices represent Euler tours of a certain 4-regular graph, and place an edge between two vertices X and X' if X' can be obtained from X with one κ-transformation.

Let $D_{X,Y}$ be the set of vertices that have a different transition system in X and Y, and call each $v \in D_{X,Y}$ a difference vertex. We call $D_{X,Y}$ the **difference set** of X and Y. In order to build a canonical path between X and Y, Tetali and Vempala attempted to fix an ordering on the difference vertices so that when starting from X, after performing a κ-transformation on each of the difference vertices in this order we would obtain

Figure 4.1: Edges incident to vertices u and v

Euler tour Y. Recall that in a 4-regular graph there will only ever be one possible κ-transformation at a vertex, and so such an ordering would completely specify a path from X to Y.
Lemma 4.1 was a key tool in their attempt to come up with such an ordering. We make extensive use of this Lemma in Chapter 5.

Lemma 4.1 (Tetali and Vempala [28])**.** *Let X and Y be two Euler tours of a given 4-regular graph G. Suppose $TS(X,v) \neq TS(Y,v)$, and $TS(X) \not\Rightarrow_v TS(Y)$, so the κ-transformation available at v in X will not result in the transition system of v in Y. Then there exists some "helper" vertex $u(\neq v)$ such that:*

- *$TS(X,u) \neq TS(Y,u)$, and*
- *in at most 3 κ-transformations, we can transform X to X' where $TS(X',u) = TS(Y,u)$, $TS(X',v) = TS(Y,v)$, and $TS(X',w) = TS(X,w)$ for $w \neq u,v$.*

Proof. Suppose we have an Euler tour X of G which has the transition system $e(v)e'$ and $f(v)f'$ at some vertex v. Assume that the transition system $e(v)f$ (and hence $e'(v)f'$) is the allowed transition system, and so can be reached via one κ-transformation on X at v. Hence the layout of X must have been in the form $...e(v)e'...f(v)f'....$ Suppose we have another Euler tour Y which has the transition system $e(v)f'$ (and hence $e'(v)f$). Suppose that the layout of Y is $...e(v)f'...e'(v)f$ and hence $TS(X) \not\Rightarrow_v TS(Y)$. There must be some vertex u interleaving with v such that the transition at u also differs in X and Y. If this were not the case, then every vertex in Y would either have the same transition system as in X, or would not interleave with v. Recall that the only way to switch the available and prohibited transition system at a vertex is to perform a κ-transformation on an interleaving vertex. Then following this reasoning, it is not possible to perform a κ-transformation to make the transition at v in Y available and so we cannot convert Euler tour X into Euler tour Y. This is a contradiction, since by Theorem 2 we can always perform a finite sequence of κ-transformations

to convert one Euler tour into another. We conclude that if we have some vertex v where $TS(X) \not\Rightarrow_v TS(Y)$ then indeed there must be some vertex u whose transition system differs in X and Y and which interleaves with v. This proves the first part of the Lemma.

For the second part of the Lemma, let vertex u have the transitions $g(u)g'$ and $h(u)h'$ in Euler tour X, arranged as $...g(u)g'...h(u)h'....$ Then the allowed transition at u generates the new tour $...g(u)h...g'(u)h'$, and the prohibited transition system is $g(u)h'$ (and hence $g'(u)h$).

There are two considerations we must make for u: first that $TS(X) \Rightarrow_u TS(Y)$, and second that $TS(X) \not\Rightarrow_u TS(Y)$. We first suppose that u has the transition system $g(u)h$ and $g'(u)h'$ in Y, arranged as $...h(u)g...h'(u)g'...$, so $TS(X) \Rightarrow_u TS(Y)$. We can write the tour X as

$$X = ...h(u)h'...e'(v)e...g(u)g'...f'(v)f...$$

Then by making the κ-transformation at u, we change the allowed and prohibited transitions at v since u and v are interleaving.

This changes the tour to

$$X' = ...h(u)g...e(v)e'...h'(u)g'...f'(v)f...$$

Now that we have transformed u, we have $TS(X) \Rightarrow_v TS(Y)$, and after performing the k-transformation at v we obtain Euler tour

$$X'' = ...h(u)g...e(v)f'...g'(u)h'...e'(v)f...$$

with $TS(X',u) = TS(Y,u)$, $TS(X',v) = TS(Y,v)$, and $TS(X',w) = TS(X,w)$ for $w \neq u,v$ as required.

Now suppose that u has the transition system $g(u)h'$ and $g'(u)h$ in Y, arranged as $...h(u)g'...g(u)h'...$ so $TS(X) \not\Rightarrow_u TS(Y)$. First, we perform the κ-transformation at v. We write the resulting tour below as X^*.

$$X^* = ...h(u)h'...e'(v)f'...g'(u)g...e(v)f...$$

This will give the transition $e(v)f$ and $e'(v)f'$ which is not the same as $TS(Y,v)$. However by performing this κ-transformation, the allowed and prohibited transformations at u have been switched, so now $TS(X) \Rightarrow_u TS(Y)$. As previously shown, if $TS(X) \not\Rightarrow_v TS(Y)$ and $TS(X) \Rightarrow_u TS(Y)$, then we can give u and v their transition systems in Y in 2 κ-transformations, giving 3 κ-transformations in total, as required. □

This lemma has the potential to be useful for specifying an ordering on the vertices of a 4-regular graph for the following reason. Recall that $D_{X,Y}$ is the set of vertices which have different transition systems in X and Y. Then for some vertex $v \in D_{X,Y}$ where $X \not\Rightarrow_v Y$, Lemma 4.1 implies that some other vertex $u \in D_{X,Y}$ can be used to help fix v to have the transition system in Y.

Tetali and Vempala used the rules defined below to attempt to specify an ordering on the vertices [28]. We know from Theorem 2 that we can always perform a finite sequence of κ-transformations to convert some Euler tour X into another Euler tour Y, so the purpose of such an ordering was not to show that we can convert some X into some Y. Rather the ordering was designed for two main purposes. Firstly, so that an ordering could be determined on the difference vertices of X and Y *before* performing any κ-transformations, by comparing X and Y, and specifying the order for the canonical path with necessary relationships remaining true throughout. Secondly, to accommodate a complementary tour W, similar to the complementary vector used in the example in section 3.2.1. We will describe the complementary tour W in detail in Section 4.2.1.1. Roughly speaking, the complementary tour W is constructed to complement an intermediate Euler tour Z on the canonical path from X to Y: where the vertices in Z have the transitions of X, the vertices in W will have the transition systems of Y, and vice versa. Below we present the rules they set out, which were found to be flawed as we will later discuss.

Rule 1

For some vertex $v \in D_{X,Y}$, suppose that $TS(X) \Rightarrow_v TS(Y)$ and $TS(Y) \not\Rightarrow_v TS(X)$. Then by Lemma 4.1, there is some vertex $u \in D_{X,Y}$ that can make the move from Y to X at v available . Then order u and v so that v immediately precedes u.

Rule 2

For a vertex $v \in D_{X,Y}v$ where $TS(X) \not\Rightarrow_v TS(Y)$, by Lemma 4.1, there will be some other vertex u that can be used to make $\mathrm{TS}(X,v) = \mathrm{TS}(Y,v)$ and $\mathrm{TS}(X,u) = \mathrm{TS}(Y,u)$. If this can be achieved in 2 moves, then let u be ordered before v. Otherwise if 3 moves are required, order v before u.

Rule 3

If for some $v \in D_{X,Y}$ we have $TS(X) \Rightarrow_v TS(Y)$ and $TS(Y) \Rightarrow_v TS(X)$, then the placement of v in the ordering is independent of the other vertices.

We demonstrate these rules with an example. Consider two Euler tours A and B on

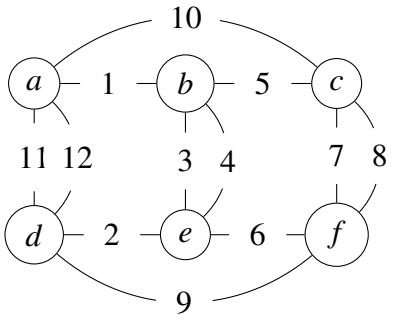

Figure 4.2: Edge labelling for $G_{2,3}$

$G_{2,3}$ where the edge labellings are shown in Figure 4.2.

$$A = 10(a)11(d)9(f)7(c)5(b)4(e)3(b)1(a)12(d)2(e)6(f)8(c)$$

$$B = 10(a)11(d)12(a)1(b)3(e)2(d)9(f)6(e)4(b)5(c)7(f)8(c)$$

We can notice that

- $TS(A,a) = TS(B,a)$;
- $TS(A,b) = TS(B,b)$;
- $TS(A,c) = TS(B,c)$;
- $TS(A) \Rightarrow_d TS(B)$ but $TS(B) \not\Rightarrow_d TS(A)$;
- $TS(A) \not\Rightarrow_e TS(B)$ and $TS(B) \not\Rightarrow_e TS(A)$;
- $TS(A) \Rightarrow_f TS(B)$ and $TS(B) \Rightarrow_f TS(A)$;

Then there will be no κ-transformations on a, b, or c as these are not difference vertices. By Rule 3, f can be placed anywhere in the ordering. By Rule 1, d must immediately precede e. By Rule 2, d must precede e. By following these rules, one of our possible orderings is d, e, f. Then our path from A to B will be

$$A = Z_0 = 10(a)11(d)9(f)7(c)5(b)4(e)3(b)1(a)12(d)2(e)6(f)8(c) \tag{4.1}$$

$$Z_1 = 10(a)11(d)12(a)1(b)3(e)4(b)5(c)7(f)9(d)2(e)6(f)8(c) \tag{4.2}$$

$$Z_2 = 10(a)11(d)12(a)1(b)3(e)2(d)9(f)7(c)5(b)4(e)6(f)8(c) \tag{4.3}$$

$$Z_3 = 10(a)11(d)12(a)1(b)3(e)2(d)9(f)6(e)4(b)5(c)7(f)8(c) = B. \tag{4.4}$$

Thus for this simple case, the rules specify an ordering on the difference vertices to convert A into B.

The rules above completely specify an ordering of the vertices in which to be transformed. A key feature of the rules is that they were designed so that if we have a path from X to Y, we should be able to construct a path from Y to X transforming the vertices in roughly the same order. However this ordering contains a fatal flaw. When ordering the vertices, Tetali and Vempala overlooked that there may be multiple vertices interleaving with a vertex v. Therefore, the rules described above do not define an ordering on the set of difference vertices at all. Apart from that, depending on their placement, the available and prohibited transition systems at v may have changed from how they were in X, so the properties they aimed to achieve are not guaranteed. This will be discussed further in Section 4.2.2. We now discuss how Tetali and Vempala tried to constructed the aforementioned complementary tour.

4.2.1.1 Complementary Tour

The canonical paths method requires a bound on the number of paths using an edge of the state space graph $G_{\mathfrak{M}}$ defined in Section 3.2. In this case, the vertices of $G_{\mathfrak{M}}$ are Euler tours, and there is an edge between two Euler tours X and Y if Y can be reached from X via one κ-transformation.

To bound the number of paths using a certain edge, Tetali and Vempala attempted to define a complementary tour W. Consider the path on the state space graph from X to Y. As each vertex in the proposed ordering is transformed, the resulting tour will be an Euler Tour since it was reached by a κ-transformation. The path from X to Y can then be described as a series of Euler Tours $X = Z_0, Z_1, ...Z_l = Y$.

We wish to consider the difference set $D_{X,Y}$ containing vertices that have different transition systems in X and Y, and the difference set $D_{Z_k,W}$ containing vertices with different transition systems in Z_k and W when we are on edge (Z_k, Z_{k+1}) of the state space graph.

They designed the complementary tour W for tour Z_k in the chain from X to Y to have the following properties in general:

- if X and Z_k have the same transition system for v then v will be allocated the transition system of Y in W;
- if Y and Z_k have the same transition system for v then v will have the transition system of X in W;

- if X, Y and Z_k all have the same transition system for v then W will allocate v this transition system also.

Their design was such that most vertices will hold these properties, with the exception of two. By analysing their ordering, they were able to show that there may be at most two vertices which differ in X and Y (and therefore are present in $D_{X,Y}$) but that do not appear in $D_{Z_k,W}$, the difference set of Z_k and W.

Then suppose we are currently using edge (Z_k, Z_{k+1}) of the state space graph, and we know tour W. We wish to determine how many different tours X and Y could have been used to construct W. By examining the difference set $D_{Z_k,W}$ of Z_k and W, we can approximate the difference set $D_{X,Y}$ of X and Y. Since there may be at most two extra vertices in $D_{X,Y}$ than $D_{Z_k,w}$, we have $n(n-1)$ possibilities for $D_{X,Y}$, where n is the number of vertices.

For each possibility for $D_{X,Y}$, for all vertices u_i not in $D_{X,Y}$ we can retrieve $TS(X,u_i)$ and $TS(Y,u_i)$ from $TS(W,u_i)$. Since vertices not in the difference set were not transformed, they should have the same transition systems in W as they had in X and Y. For those vertices $v_1,...,v_n$ in the difference set $D_{X,Y}$, we would split them depending on their position relative to the vertex v_i in the ordering, where v_i is the vertex which has been transformed by the edge (Z_k, Z_{k+1}). For vertices $v_1,...,v_{i-2}$, these will have the transition systems of Y in Z_k and the transition systems of X in W. Vertices $v_{i+2},...,v_n$ will have their transition systems of X in Z_k as they have not yet been transformed, and the transition systems of Y in W for the same reason.

For the vertices v_{k-1}, v_k, v_{k+1}, Tetali and Vempala showed that by examining the available and prohibited transition systems at these vertices, the transition systems of X and Y at v_{k-1}, v_k and v_{k+1} could also be exactly recovered.

Then the possibilities for X and Y would amount to the possibilities for $D_{X,Y}$, of which there are $n(n-1)$. Therefore, if indeed Rules 1-3 did define a total ordering on the difference vertices of X and Y, they would have had a bound of $n(n-1)$ on the possible number of Euler tours using an edge of the state space graph.

Finally, Tetali and Vempala used their bound on the edge loading to form a lower bound on the conductance of the chain of $1/2n^2$. This would imply that there is a polynomial upper bound on the mixing time of the chain. Therefore, had the proof worked as stated, they would have proven that the chain is indeed rapidly mixing.

4.2.2 Discussion

Tetali and Vempala's approach for ordering the vertices has so far resisted any attempt to fix it. As noted above, the reason for the failure of their ordering to specify a conversion of X into Y by a sequence of κ-transformations is due to the fact that they overlooked that a vertex may interleave with many other vertices. The aim of their ordering was to pair a vertex v with a unique helper u in such a way that the transition systems required at u and v would remain available until it was time to make the transformation. However other vertices ordered before v will have changed whether the required transition at v was available if they also interleaved with v.

Consider Rule 1 above. If $TS(X) \Rightarrow_v TS(Y)$ and $TS(Y) \not\Rightarrow_v TS(X)$, then we find some u satisfying the properties specified, and order the vertices v, u. The reason for doing so is that after transforming u, we will have $TS(Y) \Rightarrow_v TS(X)$ in the path from Y to X. However, reconsidering how this ordering may affect the path from X to Y, we notice that transforming v first may swap the available and prohibited transitions at u, and so the ordering may give u the wrong transition system.

Consider also Rule 3, which states that if $TS(X) \Rightarrow_v TS(Y)$ and $TS(Y) \Rightarrow_v TS(X)$ then v can go anywhere in the ordering. However, after performing κ-transformations on other difference vertices, it may no longer be the case that $TS(X) \Rightarrow_v TS(Y)$ and $TS(Y) \Rightarrow_v TS(X)$, and v may in fact require a helper.

Furthermore, a κ-transformation at some vertex w can change whether vertices v and u are interleaving. Therefore it is difficult to keep track of how vertices interleave without re-examining the entire Euler tour at each step. The complexity of the interaction between κ-transformations and how vertices interleave has meant that their proof technique has so far not been able to be fixed. However, in this dissertation we will use the structure of the $2 \times n$ grid to show how to achieve an ordering, and from that, a rapid mixing result.

We now proceed to examine the work of Creed [9], which focused on exact counting rather than approximate counting.

4.3 Creed's Approach

Creed used the transfer matrix approach, originally suggested by Golin et al. [13] to exactly count the number of Euler tours in two special cases of toroidal square grids:

$G_{2,n}$ and $G_{3,n}$. Many of the techniques he used to form the exact counting algorithm have been employed to our proof of rapid mixing on the toroidal grid $G_{2,n}$ in Chapter 5.

4.3.1 Method

As part of his work in exact counting the number of Euler Tours on grids, Creed developed the concept of legal partial transition systems, and defined classes over them.

4.3.1.1 Classes of Legal Partial Transition Systems

Recall that a transition system TS_G over a graph G partitions the edges at each vertex into pairs, forming a set of disjoint cycles over the whole graph. The transition system TS_G is an Euler tour if it induces only one cycle.
Creed considered transition systems over toroidal square grids $G_{m,n}$ and certain subgraphs of these grids. Roughly speaking, let V_k be the subgraph of $G_{m,n}$ containing only the first $k \leq n$ columns.

Definition 4.2 (Creed [9])**.** *Let $V_k = \{(i,j) :\ 0 \leq i \leq m-1,\ 0 \leq j \leq k-1\}$. Then a transition system on V_k is a legal partial transition system if it can be extended to a transition system representing an Euler tour on the whole grid $G_{m,n}$.*

Note that V_k is not strictly a vertex-induced subgraph, as it retains the trailing edges that would have connected vertex (i,k) to $(i,k+1)$ and vertex $(i,0)$ to $(i,n-1)$ for all $0 \leq i \leq m-1$.
For a transition system to be legal, it cannot contain any disjoint cycles, as extending the transition system to the whole grid would certainly create multiple disjoint cycles. Thus a legal partial transition system partitions V_k into a collection of paths which must have endpoints in column 0 and/or column k.
Consider the trailing edges from column 0 and column $k-1$ of the partial transition system, which in the full grid would link column $n-1$ to 0 and column $k-1$ to k. Let l_i be the trailing edge that would connect $\{(i,0),(i,n-1)\}$ and r_i be the trailing edge that would connect $\{(i,k-1),(i,k)\}$. We wish to examine the different ways these trailing edges can be paired together by the paths over V_k. For example, the tour in figure 4.3 pairs l_0 with r_1 and l_1 with r_0.

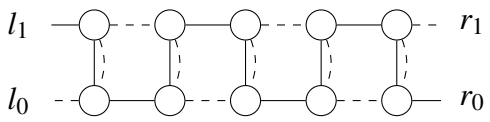

Figure 4.3: Example of a tour over a partial transition system

The set $\mathcal{P}(m)$ is the set of perfect matchings on the set $\{l_0, l_1, ..., l_m, r_0, r_1, ..., r_m\}$. Each of these perfect matchings will represent a class of transition systems that match their endpoints in this way. For example, Figure 4.3 will belong to the class $[l_0, r_1]$. Creed notes that for a grid with m rows there will be

$$|\mathcal{P}(2m)| = \binom{2m}{m} \frac{m!}{2^m} \tag{4.5}$$

such classes. For $G_{2,n}$, there are three classes of partial transition sytems: $[l_0, r_0]$, $[l_0, r_1]$, and $[l_0, l_1]$. These are represented graphically in Figure 4.4.

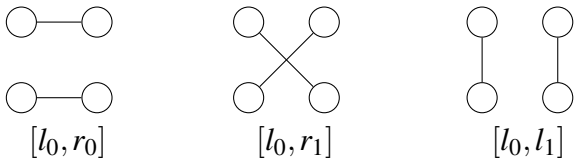

Figure 4.4: Classes of legal partial transition systems

We will make use of these classes extensively in chapter 5.
Considering only partial transition systems on one column, the 3 classes contain the following transition systems: the class $[l_0, r_0]$ includes four different column configurations:

Figure 4.5: Transition systems in class $[l_0, r_0]$

The class $[l_0, r_1]$ includes two column configurations:

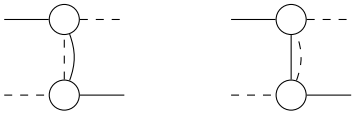

Figure 4.6: Transition systems in class $[l_0, r_1]$

The class $[l_0, l_1]$ contains the two following column configurations:

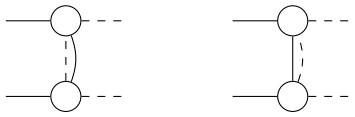

Figure 4.7: Transition systems in class $[l_0, l_1]$

In [13], Golin et al. had suggested that the transfer matrix method could be used to exactly count the number of Euler tours of a grid, however he had not defined such a transfer matrix. Creed used these classes to form such a transfer matrix, which is presented in the following section.

4.3.1.2 Transfer Matrix

The transfer matrix approach is a useful method for counting the number of positive instances for some #*P* problems. Here, it provides a way to evaluate exactly how many Euler tours there are on $G_{2,n}$ and $G_{3,n}$.

The transfer matrix A can roughly be viewed as an adjacency matrix for a graph whose vertices are the classes [26]. There will be an edge between two classes C and C' (in matrix terms, a non-zero value in $A(C,C')$) if there is a way to extend a partial transition on V_k belonging to class C to a partial transition on V_{k+1} belonging to class C' (by extend, we mean adding one column to V_k).

Suppose a partial transition system V_k has class C, and is extended by one column to a partial transition system V_{k+1} which has class C'. Then matrix entry $A(C,C')$ will be the number of ways there are to extend V_k in such a way. The number of edges, and therefore the number in position $A(C,C')$ of the transfer matrix, will then be the number of ways there are to "transfer" from class C to class C'.

The transfer matrix is designed for a grid with a fixed number of rows m. It is an $N \times N$ matrix, where N is the number of classes for $G_{m,n}$. In the case of $G_{2,n}$ we have 3 classes, and for $G_{3,n}$ we will see later that there are 15 classes.

For example, in $G_{2,n}$ consider extending V_k with class $[l_0, r_0]$ to V_{k+1} with class $[l_0, r_1]$. The only way to achieve this is by appending a column with class $[l_0, r_1]$ onto V_k. Then the number in position $A([l_0, r_0], [l_0 r_1])$ will be the number of column configurations there are that will transition $[l_0, r_0]$ into $[l_0, r_1]$, which will be the number of column configurations belonging to class $[l_0, r_1]$. From Figure 4.6, we know that there are 2 such transition systems, and so $A([l_0, r_0], [l_0, r_1]) = 2$.

The transfer matrix for $G_{2,n}$ is shown in full in Figure 4.8.

	$[l_0, r_0]$	$[l_0, r_1]$	$[l_0, l_1]$
$[l_0, r_0]$	4	2	2
$[l_0, r_1]$	2	4	2
$[l_0, l_1]$	0	0	6

Figure 4.8: Transfer matrix for $G_{2,n}$

The transfer matrix can be used to count the number of Euler tours on $G_{m,n}$. Creed showed that the total number of Euler tours on $G_{m,n}$ is given by Equation 4.6, where x and y will be defined below [9].

$$|ET(G_{m,n})| = xA^n y^T. \tag{4.6}$$

Let x be the $1 \times \mathcal{P}(2m)$ vector with entry $x_C = 1$ iff C is the class with only horizontal lines. In the case of $G_{2,m}$, this will be $[l_0, r_0]$. This can be thought of as beginning the grid with m horizontal trailing edges. Then xA^n is obtained by post-multiplying the transfer matrix n times to x. We can visualise this as extending the trailing edges to a partial transition system over all columns of the grid, but without joining the trailing edges of the first and last columns.

Finally, let y be the $1 \times \mathcal{P}(2m)$ vector with entry $y_C = 1$ if and only if joining the trailing edges l_i and r_i in a tour with class C would give a single cycle. Visually, this means $y_C = 1$ if joining the trailing edges of the first and last columns would induce an Euler tour. For example, re-consider the Euler tour in Figure 4.3. Joining the trailing edges to form a transition system over $G_{2,5}$ would induce an Euler tour.

Post-multiplying y^T to xA^n ensures that only Euler tours are counted. Hence the total number of Euler tours can indeed be calculated by Equation 4.6. It is clear that for any given n, we can compute A^n in $O(n)$ steps, hence giving a linear time algorithm. Using the transfer matrix approach, Creed calculate the number of Euler tours for $G_{2,n}$, to be $|ET(G_{2,n})| = (2n+3)6^{n-1} - 2^{n-1}$.

For the case $G_{3,n}$, Creed's transfer matrix was found to have errors, which were later corrected by Astefanoaei [3] and Marinov [22].The index with details of different partial transition classes is given in Figure 4.10 and the fixed transfer matrix is shown in

$$\begin{bmatrix} A & B & C_2 & C_2 & C_1 \\ B & A & C_2 & C_3 & C_4 \\ 0 & 0 & D & 0 & 0 \\ 0 & 0 & 0 & D & 0 \\ 0 & 0 & 0 & 0 & D \end{bmatrix}$$

Figure 4.9: Transfer matrix for $G_{3,n}$

Figure 4.9, where

$$A = \begin{bmatrix} 6 & 1 & 1 \\ 1 & 6 & 1 \\ 1 & 1 & 6 \end{bmatrix} \quad B = \begin{bmatrix} 2 & 2 & 2 \\ 2 & 2 & 2 \\ 2 & 2 & 2 \end{bmatrix} \quad C_1 = \begin{bmatrix} 2 & 1 & 1 \\ 1 & 2 & 1 \\ 1 & 1 & 2 \end{bmatrix} \quad C_2 = \begin{bmatrix} 2 & 1 & 1 \\ 1 & 1 & 2 \\ 1 & 2 & 1 \end{bmatrix} \quad C_3 = \begin{bmatrix} 1 & 2 & 1 \\ 2 & 1 & 1 \\ 1 & 1 & 2 \end{bmatrix}$$

$$C_4 = \begin{bmatrix} 1 & 2 & 1 \\ 1 & 1 & 2 \\ 2 & 1 & 1 \end{bmatrix} \quad D = \begin{bmatrix} 10 & 6 & 6 \\ 6 & 10 & 6 \\ 6 & 6 & 10 \end{bmatrix}.$$

Matrices A, B, and C_1 are the same as in the original transfer matrix developed by Creed, while the rest were later corrected by Astefanoaei and Marinov. Note that since the matrix A is a 15×15 matrix, we can compute A^n in $O(n)$ time.

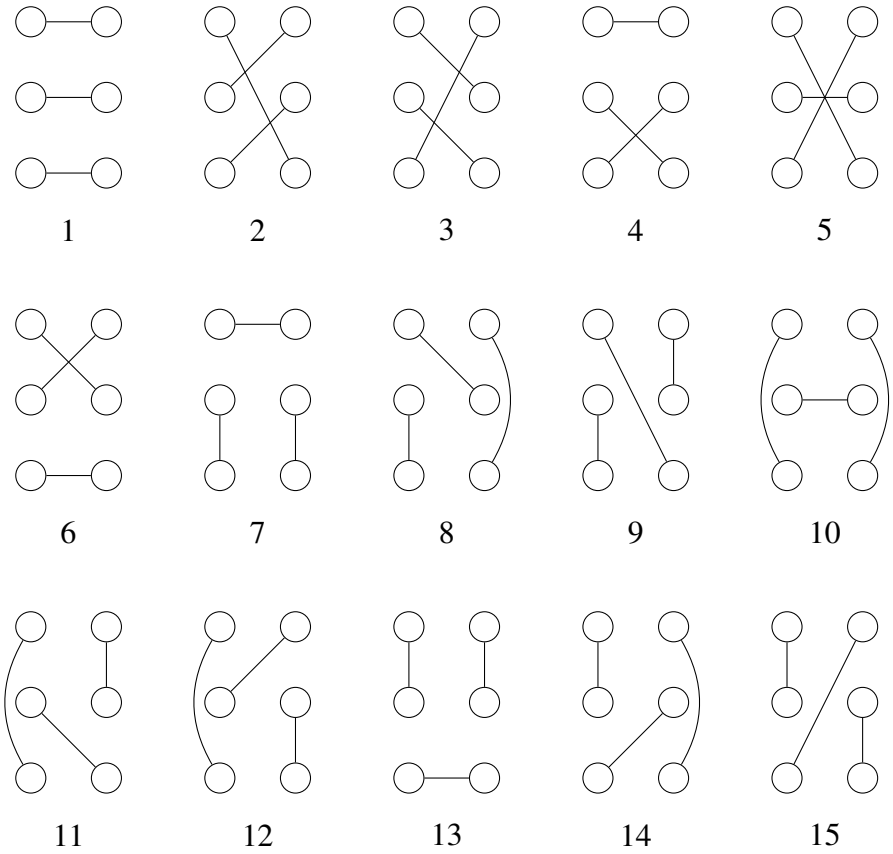

Figure 4.10: Index for the classes of $G_{3,n}$

4.3.2 Discussion

When considering the number of Euler tours on the general $G_{m,n}$ grid, Creed was able to show that the transfer matrix method would have exponential running time. For $G_{2,n}$ and $G_{3,n}$ the algorithm is linear in the number of columns, however as m increases, the number of rows and columns in A increase exponentially. Without finding some extra structure in A, we would not be able to calculate A^n efficiently. Therefore the calculation $\mathbf{x}A^n\mathbf{y}^T$, and therefore the number of Euler tours of $G_{m,n}$ requires exponential time to compute.

For example, by Equation 4.5, for $G_{4,n}$ there are 105 classes, meaning that the transfer matrix for $G_{4,n}$ is an 105×105 matrix. Computing the n^{th} power of such a matrix would not be feasible [9].

Although the work due to Creed presented in this section does not present a method to uniformly at random sample from $G_{m,n}$ in polynomial time, it does introduce some of the ideas utilised in our method for randomly sampling from $G_{2,n}$ in Chapter 5. In

particular, we will be using the classes given in Figure 4.4, and discuss how they interact with each other. This was inspired by Creed's use of the classes to build a transfer matrix; as a by-product showing which pairs of classes are compatible (by which we mean: the pairing of legal partial transition systems in these classes gives rise to another legal partial transition system) and which are incompatible. In essence, Creed's work showed that a lot of the necessary information about a legal partial transition system is held solely in its class.

Chapter 5

Mixing Time of $G_{2,n}$

5.1 Introduction

In this chapter we will present a method for approximate counting the number of Euler tours of $G_{2,n}$. To do so, we sample from the set Ω of all Euler tours of $G_{2,n}$, uniformly at random using the Markov chain Monte Carlo method with the Kotzig chain $\mathfrak{M}$ of Definition 4.1, and repeated below in Definition 5.1. In order to show that the Kotzig chain is rapidly mixing for $G_{2,n}$, we use the canonical paths method of Section 3.2. Thereby, we show that we can sample from Ω, and therefore approximately count Ω, in polynomial time.

5.1.1 Sampling and Counting

Recall that for self-reducible problems, the existence of a FPRAS to approximate the number of solutions to the problem is equivalent to randomly sampling from the set of all possible solutions in polynomial time [18].
We show that the problem of counting Euler tours on a graph is indeed self-reducible.

Lemma 5.1. *Counting Euler tours on a graph is a self-reducible problem [11].*

Proof. At any vertex v of the graph, there are $\binom{deg(v)}{2}$ choices for the transition system of v. We consider each of these transition systems individually, and form a new graph G' which does not contain vertex v, and which places an edge between two vertices paired by the transition system at v. An example of this for a 4-regular graph is presented in Figure 5.1. Picking the transition system that induces the paths $...(a)e_1(v)e_2(b)...$ and $...(c)e_3(v)e_4(d)...$, we will create a new graph G' with an edge between a and b and an edge between c and d.

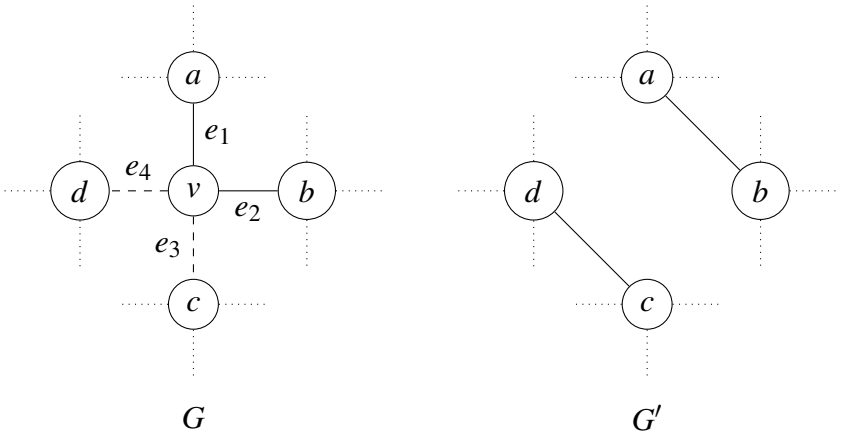

Figure 5.1: Illustration of the self-reducibility property for counting Euler tours

In this way, the problem is reduced from a single problem instance on a graph with N vertices to multiple problem instances on a graph with $N-1$ vertices. These multiple problem instances are disjoint (do not have any overlap in their respective solution sets), as two Euler tours cannot be the same if their transition systems on vertex v are different. Furthermore they partition the set of solutions, since any Euler tour must give v one of the possible transitions. Therefore the sum of the number of solutions to each of the smaller problem instances is equal to the number of solutions to the full problem instance.
Therefore the problem of counting Euler tours is self-reducible. Notice that this maintains the n-regularity of the graphs G and G'. If a vertex u was not incident to v, then the incident edges to u have not been changed and so the degree of u is maintained. If u was incident to v, then the edge from u to v has been removed but another edge incident to u has been added. Thus no matter what, the degree of u will still be n. Therefore in fact, the problem of counting Euler tours on a n-regular graph is self-reducible [11].

□

Therefore the problem of counting Euler tours on $G_{2,n}$ does indeed reduce to the problem of sampling uniformly at random from the set of all Euler tours of $G_{2,n}$, which we label Ω. We now proceed to discuss the Kotzig chain, which we will use as part of the MCMC method to uniformly at random sample from Ω.

5.1.2 Kotzig Chain

Recall the Kotzig chain in Definition 4.1 due to Tetali and Vempala [28], defined for a general $G_{m,n}$ grid. We use the same chain defined only for $G_{2,n}$, given in Definition

5.1.

Definition 5.1. *Let the Kotzig Chain $\mathfrak{M}$ be a Markov chain with state space Ω being the set of all Euler tours of $G_{2,n}$. Start at an arbitrary Euler tour X in its expanded form, and pick some vertex v u.a.r. Then:*

- *with probability $\frac{1}{2}$ perform the unique legal κ-transformation at v;*
- *with probability $\frac{1}{2}$ do nothing.*

Recall from Section 4.2.1 that the unique stationary distribution for $\mathfrak{M}$ is the uniform distribution. Therefore we can indeed use the Markov chain Monte Carlo method to sample uniformly at random from $G_{2,n}$ using $\mathfrak{M}$. The remainder of this chapter will focus on showing that we can do so in polynomial time, by proving that the Kotzig chain is rapidly mixing for $G_{2,n}$. The proof technique that we will employ in this chapter will be of a similar structure, however we succeed in provably defining a method for ordering the vertices.
To aid in the explanation of the method we will use, we introduce some new terminology.

5.1.3 Definitions

Definition 5.2. *On a grid with n columns, the complement block to column k is the block of $n-1$ columns with column k removed, and with trailing edges retained.*

Where we have an Euler tour X over $G_{2,n}$, we always constrain the vertices in the complement block to retain their transition systems in X. Then the complement block is a legal partial transition system, as by definition it can be extended to an Euler tour of the whole grid.
For example, examine the Euler tour X in Figure 5.2 where the transition system at each vertex pairs the dashed lines together and the solid lines together.

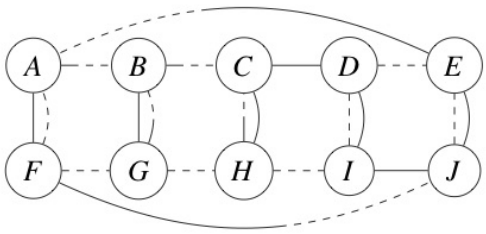

Figure 5.2: Square grid on a torus.

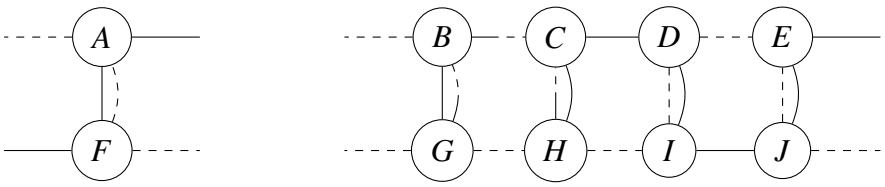

Figure 5.3: Column 1 and its complement block for Euler tour X

The Euler tour X shown in Figure 5.2 can be split into two blocks: Column 1 and its complement block, where column 1 contains the vertices A and F. These are shown in Figure 5.3.

Next, to simplify our discussion of vertices and their incident edges, we will introduce a labelling system for the edges and vertices of column k, illustrated in Figure 5.4.

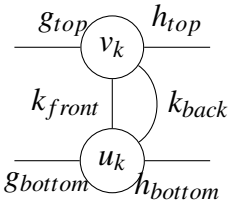

Figure 5.4: Illustration of labelling system

Remark. *We label a column k with its configuration in Euler tour X as* $X(k)$.

Another useful notation relates to the class $[l_0, r_0]$. Consider the four column configurations that belong to class $[l_0, r_0]$. These are given in Figure 5.5.

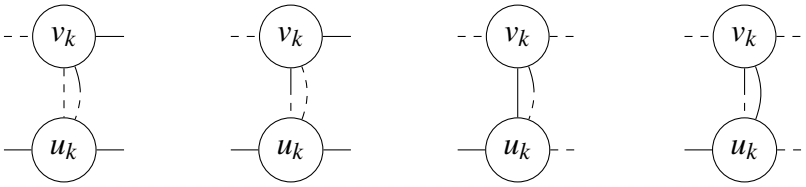

Figure 5.5: Transition systems of a column in class $[l_0, r_0]$

Notice that in each of the column configurations belonging to class $[l_0, r_0]$, there is a path which traverses the vertices in the order either u_k, v_k, u_k or v_k, u_k, v_k. Recall the definition of interleaving in Definition 2.18. For the former ordering, vertex u_k *only* interleaves with v_k. For the latter ordering, v_k *only* interleaves with u_k.

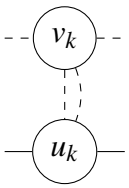

Figure 5.6: Column of class $[l_0, r_0]$

Definition 5.3. *For a column k of class $[l_0, r_0]$ containing two vertices v_k and u_k, we call a vertex v_k a closed vertex if it can **only** interleave with u_k in the expanded representation of the Euler tour. In this case, we will call u_k an open vertex, since it possibly interleaves with other vertices outside of column k.*

An illustration of this is shown in Figure 5.6, where v_k is the closed vertex and u_k is the open vertex. Note that two of the column configurations belonging to class $[l_0, r_0]$ will have u_k as the closed vertex, and two will have u_k as the open vertex.
Finally, we introduce terminology to talk about the *trajectory* of an Euler tour.

Definition 5.4. *Suppose a path enters a column from the left and leaves at the right (or vice versa). We say that its **trajectory** has been maintained. If the path enters the column from the left and leaves at the left (or respectively right), the trajectory of the path has been switched.*

Consider the four column configurations belonging to the class $[l_0, r_0]$ shown in Figure 5.5. A path entering any of these possible column configurations will have its trajectory maintained. The two column configurations belonging to class $[l_0, r_1]$ also maintain the trajectory of a path, while the column configurations belonging to class $[l_0, l_1]$ switch the trajectory of any path traversing them. This can be seen from Figures 4.5, 4.6, and 4.7 on page 35.
We now proceed to explain how the structure of $G_{2,n}$ can be exploited to bound the mixing time of the Kotzig chain.

5.2 Method

We use the canonical paths method of Section 3.2 to bound the mixing time of the Kotzig chain for $G_{2,n}$. Recall that the state space graph G is the graph whose vertices are Euler tours of $G_{2,n}$, and has an edge between two Euler tours X and X' if X can be converted into X' with one κ-transformation. We wish to define an "ordering" on the vertices that converts Euler tour X into Euler tour Y. This ordering will induce a path

in the state space graph from X to Y. Our aim is to create a set of paths between all pairs of Euler tours that do not overload any single edge of the state space graph.

An outline of our ordering is as follows. Determine which vertices have different transition systems in X and Y, and form a difference set d containing all such vertices. Indicate which columns contain any vertices in the difference set d and call these the **difference columns**. Starting with Euler tour X, these columns will have their configurations fixed to match Y in order from left to right. We may require a helper to fix some of the vertices in the columns, but we will show that for the path X to Y, we can always use a helper vertex in the next difference column. In this way, we will be able to decrease the number of difference columns to zero as we sweep through the grid from left to right.

By Theorem 2, it is possible to convert an Euler tour X into another Euler tour Y on $G_{2,n}$ using a finite number of κ-transformations. Therefore, when we define our ordering on the vertices, we are not doing so to show that X can be converted into Y. Rather, we are fixing an ordering that accommodates a *complementary tour D*.

To show that our rules for ordering the vertices do not create any overloaded edges, we form a complementary tour D, similar to the complementary tour used by Tetali and Vempala [28] explained in Section 4.2.1. Suppose that in our path from X to Y in G we traverse the sequence of tours $X, Z_1, Z_2, ..., Z_{j-1}, Y$. By design of the state space graph, these will all be Euler tours. Suppose at Euler tour Z_i the vertices in columns up to column k have their transition systems in Y, and vertices in columns after k generally (with a couple of exceptions) have their transition systems in X. Then in our complementary tour D, roughly speaking, the vertices in columns up to column k will have their transition systems in X while the vertices in columns after k will generally have their transition systems in Y.

In Section 5.2.3, we will explain how we can construct such a complementary tour D. As an overview, suppose in our path from X to Y, we reach some Euler tour Z_i where all columns up to column k have their transition systems in Y, and all columns after k generally (with the possible exception of one vertex) have their transition systems in X. Then to construct D, we will begin with Euler tour Y and sweep through the columns of the grid from left to right up to column k, changing their configurations to those in X. This is subversion of our path from X to Y, and allows us to form a complementary tour D with our desired properties.

Finally, we will show that with the aid of this complementary tour D, we are able to bound the edge loading. To explain our method of ordering the vertices, it will first

be necessary to discuss the classes of legal partial transition systems, as introduced by Creed [9].

5.2.1 Classes

Recall from Section 4.3.1.1 that there are three classes of legal partial transition systems for $G_{2,n}$. These classes are $[l_0, r_0]$, $[l_0, r_1]$, and $[l_0, l_1]$. A class for $G_{2,n}$ represents how a transition system over a block of columns pairs the trailing edges l_0, l_1, r_0, and r_1 which connect to the complementary block. For example, see the two blocks shown in Figure 5.3 on page 44. Column k belongs to class $[l_0, r_1]$ while its complement block belongs to class $[l_0, r_0]$. We present a graphical representation of these classes in Figure 5.7.

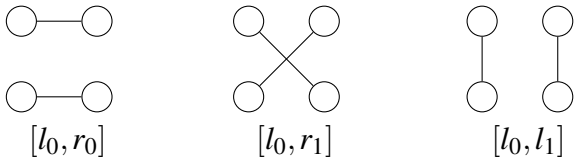

Figure 5.7: Classes of legal partial transition systems for $G_{2,n}$

For reference, we present again the 8 possible column configurations in Figures 5.8, 5.9, and 5.10.

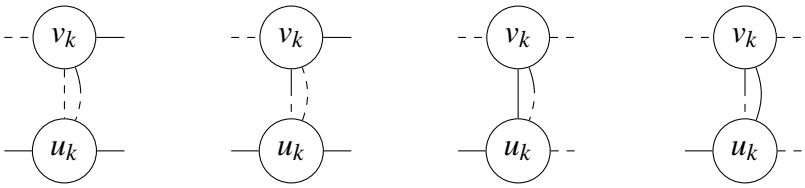

Figure 5.8: Column configurations in class $[l_0, r_0]$

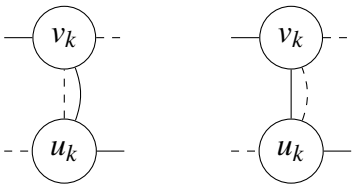

Figure 5.9: Column configurations in class $[l_0, r_1]$

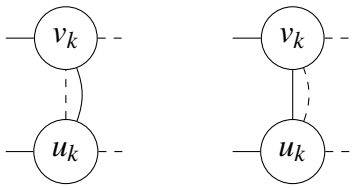

Figure 5.10: Column configurations in class $[l_0, l_1]$

We wish to show that only certain combinations of complement blocks and column configurations induce Euler tours, and are therefore compatible. By establishing this, we are then able to prove (in Section 5.2.2) that it is possible to realise this new column configuration with a few κ-transformations on vertices of that column if its class is compatible with the class of the complement block.

Definition 5.5. *A column configuration and its complement block are said to be compatible if joining their trailing edges forms an Euler tour.*

Note. *For the remainder of this section, we will discuss how to change a column configuration of the current Euler tour Z from its configuration in Z to its configuration in Y. However, the exact same technique will be used to construct D in the path from Y to X, unless specifically stated otherwise.*

We will show that we can change columns from their configurations in Z to their configurations in Y in a sequence using κ-transformations. If a column configuration and its complement block are compatible, then we can apply Lemma 4.1 of Tetali and Vempala to form such a sequence. Hence in the next few pages, we show when a column configuration and its complement block are compatible, and when extra work must be done.

5.2.1.1 Complement Block Class $[l_0, r_0]$

Suppose the complement block of our current Euler tour has class $[l_0, r_0]$. No columns in the complement block may have class $[l_0, l_1]$, as otherwise this would induce the whole block to have class $[l_0, l_1]$. Therefore in this case all columns in the complement block maintain the trajectory of a path. The transition systems on the complement block must create a path from g_{bottom} to h_{bottom} and from g_{top} to h_{top}.
Therefore we must have two paths, which necessarily must traverse all edges and vertices in the complement block (for else our original tour was not an Euler tour). The transition systems over the complement block must not induce disjoint cycles, or the

complement block may never be extended to an Euler tour. Then we have the following paths over the complement block:

$$h_{top}(v_{k+1})....(v_{k-1})g_{top}; \quad h_{bottom}(u_{k+1})...(u_{k-1})g_{bottom}.$$

All column configurations for k belonging to the class $[l_0, r_1]$ will connect h_{bottom} to g_{top} and connect h_{top} to g_{bottom} as can be seen from Figure 5.9. All column configurations for k belonging to the class $[l_0, l_1]$ will connect h_{top} to h_{bottom} and g_{top} to g_{bottom}, as can be seen from Figure 5.10. Then all such column configurations will connect the paths, forming one single cycle. We can see from Figure 5.8 that all columns in class $[l_0, r_0]$ will create two disjoint cycles. Therefore we conclude that column configurations belonging to classes $[l_0, r_1]$ and $[l_0, l_1]$ are compatible with a complement block belonging to class $[l_0, r_0]$.

5.2.1.2 Complement Block Class $[l_0, r_1]$

Suppose that the complement block belongs to class $[l_0, r_1]$. All columns in the complement block preserve the trajectory, and since the transition systems on the block pair g_{top} and h_{bottom}, the complement block will induce two edge-disjoint paths:

$$h_{top}(v_{k+1})...(u_{k-1})g_{bottom}; \quad h_{bottom}(u_{k+1})...(v_{k-1})g_{top}.$$

Consider the two column configurations for the class $[l_0, r_1]$ shown in Figure 5.9. Giving column k either of these configurations would connect v_{k+1} to u_{k-1} and v_{k-1} to u_{k+1}, forming two disjoint cycles.
Conversely, if column k has any of the six column configurations in classes $[l_0, r_0]$ and $[l_0, l_1]$, then this configuration together with the complement block will form one single cycle. A column configuration belonging to class $[l_0, r_0]$ would connect g_{top} to h_{top} and g_{bottom} to h_{bottom}, thus forming an Euler tour. Similarly a column configuration belonging to class $[l_0, l_1]$ would connect h_{top} to h_{bottom} and g_{top} to g_{bottom}, connecting the two paths. For evidence, see Figures 5.8 and 5.10.
Therefore column configurations $[l_0, r_0]$ and $[l_0, l_1]$ are compatible with a complement block belonging to class $[l_0, r_1]$.

5.2.1.3 Complement Block Class $[l_0, l_0]$

Suppose in our current Euler tour Z the complement block to column k has class $[l_0, l_1]$. In order for the trailing edge l_0 to be paired with l_1, it must be the case that there is a

column belonging to class $[l_0, l_1]$ in the complement block. Note that there may only ever be one column belonging to class $[l_0, l_1]$ in an Euler tour, as otherwise the tour would contain disjoint cycles. Then exactly one column $j(\neq k)$ will have class $[l_0, l_1]$ in the complement block of Z. All columns in the complement block besides column j will maintain the trajectory of the path, and column j will switch the trajectory. Then any path which departed column k via the edge g_{top} must revisit column k via the edge g_{below}, or vice versa. Similarly for h_{top} and h_{bottom}.
Therefore the structure of the Euler tour on the complement block to column k must induce two edge-disjoint paths:

$$h_{top}(v_{k+1})...(u_{k+1})h_{bottom}; \quad g_{top}(v_{k-1})...(u_{k-1})g_{bottom}.$$

The paths must include every edge in the complement block besides g_{top}, g_{bottom}, h_{top}, h_{bottom}, k_{front} and k_{back} exactly once for Z to have been an Euler tour. Furthermore, they must not contain any disjoint cycles or again, Z could not have been an Euler tour.
Now suppose we have a new proposed configuration for column k, $Y(k)$. We want to examine whether it is compatible with Z's complement block.
When considering how the transition systems of v_k and u_k can be arranged, we must consider how to connect the two paths in order to create a single cycle. This could not be achieved if column k had class $[l_0, l_1]$, as the tour would be split into two disconnected cycles.
To form one cycle, the vertices u_k and v_k need to have transition systems that connect the path on the left of k to the path on the right. All possible combinations of transition systems $Y(k)$ where column k has class $[l_0, r_0]$ or $[l_0, r_1]$ will achieve this.
Therefore we conclude that column configurations for column k belonging to classes $[l_0, r_0]$ and $[l_0, r_1]$ are compatible with a complement block belonging to class $[l_0, l_1]$.

Remark. *Notice that in all cases, the column configurations and complement blocks are only incompatible if they have the same class.*

Remark. *Since we have shown that a column configuration and a complement block are compatible depending on their classes, we will extend our definition of "compatible" to refer also to compatible classes (as well as transition systems).*

5.2.2 Changing Column Configurations

Recall Lemma 4.1, which supposes we have two Euler tours X and Y. The lemma states that if we have some vertex v where $TS(X) \not\Rightarrow_v TS(Y)$, then we must have some

other vertex $u(\neq v)$ which has a different transition system in X and Y and furthermore, in at most 3 κ-transformations, we will be able to change both v and u from their transition systems in X to those in Y, without changing the transition systems of any other vertices.

We use this Lemma to show that if the current complement block to column k and the new proposed column configuration for k are compatible, then we can substitute in the column configuration without the need for a helper vertex outside of column k.

Suppose on our path from X to Y we have corrected all difference columns up to $(k-1)$ to have their configurations in Y, and we have now reached Euler tour Z. We now wish to change the column configuration of column k to $Y(k)$, by using κ-transformations. In this context we present Lemma 5.2.

Lemma 5.2. *Suppose some Euler tour Z induces the complement block to column k to have a certain class which we label A. Suppose further that we wish to give column k a configuration belonging to class B. If the class B is compatible with A, then we will be able to achieve this using at most 3 κ-transformations at the vertices of column k.*

Proof. Suppose on our path from X to Y on the state space graph we reach an Euler tour Z which induces the complement block to column k to have class A. Suppose further that the column configuration for column k in Y belongs to class B, which is compatible with A (in other words, different from A). Now since we assume that the classes B and A are compatible, we know that joining the trailing edges of the complement block with the column configuration of k in Y will induce an Euler tour. Call this Euler tour Z'.

If the transition systems of both vertices in k were the same in Z and Y then we would not need to perform any κ-transformations, and the lemma would hold vacuously. Therefore we assume that at least one vertex in column k has a different transition system in Z and Y.

Suppose first that in the column configuration of k in Z, one vertex s matches its transition system in Y (and therefore also in the successive Euler tour Z'), and the other vertex t differs. Since Z' is an Euler tour, then given the statement of Lemma 4.1, the transition system of t in $TS(Y,t)$ must be the available transition at t in Z. Therefore $TS(Z) \Rightarrow_t TS(Y)$, and we can substitute in the column configuration of k in Y using a single κ-transformation at t.

Suppose instead that both vertices s and t in k have different transition systems in Z and Y. Suppose first that $TS(Z) \Rightarrow_s TS(Y)$. Consider Lemma 4.1 which guarantees that

we can connect Z to Z' using κ-transformations on s and t only. Then performing the κ-transformation at s will result in some Euler tour $\hat{Z}$ which has only t as a difference vertex with Z'. Then by the explanation above, it must be the case that $TS(\hat{Z}) \Rightarrow_t TS(Y)$, and so we can transform Z into Z' using two κ-transformations: first at s, then at t.

Now suppose again that both vertices s and t are difference vertices for Z and Y, but that $TS(Z) \not\Rightarrow {}_sTS(Y)$. Then by Lemma 4.1, there must be some vertex v in the difference set such that both vertices can have their transitions fixed to match Y using at most 3 κ-transformations. Since the only other vertex in the difference set is t, it must be the case that $t = v$. Then starting at Euler tour Z we can perform at most 3 κ-transformations on s and t to give s and t their transition systems in Y, and not perform a κ-transformation on any other vertices, as required. □

Lemma 5.2 shows that we can substitute in a column configuration using 1-3 κ-transformations on vertices of that column if its class is compatible with the class of the complement block. However it will not always be the case that the column configuration we wish to substitute in will always be compatible with the complement block. We now need to consider situations where the classes aren't compatible.

Suppose we reach intermediate Euler tour Z in our path from X to Y and that at this point, all columns up to $(k-1)$ have their configurations in Y. If the column configuration of k in Y is not compatible with the class of the complement block in Z, we will require a "helper vertex" that will change the class of the complement block. We can show that there exists such a helper, located in one of the difference columns after column k. This is achieved in Lemma 5.3. In nearly all cases, such a vertex will be located in the next difference column to the right. The one exception to this will be if we require a **special helper vertex**. This occurs if the complement block has class $[l_0, l_1]$ in Z and the column configuration for k has class $[l_0, l_1]$ in Y. We will explain "special helper vertices" and their location in detail in Part A of the proof of Lemma 5.3.

Lemma 5.3. *Suppose we have two Euler tours Z and Y. Suppose the column configuration of k in Y is not compatible with the class of the complement block to k in Z (in other words, they have the same class). Then there will exist a vertex h in one of the difference columns which, after performing a κ-transformation on h, will make the column configuration of k in Y compatible.*

The proof of Lemma 5.3 will be split into parts A and B. Recall from Remark 5.2.1.3

that a new column configuration and the existing complement block may only be incompatible if they belong to the same class. Part A will prove the lemma for when the column configuration for k belongs to class $[l_0, l_1]$ in Y, and so does the complement block in Z. Part B will prove the Lemma when the class of the column and complement are either both $[l_0, r_0]$ or both $[l_0, r_1]$.

Part A. Let $[l_0, l_1]$ be the class of the complement block in Z and let $[l_0, l_1]$ be the class of the new column k from Y. Recall that the complement block belongs to class $[l_0, l_1]$ if and only if exactly one column j in the complement block belongs to class $[l_0, l_1]$. Note that if the class of j was changed, the complement block would no longer belong to class $[l_0, l_1]$. This would result in the complement block in Z' having class $[l_0, r_0]$ or $[l_0, r_1]$, and then this would be compatible with the new column configuration for k from Y.

We show that there exists a **special helper vertex** h located in column j such that after performing a κ-transformation on h, the classes of the complement block will change, making the classes of k and the complement block compatible.

Consider column j. It may have two possible column configurations, which were presented graphically in Figure 5.10 on Page 48. These give either of the following paths over the incident edges of column j:

$$g_{top}(v_j)\, j_{back}(u_j)\, g_{bottom}; \quad h_{top}(v_j)\, j_{front}(u_j)\, h_{bottom},$$

or

$$g_{top}(v_j)\, j_{front}(u_j)\, g_{bottom}; \quad h_{top}(v_j)\, j_{back}(u_j)\, h_{bottom}.$$

Notice that in both possibilities, performing a κ-transformation at either vertex u_j or v_j would change the class of the column configuration, as required. Therefore we arbitrarily pick v_j as our helper vertex.

Finally, we notice that v_j must have been located in a difference column. Suppose not. Then both j and k have class $[l_0, l_1]$ in Y, contradicting the fact that Y is an Euler tour.

We now proceed to prove the lemma for when either the complement block and column k both belong to class $[l_0, r_0]$, or both belong to class $[l_0, r_1]$.

Part B. Suppose we have column k with class A in Y and with complement block belonging to class A in Z where $A \in \{[l_0, r_0], [l_0, r_1]\}$, so that the new column k and the existing complement block are incompatible. Recall that the difference columns are

the columns which contain one or two difference vertices for Z and Y. Label the next difference column to the right of column k as column j. We wish to show that, given a column k belonging to class A in Y and a complement block belonging to class A in Z, we can perform a κ-transformation on some vertex h in column j of Z to make $Y(k)$ and Z's complement block compatible. Therefore, by Remark 5.2.1.3, we wish to show that a κ-transformation at h changes the class of the complement block *away* from A.

The complement block of Z does not belong to class $[l_0, l_1]$, and so does not contain any columns belonging to class $[l_0, l_1]$. Thus each of the columns in the complement block in Z either belong to class $[l_0, r_0]$ or $[l_0, r_1]$. We show that performing a κ-transformation on a vertex h in the complement block which changes the parity of the number of columns belonging to class $[l_0, r_1]$, or changes j to a column with class $[l_0, l_1]$, will change the class of the whole complement block.

Recall that our framework for Part B stipulates that the complement block belongs to class $A \in \{[l_0, r_0], [l_0, r_1]\}$, so that no columns in the complement block may have class $[l_0, l_1]$. Suppose we change the column configuration of j in the complement block to have class $[l_0, l_1]$. Then the complement block will also belong to class $[l_0, l_1]$, and so the class of the whole complement block will have been changed.

Now suppose we have an even number of columns in the complement block that belong to class $[l_0, r_1]$. Label these $C_1, ..., C_{2b}$. Pair together C_1 with C_2, etc, and consider one arbitrary pair (C_i, C_{i+1}). With a slight abuse of notation we call the vertices in C_i v_i and u_i. Any path that begins $g_{top}(v_i)$ must reach column C_{i+1} via the path segment $g_{bottom}(u_{i+1})$. Such a path must leave column C_{i+1} via the path segment $(v_{i+1})h_{top}$. Then the path traversing C_i and C_{i+1} takes the form $g_{top}(v_i)...(v_{i+1})h_{top}$, and so the block of the graph starting before column C_i and ending after column C_{i+1} has class $[l_0, r_0]$. Therefore a complement block with an even number of $[l_0, r_1]$-columns (and no $[l_0, l_1]$ columns) will have class $[l_0, r_0]$. By similar reasoning, a complement block with an odd number of columns belonging to class $[l_0, r_1]$ will belong to class $[l_0, r_1]$. Therefore, changing the class of a column in the complement block to $[l_0, l_1]$ or changing the parity of the number of columns that belong to class $[l_0, r_1]$ will change the class of the complement block.

We now show that there exists a κ-transformation on a vertex in column j that will change the class of the complement block. Suppose j belongs to class $[l_0, r_0]$. Recall from Definition 5.3 that all columns belonging to this class contain one closed vertex and one open vertex. We observe that performing a κ-transformation on the closed

vertex in j will only ever swap the order that k_{front} and k_{back} appear in the tour, and will not change the class of j. See Figure 5.8 for visual proof. Then we focus instead on the open vertex h. Depending on the rest of the tour, the κ-transformation at h will either change the class of j from $[l_0, r_0]$ to $[l_0, l_1]$ or to $[l_0, r_1]$, depending on which transition was available.

Suppose instead that j belongs to class $[l_0, r_1]$, and suppose we picked the top vertex v_j as our helper vertex h (this is an arbitrary choice, in fact our reasoning will hold true for either vertex). We can observe from Figure 5.9 on Page 47 that a κ-transformation on h will either change the class of j from $[l_0, r_1]$ to $[l_0, l_1]$ or $[l_0, r_1]$, depending on which transition was available. This is because at both vertices, giving the vertex either of its alternative transition systems will change the class of j.

Thus a κ-transformation at some vertex h of the next difference column j will indeed either change the parity of the number of columns in the complement block that belong to class $[l_0, r_1]$ in Z, or will change the class of column j to $[l_0, l_1]$. Therefore a κ-transformation at h will always change the class of the complement block away from A, proving the lemma. □

Lemma 5.3 tells us that in nearly all cases, we can perform a κ-transformation on a vertex h in the next difference column of Z and Y (more-or-less equivalent to the next difference column of X and Y) that will make the complement block and the column configuration at k compatible.

Supposing we have a column configuration of k in Y that is incompatible with the complement block in Z, then crucially, this means that after transforming h, we can use Lemma 5.2 to show that the column configuration of k in Y can be substituted into Euler tour Z in a finite number of κ-transformations only in column k, obtaining Euler tour Z'.

After substituting in $Y(k)$, the helper vertex h will have a different transition system to what it had in X (and Z). Therefore comparing $Z'(j)$ to $Y(j)$, it might be that $Z'(j)$ is already matching $Y(j)$, or that $Z'(j)$ is different again to both $Y(j)$ and $X(j)$. However column j will be the next column to have its configuration changed to its configuration in Y, so it does not affect our ordering that its configuration may be neither $X(k)$ nor $Y(k)$.

Thus we have shown that we can order the vertices so that when fixing column k to have the new configuration $Y(k)$, unless a special helper is used, all κ-transformations will either be in column k, or there may be one κ-transformation in the next difference

column to help fix column k. Finally, we wish to discuss how we can minimise the number of times a special helper is used.

5.2.2.1 Minimising usage of a special helper vertex

Recall that a special helper vertex is used if the class of the complement block to column k in Z is $[l_0, l_1]$ and the class of the new proposed column configuration of k for Y is $[l_0, l_1]$. We can in fact perform an initial rotation of the grid that ensure that in the path from X to Y, this never occurs.

Before we begin any transformations on the vertices, we first rotate the grid so that if there is a column in X belonging to class $[l_0, l_1]$, it is positioned as column 0. There may only be at most one such column.

Recall that in our ordering of the vertices, we sweep though the grid from left to right changing difference columns from their configurations in Z to their configurations in Y. By rotating the grid, we ensure that if a column in X has class $[l_0, l_1]$, it has its configuration switched to that in Y immediately.

Suppose column 0 belongs to class $[l_0, l_1]$ in Y. Then our rotation ensures that the complement block in X must not belong to class $[l_0, l_1]$, and the column configuration of 0 in Y will be compatible with the class of the complement block.

Suppose that column 0 does not belong to class $[l_0, l_1]$ in Y. Then after changing the configuration of 0 in $Z_0 = X$ to become Z_1, we create an Euler tour Z_1 which does not contain any columns belonging to class $[l_0, l_1]$. As we sweep through the grid, when we fix column k, the columns to the left of k will have their configurations in Y, and the columns to the right will generally have their configurations in X. The exception will be a column j in which a vertex h has been used as a helper for column k. We know that all columns in X besides 0 must not have class $[l_0, l_1]$, and so the only column to the right of k that could possibly have class $[l_0, l_1]$ is column j. However column j is the next column we will fix. Therefore column j cannot induce a situation where the required configuration of some column i and the complement block to i in Z both belong to class $[l_0, l_1]$. Then in our path from X to Y, we never require a special helper.

Note that in the path from Y to X, this rotation will not prevent us from ever requiring a helper vertex. However, we will never need to use *more than one* special helper vertex in a path between Y and X. This is because a special helper is only used if the column configuration of k in Y belongs to class $[l_0, l_1]$, but only one column may belong to this class if Y is an Euler tour. Therefore in all but at most one case, the helper vertex will be located in the next difference column.

5.2.2.2 Discussion of ordering

Combining all we discussed in Section 5.2.2, there are two main results. Firstly, we see that in the path from X to Y, column k of the current Euler tour Z can be fixed to have its configuration in Y using only vertices in column k and at most one vertex in the next difference column (and at most 4 k-transformations on these). Secondly, in the path from Y to X, some column k can almost always be fixed using only vertices in column k, and at most one vertex in the next difference column. In the entire grid there may be at most one exception, where column k can be fixed to have its configuration in X using only vertices in column k and a special helper vertex.
This will be useful to us when we bound the mixing time for the following reason. When bounding the mixing time, we wish to show that an edge (Z_i, Z_{i+1}) of the state space graph is never overloaded. To do so, we construct a complementary tour D which, in conjunction with Euler tour Z_i, can be used to reconstruct X and Y. In our method, we will not be able to reconstruct X and Y exactly, but we will be able to limit the number of possibilities for X and Y to be $O(n^2)$. Knowing that the vertices that achieve the "fixing" of column k may only be located in certain columns will be very useful in limiting the number of possibilities for X and Y, as we will see in Section 5.2.3.1. We now proceed to explain how we may construct the complementary tour D.

5.2.3 Complementary Tour D

Suppose on our path from X to Y we reach Euler tour Z_i on the edge (Z_i, Z_{i+1}) which transforms some vertex w_k. All columns up to column k will have their configurations in Y, and all columns after k will generally have their configurations in X. The use of "generally" here is to allow for the one vertex in the next difference column to k which may have been used as a helper to fix the configuration of k. We know that column k is a difference column by construction, for otherwise w_k would not be transformed.
Our aim is to construct a complementary path $D_0 = Y, D_1, ..., D, ..., D_m = X$, with an intermediating tour D, where columns $1, .., k-1$ have had their configurations corrected. D should have the following properties:

- for columns 0 to $(k-1)$, their configurations match X;
- columns $(k+1)$ to $(n-1)$ have their configurations matching Y, with the possible exception of one vertex.

We can construct such a D using almost the exact same technique as we described to construct some Z on the path from X to Y. Starting at Y, we sweep through the columns from left to right, correcting the difference columns by substituting in the column configuration for X if it is compatible with the complement block (this will then be realised in at most 3 κ-transformations from D_i to D_{i+1} to D_{i+2} to D_{i+3}) or using a helper vertex to make them compatible if not, as in Lemma 5.3. The current Euler tour D is constructed so that column k still has its transition system in Y maintained, unless a vertex in k was required as a helper for the previous difference column. In the path from Y to X truncated at D, we have not yet made any attempt to give column k its configuration in X, and so no helper vertices will have been used in columns right of k besides potentially a special helper vertex. Therefore the next difference column to the right of k will not yet have had its configuration changed (unless it contained a special helper vertex), and so all vertices in columns to the right of k will have their configurations in Y, with the possible exception of one vertex that was used as a special helper. We can confirm from the proofs of Lemmas 5.2 and Lemma 5.3 that constructing D in this way will indeed give us our desired properties. Thus we have a method for constructing our complementary tour D.

We now show how we can use such a D to reconstruct X and Y, therefore showing how many possible pairs of Euler tours could use an edge of the state space graph.

5.2.3.1 Reproducing X and Y from D and Z_i

Suppose on the path from X to Y we traverse the edge (Z_i, Z_{i+1}). We need to bound the number of X, Y pairs which do this. We will show how to use the information from Z_i, X, and Y to re-construct the complementary tour D satisfying the properties listed above. Then in order to bound the mixing time, we wish to show that given only (Z_i, Z_{i+1}) and D, we require no more than $8^4n^2 + 64n$ guesses to reproduce X and Y.

Our main tool for reproducing X and Y will be examining the difference columns of Z_i and D. Let d be the difference columns for X and Y and let d' be the difference columns for Z_i and D. First note that the vertex being transformed by the edge (Z_i, Z_{i+1}) must either be located in column k or the next difference column j by our construction of the path from X to Y. Since we do not know which, this gives us two options for the position of column k: the column of the (Z_i, Z_{i+1}) transformation, or the "previous difference column".

Label the columns of X as $X(0), ..., X(n-1)$, and the columns of Y as $Y(0), ..., Y(n-$

1). With an abuse of notation, we write:

$$X = X(0) \cup X(1) \cup ... \cup X(n-1) \tag{5.1}$$

$$Y = Y(0) \cup Y(1) \cup ... \cup Y(n-1). \tag{5.2}$$

Now examine $\hat{D}$ and $\hat{Z}_i$, given as in Equations 5.3 and 5.4:

$$\hat{Z}_i = Y(0) \cup Y(1) \cup ... \cup Y(k-1) \cup X(k) \cup X(k+1) \cup ... \cup X(n-1) \tag{5.3}$$

$$\hat{D} = X(0) \cup X(1) \cup ... \cup X(k-1) \cup Y(k) \cup Y(k+1) \cup ... \cup Y(n-1). \tag{5.4}$$

From Equations 5.1, 5.2, 5.3, and 5.4 we can see that if a column is in the difference set of X and Y, it must also be in the difference set of $\hat{Z}_i$ and $\hat{D}$. Therefore if $Z_i = \hat{Z}_i$ and $D = \hat{D}$, we would have $d = d'$ and we could exactly reconstruct X and Y. We now show that we will need to guess at most 4 columns to reconstruct $\hat{Z}_i$ and $\hat{D}$, and therefore X and Y.

First we show that if the edge (Z_i, Z_{i+1}) transformed a vertex w_j in the next difference column j after k (so w_j is a helper vertex), then we can guess $\hat{Z}_i$ and $\hat{D}$ in $64n$ guesses. Suppose that the edge (Z_i, Z_{i+1}) transformed a vertex w_j in column j. Then we know the exact location of column k, as it is located in the previous difference column (we can examine the columns before i in Z_i and D to find where this column is). Furthermore, we will know the exact location of column j. If w_j is a helper vertex then we know that no vertex in column k has yet been transformed in the current Euler tour Z_i, so we know the configuration of $X(k)$, as it is equal to $Z(k)$. We also know that w_j has the same transition system in Z_i and X. Then $\hat{Z}$ can be reconstructed with no guesses.

We now proceed to guess $\hat{D}$, since this contains the remaining information we need to recover all of X and Y. We know that in D, all columns up to column $k-1$ have their transition systems in X. However, we do not know whether the previous difference column required a helper to achieve this. Supposing it did, then this helper would have been located in column k, and so we require 8 guesses for the configuration of $Y(k)$. Furthermore, we don't know if a previous difference column required a special helper vertex. We need n guesses to locate the column, and 8 guesses for its configuration. Since D is constructed so that no additional helpers have been used, this gives us $64n$ guesses for $\hat{D}$.

Now we show that if the edge (Z_i, Z_{i+1}) transformed a vertex w_k of the current difference column k, then we require $8^4 n^2$ guesses to reconstruct $\hat{Z}_i$ and $\hat{D}$.

Suppose that the edge (Z_i, Z_{i+1}) transformed a vertex w_k of the current difference column k. We do not know whether a helper w_j was required to make column k compatible

with the complement block. Furthermore, supposing that we did use a helper, there is a possibility that the transformation at that helper vertex gave column j its configuration in Y, and therefore j might no longer be a difference column. Therefore we require at most n guesses for the location of column j, and since there are 8 possible column configurations, we require 8 guesses for its configuration. Furthermore, we will need to guess the column configuration for k, giving a further 8 guesses. This gives a total number of $64n$ guesses for $\hat{Z}$. Now we guess $\hat{D}$, in the same way as above. We know the location of column k, so we require 8 guesses for Y_k. Again, some previous difference column may have required a special helper vertex, and we require at most n guesses to determine which column it was located in. Then we require a further 8 guesses for the configuration of that column, giving a total of $64n$ guesses for $\hat{D}$.
Therefore the total number of guesses for $\hat{Z}_i$ and $\hat{D}$, and therefore X and Y, is $8^4n^2 + 64n = O(n^2)$. We now show how we can use this bound on the number of guesses to bound the mixing time of $\mathfrak{M}$, thereby showing that the chain is rapidly mixing.

5.3 Bounding the Mixing Time

In this section, we will show that the mixing time of $\mathfrak{M}$ is bounded by $(2^{12}N^4 + 2^7N^3)(N\ln(3) + \ln(\varepsilon^{-1}))$, where N is the number of vertices in $G_{2,n}$. This implies that we can sample from $\mathfrak{M}$ in polynomial time, and therefore can approximately count the number of Euler tours of $G_{2,n}$ in polynomial time. We prove the bound using the canonical paths method of Section 3.2.
Recall that Sinclair [25] showed that for a finite, reversible, ergodic Markov chain $\mathfrak{M}$ with loop probabilities $P(X,X) \geq \frac{1}{2}$, the mixing time of $\mathfrak{M}$, denoted $\tau_X(\varepsilon)$, satisfies the following inequality:

$$\tau_X(\varepsilon) \leq \rho(ln(\pi(X)^{-1}) + ln(\varepsilon^{-1})) \tag{5.5}$$

where ρ is the maximum edge loading of the state space graph, and π is the uniform stationary distribution. We will use Equation 5.5 to bound the mixing time of $\mathfrak{M}$, by finding a bound on the value of ρ.
From Section 3.2, we have

$$\rho(\Gamma) = \max_e \frac{1}{Q(e)} \sum_{\gamma_{X,Y} \ni e} \pi(X)\pi(Y)|\gamma_{X,Y}|, \tag{5.6}$$

where $|\gamma_{X,Y}|$ is the length of the path $\gamma_{X,Y}$, and $Q(X,Y) = \pi(X)P(X,Y)$. We will go

through each element of this definition of ρ in turn to show how we can construct a bound for ρ.

Let $\eta_e(X,Y) : \{(X,Y) : \gamma_{X,Y} \ni e\} \to \Omega$ map X and Y to D for some edge $e = (Z_i, Z_{i+1})$ of the state space graph. By Section 5.2.3.1, we know that any element in the range of $\eta_e(X,Y)$ can be mapped to by at most $8^4n^2 + 64n$ pairs of X and Y. Now since π is a probability distribution, we know that

$$\sum_{X \in \Omega} \pi(X) = 1. \tag{5.7}$$

Each $D \in \Omega$ may be mapped to at most $8^4n^2 + 64n$ times, so we have the inequality

$$\sum_{\gamma_{X,Y} \ni e} \pi(\eta_e(X,Y)) \leq (8^4n^2 + 64n) \cdot \sum_{X \in \Omega} \pi(X) = 8^4n^2 + 64n. \tag{5.8}$$

We now proceed to show that $|\gamma_{X,Y}| \leq 4n$. Recall that if the configuration of a column is compatible with the complement block, then it can be substituted in using at most 3 moves by Lemma 4.1. Recall further that by Lemma 5.2, if necessary we can make the complement block compatible with the column configuration in a single move. Supposing every column in the grid is a difference column, and that every column configuration is incompatible with the complement block, our ordering of the vertices would require $4n$ transformations. This gives us an upper bound on the length of the path $\gamma_{X,Y}$.

Recall that by the design of the Markov chain, for two incident states A and B in the state space graph, if $A \neq B$ then the probability of transitioning from A to B is $\frac{1}{2}$ times the probability of picking their difference vertex v. This gives

$$P(A,B) = \frac{1}{4n} \tag{5.9}$$

where n is the number of columns in the grid, and so $2n$ is the number of vertices in the grid.

Finally, we observe that

$$\pi(A) = \pi(B) \quad \forall A, B \in \Omega. \tag{5.10}$$

This is because π is the uniform distribution, so by definition Equation 5.10 must hold. Now we have all necessary elements to find a bound on ρ, the maximum edge loading. We calculate the edge loading for an arbitrary edge $e = (Z_i, Z_{i+1})$, and note that since e was an arbitrary choice, the edge loading for any edge will be equal to the edge loading for e, and therefore so will the the maximum edge loading over all edges.

$$\frac{1}{Q(e)}\sum_{\gamma_{X,Y}\ni e}\pi(X)\pi(Y)|\gamma_{X,Y}| = \frac{1}{Q(e)}\sum_{\gamma_{X,Y}\ni e}\pi(Z_i)\pi(\eta_e(X,Y))|\gamma_{X,Y}| \tag{5.11}$$

$$= \frac{1}{\pi(Z_i)P(Z_i,Z_{i+1})}\sum_{\gamma_{X,Y}\ni e}\pi(Z_i)\pi(\eta_e(X,Y))|\gamma_{X,Y}| \tag{5.12}$$

$$= \frac{1}{\pi(Z_i)P(Z_i,Z_{i+1})}\pi(Z_i)|\gamma_{X,Y}|\sum_{\gamma_{X,Y}\ni e}\pi(\eta_e(X,Y)) \tag{5.13}$$

$$= \frac{4n}{P(Z_i,Z_{i+1})}\sum_{\gamma_{X,Y}\ni e}\pi(\eta_e(X,Y)) \tag{5.14}$$

$$\leq \frac{4n}{P(Z_i,Z_{i+1})}(8^4n^2+64n) \tag{5.15}$$

$$\leq \frac{4n}{1/4n}(8^4n^2+64n) \tag{5.16}$$

$$= 2\cdot(8^5n^4+8^3n^3), \tag{5.17}$$

or equivalently $2^{12}N^4+2^7N^3$, where $G_{2,n}$ has $N=2n$ vertices.
Now since each vertex in $G_{2,n}$ has at most 3 possible transition systems, we can form a (very slack) bound on the maximum number of Euler tours of $G_{2,n}$ as 3^N where N is the size of the vertex set of $G_{2,n}$. Then

$$\ln(|\Omega|) \leq \ln(3^N) = N\ln(3). \tag{5.18}$$

Using our bound for ρ in Equation 5.5, we find that the mixing time $\tau_X(\varepsilon)$ is bound by

$$\tau_X(\varepsilon) \leq (2^{12}N^4+2^7N^3)(\ln(|\Omega|)+\ln(\varepsilon^{-1})) \tag{5.19}$$

$$\leq (2^{12}N^4+2^7N^3)(N\ln(3)+\ln(\varepsilon^{-1})). \tag{5.20}$$

This bound is of order $O(N^5,\varepsilon^{-1})$, and is thus polynomial in N and ε^{-1}, as required.

5.4 Discussion

The main focus of this chapter was to present an ordering for the vertices that allowed us to bound the mixing time of the chain. The ordering was based principally on the idea that classes could be compatible or incompatible with each other, and that this could be exploited to show that we could change the column configurations from X to Y (or Y to X) in a sweep from left to right.
Ultimately, the reason we wanted to show that we could achieve a path from X to Y that swept through the columns was so that we could accommodate a complementary

tour D. By designing the path in such a way, a complementary tour D could be created naturally by using an almost mirror image of the technique, such that D possessed the required properties that allowed us to reconstruct X and Y. With our bound on the number of possible ways we could reconstruct X and Y, we were able to show that the mixing time of the chain was polynomial in N and ε, and so the number of Euler tours of $G_{2,n}$ can be approximated in polynomial time.

These design decisions were reached after rejecting similar techniques. Initially, the proof of Lemma 5.2 showed that for a vertex w_k in column k, there was an interleaving vertex in every single other column, so that after transforming the interleaving vertex, the available and prohibited transitions at k would be swapped. However this was a significantly longer proof, and would have required a further lemma to show that switching the available and prohibited transition systems of w_k would imply that now the configuration of column k and the complement block were compatible.

Another design feature that changed was the choice to only perform transformations on difference columns. A previous design used a vertex in column $k+1$ as a helper to a vertex in column k. The issue with this design is that it has the potential to create new difference columns. Thus after fixing a difference column, the number of difference columns may have stayed the same. It is still true with this technique that by the time the sweep had passed the final column, all columns would have been fixed, however the sweep would likely have had significantly more steps. While this would not have necessarily stopped the bound from being polynomial, it is clearly a far less computationally efficient approach.

Perhaps the biggest change in approach is the use of Lemma 5.2. In a previous version of this dissertation, a specific vertex ordering had been given for every possible change from one column configuration to another. The use of Lemma 5.2 has made the basic ideas behind the crux of the method clearer.

A design decision that played a more substantial role was the choice to rotate the grid before beginning any transformations to place any column belonging to class $[l_0, l_1]$ in X in column 0. In doing so, we removed the need to account for the possibility of a special helper vertex being used when trying to guess $\hat{Z}_i$. This meant that the number of guesses for $\hat{Z}_i$ was reduced by a factor of n, giving a much tighter polynomial bound on the possibilities for X and Y.

Chapter 6

Conclusion and Future Work

In this chapter, we revisit our initial goals for the project and discuss how they were met. We summarise the merits and limitations of the work presented, and suggestions for further work.

6.1 Initial Goals

Our primary initial goal for the project had been to provide evidence that the Markov chain due to Tetali and Vempala was rapidly mixing, by proving that the chain was rapidly mixing for certain special cases. We had initially hoped to provide a result on both $G_{2,n}$ and $G_{3,n}$, but the task of showing that the chain was rapidly mixing for $G_{2,n}$ was more difficult than initially expected, and due to time constraints no substantial progress was made on the $G_{3,n}$ case.

6.1.1 Achievements

Having written a project proposal previously [19], the basic background was already reasonably familiar. However, in order to construct a set of canonical paths that possessed the necessary properties for our proof, a deep understanding of the canonical paths method and the structure of the Markov chain was necessary. A lot of time was spent considering how properties of $G_{2,n}$ could be exploited, with eventual success.
The main achievement of the project was our final result that the chain is rapidly mixing for $G_{2,n}$. The proof shows that the chain mixes in time polynomial in N and ε^{-1}, using the canonical paths method. Thus we have met our initial goal of providing evidence that the chain may be rapidly mixing for $G_{m,n}$. A by-product of the proof

was the development of a proof technique and framework for viewing the grid that can hopefully be used to analyse other 4-regular toroidal grids.

Secondary achievements include providing a comprehensive detailing of the relevant background material, an explanation and illustration of the Markov chain Monte Carlo method, and a review of the relevant literature. Furthermore, details not included in Tetali and Vempala's paper have been filled in, such as the self reducibility of the problem of counting Euler tours, and the fact that the Markov chain has the uniform stationary distribution.

6.1.2 Limitations

Perhaps the most obvious limitation of the dissertation is that we did not have time to progress to showing that the chain was rapidly mixing for $G_{3,n}$. Given more time this would certainly be our next focus.

Considering our main objective was to provide evidence that the Markov chain may be rapidly mixing for $G_{m,n}$, a major limitation of our approach is that several aspects depend highly on the fact that the grid contains 2 rows, thus preventing the method from directly generalising. While this does not invalidate our result, it does mean that attempts to generalise would require significant amendments to the approach.

A limitation that prevents the work easily generalising is the reliance on Lemma 4.1. The lemma allows us to fix 2 difference vertices in 3 κ-transformations, and we exploit the fact that there are 2 vertices in a column. Therefore in future work we might experiment with Kotzig's theorem (Theorem 2) to see how the proof might be adapted to show that we can transform one Euler tour to another using only a polynomial number of κ-transformations on difference vertices (our polynomial requirement would allow us to bound the length of a canonical path).

Furthermore, our work exploits the fact that there are only 3 classes of legal partial transition systems for $G_{2,n}$, and explicitly uses their relationship in the proof of Lemma 5.2. Creed showed that the number of classes for $G_{m,n}$ is equal to $\binom{2m}{m}\frac{m!}{2^m} = \frac{m^m}{2^m}$, and so is exponential in m. As the number of classes grows and their relationships become more complicated, this proof technique becomes infeasible. Thus an alternative poof technique would be necessary for showing the availability of a helper vertex to substitute in column configurations.

Another limitation of the dissertation is that more work could have gone into tightening the bound on the mixing time. While the bound is certainly polynomial, the constant

factor could be reduced, for example by making more informed guesses about column configurations by considering more carefully the available and prohibited transitions. However, it is unlikely such adjustments could tighten the bound beyond a constant factor, and so would not have a significant affect on the result.

6.2 Future Work

As previously mentioned, a key area of future work would be to construct a bound on the mixing time of the chain for $G_{3,n}$. Using the framework constructed to prove this result for $G_{2,n}$, this would seem to be a reasonably straightforward task. However multiple aspects of the proof would need to be adjusted significantly. Take, for example, the proof of Lemma 5.3. The proof relies on the fact that a single move at one vertex in a column in the complement block can change the class of the entire complement block. However in $G_{2,n}$, a column configuration and a complement block are *only* incompatible if they belong to the same class. Hence the focus here was *changing* the class of the complement block, paying no heed to what it was changed *to*. In $G_{3,n}$, there are far more classes and far more combinations of incompatible classes. This means that the entire structure of this proof technique might need to be changed for $G_{3,n}$. We might hope that, in finding a way to prove the rapid mixing for $G_{3,n}$, new techniques might be developed that are more suitable for generalising to $G_{m,n}$.

This leads us to a much larger body of future work: the $G_{m,n}$ case. While our result has provided evidence that the Markov chain may be rapidly mixing for $G_{m,n}$, our work certainly does not introduce a clear insight into how this may be proven. The idea of compatible columns and complement blocks can generalise, although a lot of thought would need to go into establishing how much of the proof relied on the additional structure of $G_{2,n}$.

We must also note that when forming our bound on the number of possibilities for reconstructing X and Y, we used the fact that there were only 8 possible column configurations. As the number of classes of a column configuration grows exponentially in m, so too will the total number of column configurations. Thus in $G_{m,n}$, we will not be able to guess column configurations in the same way that we could for $G_{2,n}$.

Bibliography

[1] Jaromir Abrham and Anton Kotzig. Transformations of Euler tours. *Annals of Discrete Mathematics*, 8:65–69, 1980.

[2] Sanjeev Arora and Boaz Barak. *Computational Complexity; A Modern Approach*. Cambridge University Press, 2009.

[3] Maria Astefanoaei. *Euler Tours on the Grid*. 4th year project report, University of Edinburgh, 2014.

[4] C.W. Borchardt. Ueber eine der interpolation entsprechende darstellung der eliminations-resultante. *Journal für die reine und angewandte Mathematik*, 57:111–121, 1860.

[5] Graham R. Brightwell and Peter Winkler. Counting Eulerian circuits is #p-complete. In *Proceedings of the Second Workshop on Analytic Algorithmics and Combinatorics*, page 259–262, 2005.

[6] Prasad Chebolu, Mary Cryan, and Russell Martin. Exact counting of Euler tours for generalized series-parallel graphs. *Journal of Discrete Algorithms*, 10:110 – 122, 2012.

[7] Prasad Chebolu, Mary Cryan, and Russell Martin. Exact counting of Euler tours for graphs of bounded treewidth. *CoRR*, 2013.

[8] Stephen A. Cook. The complexity of theorem-proving procedures. In *Proceedings of the Third Annual ACM Symposium on Theory of Computing*, STOC '71, pages 151–158, New York, NY, 1971. ACM.

[9] Patrick John (Páidí) Creed. *Counting and Sampling Problems on Eulerian Graphs*. PhD thesis, University of Edinburgh, 2010.

[10] Mary Cryan. *Lecture notes in Randomness and Computation*. University of Edinburgh, 2018.

[11] Mary Cryan. *Personal communication*. 2018.

[12] Qi Ge and Daniel Stefankovic. The complexity of counting Eulerian tours in 4-regular graphs. *Algorithmica*, 63:588–601, 2010.

[13] Mordecai Golin, Yiu-Cho Leung, Yajun Wang, and Xuerong Yong. Counting structures in grid-graphs, cylinders and tori using transfer matrices: Survey and new results. *The Proceedings of the The Second Workshop on Analytic Algorithmics and Combinatorics (ANALCO05)*, 2005.

[14] Heng Guo. *Lecture notes in Computational Complexity*. University of Edinburgh, 2018.

[15] Mark Jerrum. In *Counting, sampling and integrating: algorithms and complexity*, Lectures in Mathematics – ETH Zürich, chapter 4. Birkhäuser, Basel, 2003.

[16] Mark Jerrum and Alistair Sinclair. Approximating the permanent. *SIAM Journal on Computing*, 18(6):1149 – 1178, 1989.

[17] Mark Jerrum and Alistair Sinclair. The Markov chain Monte Carlo method: An approach to approximating counting and integration. In *Approximation algorithms for NP-hard problems*. PWS Publishing Co, 1997.

[18] Mark Jerrum, Leslie Valiant, and Vijay Vazirani. Random generation of combinatorial structures from a uniform distribution. *Theoretical Computer Science*, 43:169–188, 1986.

[19] Sophia Jones. *Euler Tours on the Grid*. Informatics project proposal, University of Edinburgh, 2018.

[20] Richard Karp and Michael Luby. Monte-Carlo algorithms for the planar multiterminal network reliability problem. *Journal of Complexity*, 1(1):45–64, 1985.

[21] Anton Kotzig. Eulerian lines in finite 4-valent graphs and their transformations. In Paul Erdős and Gyula Katona, editors, *Theory of Graphs (Proceedings of the Colloquium held at Tihany, Hungary)*, pages 219–230. Academic Press, 1968.

[22] Teodor Vanislavov Marinov. *Eulerian orientations versus Euler tours*. 4th year project report, University of Edinburgh, 2016.

[23] Ben Morris and Alistair Sinclair. Random walks on truncated cubes and sampling 0-1 knapsack solutions. *SIAM Journal on Computing*, 34(1):195–226, 2002.

[24] Dana Randall. *Lecture notes in Markov Chain Monte Carlo Algorithms*. Georgia Institute of Technology, 2010.

[25] Alistair Sinclair. Improved bounds for mixing rates of Markov chains and multicommodity flow. *Combinatorics, Probability and Computing*, 1:351–370, 1992.

[26] Richard Stanley. *Enumerative Combinatorics*. The Wadsworth & Brooks/Cole Mathematics Series, Monterey, CA, 1986.

[27] Katarzyna Steliga and Dominik Szynal. On Markov-type inequalities. *International Journal of Pure and Applied Mathematics*, 58:137–152, 2010.

[28] Prasad Tetali and Santosh Vempala. Random sampling of Euler tours. *Algorithmica*, 30:376–385, 2001.

[29] W. T. Tutte and C. A. B. Smith. On unicursal paths in a network of degree 4. *The American Mathematical Monthly*, 48(4):233–237, 1941.

[30] Leslie Valiant. The complexity of enumeration and reliability problems. *Society for Industrial and Applied Mathematics*, 8(3):410–421, 1977.

[31] Leslie Valiant. The complexity of computing the permanent. *Theoretical Computer Science*, 8:189–201, 1979.

[32] T van Aardenne-Ehrenfest and N.G. de Bruijn. *Circuits and Trees in Oriented Linear Graphs*, volume 28, pages 149–163. 06 2010.