# Counting, Modular Counting and Graph Homomorphisms

Andreas-Nikolaos Gkompel-Magkakis
(Andreas Göbel)

St John's College

University of Oxford

A thesis submitted for the degree of

*Doctor of Philosophy*

Trinity 2016

*To my parents,*
*Despina and Anton.*

# Counting, Modular Counting and Graph Homomorphisms

Andreas-Nikolaos Gkompel-Magkakis
(Andreas Göbel)

St John's College
University of Oxford

A thesis submitted for the degree of
*Doctor of Philosophy*

Trinity 2016

## Abstract

A homomorphism from a graph $G$ to a graph $H$ is a function from $V(G)$ to $V(H)$ that preserves edges. Many combinatorial structures that arise in mathematics and in computer science can be represented naturally as graph homomorphisms and as weighted sums of graph homomorphisms. In this thesis we study the complexity of various problems related to graph homomorphisms.

We first study the problem $\#_k$HomsTo$H$ of counting the homomorphisms from an input graph to a fixed undirected graph $H$ modulo an integer $k$. A characteristic feature of counting modulo $k$ is that the objects to be counted can be grouped in sets of size $k$ and, thus, cancel out. These cancellations make wider classes of instances tractable than exact (non-modular) counting and, furthermore, the value of the modulus can affect the tractability of a problem. Modular counting provides a rich setting for studying the structure of homomorphism problems. In this case, the structure of the graph $H$ has a large influence on the complexity of the problem. We show the following results.

- When $p$ is a prime and $H$ is an asymmetric three, then $\#_p$HomsTo$H$ is either polynomial-time computable or $\#_p\,\mathsf{P}$-complete.

- When $H$ is a graph that has no cycles that share edges, then $\#_2$HomsTo$H$ is either polynomial-time computable or $\#_2\,\mathsf{P}$-complete.

- When $H$ is a graph that contains no 4-cycles, then $\#_2$HomsTo$H$ is either polynomial-time computable or $\#_2\,\mathsf{P}$-complete.

These types of results in computational complexity are known as dichotomy theorems. Dichotomy theorems give a complete characterisation of the complexity of a problem, depending on the values of its parameter, by showing that a problem is either tractable or hard, and that there are no values of the parameter for which the problem has intermediate complexity. Our results on $\#_2$HomsToH partially confirm a conjecture of Faben and Jerrum that was previously known to hold for trees.

We also study a counting problem related to matrix partitions of a graph, a generalisation of graph homomorphisms. Given a symmetric $D \times D$ matrix $M$ over $\{0, 1, *\}$, an $M$-partition of a graph $G$ is a partition of the vertices of $G$ into $D$ parts which are associated with the rows of $M$. The vertices of $G$ are mapped in such a way that no edge of $G$ is mapped to a 0 in $M$ and no non-edge of $G$ is mapped to a 1 in $M$. In a list $M$-partition of a graph $G$, for every vertex $v$ of $G$ we are also given a list of allowable parts. A list $M$-partition of a graph $G$ is an $M$-partition of $G$ that respects the given lists.

There has been quite a bit of work on determining for which matrices $M$ computations involving list $M$-partitions are tractable. We focus on the problem of counting list $M$-partitions, given a graph $G$ and given a list for each vertex of $G$. We identify a certain set of "tractable" matrices $M$. We give an algorithm that counts list $M$-partitions in polynomial time for every (fixed) matrix $M$ in this set. Furthermore, we give an explicit characterisation of the dichotomy theorem — counting list $M$-partitions is tractable (in FP) if the matrix $M$ has a structure called a derectangularising sequence. If $M$ has no derectangularising sequence, we show that counting list $M$-partitions is #P-hard. Finally, we show that the meta-problem of determining whether a given matrix has a derectangularising sequence is NP-complete.

Finally we study the Moran process on directed graphs. The Moran process, as studied by Lieberman, Hauert and Nowak, is a stochastic process modelling the spread of genetic mutations in populations. The process has an underlying graph in which vertices correspond to individuals. Initially, one individual (chosen uniformly at random) possesses a mutation, with fitness $r > 1$. All other individuals have fitness 1. At each step of the discrete-time process, an individual is chosen with probability proportional to its fitness, and its state (mutant or non-mutant) is passed on to an out-neighbour which is chosen uniformly at random. If the underlying graph is strongly connected then the process will eventually reach *fixation*, in which all individuals are mutants, or *extinction*, in which no individuals are mutants. An infinite family of directed graphs is said to be *strongly amplifying* if, for every $r > 1$, the extinction probability tends to 0 as the number of vertices increases. Strong amplification is a rather surprising property — it means that in such graphs, the fixation probability of a uniformly-placed initial mutant tends to 1 even though the initial mutant only has a fixed selective advantage of $r > 1$ (independent of $n$). Strong amplifiers have received quite a bit of attention, and Lieberman et al. proposed two potentially strongly-amplifying families —one of them called superstars. Heuristic arguments have been published, arguing that there are infinite families of superstars that are strongly amplifying. In this thesis we explore the amplification limits of superstars by proving a rigorous upper bound for the fixation probability of the Moran process on superstars.

# Acknowledgements

First of all I would like to express my gratitude to my supervisor Leslie Goldberg. It has been a great honour to work with an academic of her level. Leslie has supported me throughout the whole duration of my DPhil in many ways. She has taught me how to do research in mathematics, how to properly write papers and how to give engaging talks. She has always been available and approachable and has devoted a lot of time to me. She has offered me valuable advice, always on the point, for matters inside and outside academia. Her true professionalism and the integrity of her character amaze and inspire me. Leslie has been much more than a supervisor, she has also been a friend.

Coauthor in every paper I published during my DPhil, and neighbour in every office I have been was David Richerby. He has not only been my collaborator, but he has also monitored my progress throughout my whole DPhil, helping me in a variety of ways. He never refused to discuss mathematical problems with me, was always willing to read my drafts and has been present in all the conference talks I have given. His humour and jokes cheered me up every time I entered my office.

I would like to thank my examiners, Jan van den Heuvel and Standa Živný, for spending the time to read my thesis thoroughly. Their detailed comments have contributed towards improving the quality of this thesis.

I also want to express my gratitude to Stathis Zachos and Aris Pagourtzis who encouraged me to apply to work with Leslie in the first place. They introduced me to the field, while I was still an undergraduate/masters student. They have been monitoring my progress throughout these years and were always eager to listen to my talks every time I visited Greece.

I want to thank my coauthors and office mates Andreas Galanis and John Lapinskas. Working with them and sharing an office has been a real pleasure for me. I also want to thank my coauthors Coling McQuilan and Tomoyuki Yamakami.

During my studies in Oxford, I have been influenced and have collaborated with a lot of people from the department of computer science. I would like to offer my gratitude to Elias, Peter, Stefan, Yiannis, Heng, Nikola, Philip, Aris, Peter and Mathias. I would also like to thank my former officemates in Liverpool, Tony and Alkmini, who have contributed towards a creative environment during the first months of my DPhil.

While living in Oxford I interacted with a lot of people and made a lot of friends. They have given me advice, listened to my struggles and offered their help to me. I am grateful to all of them. Among them I would like to mention Stefano, Lisa, Christos, Rita, Tim, Anna, Miles, Karine and Tom.

I would also like to thank Hannah, who has given me the motivation to push through the final stages of my DPhil. I feel so grateful to have her in my life.

Finally I would like to thank my Parents, Despina and Anton, for their never ending support.

# Contents

# Chapter 1

# Introduction

## 1.1 Graph homomorphisms

A homomorphism from a graph $G$ to a graph $H$ is a function from $V(G)$ to $V(H)$ that preserves edges, in the sense of mapping every edge of $G$ to an edge of $H$; non-edges of $G$ may be mapped to either edges or non-edges of $H$. Many structures arising in graph theory can be represented naturally as homomorphisms. For example, the proper $q$-colourings of a graph $G$ correspond to the homomorphisms from $G$ to a $q$-clique. For this reason, homomorphisms from $G$ to a graph $H$ are often called "$H$-colourings" of $G$. Independent sets of $G$ correspond to the homomorphisms from $G$ to the connected graph with two vertices and one self-loop (vertices of $G$ which are mapped to the vertex with a self-loop are out of the corresponding independent set; vertices which are mapped to the other vertex are in it). Homomorphism problems can also be seen as constraint satisfaction problems (CSPs) in which the constraint language consists of a single symmetric binary relation. Partition functions in statistical physics such as the Ising model, the Potts model, and the hard-core model arise naturally as weighted sums of homomorphisms [27, 45].

### 1.1.1 Dichotomy theorems for graph homomorphism problems

Perhaps the most basic problem related to graph homomorphisms is the decision problem $\textsc{HomsTo}H$: does a graph $G$ have a homomorphism to a target graph $H$? When $H$ is a 3-clique, deciding whether there exists a homomorphism from $G$ to $H$ is equivalent to deciding whether $G$ has a proper 3-colouring, which is known to be $\mathsf{NP}$-complete. Thus, in general, the decision problem is $\mathsf{NP}$-complete. When $H$ is a 2-clique the problem of deciding whether a graph $G$ has a homomorphism to $H$ is equivalent to deciding whether $G$ is bipartite. This can be decided in polynomial time. Thus, to understand when $\textsc{HomsTo}H$ is hard and when it is tractable, we will treat the target graph $H$ as a parameter of the problem and not as part of the input.

Hell and Nešetřil were able to obtain the following result [52, Theorem 1].

**Theorem 1.1** (Hell and Nešetřil)**.** *Let $H$ be a graph. If $H$ has a self-loop or is bipartite, then* HomsTo$H$ *is in* $\mathsf{P}$*, otherwise* HomsTo$H$ *is* $\mathsf{NP}$*-complete.*

Theorem 1.1 gives a complete characterisation of the complexity of HomsTo$H$ for all the values of the parameter target graph $H$. More importantly, it shows that HomsTo$H$ is either in $\mathsf{P}$ or $\mathsf{NP}$-complete and that there are no target graphs $H$ for which the problem has intermediate complexity. Such a result in computational complexity is known as a dichotomy theorem. Dichotomy theorems give a clean way of distinguishing the borders of tractability and intractability for large classes of problems. Such a dichotomy cannot exist for the whole $\mathsf{NP}$. Ladner [57, Corollary 1.1] has shown that if $\mathsf{P} \neq \mathsf{NP}$, then there exist problems in $\mathsf{NP}$ that are neither in $\mathsf{P}$ nor $\mathsf{NP}$-complete.

**Theorem 1.2** (Lander)**.** *If* $\mathsf{P} \neq \mathsf{NP}$ *then there exist problems in* $\mathsf{NP} \setminus \mathsf{P}$ *that are not* $\mathsf{NP}$*-complete under polynomial-time Turing reductions.*

## Counting problems

There are other interesting questions related to graph homomorphisms. We next discuss the problem of counting graph homomorphisms #HomsTo$H$: given an input graph $G$ and a fixed target graph $H$, how many homomorphisms are there from $G$ to $H$? The complexity of counting problems has been an object of research for more than thirty years, due to their number of applications in a variety of fields. For example, the problem of our interest, #HomsTo$H$, finds applications in statistical physics and quantum mechanics, as partition functions can be expressed as weighted sums of homomorphisms (see [8, 27]).

The class characterising the complexity of most natural counting problems is $\#\mathsf{P}$ —the analogue of $\mathsf{NP}$ for counting problems. It is the class of function problems of the form “compute $f(x)$”, where $f(x)$ is the number of accepting paths of an non-deterministic polynomial-time Turing machine on input $x$. $\#\mathsf{P}$-completeness via Turing reductions provides strong evidence of intractability. If a $\#\mathsf{P}$-complete problem has a polynomial-time algorithm, then $\#\mathsf{P} = \mathsf{FP}$, where $\mathsf{FP}$ indicates the class of functions computable in polynomial time. $\#\mathsf{P} = \mathsf{FP}$ is considered to be highly unlikely, as this would also imply $\mathsf{P} = \mathsf{NP}$. In fact, Toda has shown [70] that for every problem in the polynomial hierarchy there is a polynomial-time Turing reduction to a problem in $\#\mathsf{P}$.

It is expected for a decision problem that is $\mathsf{NP}$-complete to also have a $\#\mathsf{P}$-hard counting version. It is open until today whether this is always true, but for all natural problems that have been studied so far, that is the case. Surprisingly though, there are problems with decision version in $\mathsf{P}$ that have $\#\mathsf{P}$-hard counting version. Valiant, in his seminal paper [71], shows that counting perfect matchings in a bipartite graph is $\#\mathsf{P}$-complete. Finding a perfect matching in a bipartite graph can be done in polynomial time by the augmenting path method.

The proof of Theorem 1.2 (see [57]) can be extended to the counting world, and show that if $\mathsf{FP} \neq \#\mathsf{P}$ then there exist problems in $\#\mathsf{P} \setminus \mathsf{FP}$ that are not $\#\mathsf{P}$-complete (under polynomial-time Turing reductions). Hence there is great incentive for obtaining dichotomy theorems for counting problems. In fact, as we will see later, dichotomy theorems have been more successful for counting problems than for decision problems.

For the problem of counting graph homomorphisms #HomsTo$H$, Dyer and Greenhill [27] have shown a dichotomy theorem, completely characterising the complexity of #HomsTo$H$.

**Theorem 1.3** (Dyer and Greenhill)**.** *Let $H$ be a graph. If every component of $H$ is a complete bipartite graph with no self-loops or a complete graph with all self-loops present, then #HomsTo$H$ can be solved in polynomial time. Otherwise #HomsTo$H$ is $\#\mathsf{P}$-complete.*

The polynomial-time algorithms for the tractable cases of #HomsTo$H$ are trivial. When $H$ is a complete graph with all self-loops present the number of homomorphisms from $G$ to $H$ is $|V(H)|^{|V(G)|}$, as every vertex of $G$ can be mapped to any vertex of $H$ by a homomorphism. Similarly, when $H$ is a complete bipartite graph with vertex bipartition $U_1, U_2$ the number of homomorphisms from a connected graph $G$ can be computed as follows. If $G$ is not a bipartite graph, then there are zero homomorphisms from $G$ to $H$. Otherwise $G$ has a bipartition $V_1, V_2$ and the number of homomorphisms from $G$ to $H$ is $|U_1|^{|V_1|}|U_2|^{|V_2|} + |U_1|^{|V_2|}|U_2|^{|V_1|}$. The latter holds because every homomorphism $\sigma$ can map the vertices in $V_j$, $j \in \{1, 2\}$, to either vertices in $U_1$ or to vertices in $U_2$, but not to vertices in both $U_1$ and $U_2$. All edges between $U_1$ and $U_2$ exist in $H$, so there are no further restrictions on $\sigma$. If $G$ is not connected we can compute the number of homomorphisms from $G$ to $H$ by taking the product of the number of homomorphisms from each component of $G$ to $H$.

As we already mentioned, partition functions in statistical physics can be represented as weighted graph homomorphisms. In weighted #HomsTo$H$, the parameter is a weighted graph $H$, that has edge and vertex weights. Let $W$ be the adjacency matrix of $H$, where $W_{u,v}$ is the weight of the edge $(u, v) \in E(H)$, and let $\{\lambda_v\}_{v \in V(H)}$ be the vertex weights. Let $G$ be an input graph. Weighted #HomsTo$H$ is the problem of computing the following sum

$$\sum_{\sigma: V(G) \to V(H)} \left( \prod_{(u,v) \in E(G)} W_{\sigma(u),\sigma(v)} \prod_{v \in V(G)} \lambda_{\sigma(v)} \right).$$

Bulatov and Grohe [13] have given the first dichotomy for weighted #HomsTo$H$ and shown that when $H$ has non-negative real weights, #HomsTo$H$ is either in $\mathsf{FP}$ or

$\#\mathsf{P}$-complete. Goldberg et al. [45] have shown a dichotomy theorem for $\#\textsc{HomsTo}H$ when the weights of the target graph $H$ are real numbers. Finally Cai, Chen and Lu [16] have shown that when $H$ has complex weights, $\#\textsc{HomsTo}H$ is either in $\mathsf{FP}$ or $\#\mathsf{P}$-complete.

### 1.1.2 Constraint satisfaction problems

Constraint satisfaction problems find applications in a wide range of fields such as theory of databases [40] and artificial inteligence [67]. In the decision version of the constraint satisfaction problem we are given a set of constraints over variables and the objective is to decide whether there exists an assignment of values to the variables that satisfies the set of constraints.

**Problem 1.4.** *Name.* CSP($\Gamma$).

*Parameter.* A domain set $D$ and a set of relations $\Gamma = \{R_1 \dots R_m\}$, where for each $j \in [m]$, $R_j : D^{r_j} \to \{0,1\}$ and $r_j \in \mathbb{N}_{>0}$.

*Input.* A finite set of constraints over variables $x_1 \dots x_n$ of the form $R_j(x_{i_{j,1}}, x_{i_{j,2}}, \dots, x_{i_{j,r_j}})$.

*Output.* "Yes" if there exists an assignment of $x_1 \dots x_n$ that satisfies all the constraints, "No" otherwise.

Constraint satisfaction problems generalise graph homomorphism problems. To see that CSP is a generalisation of $\textsc{HomsTo}H$, let $G$ be an input for $\textsc{HomsTo}H$. We describe an equivalent CSP instance. The domain of the constraint satisfaction problem is $D = V(H)$ and $\Gamma$ contains a single binary relation $R_H$, with $R_H(u,v) = 1$ when $(u,v) \in E(H)$ and $R_H(u,v) = 0$ otherwise. Thus $\textsc{HomsTo}H$ is an instance of CSP($\{R_H\}$). The input of the CSP($\{R_H\}$) contains a variable $x_v$ for every vertex $v \in V(G)$ and a constraint $R_H(x_u, x_v)$ for every edge $(u,v) \in E(G)$. As we can see from the construction, every valid homomorphism $\sigma : V(G) \to V(H)$ corresponds to an assignment of the variables $\{x_v\}_{v \in V(G)}$ that satisfies every constraint in the CSP.

The first dichotomy result for the constraint satisfaction problem was given by Schaefer [68], where they showed that, when the domain of the functions in $\Gamma$ is restricted to the Boolean domain (i.e. $\{0,1\}$), CSP($\Gamma$) is either in $\mathsf{P}$ or $\mathsf{NP}$-complete. For the complexity of CSP($\Gamma$) when the functions of $\Gamma$ have arbitrary domain, there is a famous conjecture of Feder and Vardi [38], that CSP($\Gamma$) is either in $\mathsf{P}$ or $\mathsf{NP}$-complete. Although the complexity of CSP has been intensively studied, the conjecture of Feder and Vardi remains open.

There has been more success in finding dichotomies for the counting version of the constraint satisfaction problem. In the counting constraint satisfaction problems, the constraints take the form of functions from the domain set $D$ to range over any set $A$.

**Problem 1.5.** *Name.* #CSP($\mathcal{F}$).

*Parameter.* A domain set $D$, a range set $A$ and a set of functions $\mathcal{F} = \{f_1 \dots f_m\}$, where for each $j \in [m]$ $f_j : D^{r_j} \to A$ and $r_j \in \mathbb{N}_{>0}$.

*Input.* A finite set of constraints over variables $x_1 \dots x_n$ of the form $f_j(x_{i_{j,1}}, x_{i_{j,2}}, \dots, x_{i_{j,r_j}})$.

*Output.* $\sum_{x_1,\dots,x_n \in D} \prod_j f_j(x_{i_{j,1}}, x_{i_{j,2}}, \dots, x_{i_{j,r_j}})$.

Bulatov [10] has shown that when the functions in $\mathcal{F}$ have arbitrary domain size and range $A = \{0, 1\}$, #CSP($\mathcal{F}$) is either in $\mathsf{FP}$ or $\#\mathsf{P}$-complete (see also [25]). This was the first dichotomy for a constraint satisfaction problem of arbitrary domain size. Since then, there has been a series of results [12, 15, 14] giving dichotomy theorems for #CSP($\mathcal{F}$) of arbitrary domain size, where in each of the results the range of the functions in $\mathcal{F}$ becomes more general. In the final result of this series, Cai and Chen [14] give a dichotomy for #CSP($\mathcal{F}$) with arbitrary domain size and complex valued functions, resolving the complexity of the counting constraint satisfaction problem.

## 1.2 Modular counting and graph homomorphisms

In the first part of this thesis we will focus on the problem of counting graph homomorphisms from a graph $G$ to a target graph $H$ modulo an integer. Let $\mathrm{Hom}(G \to H)$ denote the set of homomorphisms from $G$ to $H$.

**Problem 1.6.** *Name.* $\#_k$HomsTo$H$.

*Parameter.* A graph $H$ and an integer $k$.

*Input.* A graph $G$.

*Output.* $|\mathrm{Hom}(G \to H)| \pmod{k}$.

The complexity of modular counting is an interesting topic with some surprising results. Modular counting was originally studied from the point of view of decision problems, where the objective is to determine if the number of solutions is non-zero modulo $k$. The complexity class $\oplus\mathsf{P}$ was first studied by Papadimitriou and Zachos [65] and by Goldschlager and Parberry [47]. $\oplus\mathsf{P}$ consists of the all problems of the form "is $f(x)$ odd or even?", where computing $f(x)$ is a function in $\#\mathsf{P}$. Toda [70] has shown that there is a randomised polynomial-time reduction from every problem in the polynomial hierarchy to some problem in $\oplus\mathsf{P}$. As such, $\oplus\mathsf{P}$ is a large complexity class and $\oplus\mathsf{P}$-completeness seems to represent a high degree of intractability.

Early work in modular counting [17, 54, 3] includes results on the classes $\mathrm{Mod}_k\,\mathsf{P}$, consisting of all the problems of the form "is $f(x) \not\equiv 0 \pmod{k}$?", where computing

$f(x)$ is a function in $\#\mathsf{P}$. In our study of $\#_k$HomsToH it is more natural to consider modular counting from the perspective of computing functions. For an integer $k$ the complexity class $\#_k\mathsf{P}$ consists of all the problems of the form "compute $f(x)$ modulo $k$", where $f(x)$ is a function in $\#\mathsf{P}$. In the special case of $k = 2$, $\#_2\mathsf{P} = \oplus\mathsf{P}$, as the problems of $\#_2\mathsf{P}$ require a one bit answer. Throughout this thesis though, instead of the more traditional notation $\oplus\mathsf{P}$, we will use $\#_2\mathsf{P}$ to emphasise our interest in computing functions.

Faben [31] compares the complexity of the function classes $\#_k\mathsf{P}$ and the decision classes $\mathrm{Mod}_k\mathsf{P}$. It is trivial to see that any problem in $\mathrm{Mod}_k\mathsf{P}$ reduces to a problem in $\#_k\mathsf{P}$ via Turing reductions. The inverse also holds ([31, Theorem 3.1.20]) as every problem in $\#_k\mathsf{P}$ reduces to $\mathrm{Mod}_k$Sat, the problem of deciding whether the number of satisfying assignments to a CNF Boolean formula is not divisible by $k$. $\mathrm{Mod}_k$Sat is complete for $\mathrm{Mod}_k\mathsf{P}$, as the reduction in Cook's theorem showing the $\mathsf{NP}$-completeness of Sat preserves the number of solutions. Finally, Faben [31, Theorem 3.1.24] also shows the existence of a $\#_3\mathsf{P}$-complete problem for which the value is always non-zero, making its modular decision version trivial.

Although counting modulo $k$ resembles ordinary, non-modular counting, it is still very different. Clearly, if a counting problem can be solved in polynomial time, the corresponding decision and modular counting problems can also be solved in polynomial time. The converse, though, does not necessarily hold. For example consider the problem of counting perfect matchings in a bipartite graph, which we mentioned earlier. This counting problem is $\#\mathsf{P}$-complete, its decision version is in $\mathsf{P}$ and so is its modulo 2 counting version. To see that counting (modulo 2) perfect matchings in a bipartite graph is equivalent to computing the permanent of a matrix, consider the following. Let $G = (V_1, V_2, E)$ be a bipartite graph and let $A$ be the adjacency matrix of $G$ defined as follows. The rows of $A$ correspond to the vertices of $V_1$ and the columns of $A$ correspond to the vertices of $V_2$. For $v_1 \in V_1$ and $v_2 \in V_2$, set $A_{v_1,v_2} = 1$ if there is an edge between $v_1$ and $v_2$, otherwise $A_{v_1,v_2} = 0$. Counting the perfect matchings of $G$ is equivalent to computing the permanent of $A$ (see [71]). As $-1 \equiv 1 \bmod 2$, the permanent of $A$ is equivalent (modulo 2) to the determinant of $A$. The determinant of a matrix can be computed in polynomial time using Gaussian elimination.

The characteristic feature that makes the modular version of a hard counting problem tractable is cancellations. An example that illustrates how cancellations work in modular counting is NotAllEqualSAT: the problem of assigning values to Boolean variables such that each of a given set of clauses contains both true and false literals. The number of solutions is always even, since solutions can be paired up by negating every variable in one solution to obtain a second solution. This makes counting modulo 2 trivial, while determining the exact number of solutions is $\#\mathsf{P}$-complete [46] and even deciding whether a solution exists is $\mathsf{NP}$-complete [68].

Examples of problems with hard counting and decision version, but easy modular counting version are not only limited to counting modulo 2. Consider the problem of counting proper 3-colourings of a graph $G$ modulo 3, or even modulo 6. The number of 3-colourings of a graph that use all of the three colours is always a multiple of 6, since there are $3! = 6$ permutations of the colours. To count the number of 3 colourings of $G$ that use exactly 2 colours we have the following two cases. If $G$ is not bipartite, then there are no such colourings. Otherwise, $G$ is bipartite and the number of 3-colourings of $G$ that use exactly 2 colours is $3(2^c)$, where $c$ is the number of components of $G$. The factor of 3 comes from the fact that there are 3 ways to choose 2-colours. Now assume that we chose the colours in $\{1, 2\}$ to colour $G$. Let $G_1 \dots G_c$ be the components of $G$. Every component $G_i$ of $G$ is also bipartite. For $i \in [c]$, let $V_{i,L}$, $V_{i,R}$ be the bipartition of $G_i$ —since $G_i$ is connected there is only one bipartition of its vertices. If colour 1 is assigned to a vertex $v \in V_{i,L}$, then colour 1 must be assigned to every vertex in $V_{i,L}$, while colour 2 must be assigned to every vertex in $V_{i,R}$. Otherwise colour 2 is assigned to every vertex in $V_{i,L}$ (including $v$) and colour 1 is assigned to every vertex in $V_{i,R}$. Hence every component of $G$ can be coloured in two ways, and this gives the factor of $2^c$ in the number of the proper 3-colourings of $G$ that use exactly two colours. It is also trivial to count the proper 3-colourings of $G$ that use exactly one colour: If $G$ has an edge, then there are no such colourings, otherwise $G$ has no edges and there are 3 such colourings of $G$, one for each of the three colours. Thus, it is easy to count all proper 3-colourings modulo 3 or even modulo 6.

Perhaps the most surprising feature of modular counting problems is that the value of the modulus can affect the tractability of the problem. This can be seen in Valiant's famous restricted version of 3-SAT for which counting solutions is $\#\,\mathsf{P}$-complete [73], counting solutions modulo 7 is in $\mathsf{FP}$ but counting solutions modulo 2 is $\#_2\,\mathsf{P}$-complete [72]. The seemingly mysterious number 7 was subsequently explained by Cai and Lu [18], who showed that the $k$-SAT version of Valiant's problem is tractable modulo any prime factor of $2^k - 1$.

When studying the complexity of $\#_k$HomsTo$H$ we will use cancellations extensively. For example, if we wish to compute the size of a set $S$ modulo $k$, then, for any $k$-cardinality subset $X \subseteq S$, we have $|S| \equiv |S \setminus X| \pmod{k}$. This means that we can ignore the elements of $X$. When computing modulo 2, it is also helpful to partition the set $S$ into disjoint subsets $S_1, \dots, S_\ell$ exploiting the fact that $|S|$ is congruent modulo 2 to the number of odd-cardinality $S_i$.

### 1.2.1 Counting graph homomorphisms modulo a prime

Determining the complexity of $\#_k$HomsTo$H$ is an interesting topic. Even for the simplest case of $\#_k$HomsTo$H$ where $k = 2$, the complexity of $\#_2$HomsTo$H$ is strongly influenced by the structure of the target graph $H$. For example, consider the graphs $H_1$ and $H_2$ in Figure 1.1. Our results show that $\#_2$HomsTo$H_1$ is $\#_2\,\mathsf{P}$-complete, whereas

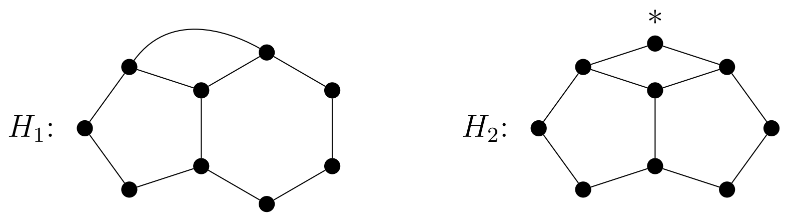

Figure 1.1: $\#_2$HomsTo$H_1$ is $\#_2$ P-complete, whereas $\#_2$HomsTo$H_2$ is in P. This, and the role of the starred vertex are explained later in the introduction.

$\#_2$HomsTo$H_2$ is in P.

The first to study the complexity of $\#_k$HomsTo$H$ were Faben and Jerrum [32]. To describe their work we first need to state a few definitions. An automorphism of a graph $G$ is a bijective function $\rho : V(G) \to V(G)$, such that $(u, v) \in E(G)$ if and only if $(\rho(u), \rho(v)) \in E(G)$. Let $k$ be a positive integer. For a function $\rho$ let $\rho^{(k)} = \rho \circ \rho \circ \cdots \circ \rho$, where the composition is applied $k$ times. An automorphism $\rho$ of order $k$ is an automorphism of a graph $H$, such that for $\ell < k$, $\rho^{(\ell)}$ is not the identity, but $\rho^{(k)}$ is the identity. An automorphism of order 2 is called an *involution*. Given a graph $H$ and an automorphism $\rho$ of $H$, $H^\rho$ denotes the subgraph of $H$ induced by the fixpoints of $\rho$. If $\rho$ has no fixpoints, then $H^\rho$ is the empty graph, which we also consider it to be a graph. We write $H \Rightarrow_k H'$ if there is an automorphism $\rho$ of order $k$ of $H$ such that $H^\rho = H'$ and we write $H \Rightarrow_k^* H'$ if either $H$ is isomorphic to $H'$ (written $H \cong H'$) or, for some positive integer $t$, there are graphs $H_1, \dots, H_t$ such that $H \cong H_1$, $H_1 \Rightarrow_k \cdots \Rightarrow_k H_t$, and $H_t \cong H'$.

Let $H$ be a graph and let $\rho$ be an involution of $H$. Faben and Jerrum showed [32, Lemma 3.3] that for any graph $G$, $|\text{Hom}(G \to H)| \equiv |\text{Hom}(G \to H^\rho)| \pmod 2$. They also showed [32, Theorem 3.7] that for every graph $H$ there is (up to isomorphism) exactly one involution-free graph $H^*$ such that $H \Rightarrow_2^* H^*$. We call $H^*$ the *involution-free reduction* of $H$. These results show that a polynomial-time algorithm for $\#_2$HomsTo$H$ exists for some graphs $H$. If the involution-free reduction $H^*$ of $H$ is the empty graph or the graph with one vertex, then $\#_2$HomsTo$H$ is (trivially) computable in polynomial time. More importantly Faben and Jerrum made the following conjecture.

**Conjecture 1.7** (Faben and Jerrum)**.** Let $H$ be a graph. If its involution-free reduction $H^*$ has at most one vertex, then $\#_2$HomsTo$H$ is in P; otherwise, $\#_2$HomsTo$H$ is $\#_2$ P-complete.

Note that our claim in Figure 1.1 is consistent with Conjecture 1.7. $H_1$ is involution-free, so its involution-free reduction is itself. On the other hand $H_2$ has an involution that exchanges the distinct vertices in the two cycles in the graph, hence $H_2 \Rightarrow_2 H_2'$, where $H_2'$ is the graph containing an edge and an isolated vertex (the isolated vertex corresponds to the vertex marked with $*$ in the figure). $H_2'$ has an involution exchanging the endpoints of its edge, so $H_2' \Rightarrow_2 H_2^*$, where $H_2^*$ is the graph containing the single vertex marked with $*$ in the figure.

Faben and Jerrum [32, Theorem 3.8] proved Conjecture 1.7 for the case in which $H$ is a tree.

**Theorem 1.8** (Faben and Jerrum)**.** *Let $H$ be a tree. If the involution-free reduction $H^*$ of $H$ has more than one vertex, then $\#_2$*HomsTo*$H$ is $\#_2$* P*-complete, otherwise $\#_2$*HomsTo*$H$ is solvable in polynomial time.*

### 1.2.2 Our results

We say that a graph $H$ is asymmetric if its automorphism group contains only the identity. In Part I of this thesis we partially prove Conjecture 1.7 of Faben and Jerrum by extending their results for $\#_2$HomsTo$H$ on trees, to further include graphs that have no cycles sharing edges and all graphs that have no 4-cycles. We also show a dichotomy for $\#_p$HomsTo$H$ when $p$ is a prime and $H$ is an asymmetric tree.

We have already discussed the results of Faben and Jerrum [32] showing that for some graphs $H$, $\#_2$HomsTo$H$ can be computed in polynomial time: if the involution-free reduction $H^*$ of $H$ has at most one vertex, then $\#_2$HomsTo$H$ is easy to compute. These results can be extended to $\#_p$HomsTo$H$ when $p$ is any prime number and not just 2. In order to identify target graphs $H$ for which $\#_p$HomsTo$H$ is polynomially computable, we reduce the original graph in a series of reductions. In each reduction we find an automorphism $\tau$ of order $p$ and delete every vertex that is not a fixpoint of $\tau$. As Faben and Jerrum show, this procedure will converge to a unique graph $H^{*p}$ with no automorphisms of order $p$. We call this graph $H^{*p}$ the *order $p$ reduced form* of $H$. Since this technique gives all known polynomial-time cases of $\#_p$HomsTo$H$, the majority of our technical results in Part I are hardness results. In more detail, we obtain the following.

#### Pinning

In Chapter 2 of this thesis we establish pinning for $\#_p$HomsTo$H$ when the target graph $H$ and the prime $p$ are, what we later define, "orbit compatible". A partial function from a set $X$ to a set $Y$ is a function $f : X' \to Y$ for some $X' \subseteq X$. For any graph $H$, a *partially $H$-labelled graph* $J = (G, \tau)$ consists of an *underlying graph $G$* and a *pinning function $\tau$*, which is a partial function from $V(G)$ to $V(H)$. A homomorphism from a partially labelled graph $J = (G, \tau)$ to $H$ is a homomorphism $\sigma\colon G \to H$ such that, for all vertices $v \in \text{dom}(\tau)$, $\sigma(v) = \tau(v)$. The problem that we study then is $\#_p$PartLabHomsTo$H$, the problem of computing $|\text{Hom}(J \to H)| \pmod p$, given a graph $H$ as the parameter of the problem and a partially $H$-labelled graph $J$ as input. Our main result is the following.

**Theorem 1.9.** *Let $p$ be a prime and let $H$ be a graph. If $H$ and $p$ are orbit compatible, then $\#_p$*PartLabHomsTo*$H$ reduces to $\#_p$*HomsTo*$H$ via polynomial-time Turing reduction.*

This result will be used in later chapters to establish hardness for $\#_p$HomsTo$H$, as showing that $\#_p$PartLabHomsTo$H$ is hard implies that $\#_p$HomsTo$H$ is hard.

## Counting homomorphisms (modulo a prime) to asymmetric trees

In Chapter 3 we extend Theorem 1.8 for all primes $p$. Recall that the order $p$ reduced form of $H$ is $H^{*p}$. Our main theorem is the following.

**Theorem 1.10.** *Let $p$ be a prime and let $H$ be a graph where $H^{*p}$ is an asymmetric tree. If $H^{*p}$ has more than one vertex, then $\#_p$*HomsTo*$H$ is $\#_p\mathsf{P}$-hard, otherwise $\#_p$*HomsTo*$H$ is computable in polynomial time.*

In the theorem above we require $H^{*p}$ to be asymmetric, while in the theorem of Faben and Jerrum (Theorem 1.8) $H^*$ is only required to be a tree. If $p > 2$, $H^{*p}$ could potentially have involutions. Faben and Jerrum show [32, Lemma 5.4] that every involution-free tree is asymmetric. So if $H$ is a tree, then $H^*$ is an asymmetric tree. Therefore, our theorem is an extension of Theorem 1.8. We require $H^{*p}$ to be asymmetric so that $H^{*p}$ and $p$ are orbit compatible. We need $H^{*p}$ and $p$ to be orbit compatible in order to use our "pinning" theorem (Theorem 1.9) and establish hardness.

Additionally, in Section 3.6 we discuss the issues arrising when characterising the complexity of $\#_k$HomsTo$H$ when $k$ is a composite. More specifically we identify a graph $P_4$ for which $\#_2$HomsTo$P_4$ is in $\mathsf{P}$, while $\#_4$HomsTo$P_4$ is $\#_2\mathsf{P}$-hard. This shows that, unlike other results in modular counting, for an integer $r$ and a prime $p$ the complexity of $\#_{p^r}$HomsTo$H$, in general, is not the same as the complexity of $\#_p$HomsTo$H$.

## Counting homomorphisms (modulo 2) to cactus graphs

Chapter 4 is based on the paper "The Complexity of Counting Homomorphisms to Cactus Graphs Modulo 2" co-authored with Leslie Goldberg and David Richerby [43]. In Chapter 4 we extend Theorem 1.8 to all cactus graphs. A cactus graph is a connected graph in which every edge belongs to at most one cycle. Cactus graphs were first defined by Harary and Uhlenbeck [50] who attributed them to the physicist Husimi and therefore called them *Husimi Trees.* Cactus graphs arise, for example, in the modelling of wireless sensor networks [5] and in the comparison of genomes [66]. Some NP-hard graph problems can be solved in polynomial time on cactus graphs [4].

The main result of Chapter 4 is the following.

**Theorem 1.11.** *Let $H$ be a graph whose involution-free reduction $H^*$ is a cactus graph. If $H^*$ has at most one vertex, then $\#_2$*HomsTo*$H$ is solvable in polynomial time; otherwise, $\#_2$*HomsTo*$H$ is $\#_2\,\mathsf{P}$-complete.*

## Counting homomorphisms (modulo 2) to square-free graphs

Chapter 5 is based on the paper "Counting Homomorphisms to Square-Free Graphs Modulo 2" co-authored with Leslie Goldberg and David Richerby [44]. In Chapter 5 we show that Theorem 1.8 can be extended to a much richer class of graphs. In particular we prove that Conjecture 1.7 holds for every graph $H$ whose involution-free reduction has no 4-cycle (whether induced or not). Graphs without 4-cycles are called "square-free" graphs. These graphs arise frequently in combinatorics, for example in connection with the strong perfect graph theorem [21] and certain graph algorithms [1].

The main theorem of Chapter 5 is the following.

**Theorem 1.12.** *Let $H$ be a graph whose involution-free reduction $H^*$ is square-free. If $H^*$ has at most one vertex, then $\#_2$*HomsTo*$H$ is in $\mathsf{P}$; otherwise, $\#_2$*HomsTo*$H$ is $\#_2\,\mathsf{P}$-complete.*

Since there can be cactus graphs that contain 4-cycles, the results of this chapter do not subsume the results of Chapter 4. To fully prove Conjecture 1.7 it remains to show that the conjecture is true for all graphs $H$ that contain at least one 4-cycle and at least two cycles that share at least one edge.

## Computing the partition function (modulo a prime) of a two-spin system with an external field

In Chapter 6 we study the complexity of the problem of computing the partition function of a two spin system on multigraphs modulo a prime ($\#_p Z_{\gamma,\lambda}$). $\#_p Z_{\gamma,\lambda}$ has three parameters: a prime $p$ and $\gamma, \lambda \in \{0, 1, \dots p-1\}$. The input for $\#_p Z_{\gamma,\lambda}$ is a multigraph $G$. A configuration $\sigma : V(G) \to \{0,1\}$ is an assignment of the two spins "0" and "1" to the vertices of $G$. Let $c(\sigma)$ denote the number of edges $(u,v)$ of the input multigraph $G$ with $\sigma(u) = \sigma(v) = 1$ and let $\ell(\sigma)$ denote the number of vertices $u \in V(G)$ with $\sigma(u) = 0$. The partition function of the model is given by:

$$Z_{\gamma,\lambda}(G) = \sum_{\sigma:V(G)\to\{0,1\}} \gamma^{c(\sigma)} \lambda^{\ell(\sigma)}.$$

So the objective of the $\#_p Z_{\gamma,\lambda}$ is to compute $Z_{\gamma,\lambda}(G) \pmod p$. From the definition of $Z_{\gamma,\lambda}(G)$, we can see that the problem is equivalent to counting weighted homomorphisms modulo $p$ from the multigraph $G$ to the following weighted graph $H$. $H$ has two vertices $v_0, v_1$ connected with an edge of weight 1. $v_0$ has a self-loop of weight 1 and $v_1$ has a self-loop of weight $\gamma$. $v_0$ has vertex weight $\lambda$ and $v_1$ has vertex weight 1.

For a prime $p$, let $i_p \in \{0, 1, \dots p-1\}$, such that $i_p^2 \equiv -1 \pmod{k}$. For $p \geq 3$ there can be either zero or two elements satisfying this definition. If there are no elements satisfying the definition, then consider the conditions involving $i_p$ vacuous. If there are two elements satisfying the definition, then it does not matter which one we choose. The main result of Chapter 6 is the following.

**Theorem 1.13.** *Let $p$ be a prime and let $\gamma, \lambda \in \{0, 1, \dots, p-1\}$. $\#_p Z_{\gamma,\lambda}$ is computable in polynomial time when one of the following holds.*

1. $\lambda = 0$.
2. $\gamma = 1$.
3. $\gamma = -1$ *and* $\lambda \in \{0, \pm 1, \pm i_p\}$.

*Furthermore, $\#_p Z_{\gamma,\lambda}$ is $\#_p\,\mathsf{P}$-complete when one of the following holds.*

4. $\lambda \not\equiv 0 \pmod{p}$ *and* $\gamma \equiv 0 \pmod{p}$.
5. $\lambda \not\equiv 0 \pmod{p}$, $\gamma \not\equiv \pm 1 \pmod{p}$ *and there exists an integer $k$ with* $\gamma^k \equiv \lambda \pmod{p}$.
6. $\lambda \not\equiv 0 \pmod{p}$, $\gamma \not\equiv \pm 1 \pmod{p}$ *and $p < 100$, where $p \neq 41$.*

## 1.3 Matrix Partitions

A matrix partition of an undirected graph is a partition of its vertices according to a matrix which specifies adjacency and non-adjacency conditions on the vertices, depending on the parts to which they are assigned. For finite sets $D$ and $D'$, the set $\{0, 1, *\}^{D \times D'}$ is the set of $|D| \times |D'|$ matrices $M$ with rows indexed by $D$ and columns indexed by $D'$ where each $M_{i,j} \in \{0, 1, *\}$. For any symmetric matrix $M \in \{0, 1, *\}^{D \times D}$, an *$M$-partition* of an undirected graph $G = (V, E)$ is a function $\sigma \colon V \to D$ such that, for distinct vertices $u$ and $v$,

1. $M_{\sigma(u),\sigma(v)} \neq 0$ if $(u, v) \in E$ and
2. $M_{\sigma(u),\sigma(v)} \neq 1$ if $(u, v) \notin E$.

Thus, $M_{i,j} = 0$ means that no edges are allowed between vertices in parts $i$ and $j$, $M_{i,j} = 1$ means that there must be an edge between every pair of vertices in the two parts and $M_{i,j} = *$ means that any set of edges is allowed between the parts $i$ and $j$. For entries $M_{i,i}$ on the diagonal of $M$, the conditions only apply to distinct vertices in part $i$. Thus, $M_{i,i} = 1$ requires that the vertices of $G$ that are mapped in part $i$ form a clique in $G$, $M_{i,i} = 0$ requires that they form an independent set and $M_{i,i} = *$ imposes no restrictions.

For example if $D = \{1, 2, 3\}$ and $M = \left[\begin{smallmatrix} 0 & * & * \\ * & 0 & * \\ * & * & 0 \end{smallmatrix}\right]$, then an $M$-partition of a graph $G$ is a proper 3-colouring of $G$. In the special case $M \in \{0, *\}^{D\times D}$, an $M$-partition of $G$ is a homomorphism from $G$ to the graph $H_M$, with vertex set $V(H_M) = D$ and edge set $E(H_M) = \{(u, v) \mid u, v \in D \text{ and } M_{u,v} = *\}$. When $M \in \{1, *\}^{D\times D}$, then an $M$-partition of $G$ is a homomorphism from the complement $\overline{G}$ of $G$ to the graph $H_M$ with vertex set $V(H_M) = D$ and edge set $E(H_M) = \{(u, v) \mid u, v \in D \text{ and } M_{u,v} = *\}$.

In the general case where $M$ has entries including both 0 and 1 $M$-partitions can represent other important graph-theoretic structures besides graph homomorphisms. For example, if $D = \{i, c\}$, $M_{i,i} = 0$, $M_{c,c} = 1$ and $M_{c,i} = M_{i,c} = *$, i.e., $M = \left[\begin{smallmatrix} 0 & * \\ * & 1 \end{smallmatrix}\right]$, then an $M$-partition of a graph is a partition of its vertices into an independent set (whose vertices are mapped to $i$) and a clique (whose vertices are mapped to $c$). The independent set and the clique may have arbitrary edges between them. Graphs that have such $M$-partitions are called split graphs [48]. Other examples of graph-theoretic structures that can be represented as $M$-partitions include $(a, b)$-graphs [7], clique-cross partitions [29], and their generalisations. "Type partitions" (see [6]) in extremal graph theory can also be represented as $M$-partitions.

Feder et al. [37] further generalise $M$-partitions by introducing lists. A *list $M$-partition* is an $M$-partition $\sigma$ that is also required to satisfy constraints on the values of each $\sigma(v)$. Let $\mathcal{P}(D)$ denote the powerset of $D$. We say that $\sigma$ *respects* a function $L\colon V(G) \to \mathcal{P}(D)$ if $\sigma(v) \in L(v)$ for all $v \in V(G)$. Thus, for each vertex $v$, $L(v)$ serves as a list of allowable parts for $v$ and a *list $M$-partition* of $G$ is an $M$-partition that respects the given list function. For technical convenience we may allow empty lists, although there are no $M$-partitions that respect any list function $L$ where $L(v) = \emptyset$ for some vertex $v$. List $M$-partitions represent even more complicated graph-theoretic structures. Examples of such list $M$-partitions arise in the proofs of the weak and strong perfect graph conjecture [62, 19] in the form of homogeneous sets. Details of such applications are given by Feder et al. [37].

Feder et al. [37] study the computational complexity of the following decision problem, which is parametrised by a symmetric matrix $M \in \{0, 1, *\}^{D\times D}$.

**Problem 1.14.** *Name.* LIST-$M$-PARTITIONS.

*Parameter.* A symmetric matrix $M \in \{0, 1, *\}^{D\times D}$.

*Input.* A pair $(G, L)$ in which $G$ is a graph and $L$ is a function $V(G) \to \mathcal{P}(D)$.

*Output.* "Yes", if $G$ has an $M$-partition that respects $L$; "no", otherwise.

Note that $M$ is a parameter of the problem rather than an input of the problem. Thus, its size is a constant which does not vary with the input.

A series of papers [33, 35, 36] described in [37] presents a complete dichotomy for LIST-$M$-PARTITIONS problems in which $M$ is a $\{0, *\}$-matrix —thus, giving a dichotomy for the "decision list graph homomorphism problem". More specifically, Feder, Hell and Huang [36] show that, for every $\{0, *\}$-matrix $M$ (and symmetrically, for every $\{1, *\}$-matrix $M$), the problem LIST-$M$-PARTITIONS is either polynomial-time solvable or NP-complete.

There has also been progress in classifying the complexity of the general LIST-$M$-PARTITIONS, where $M$ contains entries from $\{0, 1, *\}$ instead of just $\{0, *\}$ or $\{1, *\}$. Feder et al. [37, Theorem 6.1] give a complete dichotomy for the special case in which $M$ is at most $3 \times 3$, by showing that LIST-$M$-PARTITIONS is either polynomial-time solvable or NP-complete for each such matrix. Later, Feder and Hell studied the LIST-$M$-PARTITIONS problem under the name $\mathrm{CSP}^*_{1,2}(H)$ and showed [34, Corollary 3.4] that, for every $M$, LIST-$M$-PARTITIONS is either NP-complete, or is solvable in $n^{O(\log n)}$ time.

Any instance of LIST-$M$-PARTITIONS is equivalent to a restricted input instance of CSP (Problem 1.4). To see this, let $M$ be a symmetric matrix in $\{0, 1, *\}^{D \times D}$ and let $M_0$ be the relation on $D \times D$ containing all pairs $(i, j) \in D \times D$ for which $M_{i,j} \neq 1$. Let $M_1$ be the relation on $D \times D$ containing all pairs $(i, j) \in D \times D$ for which $M_{i,j} \neq 0$. Then a LIST-$M$-PARTITIONS problem with input $G, L$ can be encoded as a CSP whose constraint language includes the binary relations $M_0$ and $M_1$ and also the unary relations corresponding to the sets in the image of $L$. Each vertex $v$ of $G$ is a variable in the CSP instance with the unary constraint $L(v)$. If $(u, v)$ is an edge of $G$ then it is constrained by $M_1$. If it is a non-edge of $G$, it is constrained by $M_0$. Note that the CSP instance satisfies the restriction that every pair of distinct variables has exactly one constraint, which is either $M_0$ or $M_1$. In a general CSP instance, a pair of variables could be constrained by $M_0$ and $M_1$ or one of them, or neither. It is not clear how to code such a general CSP instance as a general[1] list partitions problem, thus even if a dichotomy for CSP was proved, it would not necessarily apply to LIST-$M$-PARTITIONS.

As we have seen in Section 1.1.2, there has been more success in proving computational dichotomies for the counting constraint satisfaction problems, than its decision counterpart. As in this thesis we are more interested in counting problems, we will focus on computational dichotomies for counting $M$-partition problems. Hell, Hermann and Nevisi [51] have considered the non-listed counting problem #$M$-PARTITIONS.

**Problem 1.15.** *Name.* #$M$-PARTITIONS.

*Parameter.* A symmetric matrix $M \in \{0, 1, *\}^{D \times D}$.

*Input.* A graph $G$.

*Output.* The number of $M$-partitions of $G$.

[1]When $M$ has entries in $\{0, *\}$ or $\{1, *\}$ LIST-$M$-PARTITIONS can be encoded as a CSP with no input restrictions.

Hell et al. prove a dichotomy for small matrices $M$ (of size at most $3 \times 3$). In particular, [51, Theorem 10] together with the graph-homomorphism dichotomy of Dyer and Greenhill [27] (Theorem 1.3) shows that, for every such $M$, #$M$-PARTITIONS is either polynomial-time solvable or $\#\mathsf{P}$-complete. An interesting feature of counting $M$-partitions identified by Hell et al. [51] is that, unlike the situation for homomorphism-counting problems, there are tractable $M$-partition problems with non-trivial counting algorithms. This also applies to our results, which are described bellow (Section 1.3.1). More recently, Dyer, Goldberg and Richerby [26] have extended the dichotomy of Hell et al. of #$M$-PARTITIONS to all matrices of size at most $4 \times 4$.

### 1.3.1 Our results

The results of Chapter 7 are based on the results of the paper "Counting List Matrix Partitions of Graphs" co-authored with Leslie Goldberg, Colin McQuillan, Tomoyuki Yamakami and David Richerby [42]. We consider the list version of the counting $M$-partition problem. More formally, we study the following computational problem.

**Problem 1.16.** *Name.* #LIST-$M$-PARTITIONS.

*Parameter.* A symmetric matrix $M \in \{0,1,*\}^{D\times D}$.

*Input.* A pair $(G, L)$ in which $G$ is a graph and $L$ is a function $V(G) \to \mathcal{P}(D)$.

*Output.* The number of $M$-partitions of $G$ that respect $L$.

The main result presented in Chapter 7 gives a dichotomy theorem for #LIST-$M$-PARTITIONS. As noted above, since there is no known coding of list $M$-partition problems as a constraint satisfaction problem without input restrictions, our theorem is not known to be implied by the dichotomy for #CSP.

**Theorem 1.17.** *For any symmetric matrix $M \in \{0,1,*\}^{D\times D}$, the problem* #LIST-$M$-PARTITIONS *is either in* $\mathsf{FP}$ *or* $\#\mathsf{P}$*-complete.*

We will explicitly state the dichotomy criterion of Theorem 1.17 in Chapter 7. We will also show that the dichotomy criterion is decidable and, in fact, in $\mathsf{NP}$. That is, the problem of deciding whether, for a given matrix $M$, the problem of #LIST-$M$-PARTITIONS is in $\mathsf{FP}$ is in $\mathsf{NP}$. The dichotomy criterion of Theorem 1.17 is the same dichotomy criterion that was shown for #$M$-PARTITIONS, where $M$ is a matrix of size at most $4\times 4$ [51, 26]. Dyer, Goldberg and Richerby [26] conjecture that this dichotomy criterion holds for #$M$-PARTITIONS for all matrices $M$ of arbitrary size.

Finally, we show that list $M$-partitions can be used to encode cardinality restrictions in $M$-partitions problems and we use this to give a polynomial-time algorithm for counting homogeneous pairs in graphs.

## 1.4 Evolutionary dynamics

In the last chapter of this thesis we study the Moran process, an algorithm introduced in biology to model the spread of genetic mutations in populations. In the original Moran process, as introduced in [64], we are given a population of $n$ individuals that can be of two types, mutants and non-mutants. The process has a parameter $r$ which is the fitness of mutants. All non-mutants have fitness 1. At each time step an individual $x$ is chosen for reproduction with probability proportional to its fitness. The chosen individual then replaces another individual $y$ of the population chosen uniformly at random, with a new individual of the same type as $x$. Thus if $x$ is a mutant, then it will replace $y$ with a mutant and if $x$ is a non-mutant it will replace $y$ with a non-mutant. When running this process indefinitely with $n-1$ non-mutants and a single mutant as a starting state, it will either reach a state where the population will consist exclusively of mutants, which we call *fixation*, or reach a state where the population will consist exclusively of non-mutants, which we call *extinction*. One of the core purposes of the Moran process is to compute the fixation probability (or equivalently the extinction probability) of a population.

Lieberman, Hauert and Nowak [59] extended the original Moran process by introducing structured populations in the form of directed graphs. In their model, which from now on we will refer to as the Moran process, the population is represented by a directed graph $G$ where the vertices correspond to the individuals of the population. In the initial state a vertex $v$ is chosen to become a mutant uniformly at random. As with the original model, at each time step, an individual $x$ is chosen for reproduction with a probability proportional to its fitness. Next, an out-neighbour $w$ of $x$ is selected uniformly at random. Finally, the state of vertex $x$ (mutant or non-mutant) is copied to vertex $w$. When run on the undirected clique with $n$ vertices, the Moran process is equivalent to the original model introduced by Moran. Undirected graphs can be viewed as directed graphs with edges in both directions.

When the Moran process is run on a finite strongly connected digraph, it will either reach fixation or extinction. When $r < 1$ then mutation is overwhelmingly likely to go extinct as the single initial mutant has lower fitness than the non-mutants that occupy every other vertex in the graph. If $r = 1$, using the argument in the proof of [23, Lemma 1], we can show that the fixation probability is $\frac{1}{n}$ in any strongly connected graph on $n$ vertices. Thus, in this thesis we will be interested in the probability that the Moran process reaches fixation when $r > 1$, given the topology of the underlying graph.

For example when the Moran process is run on $K_n$, the clique with $n$ vertices, the fixation probability is given by

$$\rho_K(r, n) = \frac{1 - \frac{1}{r}}{1 - \frac{1}{r^n}}$$

and the extinction probability is given by

$$\zeta_K(r,n) = 1 - \rho_K(r,n) = \frac{\frac{1}{r} - \frac{1}{r^n}}{1 - \frac{1}{r^n}}.$$

Note that $\rho_K(r,n)$ is also the fixation probability of the original Moran process, on unstructured populations. To see that the equations involving $\rho_K(r,n)$ and $\zeta_K(r,n)$ hold, suppose that the number of mutants is $m$. Any mutant is chosen for reproduction with probability $\frac{r}{rm+n-m}$ and with probability $\frac{n-m}{n}$ it will reproduce on a non-mutant. Any non-mutant is chosen for reproduction with probability $\frac{1}{rm+n-m}$ and with probability $\frac{m}{n}$ it will reproduce on a mutant. So the probability of the number of mutants increasing at the next time step is $\frac{rm(n-m)}{n(rm+n-m)}$ and the probability of the number of the mutants decreasing at the next time step is $\frac{m(n-m)}{n(rm+n-m)}$. So, in any given state, the number of mutants is $r$ times as likely to increase at the next step of the process as it is to decrease. Thus, the number of mutants on $K_n$ at every time step can be seen as a random walk on the integers, that starts at 1, absorbs at 0 and $n$, increases with probability $\frac{r}{r+1}$ and decreases with probability $\frac{1}{r+1}$. It is well known that this walk absorbs at $n$ with probability $\rho_K(r,n)$ and at 0 with probability $\zeta_K(r,n)$ (see e.g. [39, Chapter XIV]). Thus, as $n$ goes to infinity, the extinction probability tends to $1/r$.

When the Moran process is run on non-regular graphs, we can get lower extinction probability than $\zeta_K(r,n)$. For example consider the undirected star with $n+1$ vertices, a graph consisting of a central vertex $v^*$ and $n$ leaves that are adjacent to $v^*$. As $n \to \infty$, the extinction probability of the $n$-leaf star tends to $\frac{1}{r^2}$ (see [59, 9]). The intuition behind this extinction probability is that the initial mutant is placed on a leaf with probability $\frac{n}{n+1}$ (which tends to 1 as $n \to \infty$) and, at each step, the probability that a leaf mutant gets overwritten is $\frac{1}{n+1} \cdot \frac{m}{n}$ (which tends to 0 as $n \to \infty$), where $m$ is the number of mutants at this step.

The structure of a graph $G$ strongly influences the extinction (resp. fixation) probability of the Moran process, when run on $G$. Lieberman et al. [59] raise the question of how good of an amplifier a graph $G$ can be. To discuss this question we introduce the following definition.

**Definition 1.18.** Consider a function $\zeta(r,n)\colon \mathbb{R}_{>1} \times \mathbb{Z}_{\geq 1} \to \mathbb{R}_{\geq 0}$. An infinite family $\Upsilon$ of directed graphs is said to be *up-to-$\zeta$ fixating* if, for every $r > 1$, there is an $n_0$ (depending on $r$) so that, for every graph $G \in \Upsilon$ with $n \geq n_0$ vertices, the following is true: When the Moran process is run on $G$, starting from a uniformly-random initial mutant, the extinction probability is at most $\zeta(r,n)$.

So, for example, the infinite family of graphs containing all cliques is up-to-$\zeta_K(r,n)$ fixating and, since $\zeta_K(r,n) < 1/r$, it is also up-to-$1/r$ fixating. Similarly, the infinite family containing all stars is up-to-$1/r^2$ fixating. Lieberman et al. [59] refer to graphs which have smaller extinction probability than $\rho_K(r,n)$ (and therefore have larger

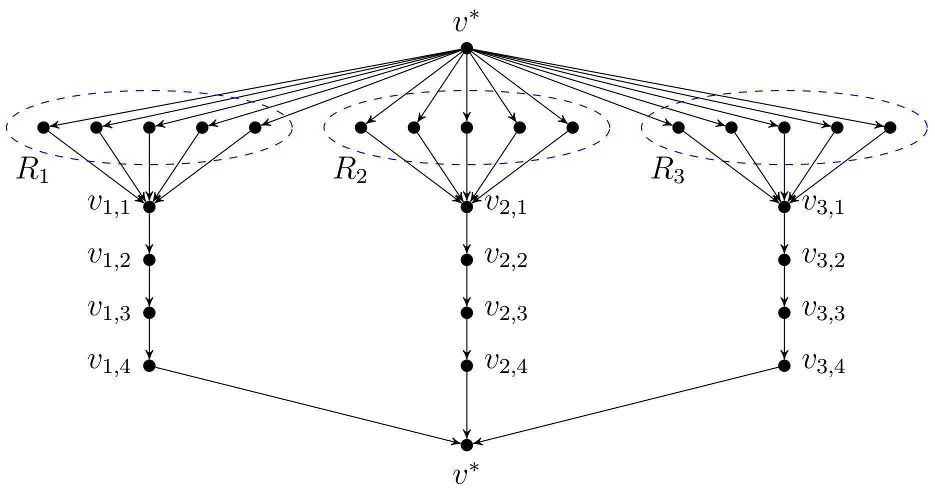

Figure 1.2: The superstar $\mathcal{S}_{4,3,5}$, with $\ell = 3$ reservoirs $R_1$, $R_2$ and $R_3$, each of size $m = 5$, connected by a path with $k = 4$ vertices to $v^*$. The centre vertex $v^*$ is shown twice, at both the top and bottom of the diagram.

fixation probability than $\zeta_K(r, n)$) as *amplifiers*.

**Definition 1.19.** An infinite family of directed graphs is *amplifying* if it is up-to-$\zeta$ fixating for a function $\zeta(r, n)$ which, for every $r > 1$, satisfies $\lim_{n\to\infty} \zeta(r, n) < 1/r$.

Thus from the two families of graphs we have discussed so far (cliques and stars) the infinite family of graphs containing all stars is up-to-$\zeta(r, n)$ fixating for a function $\zeta(r, n)$ satisfying $\lim_{n\to\infty} \zeta(r, n) = 1/r^2$, so this family of graphs is amplifying.

Lieberman et al. [59] were interested in infinite families of digraphs for which the extinction probability tends to 0, prompting the following definition.

**Definition 1.20.** An infinite family of directed graphs is *strongly amplifying* if it is up-to-$\zeta$ fixating for a function $\zeta(r, n)$ which, for every $r > 1$, satisfies $\lim_{n\to\infty} \zeta(r, n) = 0$.

Note that the infinite family of stars is not strongly amplifying since the extinction probability of stars tends to $1/r^2$ rather than to 0.

In [59], Lieberman et al. define the following infinite family of graphs.

**Definition 1.21.** Let $k$, $\ell$ and $m$ be positive integers. The $(k, \ell, m)$-*superstar* is the directed graph $\mathcal{S}_{k,\ell,m}$ defined as follows. (See Figure 1.2.) The vertex set $V(\mathcal{S}_{k,\ell,m})$ of $\mathcal{S}_{k,\ell,m}$ is the disjoint union of $\ell$ size-$m$ sets $R_1, \dots, R_\ell$ (called *reservoirs*) together with $k\ell$ vertices $v_{1,1}, v_{1,2}, \dots, v_{\ell,k}$ and a single centre vertex $v^*$. The edge set of $\mathcal{S}_{k,\ell,m}$ is given by

$$E(\mathcal{S}_{k,\ell,m}) = \bigcup_{i=1}^{\ell} \left( (\{v^*\} \times R_i) \cup (R_i \times \{v_{i,1}\}) \cup \{(v_{i,j}, v_{i,j+1}) \mid j \in [k-1]\} \cup \{(v_{i,k}, v^*)\} \right).$$

Lieberman et al. in [59] claimed that the fixation probability of a superstar with parameter $k$ tends to $1 - r^{-(k+2)}$, providing a heuristic proof sketch for their claimed results. Díaz et al. [22] showed that, for the case $k = 3$, the fixation probability of the superstars is at most $1 - \frac{r+1}{2r^5+r+1}$, which is less than the originally claimed value of $1 - r^{-5}$ for all $r \geq 1.42$. Subsequently, Jamieson-Lane and Hauert [55, Equation (5)] with a more detailed, but still heuristic, analysis claim that for superstars with parameter $k$ and with $\ell = m$, the fixation probability $\rho_k$ has the following bounds for fixed $r > 1$,

$$1 - \frac{1}{r^4(k-1)(1-\frac{1}{r})^2} - o(1) \leq \rho_k \leq 1 - \frac{1}{r^4(k-1)} + o(1), \tag{1.1}$$

where the $o(1)$ terms tend to 0 as $\ell \to \infty$.

### 1.4.1 Our results

The results of Chapter 8 are based on results of the paper "Amplifiers for the Moran Process" co-authored with Andreas Galanis, Leslie Goldberg, John Lapinskas and David Richerby [41]. We prove the following theorem about superstars.

**Theorem 1.22.** *Let $\zeta(r, n)$ be any function such that, for any $r > 1$,*

$$\lim_{n\to\infty} \zeta(r, n)(n \log n)^{1/3} = 0.$$

*Then there is no infinite family of superstars that is up-to-$\zeta$ fixating.*

Our theorem gives an upper bound for the fixation probability of the Moran process run on superstars for all the values of the parameters of a superstar. In Corollary 8.18 (page 162), we identify a wide class of parameters for which the extinction probability is provably at least $1/(1470r^4k)$. Comparing our results with the results of Jameson-Lane and Hauert (1.1), our bound is weaker by a factor of 1470. This is because in our rigorous proof we need to show concentration of all random variables, using Chernoff bounds and other bounds on probabilities. We have written the proof in order to optimise readability instead of optimising the constants, so our constants can presumably be improved.

# Part I

# Modular counting

# Introduction

In Part I of this thesis we study the complexity of two modular counting problems. Recall that the relevant complexity classes we will use for modular counting problems are the classes $\#_k\,\mathsf{P}$, consisting of all the problems of the form "compute $f(x)$ modulo $k$", where $f(x)$ is a function in $\#\,\mathsf{P}$ and $k$ is a non-negative integer. In Chapters 2–5, which are the main part of this thesis, we study the complexity of counting homomorphisms to a graph modulo an integer. Let $\mathrm{Hom}(G \to H)$ be the set of homomorphisms from $G$ to $H$. We are interested in the the following problem.

**Problem 1.6.** *Name.* $\#_k$HomsTo$H$.

*Parameter.* A graph $H$ and an integer $k$.

*Input.* A graph $G$.

*Output.* $|\mathrm{Hom}(G \to H)| \pmod{k}$.

In Chapter 2, and more specifically in Corollary 2.9, we identify some classes of target graphs $H$ for which $\#_p$HomsTo$H$ is computable in polynomial time, where $p$ is a prime. We also prove Theorem 1.9, which establishes pinning for $\#_p$HomsTo$H$, i.e. reduces $\#_p$HomsTo$H$ to $\#_p$PartLabHomsTo$H$, when $H$ is what we call "orbit compatible with $p$". We will then use pinning to identify target graphs $H$ for which the problem is hard to compute.

The main result of Chapter 3 is Theorem 1.10, which gives a dichotomy theorem for $\#_p$HomsTo$H$ for all primes $p$ and all asymmetric trees $H$. For every asymmetric tree $H$ that has at least two vertices and for every prime $p$, $\#_p$HomsTo$H$ is $\#_p\,\mathsf{P}$-hard. Trivially, for every prime $p$ and the graph $H$ that is the empty graph or the graph with one vertex, $\#_p$HomsTo$H$ is computable in polynomial time. In Section 3.6 we give an example of a graph $H$ for which $\#_2$HomsTo$H$ is computable in polynomial time while $\#_4$HomsTo$H$ is hard. This shows that, unlike other results in modular counting (e.g. Guo et al. results on $\#_k$CSP, see [49, Lemma 16 and Lemma 18]), for an integer $r$ and a prime $p$, the complexity of $\#_{p^r}$HomsTo$H$ is not the same as the complexity of $\#_p$HomsTo$H$.

In Chapters 4 and 5 we focus in counting homomorphisms to a graph modulo 2. The main result of Chapter 4 (Theorem 1.11) gives a dichotomy for $\#_2$HomsTo$H$ when $H$ is graph that has no cycles that share edges. That is, when the target graph $H$ reduces by an involution-free reduction to a non-trivial graph, then $\#_2$HomsTo$H$ is $\#_2\,\mathsf{P}$-hard. Otherwise $H$ reduces by an involution-free reduction to a graph with at most one vertex and $\#_2$HomsTo$H$ is, trivially, computable in polynomial time. In Chapter 5, Theorem 1.12 we show that, when the target graph $H$ is any graph that has no 4-cycles, then the computational dichotomy still holds for $\#_2$HomsTo$H$ with the same dichotomy criterion.

In Chapter 6 we study the complexity of computing the partition function of a two-spin system with an external field modulo a prime ($\#_p Z_{\gamma,\lambda}$). $\#_p Z_{\gamma,\lambda}$ can be either polynomial-time computable or $\#_p\,\mathsf{P}$-hard, depending on the values of its parameters $(\gamma, \lambda, p)$. We present values of $(\gamma, \lambda, p)$ for which $\#_p Z_{\gamma,\lambda}$ is computable in polynomial time and we also show values of $(\gamma, \lambda, p)$ for which $\#_p Z_{\gamma,\lambda}$ is hard.

# Chapter 2

# Counting graph homomorphisms (modulo a prime): polynomial cases and pinning

## 2.1 Introduction

This chapter sets up the tools we will use to study the complexity of $\#_p$HomsTo$H$ when $p$ is a prime. This is done in two sections.

In Section 2.3 we present the polynomial-time algorithm of Faben and Jerrum [32] which applies to some values of the parameter space (target graphs $H$ and primes $p$) of $\#_p$HomsTo$H$.

Section 2.4 establishes "pinning", a technique used in all known hardness results for counting homomorphisms modulo an integer. For any graph $H$, a *partially $H$-labelled graph $J = (G, \tau)$* consists of an *underlying graph $G$* and a *pinning function $\tau$*, which in this thesis is a partial function from $V(G)$ to $V(H)$. Thus, every vertex $v$ in the domain of $\tau$ is pinned to a *particular* vertex of $H$. A homomorphism from a partially labelled graph $J = (G, \tau)$ to $H$ is a homomorphism $\sigma\colon G \to H$ such that, for all vertices $v \in \mathrm{dom}(\tau)$, $\sigma(v) = \tau(v)$. The problem that we study is $\#_p$PartLabHomsTo$H$, the problem of computing $|\mathrm{Hom}(J \to H)| \pmod p$, given a partially $H$-labelled graph $J$.

The main result of Section 2.4 is the following.

**Theorem 1.9.** *Let $p$ be a prime and let $H$ be a graph. If $H$ and $p$ are orbit compatible, then $\#_p$PartLabHomsTo$H$ reduces to $\#_p$HomsTo$H$ via polynomial-time Turing reduction.*

Ignoring the "orbit compatibility" property we give the following intuition for the proof of Theorem 1.9. Suppose that we wanted to count the number of homomorphisms $\sigma$ from a partially labelled graph $J = (G, \tau)$ to a graph $H$, with $v \in V(G)$, $\mathrm{dom}(\tau) = \{v\}$ and $\tau(v) = a$. Let $\mathrm{Hom}((G, v) \to (H, a)) = \mathrm{Hom}(J \to H)$ be the set of homomorphisms that respect $\tau$. We could prove Theorem 1.9 by constructing a graph $G'$, such that $|\mathrm{Hom}(G' \to H)| \equiv c|\mathrm{Hom}((G,v) \to (H,a))| \pmod{p}$, where $c \in \mathbb{Z}_p$. This could be achieved as follows. Assume that there was a gadget graph $G_x$ with a distinguished vertex $x$, such that $|\mathrm{Hom}((G_x,x) \to (H,a))| \equiv c \pmod{p}$. Further assume that for any vertex $u \in V(H) - a$, $|\mathrm{Hom}((G_x,x) \to (H,u))| \equiv 0 \pmod{p}$. Now let $G'$ be the graph obtained from the union of a copy of $G$ and a copy $G_x$, by identifying $x$ with $v$. From the structure of $G'$ we have that $|\mathrm{Hom}(G' \to H)| \equiv c|\mathrm{Hom}((G,v) \to (H,a))| \pmod{p}$ and that would give us Theorem 1.9 when the pinning function $\tau$ only maps a single vertex of $G$ to $H$.

Unfortunately we can not prove that such a graph always exists. Instead we develop an algebra on gadgets and show that there is always a set of gadgets $G_1, G_2, \dots G_t$ such that $\sum_{i=1}^{t} |\mathrm{Hom}(G_i \to H)| \equiv |\mathrm{Hom}((G,v) \to (H,a))| \pmod{p}$. Let $(G,\bar{x})$ denote the graph $G$ with $k$ distinguished vertices $x_1, \dots, x_k$, where $k$ is an integer and $\bar{x} = x_1, \dots, x_k$. Essential to proving the existence of such a set of gadgets is a version of a result originally due to Lovász. This (Lemma 2.16) states that, if graphs with distinguished vertices $(H,\bar{y})$ and $(H',\bar{y}')$ are non-isomorphic, there is a graph $(G,\bar{x})$ with a different number of homomorphisms (modulo $p$) to $(H,\bar{y})$ than to $(H',\bar{y}')$. This lemma gives us enough power to guarantee the existence of such gadgets and allows us to pin not only one, but any number of vertices of $G$ at the same time.

### 2.1.1 Organisation

In Section 2.3 we survey the results that will give us a polynomial-time algorithm for identifying and solving the polynomial-time cases of $\#_p$HomsToH.

In the subsections of Section 2.4 we prove Theorem 1.9. In Section 2.4.1 we introduce some results from group theory we will need. In Section 2.4.2 we prove a version of the Lovász result we discussed earlier. In Section 2.4.3 we develop the algebra of gadgets that will allow us to establish pinning. In Section 2.4.4 we introduce "orbit compatibility", the property we require for pinning, and we prove Theorem 1.9. In Section 2.4.4, we also explain how the results we have presented are subtly different from those in the literature so existing results could not be reused directly.

## 2.2 Preliminaries

We write $[n]$ for the set $\{1, \dots, n\}$. For a set $S$ and an element $x$, we often write $S - x$ for $S \setminus \{x\}$. For a function $f$ we write $f^{(k)} = f \circ f \circ \cdots \circ f$, where composition is applied $k$ times.

**Graphs** Unless otherwise specified, graphs are undirected and have no parallel edges and no self-loops. The one exception to this is that we briefly allow self-loops in the proof of Lemma 2.16 (this is clearly stated in the proof).

**Definition 2.2** (Graph homomorphism)**.** A homomorphism from a graph $G$ to a graph $H$ is a function $\sigma : V(G) \to V(H)$, such that $(u, v) \in E(G)$ implies that $(\sigma(u), \sigma(v)) \in E(H)$.

An *isomorphism* between two graphs $G_1, G_2$ is a bijective function $f : V(G_1) \to V(G_2)$, such that $(u, v) \in E(G_1)$ if and only if $(f(u), f(v)) \in E(G_2)$. We say that $G_1$ is isomorphic to $G_2$ and denote it with $G_1 \cong G_2$ if there exists an isomorphism $f : V(G_1) \to V(G_2)$. An *automorphism* of a graph $G$ is an isomorphism from $G$ to itself. $\mathrm{Aut}(H)$ denotes the automorphism group of a graph $H$. A graph is asymmetric if its automorphism group contains only the identity. An *automorphism of order $k$* is an automorphism $\rho$ that is not the identity and $k$ is the smallest integer such that $\rho^{(k)}$ is the identity. An *involution* is an automorphism of order 2. $\mathrm{Hom}(G \to H)$ denotes the set of homomorphisms from a graph $G$ to a graph $H$.

**Partially labelled graphs** For any graph $H$, a *partially $H$-labelled graph* $J = (G, \tau)$ consists of an *underlying graph* $G$ and a pinning function $\tau$ from $V(G)$ to $V(H)$. A vertex $v$ in the domain of the pinning function is said to be *pinned* or *pinned to* $\tau(v)$. We will refer to these graphs as *partially labelled graphs* if the graph $H$ is clear from the context. We sometimes write $G(J)$ and $\tau(J)$ for the underlying graph and pinning function of a partially labelled graph, respectively. We write partial functions as sets of pairs, for example, writing $\tau = \{a \mapsto s, b \mapsto t\}$ for the partial function $\tau$ with $\mathrm{dom}(\tau) = \{a, b\}$ such that $\tau(a) = s$ and $\tau(b) = t$. A homomorphism from a partially labelled graph $J = (G, \tau)$ to $H$ is a homomorphism $\sigma \colon G \to H$ such that, for all vertices $v \in \mathrm{dom}(\tau)$, $\sigma(v) = \tau(v)$. We say that such a homomorphism *respects* $\tau$. By $\mathrm{Hom}(J \to H)$ we denote the set of homomorphisms from $V(G)$ to $V(H)$ that respect $\tau$.

**Distinguished vertices** It is often convenient to regard a graph as having some number of distinguished vertices $x_1, \ldots, x_r$ and we denote such a graph by $(G, x_1, \ldots, x_r)$. Note that the distinguished vertices need not be distinct. We sometimes abbreviate the sequence $x_1, \ldots, x_r$ as $\bar{x}$ and we use $G[\bar{x}]$ to denote the subgraph of $G$ induced by the set of vertices $\{x_1, \ldots, x_r\}$. A homomorphism from a graph $(G, x_1, \ldots, x_r)$ to $(H, y_1, \ldots, y_r)$ is a homomorphism $\sigma$ from $G$ to $H$ with the property that $\sigma(x_i) = y_i$ for each $i \in [r]$. This is the same thing as a homomorphism from the partially $H$-labelled graph $(G, \{x_1 \mapsto y_1, \ldots, x_r \mapsto y_r\})$ to $H$. Given a partially labelled graph $J = (G, \tau)$ and vertices $x_1, \ldots, x_r \notin \mathrm{dom}(\tau)$, a homomorphism from $(G, x_1, \ldots, x_r)$ to $(H, y_1, \ldots, y_r)$ is formally identical to a homomorphism from $J = (G, \tau \cup \{x_1 \mapsto y_1, \ldots, x_r \mapsto y_r\})$ to $H$.

Similarly, we say that two graphs $(G, x_1, \ldots, x_r)$ and $(H, y_1, \ldots, y_s)$ are isomorphic if $r = s$ and there is an isomorphism $\rho \colon V(G) \to V(H)$ such that $\rho(x_i) = y_i$ for each

$i \in [r]$ (note that we may have $G = H$). An automorphism of $(G, x_1, \ldots, x_r)$ is just an automorphism $\rho$ of $G$ with the property that $\rho(x_i) = x_i$ for each $i \in [r]$.

## 2.3 Polynomial-time computable classes of target graphs

We begin by identifying classes of graphs $H$ for which $\#_p$HomsTo$H$ can be solved in polynomial time. As in most counting problems, cancellations play a key role to tractability. When counting graph homomorphisms modulo a prime $p$, the automorphisms of order $p$ of the target graph $H$ help us identify groups of homomorphisms that cancel out. More specifically assume that the target graph $H$ has an automorphism $\rho$ of order $p$. For any homomorphism $\sigma$ from the input graph $G$ to $H$, $\sigma \circ \rho$ is also a homomorphism from $G$ to $H$. In this way we have a set of cardinality $p$ which contains the homomorphisms $\sigma \circ \rho^{(j)}$, where $j \in [p]$. The theorem of Faben and Jerrum [32, Theorem 3.4] captures this intuition. In order to state Theorem 3.4 of Faben and Jerrum we need the following definition.

**Definition 2.3.** Let $H$ be a graph and $\rho$ an automorphism of $H$. $H^\rho$ is the subgraph of $H$ induced by the fixed points of $\rho$.

**Theorem 2.4** (Faben and Jerrum)**.** *For any prime $p$, if $H$ is a graph, and $\rho$ an automorphism of $H$ of order $p$, $|\mathrm{Hom}(G \to H)| \equiv |\mathrm{Hom}(G \to H^\rho)| \pmod{p}$.*

We can now apply the theorem to $H^\rho$ recursively and end up with an even smaller graph.

**Definition 2.5.** $H \Rightarrow_p H'$ if there is an automorphism $\rho$ of $H$ of order $p$ such that $H^\rho = H'$. We will also write $H \Rightarrow_p^* H'$ if either $H \cong H'$ or, for some positive integer $k$, there are graphs $H_1, \ldots, H_k$ such that $H \cong H_1$, $H_1 \Rightarrow_p \cdots \Rightarrow_p H_k$, and $H_k \cong H'$.

Faben and Jerrum have also shown the following [32, Theorem 3.7]

**Theorem 2.6** (Faben and Jerrum)**.** *Given a graph $H$ and a prime $p$, there is (up to isomorphism) exactly one graph $H^{*p}$ that has no automorphism of order $p$ and $H \Rightarrow_p^* H^{*p}$.*

**Definition 2.7.** We call the unique (up to isomorphism) graph $H^{*p}$, with $H \Rightarrow_p^* H^{*p}$, the *order $p$ reduced form* of $H$.

Figure 2.1 illustrates Theorem 2.6, giving an example of an order 3 reduced form of a graph. To compute the number of homomorphisms from $G$ to $H$ modulo $p$ ($\#_p$HomsTo$H$), it suffices to compute the number of homomorphisms from $G$ to $H^{*p}$ modulo $p$.

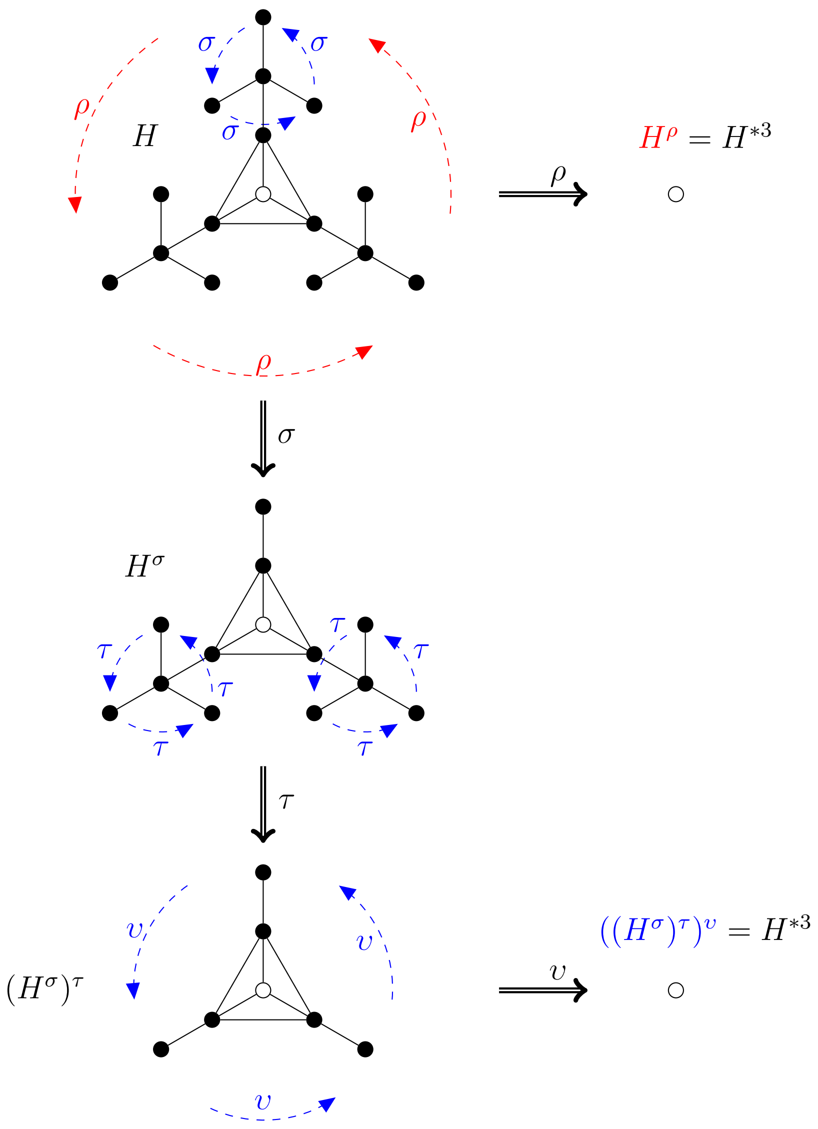

Figure 2.1: An example of the order 3 reduced form $H^{*3}$ of the graph $H$. Here we indicate two different ways of $H \Rightarrow_3^* H^{*3}$. The automorphism $\rho$ has order 3, is indicated with red colour and $H^\rho = H^{*3}$. $\sigma$, $\tau$ and $\upsilon$ are all automorphisms of order 3, and are indicated with blue colour and $((H^\sigma)^\tau)^\upsilon = H^{*3}$.

Recall that #HomsTo$H$ is the problem of computing the number of homomorphisms from $G$ to $H$ ($|\mathrm{Hom}(G \to H)|$). As we have seen in Section 1.1.1, the dichotomy theorem of Dyer and Greenhil [27, Theorem 1.1] completely characterizes the complexity of #HomsTo$H$.

**Theorem 1.3** (Dyer and Greenhil)**.** *Let $H$ be a graph that can have self-loops. If every component of $H$ is a complete bipartite graph with no self-loops or a complete graph with all self-loops present, then* #HomsTo$H$ *can be solved in polynomial time. Otherwise* #HomsTo$H$ *is* #P*-complete.*

A polynomial time algorithm for #HomsTo$H$ would immediately give us a polynomial time algorithm for $\#_p$HomsTo$H$. In this thesis we study the problem of

$\#_p$HomsTo$H$ when $H$ is a simple graph, with no parallel edges and no self-loops. If $H$ has no self-loops, then $H^{*p}$ is also a simple graph. By repeatedly applying the "$\Rightarrow_p$" operation we do not introduce parallel edges or self-loops in the graph, as we are only removing vertices from $H$. Combining the above with Theorems 2.4, 2.6 and 1.3 we have the following.

**Corollary 2.9.** *Let $H$ be a graph. If every component of $H^{*p}$ is a complete bipartite graph with no self-loops, then $\#_p$HomsTo$H$ is computable in polynomial time.*

## 2.4 Partially labelled graphs and pinning

In this section we study the problem $\#_p$PartLabHomsTo$H$ which we briefly described in Section 1.2.2 and Section 2.1 and we formally introduce here.

**Problem 2.10.** *Name.* $\#_p$PartLabHomsTo$H$.

*Parameter.* A graph $H$ and a prime $p$.

*Input.* A partially $H$-labelled graph $J = (G, \tau)$.

*Output.* $|\mathrm{Hom}(J \to H)| \pmod p$.

To show that $\#_p$HomsTo$H$ is at least as hard as $\#_p$PartLabHomsTo$H$ we need to develop some machinery. This generalizes the theorems and lemmas of the current author, Goldberg and Richerby [44] so they can be applied for any prime $p$, and not just when $p = 2$. As in [44] this follows the presentation of similar material by Faben and Jerrum [32] and the earlier paper of the current author, Goldberg and Richerby [43] which, in turn, draw on the work of Lovász [60] and Hell and Nešetřil [53].

### 2.4.1 Group-theoretic background

We will require some results from group theory. All of them can be found in the handbook of Armstrong [2].

For the first, see, e.g., [2, Theorem 13.1].

**Theorem 2.11** (Cauchy's group theorem)**.** *If $\mathcal{G}$ is a finite group and a prime $p$ divides $|\mathcal{G}|$, then $\mathcal{G}$ contains an element of order $p$.*

For a permutation group $\mathcal{G}$ acting on a set $X$, the *orbit* of an element $x \in X$ is the set $\mathrm{Orb}_{\mathcal{G}}(x) = \{\pi(x) \mid \pi \in \mathcal{G}\}$. For a graph $H$, we will abuse notation mildly by writing $\mathrm{Orb}_H(\cdot)$ instead of $\mathrm{Orb}_{\mathrm{Aut}H}(\cdot)$.

The following is a corollary of the orbit–stabiliser theorem [2, Corollary 17.3].

**Theorem 2.12.** *Let $\mathcal{G}$ be a finite permutation group acting on a set $X$. For every $x \in X$, $|\mathrm{Orb}_{\mathcal{G}}(x)|$ divides $|\mathcal{G}|$.*

These two theorems have the following corollary about the size of orbits under the automorphism group of graphs that have no automorphisms of order $p$.

**Corollary 2.13.** *Let $H$ be a graph that has no automorphism of order $p$, where $p$ is a prime. Every orbit of a tuple $\bar{y} \in (V(H))^r$ under the action of $\mathrm{Aut}(H)$ has non-zero cardinality modulo $p$.*

*Proof.* By Theorem 2.11, $|\mathrm{Aut}(H)| \not\equiv 0 \pmod{p}$, since the group contains no element of order $p$. Consider the natural action of $\mathrm{Aut}(H)$ on $(V(H))^r$. By Theorem 2.12, the size of the orbit of $\bar{y}$ in $H$ divides $|\mathrm{Aut}(H)|$ so $|\mathrm{Orb}_H(\bar{y})|$ is also non-zero modulo $p$. $\square$

Finally, we will use Fermat's little theorem (see [2, Theorem 11.6]).

**Theorem 2.14** (Fermat's little theorem)**.** *If $p$ is a prime and $a$ is not a multiple of $p$, then $a^{p-1} \equiv 1 \pmod{p}$.*

### 2.4.2 A Lovász-style lemma

Lovász proved that two graphs $H$ and $H'$ are isomorphic if and only if $|\mathrm{Hom}(G \to H)| = |\mathrm{Hom}(G \to H')|$ for every graph $G$ (in fact, in [61], he proved the analogous result for general relational structures but we do not need this here). We show that this result remains true even if we replace equality of the number of homomorphisms with equivalence modulo a prime $p$. The current author, Goldberg and Richerby [44] have also shown this result for modulo 2. Faben and Jerrum also showed this [32, Lemma 3.10], but in a less general setting than the one that we need. As in [44] our proof is based on the presentation of Hell and Nešetřil [53, Section 2.3].

For the proof we need some definitions, which are used only in this section. We say that two $r$-tuples $\bar{x}$ and $\bar{y}$ *have the same equality type* if, for all $i, j \in [r]$, $x_i = x_j$ if and only if $y_i = y_j$. Let $\mathrm{InjHom}((G, \bar{x}) \to (H, \bar{y}))$ be the set of injective homomorphisms from $(G, \bar{x})$ to $(H, \bar{y})$.

Before proving the main lemma, we prove a simple fact about injective homomorphisms and equality types of distinguished variables.

**Lemma 2.15.** *Let $(G, \bar{x})$ and $(H, \bar{y})$ be graphs, each with $r$ distinguished vertices. If $\bar{x}$ and $\bar{y}$ do not have the same equality type, then $|\mathrm{InjHom}((G, \bar{x}) \to (H, \bar{y}))| = 0$.*

*Proof.* If there are $i, j \in [r]$ such that $x_i = x_j$ but $y_i \neq y_j$, then there are no homomorphisms (injective or otherwise) from $(G, \bar{x})$ to $(H, \bar{y})$, since $x_i$ cannot be mapped simultaneously to both $y_i$ and $y_j$. Otherwise, there must be $i, j \in [r]$ such that $x_i \neq x_j$ but $y_i = y_j$. Then no homomorphism $\eta$ can be injective because we must have $\eta(x_i) = \eta(x_j) = y_i$. $\square$

**Lemma 2.16.** *Let $p$ be a prime and let $(H, \bar{y})$ and $(H', \bar{y}')$ be graphs that have no automorphism of order $p$, each with $r$ distinguished vertices. Then $(H, \bar{y}) \cong (H', \bar{y}')$*

*if and only if, for all (not necessarily connected) graphs $(G,\bar{x})$ with $r$ distinguished vertices,*

$$|\mathrm{Hom}((G,\bar{x}) \to (H,\bar{y}))| \equiv |\mathrm{Hom}((G,\bar{x}) \to (H',\bar{y}'))| \pmod{p}\,. \tag{2.1}$$

*Proof.* If $(H,\bar{y})$ and $(H',\bar{y}')$ are isomorphic, it follows trivially that (2.1) holds for all graphs $(G,\bar{x})$. For the other direction, suppose that (2.1) holds for all $(G,\bar{x})$.

First, we claim that this implies that $\bar{y}$ and $\bar{y}'$ have the same equality type. If they have different equality types then, without loss of generality, we may assume that there are distinct indices $i$ and $j$ such that $y_i = y_j$ but $y'_i \neq y'_j$. Let $G$ be the graph on vertices $\{y_1,\dots,y_r\}$ with no edges: we see that $|\mathrm{Hom}((G,\bar{y}) \to (H,\bar{y}))| = 1 \neq |\mathrm{Hom}((G,\bar{y}) \to (H',\bar{y}'))| = 0$, contradicting the assumption that (2.1) holds for all $G$.

Second, we show by induction on the number of vertices in $G$ that, if (2.1) holds for all $(G,\bar{x})$ then, for all $(G,\bar{x})$,

$$|\mathrm{InjHom}((G,\bar{x}) \to (H,\bar{y}))| \equiv |\mathrm{InjHom}((G,\bar{x}) \to (H',\bar{y}'))| \pmod{p}\,. \tag{2.2}$$

Specifically, under the assumption that (2.1) holds for all $(G,\bar{x})$, we show that (2.2) holds for all $(G,\bar{x})$ with $|V(G)| \leq n_0$ for a suitable value $n_0$ and that, if (2.2) holds for all $(G,\bar{x})$ with $|V(G)| < n$, it also holds for any $(G,\bar{x})$ with $|V(G)| = n$.

Let $n_0 = |\{y_1,\dots,y_r\}| = |\{y'_1,\dots,y'_r\}|$ be the number of distinct elements in $\bar{y}$. For the base case of the induction, consider any graph $(G,\bar{x})$ with $|V(G)| \leq n_0$. If $\bar{x}$ does not have the same equality type as $\bar{y}$ and $\bar{y}'$ (which is guaranteed if $|V(G)| < n_0$) then, by Lemma 2.15,

$$|\mathrm{InjHom}((G,\bar{x}) \to (H,\bar{y}))| = |\mathrm{InjHom}((G,\bar{x}) \to (H',\bar{y}'))| = 0\,.$$

If $\bar{x}$ has the same equality type as $\bar{y}$ and $\bar{y}'$ then, in particular, every vertex of $G$ is distinguished. Any homomorphism from $(G,\bar{x})$ to $(H,\bar{y})$ or $(H',\bar{y}')$ is injective, so

$$\begin{aligned}|\mathrm{InjHom}((G,\bar{x}) \to (H,\bar{y}))| &= |\mathrm{Hom}((G,\bar{x}) \to (H,\bar{y}))| \\ &= |\mathrm{Hom}((G,\bar{x}) \to (H',\bar{y}'))| \\ &= |\mathrm{InjHom}((G,\bar{x}) \to (H',\bar{y}'))|\,,\end{aligned}$$

where the second equality is by the assumption that (2.1) holds for $(G,\bar{x})$.

For the inductive step, let $n > n_0$ and assume that (2.2) holds for all $(G,\bar{x})$ with $|V(G)| < n$. Now, consider some $(G,\bar{x})$ with $|V(G)| = n$.

Given any homomorphism $\sigma$ from $(G,\bar{x})$ to $(H,\bar{y})$, we can define an equivalence relation $\theta$ on $V(G)$ by $(u,v) \in \theta$ if and only if $\sigma(u) = \sigma(v)$. (Note that, if $\sigma$ is injective, then $\theta$ is just the equality relation on $V(G)$.) Write $[\![u]\!]$ for the $\theta$-equivalence class of a vertex $u \in V(G)$. Let $G/\theta$ be the graph whose vertex set is $\{[\![u]\!] \mid u \in V(G)\}$ and whose edge set is $\{(\llbracket u \rrbracket, \llbracket v \rrbracket) \mid (u,v) \in E(G)\}$. For graphs with distinguished vertices, we write $(G, x_1, \dots, x_r)/\theta = (G/\theta, \llbracket x_1 \rrbracket, \dots, \llbracket x_r \rrbracket)$. The homomorphism $\sigma$ from $(G, \bar{x})$ to $(H, \bar{y})$ corresponds to an injective homomorphism from $(G, \bar{x})/\theta$ to $(H, \bar{y})$.

Note that, if there are adjacent vertices $u$ and $v$ in $G$ such that $(u, v) \in \theta$ for some equivalence relation $\theta$, the graph $G/\theta$ has a self-loop on the vertex $\llbracket u \rrbracket$. This is not a problem. Because $H$ is self-loop-free, there are no homomorphisms (injective or otherwise) from such a graph $G/\theta$ to $H$. For the same reason, there are no homomorphisms from $G$ to $H$ that map adjacent vertices $u$ and $v$ to the same place. Therefore, this particular $\theta$ does not correspond to any homomorphism from $G$ to $H$ and contributes zero to the sums below, as required.

We have

$$|\mathrm{Hom}((G,\bar{x}) \to (H,\bar{y}))| = |\mathrm{InjHom}((G,\bar{x}) \to (H,\bar{y}))| + \sum_\theta |\mathrm{InjHom}((G,\bar{x})/\theta \to (H,\bar{y}))|$$
$$|\mathrm{Hom}((G,\bar{x}) \to (H',\bar{y}'))| = |\mathrm{InjHom}((G,\bar{x}) \to (H',\bar{y}'))| + \sum_\theta |\mathrm{InjHom}((G,\bar{x})/\theta \to (H',\bar{y}'))|,$$

where the sums are over all equivalence relations $\theta$, except for the equality relation.

The left-hand sides of these equations are equivalent modulo $p$ by assumption. The sums over $\theta$ on the right are equivalent modulo $p$ by the inductive hypothesis since $\theta$ is not the equality relation, so $G/\theta$ has fewer vertices than $G$. Therefore, (2.2) holds for the graph under consideration.

Finally, it remains to prove that (2.2) holding for all $(G, \bar{x})$ implies that $(H, \bar{y}) \cong (H', \bar{y}')$. To see this, take $(G, \bar{x}) = (H, \bar{y})$. An injective homomorphism from a graph to itself is an automorphism and, since $(H, \bar{y})$ has no automorphism of order $p$, $\mathrm{Aut}(H, \bar{y})$ has no element of order $p$, so $|\mathrm{Aut}(H, \bar{y})| \not\equiv 0 \pmod{p}$ by Cauchy's group theorem (Theorem 2.11). By (2.2), the number of injective homomorphisms from $(H, \bar{y})$ to $(H', \bar{y}')$ is not equivalent to $0 \pmod{p}$, which means that there is at least one such homomorphism. Similarly, taking $(G, \bar{x}) = (H', \bar{y}')$ shows that there is an injective homomorphism from $(H', \bar{y}')$ to $(H, \bar{y})$ and, therefore, the two graphs are isomorphic. □

### 2.4.3 Implementing vectors

The presentation in this section follows very closely that of the current author, Goldberg and Richerby [44], and consequently that of Faben and Jerrum [32], extended to $r$-tuples of distinguished vertices and modulo any prime $p$.

**Definition 2.17.** Let $r \in \mathbb{N}_{>0}$. Fix a graph $H$, that has no automorphism of order $p$, and an enumeration $\bar{y}_1, \dots, \bar{y}_\mu$ of $(V(H))^r$ such that, for every $\bar{y} \in (V(H))^r$, there is exactly one $i \in [\mu]$ such that $(H, \bar{y}) \cong (H, \bar{y}_i)$. We refer to such an enumeration as an enumeration of $(V(H))^r$ *up to isomorphism.*

Note that the number $\mu$ of tuples in the enumeration depends on $H$ and not only in $|V(H)|$.

**Definition 2.18.** Fix a graph $H$, that has no automorphism of order $p$. Let $r \in \mathbb{N}_{>0}$ and let $\bar{y}_1, \ldots, \bar{y}_\mu$ be an enumeration of $(V(H))^r$ up to isomorphism. Let $(G, \bar{x})$ be a graph with $r$ distinguished vertices. We define the vector $\mathbf{v}_H(G, \bar{x}) \in \{0, 1, \ldots, p-1\}^\mu$ where, for each $i \in [\mu]$, the $i$th component of $\mathbf{v}_H(G, \bar{x})$ is given by

$$\Big(\mathbf{v}_H(G, \bar{x})\Big)_i \equiv |\mathrm{Hom}((G, \bar{x}) \to (H, \bar{y}_i))| \pmod p .$$

We say that $(G, \bar{x})$ *implements* this vector.

Define $\oplus^p$ and $\otimes^p$ to be, respectively, component-wise addition and multiplication modulo $p$, of vectors in $(\mathbb{Z}_p)^\mu$.

**Lemma 2.19.** *Let $\bar{x} = x_1 \ldots x_r$, where $r \in \mathbb{N}_{>0}$, and let $(G_1, \bar{x})$ and $(G_2, \bar{x})$ be graphs such that $V(G_1) \cap V(G_2) = \{x_1, \ldots, x_r\}$. Then,*

$$\mathbf{v}_H(G_1 \cup G_2, \bar{x}) = \mathbf{v}_H(G_1, \bar{x}) \otimes^p \mathbf{v}_H(G_1, \bar{x}) .$$

*Proof.* A function $\sigma \colon V(G_1) \cup V(G_2) \to V(H)$ is a homomorphism from $(G_1 \cup G_2, \bar{x})$ to $(H, \bar{y})$ if and only if, for each $i \in \{1, 2\}$, the restriction of $\sigma$ to $V(G_i)$ is a homomorphism from $(G_i, \bar{x})$ to $(H, \bar{y})$. $\square$

In contrast to the component-wise multiplication of $\mathbf{v}_H(G, \bar{x})$, given $(G_1, \bar{x}_1)$ and $(G_2, \bar{x}_2)$, it is not obvious that there is a graph $(G, \bar{x})$, with $\mathbf{v}_H(G, \bar{x}) = \mathbf{v}_H(G_1, \bar{x}_1) \oplus^p \mathbf{v}_H(G_2, \bar{x}_2)$. Given graphs with distinguished vertices $(G_1, \bar{x}_1), \ldots, (G_t, \bar{x}_t)$, we define

$$\mathbf{v}_H\Big((G_1, \bar{x}_1) + \cdots + (G_t, \bar{x}_t)\Big) = \mathbf{v}_H(G_1, \bar{x}_1) \oplus^p \cdots \oplus^p \mathbf{v}_H(G_t, \bar{x}_t)$$

and we say that a vector $\mathbf{v} \in (\mathbb{Z}_p)^\mu$ is *$H$-implementable* if it can be expressed as such a sum.

**Lemma 2.20.** *Let $S \subseteq (\mathbb{Z}_p)^\mu$ be closed under $\oplus^p$ and $\otimes^p$. If $1^\mu \in S$ and, for every distinct $i, j \in [\mu]$, there is a tuple $s = s_1 \ldots s_\mu \in S$ with $s_i \neq s_j$, then $S = (\mathbb{Z}_p)^\mu$.*

*Proof.* It suffices to show that all of the basis vectors of the standard basis[1] in $(\mathbb{Z}_p)^\mu$ are in $S$. Since $S$ is closed under $\oplus^p$ and $\otimes^p$ it follows that all of $(\mathbb{Z}_p)^\mu$ is in $S$.

We show that all the basis vectors are in $S$ by induction on $\mu$. If $\mu = 1$ the induction hypothesis clearly holds as the all-ones vector is the only vector in the standard basis. Assume that the induction hypothesis holds for $\mu - 1$. Then we can construct vectors that agree with the standard basis in the first $\mu - 1$ places without being able to control

[1]The standard basis is the set $\{100\ldots00, 010\ldots00, \ldots, 000\ldots01\}$

what happens in the $\mu$-th place. From the latter and the statement of the lemma, that $1^\mu \in S$, we obtain the following vectors.

$$\begin{array}{lcl}
\mathbf{v}_0 & = & (1 \;\; 1 \;\; 1 \;\; \ldots \;\; 1 \;\;\; 1) \\
\mathbf{v}_1 & = & (1 \;\; 0 \;\; 0 \;\; \ldots \;\; 0 \;\; x_1) \\
\mathbf{v}_2 & = & (0 \;\; 1 \;\; 0 \;\; \ldots \;\; 0 \;\; x_2) \\
\vdots & & \qquad\qquad\quad \vdots \\
\mathbf{v}_\mu & = & (0 \;\; 0 \;\; 0 \;\; \ldots \;\; 1 \;\; x_\mu)
\end{array}$$

where the $x_i$ can take any value in $\mathbb{Z}_p$.

Let $r$ be an integer and let $\mathbf{v} \in (\mathbb{Z}_p)^\mu$. Since $p$ is fixed for the rest of this proof, let $\mathbf{v}^r = \mathbf{v} \otimes^p \cdots \otimes^p \mathbf{v}$, where $\mathbf{v}$ appears $r$-times and let $r\mathbf{v} = \mathbf{v} \oplus^p \cdots \oplus^p \mathbf{v}$, where $\mathbf{v}$ appears $r$-times. Consider the values of each $x_i$. If $x_i \neq 0$, by Theorem 2.14 we have $x_i^{p-1} \equiv 1 \pmod p$. Hence $\mathbf{v}_i^{p-1} = (00\ldots010\ldots01)$. So from now on we can assume that for each $i \in [\mu]$, $x_i \in \{0,1\}$. We have the following three cases.

Case 1. For all $i \in [\mu]$, $x_i = 0$. Then the vector $\mathbf{v} = \mathbf{v}_0 \oplus^p \bigoplus^p{}_{i\in[\mu]}(p-1)\mathbf{v}_i = 0\ldots01$ is the remaining vector that completes the standard basis.

Case 2. There are at least two $i, j$ such that $x_i, x_j = 1$. Then $\mathbf{v} = \mathbf{v}_i \otimes^p \mathbf{v}_j = 0\ldots01$. To obtain the remaining vectors of the standard basis, for each $i \in [\mu]$ with $x_i \neq 0$, we take the vector $\mathbf{v}_i \oplus^p (p-1)\mathbf{v}$.

Case 3. There is exactly one $i \in [\mu]$ with $x_i = 1$. From the statement of the lemma there is a vector $\mathbf{u} \in S$, with $(\mathbf{u})_i = a$, $(\mathbf{u})_\mu = b$, where $a \neq b$. First assume that $a > b$. Let $\mathbf{u}_i = \mathbf{u} \otimes^p \mathbf{v}_i = 0\ldots0a0\ldots0b$ and let $\mathbf{v}_a = (p-a)\mathbf{v}_i = 0\ldots0(p-a)0\ldots0(p-a)$. Then $\mathbf{u}_i \oplus^p \mathbf{v}_a = 0\ldots0(p-a+b)$. Since $a > b$, $(p-a+b)$ is not a multiple of $p$ so, by Theorem 2.14, $(p-a+b)^{p-1} \equiv 1 \pmod p$. Thus, $\mathbf{v} = (\mathbf{u}_i \oplus^p \mathbf{v}_a)^{p-1} = 0\ldots01$ and $\mathbf{v}'_i = (p-1)\mathbf{v} \oplus^p \mathbf{v}_i = 0\ldots010\ldots0$ complete the standard basis.

Now assume that $a < b$. Let $\mathbf{v}_b = (p-b)\mathbf{v}_i = 0\ldots0(p-b)0\ldots0(p-b)$ and so $\mathbf{u}_i \oplus^p \mathbf{v}_b = 0\ldots0(p+a-b)0\ldots0$. Since $a < b$, $(p+a-b)$ is not a multiple of $p$ so, by Theorem 2.14, $(p+a-b)^{p-1} \equiv 1 \pmod p$. Thus $\mathbf{v}' = (\mathbf{u}_i \oplus^p \mathbf{v}_b)^{p-1} = 0\ldots010\ldots0$ and $\mathbf{v}''_i = (p-1)\mathbf{v}' \oplus^p \mathbf{v}_i = 0\ldots01$ complete the standard basis. □

**Corollary 2.21.** *Let $H$ be a graph with no automorphism of order $p$. Every $\mathbf{v} \in \{0,1,\ldots,p-1\}^\mu$ is $H$-implementable.*

*Proof.* Let $S$ be the set of $H$-implementable vectors. $S$ is clearly closed under $\oplus^p$, and is closed under $\otimes^p$ by Lemma 2.19. Let $G$ be the graph on vertices $\{x_1,\ldots,x_r\}$, with no edges. $1^\mu$ is implemented by $(G, x_1, \ldots, x_r)$, which has exactly one homomorphism to every $(H, \bar{y}_i)$. Finally, for every distinct pair $i, j \in [\mu]$, $(H, \bar{y}_i)$ and $(H, \bar{y}_j)$ are not isomorphic, by definition of the enumeration of $r$-tuples. Therefore, by Lemma 2.16, there is a graph $(G, \bar{x})$ such that

$$|\mathrm{Hom}((G,\bar{x}) \to (H,\bar{y}_i))| \not\equiv |\mathrm{Hom}((G,\bar{x}) \to (H,\bar{y}_j))| \pmod p\,.$$

$(G, \bar{x})$ implements a vector $\mathbf{v}$ whose $i$th and $j$th components are different. The corollary follows from Lemma 2.20. $\square$

### 2.4.4 Pinning

Recall the definition of an enumeration $\bar{y}_1, \ldots, \bar{y}_\mu$ of $(V(H))^r$ up to isomorphism (Definition 2.17). What will allow us to pin, i.e. show that $\#_p\textsc{PartLabHomsTo}H$ reduces to $\#_p\textsc{HomsTo}H$ is the following property of $H$ and $p$.

**Definition 2.22.** Let $H$ be a graph and let $p$ be a prime. We say that $H$ and $p$ are *orbit compatible* if $H$ has no automorphism of order $p$ and for every positive integer $r$ and every tuple $\bar{y}$ in $(V(H))^r$, $|\mathrm{Orb}_H(\bar{y})| \equiv 1 \pmod{p}$.

To see why we need $H$ and $p$ to be orbit compatible in order to reduce the problem $\#_p\textsc{PartLabHomsTo}H$ to $\#_p\textsc{HomsTo}H$ consider the following example. Let $p$ be a prime, let $G$ be a graph with $x \in V(G)$ and let $H$ be a graph with $y \in V(H)$. Let $J = (G, \tau)$ be the partially labelled graph with $\tau = \{x \mapsto y\}$. Suppose that we want to compute $|\mathrm{Hom}(J \to H)| \pmod{p}$ using an oracle for $\#_p\textsc{HomsTo}H$. Using the equivalent view of graphs with distinguished vertices, computing $|\mathrm{Hom}(J \to H)| \pmod{p}$ is equivalent to computing $|\mathrm{Hom}((G, x) \to (H, y))| \pmod{p}$. Let $(G', x')$ be a graph with a distinguished vertex, such that $|\mathrm{Hom}((G', x') \to (H, y))| \equiv 1 \pmod{p}$ and $|\mathrm{Hom}((G', x') \to (H, z))| \equiv 0 \pmod{p}$, whenever $z \notin \mathrm{Orb}_H(y)$. For every $y' \in \mathrm{Orb}_H(y)$, $(H, y') \cong (H, y)$, so, by Lemma 2.16, $|\mathrm{Hom}((G', x') \to (H, y'))| \equiv 1 \pmod{p}$. Let $G''$ be the graph obtained from the union of a copy of $G$ together with a copy of $G'$, by identifying $x$ with $x'$. From the structure of $G''$, $|\mathrm{Orb}_H(y)| \cdot |\mathrm{Hom}((G, x) \to (H, y))| \equiv |\mathrm{Hom}(G'' \to H)| \pmod{p}$. This example shows that if $|\mathrm{Orb}_H(y)|$ is a constant number we can compute $|\mathrm{Hom}(J \to H)| \pmod{p}$ by using the oracle for $\#_p\textsc{HomsTo}H$. It turns out that for all the cases we need to show that $|\mathrm{Orb}_H(y)|$ is a constant number, we can show that it is in fact 1. Therefore, we will use the current definition of orbit compatibility which also simplifies some of our proofs.

In the reduction from a general instance of $\#_p\textsc{PartLabHomsTo}H$ to an instance of $\#_p\textsc{HomsTo}H$ we will find (non-constructively) a set of graphs $G_1 \ldots G_t$, such that $\sum_{i=1}^{t} |\mathrm{Hom}(G_i \to H)| \equiv |\mathrm{Hom}(J \to H)| \pmod{p}$. To show that this is the case we need the following lemma that relies on the fact that $H$ and $p$ are orbit compatible.

**Lemma 2.23.** *Let $p$ be a prime and let $H$ be a graph such that $H$ and $p$ are orbit compatible. Let $\bar{y}_1, \ldots, \bar{y}_\mu$ be an enumeration of $V(H)^r$ up to isomorphism. For any graph $G, \bar{x}$ with $r$ distinguished vertices,*

$$|\mathrm{Hom}(G \to H)| \equiv \sum_{i \in [\mu]} (\mathbf{v}_H(G, \bar{x}))_i \pmod{p}.$$

*Proof.* By definition,

$$\sum_{i\in[\mu]} (\mathbf{v}_H(G,\bar{x}))_i \equiv \sum_{i\in[\mu]} |\mathrm{Hom}((G,\bar{x}) \to (H,\bar{y}_i))| \pmod{p}.$$

As $H$ and $p$ are orbit compatible for each $i \in \mu$ , $|\mathrm{Orb}_H(\bar{y}_i)| \equiv 1 \pmod{p}$. We have

$$\sum_{i\in[\mu]} (\mathbf{v}_H(G,\bar{x}))_i \equiv \sum_{i\in[\mu]} |\mathrm{Orb}_H(\bar{y}_i)|\,|\mathrm{Hom}((G,\bar{x}) \to (H,\bar{y}_i))| \pmod{p}.$$

For any $\bar{y} \in \mathrm{Orb}_H(\bar{y}_i)$, $|\mathrm{Hom}((G,\bar{x}) \to (H,\bar{y}))| = |\mathrm{Hom}((G,\bar{x}) \to (H,\bar{y}_i))|$. This is because composing a homomorphism from $(G,\bar{x})$ to $(H,\bar{y})$ with an isomorphism from $(H,\bar{y})$ to $(H,\bar{y}_i)$ gives a homomorphism from $(G,\bar{x})$ to $(H,\bar{y}_i)$. So,

$$\sum_{i\in[\mu]} (\mathbf{v}_H(G,\bar{x}))_i \equiv \sum_{i\in[\mu]} \sum_{\bar{y}\in\mathrm{Orb}_H(\bar{y}_i)} |\mathrm{Hom}((G,\bar{x}) \to (H,\bar{y}))|. \tag{2.3}$$

Every homomorphism from $G$ to $H$ must map $\bar{x}$ to some tuple $\bar{y}$. All such tuples are included exactly once in the double sum of the right hand side of the display equation 2.3. So $\sum_{i\in[\mu]}(\mathbf{v}_H(G,\bar{x}))_i \equiv |\mathrm{Hom}(G \to H)|$ and the lemma follows. □

Now we can show the main theorem of this chapter.

**Theorem 1.9.** *Let $p$ be a prime and let $H$ be a graph. If $H$ and $p$ are orbit compatible, then $\#_p$PartLabHomsTo$H$ reduces to $\#_p$HomsTo$H$ via polynomial-time Turing reduction.*

*Proof.* Let $J = (G,\tau)$ be an instance of $\#_p$PartLabHomsTo$H$. Let $\bar{x} = x_1,\dots,x_r$ be an enumeration of $\mathrm{dom}(\tau)$ and let $\bar{y} = y_1,\dots,y_r = \tau(x_i),\dots,\tau(x_r)$. Moving from the world of partially $H$-labelled graphs to the equivalent view of graphs with distinguished vertices, we wish to compute $|\mathrm{Hom}((G,\bar{x}) \to (H,\bar{y}))|$, modulo $p$.

By definition of the enumeration (up to isomorphism) $\bar{y}_1,\dots,\bar{y}_\mu$, there is some $k \in [\mu]$ such that $(H,\bar{y}) \cong (H,\bar{y}_k)$. Let $\mathbf{v}$ be the vector that has a 1 in position $k$ and has 0 in every other position. By Corollary 2.21, $\mathbf{v}$ is implemented by some sequence $(\Theta_1,\bar{x}_1),\dots,(\Theta_t,\bar{x}_t)$ of graphs with $r$-tuples of distinguished vertices.

For each $i \in [t]$, let $(G_i,\bar{x})$ be the graph that results from taking the union of disjoint copies of $G$ and $\Theta_i$ and identifying the $j$th element of $\bar{x}$ with the $j$th element of $\bar{x}_i$ for each $j \in [t]$. We have

$$\begin{aligned}
\mathbf{v}_H(G,\bar{x}) \otimes^p \mathbf{v} &= \mathbf{v}_H(G,\bar{x}) \otimes^p \mathbf{v}_H\Big((\Theta_1,\bar{x}_1) + \cdots + (\Theta_t,\bar{x}_t)\Big)\\
&= \bigoplus\nolimits^p_{i\in[t]} \Big(\mathbf{v}_H(G,\bar{x}) \otimes^p \mathbf{v}_H(\Theta_i,\bar{x}_i)\Big)\\
&= \bigoplus\nolimits^p_{i\in[t]} \mathbf{v}_H(G_i,\bar{x})\,.
\end{aligned}$$

Now, sum the components of the vectors on the two sides of the equation. On the right, since $H$ and $p$ are orbit compatible, by Lemma 2.23, we have a value congruent

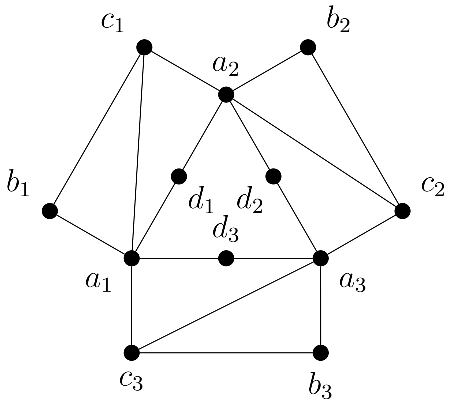

Figure 2.2: An involution-free graph $H$ illustrating the difference between pinning vertices to orbits of vertices and pinning a tuple of vertices to an orbit of a tuple.

modulo $p$ to $\sum_{i\in[t]} |\mathrm{Hom}(G_i \to H)|$. This can be computed by making $t$ calls to an oracle for $\#_p$HomsTo$H$, and $t$ is bounded above by a constant, since $H$ is fixed. On the left, we have, $|\mathrm{Hom}((G,\bar{x}) \to (H,\bar{y}))|$, modulo $p$, which is what we wish to compute. $\square$

The result we have proved here extends the result of the current author, Goldberg and Richerby [44, Theorem 3.1] to hold for all primes, under the condition that $H$ and $p$ are orbit compatible. It appears to be similar to the result of Faben and Jerrum [32, Corollary 4.18] and to an earlier paper of the current author, Goldberg and Richerby [43, Theorem 3.2] but there is an important difference. In [43], we wished to pin $r$ vertices of $G$, each to the orbit of a vertex of $H$. In this thesis, we focus on the problem $\#_p$PartLabHomsTo$H$, where we pin vertices of $G$ to individual vertices of $H$. In order to achieve this, we essentially pin an $r$-tuple of vertices of $G$ to the orbit of an $r$-tuple of vertices in $H$. To see the difference, consider the graph $H$ in Figure 2.2. The orbits of single vertices are $\{a_1, a_2, a_3\}, \ldots, \{d_1, d_2, d_3\}$. There are six homomorphisms from the single edge $(x, y)$ to $H$ that map $x$ to the orbit of $a_1$ and $y$ to the orbit of $d_1$ but only three that map the pair $(x, y)$ to the orbit of the pair $(a_1, d_1)$, which is $\{(a_1, d_1), (a_2, d_2), (a_3, d_3)\}$.

# Chapter 3

# Counting homomorphisms (modulo a prime) to asymmetric trees

## 3.1 Introduction

In this chapter we study the complexity of $\#_p$HOMSTO$H$ when $p$ is a prime and $H$ is an asymmetric tree. Asymmetric trees are connected graphs that have no cycles and their automorphism group only contains the trivial automorphism. To state the main theorem of this chapter recall Definition 2.7 (Page 28) of the order $p$ reduced form $H^{*p}$ of a graph $H$.

**Theorem 1.10.** *Let $p$ be a prime and let $H$ be a graph where $H^{*p}$ is an asymmetric tree. If $H^{*p}$ has more than one vertex then $\#_p$HOMSTO$H$ is $\#_p\mathsf{P}$-hard, otherwise $\#_p$HOMSTO$H$ is computable in polynomial time.*

As we have already explained in Section 1.2.2, Theorem 1.10 actually extends Theorem 1.8 of Faben and Jerrum [32, Theorem 3.8] to all prime moduli. If $H$ is an asymmetric tree, then $H^{*p} = H$, as $H$ has no non-trivial automorphisms. We have the following corollary.

**Corollary 3.2.** *Let $p$ be a prime and $H$ be an asymmetric tree. If $H$ has more than one vertex then $\#_p$HOMSTO$H$ is $\#_p\mathsf{P}$-complete, otherwise $\#_p$HOMSTO$H$ is computable in polynomial time.*

To prove Theorem 1.10 we will use the pinning technique developed in Chapter 2. When $H$ is asymmetric its automorphism group only contains the trivial automorphism, so for any prime $p$, $H$ and $p$ are orbit compatible. Thus we can use Theorem 1.9 and reduce $\#_p$PARTLABHOMSTO$H$ to $\#_p$HOMSTO$H$. In order to show hardness for $\#_p$HOMSTO$H$ it suffices to show hardness for $\#_p$PARTLABHOMSTO$H$. The starting point of our reduction is $\#_pZ_\gamma$: the problem of computing the partition function of a two-spin system on a multigraph modulo a prime. The partition function of a multigraph $G$ is given by

$$Z_\gamma(G) = \sum_{\sigma:V(G)\to\{0,1\}} \gamma^{c(\sigma)},$$

where $c(\sigma)$ denotes the number of edges $(u,v)$ (including self-loops) with $\sigma(u) = \sigma(v) = 1$. An immediate corollary of a result of Guo et al. [49] shows that $\#_p Z_\gamma$ is $\#_p\mathsf{P}$-hard. We then show that for all non-trivial graphs $H$, $\#_p Z_\gamma$ reduces to $\#_p$PartLabHomsTo$H$.

The reduction from $\#_p Z_\gamma$ to $\#_p$PartLabHomsTo$H$ (Theorem 3.10) works as follows. Given a multigraph $G$ as input of $\#_p Z_\gamma$ we construct a partially labelled simple graph $J$, input for $\#_p$PartLabHomsTo$H$. To construct $J$ we replace the vertices and edges of $G$ with partially labelled graphs, which we call “gadgets”. In $J$, the gadget corresponding to the vertex $v$ of $G$ has a vertex $y^v$. We also choose an appropriate vertex $i$ in $H$. Any homomorphism $\sigma$ from $J$ to the target graph $H$ defines a set $I(\sigma) = \{v \in V(G) \mid \sigma(y^v) = i\}$. The configuration of the gadgets ensures that a set $I \subseteq V(G)$ has an appropriate number of homomorphisms $\sigma$ with $I(\sigma) = I$. Each two-spin configuration $\tau$ of $G$ has weight $\gamma^{c(\tau)}$, where $c(\tau)$ can be determined by the set $I'(\tau)$ of vertices that are mapped to “1” by $\tau$. That weight is equivalent to the number of homomorphisms $\sigma$ with $I(\sigma) = I'(\tau)$. Thus, the two-spin partition function of $G$ is equivalent to $|\mathrm{Hom}(J \to H)|$, modulo $p$.

We call the gadgets that we use $(\gamma,p)$-gadgets (Definition 3.9). A $(\gamma,p)$-gadget is chosen according to the structure and properties of $H$. If for an asymmetric graph $H$ we can find a $(\gamma,p)$-gadget, then we can reduce $\#_p Z_\gamma$ to $\#_p$HomsTo$H$. The reduction from $\#_p Z_\gamma$ to $\#_p$HomsTo$H$ can be seen as assigning colours to both the vertices and the edges of $G$, where each “colour” is a vertex of $H$. Each $(\gamma,p)$-gadget has three main parts. One part of the gadget controls which colours can be assigned to each vertex, where each colour corresponds to a spin. The second part controls which colours can be assigned to each edge, where the edge–colours determine the interractions among the vertex–colours. Finally, a third part determines how many homomorphisms there are from $G$ to $H$, given the choice of colours for the vertices and edges.

Since we are working with trees, we are able to use gadgets with very simple structure: our gadgets are essentially paths and we exploit the limited structure of asymmetric trees. We show that every non-trivial asymmetric tree $H$ contains three consecutive vertices of appropriate degrees (Lemma 3.13). These three consecutive vertices can then be used to obtain a $(3,p)$-gadget for $H$ (Lemmas 3.14 and 3.15).

Finally, in this chapter, we discuss the complexity of $\#_k$HomsTo$H$ when $k$ is a composite. The polynomial-time algorithm of Faben and Jerrum works for $\#_k$HomsTo$H$ only when $k$ is a prime. By the Chinese remainder theorem (Theorem 3.20), if $p_1^{r_1}p_2^{r_2}\dots p_m^{r_m} = k$ is the prime factorisation of $k$, then for each $j \in [m]$, the complexity of $\#_k$HomsTo$H$ is equivalent to the complexity of $\#_{p_j^{r_j}}$HomsTo$H$. For a prime $p$ and an integer $r$, if $\#_p$HomsTo$H$ is hard, then $\#_{p^r}$HomsTo$H$ is also hard.

The inverse is not always true. In modular counting problems there have been cases in the literature (Guo et al. [49]) where the complexity of a modular counting problem $\#_p A$ is equivalent to the complexity of $\#_{p^r} A$. We will illustrate that this is not the case with counting graph homomorphisms, as we identify a graph $P_4$ where $\#_2\textsc{HomsTo}P_4$ is easy while $\#_4\textsc{HomsTo}P_4$ is hard.

### 3.1.1 Organisation

In Section 3.3 we formally define $\#_p Z_\gamma$ and show that $\#_p Z_\gamma$ is hard for some value of its parameters. In Section 3.4 we define $(\gamma, p)$-gadgets and show that when $H$ has a $(\gamma, p)$-gadget then $\#_p\textsc{HomsTo}H$ is $\#_p\,\mathsf{P}$-complete. In Section 3.5 we show that every asymmetric tree $H$ has a $(3, p)$-gadget and we also prove Theorem 1.10. Finally in Section 3.6 we show that $\#_4\textsc{HomsTo}P_4$ is $\#_2\,\mathsf{P}$-hard but $\#_2\textsc{HomsTo}P_4$ is computable in polynomial time.

## 3.2 Preliminaries

As in the previous chapter, graphs are undirected and have no parallel edges or self-loops. Multigraphs are also undirected but they may have parallel edges and may have (multiple) self-loops. We will also use partially labelled graphs and graphs with distinguished vertices which we introduced in Section 2.2. We use $\Gamma_H(v)$ to denote the set of neighbours of vertex $v$ in $H$ and $\deg_H(v)$ to denote the degree of $v$ in $H$. A path $P$ is a graph with vertex set $V(P)$ and edge set $E(P)$. However, where convenient, we specify $P$ by simply listing the vertices of the path, in order. We use $d_H(u, v)$ to denote the length of a shortest path from $u$ to $v$ in $H$.

## 3.3 Two-spin systems

We consider the following two-spin model which applies to spin systems on multigraphs. The model has two parameters, a prime number $p$ and an integer $\gamma \in \mathbb{Z}_p$. A configuration $\sigma : V(G) \to \{0, 1\}$ is an assignment of the two spins "0" and "1" to the vertices of $G$. Let $c(\sigma)$ denote the number of edges $(u, v)$ of $G$ with $\sigma(u) = \sigma(v) = 1$. note that it may be $u = v$, so $c(\sigma)$ also includes the number of self-loops $(v, v)$ of $G$ with $\sigma(v) = 1$. The partition function of the model is given by:

$$Z_\gamma(G) = \sum_{\sigma : V(G) \to \{0,1\}} \gamma^{c(\sigma)}.$$

Formally, we are interested in the following modular counting problem.

**Problem 3.3.** *Name.* $\#_k Z_\gamma$.

*Parameter.* $k \in \mathbb{N}_{>0}$ and $\gamma \in \mathbb{Z}_k$.

*Input.* A multigraph $G$.

*Output.* $Z_\gamma(G) \pmod{k}$.

We will identify values of the parameters $k$ and $\gamma$, for which $\#_k Z_\gamma$ is $\#_k\,\mathsf{P}$-hard, to use it as a starting point for the reductions in Section 3.4. To do so, we need to define the Boolean constraint satisfaction problem.

**Problem 3.4.** *Name.* $\#_k\mathrm{CSP}(\mathcal{F})$.

*Parameter.* $k \in \mathbb{N}_{>0}$ and a set of Boolean functions $\mathcal{F} = \{f_1 \dots f_m\}$, where for each $j \in [m]$, $f_j : \{0,1\}^{r_j} \to \mathbb{Z}_p$ and $r_j \in \mathbb{N}_{>0}$.

*Input.* A finite set of constraints over Boolean variables $x_1 \dots x_n$ of the form $f_j(x_{i_{j,1}}, x_{i_{j,2}}, \dots, x_{i_{j,r_j}})$.

*Output.* $\sum_{x_1,\dots,x_n \in \{0,1\}} \prod_j f_j(x_{i_{j,1}}, x_{i_{j,2}}, \dots, x_{i_{j,r_j}}) \pmod{k}$.

The complexity of $\#_k$CSP has been studied by Guo et al. [49]. We are interested in two of their hardness results [49, Corollary 2 and Lemma 11], which will give us the cases for which $\#_p Z_\gamma$ is hard. Before we state their results we define the following.

**Definition 3.5.** We say that a Boolean binary function $F : \{0,1\}^2 \to \mathbb{N}$ is *expressed by the matrix* $A = \left[\begin{smallmatrix} a & b \\ c & d \end{smallmatrix}\right]$ if for $i, j \in \{0,1\}$, $F(i,j) = A_{i,j}$.

The first result of Guo et al. we will use [49, Corollary 2] is a corollary of a result of Faben [30, Theorem 4.11].

**Lemma 3.6** (Guo et al.)**.** *Let $F_1 : \{0,1\}^2 \to \{0,1\}$ be the Boolean binary function expressed by the matrix $A_1 = \left[\begin{smallmatrix} 1 & 1 \\ 1 & 0 \end{smallmatrix}\right]$ and let $F_2 : \{0,1\}^2 \to \{0,1\}$ be the Boolean binary function expressed by the matrix $A_2 = \left[\begin{smallmatrix} 0 & 1 \\ 1 & 1 \end{smallmatrix}\right]$. For all integers $k$, $\#_k\mathrm{CSP}(\{F_1\})$ and $\#_k\mathrm{CSP}(\{F_2\})$ are $\#_k\,\mathsf{P}$-hard.*

The other result of Guo et al. we will use is the following [49, Lemma 11].

**Lemma 3.7** (Guo et al.)**.** *Let $p$ be a prime and let $F : \{0,1\}^2 \to \mathbb{Z}_p$ be the Boolean binary function expressed by the matrix $\left[\begin{smallmatrix} a & b \\ c & d \end{smallmatrix}\right]$, where $abcd \not\equiv 0 \pmod{p}$ and $a^2d^2 \not\equiv b^2c^2 \pmod{p}$. $\#_p\mathrm{CSP}(\{F\})$ is $\#_p\,\mathsf{P}$-hard.*

From the definitions of $\#_p Z_\gamma$ and $\#_p$CSP, we can see that $\#_p Z_\gamma$ is equivalent to $\#_p\mathrm{CSP}(\{F\})$, where $F$ is the Boolean binary function expressed by the matrix $\left[\begin{smallmatrix} 1 & 1 \\ 1 & \gamma \end{smallmatrix}\right]$. We have the following.

**Corollary 3.8.** *For every prime $p$ and $\gamma \in \mathbb{Z}_p$ with $\gamma^2 \not\equiv 1 \pmod{p}$, $\#_p Z_\gamma$ is $\#_p\,\mathsf{P}$-hard.*

*Proof.* If $\gamma \equiv 0 \pmod{p}$ then, by Lemma 3.6 $\#_p Z_\gamma$ is $\#_p\,\mathsf{P}$-hard. Otherwise, $\gamma \not\equiv 0 \pmod{p}$ and $\#_p Z_\gamma$ is $\#_p\,\mathsf{P}$-hard by Lemma 3.7. □

## 3.4 Gadgets

Let $H$ be a target graph for $\#_p$HOMSTO$H$. To define the gadgets that we use, we need choose a set $\Omega_y \subseteq V(H)$ and a vertex $i \in \Omega_y$. Given a multigraph $G$ whose two-spin partition function we wish to compute modulo $p$, we will construct a partially $H$-labelled graph $J$ and consider homomorphisms from $J$ to $H$. $J$ will contain a copy of $V(G)$ and we will be interested in homomorphisms that map every vertex in this copy to $\Omega_y$. Vertices mapped to $i$ will be vertices mapped to "1" in the two-spin configuration under consideration; vertices mapped to $\Omega_y - i$ will be the vertices mapped to "0".

**Definition 3.9.** A $(\gamma, p)$-*gadget* $(i, s, (J_1, y), (J_2, z), (J_3, y, z))$ consists of vertices $i$ and $s$ of $H$ together with three partially labelled graphs with distinguished vertices $(J_1, y)$, $(J_2, z)$ and $(J_3, y, z)$, where $y$ and $z$ are distinct, satisfying the properties explained bellow. Let

$$\begin{aligned}
N_1(v) &= |\mathrm{Hom}((J_1, y) \to (H, v))| \\
N_2(v) &= |\mathrm{Hom}((J_2, z) \to (H, v))| \\
N_3(u, v) &= |\mathrm{Hom}((J_3, y, z) \to (H, u, v))| \\
\Omega_y &= \{u \in V(H) \mid N_1(v) \equiv 1 \pmod p\} \\
\Omega_z &= \{u \in V(H) \mid N_2(v) \not\equiv 0 \pmod p\}\,.
\end{aligned}$$

The properties that we require are:

1. $i \in \Omega_y$, $|\Omega_y| \equiv 2 \pmod p$ and, for each $u \notin \Omega_y$, $N_1(u) \equiv 0 \pmod p$.
2. $\sum_{b \in \Omega_z} N_2(b) \equiv \gamma\kappa$ and $N_2(s) \equiv \kappa \pmod p$, where $\kappa \not\equiv 0 \pmod p$.
3. For each $o \in \Omega_y - i$ and each $x \in \Omega_z - s$, $N_3(o, x) \equiv 0 \pmod p$.
4. And for each $o \in \Omega_y - i$ and each $x \in \Omega_z - s$ the following hold:
   (a) $N_3(i, s) \equiv 1 \pmod p$
   (b) $N_3(o, s) \equiv 1 \pmod p$
   (c) $N_3(i, x) \equiv 1 \pmod p$. □

Now we can show that for every graph $H$ and every prime $p$, such that $H$ and $p$ are orbit compatible and $H$ has a $(\gamma, p)$-gadget, the problem of counting (modulo $p$) graph homomorphisms to $H$ is $\#_p\mathsf{P}$-hard. We will use the following notation to build partially labelled graphs containing many copies of some subgraph. For any "tag" $T$ (which we will treat just as an arbitrary string) and any partially labelled graph $J$, denote by $J^T$ a copy of $J$ with every vertex $v \in V(G(J))$ renamed $v^T$.

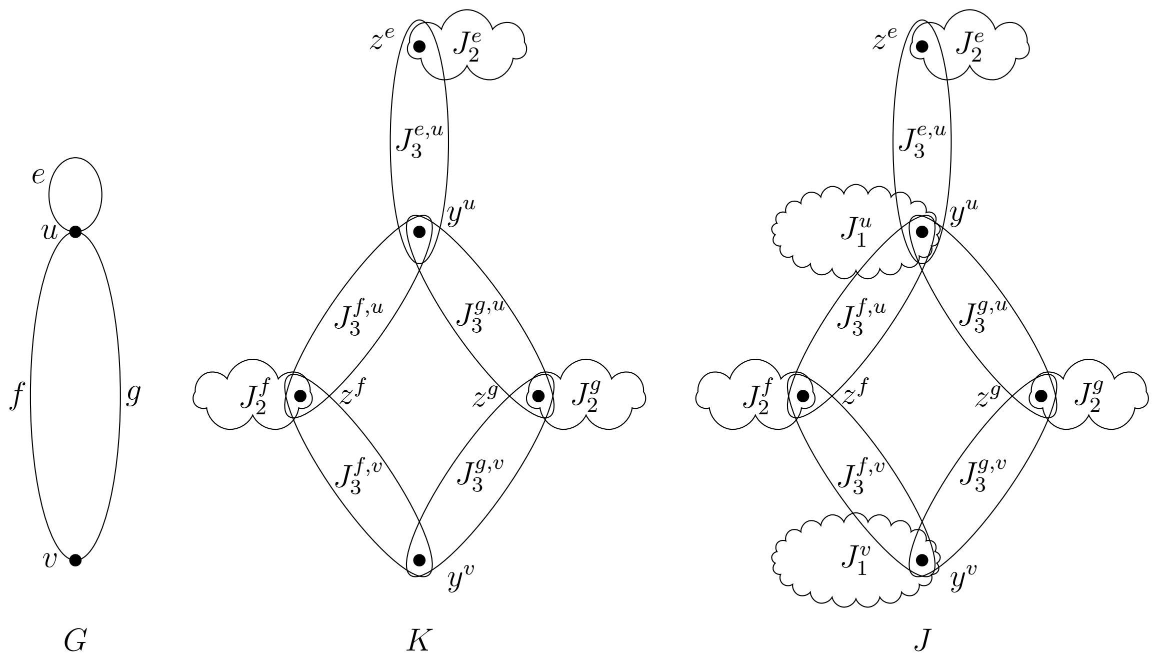

Figure 3.1: The construction of the partially labelled graphs $K$ and $J$ from an example graph $G$, as in the proof of Theorem 3.10. Here graph $G$ consists of two vertices $u, v$ connected with the parallel edges $f, g$ and the self-loop $e$ on vertex $u$.

**Theorem 3.10.** *Let $p$ be a prime and let $H$ be a graph such that $H$ and $p$ are orbit compatible. If $H$ has a $(\gamma, p)$-gadget, with $\gamma \in \mathbb{Z}_p$ and $\gamma^2 \not\equiv 1 \pmod{p}$, then $\#_p$*HomsTo*$H$ is $\#_p\mathsf{P}$-hard.*

*Proof.* Let $(i, s, (J_1, y), (J_2, z), (J_3, y, z))$ be the $(\gamma, p)$-gadget for $H$ and recall the sets $\Omega_y$ and $\Omega_z$ from Definition 3.9. We will reduce $\#_p Z_\gamma$ to $\#_p$PartLabHomsTo$H$. The rest of the proof follows from Corollary 3.8 and Theorem 1.9. Given an input multigraph $G$ to $\#_p Z_\gamma$, we construct an appropriate partially $H$-labelled graph $J$ and show that $|\mathrm{Hom}(J \to H)| \equiv \kappa^{|E(G)|} Z_\gamma(G) \pmod{p}$, where $\kappa$ comes from property 2 of Definition 3.9.

We construct $J$ in two stages (see Figure 3.1). Take the union of disjoint copies $J_3^{e,v}$ of $J_3$ for every edge $e \in G$ and each endpoint $v$ of $e$. If $e$ is a self-loop of $G$, then $J$ will contain a single copy of $J_3^{e,v}$. For each edge $e = (u, v) \in G$, identify the vertices $z^{e,u}$ and $z^{e,v}$ — if $e = (v, v)$ ignore this step. For each vertex $v \in G$, identify all the vertices $y^{e,v}$ such that $e$ has $v$ as an endpoint. For every edge $e \in G$ add a disjoint copy $J_2^e$ of $J_2$ and identify $z^e$ with the already-identified vertices $z^{e,v}$. Call this graph $K$.

We show that $K$ is a simple graph and not a multigraph. By construction, every edge $e$ with distinct endpoints in $G$ corresponds to the graphs $J_3^{e,v}$, $J_3^{e,u}$ and $J_2^e$ in $K$, where $z^{e,v}$, $z^{e,u}$ and $z^e$ are identified. Every pair of parallel edges $e$ and $f$ forms a 2-cycle in $G$. In $K$, this corresponds to a cycle of length $4d_{J_3}(y, z)$. Since $J_3$ is a simple graph with no parallel edges and $d_{J_3}(y, z) > 0$, there are no 1-cycles or 2-cycles in $K$ that correspond to $e$ and $f$ in $G$. So there are no self-loops or parallel edges in $K$

that correspond to $e$ and $f$. Let $e = (v, v)$ be a self-loop in $G$. By construction, this will correspond to the graphs $J_3^{e,v}$ and $J_2^e$ as subgraphs of $K$, where $z^{e,v}$ is identified with $z^e$. As both $J_2$ and $J_3$ are simple graphs, they do not introduce any self-loops or parallel edges in $K$. Hence $K$ is a simple graph with no self-loops or parallel edges.

To make $J$, take $K$ and add a disjoint copy $J_1^v$ of $J_1$ for every vertex $v \in G$ and identify $y^v$ with the already-identified vertices $y^{e,v}$. Since $K$ and $J_1$ are simple graphs, by construction $J$ is also a simple graph.

We now proceed to show that $|\mathrm{Hom}(J \to H)| \equiv \kappa^{|E(G)|} Z_\gamma(G) \pmod p$.

For a homomorphism $\sigma \in \mathrm{Hom}(K \to H)$, let $[\![\sigma]\!]$ be the set of extensions of $\sigma$ to homomorphisms from $J$ to $H$, i.e.,

$$[\![\sigma]\!] = \{\sigma' \in \mathrm{Hom}(J \to H) \mid \sigma(v) = \sigma'(v) \text{ for all } v \in V(G(K))\}.$$

Every homomorphism from $J$ to $H$ is the extension of a unique homomorphism from $K$ to $H$, so we have

$$|\mathrm{Hom}(J \to H)| \quad = \sum_{\sigma \in \mathrm{Hom}(K \to H)} |[\![\sigma]\!]|. \tag{3.1}$$

From the structure of $J$, we have

$$|[\![\sigma]\!]| = \prod_{v \in V(G)} \Big|\mathrm{Hom}((J_1, y) \to (H, \sigma(y^v)))\Big|.$$

By Definition 3.9 (property 1), $|\mathrm{Hom}((J_1, y) \to (H, a))| \equiv 1 \pmod p$ if $a \in \Omega_y$, otherwise $|\mathrm{Hom}((J_1, y) \to (H, a))| \equiv 0 \pmod p$. Therefore, $|[\![\sigma]\!]|$ is non-zero (modulo $p$) if and only if $\sigma$ maps every vertex $y^v$ into $\Omega_y$; call such a homomorphism "legitimate" with respect to $J_1$. We can rewrite (3.1) as

$$|\mathrm{Hom}(J \to H)| \equiv |\{\sigma \in \mathrm{Hom}(K \to H) \mid \sigma \text{ is legitimate}\}| \pmod p, \tag{3.2}$$

and, from this point, we restrict our attention to legitimate homomorphisms.

Given a legitimate homomorphism $\sigma \in \mathrm{Hom}(K \to H)$, let $\sigma|_Y$ be the restriction of $\sigma$ to the domain $\{y^v \mid v \in V(G)\}$. Write $\sigma \sim_Y \sigma'$ if $\sigma|_Y = \sigma'|_Y$ and write $[\![\sigma]\!]_Y$ for the $\sim_Y$-equivalence class of $\sigma$. The classes $[\![\sigma]\!]_Y$ partition the set of legitimate homomorphisms from $K$ to $H$.

Recall from Definition 3.9 that $N_2(v) = |\mathrm{Hom}((J_2, z) \to (H, v))|$ and recall that $N_3(u, v) = |\mathrm{Hom}((J_3, y, z) \to (H, u, v))|$. Let

$$n(u, u') = \sum_{b \in \Omega_z} N_3(u, b) N_3(u', b) N_2(b)$$

and let

$$n'(u) = \sum_{b \in \Omega_z} N_3(u,b)N_2(b).$$

From the structure of $K$ we have,

$$\left|\llbracket\sigma\rrbracket_Y\right| = \left( \prod_{\substack{(u,v)\in E(G),\\ u\neq v}} n(\sigma(u),\sigma(v)) \right) \left( \prod_{(u,u)\in E(G)} n'(\sigma(u)) \right).$$

We will claim that for $u \in \Omega_y$ and for $o \in \Omega_y \setminus \{i\}$ the following hold:

- $n(i,i) \equiv \gamma\kappa \pmod{p}$,
- $n'(i) \equiv \gamma\kappa \pmod{p}$,
- $n(o,u) \equiv n(u,o) \equiv \kappa \pmod{p}$ and
- $n'(o) \equiv \kappa \pmod{p}$.

To prove the claim, recall from Definition 3.9 that $\sum_{b\in\Omega_z} N_2(b) = \gamma\kappa$ (property 2) and that for $b \in \Omega_z$, $N_3(i,b) \equiv 1 \pmod{p}$ (property 4). Therefore, both $n(i,i) \equiv \sum_{b\in\Omega_z} N_2(b) \equiv \gamma\kappa \pmod{p}$ and $n'(i) \equiv \sum_{b\in\Omega_z} N_2(b) \equiv \gamma\kappa \pmod{p}$. For $o \in \Omega_y \setminus \{i\}$, $N_3(o,s) \equiv 1 \pmod{p}$ and for $x \in \Omega_z \setminus \{s\}$, $N_3(o,x) \equiv 0 \pmod{p}$ —this comes from properties 3 and 4 of Definition 3.9. So, when at least one of $u$ or $u'$ is not $i$, the only non-zero term in the sum defining $n(u,u')$ is $N_3(u,s)N_3(u',s)N_2(s)$. As $N_3(u,s) \equiv N_3(u',s) \equiv 1 \pmod{p}$, from property 2 we have $n(u,u') \equiv N_2(s) \equiv \kappa \pmod{p}$. Similarly, $n'(o) \equiv \kappa \pmod{p}$. This concludes the proof of the claim.

For a homomorphism $\sigma : V(K) \to V(H)$, let $c'(\sigma)$ be the number of edges $(v,v') \in E(G)$, where $v$ and $v'$ are not necessarily distinct, with $\sigma(v) = \sigma(v') = i$. From the above claim, we have,

$$|\llbracket\sigma\rrbracket_Y| \equiv \kappa^{|E(G)|}\gamma^{c'(\sigma)}. \tag{3.3}$$

Choose representatives $\sigma_1, \ldots, \sigma_k$, one from each $\sim_Y$-equivalence class. From (3.2) and the fact that the classes $\llbracket\sigma\rrbracket_Y$ partition the legitimate homomorphisms from $K$ to $H$, we have,

$$|\mathrm{Hom}(J \to H)| \equiv \sum_{j=1}^{k} \left|\llbracket\sigma_j\rrbracket_Y\right| \pmod{p}.$$

From (3.3), the definition of $\sim_Y$-equivalence and the definition of a legitimate homomorphism we have,

$$|\mathrm{Hom}(J \to H)| \equiv \kappa^{|E(G)|} \sum_{\sigma:V(G)\to\Omega_y} \gamma^{c'(\sigma)} \pmod{p}. \tag{3.4}$$

Given $\sigma, \sigma' : V(G) \to \Omega_y$, we write $\sigma \sim_i \sigma'$ if, for all $v \in V(G)$, $\sigma(v) = i \Leftrightarrow \sigma'(v) = i$. Let $\llbracket\sigma\rrbracket_i$ denote the $\sim_i$ equivalence-class of $\sigma$. The classes $\llbracket\sigma\rrbracket_i$ partition the set of functions from $V(G)$ to $\Omega_y$ and, by definition, for $\sigma \sim_i \sigma'$, $c'(\sigma) = c'(\sigma')$.

Choose representatives $\sigma_1, \dots \sigma_m$, one from each $\sim_i$ equivalence class. From (3.4) we have

$$|\mathrm{Hom}(J \to H)| \equiv \kappa^{|E(G)|} \sum_{j=1}^{m} |[\![\sigma]\!]_i| \gamma^{c'(\sigma_j)} \pmod{p}. \tag{3.5}$$

For $\sigma : V(G) \to \Omega_y$, let $\ell(\sigma)$ denote the number of vertices $v \in V(G)$ with $\sigma(v) \in \Omega_y \setminus \{i\}$. $|[\![\sigma]\!]_i| \equiv (|\Omega_y| - 1)^{\ell(\sigma)} \pmod{p}$. Recall from property 1 of Definition 3.9 that $|\Omega_y| \equiv 2 \pmod{p}$, so $|[\![\sigma]\!]_i| \equiv 1 \pmod{p}$. For each $\sigma : V(G) \to \Omega_y$, there is exactly one configuration $\sigma' : V(G) \to \{0, 1\}$, such that for each $v \in V(G)$, $\sigma(v) = i \Leftrightarrow \sigma'(v) = 1$. Therefore there is a bijection between the equivalence classes $[\![\sigma]\!]_i$ and distinct mappings $\sigma' : V(G) \to \{0, 1\}$. Using this bijection and (3.5) we have,

$$|\mathrm{Hom}(J \to H)| \equiv \kappa^{|E(G)|} \sum_{\sigma:V(G)\to\{0,1\}} \gamma^{c(\sigma)} \pmod{p}.$$

Recall the definition of the partition function $Z_\gamma(G) = \sum_{\sigma:V(G)\to\{0,1\}} \gamma^{c(\sigma)}$. We have

$$|\mathrm{Hom}(J \to H)| \equiv \kappa^{|E(G)|} Z_\gamma(G) \pmod{p}.$$ □

In this chapter we are interested in asymmetric trees. Recall from Section 2.2, that if a graph $H$ is asymmetric, its automorphism group contains only the trivial automorphism. Hence, for any prime $p$, $H$ has no automorphism of order $p$ and for every $r \in \mathbb{N}_{>0}$ and for every $r$-tuple $\bar{y} \in (V(H))^r$, $|\mathrm{Orb}_H(\bar{y})| \equiv 1 \pmod{p}$. Recall Definition 2.22 of orbit compatibility. We have the following observation.

**Observation 3.11.** Let $p$ be a prime and let $H$ be an asymmetric graph. $H$ and $p$ are orbit compatible.

To show hardness for $\#_p$HomsTo$H$, when $H$ is an asymmetric tree we can now use the following.

**Corollary 3.12.** *Let $p$ be a prime and let $H$ be an asymmetric graph. If $H$ has a $(\gamma, p)$-gadget, with $\gamma \in \mathbb{Z}_p$ and $\gamma^2 \not\equiv 1 \pmod{p}$, then $\#_p$HomsTo$H$ is $\#_p\mathsf{P}$-hard.*

*Proof.* This is immediate from Theorem 3.10 and Observation 3.11. □

## 3.5 A dichotomy for asymmetric trees

In this section we will show that every asymmetric tree has a $(\gamma, p)$-gadget, whith $\gamma^2 \not\equiv 1 \pmod{p}$ for all primes $p$. Our first lemma shows that every asymmetric tree has three adjacent vertices of appropriate degree. We will use this to construct $(\gamma, p)$-gadgets.

The following lemma extends [32, Lemma 5.3] of Faben and Jerrum by following a similar proof approach.

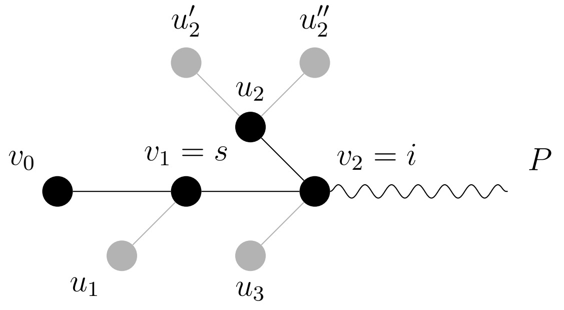

Figure 3.2: An illustration of the proof of Lemma 3.13. The vertices that appear with grey are the vertices that their existence is contradicted in the proof. The rest of the longest path of the graph $P$ is indicated with a wavy line.

**Lemma 3.13.** *Every asymmetric tree $H$ with more than one vertex has a vertex $s$ of degree 2 that is adjacent to a leaf. Furtherore, $s$ is adjacent to a vertex $i$, where $\deg_H(i) \in \{2, 3\}$.*

*Proof.* Let $P = v_0 \ldots v_k$ be a longest path in $H$. Since stars have non-trivial automorphisms, $H$ cannot be a star and $k \geq 3$. Let $v_1 = s$ and $v_2 = i$. For an illustration of the proof that follows see Figure 3.2.

From the maximality of the length of $P$, $v_0$ must be a leaf vertex. Now assume for contradiction that $u_1 \in \Gamma_H(v_1) \setminus \{v_0, v_2\}$. If $u_1$ is not a leaf, then the maximality of the length of $P$ is contradicted. If $u_1$ is a leaf, then $H$ has a non-trivial automorphism $\tau$, with $\tau(u_1) = v_0$ and $\tau(v_0) = u_1$. Hence $u_1$ cannot exist and $v_1$ has degree 2.

We claim that every neighbour of $v_2$ that is not in $P$ must be a leaf. Let $u_2 \in \Gamma_H(v_2) \setminus \{v_1, v_3\}$. Assume for contradiction that $u_2$ is not a leaf. By the maximality of the length of $P$, any vertex $u_2' \in \Gamma_H(u_2) \setminus \{v_2\}$ must be a leaf. First assume that there are at least two vertices $u_2', u_2'' \in \Gamma_H(u_2) \setminus \{v_2\}$. Since both of them can only be leaves, $H$ has a non-trivial automorphism $\tau$ with $\tau(u_2') = u_2''$ and $\tau(u_2'') = u_2'$. Hence $u_2$ can only have one neighbour $u_2'$ that is a leaf. But now $H$ has a non-trivial automorphism $\tau$ with $\tau(v_0) = u_2'$, $\tau(v_1) = u_2$, $\tau(u_2') = v_0$ and $\tau(u_2) = v_1$, so the claim follows.

Finally we show that $v_2$ can only have one neighbour $u_2 \in \Gamma_H(v_2) \setminus \{v_1, v_3\}$. If there was $u_3 \neq u_2$ with $u_2 \in \Gamma_H(v_2) \setminus \{v_1, v_3\}$ then $H$ would have a non-trivial automorphism $\tau$ with $\tau(u_2) = u_3$ and $\tau(u_3) = u_2$. □

Now we will show that if $H$ has a degree 2 vertex adjacent to a vertex of degree $\gamma$, then $H$ has a $(\gamma, p)$-gadget.

**Lemma 3.14.** *Let $p$ be a prime and let $H$ be a tree. If $H$ has a vertex $s$ of degree 2 that is adjacent to a vertex $i$ of degree $\gamma$, then $H$ has a $(\gamma, p)$-gadget.*

*Proof.* We define the $(\gamma, p)$-gadget of $H$ to be $(i, s, (J_1, y), (J_2, z), (J_3, y, z))$, where $J_1, J_2, J_3$ are defined as follows (See Figure 3.3):

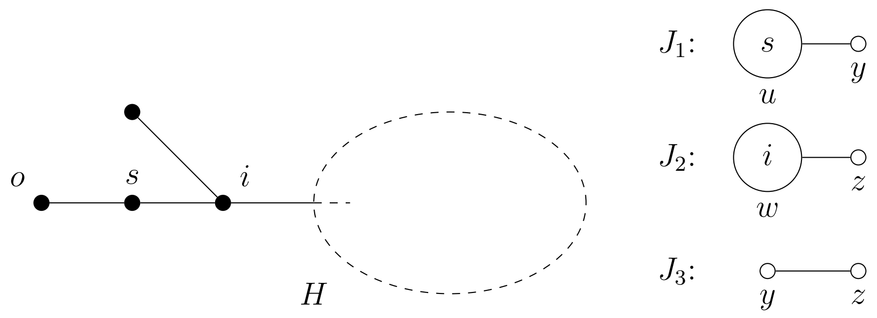

Figure 3.3: A $(\gamma, p)$-gadget for the asymmetric tree $H$ as in the proof of Lemma 3.14, where $\gamma = 3$. In the figure we assumed that $o$ is a leaf, but it is not needed for the proof of Lemma 3.14. Normal vertices appear as black dots, distinguished vertices as small white circles. Pinned vertices appear as large white circles where the label inside the vertex indicates what the vertex is pinned to. The ellipse denotes the rest of the graph $H$.

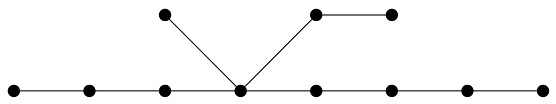

Figure 3.4: An example of an asymmetric tree that does not have a degree 2 vertex adjacent to a degree 3 vertex.

- $J_1$ is the edge $(u, y)$, with $\tau(J_1) = \{u \mapsto s\}$.
- $J_2$ is the edge $(w, z)$, with $\tau(J_2) = \{w \mapsto i\}$.
- $J_3$ is the edge $(y, z)$.

It remains to show that properties 1-4 of Definition 3.9 hold.

$G(J_1)$ is just an edge so, for each $v \in \Gamma_H(s)$, $N_1(v) = 1$ and, for every $v' \notin \Gamma_H(v_1)$, it holds that $N_1(u) = 0$. Therefore $\Omega_y = \Gamma_H(s)$. Clearly $i \in \Omega_y$ and $|\Omega_y| = \deg_H(s) = 2$. Hence property 1 of Definition 3.9 holds.

Similarly, as $J_2$ is a simple edge, $\Omega_z = \Gamma_H(i)$ and for any $b \in \Omega_z$, $N_2(b) = 1 \not\equiv 0 \pmod p$. Therefore $\sum_{b \in \Omega_z} N_2(b) = \deg_H(i) = \gamma$ and $N_2(s) = 1$, hence property 2 of Definition 3.9 holds, with $\kappa = 1$.

$G(J_3)$ is a simple edge, so for $a, b \in V(G)$, $N_3(a, b) = 1$ if and only if $a$ is adjacent to $b$, otherwise $N_3(a, b) = 0$. Let $o \in \Omega_y - i$ and let $x \in \Omega_z - s$. Since there is a path $P = osix$ in $H$, $o$ is not adjacent to $x$, otherwise there would be a cycle in $H$. Thus $N_3(o, x) = 0$ and property 3 of Definition 3.9 holds. Since $\Omega_y = \Gamma_H(s)$, both $i$ and $o$ are adjacent to $s$. Therefore, $N_3(i, s) = N_3(o, s) = 1$ and properties 4(a) and 4(b) of Definition 3.9 hold. Every $x \in \Omega_z - s$ is adjacent to $i$, so $N_3(i, x) = 1$ and property 4(c) Definition 3.9 holds. □

When $p = 3$ there exists assymetric trees, for which applying Lemma 3.14 would give us a $(2, 3)$-gadget. Such an example is shown in Figure 3.4. This is not sufficient

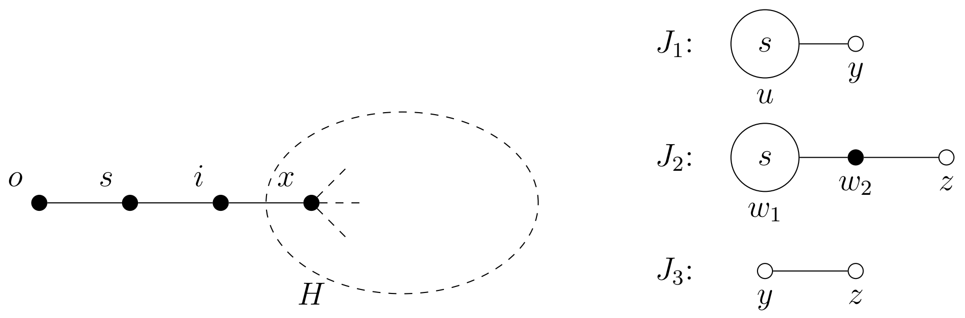

Figure 3.5: A $(3, 3)$-gadget for the asymmetric tree $H$ as in the proof of Lemma 3.15. Normal vertices appear as black dots, distinguished vertices as small white circles. Pinned vertices appear as large white circles where the label inside the vertex indicates what the vertex is pinned to. The ellipse denotes the rest of the graph $H$.

to give us hardness for $\#_3$HomsTo$H$, as $2^2 \equiv 1 \pmod 3$, so Theorem 3.10 doesn't apply. By Lemma 3.13, if $H$ is an asymmetric tree that does not have a degree 2 vertex adjacent to a degree 3 vertex, then it must have two adjacent vertices of degree 2 and one of them is adjacent to a leaf. It turns out that this structure is sufficient to give us a $(3, 3)$-gadget for $H$.

**Lemma 3.15.** *Let $p$ be a prime and let $H$ be a tree. If $H$ has a vertex $s$ of degree 2 that is adjacent to a leaf $o$ and a vertex $i$ of degree* 2*, then $H$ has a* $(3, 3)$*-gadget.*

*Proof.* We define the $(3, p)$-gadget of $H$ to be $(i, s, (J_1, y), (J_2, z), (J_3, y, z))$, where $J_1, J_2, J_3$ are defined as follows (see Figure 3.5):

- $J_1$ is the edge $(u, y)$, with $\tau(J_1) = \{u \mapsto s\}$.
- $J_2$ is the path $w_1w_2z$, with $\tau(J_2) = \{w_1 \mapsto s\}$.
- $J_3$ is the edge $(y, z)$.

It remains to show that properties 1-4 of Definition 3.9 hold.

$G(J_1)$ is just an edge so, for each $v \in \Gamma_H(s)$, $N_1(v) = 1$ and, for every $v' \notin \Gamma_H(v_1)$, it holds that $N_1(u) = 0$. Therefore $\Omega_y = \Gamma_H(s)$. Clearly $i \in \Omega_y$ and $|\Omega_y| = \deg_H(s) = 2$. Hence, property 1 of Definition 3.9 holds.

Let $x \neq s$ be the other neighbour of $i$ and consider $G(J_2)$. Any homomorphism $\sigma : V(G(J_2)) \to H$ that respects $\tau(J_2)$ must have $\sigma(w_2) \in \Gamma_H(s) = \{o, i\}$. If $\sigma(w_2) = o$, then $\sigma(z) = s$, otherwise $\sigma(w_2) = i$ and $\sigma(z) \in \Gamma_H(i) = \{s, x\}$. This shows that $\Omega_z = \{s, x\}$, $N_2(s) = 2$ and $N_2(x) = 1$. So $\sum_{b\in\Omega_z} N_2(b) = 3 \equiv 6 \pmod 3$ and property 2 of Definition 3.9 holds for $\kappa = 2$.

$G(J_3)$ is a simple edge, so for $a, b \in V(G)$, $N_3(a, b) = 1$ if and only if $a$ is adjacent to $b$, otherwise $N_3(a, b) = 0$. Since $o$ is a leaf that is only adjacent to $s$, $N_3(o, x) = 0$ and property 3 of Definition 3.9 holds.

Since $\Omega_y = \Gamma_H(s)$, both $i$ and $o$ are adjacent to $s$. Therefore $N_3(i,s) = N_3(o,s) = 1$ and properties 4(a) and 4(b) of Definition 3.9 hold. Every $x \in \Omega_z - s$ is adjacent to $i$, so $N_3(i,x) = 1$ and property 4(c) Definition 3.9 holds. $\square$

Recall that we have shown that when an asymmetric graph $H$ contains a $(\gamma, p)$-gadget, where $\gamma^2 \not\equiv 1 \pmod{p}$ then $\#_p\textsc{HomsTo}H$ is $\#_p\,\mathsf{P}$-hard (Theorem 3.10). The latter combined with Lemmas 3.13, 3.14 and 3.15 suffices to show that, for every asymmetric tree $H$ with more than one vertex $\#_p\textsc{HomsTo}H$ is $\#_p\,\mathsf{P}$-hard.

**Lemma 3.16.** *Let $H$ be an asymmetric tree with more than one vertex. For any prime $p \geq 3$, $\#_p\textsc{HomsTo}H$ is $\#_p\,\mathsf{P}$-hard.*

*Proof.* Since $H$ is an asymmetric tree and has more than one vertex we can apply Lemma 3.13 to find a vertex $s$, with $\deg_H(s) = 2$, that is adjacent to a leaf $o$ and a vertex $i$ with $\deg_H(i) \in \{2,3\}$. Let $\gamma = \deg_H(i)$.

If $p > 3$, then, by Lemma 3.14, $H$ has a $(\gamma,p)$-gadget. Since $\gamma \in \{2,3\}$ and $p > 3$, we have that $\gamma^2 \not\equiv 1 \pmod{p}$ and the lemma follows from Theorem 3.10.

If $p = 3$ and $\gamma = 3$, then, by Lemma 3.14, $H$ has a $(3,3)$-gadget; otherwise $\gamma = 2$ and, by Lemma 3.15, $H$ has a $(3,3)$-gadget. The lemma follows from Theorem 3.10. $\square$

When $p = 2$, hardness for $\#_2\textsc{HomsTo}H$ comes from the results of Faben and Jerrum [32, Lemma 5.4] and [32, Theorem 5.6].

**Lemma 3.17** (Faben and Jerrum, Lemma 5.4)**.** *An involution-free tree has trivial automorphism group.*

**Theorem 3.18** (Faben and Jerrum, Theorem 5.6)**.** *Given an involution-free tree $H$ with more than one vertex, $\#_2\textsc{HomsTo}H$ is $\#_2\,\mathsf{P}$-complete.*

We have shown that for all primes $p$ and all asymmetric graphs $H$ with more than one vertex $\#_p\textsc{HomsTo}H$ is $\#_p\,\mathsf{P}$-hard. Recall Definition 2.7 of an order $p$ reduced form of a graph $H$. By Theorem 2.6, $H^{*p}$ is unique (up to isomorphism). By Theorem 2.4, $\#_p\textsc{HomsTo}H$ has the same complexity as $\#_p\textsc{HomsTo}H^{*p}$.

We can now prove the main theorem of this chapter.

**Theorem 1.10.** *Let $p$ be a prime and let $H$ be a graph whose order $p$ reduced form $H^{*p}$ is an asymmetric tree. If $H^{*p}$ has more than one vertex then $\#_p\textsc{HomsTo}H$ is $\#_p\,\mathsf{P}$-hard, otherwise $\#_p\textsc{HomsTo}H$ is computable in polynomial time.*

*Proof.* By Theorem 2.4, $\#_p\textsc{HomsTo}H$ and $\#_p\textsc{HomsTo}H^{*p}$ have the same computational complexity. If $H^{*p}$ has at most one vertex, then, by Corollary 2.9, $\#_p\textsc{HomsTo}H$ is computable in polynomial-time. Otherwise, $H^{*p}$ has more than one vertex. If $p = 2$ $\#_p\textsc{HomsTo}H$ is $\#_2\,\mathsf{P}$-complete by Theorem 3.18. Otherwise, $p > 2$ and $\#_p\textsc{HomsTo}H$ is $\#_p\,\mathsf{P}$-complete by Lemma 3.16. $\square$

## 3.6 Composite Numbers

In this section we will consider the complexity of $\#_k$HOMSTO$H$, when $k$ is a composite number and $H$ is a graph. We will use the Chinese remainder theorem. For a proof, see, e.g., [69, Theorem 2.6].

**Theorem 3.20** (Chinese remainder theorem). *Let $\{k_j\}_{j=1}^m$ be a pairwise relatively prime family of positive integers, and let $a_1, \ldots, a_m$ be arbitrary integers. Then there exists a solution $a \in \mathbb{N}$ to the system of congruences*

$$a \equiv a_j \pmod{k_j} \qquad (j = 1, \ldots, m).$$

*Moreover, any $a' \in \mathbb{N}$ is a solution to this system of congruences if and only if $a \equiv a'$ $\pmod{k}$, where $k = \prod_{j=1}^m k_j$.*

Let $H$ be a graph and consider $\#_k$HOMSTO$H$, where $k \in \mathbb{N}_{>0}$ is a composite. Let $\prod_{j=1}^m p_j^{r_j}$ be the prime factorisation of $k$. If $\#_k$HOMSTO$H$ can be solved in polynomial time, then for each $j \in [m]$, $\#_{p_j^{r_j}}$HOMSTO$H$ can also be solved in polynomial time: since $p_j^{r_j}$ is a factor of $k$ we take the solution of $\#_k$HOMSTO$H$ modulo $p_j^{r_j}$ and obtain a solution for $\#_{p_j^{r_j}}$HOMSTO$H$. From the Chinese remainder theorem, (Theorem 3.20) the inverse is also true: if for each $j \in [m]$ we can solve $\#_{p_j^{r_j}}$HOMSTO$H$ in polynomial time, then we can also solve $\#_k$HOMSTO$H$ in polynomial time.

Now let $H$ be a graph and let $k = p^r$, where $p$ is a prime and $r$ is a positive integer. If $\#_k$HOMSTO$H$ is computable in polynomial time then so is $\#_p$HOMSTO$H$, by simply taking the solution of $\#_k$HOMSTO$H$ modulo $p$. The inverse does not always hold. Surprisingly, Guo et al. [49] were able to obtain such a result for the constraint satisfaction problem. Recall the definition of $\#_k$CSP (Problem 3.4). They were able to show [49, Lemma 4.1 and Lemma 4.3] that, when $p$ is a prime, $\#_{p^r}$CSP is computable in polynomial time if $\#_p$CSP is computable in polynomial time.

Naturally, one would ask if a result similar to the result of Guo et al. can be obtained for $\#_k$HOMSTO$H$. We give a negative answer to this question: we show that there is a graph ($P_4$) such that $\#_2$HOMSTO$P_4$ is computable in polynomial time, while $\#_4$HOMSTO$P_4$ is $\#_2$ P-hard.

**Definition 3.21.** We write $P_4$ for the path $v_1v_2v_3v_4$.

That $\#_2$HOMSTO$P_4$ is computable in polynomial time comes trivially from Corollary 2.9 and the fact that $P_4$ has an involution.

**Corollary 3.22.** $\#_2$HOMSTO$P_4$ *is computable in polynomial time.*

*Proof.* $\tau = \{v_1 \mapsto v_4, v_4 \mapsto v_1, v_2 \mapsto v_3, v_3 \mapsto v_2\}$ is an involution for $P_4$ that has no fixed points, so $P_4^{*2}$ is the empty graph. Trivially, for any non-empty input graph $G$, $\#_2$HOMSTO$P_4^{*2}$ is always zero. □

In the rest of this section we will show that $\#_4$HomsToP$_4$ is hard via a series of polynomial time Turing reductions. Let $\mathcal{I}(G)$ be the set of independent sets of $G$. The starting point for our reductions is the following problem.

**Problem 3.23.** *Name.* $\#_k$BIS.

*Input.* A bipartite graph $G$.

*Output.* $|\mathcal{I}(G)| \pmod{k}$.

Faben [30, Theorem 3.7] shows the following:

**Theorem 3.24** (Faben)**.** *For all integers $k$, $\#_k$BIS is $\#_k\mathsf{P}$-complete.*

For our hardness proof to work, we will we need the input graph of $\#_k$BIS to be connected. So an intermediate stop in our chain of reductions is the connected version of $\#_k$BIS.

**Problem 3.25.** *Name.* $\#_k$conBIS.

*Input.* A bipartite, connected graph $G$.

*Output.* $|\mathcal{I}(G)| \pmod{k}$

The next lemma shows that $\#_k$conBIS is also hard for all integers.

**Lemma 3.26.** *For all integers $k$, $\#_k$conBIS is $\#_k\mathsf{P}$-complete.*

*Proof.* We give a Turing reduction from $\#_k$BIS and the lemma follows from Theorem 3.24. Let $G$ be an input for $\#_k$BIS and, since $G$ is bipartite, let $V_1, V_2$ be a bipartition of $G$, such that for each isolated vertex $v \in V(G)$, $v \in V_1$. We construct an instance $G'$ for $\#_k$conBIS by adding an extra vertex $v$ to a copy of $G$ and connecting $v$ with an edge to all the vertices in $V_1$. That is, $V(G') = V(G) \cup \{v\}$ and $E(G') = E(G) \cup \{(u,v) \mid u \in V_1\}$. This construction can be done in polynomial-time.

We claim that $|\mathcal{I}(G)| + 2^{|V_2|} = |\mathcal{I}(G')|$. Let $\mathcal{I}_0(G') = \{J \in \mathcal{I}(G') \mid v \in J\}$ and $\mathcal{I}_1(G') = \{J \in \mathcal{I}(G') \mid v \notin J\}$. $\mathcal{I}_0(G')$ and $\mathcal{I}_1(G')$ partition $\mathcal{I}(G')$. For every $J \in \mathcal{I}_0(G')$, it must be the case that $J \cap V_1 = \emptyset$, as every vertex in $V_1$ is adjacent to $v$ in $G'$. Any subset of $V_2$ can be an independent set in $\mathcal{I}_0(G')$, hence $|\mathcal{I}_0(G)| = 2^{|V_2|}$. To conclude the proof of the claim we will show that $|\mathcal{I}_1(G')| = |\mathcal{I}(G)|$. Since $v$ is not in any independent set in $|\mathcal{I}_1(G')|$, every independent set of $G$ is an independent set in $|\mathcal{I}_1(G')|$ and vice versa. The lemma follows. □

We are now ready to prove the main result of this section.

**Lemma 3.27.** *Let $P_4$ be the path $v_1v_2v_3v_4$. $\#_4$HomsToP$_4$ is $\#_2\mathsf{P}$-hard.*

*Proof.* We will show that $\#_2$conBIS reduces to $\#_4$HomsTo$P_4$. Let $G$ be an instance of $\#_2$conBIS and let $\mathcal{I}(G)$ be the set of independent sets of $G$. We will show that $2|\mathcal{I}(G)| = |\mathrm{Hom}(G \to P_4)|$.

Let $\sigma \in \mathrm{Hom}(G \to P_4)$. We first show that the set $J = \{u \in V(G) \mid \sigma(u) \in \{v_1, v_4\}\}$ is an independent set of $G$. Let $u_1, u_2 \in J$. From the definition of $J$, we have $(\sigma(u_1), \sigma(u_2)) \notin E(G)$. As $\sigma$ is a homomorphism, there can be no edge $(u_1, u_2) \in J$, so $J$ is an independent set of $G$.

Let $J \in \mathcal{I}(G)$ and let $V_1, V_2$ be a bipartition of $V(G)$. We define $\sigma_J : V(G) \to V(H)$ to be the following mapping:

$$\sigma_J(u) = \begin{cases} v_1, & \text{if } v \in V_1 \cap J \\ v_2, & \text{if } v \in V_2 \setminus J \\ v_3, & \text{if } v \in V_1 \setminus J \\ v_4, & \text{if } v \in V_2 \cap J. \end{cases}$$

Let $(u_1, u_2) \in E(G)$. We will show that $(\sigma_J(u_1), \sigma_J(u_2)) \in E(P_4)$. Since $G$ is bipartite and $(u_1, u_2) \in E(G)$, assume, without loss of generality, that $u_1 \in V_1$ and $u_2 \in V_2$. By the definition of $\sigma_J$ we have that $\sigma_J(u_1) \in \{v_1, v_3\}$ and $\sigma_J(u_2) \in \{v_2, v_4\}$. Assume for contradiction that $\sigma_J(u_1) = v_1$ and $\sigma_J(u_2) = v_4$. For the latter to hold, it must be that both $u_1, u_2 \in J$, but this is a contradiction since $J$ is an independent set and $(u_1, u_2) \in E(G)$. That is, $(\sigma_J(u_1), \sigma_J(u_2)) \in E(P_4)$, and therefore $\sigma_J \in \mathrm{Hom}(G \to P_4)$.

Let $\tau = \{v_1 \mapsto v_4, v_4 \mapsto v_1, v_2 \mapsto v_3, v_3 v_2\}$ be the involution of $P_4$ and consider the homomorphism $\sigma'_J = \sigma_J \circ \tau$. By definition:

$$\sigma'_J(u) = \begin{cases} v_1, & \text{if } v \in V_2 \cap J \\ v_2, & \text{if } v \in V_1 \setminus J \\ v_3, & \text{if } v \in V_2 \setminus J \\ v_4, & \text{if } v \in V_1 \cap J. \end{cases}$$

Since $G$ is connected, there is no subset $S \subseteq V(G)$, such that the mapping

$$\rho(u) = \begin{cases} \sigma_J(u), & \text{if } v \in S \\ \sigma'_J(u), & \text{otherwise} \end{cases}$$

is a valid homomorphism. Therefore, $\sigma_J$ and $\sigma'_J$ and are uniquely determined by $J$ and every homomorphism of $G$ is of the form $\sigma_J$ or $\sigma'_J$ for some $J \in \mathcal{I}(G)$. This proves the claim, that $2|\mathcal{I}(G)| = |\mathrm{Hom}(G \to P_4)|$, and establishes the lemma. □

# Chapter 4

# Counting homomorphisms (modulo 2) to cactus graphs

The results of this chapter are published in the paper "The Complexity of Counting Homomorphisms to Cactus Graphs Modulo 2" co-authored with Leslie Goldberg and David Richerby [43].

## 4.1 Introduction

In this chapter we study the complexity of $\#_2$HomsTo$H$ when $H$ is a cactus graph. A cactus graph is a connected graph in which every edge belongs to at most one cycle. The main result of this chapter is the following.

**Theorem 1.11.** *Let $H$ be a graph whose involution-free reduction $H^*$ is a cactus graph. If $H^*$ has at most one vertex then $\#_2$HomsTo$H$ is solvable in polynomial time; otherwise, $\#_2$HomsTo$H$ is $\#_2\mathsf{P}$-complete.*

If $H$ is a cactus graph, then so is every induced subgraph, including its involution-free reduction $H^*$. Thus, we have the following corollary, which proves the conjecture of Faben and Jerrum (Conjecture 1.7) for cactus graphs.

**Corollary 4.2.** *Let $H$ be a simple graph in which every edge belongs to at most one cycle. If the involution-free reduction of $H$ has at most one vertex then $\#_2$HomsTo$H$ is solvable in polynomial time. Otherwise, $\#_2$HomsTo$H$ is complete for $\#_2\mathsf{P}$ with respect to polynomial-time Turing reductions.*

In order to prove the hardness result in Theorem 1.11, we introduce three graph-theoretic notions: cactus gadgets, partial cactus gadgets, and mosaics. Cactus gadgets and partial cactus gadgets are structures for proving $\#_2\mathsf{P}$-hardness. Mosaics are graphs built on unions of 4-cycles. They are what is left in inductive cases where cactus gadgets don't exist and we use them inductively to prove overall hardness. Our approach is therefore recursive: we show how to decompose involution-free cactus graphs at cut vertices in such a way that every component contains at least one of these three induced structures. We then show how to combine these structures to obtain cactus gadgets in the original graph.

Unlike in our paper, where we directly proved that if an involution-free cactus graph $H$ contains a cactus gadget, then the $\#_2$HomsTo$H$ is $\#_2$ P-complete, in this thesis we take advantage of Theorem 3.10 of Section 3.4 to establish hardness. We show that, if a graph $H$ contains a cactus gadget, then $H$ contains a $(0, 2)$-gadget.

The most technical part of this chapter is showing that every non-trivial involution-free cactus graph does actually contain a cactus gadget. The presence of cycles in the graph greatly complicates the structure of the argument, hence the need to define cactus gadgets, partial cactus gadgets and mosaics and to decompose cactus graphs into components with these three different structures, which can then be combined to form cactus gadgets.

Theorem 1.11 gives a dichotomy for cactus graphs. If the involution-free reduction of $H$ has at most one vertex then $\#_2$HomsTo$H$ is in FP. Otherwise, it is $\#_2$ P-complete. Furthermore, the meta-problem of determining which is the case, given input $H$, is computationally easy. Finding an involution of $H$ reduces in polynomial time to computing the size of $H$'s automorphism group modulo 2 (see Faben [31, Chapter 7]). The latter problem is in FP for cactus graphs because, for example, cactus graphs are planar and have tree-width at most 2.

### 4.1.1 Organisation

We define cactus gadgets in Section 4.3 and also show that if $H$ has a cactus gadget then $H$ has a $(0, 2)$-gadget. In section 4.4 we define partial cactus gadgets. In Section 4.5 we define mosaics and show some of their key properties. In Section 4.6 we show how to combine partial cactus gadgets and mosaics in order to obtain cactus gadgets. We show that every involution-free cactus graph contains a cactus gadget in Section 4.7 and we combine everything together in Section 4.8 in order to obtain the main dichotomy theorem for cactus graphs.

## 4.2 Preliminaries

As in the previous chapter, graphs are undirected and have no parallel edges or self-loops. We will also use the partially labelled graphs and graphs with distinguished vertices that we introduced in Section 2.2.

For a graph $G$ and a vertex set $U \subseteq V(G)$, let $G[U]$ denote the subgraph of $G$ induced by $U$. Our analysis is based primarily on decomposing graphs, particularly by splitting them at cut vertices, as follows.

**Definition 4.3.** Given a graph $H$ with a cut vertex $v$, let $H'_1, \ldots, H'_\kappa$ be the connected components of $H - \{v\}$. The *split* of $H$ at $v$ is the set of graphs $\{H_1, \ldots, H_\kappa\}$, where $H_j = H[V(H'_j) \cup \{v\}]$.

Given two graphs $G$ and $H$ (not necessarily vertex-disjoint), let $G \cup H$ denote the graph $(V(G) \cup V(H), E(G) \cup E(H))$. If $F$ is a set of edges, let $V(F)$ denote the set of endpoints of edges in $F$ and let $G \cup F$ denote the graph $G \cup (V(F), F)$. Given sets $V' \subseteq V(G)$ and $E' \subseteq E(G)$, let $G - V' = G[V(G) \setminus V']$ and let $G - E' = (V(G), E(G) \setminus E')$. We use the phrase "$j$-walk" in a graph to refer to a walk of length $j$. Walks may repeat both vertices and edges.

We use $\Gamma_H(v)$ to denote the set of neighbours of vertex $v$ in $H$ and $\deg_H(v)$ to denote the degree of $v$ in $H$. We use $d_H(u,v)$ to denote the length of a shortest path from $u$ to $v$ in $H$. If $S \subseteq V(H)$ then $d_H(u,S)$ denotes $\min\{d_H(u,v) \mid v \in S\}$.

A path $P$ is a graph with vertex set $V(P)$ and edge set $E(P)$. However, where convenient, we specify $P$ by simply listing the vertices of the path, in order. As such, if $P = x_1 \ldots x_\ell$ and $P' = y_1 \ldots y_{\ell'}$, we write $Pz$ for the path $P \cup \{(x_\ell, z)\}$, $PP'$ for the path $P \cup \{(x_\ell, y_1)\} \cup P'$ and so on. We also use $\ell(P)$ to denote $|E(P)|$, the length of the path $P$. Similarly, we view a cycle $C$ as a graph, but we sometimes specify $C$ by listing its vertices in order. Paths and cycles do not repeat vertices; The length of a path $P$ is $\ell(P) = |E(P)|$. The length of a cycle $C$ is $\ell(C) = |E(C)|$.

Recall that an involution is an automorphism of order 2. Also recall Definition 2.7 of an order $p$ reduced form $H^{*p}$ of a graph $H$. In the special case of $p = 2$ we will call this graph the *involution-free reduction* of $H$ and simply denote it with $H^*$.

## 4.3 Hardness modulo 2 and cactus gadgets

Following the presentation of Chapter 3 we first show that if $H$ is orbit compatible with 2 and has the appropriate structure then $\#_2$HomsTo$H$ is $\#_2\mathsf{P}$-hard. Recall the definition of a $(\gamma, p)$-gadget (Definition 3.9). In this chapter we are interested in $p = 2$ so, according to Theorem 3.10, for $\#_2$HomsTo$H$ to be hard, $H$ must be orbit compatible with 2 and contain a $(0,2)$-gadget. It turns out that any involution free graph $H$ is orbit compatible with 2.

**Lemma 4.4.** *If $H$ is an involution-free graph, then $H$ and 2 are orbit compatible.*

*Proof.* Since $H$ is involution-free, by Corollary 2.13 every orbit of a tuple $\bar{y} \in (V(H))^r$ under the action of $\mathrm{Aut}(H)$ has nonzero cardinality modulo 2. That is $|\mathrm{Orb}_H(\bar{y})| \equiv 1 \pmod 2$ and the lemma follows. □

**Corollary 4.5.** *Let $H$ be an involution-free graph. If $H$ contains a $(0,2)$-gadget, then $\#_2$HomsTo$H$ is $\#_2\mathsf{P}$-complete.*

*Proof.* By Lemma 4.4, $H$ and 2 are orbit compatible. Since $H$ contains $(0,2)$-gadget and $0^2 \not\equiv 1 \pmod 2$, by Theorem 3.10 we have that $\#_2$HomsTo$H$ is $\#_2\mathsf{P}$-complete. □

To obtain hardness for $\#_2$HomsTo$H$ when $H$ is a cactus graph we will use what we call a cactus gadget.

**Definition 4.6.** A *cactus gadget* in a graph $H$ is a tuple $(\beta, s, t, O, i, K, k, w)$ where $\beta$ is a positive integer, $s$, $t$ and $i$ are vertices of $H$, $(O, \{i\}, K)$ is a partition of $\Gamma_H(s)$, and $k\colon K \to \mathbb{N}_{>0}$ and $w\colon K \to V(H)$ are functions. The following conditions must be satisfied.

1. $|O|$ is odd.
2. For any $o \in O$ and $y' \in O \cup \{i\}$, $s$ is the unique vertex that is adjacent to $o$ and $y$ and has an odd number of $\beta$-walks to $t$.
3. There are an even number of $(1+\beta)$-walks from $i$ to $t$.
4. For all $u \in K$, $w(u)$ has an even number of $k(u)$-walks to $u$ and an odd number of $k(u)$-walks to every vertex in $O \cup \{i\}$.

In our paper [43] cactus gadgets are called "hardness gadgets". In this thesis we have renamed them to cactus gadgets to distinguish between the other gadgets we use for proving that a problem is hard to compute.

To show that when an involution-free graph $H$ contains a cactus gadget, the problem of $\#_2$HomsTo$H$ is hard we would have to show an analogue of Corollary 4.5. In [43, Theorem 8.7] we proved that if an involution-free graph $H$ has a cactus gadget, then $\#_2$HomsTo$H$ is $\#_2\,\mathsf{P}$-hard directly. We don't need to do this here as it turns out that when a graph $H$ has a cactus gadget, then $H$ also has a $(0,2)$-gadget, so the hardness of $\#_2$HomsTo$H$ follows immediately.

The Properties 2-4 of a cactus gadget are stated in terms of number of walks of appropriate length between two vertices of $H$. The properties of a $(\gamma, p)$-gadget (Definition 3.9) are stated in terms of number of homomorphisms from a partially labelled graph $J$ with distinguished vertices to the target graph $H$. The following lemma will be helpful for converting the number of paths between two vertices in $H$ to the number of homomorphisms from $J$ to $H$.

**Lemma 4.7.** *Let $H$ be a graph, and let $v, w \in V(H)$. If $P$ is the path $x_1 \dots x_k x$ together with $\tau(P) = \{x_1 \mapsto w\}$, then $|\mathrm{Hom}((P,x) \to (H,v))|$ is equal to the number of $k$-walks from $w$ to $v$ in $H$.*

*Proof.* Let $W(a,b,k)$ denote the number of $k$-walks between the vertices $a$ and $b$. We prove the lemma by induction on $k$. For the base case $k = 1$, $P$ is the edge $(x_1, x)$. If $w$ is adjacent to $v$ in $H$, then $W(w,v,1) = 1 = |\mathrm{Hom}((P,x) \to (H,v))|$, otherwise $w$ and $v$ are not adjacent, so $W(w,v,1) = 0 = |\mathrm{Hom}((P,x) \to (H,v))|$.

Assuming that the lemma holds for $k$ we will show that the lemma holds for $k+1$. Every walk from $w$ to $v$ must go through a neighbour of $w$, so $W(w,v,k+1) =$

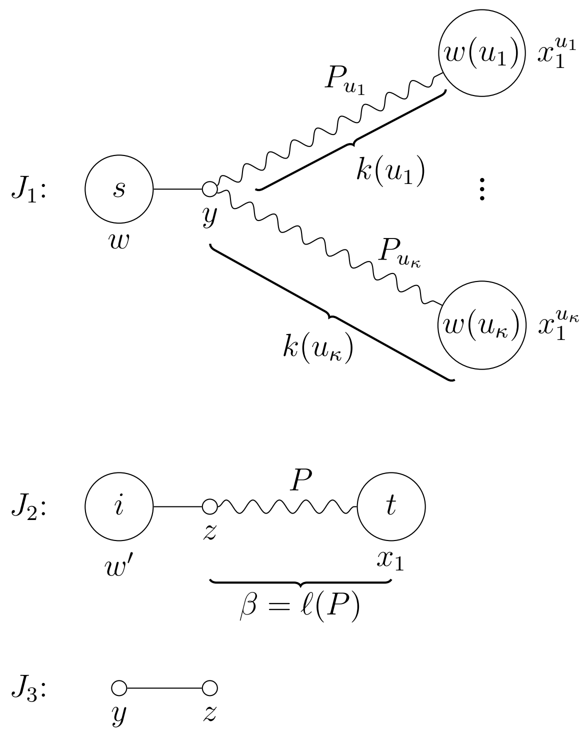

Figure 4.1: Using a cactus gadget $(\beta, s, t, O, i, K, k, w)$ in $H$ to find a $(0, 2)$-gadget in $H$, as in the proof of Lemma 4.8. Paths are shown with wavy lines, and their length is indicated with braces. Here we have assumed that $K = \bigcup_{j=1}^{\kappa}\{u_k\}$.

$\sum_{u \in \Gamma_H(w)} W(u, v, k)$. Let $P = x_1 \ldots x_{k+1}x$, with $\tau(P) = \{x_1 \mapsto w\}$ and let $P' = x_2 \ldots x_{k+1}x$. Every homomorphism from $P$ to $H$ that respects $\tau(P)$ must map $x_2$ to a neighbour of $w$, so $|\mathrm{Hom}((P, x) \to (H, v))| = \sum_{u \in \Gamma_H(w)} |\mathrm{Hom}((P', x_2, x) \to (H, u, v))|$. Recall that, given a partially labelled graph $J = (G, \tau)$ and vertices $x_1, \ldots, x_r \notin \mathrm{dom}(\tau)$, a homomorphism from $(G, x_1, \ldots, x_r)$ to $(H, y_1, \ldots, y_r)$ is formally identical to a homomorphism from $J = (G, \tau \cup \{x_1 \mapsto y_1, \ldots, x_r \mapsto y_r\})$ to $H$. By the latter and the induction hypothesis we have that for any $u \in \Gamma_H(w)$, $|\mathrm{Hom}((P', x_2, x) \to (H, u, v))| = W(u, v, k)$. So $|\mathrm{Hom}((P, x) \to (H, v))| = W(w, v, k + 1)$ and the lemma follows. $\square$

**Lemma 4.8.** *Let $H$ be a cactus graph. If $H$ has a cactus gadget then $H$ has a $(0, 2)$-gadget.*

*Proof.* Let $(\beta, s, t, O, i, K, k, w)$ be the cactus gadget for $H$. Recall Definition 3.9 of a $(\gamma, p)$-gadget. We define the $(0, 2)$-gadget of $H$ to be $(i, s, (J_1, y), (J_2, z), (J_3, y, z))$, where $J_1, J_2, J_3$ are defined as follows (see Figure 4.1):

- $J_1$ is the edge $(w, y)$ together with the set of paths $\{P_u = x_1^u \ldots x_{k(u)}^u y \mid u \in K\}$. $\tau(J_1) = \{w \mapsto s\} \cup \bigcup_{u \in K}\{x_1^u \mapsto w(u)\}$.
- $J_2$ is the edge $(w', z)$ together with the path $P = x_1 \ldots x_\beta z$. $\tau(J_2) = \{w' \mapsto i, x_1 \mapsto t\}$.

- $J_3$ is the edge $(y, z)$.

Recall the definition of $N_1(v)$ from Definition 3.9. We claim that $\Omega_y = O \cup \{i\}$ and that for every other vertex $v \in V(H)$, $N_1(v) \equiv 0 \pmod 2$. From the structure of $J_1$, for $v \in V(H)$,

$$N_1(v) = |\mathrm{Hom}(((w, y), y) \to (H, v))| \prod_{u \in K} |\mathrm{Hom}((P_u, y) \to (H, v))|.$$

For any $v \notin \Gamma_H(s)$, $|\mathrm{Hom}(((w, y), y) \to (H, v))| = 0$, hence $\Omega_y \subseteq \Gamma_H(s)$. By property 4 of Definition 4.6, for $y' \in O \cup \{i\}$ and for all $u \in K$, $w(u)$ has an odd number of $k(u)$-walks to $y'$. By Lemma 4.7 we have that for $y' \in O \cup \{i\}$ and for all $u \in K$, $|\mathrm{Hom}((P_u, y) \to (H, y'))| \equiv 1 \pmod 2$, which implies that for every $y' \in O \cup \{i\}$, $N_1(y') \equiv 1 \pmod 2$. Again, by property 4 of Definition 4.6, for all $u \in K$, $w(u)$ has an even number of $k(u)$-walks to $u$. Hence by Lemma 4.7 for every $u \in K$, $|\mathrm{Hom}((P_u, y) \to (H, u))| \equiv 0 \pmod 2$ and, consequentially, $N_1(u) \equiv 0 \pmod 2$. The claim that $\Omega_y = O \cup \{i\}$ follows and, trivially, for every $v \notin \Omega_y$, $N_1(v) \equiv 0 \pmod 2$. By property 1 of Definition 4.6 we have that $|O|$ is odd, so $|\Omega_y| \equiv 2 \pmod 2$. This establishes Property 1 of Definition 3.9.

Recall the definitions of $N_2(v)$ and $\Omega_z$ from Definition 3.9. From the structure of $J_2$, for $v \in V(H)$,

$$N_2(v) = |\mathrm{Hom}(((w', z), z) \to (H, v))| \cdot |\mathrm{Hom}((P, z) \to (H, v))|.$$

For any $v \notin \Gamma_H(i)$, $|\mathrm{Hom}(((w', z), z) \to (H, v))| = 0$, hence $\Omega_z \subseteq \Gamma_H(i)$. Let $W(a, b, \ell)$ denote the number of $\ell$ walks from $a$ to $b$. By property 3 of Definition 4.6 we have that $W(i, t, \beta + 1) \equiv 0 \pmod 2$. But $W(i, t, \beta + 1) = \sum_{v \in \Gamma_H(i)} W(v, t, \beta)$. By Lemma 4.7 we have $W(i, t, \beta+1) = \sum_{v \in \Gamma_H(i)} |\mathrm{Hom}((P, z) \to (H, v))|$, and since for each $v \in \Gamma_H(i)$, $|\mathrm{Hom}(((w', z), z) \to (H, v))| \equiv 1 \pmod 2$, we have that $W(i, t, \beta+1) \equiv \sum_{v \in \Gamma_H(i)} N_2(v) \pmod 2$. By the definition of $\Omega_z$ and by $\Omega_z \subseteq \Gamma_H(i)$ we have that $\sum_{v \in \Omega_z} N_2(v) \equiv W(i, t, \beta + 1) \equiv 0 \pmod 2$. By property 2 we have that $s$ has an odd number of $\beta$-walks to $t$, which implies that $N_2(s) \equiv 1 \pmod 2$. Hence property 2 of Definition 3.9 holds with $\kappa \equiv 1 \pmod 2$.

Recall the definition of $N_3(a, b)$ from Definition 3.9. By property 2 of Definition 4.6, $s$ is the unique common neighbour of any $o \in O$ and $i$ that has an odd number of $\beta$-walks to $t$. By Lemma 4.7, $s$ is the unique vertex in $\Gamma_H(o) \cap \Gamma_H(i)$ with $|\mathrm{Hom}((P, z) \to (H, s))| \equiv 1$. Thus, for any vertex $x \in \Omega_z \subseteq \Gamma_H(i)$ with $x \neq s$, since $N_2(x) \equiv |\mathrm{Hom}((P, z) \to (H, x))| \equiv 1 \pmod 2$, $x$ is not adjacent to $o$ in $H$. Therefore, $N_3(o, x) \equiv 0 \pmod 2$ and property 3 of Definition 3.9 holds.

To establish property 4 of Definition 3.9 recall that $J_3$ is an edge and $i, o \in \Gamma_H(s)$, so $N_3(i, s) = N_3(o, s) = 1$. Also $\Omega_z \subseteq \Gamma_H(i)$, so for any $x \in \Omega_z - s$, $N_3(i, x) = 1$. $\square$

**Corollary 4.9.** *Let $H$ be an involution-free cactus graph. If $H$ has a cactus gadget, then $\#_2$*HomsTo*$H$ is $\#_2\,\mathsf{P}$-complete.*

*Proof.* By Lemma 4.8 $H$ has a $(0, 2)$-gadget. The corollary follows from Corollary 4.5. □

## 4.4 Cactus gadgets and partial cactus gadgets

Our key technical result is that every non-trivial, involution-free cactus graph contains a cactus gadget (Theorem 4.28). The conditions of a cactus gadget simplify if $\beta = 1$, since having an odd number of 1-walks to a vertex is the same as being adjacent to it. In cases where $\beta = 1$, we will use this simplified condition without comment.

Our analysis is based primarily on decompositions of graphs into their subgraphs, so we need conditions under which a cactus gadget in an induced subgraph of $H$ is a cactus gadget in $H$. The idea here is that, if a cactus gadget satisfies the distance requirements for a vertex $v$, the structure of the graph "beyond" $v$ cannot interfere with the gadget's paths.

**Definition 4.10.** Consider a cactus gadget $(\beta, s, t, O, i, K, k, w)$ in $H$ and a vertex $v \in V(H)$. The *primary distance requirement* of the gadget with respect to $v$ is

$$d_H(v, O \cup \{i\}) + d_H(v, t) > \beta - 1.$$

The *secondary distance requirement* of the gadget with respect to $v$ is that, for each $u \in K$,

$$d_H(v, w(u)) + d_H(v, O \cup \{i, u\}) > k(u) - 2.$$

Suppose that $H_1, \dots, H_\kappa$ is a split of the graph $H$ at some cut vertex $v$. If there is a cactus gadget $\Gamma$ in $H_1$ that satisfies the distance restrictions for $v$, it is easy to see that it also satisfies the distance restrictions for all $x \in V' = V(H_2) \cup \cdots \cup V(H_\kappa)$, since any path from $H_1$ to $V'$ must go through $v$. This ensures that $\Gamma$ is also a cactus gadget in $H$, since the number of walks of various lengths required by the definition of the cactus gadget cannot be affected by vertices beyond $v$.

In some cases, our decomposition might yield subgraphs that do not contain cactus gadgets. We are still able to make progress using structures that can be combined with other parts of the graph to produce a cactus gadget. A partial cactus gadget is, essentially, a simplified cactus gadget that has $K = \emptyset$ and that doesn't yet have a "$t$" vertex: at a later point, we will find a vertex $t$ with the properties necessary to produce a full cactus gadget.

**Definition 4.11.** A *partial cactus gadget* in a graph with distinguished vertices $(H, x)$ is a tuple $(s, i, O, P)$ where $s$ is a vertex of $H$, $(\{i\}, O)$ is a partition of $\Gamma_H(s)$, and $P$ is a path in $H$ satisfying the following conditions.

- $|O|$ is odd.
- $P$ is the unique shortest path from $x$ to $i$ in $H$.
- $Ps$ is the unique shortest path from $x$ to $s$ in $H$.
- For each $o \in O$, $Pso$ is the unique shortest path from $x$ to $o$ in $H$.

## 4.5 Mosaics

The final structure that can arise from our decompositions is a subgraph made entirely from 4-cycles and from edges between vertices of those cycles and additional vertices of degree 1. Some mosaics (the "shortcut mosaics" defined below) already contain cactus gadgets. In the other cases, we identify structures called "2,3-paths" in mosaics and these will provide a "$t$" vertex for a partial cactus gadget elsewhere in the decomposed graph.

**Definition 4.12.** A *mosaic* is a graph with distinguished vertices defined as follows.

- An unbristled mosaic is the one-vertex graph with distinguished vertices or a cactus graph with a distinguished vertex that is a union of 4-cycles.
- A *mosaic* is a graph with distinguished vertices $(H, x)$ for which there is a partition $(V', V'')$ of $V(H)$ such that $x$ is in $V'$, $(H[V'], x)$ is an unbristled mosaic, and $E(H) - E(H[V'])$ is a perfect matching between $V''$ and a subset of $V'$. The edges of the matching are called *bristles*.
- A *proper mosaic* is a mosaic that contains at least one cycle.

Note that every vertex of a mosaic is adjacent to at most one bristle. Note also that the one-vertex graph with distinguished vertices and an edge with a distinguished vertex are both mosaics (but not proper mosaics).

**Definition 4.13.** A *2,3-path* in a graph with distinguished vertices $(H, x)$ is a tuple $(P, v_2, v_3)$ such that $v_2$ and $v_3$ are in the same cycle of $H$ and that, for $j \in \{2, 3\}$, $\deg_H(v_j) = j$ and $Pv_j$ is the unique shortest $x$–$v_j$ path in $H$.

**Lemma 4.14.** *Every involution-free proper mosaic $(H, x)$ contains a 2,3-path.*

*Proof.* See Figure 4.2. Write $x_1 = x$ and let $P = x_1 \dots x_\ell$ be a longest path from $x_1$ in $H$ that uses only edges from cycles and uses at most one edge from each cycle. $P$ contains at least one edge because $x$ is on a cycle. Let $C = x_{\ell-1}x_\ell yzx_{\ell-1}$ be the cycle containing $x_{\ell-1}$ and $x_\ell$.

$x_\ell$ is on a cycle, so $\deg(x_\ell) \geq 2$. Also, $\deg(x_\ell) \leq 3$ since, otherwise, $x_\ell$ would have a neighbour $x_{\ell+1}$ on a cycle other than $C$ and the path $Px_{\ell+1}$ would contradict the choice

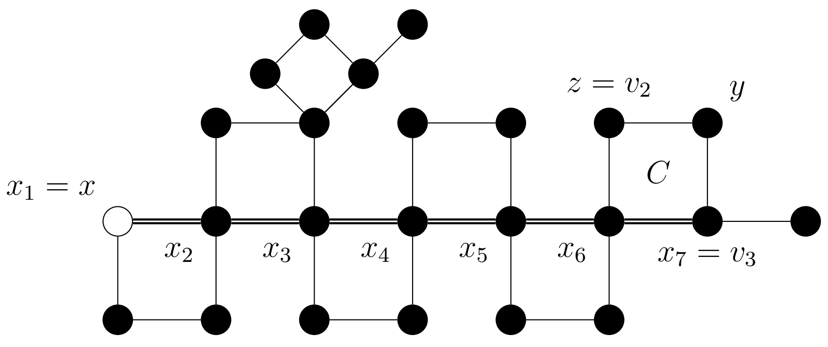

Figure 4.2: A mosaic with the 2,3-path $(P, v_2, v_3)$ that can be found using Lemma 4.14. $P$ is drawn with a double line.

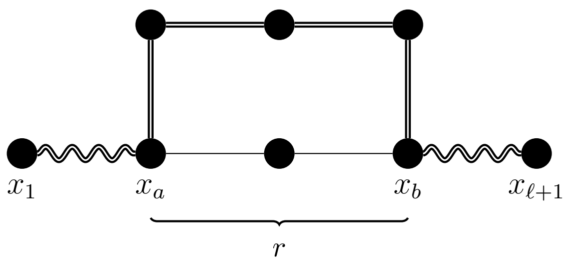

Figure 4.3: An example of a walk in $T_1(P)$, shown with double lines.

of $P$. By the same argument, $2 \leq \deg(z) \leq 3$. Furthermore, $\deg(x_\ell) \neq \deg(z)$ or $H$ would have an involution exchanging these two vertices. Note that $P' = x_1 \ldots x_{\ell-1}z$ is the unique shortest $x$–$z$ path in $H$, since any other $x$–$z$ path must include edges from exactly the same cycles as $P'$ and must include at least two edges from one of them. Thus, either $(x_1 \ldots x_{\ell-1}, x_\ell, z)$ or $(x_1 \ldots x_{\ell-1}, z, x_\ell)$ is a 2,3-path. □

In several cases, we have a unique shortest path of length $\ell$ between two vertices in a graph and we are interested in the number of $(\ell + 2)$ walks between those two vertices. The following definition helps us count such walks.

**Definition 4.15.** Let $P = x_1 \ldots x_{\ell+1}$ be a path in a graph $H$. We define the following three sets of $(\ell + 2)$-walks from $x_1$ to $x_{\ell+1}$.

1. $T_1(P)$ is the set of walks that differ from $P$ by going the long way around a $(2r + 2)$-cycle from which $P$ uses $r$ consecutive edges, as shown in Figure 4.3. Formally, these are the walks $x_1 \ldots x_a P' x_b \ldots x_{\ell+1}$, where $1 \leq a < b \leq \ell + 1$ and $x_a P' x_b$ is a $(b - a + 2)$-path in $H - \{x_{a+1}, \ldots, x_{b-1}\}$.

2. $T_2(P)$ is the set of walks that differ from $P$ by going the long way around cycles of length $2r + 1$ and $2r' + 1$, from which $P$ uses $r$ and $r'$ consecutive edges, respectively, as shown in Figure 4.4. Formally, these are the walks of the form $x_1 \ldots x_a P' x_b \ldots x_{a'} P'' x_{b'} \ldots x_{\ell+1}$ where $1 \leq a < b \leq a' < b' \leq \ell + 1$, $x_a P' x_b$ is a $(b - a + 1)$-path in $H - \{x_{a+1}, \ldots, x_{b-1}\}$ and $x_{a'} P'' x_{b'}$ is a $(b' - a' + 1)$-path in $H - \{x_{a'+1}, \ldots, x_{b'-1}\}$.

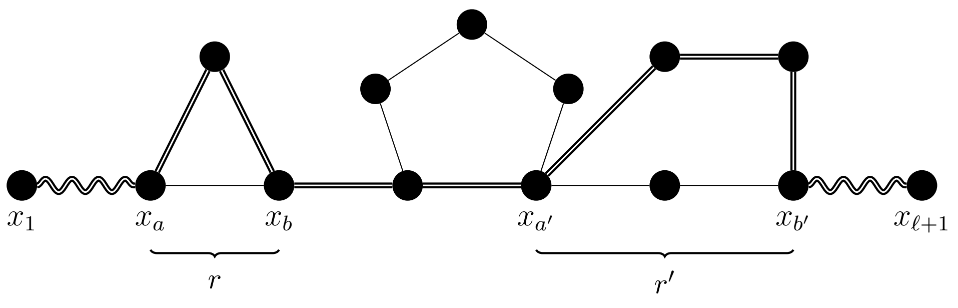

Figure 4.4: An example of a walk in $T_2(P)$, shown with double lines. The wavy lines represent an ommited portion of the graph whose structure is unimportant, except that it contains a unique shortest path between the endpoints of each wavy line.

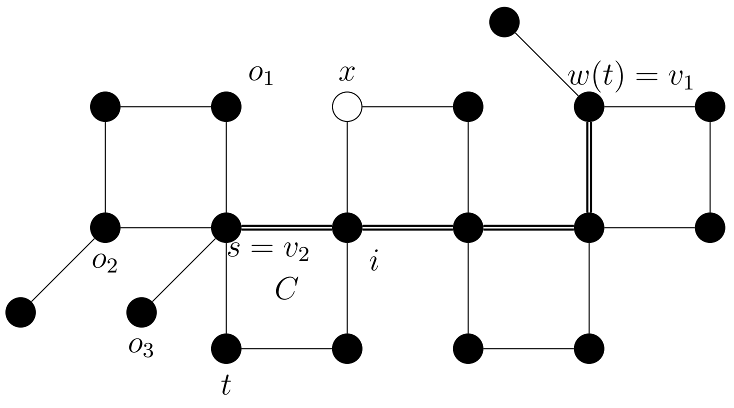

Figure 4.5: A mosaic containing containing a cactus gadget as in Lemma 4.16. The path $P$ is shown with a double line and $O = \{o_1, o_2, o_3\}$.

3. $T_3(P)$ is the set of walks $x_1 \dots x_a z x_a \dots x_{\ell+1}$, where $1 \le a \le \{\ell + 1\}$ and $z \in \Gamma_H(x_a)$ (we allow the case $z = x_{a\pm1}$).

We refer to the cycles appearing in the definition of $T_1$ and $T_2$ as *detour cycles.*

It is easy to see that, when $P$ is the unique shortest $x_1$–$x_{\ell+1}$ path in a cactus graph, $T_1(P)$, $T_2(P)$ and $T_3(P)$ is a partition of the set of all $(\ell(P)+2)$-walks from $x_1$ to $x_{\ell+1}$.

**Lemma 4.16.** *Let $H$ be a cactus graph containing distinct odd-degree vertices $v_1$ and $v_2$, with a unique shortest path $P$ between them. Suppose that every edge of $P$ is on a 4-cycle of $H$ and no 4-cycle of $H$ contains two edges of $P$. Then $H$ contains a cactus gadget that satisfies the distance requirements for every $v \in (V(H) \setminus V(P)) \cup \{v_2\}$.*

Note that the premise of the lemma is symmetric about $v_1$ and $v_2$, while the conclusion is not. By symmetry, we could, of course, find a different cactus gadget satisfying the distance requirements for $v_1$ instead of $v_2$.

*Proof.* See Figure 4.5. Choose $v_1$, $v_2$ and $P$ satisfying the given conditions so that $\ell(P)$ is as small as possible. Note that $v_1$ and $v_2$ have degree at least 3, since they have odd degree and are on cycles.

Let $\beta = 1$. Let $s = v_2$, let $i$ be the neighbour of $s$ in $P$ and let $t$ be the neighbour of $s$ that is in the same 4-cycle as $i$; call this cycle $C$. Let $K = \{t\}$, and $O = \Gamma_H(s)\backslash(\{i\} \cup K)$. Let $w(t) = v_1$ and $k(t) = \ell(P) + 1$.

To see that $\Gamma = (\beta, s, t, O, i, K, k, w)$ is a cactus gadget, note first that $|O|$ is odd, since $\deg_H(s)$ is odd. Consider any $o \in O$ and $y' \in O \cup \{i\}$. Since $H$ is a cactus graph, $s$ is the unique vertex adjacent to $o$, $y'$ and $t$. However, there are two vertices that are adjacent to $i$ and $t$: $s$ and the fourth vertex on cycle $C$. Therefore, there are two $(\beta+1)$-walks from $i$ to $t$. Finally, consider vertex $w(t) = v_1$. Vertex $v_1$ has two $(\ell(P)+1)$-walks to $t$ (going around the cycle $C$ in either direction). For every vertex $o \in O$, $Po$ is the unique $(\ell(P)+1)$-walk from $v_1$ to $o$.

We will finish the proof that $\Gamma$ is a cactus gadget by showing that $w(t) = v_1$ has an odd number of $(\ell(P)+1)$-walks to $i$. Let $P_i$ be the length-$(\ell(P)-1)$ prefix of $P$. $P_i$ is the unique shortest path from $v_1$ to $i$. Consider Definition 4.15, with $x_1 = v_1$, $\ell = \ell(P_i)$ and $x_{\ell+1} = i$. Since $P_i$ only uses one edge from each cycle it meets and each such cycle is a 4-cycle, $T_2(P_i) = \emptyset$. There are $\ell(P_i)$ walks in $T_1(P_i)$, since every edge of $P$ (hence, every edge of $P_i$) is on a distinct 4-cycle of $H$. The number of walks in $T_3(P_i)$ is $\sum_{v\in P_i} \deg_H(v) - \ell(P_i)$ since every edge adjacent to $P_i$ may be repeated, but edges in $P_i$ should not be counted twice. The total number of walks is therefore $\sum_{v\in P_i} \deg_H(v)$. This is odd since $\deg_H(v_1)$ is odd, and every vertex in $P_i$ other than $v_1$ has even degree (otherwise the minimality of $\ell(P)$ would be contradicted).

The cactus gadget $\Gamma$ satisfies the primary distance requirement $d_H(v, O \cup \{i\}) + d_H(v, t) > 0$ for any $v \in V(H)$ since $t \notin O \cup \{i\}$, so at least one of the terms $d_H(v, O \cup \{i\})$ and $d_H(v, t)$ is positive. Now consider any $v \in V(H) \setminus V(P_i)$. We wish to show that the secondary distance requirement $d_H(v, v_1) + d_H(v, O \cup \{i, t\}) > \ell(P) - 1$ is satisfied. We do this by establishing the following inequalities:

$$d_H(v, v_1) + d_H(v, i) > \ell(P) - 1 \tag{4.1}$$

$$d_H(v, v_1) + d_H(v, O \cup \{t\}) > \ell(P) - 1. \tag{4.2}$$

Establishing (4.1) is easy. Since $v \notin P_i$ and $P_i$ is the unique shortest path from $v_1$ to $i$, $d_H(v, v_1) + d_H(v, i) > \ell(P_i) = \ell(P) - 1$. Establishing (4.2) is similar. For each $y' \in O \cup \{t\}$, $d_H(v_1, y') = \ell(P) + 1$. Therefore, for any $v \in V(H) \setminus V(P_i)$, $d_H(v, v_1) + d_H(v, y') \geq \ell(P) + 1$. □

**Definition 4.17.** A *shortcut* in a mosaic $(H, x)$ is a pair of odd-degree vertices $v_1, v_2$, with degree at least 3, that have a unique shortest path $P$ between them, and this path does not contain $x$. A *shortcut mosaic* is a mosaic that contains a shortcut.

If $v_1, v_2$ is a shortcut in a mosaic $(H, x)$ then $v_1$ and $v_2$ are on cycles (since their degrees are at least 3), so every edge of the unique shortest path $P$ between them is on a 4-cycle. Since $P$ is unique, these edges are on distinct 4-cycles. Thus, Lemma 4.16 has the following corollary.

**Corollary 4.18.** *If $(H, x)$ is a shortcut mosaic then $H$ contains a cactus gadget that satisfies the distance requirements for $x$.*

## 4.6 Combination lemmas

We mostly proceed by splitting graphs at cut vertices and investigating the resulting components. In this section, we present a number of technical lemmas that show how to combine structures in the various parts of a graph split to obtain cactus gadgets.

**Observation 4.19.** If $\{H_1, \ldots, H_\kappa\}$ is the split of an involution-free graph $H$ at a cut vertex $v$ then, for each $j \in [\kappa]$, the graph with distinguished vertex $(H_j, v)$ is involution-free even though $H_j$ itself might not be involution-free. To see that $(H_j, v)$ is involution-free, note that an involution of $H_j$ that fixes $v$ induces an involution of $H$.

**Lemma 4.20.** *Let $x$ be a cut vertex of an involution-free cactus graph $H$. If there exists a split of $H$ at $x$ into $\{H_1, \ldots, H_\kappa\}$ such that $(H_1, x)$ and $(H_2, x)$ are both proper mosaics then $H$ has a cactus gadget which satisfies the distance requirements for every vertex $v \in V(H) \setminus (V(H_1) \cup V(H_2))$.*

*Proof.* If, for $j = 1$ or $j = 2$, $(H_j, x)$ is a shortcut mosaic then, by Corollary 4.18, $H_j$ contains a cactus gadget $\Gamma$ that satisfies the distance requirements for $x$. Since $x$ is a cut vertex, $\Gamma$ is a cactus gadget in $H$ and satisfies the distance restrictions for every vertex outside $H_1$ and $H_2$.

Suppose that neither of $(H_1, x)$ and $(H_2, x)$ is a shortcut mosaic. For $j \in \{1, 2\}$, apply Lemma 4.14 to $(H_j, x)$ to obtain a 2,3-path $(P_j, y_j, z_j)$. Since $z_j \neq x$ and $x$ is a cut vertex, $\deg_H(z_j) = \deg_{H_j}(z_j) = 3$. $z_1 P_1 P_2 z_2$ is the unique shortest $z_1$–$z_2$ path in $H$ and each edge of this path is in a different 4-cycle. By Lemma 4.16, $H$ contains a cactus gadget which satisfies the distance requirements for every $v \in (V(H) \setminus V(P)) \cup \{z_2\}$. □

**Lemma 4.21.** *Let $H$ be a cactus graph with a cut vertex $x$ of odd degree. If there exists a split of $H$ at $x$ into $\{H_1, \ldots, H_\kappa\}$ such that $(H_1, x)$ is an involution-free proper mosaic, then $H$ has a cactus gadget that satisfies the distance requirements for every $v \in V(H) \setminus V(H_1)$.*

*Proof.* If $(H_1, x)$ is a shortcut mosaic then Corollary 4.18 gives a cactus gadget in $H_1$. As in the previous lemma, this is a cactus gadget in $H$ and satisfies the distance requirements. If $(H_1, x)$ is not a shortcut mosaic, apply Lemma 4.14 to $(H_1, x)$ to obtain a 2,3-path $(P, v_2, v_3)$. Since $v_3 \neq x$ and $x$ is a cut vertex, $\deg_H(v_3) = \deg_{H_1}(v_3) = 3$ and $P$ is the unique shortest path in $H$ between $v_3$ and $x$. Since $(H_1, x)$ is a proper mosaic, every edge of $P$ is on a distinct 4-cycle of $H$. By Lemma 4.16, $H$ contains a cactus gadget that satisfies the distance requirements for every $v \in (V(H) \setminus V(P)) \cup \{x\}$, which proves the lemma. □

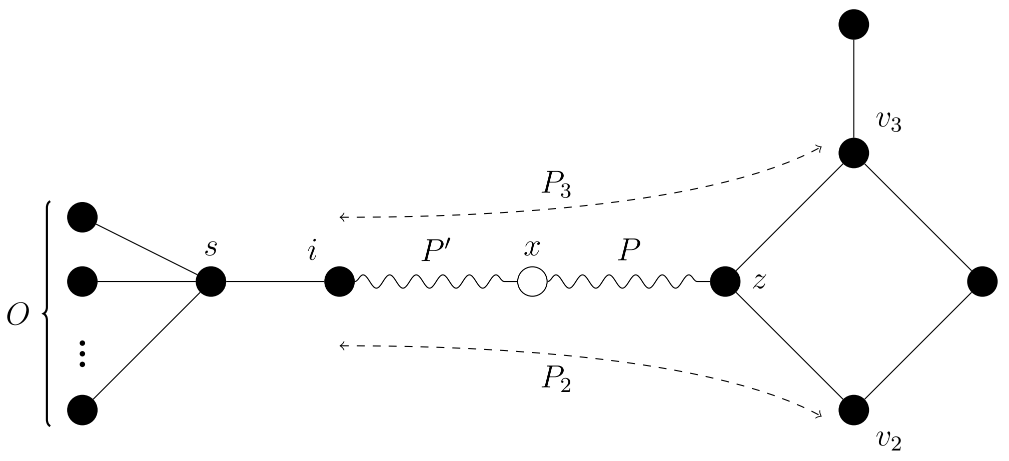

Figure 4.6: The paths appearing in the proof of Lemma 4.22.The wavy lines represent an ommited portion of the graph whose structure is unimportant, except that it contains a unique shortest path between the endpoints of each wavy line.

**Lemma 4.22.** *Let $H$ be a cactus graph. Suppose that $\{H_1, \dots, H_\kappa\}$ is the split of $H$ at $x$, that $(H_1, x)$ contains a 2,3-path $(P, v_2, v_3)$ and that $(H_2, x)$ contains a partial cactus gadget $(s, i, O, P')$. Then $H$ has a cactus gadget that satisfies the distance requirements for very vertex $v \in V(H) \setminus (V(PP') \cup \{v_2, v_3\})$.*

*Proof.* See Figure 4.6. Let $z$ be the endpoint of the path $P$ that is not $x$, or let $z = x$ if $\ell(P) = 0$. Since $H$ is a cactus graph and $v_2$ and $v_3$ are on a cycle together, the cycle must contain the edges $(z, v_2)$ and $(z, v_3)$. Since $v_2 \neq x$ and $v_3 \neq x$ and $x$ is a cut vertex, $\deg_H(v_2) = 2$ and $\deg_H(v_3) = 3$. It is easy to see that $(P'P, v_2, v_3)$ is a 2,3-path in $H$. For $j \in \{2, 3\}$, let $P_j = P'Pv_j$.

We next show that the number of $(\ell(P_2) + 2)$-walks from $i$ to $v_2$ differs in parity from the number of $(\ell(P_2) + 2)$-walks from $i$ to $v_3$. There are two kinds of walks to consider — those that detour around cycles, and those that repeat an edge. Using the notation from Definition 4.15, walks that detour around cycles are in $T_1$ or $T_2$ and those that repeat an edge are in $T_3$. Since $(z, v_2)$ and $(z, v_3)$ are not on any cycles other than the one that contains them both, all of these walks pass through $z$, progressing on to $v_2$ or to $v_3$. Furthermore, the number of $(\ell(P_2)+2)$-walks that detour around cycles is the same for both endpoints, $v_2$ and $v_3$. Now note that the number of $(\ell(P_2) + 2)$-walks in $T_3(P_3)$ is exactly one more than the number in $T_3(P_2)$, because $\deg_H(v_3) = \deg_H(v_2) + 1$.

We can now define the cactus gadget. Let $\beta = \ell(P_2) + 1$. $s$, $O$ and $i$ are already defined by the partial cactus gadget; let $K = \emptyset$. Choose $t \in \{v_2, v_3\}$ so that the number of $(1 + \beta)$-walks from $i$ to $t$ is even.

To see that this is a cactus gadget, consider $o \in O$ and $y' \in O \cup \{i\}$. $s$ is the only vertex that is both adjacent to $o$ and $y'$ and has an odd number of $(\ell(P_2)+1)$-walks to $t$ (otherwise, there would be more than one shortest path from $o$ to $t$). By construction,

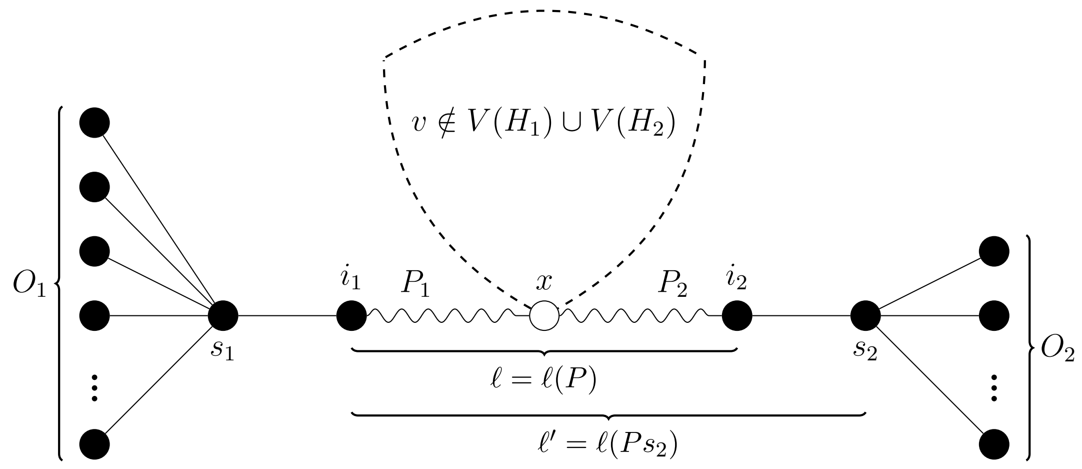

Figure 4.7: The two partial cactus gadgets of Lemma 4.24.The wavy lines represent an ommited portion of the graph whose structure is unimportant, except that it contains a unique shortest path between the endpoints of each wavy line.

there are an even number of $(1+\beta)$-walks from $i$ to $t$. $K = \emptyset$ so the requirements on $K$ are vacuous.

Now consider the primary distance requirement $d_H(v, O \cup \{i\}) + d_H(v,t) > \beta - 1 = \ell(P_2)$. The unique shortest path in $H$ from $t$ to $O \cup \{i\}$ is either $P_2$ or $P_3$ and this path has length $\ell(P_2)$ so the requirement is satisfied for any vertex $v$ that is not on $P_2$ or $P_3$, which is to say, any vertex of $V(H) \setminus (V(P) \cup V(P') \cup \{v_2, v_3\})$, as required. There are no secondary distance requirements, since $K = \emptyset$, so the lemma is proved. □

**Corollary 4.23.** *Let $H$ be an involution-free cactus graph and let $x$ be a cut vertex of $H$. If there exists a split of $H$ at $x$ into $\{H_1, \ldots, H_\kappa\}$ such that $(H_1, x)$ is a proper mosaic and $(H_2, x)$ contains a partial cactus gadget $(s, i, O, P)$, then $H$ has a cactus gadget that satisfies the distance requirements for very vertex $v \in V(H) \setminus V(H_1 \cup P)$.*

*Proof.* By Lemma 4.14, $(H_1, x)$ contains a 2,3-path; Lemma 4.22 gives the cactus gadget. □

**Lemma 4.24.** *Let $x$ be a cut vertex of an involution-free cactus graph $H$. If there is a split of $H$ at $x$ into $\{H_1, \ldots, H_\kappa\}$, such that $(H_1, x)$ contains a partial cactus gadget $(s_1, i_1, O_1, P_1)$ and $(H_2, x)$ contains a partial cactus gadget $(s_2, i_2, O_2, P_2)$, then $H$ contains a cactus gadget that satisfies the distance requirements for every vertex $v \in V(H) \setminus V(P_1P_2s_2)$.*

*Proof.* See Figure 4.7. The two partial cactus gadgets ensure that $P = P_1P_2$ is the unique shortest path in $H$ from $i_1$ to $i_2$ and $P_1P_2s_2$ is the unique shortest path from $i_1$ to $s_2$. Let $\ell = \ell(P)$ and $\ell' = \ell(Ps_2) = \ell + 1$.

We first show that the number of $(\ell+2)$-walks from $i_1$ to $i_2$ (walks which use two more edges than $P$) differs in parity from the number of $(\ell'+2)$-walks from $i_1$ to $s_2$. To do this, we use the sets of walks $T_1$, $T_2$ and $T_3$ from Definition 4.15.

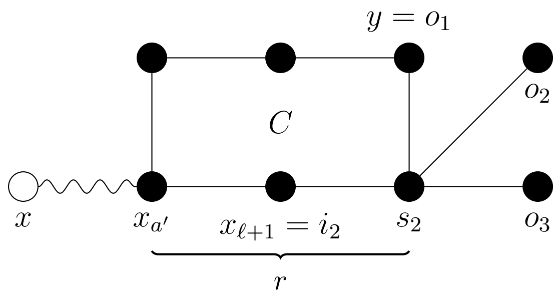

Figure 4.8: An example of an impossible detour cycle $C$ in $T_1(Ps_2)$, using neighbours of $s_2$. Here $O_2 = \{o_1, o_2, o_3\}$. The wavy lines represent an ommited portion of the graph whose structure is unimportant, except that it contains a unique shortest path between the endpoints of each wavy line.

First, we show that $T_1(P) = T_1(Ps_2)$, by arguing that every detour cycle $C$ that is available from $Ps_2$ is also available from $P$. Consider a walk $x_1 \dots x_a P' x_b \dots x_{\ell'} s_2$ as in the definition of $T_1(Ps_2)$, where $x_1 = i_1$ and $x_{\ell'} = i_2$. The claim is obvious if $x_b \in P$ so suppose for contradiction that $x_b = s_2$ and the detour cycle is $x_a \dots x_{\ell'} s_2 P' x_a$. Then, by the definition of partial cactus gadgets, the neighbour $y$ of $s_2$ in $P'$ is in $O_2$ (see Figure 4.8). However, the definition also requires that every vertex in $O_2$ has a unique shortest path to $x$ but the detour cycle gives two paths of length $\ell(P_2) + 2$ from $y$ to $x$.

A similar argument shows that $T_2(P) = T_2(Ps_2)$. This is obvious when $x_{b'} \in P$. If $x_{b'} = s_2$, the same construction gives a contradiction: the path from $y$ to $x$ via $P''$ is shorter than the one via $s_2$, which the definition requires to be the unique shortest $y$–$x$ path.

Finally, we show that $T_3(P) \not\equiv T_3(Ps_2) \mod 2$. This is because $s_2$ offers an additional set of edges that may be repeated, $\{(s_2, o) \mid o \in O_2\}$, and there are odd number of edges in this set since $|O_2|$ is odd.

We now construct the cactus gadget. If the number of $(\ell + 2)$-walks from $i_1$ to $i_2$ is even, then let $t = i_2$ and $\beta = \ell + 1$. Otherwise, let $t = s_2$ and $\beta = \ell' + 1$. In either case, the number of $(1 + \beta)$-walks from $i_1$ to $t$ is even. Then let $s = s_1$, $O = O_1$, $i = i_1$ and $K = \emptyset$.

To see that this is a cactus gadget, consider $o \in O$ and $y' \in O \cup \{i\}$. $s$ is the unique vertex adjacent to $o$ and to $y$ which has an odd number of $\beta$-walks to $t$. (The odd number is one.) The construction guarantees an even number of $(1 + \beta)$-walks from $i_1$ to $t$.

Now consider the primary distance requirement $d_H(v, O \cup \{i\}) + d_H(v, t) > \beta - 1$. The unique shortest path in $H$ from $t$ to $O \cup \{i\}$ has length $\beta - 1$ so the requirement is satisfied for any vertex $v$ that is not on this path. This is guaranteed by the restriction on $v$ in the statement of the lemma. There are no secondary distance requirements, since $K = \emptyset$. □

**Lemma 4.25.** *Let $(H, x)$ be a cactus graph with a distinguished vertex containing a*

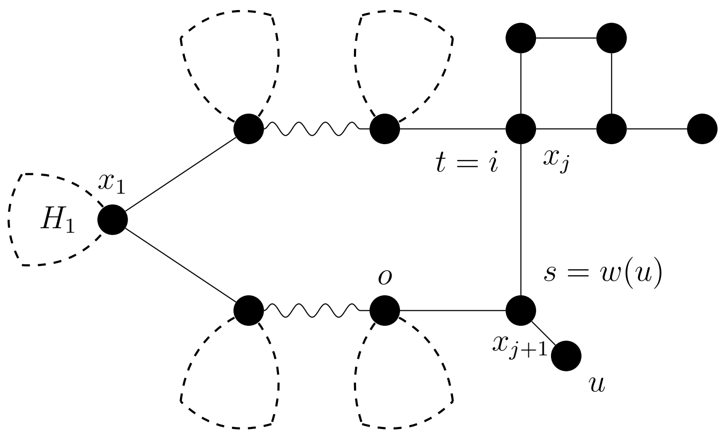

Figure 4.9: The cactus gadget of Case 1.1 of Lemma 4.25. In this case, $K = \{u\}$.The wavy lines represent an ommited portion of the graph whose structure is unimportant, except that it contains a unique shortest path between the endpoints of each wavy line.

*cycle $C = x_1x_2 \dots x_\ell x_1$ where $\ell \neq 4$. For $j \in [\ell]$, let $H_j$ be the component containing $x_j$ in the graph $H - E(C)$. Suppose that the graph with distinguished vertices $(H[(V(H) \setminus V(H_1)) \cup \{x_1\}], x_1)$ is involution-free. If $\ell$ is even, let $\mathcal{J} = [\ell] \setminus \{1, \ell/2+1\}$. Otherwise, let $\mathcal{J} = [\ell] \setminus \{1\}$. If, for each $j \in \mathcal{J}$, $(H_j, x_j)$ is a mosaic, then $H$ contains a cactus gadget that satisfies the distance requirements for each $v \in V(H_1)$.*

*Proof.* Note that for each $j \in \mathcal{J}$, $(H_j, x_j)$ is involution-free. We start by dispensing with some easy cases. First, if there is a $j \in \mathcal{J}$ such that $(H_j, x_j)$ is a shortcut mosaic then, by Corollary 4.18, $H_j$ contains a cactus gadget which satisfies the distance requirements for $x_j$ and this is also a cactus gadget in $H$ that satisfies the distance requirements for all $v \in V(H) \setminus V(H_j)$, in particular for all $v \in V(H_1)$. Second, suppose that there is a $j \in \mathcal{J}$ such that $(H_j, x_j)$ is a proper mosaic and $\deg_H(x_j)$ is odd. $x_j$ is a cut vertex of $H$ since $H$ is a cactus graph. Therefore, Lemma 4.21 guarantees that $H$ has a cactus gadget which satisfies the distance requirements for every $v \in V(H) \setminus V(H_j)$.

Thus, we can assume with loss of generality that for every $j \in \mathcal{J}$, $(H_j, x_j)$ is a shortcut-free mosaic. The two possibilities are:

- $(H_j, x_j)$ is a (possibly trivial) shortcut-free mosaic and $\deg_H(x_j)$ is even, or
- $(H_j, x_j)$ consists of a single bristle.

Since $(H[(V(H) \setminus V(H_1)) \cup \{x_1\}], x_1)$ is involution-free, there is some $j \in \mathcal{J}$ such that $\deg_H(x_j)$ is even. Otherwise, for each $j \in \mathcal{J}$, $(H_j, x_j)$ is a bristle. Hence $H$ has an involution which fixes $H_1$ but exchanges $H_{1+d}$ with $H_{\ell+1-d}$ for each $d \in \{1, \dots, \lfloor \ell/2 \rfloor\}$. We will consider two cases, depending on $\ell$.

**Case 1.** $\ell$ is odd. We split the analysis into two cases.
**Case 1.1.** There is a $j \in \{\lceil \ell/2 \rceil, \lceil \ell/2 \rceil + 1\}$ such that $\deg_H(x_j)$ is even. See Figure 4.9. Without loss of generality, suppose that $j = \lceil \ell/2 \rceil$ (otherwise this could be achieved by

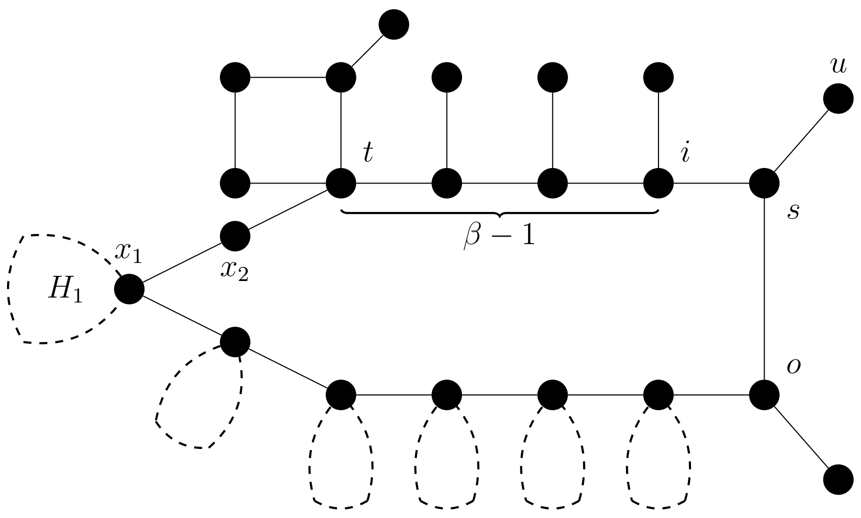

Figure 4.10: The cactus gadget of Case 1.2 of Lemma 4.25. Here, $\ell = 13$, $\beta = 4$, and $K = \{u\}$.

relabelling the vertices of $C$, going the other way around the cycle). We will construct a cactus gadget. Let $\beta = 1$, $s = x_{j+1}$, $t = x_j$, and $i = t$. Let $o$ be the neighbour of $s$ in $C$ that is not $t$. (That is, $o = x_1$ if $\ell = 3$ and $o = x_{j+1}$, otherwise.) Let $O = \{o\}$ and let $K = \Gamma_H(s) \setminus \{o, i\}$. For every $u \in K$ let $w(u) = s$ and $k(u) = \ell - 1$.

To see that this is a cactus gadget, consider $o$ and $y' \in \{o, i\}$. $s$ is the only vertex that is adjacent to $o$, $y'$ and $t$. However, there are an even number of vertices that are adjacent to $t$ since $\deg_H(t)$ is even.

Consider the $(\ell - 1)$-walks from $w(u) = s$ to each $u \in K$, noting that every vertex in $K$ is a neighbour of $s$. Since $s = x_{\lceil \ell/2 \rceil + 1}$, no $(\ell - 1)$-walk from $s$ to one of its neighbours can use any edge that is not in $H' = C \cup H_3 \cup \ldots H_\ell$. Since each of $(H_3, x_3), \ldots, (H_\ell, x_\ell)$ is a mosaic, $C$ is the only odd cycle in $H'$. Since $\ell - 1$ is even and the distance from $s$ to $u$ is one, which is odd, there is exactly one $(\ell - 1)$-walk from $s$ to each of its two neighbours $i$ and $o$ in $C$ (going the long way around the cycle) and there are no $(\ell - 1)$-walks from $s$ to $K$, the set of its neighbours outside $C$.

Now consider the primary distance requirement $d_H(v, \{o, i\}) + d_H(v, t) > 0$. This is satisfied for any $v \in V(H) \setminus \{t\}$, including each $v \in V(H_1)$. Finally, consider the secondary distance requirement $d_H(v, s) + d_H(v, \{o, i, u\}) > \ell - 3$. This is true for any $v \in V(H_1)$ since $d_H(x_1, s) + d_H(x_1, \{o, i, u\}) = d_H(x_1, s) + d_H(x_1, o) = \lfloor \ell/2 \rfloor + \lfloor \ell/2 \rfloor - 1 = \ell - 2$.

**Case 1.2.** We are not in Case 1.1 but there is still a $j \in \mathcal{J}$ such that $\deg_H(x_j)$ is even. Without loss of generality, we may assume that $j < \lceil \ell/2 \rceil$, numbering the vertices of the cycle the other way around, if necessary.

Again we will construct a cactus gadget. See Figure 4.10. This time, let $s = x_{\lceil \ell/2 \rceil}$ and $i = x_{\lceil \ell/2 \rceil - 1}$. Since we are not in Case 1.1, $\deg_H(s) = 3$. Choose $t \in \{x_2, \ldots, x_{\lceil \ell/2 \rceil - 1}\}$ such that $\deg_H(t)$ is even and $d_H(t, i)$ is as small as possible. Let $\beta = d_H(t, s)$. Let $o = x_{\lceil \ell/2 \rceil + 1}$, which also has degree 3, since we are not in Case 1.1; let

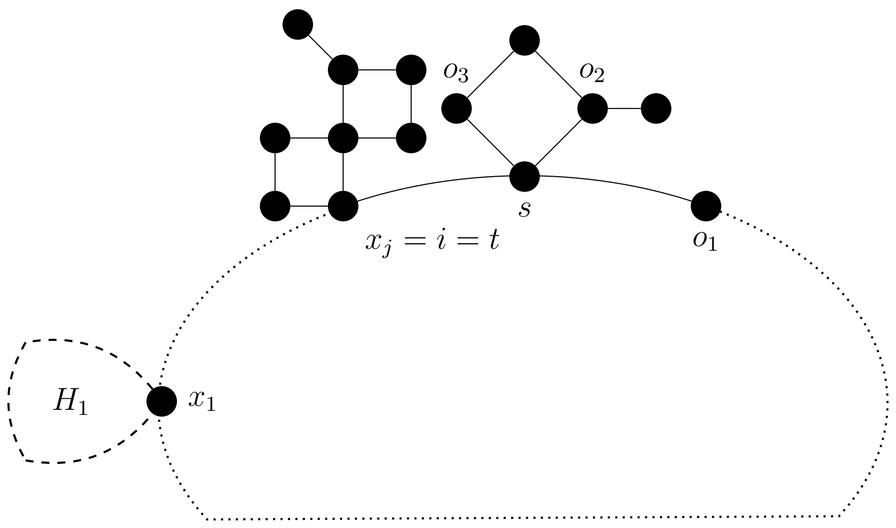

Figure 4.11: An example of a cactus gadget for Case 2.1 of Lemma 4.25. Here $O = \{o_1, o_2, o_3\}$; the dotted line indicates omitted portions of the cycle $C$.

$O = \{o\}$. Let $u$ be the neighbour of $s$ that is not in $C$ and let $K = \{u\}$. Let $w(u) = s$ and $k(u) = \ell - 1$.

To see that this is a cactus gadget, consider $o$ and $y' \in \{o, i\}$. $d_H(o, t) = \beta + 1$ and $s$ is the only neighbour of $o$ that has any $\beta$-walks to $t$, and it has one such walk. Also, $s$ is adjacent to $y'$. On the other hand, there are an even number of $(\beta + 1)$-walks from $i$ to $t$. All of these walks repeat an edge, and there are an even number of edges that can be repeated because, by the choice of $t$, every vertex between $i$ and $t$ has odd degree. The proof that $w(u) = s$ has an even number of $k(u)$-walks to $u$ and odd number of $k(u)$-walks to each of $o$ and $i$ is exactly the same as in the proof of Case 1.1.

Now consider the primary distance requirement $d_H(v, \{o, i\}) + d_H(v, t) > \beta - 1$. This follows for any $v$ which is not on the unique shortest path in $H$ from $t$ to $i$, which includes every $v \in V(H_1)$. Finally, consider the secondary distance requirement $d_H(v, s) + d_H(v, \{o, i, u\}) > \ell - 3$. As in Case 1.1, this is true for any $v \in V(H_1)$ since $d_H(x_1, s) + d_H(x_1, \{o, i, u\}) = \ell - 2$.

**Case 2.** $\ell$ is even. Recall that $\ell \neq 4$ by the hypothesis of the lemma. Choose $j \in \mathcal{J}$ such that $\deg_H(x_j)$ is even; again, we may assume that $j \leq \ell - 2$, and $1 \notin \mathcal{J}$ by definition. Let $s = x_{j+1}$. We will construct a cactus gadget for $H$ in each of two cases.
**Case 2.1.** $\deg_H(s)$ is even. See Figure 4.11. Let $\beta = 1$, $i = t = x_j$ and $O = \Gamma_H(s) \backslash \{i\}$. Note that $|O|$ is odd. Let $K = \emptyset$.

To see that this is a cactus gadget, consider any $o \in O$ and $y' \in O \cup \{i\}$. $s$ is the unique vertex adjacent to $o$, $y'$ and $t$. Since $\deg_H(x_j)$ is even, there are an even number of 2-walks from $i$ to $t = i$. Consider the primary distance requirement $d_H(v, O \cup \{i\}) + d_H(v, t) > 0$. This is satisfied for every $v \neq t$, including every $v \in V(H_1)$. There are no secondary distance requirements since $K = \emptyset$.
**Case 2.2.** $\deg_H(s)$ is odd so, in fact, $\deg_H(s) = 3$. See Figure 4.12.

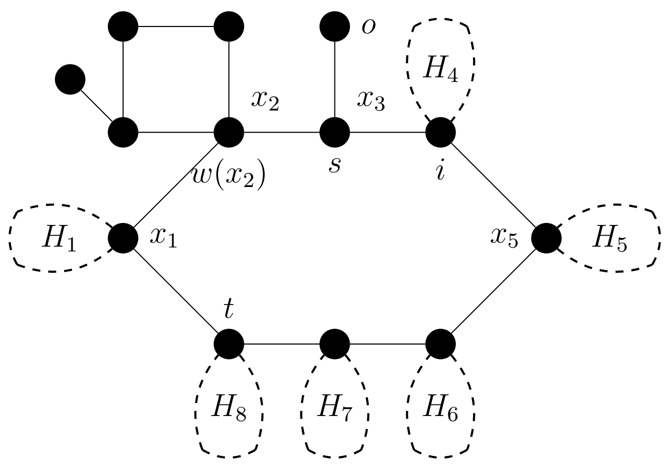

Figure 4.12: An example of a cactus gadget for Case 2.2 of Lemma 4.25. In this case, $\ell = 8$, $\beta = 3$ and $j = 2$.

Let $\beta = \ell/2 - 1$. Let $i = x_{j+2}$, $K = \{x_j\}$ and $O = \Gamma_H(s) \setminus \{x_j, x_{j+2}\}$, which is a single vertex: call this $o$. Let $t$ be the unique vertex in $C$ at distance $\ell/2$ from $i$. Let $w(x_j) = x_j$ and $k(x_j) = 2$.

To see that this is a cactus gadget, consider $o$ and any $y' \in \{o, i\}$. There is a unique $(\beta + 1)$-walk from $o$ to $t$, and this goes via $s$. Thus, $s$ is the unique vertex adjacent to $o$ and $y'$ that has an odd number (one) of $\beta$-walks to $t$. By construction, there are exactly two $(\beta + 1)$-walks from $i$ to $t$, one going each way around the cycle $C$. Since $\deg_H(x_j)$ is even, there are an even number of 2-walks from $x_j$ to itself, but there is exactly one 2-walk from $x_j$ to each of $o$ and $i$. The primary distance requirement is $d_H(v, \{o, i\}) + d_H(v, t) > \beta - 1$. Since $d_H(t, \{o, i\}) = \beta + 1$, this holds for any $v$. Finally, the secondary distance requirement is $d_H(v, x_j) + d_H(v, \{o, i\} \cup K) > 0$. This holds for any $v \neq x_j$, including all $v \in V(H_1)$. □

## 4.7 Cactus gadgets in cactus graphs

In this section, we show that every non-trivial involution-free cactus graph contains a cactus gadget. We first show that every involution-free cactus graph with a distinguished vertex that is not a mosaic contains a cactus gadget or a partial cactus gadget.

For graphs with distinguished vertices, Observation 4.19 allows an inductive proof. Given an involution-free graph with distinguished vertices $(H, x)$ in which $x$ is a cut vertex, we can take the split $\{H_1, \ldots, H_\kappa\}$ and recurse on the graphs with distinguished vertices $\{(H_1, x), \ldots, (H_\kappa, x)\}$, since these are also involution-free. If $x$ is not a cut vertex or splitting at $x$ does not give helpful subgraphs with distinguished vertex, we instead cut $H$ up by deleting the edges of an appropriate cycle $C$ to give components which can have the vertices in $C$ as distinguished vertices. One of these contains $x$, so needs special attention; the others are dealt with inductively.

Finally, we need to show that every non-trivial, involution-free cactus graph contains a cactus gadget. To do this, we temporarily distinguish a suitable vertex.

**Lemma 4.26.** *An involution-free tree with a distinguished vertex $(H, x)$ with at least three vertices contains a partial cactus gadget.*

*Proof.* Let $o$ be a leaf of $H$ at maximal distance from $x$. Let $s$ be the neighbour of $o$. Since $(H, x)$ is involution-free and $H$ contains at least three vertices $d_H(x, o) > 1$ so $s \neq x$. Let $i$ be the neighbour of $s$ on the path to $x$. Any neighbour of $s$ that is not on the path to $x$ must be a leaf as, otherwise, there would be a leaf farther from $x$ than $o$ is. But no vertex in an involution-free tree can be adjacent to more than one leaf, so $\deg_H(s) = 2$. Let $P$ be the (unique) path in the tree $H$ from $i$ to $x$. The partial cactus gadget is $(s, i, \{o\}, P)$. □

We say that a cut vertex $x \in V(H)$ is *cycle-separating* if at least two of the components of the split of $H$ at $x$ contain cycles.

**Lemma 4.27.** *Let $(H, x)$ be a connected, involution-free cactus graph with a distinguished vertex. Then at least one of the following is true:*

- *$H$ contains a cactus gadget satisfying the distance requirements for $x$, or*
- *$(H, x)$ contains a partial cactus gadget, or*
- *$(H, x)$ is a shortcut-free mosaic.*

*Proof.* The proof is by induction on the number of cycles in $H$. If $H$ is acyclic and is a single vertex or a single edge, $(H, x)$ is a shortcut-free mosaic; if it is acyclic and has more than one edge, it contains a partial cactus gadget by Lemma 4.26.

Otherwise, $H$ contains at least one cycle. If $x$ is a cycle-separating cut vertex, let $\{H_1, \dots, H_\kappa\}$ be the split of $H$ at $x$. Every $(H_j, x)$ is an involution-free cactus graph and has fewer cycles than $H$. If some $H_j$ contains a hardness gadget that satisfies the distance requirements for $x$, this is also a cactus gadget in $H$ and still satisfies the distance requirements. Likewise, a partial cactus gadget in some $(H_j, x)$ is also a partial cactus gadget in $(H, x)$. If there is no hardness or partial cactus gadget in any $(H_j, x_j)$ then, by the inductive hypothesis, every $(H_j, x)$ is a shortcut-free mosaic. It follows that $(H, x)$ is, itself, a shortcut-free mosaic. It is a mosaic because involution-freedom of $(H, x)$ guarantees that the $(H_j, x)$ are pairwise non-isomorphic so, in particular, $x$ has at most one bristle in $H$. It is shortcut-free because any shortcut in $(H, x)$ must be inside some $(H_j, x)$, but all of them are shortcut-free.

For the remainder of the proof, we assume that $x$ is not a cycle-separating cut vertex (indeed, it is not necessarily even a cut vertex). Let $C = x_1 \dots x_\ell x_1$ be a cycle such that there is a path from $x$ to $x_1$ in which only $x_1$ is on a cycle. (If $x$ is on a

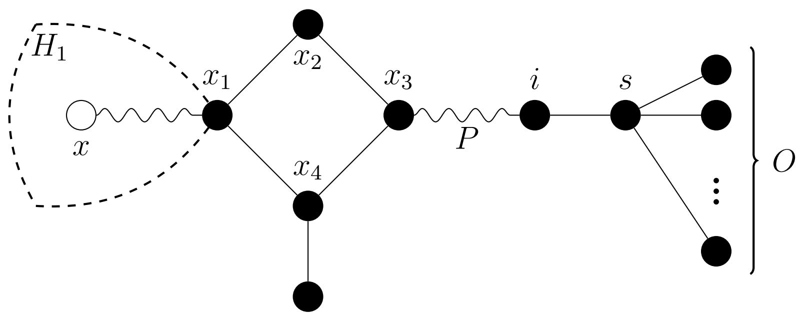

Figure 4.13: An example for Case 1 of Lemma 4.27. The wavy lines represent an ommited portion of the graph whose structure is unimportant, except that it contains a unique shortest path between the endpoints of each wavy line.

cycle, then $C$ is this cycle, $x_1 = x$ and the path is trivial.) For $j \in [\ell]$, let $H_j$ be the component of $H - E(C)$ that contains $x_j$. Thus, $x \in V(H_1)$. We will use the fact below that $x = x_1$ if $x$ is on a cycle. Otherwise, there is a unique path in $H$ from $x$ to $x_1$.

Each of the graphs with distinguished vertices $(H_j, x_j)$ is a cactus graph with fewer cycles than $H$ so, by the inductive hypothesis, each contains a cactus gadget satisfying the distance restrictions for $x_j$, contains a partial cactus gadget, or is a shortcut-free mosaic. If any of $H_2, \dots, H_\ell$ contains a cactus gadget, this is a cactus gadget in $H$ so we are done.

If $\ell$ is odd, let $\mathcal{J} = \{2, \dots, \ell\}$; otherwise, let $\mathcal{J} = \{2, \dots, \ell\} \setminus \{\ell/2 + 1\}$. Thus, $\mathcal{J}$ is the set of indices $j > 1$ such that $H$ contains a unique shortest path from $x$ to $x_j$.

Suppose that, for some $j \in \mathcal{J}$, $(H_j, x_j)$ contains a partial cactus gadget $(s, i, O, P)$. In $H$, $x$ has a unique shortest path $P'$ to $x_j$ and $(s, i, O, P'P)$ is a partial cactus gadget in $(H, x)$.

Otherwise, for every $j \in \mathcal{J}$, $(H_j, x_j)$ is a shortcut-free mosaic. If $\ell \neq 4$ then, by Lemma 4.25, $H$ contains a cactus gadget that satisfies the distance requirements for every vertex in $H_1$, which includes $x$.

We are left with the case $\ell = 4$. $(H_2, x_2)$ and $(H_4, x_4)$ are mosaics and $(H_3, x_3)$ contains a partial cactus gadget or is a shortcut-free mosaic.

**Case 1.** $(H_3, x_3)$ contains a partial cactus gadget.

If $(H_2, x_2)$ is a proper mosaic then by Lemma 4.14, it contains a 2,3-path. Then, by Lemma 4.22, $H$ contains a cactus gadget that satisfies the distance requirements for every vertex $v \in V(H) \setminus (V(H_2) \cup V(H_3))$ and this includes $v = x$. Similarly, there is a cactus gadget if $(H_4, x_4)$ is a proper mosaic.

So suppose that neither of $(H_2, x_2)$ and $(H_4, x_4)$ is a proper mosaic. Since $(H, x_1)$ is involution-free, one of $(H_2, x_2)$ and $(H_4, x_4)$ is a single edge and the other is a single vertex. Suppose without loss of generality that $x_2$ is a single vertex. See Figure 4.13. Now let $H_1' = C \cup H_1 \cup H_2 \cup H_4$ and let $P'$ be the empty path. Then $(P', x_2, x_4)$ is a

2-3 path in $(H_1', x_3)$. By Lemma 4.22, $H$ has a cactus gadget that satisfies the distance requirements for every $v \in V(H) \setminus (V(H_3) \cup \{x_2, x_4\})$ including $v = x$.

**Case 2.** $(H_3, x_3)$ is a shortcut-free mosaic, which means that $H' = C \cup H_2 \cup H_3 \cup H_4$ is an involution-free proper mosaic when either $x_1$ or $x_3$ is distinguished.

First, suppose that $x \neq x_1$. If $\deg_H(x_1)$ is odd then, by Lemma 4.21, $H$ has a cactus gadget that satisfies the distance requirements for $x_1$ and this also satisfies the distance requirements for $x$. If $\deg_H(x_1)$ is even, let $s = x_1$, let $i$ be $s$'s neighbour on the shortest path to $x$ and let $P$ be the unique shortest path from $x$ to $i$ in $H$. $(s, i, \Gamma_H(s) \setminus \{i\}, P)$ is a partial cactus gadget in $H$.

Finally, suppose that $x = x_1$. Since $x$ is not a cycle-separating cut vertex, every component of the split of $H$ at $x$ apart from $(H', x)$ is a tree. If any of these contains more than one edge, it contains a partial cactus gadget by Lemma 4.26. Otherwise, either $(H', x)$ is the unique component of the split, or there is exactly one other component, which is the one-edge tree with $x$ as distinguished vertex. In either case, $(H, x)$ is a mosaic. If $(H, x)$ is shortcut-free, we are done; if not, it contains a cactus gadget satisfying the distance requirements for $x$, by Corollary 4.18. □

We now show how to find cactus gadgets in cactus graphs without distinguished vertices by choosing an appropriate vertex to act temporarily as distinguished.

**Theorem 4.28.** *Every involution-free cactus graph $H$ with more than one vertex contains a cactus gadget.*

*Proof.* Let $H$ be an involution-free cactus graph with more than one vertex. Split $H$ at a cut vertex $x$ into components with distinguished vertices $(H_1', x), \dots (H_k', x)$ with $|V(H_1')| \geq \dots \geq |V(H_k')|$, choosing $x$ to maximize $|V(H_2')|$. Each $(H_j', x)$ is an involution-free, cactus graph with a distinguished vertex and, if any of them contains a cactus gadget satisfying the distance requirements for $x$, then this is also a cactus gadget in $H$ and we are done. Otherwise, by Lemma 4.27 each $(H_j', x)$ contains a partial cactus gadget or is a shortcut-free mosaic.

If $|V(H_2')| > 2$, then each of $(H_1', x)$ and $(H_2', x)$ is either a proper mosaic or contains a partial cactus gadget. Therefore, $H$ contains a cactus gadget, by Lemma 4.24 (two partial cactus gadgets), Lemma 4.20 (two proper mosaics) or Corollary 4.23 (one of each).

$|V(H_2')|$ cannot be 1 since $H$ is involution free. So suppose that $|V(H_2')| = 2$. $H$ is not a tree because then there would be a vertex $y$ with $\deg_H(y) \geq 3$, and choosing $x = y$ would give $|V(H_2')| > 2$. $H$ does not have two cycles because every involution-free cactus graph with two cycles contains a vertex $y$ with $\deg(y) \geq 3$ and choosing $x = y$ would give $|V(H_2')| > 2$. Further, $x$ must be on $H$'s single cycle as, otherwise, we could have chosen a vertex on the path from $x$ to the cycle as our cut vertex and, again, obtained $|V(H_2')| > 2$.

Let the single cycle $C$ of $H$ be $x_1x_2\cdots x_\ell x_1$. For $j \in [\ell]$, let $H_j$ be the component containing $x_j$ in $H - E(C)$. Clearly, $|V(H_j)| \le 2$ — otherwise we could have chosen $x = x_j$ to achieve $|V(H_2')| > 2$.

For $H$ to be involution-free, we must have $\ell \ge 6$. Since $H$ is involution-free, the graph with distinguished vertices $(H[V(H) \setminus V(H_1) \cup \{x_1\}], x_1)$ is involution free, and for each $j \in [\ell(C)] \setminus \{1\}$, $H_j$ is an isolated vertex or a bristle, so $(H_j, x_j)$ is a mosaic. Lemma 4.25 guarantees the existence of a cactus gadget in $H$. □

## 4.8 A dichotomy theorem for cactus graphs

We have shown that all connected, involution-free cactus graphs contain a cactus gadget and that $\#_2$HomsTo$H$ is $\#_2\,\mathsf{P}$-complete for any involution-free cactus graph that has a cactus gadget. To deal with graphs that have involutions, we use reduction by involutions. By Theorem 2.6, every graph $H$ has a unique (up to isomorphism) involution-free reduction $H^*$. By Theorem 2.4, $\#_2$HomsTo$H$ has the same complexity as $\#_2$HomsTo$H^*$.

If $H$ is a tree (as it was for Faben and Jerrum), then its involution-free reduction $H^*$ is connected. However, for graphs that contain cycles, the fact that $H$ is connected does not imply that $H^*$ is connected.[1] Another result that we need from Faben and Jerrum is [32, Theorem 6.1], which allows us to deal with disconnected graphs:

**Lemma 4.29.** *Let $H$ be an involution-free graph. If $H$ has a component $H^*$ for which $\#_2$HomsTo$H^*$ is $\#_2\,\mathsf{P}$-complete, then $\#_2$HomsTo$H$ is $\#_2\,\mathsf{P}$-complete.*

We can now prove the main theorem of this chapter.

**Theorem 1.11.** *Let $H$ be a graph whose involution-free reduction $H^*$ is a cactus graph. If $H^*$ has at most one vertex then $\#_2$HomsTo$H$ is solvable in polynomial time; otherwise, $\#_2$HomsTo$H$ is $\#_2\,\mathsf{P}$-complete.*

*Proof.* Let $H^*$ be the involution-free reduction of $H$. By Theorem 2.4, $\#_2$HomsTo$H$ has the same complexity as $\#_2$HomsTo$H^*$. If $H^*$ has at most one vertex, then, by Corollary 2.9, $\#_2$HomsTo$H^*$ is in $\mathsf{P}$. Otherwise, let $H^{**}$ be any component of $H^*$ with more than one vertex. Such a component must exist since, otherwise, $H^*$ would be a graph with at least two vertices and no edges, and any such graph has an involution.

If $H^{**}$ has two or more vertices, then it has a cactus gadget by Theorem 4.28.

We have established that either $H^*$ has at most one vertex, in which case both $\#_2$HomsTo$H^*$ and $\#_2$HomsTo$H$ are in $\mathsf{P}$, or that some component $H^{**}$ of $H^*$ has a cactus gadget. In the latter case, $\#_2$HomsTo$H^{**}$ is $\#_2\,\mathsf{P}$-complete by Corollary 4.9.

[1]For example, consider non-isomorphic, disjoint, connected, involution-free graphs $H_1$ and $H_2$ and let $H$ be a graph made by adding two disjoint paths of the same length from some vertex $x_1 \in H_1$ to some vertex $x_2 \in H_2$. The only involution of this graph exchanges the interior vertices of the two paths, so $H^* = H_1 \cup H_2$, which is disconnected.

$\#_2$HomsTo$H^*$ is $\#_2\,\mathsf{P}$-complete by Lemma 4.29, so $\#_2$HomsTo$H$ is $\#_2\,\mathsf{P}$-complete. □

# Chapter 5

# Counting homomorphisms (modulo 2) to square-free graphs

The results of this chapter are published in the paper "Counting Homomorphisms to Square-Free Graphs Modulo 2" co-authored with Leslie Goldberg and David Richerby [44].

## 5.1 Introduction

In this chapter we study the complexity of $\#_2$HomsToH when $H$ is a graph that has no 4-cycle (whether induced or not). In particular, we prove the conjecture of Faben and Jerrum for every graph $H$ whose involution-free reduction has no 4-cycle. Since there can be cactus graphs that contain 4-cycles, the results of this chapter do not subsume the results of Chapter 4. Graphs without 4-cycles are called "square-free" graphs. Our main theorem is the following.

**Theorem 1.12.** *Let $H$ be a graph whose involution-free reduction $H^*$ is square-free. If $H^*$ has at most one vertex, then $\#_2$HomsToH is in $\mathsf{P}$; otherwise, $\#_2$HomsToH is $\#_2\mathsf{P}$-complete.*

If $H$ is square-free, then so is every induced subgraph, including its involution-free reduction $H^*$. Thus, we have the following corollary.

**Corollary 5.2.** *Let $H$ be a square-free graph. If its involution-free reduction $H^*$ has at most one vertex, then $\#_2$HomsToH is in $\mathsf{P}$; otherwise, $\#_2$HomsToH is $\#_2\mathsf{P}$-complete.*

In order to prove that $\#_2$HomsToH is $\#_2\mathsf{P}$-hard it is necessary to take a much more abstract approach than we did in Chapters 3 and 4. Since it is not possible to decompose $H$ using a tree-like decomposition as we did in Theorem 4.28 we will identify the $(\gamma, p)$-gadgets we defined in Section 3.4 in all involution-free square-free graphs directly. Then, the hardness part of Theorem 1.12 follows directly from Corollary 4.5.

When counting modulo 2, the only $(\gamma, p)$-gadgets that give hardness are $(0,2)$-gadgets. The much more general nature of $(0,2)$-gadgets, compared to the cactus gadgets, makes them much easier to find and, in some cases, allows us to prove the existence of parts of them non-constructively. (Recall that gadgets depend only on the fixed graph $H$ and not on the input $G$ so they can be hard-coded into the reduction — there is no need to find one constructively.) We no longer need to find unique shortest paths in $H$ or, indeed, any paths at all. In fact, all the gadgets that we construct in this chapter use a "caterpillar gadget" (Definition 5.4) which allows us to use *any* specified path in the graph $H$ instead of relying on a unique shortest path, as in Chapter 4. Thus we no longer need to decompose $H$ in order to find gadgets but, instead, we find $(0,2)$-gadgets in $H$ "in situ". Note that caterpillar gadgets work only in square-free graphs.

When a graph has two even-degree vertices, we can directly use those vertices and a caterpillar gadget to produce a $(0,2)$-gadget (see Lemma 5.9). This already provides a self-contained proof of Faben and Jerrum's dichotomy for trees. Next, for graphs with only one even-degree vertex, we show (Corollary 5.11) that deleting an appropriate set of vertices leaves a component with two even-degree vertices and show (Lemma 5.13) how to simulate that vertex deletion with gadgets. This leaves only graphs in which every vertex has odd degree. In such a graph, we are able to use any shortest odd-length cycle to construct a gadget (Lemma 5.19). If there are no odd cycles, the graph is bipartite. In this interesting case (Lemma 5.22) we use our version of Lovász's result to find a gadget non-constructively.

### 5.1.1 Organisation

The gadgets that we use are a special case of $(\gamma, p)$-gadgets (Definition 3.9) which we formally define in Section 5.3. Section 5.3.1 introduces a gadget that we use extensively, but which requires $H$ to be square-free, as discussed in Section 5.3.2. In Section 5.4, we show how to find hardness gadgets for all square-free graphs and, in Section 5.5, we tie everything together to prove the dichotomy theorem.

## 5.2 Preliminaries

As in the previous chapters, graphs are undirected and have no parallel edges or self-loops. We will also use the partially labelled graphs and graphs with distinguished vertices that we introduced in Section 2.2. We use $\Gamma_H(v)$ to denote the set of neighbours of vertex $v$ in $H$ and $\deg_H(v)$ to denote the degree of $v$ in $H$. A path $P$ is a graph with vertex set $V(P)$ and edge set $E(P)$. However, where convenient, we specify $P$ by simply listing the vertices of the path, in order.

Recall that an involution is an automorphism of order 2. Further recall Definition 2.7 of an order $p$ reduced form $H^{*p}$ of a graph $H$. In the special case of $p = 2$ we will call this graph the *involution-free reduction* of $H$ and simply denote it with $H^*$.

## 5.3 Gadgets for counting modulo 2

Recall that, by Corollary 4.5, if an involution-free graph $H$ has a $(0, 2)$-gadget, then $\#_2$HomsTo$H$ is $\#_2$ P-complete. So, the technical work in this chapter is about finding $(0, 2)$-gadgets in square-free graphs $H$. Recall Definition 3.9 of a $(\gamma, p)$-gadget. A $(0, 2)$-gadget is a $(\gamma, p)$-gadget for $\gamma = 0$ and $p = 2$, so we can simplify the definition of a $(0, 2)$-gadget to the following:

**Definition 5.3.** A $(0, 2)$-*gadget* $(i, s, (J_1, y), (J_2, z), (J_3, y, z))$ for a graph $H$ consists of vertices $i$ and $s$ of $H$ together with three connected, partially $H$-labelled graphs with distinguished vertices $(J_1, y)$, $(J_2, z)$ and $(J_3, y, z)$, where $y$ and $z$ are distinct, that satisfy certain properties as explained below. Let

$$\begin{aligned}\Omega_y &= \{a \in V(H) \mid |\mathrm{Hom}((J_1, y) \to (H, a))| \text{ is odd}\},\\ \Omega_z &= \{b \in V(H) \mid |\mathrm{Hom}((J_2, z) \to (H, b))| \text{ is odd}\}, \text{ and}\\ N_3(u, v) &= |\mathrm{Hom}((J_3, y, z) \to (H, u, v))|.\end{aligned}$$

The properties that we require are the following.

1. $|\Omega_y|$ is even and $i \in \Omega_y$.
2. $|\Omega_z|$ is even and $s \in \Omega_z$.
3. For each $o \in \Omega_y - i$ and each $x \in \Omega_z - s$, $N_3(o, x)$ is even.
4. $N_3(i, s)$ is odd and, for each $o \in \Omega_y - i$ and each $x \in \Omega_z - s$, $N_3(o, s)$ and $N_3(i, x)$ are odd.

The main simplification is that of property 2. Recall Definition 3.9 to see, that for each $b \in \Omega_z$, $N_2(b) \equiv 1 \pmod 2$, so $\sum_{b\in\Omega_z} N_2(b) = |\Omega_z|$ and $\kappa \equiv 1 \pmod 2$.

### 5.3.1 Caterpillar gadgets

All our $(0, 2)$-gadgets use the following "caterpillar gadgets" as $J_3$. We will also use two further kinds of gadget, "neighbourhood gadgets" and "$\ell$-cycle gadgets", but we defer their definitions to the sections where they are used. As we will see in the following section, caterpillar gadgets rely on $H$ being square-free.

**Definition 5.4.** For a path $P = v_0 \dots v_k$ in $H$, with $k \geq 1$, define the *caterpillar gadget* $J_P = (G, \tau)$ as follows (see Figure 5.1). $V(G) = \{u_1, \dots, u_{k-1}, w_1, \dots, w_{k-1}, y, z\}$ and $G$ is the path $yu_1 \dots u_{k-1}z$ together with edges $(u_j, w_j)$ for $1 \leq j \leq k - 1$. $\tau = \{w_1 \mapsto v_1, \dots, w_{k-1} \mapsto v_{k-1}\}$.

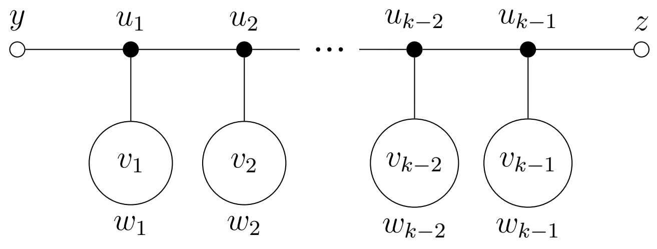

Figure 5.1: The caterpillar gadget corresponding to a path $v_0 \dots v_k$. The vertices $w_1, \dots, w_{k-1}$ in the gadget are pinned to vertices $v_1, \dots, v_{k-1}$ in $H$, respectively. Normal vertices appear as black dots, distinguished vertices as small white circles. Pinned vertices appear as large white circles where the label inside the vertex indicates what the vertex is pinned to.

Note that, if $P$ is a single edge, $G(J_P)$ is also the single edge $(y, z)$ and $\tau(J_P) = \emptyset$.

In the following, we will repeatedly make use of the following fact about square-free graphs: if two distinct vertices have a common neighbour, they must have a unique common neighbour, since a pair of vertices with two common neighbours would form a 4-cycle.

**Lemma 5.5.** *Let $H$ be a square-free graph, let $k > 0$ and let $P = v_0 \dots v_k$ be a path in $H$.*

1. *For any $a \in \Gamma_H(v_0) - v_1$ and $\sigma \in \mathrm{Hom}((J_P, y) \to (H, a))$, $\sigma(u_j) = v_{j-1}$ for all $j \in [k-1]$.*
2. *For any $b \in \Gamma_H(v_k) - v_{k-1}$ and $\sigma \in \mathrm{Hom}((J_P, z) \to (H, b))$, $\sigma(u_j) = v_{j+1}$ for all $j \in [k-1]$.*

*Proof.* The result is trivial for $k = 1$ so we assume $k > 1$. We prove the first part, by induction on $j$. The second part follows by symmetry (call the vertices on the path $v_k \dots v_0$ instead of $v_0 \dots v_k$).

First, take $j = 1$. From the structure of $J_P$, $\sigma(u_1)$ must be a neighbour of $\sigma(y) = a$ and of $v_1$, which are distinct vertices. $v_0$ is a common neighbour of $a$ and $v_1$, so it must be their unique common neighbour, so $\sigma(u_1) = v_0$. Now, suppose that $\sigma(u_{j-1}) = v_{j-2}$. As in the base case, $\sigma(u_j)$ must be some neighbour of $v_{j-2}$ and $v_j$, which are distinct. $v_{j-1}$ is such a vertex, so it is the unique such vertex. ☐

**Lemma 5.6.** *Let $H$ be a square-free graph. Let $k > 0$ and let $P = v_0 \dots v_k$ be a path in $H$ with $\deg_H(v_j)$ odd for all $j \in \{1, \dots, k-1\}$. Let $\Omega_y \subseteq \Gamma_H(v_0)$ and $\Omega_z \subseteq \Gamma_H(v_k)$, with $i = v_1 \in \Omega_y$ and $s = v_{k-1} \in \Omega_z$. For each $o \in \Omega_y - i$ and each $x \in \Omega_z - s$:*

1. *$|\mathrm{Hom}((J_P, y, z) \to (H, o, x))| = 0$,*
2. *$|\mathrm{Hom}((J_P, y, z) \to (H, o, s))| = 1$,*
3. *$|\mathrm{Hom}((J_P, y, z) \to (H, i, x))| = 1$ and*

*4.* $|\mathrm{Hom}((J_P, y, z) \to (H, i, s))|$ *is odd.*

*Proof.* If $k = 1$, $i = v_1$, $s = v_0$, $G(J_P)$ is the single edge $(y, z)$ and $\tau(J_P) = \emptyset$. For any $o \in \Omega_y - i$ and $x \in \Omega_y - s$, we have $(o, s), (i, s), (i, x) \in E(H)$ so $(o, x) \notin E(H)$ because $H$ is square-free. Parts 1–4 are immediate. For the remainder of the proof, we may assume that $k \geq 2$. Note that when $k = 2$, $i = s = v_1$ and this is the unique common neighbour of $v_0$ and $v_2$ in $H$.

For part 1, suppose, towards a contradiction, that $\sigma \in \mathrm{Hom}((J_P, y, z) \to (H, o, x))$. In particular, $\sigma \in \mathrm{Hom}((J_P, y) \to (H, o))$ so, by Lemma 5.5(1), $\sigma(u_1) = v_0$. We also have $\sigma \in \mathrm{Hom}((J_P, z) \to (H, x))$ so, by Lemma 5.5(2), $\sigma(u_1) = v_2$. But $P$ is a simple path so $v_0 \neq v_2$.

For part 2, let $\sigma \in \mathrm{Hom}((J_P, y, z) \to (H, o, s))$. Since $\sigma \in \mathrm{Hom}((J_P, y) \to (H, o))$, $\sigma(u_j) = v_{j-1}$ for all $i \in [k-1]$ by Lemma 5.5(1). But now, $\sigma$ is completely determined, so it is the unique element of $\mathrm{Hom}((J_P, y, z) \to (H, o, s))$. Part 3 follows similarly from Lemma 5.5(2).

For part 4, first note that there is a homomorphism $\sigma^+ \in \mathrm{Hom}((J_P, y, z) \to (H, i, s))$ with $\sigma^+(u_j) = v_{j+1}$ for all $j \in [k-1]$. Now, for $m \in [k-1]$, let

$$S_m = \{\sigma \in \mathrm{Hom}((J_P, y, z) \to (H, i, s)) \mid m \text{ is minimal such that } \sigma(u_m) \neq v_{m+1}\}.$$

The sets $\{\sigma^+\}$ and $S_1, \ldots, S_{k-1}$ partition $\mathrm{Hom}((J_P, y, z) \to (H, i, s))$.

We claim that, for any $\sigma \in S_m$, $\sigma(u_j) = v_{j-1}$ for all $j > m$. This is trivial for $S_{k-1}$ so let $\sigma \in S_m$ with $m < k-1$. $\sigma(u_{m+1})$ must be a neighbour of both $\sigma(w_{m+1}) = v_{m+1}$ and $\sigma(u_m) \in \Gamma_H(v_m)$. By definition of $S_m$, these are distinct vertices so $v_m$ is their unique common neighbour and so $\sigma(u_{m+1}) = v_m$. Now, if $\sigma(u_j) = v_{j-1}$ for some $j \in \{m+1, \ldots, k-2\}$, then $\sigma(u_{j+1})$ must be a neighbour of both $\sigma(w_{j+1}) = v_{j+1}$ and $v_{j-1}$: $v_j$ is the unique such vertex, so $\sigma(u_{j+1}) = v_j$. This establishes the claim.

But, now, for any $\sigma \in S_m$, we have $\sigma(u_j) = v_{j+1}$ for $j < m$ and $\sigma(u_j) = v_{j-1}$ for $j > m$. $\sigma(y) = i$, $\sigma(z) = s$ and $\sigma(w_j) = v_j$ for each $j \in [k-1]$. Finally, $\sigma(u_m)$ may take any value in $\Gamma_H(v_m) - v_{m+1}$. It follows that, for all $m$, $|S_m| = \deg_H(v_m) - 1$, which is even. $|\mathrm{Hom}((J_P, y, z) \to (H, i, s))| = 1 + \sum_m |S_m|$, which is odd, as required. □

### 5.3.2 Caterpillar gadgets and 4-cycles

Before proceeding to find $(0, 2)$-gadgets for square-free graphs in the next section, we pause to show why 4-cycles cause problems for caterpillar gadgets and, in particular, why Lemma 5.6 does not apply to graphs containing 4-cycles.

Consider first the one-edge caterpillar gadget $J_1$ associated with the path $v_0 v_1$ in the graph $H_1$ in Figure 5.2. This corresponds to $k = 1$ in Lemma 5.6 and we have $i = v_1$ and $s = v_0$. Taking $\Omega_y = \Gamma_{H_1}(v_0) = \{v_0', v_1\}$ and $\Omega_z = \Gamma_{H_1}(v_1) = \{v_0, v_1'\}$ satisfies the conditions of the lemma. However, taking $o = v_0' \in \Omega_y - i$ and $x = v_1' \in \Omega_z - s$, we

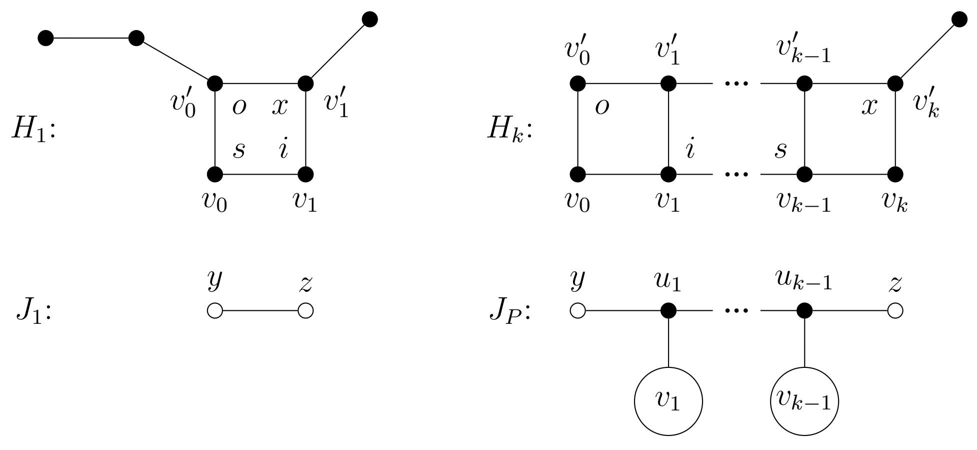

Figure 5.2: Examples of graphs containing 4-cycles for which caterpillar gadgets (Definition 5.4 and Lemma 5.6) fail. The graphs $H_1$ and $H_k$ ($k \geq 2$) are shown, along with the caterpillar gadgets $J_1$ and $J_P$, corresponding to the paths $v_0v_1$ and $v_0 \dots v_k$, respectively. The labels $o$, $s$, $i$ and $x$ are referenced in the text. Normal vertices appear as black dots, distinguished vertices as small white circles. Pinned vertices appear as large white circles where the label inside the vertex indicates what the vertex is pinned to.

have $|\mathrm{Hom}((J_1, y, z) \to (H, o, x))| = 1$ so part 1 of the lemma does not hold. However, the other three parts hold, as

$$\begin{aligned} |\mathrm{Hom}((J_1, y, z) \to (H, o, s))| &= |\mathrm{Hom}((J_1, y, z) \to (H, i, x))| \\ &= |\mathrm{Hom}((J_1, y, z) \to (H, i, s))| = 1\,. \end{aligned}$$

Now, consider longer paths such as the path $P = v_0 \dots v_k$ in $H_k$ in Figure 5.2, for some $k \geq 2$. The associated caterpillar gadget $J_P$ is also shown in the figure. For each $j \in \{1, \dots, k-1\}$, $\deg_{H_k}(v_i)$ is odd. We have $i = v_1$ and $s = v_{k-1}$ (with $i = s$ in the case $k = 2$). Again, take $\Omega_y = \Gamma_{H_k}(v_0) = \{v'_0, v_1\}$, take $\Omega_z = \Gamma_{H_k}(v_k) = \{v_{k-1}, v'_k\}$ and take $o = v'_0 \in \Omega_y - i$ and $x = v'_k \in \Omega_z - s$.

Once again part 1 of the lemma fails. We have $|\mathrm{Hom}((J_P, y, z) \to (H_k, o, x))| = 1$, since there is a homomorphism that maps $u_j$ to $v'_j$ for each $j \in \{1, \dots, k-1\}$. This is the only possible homomorphism from $(J_P, y, z)$ to $(H_k, o, x)$ since there is only one $k$-path from $o$ to $x$ that the $k$-path in $J_P$ can be mapped to. For a $(0, 2)$-gadget, it would suffice for $|\mathrm{Hom}((J_P, y, z) \to (H_k, o, x))|$ to be even (not necessarily zero) but it is odd for every $k$.

For $H_k$, the other parts of the lemma fail, too. We have

$$|\mathrm{Hom}((J_P, y, z) \to (H, o, s))| = |\mathrm{Hom}((J_P, y, z) \to (H, i, x))| = k.$$

When the target is $(H, o, s)$, the $k$-path in $J_P$ can be mapped to any of the $k$ $k$-paths in $H_k$ from $o$ to $s$ (following along $v'_0v'_1 \dots$ and then dropping down along an edge $v'_jv_j$ and then following $v_jv_{j+1} \dots v_{k-1}$). The case with target $(H, i, x)$ is similar.

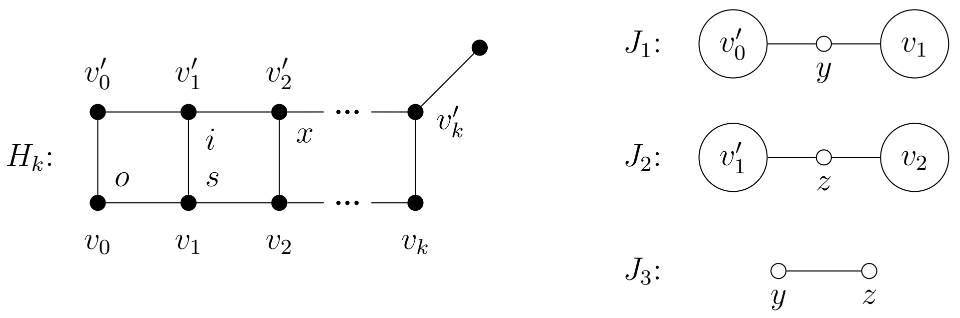

Figure 5.3: A $(0, 2)$-gadget for the graph $H_k$ (see also Figure 5.2). Normal vertices appear as black dots, distinguished vertices as small white circles. Pinned vertices appear as large white circles where the label inside the vertex indicates what the vertex is pinned to.

So in both cases, the number of homomorphisms is $k$. When $k$ is odd, this is not a real problem. The purpose of Lemma 5.6 is to show that caterpillar gadgets can be used as $J_3$ in a $(0, 2)$-gadget, and the definition of hardness gadgets only requires that $N_3(o, s)$ and $N_3(i, x)$ (i.e., $|\mathrm{Hom}((J_P, y, z) \to (H, o, s))|$ and $|\mathrm{Hom}((J_P, y, z) \to (H, i, x))|$, respectively) be odd and not necessarily 1. However, this relaxation doesn't help when $k$ is even.

Finally, for part 4, consider a homomorphism from $(J_P, y, z)$ to $(H, i, s)$. The image of the path $yu_1 \dots u_{k-1}z$ in $H$ must be a $k$-walk $v_1x_1 \dots x_{k-1}v_{k-1}$ with the property that $x_j$ is adjacent to $v_j$ for each $j \in \{1, \dots, k-1\}$. This means that $x_j \in \{v_{j-1}, v_j', v_{j+1}\}$. There are two kinds of $k$-walk satisfying these criteria. The first kind uses only the vertices $\{v_0, \dots, v_k\}$. Such a walk must be either $v_1v_0v_1v_2 \dots v_{k-1}$ or $v_1 \dots v_\alpha v_{\alpha+1} v_\alpha \dots v_{k-1}$ for some $\alpha \in \{1, \dots, k-1\}$. The second kind uses some of the vertices $\{v_1', \dots, v_{k-1}'\}$. Such a walk must be of the form $v_1 \dots v_\alpha v_\alpha' \dots v_\beta' v_\beta \dots v_{k-1}$ for some $1 \le \alpha \le \beta \le k-1$. There are $k$ walks of the first kind and $\frac{1}{2}k(k-1)$ of the second. Thus,

$$|\mathrm{Hom}((J_1, y, z) \to (H, i, s))| = k + \tfrac{1}{2}k(k-1) = \tfrac{1}{2}k(k+1)\,,$$

which is odd if and only if $k$ is congruent to 1 or 2, mod 4 but is required to be odd for all $k$.

We note that $\#_2$HomsTo$H_1$ is $\#_2\,\mathsf{P}$-complete, as is $\#_2$HomsTo$H_k$, for every $k \ge 2$. $H_1$ is an involution-free cactus graph with more than one vertex so it is hard by the main theorem of Chapter 4. We claim that $\mathcal{X} = (i, s, (J_1, y), (J_2, z), (J_3, y, z))$, as shown in Figure 5.3, is a $(0, 2)$-gadget for $H_k$. We have $\Omega_y = \{v_0, v_1'\} = \{o, i\}$ and $\Omega_z = \{v_1, v_2'\} = \{s, x\}$: both are even and $i \in \Omega_y$ and $s \in \Omega_z$. There is no edge $ox$ in $H_k$ so $N_3(o, x) = 0$, which is even. There are edges $os$, $ix$ and $is$ in $H_k$, so $N_3(o, s) = N_3(i, x) = N_3(i, s) = 1$, which is odd. This establishes that $\mathcal{X}$ is a $(0, 2)$-gadget so, since $H_k$ is involution-free, $\#_2$HomsTo$H_k$ is $\#_2\,\mathsf{P}$-complete by Corollary 4.5. Ironically, the part $J_3$ of $\mathcal{X}$ is the one-edge caterpillar gadget associated with the path $v_1v_1'$ in $H_k$. The failure of Lemma 5.6 in the presence of 4-cycles only means that caterpillar gadgets are not guaranteed to work, not that they never work.

## 5.4 Finding $(0, 2)$-gadgets

In this section, we show how to find $(0, 2)$-gadgets for all connected, involution-free, square-free graphs. The simplest case is when the graph contains at least two vertices of even degree. Faben and Jerrum [32] used the fact that all involution-free trees have at least two even-degree vertices, though we use different gadgets because we are dealing with graphs containing cycles as well as trees. For graphs with only one even-degree vertex, we show that an appropriate vertex deletion produces a component with more than one even-degree vertex and show how to simulate such a vertex deletion using gadgets.

This leaves graphs where every vertex has odd degree. In Section 5.4.2, we show how to use odd-length cycles to find a $(0, 2)$-gadget. The remaining case, bipartite graphs in which every vertex has odd degree, is covered in Section 5.4.3, where we use Corollary 5.21, our version of Lovász's result, to non-constructively demonstrate that a $(0, 2)$-gadget always exists.

We will use the following fact.

**Lemma 5.7.** *An involution-free graph with at least two vertices but at most one even-degree vertex contains a cycle.*

*Proof.* We prove the contrapositive. Let $G$ be an involution-free acyclic graph. At most one component of $G$ is an isolated vertex so, if $G$ has two or more vertices, it has at least one component with two or more vertices. This component is an involution-free tree which, by [32, Lemma 5.3], contains at least two vertices of even degree. □

### 5.4.1 Even-degree vertices

We prove that involution-free graphs containing at least one vertex of positive, even degree have a $(0, 2)$-gadget. In this section, we will use one extra kind of gadget.

**Definition 5.8.** For a vertex $v \in V(H)$, define the *neighbourhood gadget* $J_{\Gamma(v),x} = (G, \{w \mapsto v\})$, where $G$ is the single edge $(x, w)$.

It is immediate from the definition that, for any $v \in V(H)$,

$$|\mathrm{Hom}((J_{\Gamma(v),x}, x) \to (H, u))| = \begin{cases} 1 & \text{if } u \in \Gamma_H(v) \\ 0 & \text{otherwise.} \end{cases}$$

We first show how to find hardness gadgets for connected graphs containing at least two even-degree vertices (their degree must be positive, since the graph is connected) and then deal with the harder case of graphs containing exactly one vertex of positive, even degree. The following lemma constructs a caterpillar gadget, so the lemma depends on $H$ being square-free. The extended conclusion about pinned vertices is needed for technical reasons in the proof of Lemma 5.13.

**Lemma 5.9.** *Let $H$ be a connected, square-free graph with at least two even-degree vertices. Then $H$ has a $(0,2)$-gadget $(i, s, (J_1, y), (J_2, z), (J_3, y, z))$. Furthermore, we can choose $J_1$, $J_2$ and $J_3$ so that each contains at least one pinned vertex.*

*Proof.* Let $v_0 \dots v_m$ be a path in $H$ between distinct even-degree vertices $v_0$ and $v_m$ and let $P = v_0 \dots v_k$, where $k \in \{1, \dots, m\}$ is minimal such that $\deg_H(v_k)$ is even. We claim that $(v_1, v_{k-1}, (J_{\Gamma(v_0),y}, y), (J_{\Gamma(v_k),z}, z), (J_P, y, z))$ is a $(0,2)$-gadget. $|\Omega_y|$ and $|\Omega_z|$ are even because $v_0$ and $v_k$ have even degree; and they contain $v_1$ and $v_{k-1}$, respectively. The remaining properties required by Definition 5.3 hold by Lemma 5.6, since $v_1, \dots, v_{k-1}$ have odd degree.

Each of $J_{\Gamma(v_0),y}$ and $J_{\Gamma(v_k),z}$ contains a pinned vertex and, if $k > 1$, $J_P$ also contains at least one pinned vertex. If $k = 1$, then $G(J_P)$ is the single edge $(y, z)$ and $\tau(J_P) = \emptyset$. However, we may add to $G(J_P)$ a new vertex $w_0$ and an edge $(w_0, y)$ and set $\tau(J_P) = \{w_0 \mapsto v_0\}$: this requires $y$ to be mapped to a neighbour of $v_0$. This has no effect on the $(0,2)$-gadget since Definition 5.3 only imposes requirements on $|\mathrm{Hom}((J_3, y, z) \to (H, a, b))|$ when $a \in \Omega_y$. Since $\Omega_y = \Gamma_H(v_0)$, we are already only considering homomorphisms that map $y$ to a neighbour of $v_0$ and the change to $J_3$ is merely restating this condition. $\square$

It is worth noting that, since all involution-free trees have at least two even-degree vertices, Lemma 5.9 implies Faben and Jerrum's dichotomy for $\#_2$HomsToH where $H$ is a tree [32]. They also use two even-degree vertices but their gadgets rely on the fact that there is a unique path between two vertices of a tree, which doesn't hold in general graphs. However, from Lemma 5.9, we conclude that uniqueness of the path is not required and we can prove hardness even when there are multiple paths between even-degree vertices.

To handle graphs with fewer than two vertices of even degree, we first investigate the results of deleting vertices from such graphs. If we delete the unique even-degree vertex from a connected graph, then each component of the resulting graph contains at least one vertex of even degree. If we are lucky, one of the resulting components will contain two or more vertices of even degree, raising the hope that we can use Lemma 5.9 to prove $\#_2\mathsf{P}$-completeness. If all of the resulting components have exactly one even-degree vertex, then we can iterate, deleting those vertices to obtain yet more fragments. As long as the graph contains at least one cycle, it is not hard to see that we can eventually obtain a component with two or more even-degree vertices. However, to apply Lemma 5.9, we must ensure that the resulting component has no involution. We prove this in the following two lemmas.

**Lemma 5.10.** *Let $H$ be an involution-free graph with exactly one vertex $v$ of positive, even degree. Then $H' = H - v$ is also involution-free.*

*Proof.* Each vertex $u \in \Gamma_H(v)$ has odd degree in $H$ and has exactly one neighbour removed, so $\deg_{H'}(u)$ is even. Suppose, towards a contradiction, that $\rho$ is an involution of $H'$. No automorphism can map an odd-degree vertex to an even-degree vertex or vice-versa and $\Gamma_H(v)$ is exactly the set of even-degree vertices in $H'$. Therefore, the restriction of $\rho$ to the neighbours of $v$ is a permutation. Define $\hat{\rho}\colon V(H) \to V(H)$ by $\hat{\rho}(v) = v$ and $\hat{\rho}(w) = \rho(w)$ for $w \neq v$. $\hat{\rho}$ preserves all edges in $H'$ and all edges incident on $v$ in $H$, so it is an involution of $H$, contradicting the supposition that $H$ has no involution. $\square$

So far, we have described our goal as being to iteratively delete vertices until we find a component with more than one even-degree vertex. This is a useful intuition but we do not know how to simulate such a sequence of vertex deletions using gadgets. Instead, we show how to achieve the goal of a component with more than one even-degree vertex by deleting a set of vertices, which we do know how to do with a gadget.

For a vertex $v \in V(H)$ and an integer $r \geq 0$, let $B_r(v) = \{u \in V(H) \mid d(u,v) = r\}$.

**Corollary 5.11.** *Let $H$ be an involution-free graph that has exactly one vertex $v$ of positive, even degree. For some $r$, $H - B_r(v)$ has an involution-free component $H^*$ that does not contain $v$ but does contain at least two even-degree vertices. Furthermore, we can take $r = \min\{d(v,w) \mid w \text{ is on a cycle}\}$.*

*Proof.* $H$ contains a cycle by Lemma 5.7 so we can take $r$ as in the statement of the lemma and this is well-defined. If $r = 0$, then $v$ is in some cycle $C$ in $H$. $H - v$ has no involution by Lemma 5.10, so no component of $H - v$ has an involution. The component $H^*$ of $H - v$ that contains $C - v$ contains at least two vertices of $\Gamma_H(v)$ ($v$'s two neighbours in $C$) and these vertices have even degree in $H^*$. $H^*$ does not, of course, contain $v$.

Suppose that $r > 0$. By the choice of $r$, there must be a component $H'$ of $H - B_{r-1}(v)$ that contains a vertex $v_r \in B_r(v)$ that is in a cycle $C'$ of $H'$. Since no vertex at distance less than $r$ from $v$ is in a cycle in $H$, there is a unique path from $v$ to $v_r$. Let this be $v_0 \dots v_r$, where $v = v_0$. A simple induction on $j = 0, \dots, r-1$, using Lemma 5.10, shows that the component of $H - v_j$ containing $v_r$ has no involution, does not contain $v$ and has exactly one even-degree vertex: namely, $v_{j+1}$. In particular, the component of $H - v_{r-1}$ that contains $v_r$ is $H'$. But, now, the component of $H' - v_r$ that contains $C' - v_r$ has no involution (because no component of $H' - v_r$ has an involution) and contains at least two vertices of even degree (because $v_r$ has at least two neighbours in $C'$). Further, this component is the component $H^*$ of $H - B_r(v)$ that we seek. $\square$

Thus, starting with an involution-free graph $H$ containing only one vertex of positive, even degree, we have shown how to make a set of vertex deletions (some set $B_r(v)$)

to obtain an involution-free component $H^*$ with at least two even-degree vertices. We now show that we can achieve these vertex deletions using gadgetry. The following technical lemma allows us to construct a gadget that, in a sense, "selects" the vertices of $H^*$ within $H$.

**Lemma 5.12.** *Let $H$ be a graph, let $P = x_0 \dots x_{r+1}$ with $r \geq 0$ be a path in $H$ and let $w \in V(H)$. If every vertex in $H$ within distance $r-1$ of $w$ has odd degree, then $|\mathrm{Hom}((P, x_0) \to (H, w))|$ has opposite parity to the number of distinct $r$-paths in $H$ from $w$ to vertices of even degree.*

*Proof.* We prove the lemma by induction on $r$. For $r = 0$, the result is trivial. The condition on vertices within distance $r-1$ is vacuous. The number of 0-paths from $w$ to vertices of even degree is zero if $\deg(w)$ is odd; it is one if $\deg(w)$ is even; and $|\mathrm{Hom}((P, x_0) \to (H, w))| = \deg(w)$.

Suppose the result holds for the path $P = x_0 \dots x_{r+1}$ and consider the path $Px_{r+2}$ and a graph $H$ in which every vertex within distance $r$ of $w$ has odd degree.

Every homomorphism $\sigma$ from $(Px_{r+2}, x_0)$ to $(H, w)$ induces a homomorphism $\hat{\sigma}$ from $(P, x_0)$ to $(H, w)$. Write $\sigma \sim \sigma'$ if $\hat{\sigma} = \hat{\sigma}'$. $\sim$ is an equivalence relation and its equivalence classes partition $\mathrm{Hom}((Px_{r+2}, x_0) \to (H, w))$. Let $[\![\sigma]\!]$ be the $\sim$-equivalence class of $\sigma$.

If every vertex within distance $r$ of $w$ in $H$ has odd degree, there are no $r$-paths from $w$ to vertices of even degree so, by the inductive hypothesis, there are an odd number of homomorphisms from $(P, x_0)$ to $(H, w)$, so there are an odd number of equivalence classes. Further, $|[\![\sigma]\!]| = \deg(\sigma(x_{r+1}))$ (this is well-defined since $\sigma(x_{r+1}) = \hat{\sigma}(x_{r+1})$, so all homomorphisms $\sigma' \in [\![\sigma]\!]$ agree on the value of $\sigma'(x_{r+1})$). Any vertex of even degree is at distance at least $r+1$ from $w = \sigma(x_0)$ so, if $\deg_H(\sigma(x_{r+1}))$ is even, then the $r$-walk $\sigma(x_0)\sigma(x_1) \dots \sigma(x_{r+1})$ is, in fact, a simple $(r+1)$-path. Therefore, the number $N$ of even-cardinality equivalence classes is equal to the number of $(r+1)$-paths in $H$ from $w$ to a vertex of even degree, and subtracting these from the total number of equivalence classes gives $|\mathrm{Hom}((Px_{r+2}, x_0) \to (H, w))| \equiv 1 - N \bmod 2$, as required. □

Now, we can obtain a $(0,2)$-gadget for $H$ by combining the "selection gadget" with the $(0,2)$-gadget for the subgraph $H^*$ given to us by Corollary 5.11.

**Lemma 5.13.** *Any involution-free, square-free graph $H$ that has exactly one vertex $v$ of positive, even degree has a $(0,2)$-gadget.*

*Proof.* Let $r = \min\{d(v, w) \mid w \text{ is on a cycle}\}$. By Corollary 5.11, there exists an involution-free component $H^*$ of $H - B_r(v)$ that does not contain $v$ but contains at least two vertices of even degree. $H^*$ is square-free because it is an induced subgraph of a square-free graph. Therefore, by Lemma 5.9, there is a $(0,2)$-gadget in $H^*$, $\mathcal{X}^* = (i, s, (J_1^*, y), (J_2^*, z), (J_3^*, y, z))$, in which each of $J_1^*$, $J_2^*$ and $J_3^*$ contains a pinned vertex.

We construct a $(0,2)$-gadget $\mathcal{X}$ for $H$ from $\mathcal{X}^*$. Let $P$ be a path of length $r+1 \geq 1$, with vertices $x_0 \dots x_{r+1}$. Let $J_1 = (G, \tau)$ be the partially $H$-labelled graph such that $\tau = \tau(J_1^*)$ and $G$ is defined from $G(J_1^*)$ as follows: start with $G(J_1^*)$ and, for every vertex $u \in G(J_1^*)$, add a new copy of $P$ and identify that copy's vertex $x_0$ with $u$. Define $J_2$ and $J_3$ similarly, from $J_2^*$ and $J_3^*$. We claim that the tuple

$$\mathcal{X} = \Big(i, s, (J_1, y), (J_2, z), (J_3, y, z)\Big)$$

is the desired $(0,2)$-gadget for $H$.

To find out what $\mathcal{X}$ does, we first consider homomorphisms from one copy of the path $P$ to $H$. For a vertex $w \in V(H)$, let $N_w = |\mathrm{Hom}((P, x_0) \to (H, w))|$. If $d(v,w) = r$ (i.e., $w \in B_r(v)$), then there is a unique $r$-path from $w$ to a vertex of even degree. This is because $v$ is the unique vertex of even degree and, if there were distinct $r$-paths $Q_1$ and $Q_2$ from $w$ to $v$ then $Q_1 \cup Q_2$ would contain a cycle, which would contain vertices at distance strictly less than $r$ from $v$, contradicting the definition of $r$. If $d(v,w) > r$, then there are no $r$-paths from $w$ to even-degree vertices. Therefore, by Lemma 5.12, $N_w$ is even if $d(v,w) = r$ and $N_w$ is odd if $d(v,w) > r$ (we will see that the parity of $N_w$ does not matter if $d(v,w) < r$).

Now, let $a \in V(H)$ and consider homomorphisms $\sigma, \sigma' \in \mathrm{Hom}((J_1, y) \to (H, a))$. Write $\sigma \sim \sigma'$ if $\sigma(u) = \sigma'(u)$ for all $u \in V(G(J_1^*))$ and write $[\![\sigma]\!]$ for the $\sim$-equivalence class containing $\sigma$. $|\mathrm{Hom}((J_1, y) \to (H, a))|$ is the sum of the sizes of the $\sim$-equivalence classes. For any $\sigma$, we have

$$|[\![\sigma]\!]| \;=\; \prod_{x \in V(G(J_1^*))} |\mathrm{Hom}((P, x_0) \to (H, \sigma(x)))|\,.$$

Therefore, $|[\![\sigma]\!]|$ is even if $\sigma$ maps any vertex of $G(J_1^*)$ into $B_r(v)$. In this case, $|[\![\sigma]\!]|$ contributes nothing to the sum, modulo 2.

Thus, we may restrict our attention to homomorphisms from $J_1^*$ to $H$ that have no vertex in $B_r(v)$ in their image. $J_1^*$ is connected and contains a vertex pinned to a vertex in $H^*$. Therefore, restricting to homomorphisms that have no vertex in $B_r(v)$ in their image means restricting to homomorphisms whose image is wholly within $H^*$. For any vertex $w \in H^*$, $d_H(v,w) > r$, so this gives

$$|\mathrm{Hom}((J_1, y) \to (H, a))| \equiv |\mathrm{Hom}((J_1^*, y) \to (H^*, a))| \pmod 2\,,$$

for any $a \in V(H^*)$ and $|\mathrm{Hom}((J_1, y) \to (H, a))| \equiv 0 \bmod 2$, for $a \notin V(H^*)$; and similarly for $J_2$ and $J_3$. Thus, since $\mathcal{X}^*$ is a $(0,2)$-gadget for $H^*$, $\mathcal{X}$ is a $(0,2)$-gadget for $H$. $\square$

The proof of Lemma 5.13 does not explicitly use caterpillar gadgets. However, the $(0,2)$-gadget $\mathcal{X}$ is constructed from $\mathcal{X}^*$, which was produced by Lemma 5.9. It follows

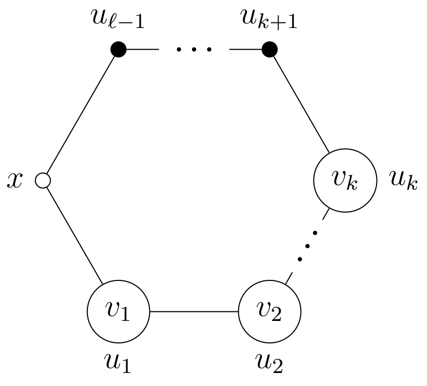

Figure 5.4: The $\ell$-cycle gadget $J_{\ell,P,x}$ corresponding to a path $P = v_1 \dots v_k$ in an $\ell$-cycle in $H$. Normal vertices appear as black dots, distinguished vertices as small white circles. Pinned vertices appear as large white circles where the label inside the vertex indicates what the vertex is pinned to.

that $J_3^*$ is a caterpillar gadget, so Lemma 5.13 requires $H$ to be square-free, as stated.

### 5.4.2 Odd cycles

In the previous section, we showed how to find a $(0, 2)$-gadget for any involution-free, square-free graph containing at least one vertex of even degree. In this section, we show that any square-free graph in which all vertices have odd degree has a $(0, 2)$-gadget if it has an odd cycle. We first introduce a gadget for selecting certain vertices in cycles.

**Definition 5.14.** (See Figure 5.4). Let $P = v_1 \dots v_k$ be a path in $H$. For any $\ell > \max\{2, k\}$, define the *$\ell$-cycle gadget* $J_{\ell,P,x} = (G, \tau)$ where $G$ is the cycle $xu_1 \dots u_{\ell-1}x$ and $\tau = \{u_1 \mapsto v_1, \dots, u_k \mapsto v_k\}$.

We say that the *odd-girth* of a graph is the length of its shortest odd-length cycle. By convention, the odd-girth of a graph without odd cycles is infinite; in the following, we write "a graph whose odd-girth is $\ell$" as a short-hand for "a graph whose odd-girth is finite and equal to $\ell$."

**Lemma 5.15.** *Let $H$ be a graph whose odd-girth is $\ell$ and let $G$ be an $\ell$-cycle. The image of $G$ under any homomorphism from $G$ to $H$ is an $\ell$-cycle in $H$.*

*Proof.* Let $G = u_0 \dots u_{\ell-1}u_0$. Since $G$ is an $\ell$-cycle and $H$ contains an $\ell$-cycle, $\mathrm{Hom}(G \to H)$ is non-empty so let $\sigma \in \mathrm{Hom}(G \to H)$. Let $C$ be the image of $G$ under $\sigma$, i.e., subgraph of $H$ consisting of vertices $\{\sigma(u_0), \dots, \sigma(u_{\ell-1})\}$ and edges $\{(\sigma(u_j), \sigma(u_{j+1})) \mid 0 \le j < \ell\}$, with addition on indices carried out modulo $\ell$. Suppose towards a contradiction that $C$ is not an $\ell$-cycle. Since $C$ has at most $\ell$ vertices and at most $\ell$ edges, it cannot have an $\ell$-cycle as a proper subgraph. Since $H$ has no odd cycles shorter than $\ell$, $C$ must be bipartite. But then the walk $\sigma(u_0)\sigma(u_1) \dots \sigma(u_{\ell-1})\sigma(u_0)$ is an odd-length walk from a vertex to itself and no such walk can exist in a bipartite graph. ∎

**Corollary 5.16.** *Let $H$ be a graph whose odd-girth is $\ell$. For any path $P$ on fewer than $\ell$ vertices, $|\mathrm{Hom}((J_{\ell,P,x},x) \to (H,v))|$ is the number of $\ell$-cycles in $H$ that contain the path $vP$.*

*Proof.* By Lemma 5.15, the image of $G(J_{\ell,P,x})$ under any homomorphism to $H$ is an $\ell$-cycle in $H$ and, because of the pinning and distinguished vertex, this cycle must contain the path $vP$. □

Let $\#C_\ell(vw)$ be the number of $\ell$-cycles in $H$ containing the edge $(v,w)$.

**Lemma 5.17.** *Let $H$ be a graph whose odd-girth is $\ell$. Every vertex $v \in V(H)$ has an even number of neighbours $w$ such that $\#C_\ell(vw)$ is odd.*

*Proof.* If $v$ is not in any $\ell$-cycle, the claim is vacuous: the even number is zero. Otherwise, let $C = vw_1 \dots w_{\ell-1}v$ be an $\ell$-cycle in $H$. If $w_j \in \Gamma_H(v)$ for some even $j \neq \ell - 1$, the odd cycle $vw_1 \dots w_j v$ contradicts the stated odd-girth of $H$. If $w_j \in \Gamma_H(v)$ for some odd $j \neq 1$, the odd cycle $vw_j \dots w_{\ell-1}v$ contradicts the odd-girth. Therefore, $w_1$ and $w_{\ell-1}$ are the only vertices in $C$ that are adjacent to $v$ and every $\ell$-cycle through $v$ contributes exactly 2 to $\sum_{w\in\Gamma_H(v)} \#C_\ell(vw)$. Therefore, the sum is even, so it has an even number of odd terms. □

**Lemma 5.18.** *Let $H$ be a square-free graph whose odd-girth is $\ell$. If $H$ contains an edge that is in an odd number of $\ell$-cycles, then $H$ has a $(0,2)$-gadget.*

Note that, for the case $\ell = 3$, any edge in a 3-cycle in $H$ must be in exactly one 3-cycle since, if an edge $(x,y)$ is in distinct 3-cycles $xyzx$ and $xyz'x$, then $xzyz'x$ is a 4-cycle in $H$, which is forbidden by the hypothesis of the lemma. The absence of 4-cycles is also required for the caterpillar gadget produced in the proof.

*Proof.* Let $(i,s)$ be an edge in an odd number of $\ell$-cycles in $H$. Let $J_1$ be the $\ell$-cycle gadget $J_{\ell,s,y}$ (so $\tau(J_1) = \{u_1 \mapsto s\}$) and let $J_2$ be the $\ell$-cycle gadget $J_{\ell,i,z}$. Let $G(J_3)$ be the single edge $(y,z)$ and let $\tau(J_3) = \emptyset$ ($J_3$ is, technically, a caterpillar gadget but it is easier to analyse it directly).

We claim that $(i, s, (J_1, y), (J_2, z), (J_3, y, z))$ is a hardness gadget for $H$. By Corollary 5.16, $|\mathrm{Hom}((J_{\ell,s,y}, y) \to (H, v))|$ is the number of $\ell$-cycles in $H$ that contain the edge $(v,s)$, so

$$\Omega_y = \{v \in V(H) \mid (v,s) \text{ is in an odd number of } \ell\text{-cycles}\}.$$

Thus, $|\Omega_y|$ is even by Lemma 5.17. $\Omega_y$ contains $i$ by the choice of the edge $(i,s)$ in an odd number of $\ell$-cycles. Similarly, $\Omega_z$ is even and contains $s$. To verify the remaining properties required by Definition 5.3, note that $J_3$ is a single edge so, for any $a,b \in V(H)$, $|\mathrm{Hom}((J_3,y,z) \to (H,a,b))|$ is 1 if $(a,b) \in E(H)$ and 0, otherwise. We have $\Omega_y \subseteq \Gamma_H(s)$ and $\Omega_z \subseteq \Gamma_H(i)$ so, for any $o \in \Omega_y - i$ and any $x \in \Omega_z - s$, $H$ contains

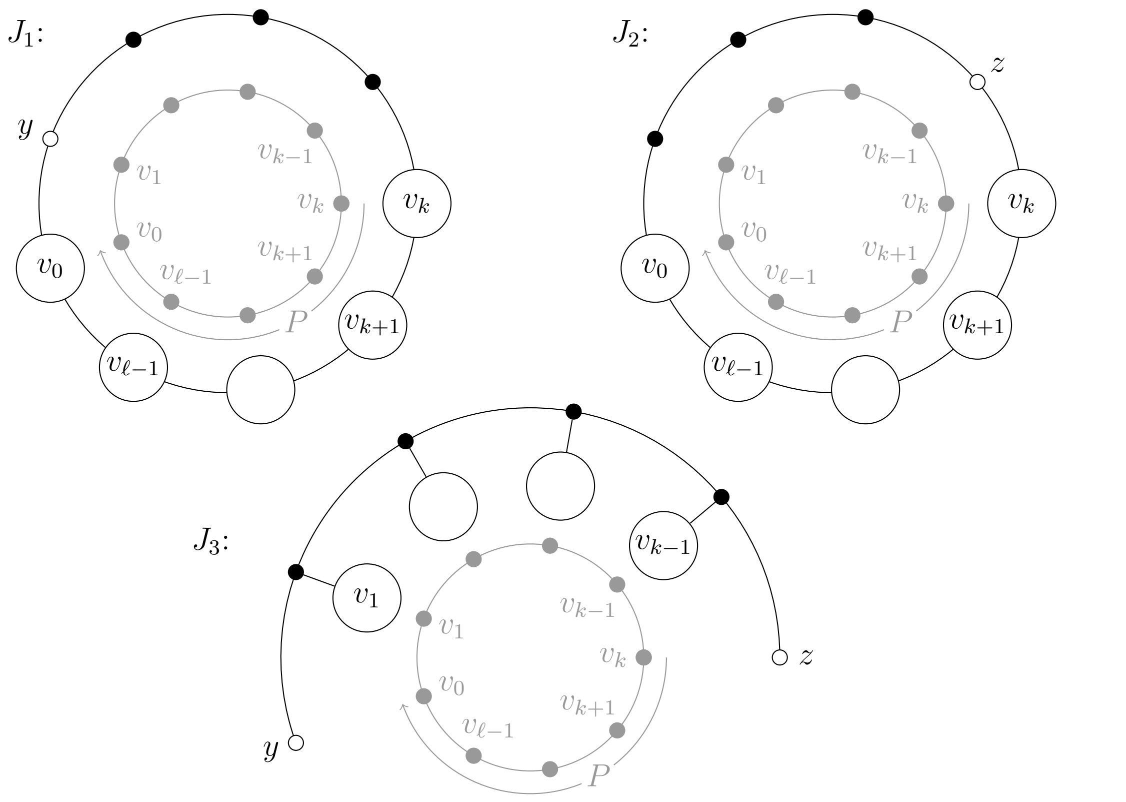

Figure 5.5: The parts $J_1$, $J_2$ and $J_3$ of the $(0, 2)$-gadget constructed in the proof of Lemma 5.19. The corresponding cycle in $H$ is indicated in grey within each gadget. The path $P = v_k \dots v_{\ell-1}v_0$ is undirected but the arrow indicates the order in which the vertices are listed. Normal vertices appear as black dots, distinguished vertices as small white circles. Pinned vertices appear as large white circles where the label inside the vertex indicates what the vertex is pinned to.

the edges $(o, s)$, $(s, i)$ and $(i, x)$ but it cannot contain the edge $(o, x)$ because $H$ is square-free. □

**Lemma 5.19.** *Let $H$ be a square-free graph in which every vertex has odd degree. If $H$ contains an odd cycle, then it has a $(0, 2)$-gadget.*

*Proof.* Let $\ell$ be the odd-girth of $H$. If $H$ contains an edge in an odd number of $\ell$-cycles (which is guaranteed for $\ell = 3$, since $H$ is square-free), then $H$ has a $(0, 2)$-gadget by Lemma 5.18. So, for the remainder of the proof, we may assume that the shortest odd cycle in $H$ has length $\ell > 4$ and that every edge is in a (not necessarily positive) even number of $\ell$-cycles.

Let $P = v_kv_{k+1} \dots v_{\ell-1}v_0$ be a longest path that is in a positive, even number of $\ell$-cycles (see Figure 5.5; it turns out to be most convenient to label the vertices in this order; the path has length $\ell - k$). Such a path certainly exists because any edge in an $\ell$-cycle is in a positive, even number of them. So, in particular, $P$ contains at least one edge. Further $P$ has fewer than $\ell - 1$ edges, because any path on $\ell - 1$ edges is in at most one $\ell$-cycle, since $H$ has no parallel edges. Let $C = v_0v_1 \dots v_{\ell-1}v_0$ be an $\ell$-cycle containing $P$. Let $\mathrm{rev}(P) = v_0 v_{\ell-1} \dots v_k$ be the path $P$ with the vertices listed in the reverse order.

Let $i = v_1$ and $s = v_{k-1}$. Let $J_1$ be the $\ell$-cycle gadget $J_{\ell,\mathrm{rev}(P),y}$, let $J_2$ be the $\ell$-cycle gadget $J_{\ell,P,z}$, and let $J_3$ be the caterpillar gadget $J_{v_0 \dots v_k}$.

We claim that $(i, s, (J_1, y), (J_2, z), (J_3, y, z))$ is a $(0, 2)$-gadget for $H$. Since $P$ was chosen to be a longest path in a positive, even number of $\ell$-cycles, any path $uP$ in $H$ must be in an odd number of $\ell$-cycles or in none at all. Since $P$ itself is in an even number of $\ell$-cycles, the number of extensions $uP$ in an odd number of cycles must be even. By Corollary 5.16, $|\mathrm{Hom}((J_{\ell,P,z}, z) \to (H, u))|$ is the number of $\ell$-cycles in $H$ that contain the path $uP$. Therefore, $\Omega_z$ is precisely the set of vertices $u$ such that $uP$ is in an odd number of $\ell$-cycles, so we have established that $|\Omega_z|$ is even. Since $sP$ is an extension of $P$, it is not in a positive, even number of $\ell$-cycles; it is in at least one $\ell$-cycle (namely, $C$) so it is in an odd number of them. Therefore, $s \in \Omega_z$. Similarly, $|\Omega_y|$ is even and $i \in \Omega_y$.

It remains to verify that the conditions of Lemma 5.6 hold for $J_3$, so that lemma gives us the remaining properties we need from Definition 5.3. All vertices in $H$ have odd degree by assumption, including in particular the interior vertices of $P$. We have already established that $i = v_1 \in \Omega_y$ and $s = v_{k-1} \in \Omega_z$. Finally, $\Omega_y \subseteq \Gamma_H(v_0)$ because, in $G(J_1)$, $y$ is adjacent to a vertex that is pinned to $v_0$. Similarly, $\Omega_z \subseteq \Gamma_H(v_k)$. $\square$

### 5.4.3 Bipartite graphs

The only remaining case is bipartite graphs $H$ in which every vertex has odd degree. We show that, if $H$ has an "even gadget", it has a $(0, 2)$-gadget. And it turns out that every connected bipartite graph with more than one edge has an even gadget.

**Definition 5.20.** An *even gadget* for a bipartite graph $H$ with at least one edge is an edge $(a, b)$ of $H$ together with a connected bipartite graph $G$ with a distinguished edge $(w, x)$ such that $|\mathrm{Hom}((G, w, x) \to (H, a, b))|$ is even.

Note that, for bipartite $G$ and $H$, with edges $(w, x)$ and $(a, b)$, respectively, there is always at least one homomorphism from $(G, w, x)$ to $(H, a, b)$, since the whole of $G$ can be mapped to the edge $(a, b)$. So, although Definition 5.20 only requires $|\mathrm{Hom}((G, w, x) \to (H, a, b))|$ to be even, the number of homomorphisms is always non-zero.

We give a nonconstructive proof that every connected bipartite graph with more than one edge has an even gadget. We will use the following corollary of the proof of Lemma 2.16, which restricts to a certain class of connected graphs.

**Corollary 5.21.** *Let $(H, \bar{y})$ and $(H', \bar{y}')$ be connected involution-free graphs, each with $r$ distinguished vertices, such that $H[\bar{y}]$ and $H'[\bar{y}']$ are also connected. Then $(H, \bar{y}) \cong (H', \bar{y}')$ if and only if, for all connected graphs $(G, \bar{x})$ with $r$ distinguished vertices, such* *that $G[\bar{x}]$ is connected,*

$$|\mathrm{Hom}((G,\bar{x}) \to (H,\bar{y}))| \equiv |\mathrm{Hom}((G,\bar{x}) \to (H',\bar{y}'))| \pmod{2}\,. \tag{5.1}$$

*Proof.* For brevity, we refer to $(G,\bar{x})$ as *appropriate* if it is connected, it has $r$ distinguished vertices and $G[\bar{x}]$ is connected.

As in the proof of Lemma 2.16, the "only if" direction is trivial, so we suppose that (5.1) holds for all appropriate $(G,\bar{x})$. Also, $\bar{y}$ and $\bar{y}'$ must have the same equality type. If they do not, we may assume there are distinct $i$ and $j$ with $y_i = y_j$ but $y'_i \neq y'_j$, and take $G = H[\bar{y}]$. $(G,\bar{y})$ is appropriate but we have $|\mathrm{Hom}((G,\bar{y}) \to (H,\bar{y}))| = 1 \neq |\mathrm{Hom}((G,\bar{y}) \to (H',\bar{y}'))| = 0$, which contradicts the assumption that (5.1) holds for all appropriate $(G,\bar{x})$.

The proof that (5.1) holding for every appropriate $G$ implies that (2.2) holds for $p = 2$ and for every appropriate $G$ proceeds by induction on $|V(G)|$, as in the proof of Lemma 2.16. The base cases are unchanged. To see that the inductive step remains valid, let $(G,\bar{x})$ be appropriate and let $\theta$ be any equivalence relation on $V(G)$. We claim that $(G,\bar{x})/\theta$ is also appropriate. By construction, $(G,\bar{x})/\theta$ has $r$ distinguished vertices. It is connected because it is the result of identifying vertices in a connected graph; $(G/\theta)[\llbracket x_1\rrbracket,\dots,\llbracket x_r\rrbracket]$ is connected for the same reason.

This establishes that (2.2) holds for $p = 2$ and all appropriate $(G,\bar{x})$. Since $(H,\bar{y})$ and $(H',\bar{y}')$ are both appropriate, we can complete the proof in the same way as in the proof of Lemma 2.16, substituting each of these graphs in turn for $(G,\bar{x})$ in (2.2). □

Suppose that $H$ is any connected bipartite graph with more than one edge such that, for some edge $(a,b)$ of $H$, $(H,a,b)$ is involution-free. We will show that $H$ has an even gadget. If, furthermore, $H$ is square-free, this even gadget gives a $(0,2)$-gadget. If $H$ is also involution-free, the $(0,2)$-gadget implies $\#_2\mathsf{P}$-completeness of $\#_2$HomsTo$H$, by Corollary 4.5.

**Lemma 5.22.** *Suppose that $H$ is a connected bipartite graph with more than one edge such that, for some edge $(a,b)$ of $H$, $(H,a,b)$ is involution-free. Then $H$ has an even gadget.*

*Proof.* Let $H$ be a graph satisfying the conditions in the statement of the lemma. Let $K_2$ be the graph consisting of the single edge $(a,b)$. Clearly, $(K_2,a,b)$ is involution-free (since there are no non-trivial automorphisms of $K_2$ that fix $a$ and $b$) and $H \ncong K_2$ since $H$ has more than one edge, so $(H,a,b) \ncong (K_2,a,b)$. By Corollary 5.21 (taking $H' = K_2$ and $\bar{y} = \bar{y}' = (a,b)$), there is a connected graph $(G,w,x)$ with distinguished vertices $w$ and $x$ such that $(w,x)$ is an edge and

$$|\mathrm{Hom}((G,w,x) \to (H,a,b))| \not\equiv |\mathrm{Hom}((G,w,x) \to (K_2,a,b))| \pmod{2}\,. \tag{5.2}$$

$G$ must be bipartite — otherwise

$$|\mathrm{Hom}((G, w, x) \to (H, a, b))| = |\mathrm{Hom}((G, w, x) \to (K_2, a, b))| = 0\,,$$

contradicting (5.2). Thus, $|\mathrm{Hom}((G, w, x) \to (K_2, a, b))| = 1$, so the edge $(a, b)$ of $H$ together with $(G, w, x)$ is an even gadget. □

**Lemma 5.23.** *Suppose that $H$ is a connected, bipartite, square-free graph with more than one edge such that, for some edge $(a, b)$ of $H$, $(H, a, b)$ is involution-free. Suppose that every vertex of $H$ has odd degree. Then $H$ has a $(0, 2)$-gadget.*

*Proof.* By Lemma 5.22, $H$ has an even gadget. Choose an even gadget consisting of an edge $(i, s)$ of $H$ and a connected bipartite graph $G$ with distinguished edge $(w, x)$ so that $N = |\mathrm{Hom}((G, w, x) \to (H, i, s))|$ is even. Choose the even gadget so that the number of vertices of $G$ is as small as possible. There is a homomorphism from $G$ to the edge $(i, s)$ so $N > 0$. $N$ is even, so $G$ cannot be a single edge.

First, we show that $\deg_G(w) \geq 2$. Suppose, towards a contradiction, that $x$ is the only neighbour of $w$ in $G$, i.e. $\deg_G(w) = 1$. If this is the case, then $x$ must have some neighbour $w' \neq w$, since $G$ is not a single edge. We have

$$\begin{aligned} 0 &\equiv |\mathrm{Hom}((G, w, x) \to (H, i, s))| \pmod 2 \\ &\equiv |\mathrm{Hom}((G - w, x) \to (H, s))| \pmod 2 \\ &= \sum_{c \in \Gamma_H(s)} |\mathrm{Hom}((G - w, x, w') \to (H, s, c))|\,. \end{aligned}$$

Since every vertex in $H$ has odd degree, the sum has an odd number of terms. Since the total is even, there must be some $c$ such that $|\mathrm{Hom}((G - w, x, w') \to (H, s, c))|$ is even, contradicting the choice of $G$. By the same argument, also, $\deg_G(x) \geq 2$.

For any vertex $v \in V(G)$, let

$$C(v) = \{c \in V(H) \mid |\mathrm{Hom}((G, w, x, v) \to (H, i, s, c))| \text{ is odd}\}\,.$$

Note that, for any $v \in V(G)$, $|C(v)|$ is even since, otherwise, $N$ would be odd.

We now show that $C(y) \neq \emptyset$ for every $y \in \Gamma_G(x) \setminus \{w\}$. If $C(y) = \emptyset$, then, in particular, $i \notin C(y)$, so $|\mathrm{Hom}((G, w, x, y) \to (H, i, s, i))|$ is even. But then $|\mathrm{Hom}((G', w, x) \to (H, i, s))|$ is even, where $G'$ is the graph made from $G$ by identifying the (distinct) vertices $w$ and $y$ and calling the resulting vertex $w$. This contradicts minimality in the choice of $G$. Similarly, $C(z) \neq \emptyset$ for every $z \in \Gamma_G(w) \setminus \{x\}$. Choose vertices $y \in \Gamma_G(x) \setminus \{w\}$ and $z \in \Gamma_G(w) \setminus \{x\}$.

Finally, let $J$ be the partially $H$-labelled graph $(G, \{w \mapsto i, x \mapsto s\})$ and let $G(J_3)$ be the single edge $(y, z)$ and $\tau(J_3) = \emptyset$. We show that $(i, s, (J, y), (J, z), (J_3, y, z))$ is a $(0, 2)$-gadget for $H$. $\Omega_y = C(y)$ is even and $i \in C(y)$; likewise, $\Omega_z = C(z)$ is even and $s \in C(z)$.

By the choice of $J$, $\Omega_y \subseteq \Gamma_H(s)$ and $\Omega_z \subseteq \Gamma_H(i)$. For any $o \in \Omega_y - i$ and $x \in \Omega_z - s$, $H$ contains edges $(o, s)$, $(s, i)$ and $(i, x)$ so it does not contain the edge $(o, x)$ as it is square-free. Therefore, $N_3(o, s) = N_3(i, s) = N_3(i, x) = 1$ and $N_3(o, x) = 0$ and we have established all the conditions of Definition 5.3. □

## 5.5 Main theorem

We have shown that all connected, square-free, involution-free graphs (and some disconnected graphs, too) have $(0, 2)$-gadgets and that $\#_2$HomsTo$H$ is $\#_2\mathsf{P}$-complete for any involution-free graph that has a $(0, 2)$-gadget. As in Chapter 4 to deal with graphs that have involutions, we use reduction by involutions. By Theorem 2.6, $H$ reduces by involutions to a unique up to isomorphism graph $H^*$. By Theorem 2.4 $\#_2$HomsTo$H$ has the same complexity as $\#_2$HomsTo$H^*$.

We can now prove our main result.

**Theorem 1.12.** *Let $H$ be a graph whose involution-free reduction $H^*$ is square-free. If $H^*$ has at most one vertex, then $\#_2$*HomsTo*$H$ is in $\mathsf{P}$; otherwise, $\#_2$*HomsTo*$H$ is $\#_2\mathsf{P}$-complete.*

*Proof.* As we noted above, $\#_2$HomsTo$H$ has the same complexity as $\#_2$HomsTo$H^*$. If $H^*$ has at most one vertex, then, by Corollary 2.9, $\#_2$HomsTo$H^*$ is in $\mathsf{P}$. Otherwise, let $H^{**}$ be any component of $H^*$ with more than one vertex. Such a component must exist since, otherwise, $H^*$ would be a graph with at least two vertices and no edges, and any such graph has an involution.

If $H^{**}$ has two or more vertices of even degree, then it has a $(0, 2)$-gadget by Lemma 5.9. If $H^{**}$ has exactly one vertex of even degree, it has a $(0, 2)$-gadget by Lemma 5.13. If the previous cases do not apply, then every vertex of $H^{**}$ must have odd degree. By Lemma 5.7, $H^{**}$ contains a cycle. If it contains an odd cycle, it has a $(0, 2)$-gadget by Lemma 5.19. Otherwise, $H^{**}$ is bipartite. By construction, $H^{**}$ is connected and square-free. Since $H^{**}$ contains a cycle, it has more than one edge. Since it is involution-free, it certainly contains an edge $(a, b)$ so that $(H^{**}, a, b)$ is involution-free. Every vertex of $H^{**}$ has odd degree, so it has a $(0, 2)$-gadget by Lemma 5.23.

We have established that either $H^*$ has at most one vertex, in which case both $\#_2$HomsTo$H^*$ and $\#_2$HomsTo$H$ are in $\mathsf{P}$, or that some component $H^{**}$ of $H^*$ has a $(0, 2)$-gadget. In the latter case, $\#_2$HomsTo$H^{**}$ is $\#_2\mathsf{P}$-complete by Corollary 4.5. $\#_2$HomsTo$H^*$ is $\#_2\mathsf{P}$-complete by Lemma 4.29, so $\#_2$HomsTo$H$ is $\#_2\mathsf{P}$-complete. □

# Chapter 6

# Computing the partition function (modulo a prime) of a two-spin system with an external field

## 6.1 Introduction

In this chapter we study the complexity of $\#_p Z_{\gamma,\lambda}$, the problem of computing (modulo a prime $p$) the partition function of a two spin system on multigraphs with an external field. The two-spin model with external field has three parameters, a prime $p$ and $\gamma, \lambda \in \{0, 1, \dots p-1\}$. A configuration $\sigma : V(G) \to \{0, 1\}$ is an assignment of the two spins "0" and "1" to the vertices of $G$. Let $c(\sigma)$ denote the number of edges $(u, v)$ of $G$ with $\sigma(u) = \sigma(v) = 1$ and let $\ell(\sigma)$ denote the number of vertices $u$ of $G$ with $\sigma(u) = 0$. The partition function of the model is given by:

$$Z_{\gamma,\lambda}(G) = \sum_{\sigma:V(G)\to\{0,1\}} \gamma^{c(\sigma)} \lambda^{\ell(\sigma)}.$$

Formally, $\#_p Z_{\gamma,\lambda}$ is the following modular counting problem.

**Problem 6.1.** *Name:* $\#_p Z_{\gamma,\lambda}$.

*Parameter:* A prime $p$ and $\gamma, \lambda \in \{0, 1, \dots p-1\}$.

*Input:* A multigraph $G$.

*Output:* $Z_{\gamma,\lambda}(G) \pmod p$.

In the definition above we could have $\gamma, \lambda \in \mathbb{Z}$, instead of restricting $\gamma, \lambda \in \{0, 1, \dots p-1\}$. This does not weaken our results. To see this let $\gamma, \lambda \in \{0, 1, \dots p-1\}$ and let $\gamma', \lambda' \in \mathbb{Z}$ with $\gamma' \equiv \gamma \pmod p$ and $\lambda' \equiv \lambda \pmod p$. From the definition of $Z_{\gamma,\lambda}$, for any graph $G$, $Z_{\gamma,\lambda}(G) \equiv Z_{\gamma',\lambda'}(G) \pmod p$.

Recall that for a prime $p$, we let $i_p \in \{0, 1, \dots p-1\}$, such that $i_p^2 \equiv -1 \pmod k$. For $p \geq 3$ there can be either zero or two elements satisfying this definition. If there are no elements satisfying the definition, then consider the conditions involving $i_p$ vacuous. If there are two elements satisfying the definition, then it does not matter which one we choose. The main theorem of this chapter is the following.

**Theorem 1.13.** *Let $p$ be a prime and let $\gamma, \lambda \in \{0, 1, \dots p-1\}$. $\#_p Z_{\gamma,\lambda}$ is computable in polynomial time when one of the following holds.*

1. $\lambda = 0$.
2. $\gamma = 1$.
3. $\gamma = -1$ *and* $\lambda \in \{0, \pm 1, \pm i_p\}$.

*Furthermore, $\#_p Z_{\gamma,\lambda}$ is $\#_p\mathsf{P}$-complete when one of the following holds.*

4. $\lambda \not\equiv 0 \pmod{p}$ *and* $\gamma \equiv 0 \pmod{p}$.
5. $\lambda \not\equiv 0 \pmod{p}$, $\gamma \not\equiv \pm 1 \pmod{p}$ *and there exists an integer $k$ with* $\gamma^k \equiv \lambda \pmod{p}$.
6. $\lambda \not\equiv 0 \pmod{p}$, $\gamma \not\equiv \pm 1 \pmod{p}$ *and* $p < 100$, *where* $p \neq 41$.

Theorem 1.13 extends the results we have presented in Section 3.3, for the complexity of $\#_p Z_\gamma$ (Problem 3.3), the problem of computing the partition function of a two spin system on multigraphs without an external field. As we saw in Section 3.3 any instance multigraph $G$ of $\#_p Z_\gamma$ can be seen as an instance of $\#_p\mathrm{CSP}(\{E\})$, where $E$ is the binary function expressing the edge interactions for $G$. Hence Corollary 3.8 that describes the complexity of $\#_p Z_\gamma$ is an immediate corollary of the dichotomy theorem on the complexity of $\#_k\mathrm{CSP}$ by Guo et al. [49].

It is obvious from the definitions of Problem 3.3 and Problem 6.1 that $\#_p Z_\gamma$ is the special case of $\#_p Z_{\gamma,\lambda}$ with $\lambda = 1$. Like $\#_p Z_\gamma$, $\#_p Z_{\gamma,\lambda}$ can also be seen as a special case of $\#_p\mathrm{CSP}$. More precisely any instance multigraph $G$ of $\#_p Z_{\gamma,\lambda}$ can be seen as a restricted input of $\#_p\mathrm{CSP}(\{E, U\})$, where $E$ is the binary function expressing the edge interactions of $G$ and $U$ is the unary function expressing the vertex weights for $G$. It is a restricted instance, since the unary function $U$ is applied exactly once to each variable of the input.

Tractability results for $\#_p\mathrm{CSP}(\{E, U\})$ carry over to $\#_p Z_{\gamma,\lambda}$. If $\#_p\mathrm{CSP}(\{E, U\})$ is tractable, so is every restricted input $\#_p Z_{\gamma,\lambda}$. So the polynomial cases of $\#_p Z_{\gamma,\lambda}$ in Theorem 1.13 come directly from the dichotomy theorem of Guo et al. [49] for the complexity of $\#_k\mathrm{CSP}$. For the hardness results though that is not true in general, as hardness for $\#_p\mathrm{CSP}(\{E, U\})$, does not imply hardness for $\#_p Z_{\gamma,\lambda}$.

To prove our hardness results we reduce $\#_p Z_\gamma$ to $\#_p Z_{\gamma,\lambda}$ for various values of the parameter space. Our reductions, informally, work as follows: Let $G$ be the input multigraph for $\#_p Z_\gamma$. We construct an input multigraph $G'$ for $\#_p Z_{\gamma,\lambda}$ in the following way. Let $(J, x)$ be an "appropriate" gadget-multigraph with a distinguished vertex.

Take the union of $|V(G)|$ disjoint copies of $(J, x)$ and a copy of $G$. For every vertex $v$ of $G$, identify $x$ with $v$. By appropriate we mean that the gadget $(J, x)$ cancells out the vertex weights in $Z_{\gamma,\lambda}$ in the following way. The total "weight" contribution of $J$ to $Z_{\gamma,\lambda}(J)$ when $x$ is mapped to "0" is equivalent to the total "weight" contribution of $J$ to $Z_{\gamma,\lambda}(J)$ when $x$ is mapped to "1". Therefore $Z_{\gamma,\lambda}(G') = k\,(Z_{\gamma,\lambda}(J))^{|V(G)|}\,Z_{\gamma}(G)$, where $k$ is a constant. Since $J$ is a fixed graph, if we where able to compute $Z_{\gamma}(G)$ we would be able to compute $Z_{\gamma,\lambda}(G')$. Therefore, in order to prove hardness for $\#_p Z_{\gamma,\lambda}$ it suffices to find appropriate gadgets $(J, x)$ for the values of the parameters $\gamma$, $\lambda$ and $p$.

To find appropriate gadgets, we also establish pinning for $\#_p Z_{\gamma,\lambda}$. The pinning we use for $\#_p Z_{\gamma,\lambda}$ is similar to the pinning we used for $\#_p$HomsToH: $\#_p$Pinned$Z_{\gamma,\lambda}$ has as an input a multigraph $G$ and a pinning function $\tau : V(G) \to \{0, 1\}$ and as output the value of partition function $Z_{\gamma,\lambda}(G)$ (modulo $p$) over all configuration that respect $\tau$. We show that $\#_p$Pinned$Z_{\gamma,\lambda}$ reduces to $\#_p Z_{\gamma,\lambda}$. This allows us to use multigraphs $J$ that have vertices pinned to either "0" or "1" as gadgets, in order to show hardness for $\#_p Z_{\gamma,\lambda}$.

By using cliques as gadgets we can prove that $\#_p Z_{\gamma,\lambda}$ is hard when $\lambda \not\equiv 0 \pmod{p}$ and $\gamma = 0$. By using two vertices connected with $k$ parallel edges where one of them is pinned to "1" as a gadget, we can show hardness for $\#_p Z_{\gamma,\lambda}$when $\lambda \not\equiv 0 \pmod{p}$, $\gamma \not\equiv \pm 1 \pmod{p}$ and $\gamma^k \equiv \lambda \pmod{p}$. Finally, using a combination of cliques, paths and the "two vertices with $k$ parallel edges" gadget we show that $\#_p Z_{\gamma,\lambda}$ is hard for small primes when $\lambda \not\equiv 0 \pmod{p}$, $\gamma \not\equiv \pm 1 \pmod{p}$.

### 6.1.1 Organisation

In Section 6.3 we explain our choice of vertex weights. In Section 6.4 we give values of the parameters for which $\#_p Z_{\gamma,\lambda}$ is computable in polynomial time.

In Section 6.5 we show that for other values of the parameters, $\#_p Z_{\gamma,\lambda}$ is $\#_p$ P-complete. More specifically in Section 6.5.1 we establish pinning for $\#_p Z_{\gamma,\lambda}$. In Section 6.5.2 we identify a property for the gadgets we will use, in order to obtain hardness results. In Section 6.5.3 we find appropriate gadgets for values of the parameter space.

Finally, in Section 6.6 we tie all our results together in the main theorem of this section.

## 6.2 Preliminaries

In this chapter we only use multigraphs. For a multigraph $G$ and a subset of its edges $U \subseteq V$ we denote with $G[U]$ the sub-multigraph of $G$ induced by $U$. For a multigraph $G$ and $v \in V(G)$ we denote with $G - v$ the sub-multigraph of $G$ induced by $V \setminus \{v\}$. As with graphs we specify simple paths $P$ by simply listing the vertices of the path, in order.

Let $\mathbb{Z}_p$ be the additive group of integers modulo $p$. For a prime $p$ and an integer $k \in \mathbb{Z}_p$, when working in $\mathbb{Z}_p$ we will, sometimes, denote $p-k$ with $-k$.

## 6.3 Vertex weights

We defined $\#_p Z_{\gamma,\lambda}$ so that vertices contribute a factor of $\lambda$ when they are mapped to “0”. We can also consider the version of the problem when the unary weight comes from the vertices mapped to “1”. Let $G$ be a multigraph, let $\sigma : V(G) \to \{0,1\}$ and let $\ell'(\sigma) = |V(G)| - \ell(\sigma)$, i.e. the number of vertices mapped to “1” by $\sigma$. Define

$$Z'_{\gamma,\lambda}(G) = \sum_{\sigma:V(G)\to\{0,1\}} \gamma^{c(\sigma)}\lambda^{\ell'(\sigma)}.$$

Let $p$ be a prime and let $\lambda^{-1}$ be the inverse of $\lambda$ in $\mathbb{Z}_p$. From the definitions of $Z_{\gamma,\lambda}$, $Z'_{\gamma,\lambda}$ and $\ell'$ we have,

$$\begin{aligned}
(\lambda^{-1})^n Z_{\gamma,\lambda}(G) &\equiv (\lambda^{-1})^n \sum_{\sigma:V(G)\to\{0,1\}} \gamma^{c(\sigma)}\lambda^{\ell(\sigma)} && \pmod{p}\\
&\equiv \sum_{\sigma:V(G)\to\{0,1\}} \gamma^{c(\sigma)}(\lambda^{-1})^{|V(G)|-\ell(\sigma)} && \pmod{p}\\
&\equiv \sum_{\sigma:V(G)\to\{0,1\}} \gamma^{c(\sigma)}(\lambda^{-1})^{\ell'(\sigma)} && \pmod{p}\\
&\equiv Z'_{\gamma,\lambda^{-1}}(G). && \pmod{p}
\end{aligned}$$

That is for any multigraph $G$, if we can compute $Z_{\gamma,\lambda}(G)$ modulo a prime $p$, then we can compute $Z'_{\gamma,\lambda^{-1}}(G)$ modulo a prime $p$. So classifying the complexity of $\#_p Z_{\gamma,\lambda}$, for every prime $p$ and every $\lambda \in \mathbb{Z}_p$, is equivalent to determining the complexity of computing $Z'_{\gamma,\lambda}$ modulo $p$.

## 6.4 Polynomial solvable cases

As we have already mentioned in Section 3.3, Guo et al. [49] study the complexity of $\#_k$CSP and give a dichotomy theorem completely characterising its complexity. We will review the results of [49] and explain how they are relevant to identifying values of the parameters $\gamma$, $\lambda$ and $p$, for which $\#_p Z_{\gamma,\lambda}$ is computable in polynomial time.

Recall definition 3.5 of a Boolean binary function that is expressed by a matrix. To show the dichotomy for $\#_k$CSP Guo et al. define the classes of functions $\mathcal{P}_k$ and $\mathcal{A}_k$.

**Definition 6.3.** Let $\mathcal{U}$ be the set of Boolean unary functions $f : \{0,1\} \to \mathbb{N}$, let $F_1 : \{0,1\}^2 \to \{0,1\}$ be the Boolean binary function expressed by the matrix $\left[\begin{smallmatrix}1&0\\0&1\end{smallmatrix}\right]$ and

| $x$ | $y$ | $L_1(X)$ | $L_2(X)$ | $L_3(X)$ | $L_4(X)$ | $L_1(X)+3L_2(X)+3L_3(X)+L_4(X)$ |
|---|---|---|---|---|---|---|
| 0 | 0 | 1 | 1 | 1 | 1 | 8 |
| 1 | 0 | 0 | 0 | 1 | 1 | 4 |
| 0 | 1 | 0 | 1 | 0 | 1 | 4 |
| 1 | 1 | 1 | 0 | 0 | 1 | 2 |

Table 6.1: The table for the values of the indicator functions $L_1, \dots, L_4$ as they appear in the proof of Lemma 6.6.

let $F_2 : \{0,1\}^2 \to \{0,1\}$ be the Boolean binary function expressed by the matrix $\left[\begin{smallmatrix} 0 & 1 \\ 1 & 0 \end{smallmatrix}\right]$.

Let $\mathcal{P}$ contain functions of the form $\prod_i f_i(x_{i_1}, \dots, x_{i_{r_i}})$, where $f_i \in \mathcal{U} \cup \{F_1, F_2\}$. The functions of $\mathcal{P}$ are also called product functions. For an integer $k$, let $\mathcal{P}_k = \{f : \{0,1\}^r \to \mathbb{Z}_p \mid f \equiv g \pmod{k}$, where $g \in \mathcal{P}\}$.

**Definition 6.4.** Let $X$ be an $r+1$ dimensional column vector $(x_1, \dots, x_r, 1)$ of $\{0,1\}^{r+1}$. For an integer $r'$, let $A$ be a $r' \times (r+1)$ Boolean matrix and let

$$\chi_{AX} = \begin{cases} 1 & \text{if } AX = 0 \\ 0 & \text{otherwise.} \end{cases}$$

$\mathcal{A}_k$ contains the affine functions. That is, $\mathcal{A}_k$ is the class of functions of the form $\chi_{AX} \cdot (i_k)^{L_1(X)+\dots+L_m(X)}$, where for $j \in [m]$, $L_j(X)$ is an indicator function $\chi_{\langle \alpha_j, X \rangle}$, $\alpha_j$ is an $(r+1)$-dimensional vector and the inner product $\langle \alpha_j, X \rangle$ is taken over $\mathbb{Z}_2$.

We are now ready to state the dichotomy result of Guo et al. [49, Lemma 3.10] for $\#_p$CSP.

**Lemma 6.5** (Guo et al.)**.** *Let $p$ be an odd prime and let $\mathcal{F}$ be a class of functions mapping Boolean inputs to $\mathbb{Z}_p$. If $\mathcal{F} \subseteq \mathcal{P}_p$ or $\mathcal{F} \subseteq \mathcal{A}_p$, then $\#_p\mathrm{CSP}(\mathcal{F})$ is computable in polynomial time. Otherwise $\#_p\mathrm{CSP}(\mathcal{F})$ is $\#_p\mathsf{P}$-hard.*

Let $E_\gamma$ be the function expressed by the matrix $\left[\begin{smallmatrix} 1 & 1 \\ 1 & \gamma \end{smallmatrix}\right]$. Additionally let $U_\lambda$ to be the unary function with $U_\lambda(0) = \lambda$ and $U_\lambda(1) = 1$. When $\lambda = 1$, $\#_p Z_{\gamma,1}$ is equivalent to $\#_p\mathrm{CSP}(\{E_\gamma\})$. In any other case $\#_p Z_{\gamma,\lambda}$ is essentialy a restricted input of $\#_p\mathrm{CSP}(\{E_\gamma, U_\lambda\})$, where the unary function $U_\lambda$ is applied to every variable of the input exactly once. When $\#_p\mathrm{CSP}(\{E_\gamma, U_\lambda\})$ is computable in polynomial time then $\#_p Z_{\gamma,\lambda}$ is computable in polynomial time, as it is a special case of $\#_p\mathrm{CSP}(\{E_\gamma, U_\lambda\})$. So by Lemma 6.5 when $E_\gamma, U_\lambda \in \mathcal{P}_p$ or when $E_\gamma, U_\lambda \in \mathcal{A}_p$, we have that $\#_p Z_{\gamma,\lambda}$ is computable in polynomial time.

**Lemma 6.6.** *Let $p$ be a prime. The following hold.*

1. *For all $\gamma \in \mathbb{Z}_p$, $\#_p Z_{\gamma,0}$ is computable in polynomial time.*
2. *For all $\lambda \in \mathbb{Z}_p$, $\#_p Z_{1,\lambda}$ is computable in polynomial time.*

*3. For $\lambda \in \{0, \pm 1, \pm i_p\}$, $\#_p Z_{-1,\lambda}$ is computable in polynomial time.*

*Proof.* Let $G$ be a multigraph. When $\lambda \equiv 0 \pmod{p}$, $Z_{\gamma,0}(G) = \gamma^{|E(G)|}$, so (1) follows.

To show (2) notice that $E_1 \in \mathcal{P}_p$ as $E_1$ is a product of two unary functions: $E_1(x,y) = g(x)g(y)$ with $g(x) = 1$. For any $\lambda \in \mathbb{Z}_p$, $U_\lambda$ is a unary function, therefore a member of $\mathcal{P}_p$. From Lemma 6.5, $\#_p\mathrm{CSP}(\{E_1, U_\lambda\})$ is computable in polynomial time and so is $\#_p Z_{1,\lambda}$.

Now we will show (3). Recall Definition 6.4 of $\mathcal{A}_p$. Let $A = [0,0,0]$ and let $X = (x, y, 1)$, so $AX = (0)$ and, therefore, $\chi_{AX}(x,y) = 1$. Now let $\alpha_1 = (1,1,1)$, $\alpha_2 = (1,0,1)$, $\alpha_3 = (0,1,1)$ and $\alpha_4 = (0,0,1)$. So $L_1(x) \equiv x + y + 1 \pmod{2}$, $L_2(X) \equiv x + 1 \pmod{2}$, $L_3(X) \equiv y + 1 \pmod{2}$ and $L_4(X) = 1$. As we can see from Table 6.1 $E_{-1} = \chi_{AX} \cdot i_p^{L_1(X)+3L_2(X)+3L_3(X)+L_4(X)}$, so $E_{-1} \in \mathcal{A}_p$.

We claim that when $\lambda \in \{0, 1, i_p, -1, -i_p\}$, $U_\lambda \in \mathcal{A}_p$. Let $X' = (x, 1)$. Let $A_0 = [0,0]$ and let $\alpha_5 = (1,1)$. So $\chi_{A_0X'}(x) = 1$ and $L_5(X) \equiv x + 1 \pmod{2}$. Therefore, $U_{i_p} = \chi_{A_0X'} \cdot i_p^{L_5(X')}$, $U_{-1} = \chi_{A_0X'} \cdot i_p^{2L_5(X')}$, $U_{-i_p} = \chi_{A_0X'} \cdot i_p^{3L_5(X')}$ and $U_1 = \chi_{A_0X'} \cdot i_p^{4L_1(X')}$. From the above and Lemma 6.5, when $\lambda \in \{0, 1, i_p, -1, -i_p\}$, $\#_p\mathrm{CSP}(\{E_{-1}, U_\lambda\})$ is computable in polynomial time and so is $\#_p Z_{-1,\lambda}$. □

## 6.5 Hardness results

In this section we will show values of the parameter space $(\gamma, \lambda, p)$ for which $\#_p Z_{\gamma,\lambda}$ is hard to compute. Lemma 6.13 below shows that $\#_p Z_{\gamma,\lambda}$ is $\#_p\,\mathsf{P}$-hard for all primes $p$ when $\lambda \not\equiv 0 \pmod{p}$ and $\gamma \equiv 0 \pmod{p}$. Lemma 6.14 shows that when $\lambda \not\equiv 1 \pmod{p}$ and $\lambda = \gamma^k \pmod{p}$, $\#_p Z_{\gamma,\lambda}$ is $\#_p\,\mathsf{P}$-hard. Lemma 6.18 shows that, for all primes $p < 100$, where $p \neq 41$, when $\gamma^2 \not\equiv 1 \pmod{p}$ $\#_p Z_{\gamma,\lambda}$ is $\#_p\,\mathsf{P}$-complete.

When $\lambda \notin \{0, \pm 1, \pm i_p\}$ the complexity of $\#_p Z_{-1,\lambda}$ remains unknown. We have, by Lemma 6.5, that $\#_p\mathrm{CSP}(\{E_{-1}, U_\lambda\})$ is hard.[1] Since $\#_p Z_{-1,\lambda}$ is a restricted input for $\#_p\mathrm{CSP}(\{E_{-1}, U_\lambda\})$, hardness for $\#_p\mathrm{CSP}(\{E_{-1}, U_\lambda\})$ does not imply hardness for $\#_p Z_{-1,\lambda}$. $\#_p Z_{-1,1}$ is computable in polynomial time and a reduction similar to the reductions we present in this chapter would not give us hardness for $\#_p Z_{-1,\lambda}$.

### 6.5.1 Pinning in two-spin systems

The gadgets we will use for our reductions from $\#_p Z_\gamma$ to $\#_p Z_{\gamma,\lambda}$ will be multigraphs with vertices pinned to either "0" or "1". In the setting of $\#_p Z_{\gamma,\lambda}$, a *pinning function* $\tau$ of a multigraph $G$ is a partial function from $V(G)$ to $\{0,1\}$. A vertex $v$ in the domain of the pinning function is said to be *pinned* or *pinned to* $\tau(v)$. A *partially pinned multigraph* $J = (G, \tau)$ consists of an *underlying graph* $G$ and a pinning function $\tau$ from $V(G)$ to $\{0,1\}$.

[1] $E_{-1} \notin \mathcal{P}_p$ as it cannot be a multiple of $F_1$ or $F_2$ from Definition 6.3 and it cannot be a multiple of only unary functions. Also $U_\lambda \notin \mathcal{A}_p$ as the functions in $\mathcal{A}_p$ can only have values in $\{0, \pm 1, \pm i_p\}$.

**Definition 6.7.** Let $G$ be a multigraph and let $\tau : V(G) \to \{0,1\}$ be a pinning function of $G$. $R(G,\tau) = \{\sigma : V(G) \to \{0,1\} \mid \sigma(v) = \tau(v) \text{ for } v \in \mathrm{dom}(\tau)\}$ is the set of mappings that respect $\tau$.

Given a partially pinned multigraph $(G,\tau)$ we define $Z^{\tau}_{\gamma,\lambda}(G) = \sum_{\sigma \in R(G,\tau)} \gamma^{c(\sigma)}\lambda^{\ell(\sigma)}$. The pinned version of $\#_pZ_{\gamma,\lambda}$ is the following.

**Problem 6.8.** *Name.* $\#_p$PINNED$Z_{\gamma,\lambda}$.

*Parameter.* A prime $p$ and $\gamma, \lambda \in \mathbb{Z}_p$.

*Input.* A partially pinned multigraph $J = (G,\tau)$.

*Output.* $Z^{\tau}_{\gamma,\lambda}(G) \pmod p$.

The following lemma shows that computing $\#_pZ_{\gamma,\lambda}$ is at least as hard as computing $\#_p$PINNED$Z_{\gamma,\lambda}$.

**Lemma 6.9.** *For any prime $p$ $\#_p$PINNED$Z_{\gamma,\lambda}$ reduces to $\#_pZ_{\gamma,\lambda}$ under polynomial-time Turing reductions.*

*Proof.* Let $J = (G,\tau)$ be an instance of $\#_p$PINNED$Z_{\gamma,\lambda}$. Let $V_0 = \{v \in \mathrm{dom}(\tau) \mid \tau(v) = 0\}$ be the set of variables pinned to "0" and $V_1 = \mathrm{dom}(\tau) \setminus V_0$ the set of variables pinned to "1". $Z^{\tau}_{\gamma,\lambda}(G) = \lambda^{|V_0|}Z^{\tau}_{\gamma,\lambda}(G[V \setminus V_0])$ so it suffices to show that the lemma holds for instances with $V_0 = \emptyset$.

Let $E_1 = E(G[V_1])$ be the set of edges of $G$ between the vertices of $V_1$. Let $G'$ be the sub-multigraph of $G$ where all the edges in $E_1$ have been deleted. $Z^{\tau}_{\gamma,\lambda}(G) = \gamma^{|E_1|}Z^{\tau}_{\gamma,\lambda}(G')$, so it suffices to show that the lemma holds for inputs $J$ where $E_1 = \emptyset$.

Let $G''$ be the multigraph produced from $G$, by identifying all the vertices in $V_1$. Call this vertex $v_1$. By construction, $Z^{\tau}_{\gamma,\lambda}(G) = Z^{\tau}_{\gamma,\lambda}(G'')$. $Z_{\gamma,\lambda}(G'') = Z^{\tau}_{\gamma,\lambda}(G'') + \lambda Z_{\gamma,\lambda}(G'' - v_1)$ and the lemma follows. □

### 6.5.2 Gadgets for two-spin systems

In order to prove hardness for $\#_pZ_{\gamma,\lambda}$, we need to find gadgets which, by attaching to every vertex of $G$, will cancel out the weights of $\lambda$ that the vertices mapped to "0" contribute to the partition function. Our gadgets $(J,x)$ consist of a partially pinned multigraph $J = (G,\tau)$ with a distinguished vertex $x \in V(G(J))$. Recall Definition 6.7 of $R(G,\tau)$. To express the properties of the gadgets we will need the following definitions.

**Definition 6.10.** Let $(J,x)$ be a partially pinned multigraph with a distinguished vertex. Let $G = G(J)$ and let $\tau = \tau(J) : V(G(J)) \to \{0,1\}$ be the pinning function of $J$, such that $x \notin \mathrm{dom}(\tau)$. We define

$$Z^{\tau,0}_{\gamma,\lambda}(G,x) = \sum_{\substack{\sigma\in R(G,\tau),\\ \sigma(x)=0}} \gamma^{c(\sigma)}\lambda^{\ell(\sigma)} \quad \text{and}$$

$$Z^{\tau,1}_{\gamma,\lambda}(G,x) = \sum_{\substack{\sigma\in R(G,\tau),\\ \sigma(x)=1}} \gamma^{c(\sigma)}\lambda^{\ell(\sigma)}.$$

As in Theorem 3.10 we will use the following notation to build partially pinned multigraphs containing many copies of some subgraph. For any "tag" $T$ (which we will treat just as an arbitrary string) and any partially labelled multigraph $J$, denote by $J^T$ a copy of $J$ with every vertex $v \in V(G(J))$ renamed $v^T$.

**Definition 6.11.** Given a multigraph $G$ and a partially pinned multigraph with a distinguished vertex $(J,x)$ we define $G \odot (J,x)$ to be the partially pinned multigraph constructed in the following way. Take a copy of $G$ together with the union of disjoint copies $G^v(J)$ of $G(J)$ for each $v \in V(G)$. For each vertex $v \in V(G)$ identify $v$ with $x^v$. Let $\tau(G \odot (J,x)) = \bigcup_{v\in V(G)} \tau(G^v(J))$.

The following theorem shows that if we find an appropriate partially pinned multigraph with a distinguished vertex $(J,x)$, we can obtain hardness for $\#_p Z_{\gamma,\lambda}$.

**Lemma 6.12.** *Let $p$ be a prime and let $\gamma,\lambda \in \mathbb{Z}_p$, with $\gamma^2 \not\equiv 1 \pmod p$ and $\lambda \not\equiv 0 \pmod p$. If there exists a partially pinned multigraph with a distinguished vertex $(J,x)$ and $\tau = \tau(J)$, such that $Z^{\tau,0}_{\gamma,\lambda}(G(J),x) \equiv Z^{\tau,1}_{\gamma,\lambda}(G(J),x) \not\equiv 0 \pmod p$, then $\#_p Z_{\gamma,\lambda}$ is $\#_p\mathsf{P}$-hard.*

*Proof.* We first show that $\#_p Z_{\gamma,1}$ reduces to $\#_p\textsc{Pinned}Z_{\gamma,\lambda}$. Let $G$ be the input of $\#_p Z_{\gamma,1}$ and let $(J,x)$ be a partially pinned multigraph with a distinguished vertex, such that $Z^{\tau,0}_{\gamma,\lambda}(G(J),x) \equiv Z^{\tau,1}_{\gamma,\lambda}(G(J),x) \equiv A \pmod p$. Let $J' = G \odot (J,x)$ be the partially pinned graph, as in Definition 6.11. $J'$ will be the input for $\#_p\textsc{Pinned}Z_{\gamma,\lambda}$. We denote $G' = G(J')$ and $\tau' = \tau(J')$.

For $\sigma,\sigma' \in R(G',\tau')$ we write $\sigma \sim_G \sigma'$ if $\sigma|_G = \sigma'|_G$. We also write $[\![\sigma]\!]_G$ for the $\sim_G$-equivalence class of $\sigma$. The classes $[\![\sigma]\!]_G$ partition the set of configurations of $J'$. For $\sigma \in R(G',\tau')$ we define $W([\![\sigma]\!]_G) = \sum_{\sigma\in[\![\sigma]\!]_G} \gamma^{c(\sigma)}\lambda^{\ell(\sigma)}$. Let $\sigma \in R(G',\tau')$, be a configuration of $G'$ that respects $\tau'$. $\sigma$ maps $\ell(\sigma|_G)$ vertices of $G$ to "0" and $V(G) - \ell(\sigma|_G)$ vertices of $G$ to "1". Therefore,

$$W([\![\sigma]\!]_G) = \gamma^{c(\sigma|_G)} \left( \sum_{\substack{\sigma'\in R(G(J),\tau(J)),\\ \sigma'(x)=0}} \gamma^{c(\sigma')}\lambda^{\ell(\sigma')} \right)^{\ell(\sigma|_G)} \left( \sum_{\substack{\sigma'\in R(G(J),\tau(J)),\\ \sigma'(x)=1}} \gamma^{c(\sigma')}\lambda^{\ell(\sigma')} \right)^{V(G)-\ell(\sigma|_G)}.$$

From Definition 6.10 we now have,

$$W(\llbracket\sigma\rrbracket_G) = \gamma^{c(\sigma|_G)}\left(Z^{\tau,0}_{\gamma,\lambda}(G(J),x)\right)^{\ell(\sigma|_G)}\left(Z^{\tau,1}_{\gamma,\lambda}(G(J),x)\right)^{V(G)-\ell(\sigma|_G)}.$$

By the choice of $J$, $Z^{\tau,0}_{\gamma,\lambda}(G(J),x) \equiv Z^{\tau,1}_{\gamma,\lambda}(G(J),x) \equiv A \pmod{p}$, so

$$W(\llbracket\sigma\rrbracket_G) \equiv \gamma^{c(\sigma|_G)}A^{|V(G)|} \pmod{p}. \tag{6.1}$$

Now let $\sigma_1 \dots \sigma_k$ be representatives of the $\sim_G$-equivalence classes.

$$\begin{aligned}
Z^{\tau}_{\gamma,\lambda}(G') &= \sum_{\sigma\in R(G',\tau')} \gamma^{c(\sigma)}\lambda^{\ell(\sigma)} \\
&= \sum_{i=1}^{k}\sum_{\sigma\in\llbracket\sigma_i\rrbracket_G} \gamma^{c(\sigma)}\lambda^{\ell(\sigma)} \\
&= \sum_{i=1}^{k} W(\llbracket\sigma_i\rrbracket_G) \\
&\equiv A^{|V(G)|}\sum_{i=1}^{k}\gamma^{c(\sigma_i|_G)} \pmod{p} \\
&= A^{|V(G)|}Z_{\gamma,1}(G).
\end{aligned}$$

The equivalence above comes from Equation 6.1.

We have shown that $\#_pZ_{\gamma,1}$ reduces to $\#_p\textsc{Pinned}Z_{\gamma,\lambda}$. $\#_pZ_{\gamma,1} = \#_pZ_\gamma$ and by Corollary 3.8 $\#_pZ_\gamma$ is $\#_p\,\mathsf{P}$-hard. By Lemma 6.9, $\#_p\textsc{Pinned}Z_{\gamma,\lambda}$ reduces to $\#_pZ_{\gamma,\lambda}$. □

### 6.5.3 Finding gadgets

In this section we show values of the parameter space $(\gamma,\lambda,p)$ for which $\#_pZ_{\gamma,\lambda}$ is $\#_p\,\mathsf{P}$-hard. All the proofs of lemmas in this section essentially identify a partially pinned multigraph $(J,x)$ with a distinguished vertex so that Lemma 6.12 applies. First we show that when $\gamma \equiv 0$ and $\lambda \not\equiv 0$ $\#_pZ_{\gamma,\lambda}$ is hard. The gadget is a clique of appropriate size.

**Lemma 6.13.** *For every prime $p$ and $\lambda \not\equiv 0 \pmod{p}$, $\#_pZ_{0,\lambda}$ is $\#_p\,\mathsf{P}$-hard.*

*Proof.* Let $p$ prime and $\lambda \not\equiv 0 \pmod{p}$. Define $(J,x)$ to be the following partially pinned multigraph with a distinguished vertex. $G(J)$ is the clique $K_k$ with $k > 0$ and $k \equiv p+2-\lambda \pmod{p}$ vertices, where $x \in V(G(J))$ and $\tau = \tau(J) = \emptyset$.

$Z^{\tau,0}_{\gamma,\lambda}(G(J),x) = \lambda^{k-1}(\lambda+k-1)$ and $Z^{\tau,1}_{\gamma,\lambda}(G(J),x) = \lambda^{k-1}$. If we chose $k = p+2-\lambda$, then

$$\begin{aligned}
Z^{\tau,0}_{\gamma,\lambda}(G(J),x) &= \lambda^{k-1}(\lambda+p+2-\lambda-1) \\
&\equiv \lambda^{k-1} && \pmod{p} \\
&\equiv Z^{\tau,1}_{\gamma,\lambda}(G(J),x). && \pmod{p}
\end{aligned}$$

Since $\lambda$ is non-zero and not a multiple of $p$, $\lambda^{k-1}$ is also non-zero and not a multiple of $p$. $Z^{\tau,1}_{\gamma,\lambda}(G(J),x) \not\equiv 0 \pmod p$ and the proof follows from Lemma 6.12. □

The next lemma shows that if $\gamma$ is a generator in the multiplicative group $\mathbb{Z}_p^*$ or if $\gamma$ is in the same orbit as $\lambda$ in the multiplicative group $\mathbb{Z}_p^*$ the $\#_pZ_{\gamma,\lambda}$ is hard. The gadget that we use in order to derive hardness is a pair of vertices connected with the appropriate number of parallel edges, where one of the vertices is pinned to "1".

**Lemma 6.14.** *Let $p$ be a prime and let $\gamma, \lambda \in \mathbb{Z}_p$, with $\gamma^2 \not\equiv 1 \pmod p$ and $\lambda \not\equiv 0 \pmod p$. If there exists $k \in \mathbb{N}$ such that $\gamma^k \equiv \lambda \pmod p$, then $\#_pZ_{\gamma,\lambda}$ is $\#_p\mathsf{P}$-hard.*

*Proof.* Let $(J,x)$ be the partially pinned multigraph with two vertices $x$ and $y$ and $k$ parallel edges $(x,y)$. Let $\tau = \tau(J) = \{y \mapsto 1\}$. $Z^{\tau,0}_{\gamma,\lambda}(J,x) = \lambda$ and $Z^{\tau,1}_{\gamma,\lambda}(J,x) = \gamma^k$. Since $\gamma^k \equiv \lambda \not\equiv 0 \pmod p$, the proof follows from Lemma 6.12. □

Observe that for any partially pinned multigraph with a distinguished vertex $(J,x)$ with a pinning function $\tau = \tau(J)$, where $x \notin \operatorname{dom}(\tau)$, $Z^{\tau,0}_{\gamma,\lambda}(G(J),x)$ is a multiple of $\lambda$. That is because every term of $\sum_{\sigma \in R(J,\tau)}^{\sigma(x)=0} \gamma^{c(\sigma)}\lambda^{\ell(\sigma)}$ is a multiple of $\lambda$ since $x$ is always mapped to "0". We have the following observation.

**Observation 6.15.** Let $(J_1,x_1)\dots(J_k,x_k)$ be a set of partially pinned multigraphs with distinguished vertices and let for $i \in [k]$, $\tau_i = \tau(J_i)$. Define $(J,x)$ to be the graph constructed by identifying all vertices $x_i$ and let $\tau = \bigcup_{i\in[k]} \tau_i$. If $Z^{\tau_i,0}_{\gamma,\lambda}(J_i,x_i) = \lambda A_i$ and $Z^{\tau_i,1}_{\gamma,\lambda}(J_i,x_i) = B_i$ then $Z^{\tau,0}_{\gamma,\lambda}(J,x) = \lambda\prod_{i\in[k]} A_i$ and $Z^{\tau,1}_{\gamma,\lambda}(J,x) = \prod_{i\in[k]} B_i$.

Some of the graphs we consider as gadgets are cliques. For the clique $K_k$, $Z_{\gamma,\lambda}(K_k) = \sum_{i=0}^{k}\binom{k}{i}\lambda^i\gamma^{\binom{k-i}{2}}$. This is true because there are $\binom{k}{i}$ mappings $\sigma$ with $\ell(\sigma) = i$. For every such mapping $\sigma$, $k-i$ vertices are mapped to "1", hence $c(\sigma) = \binom{k-i}{2}$. Based on the latter, we have the following observation.

**Observation 6.16.** Let $(J,x)$ be a partially pinned graph, where $G(J) = K_{k+1}$ is the clique with $k+1$ vertices, one of which is $x$ and let $\tau = \tau(J) = \emptyset$. Then,

$$Z^{\tau,0}_{\gamma,\lambda}(G(J),x) = \lambda\sum_{i=0}^{k}\binom{k}{i}\lambda^i\gamma^{\binom{k-i}{2}}$$

and

$$Z^{\tau,1}_{\gamma,\lambda}(G(J),x) = \sum_{i=0}^{k}\binom{k}{i}\lambda^i\gamma^{\binom{k-i}{2}}\gamma^{k-i}.$$

The other gadgets we will consider are paths.

**Observation 6.17.** Let $P_2 = xx_1x_2$ be the path of length 2 and let $P_3 = xx_1x_2x_3$ be the path of length 3. We have,

$$\begin{aligned}
Z^{\tau,0}_{\gamma,\lambda}(P_2, x) &= \lambda(\lambda^2 + 2\lambda + \gamma)\\
Z^{\tau,1}_{\gamma,\lambda}(P_2, x) &= \lambda^2 + \lambda\gamma + \lambda + \gamma^2\\
Z^{\tau,0}_{\gamma,\lambda}(P_3, x) &= \lambda(\lambda^3 + 3\lambda^2 + 2\lambda\gamma + \lambda + \gamma^2)\\
Z^{\tau,1}_{\gamma,\lambda}(P_3, x) &= \lambda^3 + 2\lambda^2 + \lambda^2\gamma + 2\lambda\gamma + \lambda\gamma^2 + \gamma^3.
\end{aligned}$$

Using the previous observations we can produce a computer assisted proof identifying the values of the parameters $\gamma$, $\lambda$ and $p$ that make $\#_pZ_{\gamma,\lambda}$ $\#_p\mathsf{P}$-hard, when $p$ is small.

**Lemma 6.18.** *For all primes $p < 100$, $p \neq 41$ and all $\gamma, \lambda \in \mathbb{Z}_p$ with $\gamma^2 \not\equiv 1 \pmod p$ and $\lambda \not\equiv 0 \pmod p$, $\#_pZ_{\gamma,\lambda}$ is $\#_p\mathsf{P}$-hard.*

*Proof.* Let $\mathbf{k} = (k_0, k_1 \dots, k_m)$ be a vector of $(\mathbb{Z}_p)^{m+1}$ for $m \in \mathbb{N}_{>0}$. Let $(J(\mathbf{k}), x)$ be the partially pinned multigraph with a distinguished vertex $x$. $G(J(\mathbf{k}))$ is constructed by taking the union of disjoint copies of the following multigraphs and identifying their distinguished vertices.

- The multigraph with vertex set $\{x, y\}$ and $k_0$ parallel edges $(x, y)$, where $x$ is the distinguished vertex.
- For each $j \in [m-2]$, $k_j$ copies of the clique $K_{j+1}$, where $x_j \in V(K_{j+1})$ is the distinguished vertex.
- $k_{m-1}$ copies of the path $P_2 = xx_1x_2$, where $x$ is the distinguished vertex.
- $k_m$ copies of the path $P_3 = xx_1x_2x_3$, where $x$ is the distinguished vertex.

Let $\tau(J(\mathbf{k})) = \tau = \{y \mapsto 1\}$.

From Observations 6.15, 6.16, 6.17 and the proof of Lemma 6.14 we have,

$$Z^{\tau,0}_{\gamma,\lambda}(G(J(\mathbf{k})), x) = \lambda\left(\prod_{j=1}^{m-2}\Big(\sum_{i=0}^{j}\binom{j}{i}\lambda^i\gamma^{\binom{j-i}{2}}\Big)^{k_j}\right)\Big(Z^{\tau,0}_{\gamma,\lambda}(P_2, x)/\lambda\Big)^{k_{m-1}}\Big(Z^{\tau,0}_{\gamma,\lambda}(P_3, x)/\lambda\Big)^{k_m}$$

and

$$Z^{\tau,1}_{\gamma,\lambda}(G(J(\mathbf{k})), x) = (\gamma)^{k_0}\left(\prod_{j=1}^{m-2}\Big(\sum_{i=0}^{j}\binom{j}{i}\lambda^i\gamma^{\binom{j-i}{2}}\gamma^{j-i}\Big)^{k_j}\right)\Big(Z^{\tau,1}_{\gamma,\lambda}(P_2, x)\Big)^{k_{m-1}}\Big(Z^{\tau,1}_{\gamma,\lambda}(P_3, x)\Big)^{k_m}.$$

Using the program presented in the Appendix, we can show that for all primes $p < 100$, with $p \neq 41$ and for all $\gamma, \lambda \in \mathbb{Z}_p$, where $\gamma^2 \not\equiv 1 \pmod p$ and $\lambda \not\equiv 0 \pmod p$, there exists $\mathbf{k} \in (\mathbb{Z}_p)^{m+1}$ for $m \in \mathbb{N}_{>0}$, such that $Z^{\tau,0}_{\gamma,\lambda}(J(\mathbf{k}), x) \equiv Z^{\tau,1}_{\gamma,\lambda}(J(\mathbf{k}), x) \not\equiv 0 \pmod p$. The lemma follows from Lemma 6.12. □

The program we used in the above proof, only failed to find a gadget for $p = 41, \gamma = 18$, $\lambda = 6$. We conjecture that there exists a gadget for these parameter values, so for all $\gamma, \lambda \in \mathbb{Z}_{41}$ with $\gamma^2 \not\equiv 1 \pmod{p}$ and $\lambda \not\equiv 0 \pmod{p}$, $\#_{41}Z_{\gamma,\lambda}$ is $\#_p\mathsf{P}$-hard.

## 6.6 Summary of results

Now we can summarise the results of this section. We will use the following.

**Lemma 6.19.** *Let $p$ be a prime and let $\gamma \in \mathbb{Z}_p$. $\gamma^2 \equiv 1 \pmod{p}$ if and only if $\gamma = \pm 1 \pmod{p}$.*

*Proof.* If $\gamma \equiv \pm 1 \pmod{p}$ then, trivially $\gamma^2 \equiv 1 \pmod{p}$. Now assume that $\gamma^2 \equiv 1 \pmod{p}$ we have,

$$\begin{aligned} \gamma^2 &\equiv 1 \pmod{p} \\ (\gamma - 1)(\gamma + 1) &\equiv 0 \pmod{p}. \end{aligned}$$

The latter equation is only true when $\gamma \equiv \pm 1 \pmod{p}$, so the lemma follows. □

The main theorem of this section summarises our results on the complexity of $\#_pZ_{\gamma,\lambda}$.

**Theorem 1.13.** *Let $p$ be a prime and let $\gamma, \lambda \in \mathbb{Z}_p$. $\#_pZ_{\gamma,\lambda}$ is computable in polynomial time when one of the following holds.*

1. $\lambda = 0$.
2. $\gamma = 1$.
3. $\gamma = -1$ *and* $\lambda \in \{0, \pm 1, \pm i_p\}$.

*Furthermore, $\#_pZ_{\gamma,\lambda}$ is $\#_p\mathsf{P}$-complete when one of the following holds.*

4. $\lambda \not\equiv 0 \pmod{p}$ *and* $\gamma \equiv 0 \pmod{p}$.
5. $\lambda \not\equiv 0 \pmod{p}$, $\gamma \not\equiv \pm 1 \pmod{p}$ *and there exists an integer $k$ with* $\gamma^k \equiv \lambda \pmod{p}$.
6. $\lambda \not\equiv 0 \pmod{p}$, $\gamma \not\equiv \pm 1 \pmod{p}$ *and* $p < 100$, *where* $p \neq 41$.

*Proof.* By Lemma 6.6, $\#_pZ_{\gamma,\lambda}$ is computable in polynomial time when (1)-(3) hold. By Lemma 6.13 $\#_pZ_{\gamma,\lambda}$ is $\#_p\mathsf{P}$-hard when (4) holds. By Lemma 6.19 and Lemma 6.14 $\#_pZ_{\gamma,\lambda}$ is hard when (5) holds. Finally, by Lemma 6.19 and Lemma 6.18 we have that $\#_pZ_{\gamma,\lambda}$is hard when (6) holds. □

# Part II

# Counting problems

# Chapter 7

# Counting list matrix partitions of graphs

This chapter are published in the paper "Counting List Matrix Partitions of Graphs" co-authored with Leslie Goldberg, Colin McQuillan, Tomoyuki Yamakami and David Richerby [42].

## 7.1 Introduction

Recall from Section 1.3 of the introduction, that a matrix partition of an undirected graph is a partition of its vertices according to a matrix which specifies adjacency and non-adjacency conditions on the vertices, depending on the parts to which they are assigned. For finite sets $D$ and $D'$, the set $\{0,1,*\}^{D\times D'}$ is the set of $|D| \times |D'|$ matrices $M$ with rows indexed by $D$ and columns indexed by $D'$ where each $M_{i,j} \in \{0,1,*\}$. For any symmetric matrix $M \in \{0,1,*\}^{D\times D}$, an *$M$-partition* of an undirected graph $G = (V,E)$ is a function $\sigma\colon V \to D$ such that, for distinct vertices $u$ and $v$,

1. $M_{\sigma(u),\sigma(v)} \neq 0$ if $(u,v) \in E$ and
2. $M_{\sigma(u),\sigma(v)} \neq 1$ if $(u,v) \notin E$.

Also recall that a list $M$-partition is an $M$-partition that respects a given list function. In this chapter we study the complexity of the following problem.

**Problem 1.16.** *Name.* #LIST-$M$-PARTITIONS.

*Parameter.* A symmetric matrix $M \in \{0,1,*\}^{D\times D}$.

*Input.* A pair $(G,L)$ in which $G$ is a graph and $L$ is a function $V(G) \to \mathcal{P}(D)$.

*Output.* The number of $M$-partitions of $G$ that respect $L$.

Our main result is a dichotomy theorem for the counting list $M$-partition problem, where $M$ a symmetric matrix of arbitrary size.

**Theorem 1.17.** *For any symmetric matrix $M \in \{0,1,*\}^{D\times D}$, the problem* #List-$M$-partitions *is either in* $\mathsf{FP}$ *or* $\#\mathsf{P}$*-complete.*

To prove Theorem 1.17, we investigate the complexity of the more general counting problem $\#\mathcal{L}$-$M$-partitions, which has two parameters — a matrix $M \in \{0,1,*\}^{D\times D}$ and a (not necessarily proper) subset $\mathcal{L}$ of $\mathcal{P}(D)$. In this problem, we only allow sets in $\mathcal{L}$ to be used as lists.

**Problem 7.3.** *Name.* $\#\mathcal{L}$-$M$-partitions.

*Parameter.* A symmetric matrix $M \in \{0,1,*\}^{D\times D}$ and a subset $\mathcal{L}$ of $\mathcal{P}(D)$.

*Input.* A pair $(G, L)$ where $G$ is a graph and $L$ is a function $V(G) \to \mathcal{L}$.

*Output.* The number of $M$-partitions of $G$ that respect $L$.

Note that $M$ and $\mathcal{L}$ are fixed parameters of $\#\mathcal{L}$-$M$-partitions — they are not part of the input instance. The problem #List-$M$-partitions is just the special case of $\#\mathcal{L}$-$M$-partitions where $\mathcal{L} = \mathcal{P}(D)$.

We say that a set $\mathcal{L} \subseteq \mathcal{P}(D)$ is *subset-closed* if $A \in \mathcal{L}$ implies that every subset of $A$ is in $\mathcal{L}$. This closure property is referred to as the "inclusive" case in [34].

**Definition 7.4.** Given a set $\mathcal{L} \subseteq \mathcal{P}(D)$, we write $\mathcal{S}(\mathcal{L})$ for its subset-closure, which is the set $\mathcal{S}(\mathcal{L}) = \{X \mid \text{for some } Y \in \mathcal{L},\ X \subseteq Y\}$.

We prove the following theorem, which immediately implies Theorem 1.17.

**Theorem 7.5.** *Let $M$ be a symmetric matrix in $\{0,1,*\}^{D\times D}$ and let $\mathcal{L} \subseteq \mathcal{P}(D)$ be subset-closed. The problem $\#\mathcal{L}$-$M$-partitions is either in* $\mathsf{FP}$ *or* $\#\mathsf{P}$*-complete.*

Note that this does not imply a dichotomy for the counting $M$-partitions problem without lists. The problem with no lists corresponds to the case where every vertex of the input graph $G$ is assigned the list $D$, allowing the vertex to be potentially placed in any part. Thus, the problem without lists is equivalent to the problem $\#\mathcal{L}$-$M$-partitions with $\mathcal{L} = \{D\}$, but Theorem 7.5 applies only to the case where $\mathcal{L}$ is subset-closed.

### 7.1.1 Polynomial-time algorithms and an explicit dichotomy

We now introduce the concepts needed to give an explicit criterion for the dichotomy in Theorem 7.5 and to provide polynomial-time algorithms for all tractable cases. We use standard definitions of relations and their arities, compositions and inverses.

**Definition 7.6.** For any symmetric $M \in \{0,1,*\}^{D\times D}$ and any sets $X, Y \in \mathcal{P}(D)$, define the binary relation

$$H^M_{X,Y} = \{(i,j) \in X \times Y \mid M_{i,j} = *\}.$$

The intractability condition for the problem #$\mathcal{L}$-$M$-PARTITIONS begins with the following notion of rectangularity, which was introduced by Bulatov and Dalmau [11].

**Definition 7.7.** A relation $R \subseteq D \times D'$ is *rectangular* if, for all $i, j \in D$, and $i', j' \in D'$,

$$(i,i'), (i,j'), (j,i') \in R \implies (j,j') \in R\,.$$

Note that the intersection of two rectangular relations is itself rectangular. However, the composition of two rectangular relations is not necessarily rectangular: for example, $\{(1,1),(1,2),(3,3)\} \circ \{(1,1),(2,3),(3,1)\} = \{(1,1),(1,3),(3,1)\}$.

Our dichotomy criterion will be based on what we call $\mathcal{L}$-$M$-derectangularising sequences. In order to define these, we introduce the notions of pure matrices and $M$-purifying sets.

**Definition 7.8.** Given index sets $X$ and $Y$, a matrix $M \in \{0,1,*\}^{X\times Y}$ is *pure* if it has no 0s or has no 1s.

Pure matrices correspond to ordinary graph homomorphism problems. As we noted above, $M$-partitions of $G$ correspond to homomorphisms of $G$ when $G$ is a $\{0,*\}$-matrix. The same is true (by complementation) when $G$ is a $\{1,*\}$-matrix.

**Definition 7.9.** For any $M \in \{0,1,*\}^{D\times D}$, a set $\mathcal{L} \subseteq \mathcal{P}(D)$ is *$M$-purifying* if, for all $X, Y \in \mathcal{L}$, the $X$-by-$Y$ submatrix $M|_{X\times Y}$ is pure.

For example, consider the matrix

$$M = \begin{bmatrix} 1 & * & 0 \\ * & 1 & * \\ 0 & * & 1 \end{bmatrix},$$

with rows and columns indexed by $\{0,1,2\}$ in the obvious way. The matrix $M$ is not pure but for $\mathcal{L} = \{\{0,1\},\{2\}\}$, the set $\mathcal{L}$ is $M$-purifying and so is the closure $\mathcal{S}(\mathcal{L})$.

**Definition 7.10.** An *$\mathcal{L}$-$M$-derectangularising sequence* of length $k$ is a sequence $D_1, \dots, D_k$ with each $D_i \in \mathcal{L}$ such that:

1. $\{D_1, \dots, D_k\}$ is $M$-purifying and
2. the relation $H^M_{D_1,D_2} \circ H^M_{D_2,D_3} \circ \cdots \circ H^M_{D_{k-1},D_k}$ is not rectangular.

If there is an $i \in \{1,\dots,k\}$ such that $D_i$ is the empty set then the relation $H = H^M_{D_1,D_2} \circ H^M_{D_2,D_3} \circ \cdots \circ H^M_{D_{k-1},D_k}$ is the empty relation, which is trivially rectangular. If there is an $i$ such that $|D_i| = 1$ then $H$ is a Cartesian product, and is therefore rectangular. It follows that $|D_i| \geq 2$ for each $i$ in a derectangularising sequence.

We can now state our explicit dichotomy theorem, which implies Theorem 7.5 and, hence, Theorem 1.17.

**Theorem 7.11.** *Let $M$ be a symmetric matrix in $\{0,1,*\}^{D\times D}$ and let $\mathcal{L}\subseteq\mathcal{P}(D)$ be subset-closed. If there is an $\mathcal{L}$-$M$-derectangularising sequence, then the problem $\#\mathcal{L}$-$M$-*PARTITIONS *is* #P*-complete. Otherwise, it is in* FP.

### 7.1.2 Organisation and outline of the proof

Sections 7.3, 7.4 and 7.5 develop a polynomial-time algorithm which solves the problem $\#\mathcal{L}$-$M$-PARTITIONS whenever there is no $\mathcal{L}$-$M$-derectangularising sequence. The algorithm involves several steps.

First, consider the case in which $\mathcal{L}$ is subset-closed and $M$-purifying. In this case, Proposition 7.20 presents a polynomial-time transformation from an instance of the problem $\#\mathcal{L}$-$M$-PARTITIONS to an instance of a related counting CSP. Algorithm 3 exploits special properties of the constructed CSP instance so that it can be solved in polynomial time using a CSP technique called arc-consistency. (This is proved in Lemma 7.23.) This provides a solution to the original $\#\mathcal{L}$-$M$-PARTITIONS problem for the $M$-purifying case.

The case in which $\mathcal{L}$ is not $M$-purifying is tackled in Section 7.5. Section 7.5.1 gives algorithms for constructing the relevant data structures, which include a special case of sparse-dense partitions and also subcube decompositions. Algorithm 9 uses these data structures (via Algorithms 4, 5, 6, 7 and 8) to reduce the $\#\mathcal{L}$-$M$-PARTITIONS problem to a sequence of problems $\#\mathcal{L}_i$-$M$-PARTITIONS where $\mathcal{L}_i$ is $M$-purifying. Finally, the polynomial-time algorithm is presented in Algorithms 10 and 11. For every $\mathcal{L}$ and $M$ where there is no $\mathcal{L}$-$M$-derectangularising sequence, either Algorithm 10 or Algorithm 11 defines a polynomial-time function $\#\mathcal{L}$-$M$-PARTITIONS for solving the $\#\mathcal{L}$-$M$-PARTITIONS problem, given an input $(G, L)$. The function $\#\mathcal{L}$-$M$-PARTITIONS is not recursive. However, its *definition* is recursive in the sense that the function $\#\mathcal{L}$-$M$-PARTITIONS defined in Algorithm 11 calls a function $\#\mathcal{L}_i$-$M$-PARTITIONS where $\mathcal{L}_i$ is a subset of $\mathcal{P}(D)$ whose cardinality is smaller than $\mathcal{L}$. The function $\#\mathcal{L}_i$-$M$-PARTITIONS is, in turn, defined either in Algorithm 10 or in 11.

The proof of Theorem 7.11 shows that, when Algorithms 10 and 11 fail to solve the problem $\#\mathcal{L}$-$M$-PARTITIONS, the problem is #P-complete.

### 7.1.3 Complexity of the dichotomy criterion

Theorem 7.11 gives a precise criterion under which the problem $\#\mathcal{L}$-$M$-PARTITIONS is in FP or #P-complete, where $\mathcal{L}$ and $M$ are considered to be fixed parameters. In Section 7.6, we address the computational problem of determining which is the case, now treating $\mathcal{L}$ and $M$ as inputs to this "meta-problem". Dyer and Richerby [25] studied the corresponding problem for the #CSP dichotomy, showing that determining whether a constraint language $\Gamma$ satisfies the criterion for their #CSP($\Gamma$) dichotomy is reducible to the graph automorphism problem, which is in NP. We are interested in the following computational problem, which we show to be NP-complete.

**Problem 7.12.** *Name.* ExistsDerectSeq.

*Input.* An index set $D$, a symmetric matrix $M$ in $\{0,1,*\}^{D\times D}$ (represented as an array) and a set $\mathcal{L} \subseteq \mathcal{P}(D)$ (represented as a list of lists).

*Output.* "Yes", if there is an $\mathcal{S}(\mathcal{L})$-$M$-derectangularising sequence; "no", otherwise.

**Theorem 7.13.** ExistsDerectSeq *is* NP*-complete under polynomial-time many-one reductions.*

Note that, in the definition of the problem ExistsDerectSeq, the input $\mathcal{L}$ is not necessarily subset-closed. Subset-closedness allows a concise representation of some inputs: for example, $\mathcal{P}(D)$ has exponential size but it can be represented as $\mathcal{S}(\{D\})$, so the corresponding input is just $\mathcal{L} = \{D\}$. In fact, our proof of Theorem 7.13 uses a set of lists $\mathcal{L}$ where $|X| \leq 3$ for all $X \in \mathcal{L}$. Since there are at most $|D|^3 + 1$ such sets, our NP-completeness proof would still hold if we insisted that the input $\mathcal{L}$ to ExistsDerectSeq must be subset-closed.

Let us return to the original problem #List-$M$-partitions, which is the special case of the problem #$\mathcal{L}$-$M$-partitions where $\mathcal{L} = \mathcal{P}(D)$. This leads our interest to the following computational problem.

**Problem 7.14.** *Name.* MatrixHasDerectSeq.

*Input.* An index set $D$ and a symmetric matrix $M$ in $\{0,1,*\}^{D\times D}$ (represented as an array).

*Output.* "Yes", if there is a $\mathcal{P}(D)$-$M$-derectangularising sequence; "no", otherwise.

Theorem 7.13 does not quantify the complexity of MatrixHasDerectSeq because its proof relies on a specific choice of $\mathcal{L}$ which, as we have noted, is not $\mathcal{P}(D)$. Nevertheless, the proof of Theorem 7.13 has the following corollary.

**Corollary 7.15.** MatrixHasDerectSeq *is in* NP.

### 7.1.4 Cardinality constraints

Many combinatorial structures can be represented as $M$-partitions with the addition of cardinality constraints on the parts. For example, it might be required that certain parts be non-empty or, more generally, that they contain at least $k$ vertices for some fixed $k$.

Feder et al. [37] showed that the problem of determining whether such a structure exists in a given graph can be reduced to a List-$M$-partitions problem in which the cardinality constraints are expressed using lists. In Section 7.7, we extend this to counting. We show that any #$M$-PARTITIONS problem with additional cardinality constraints of the form, "part $d$ must contain at least $k_d$ vertices" is polynomial-time Turing reducible to #LIST-$M$-PARTITIONS. As a corollary, we show that the "homogeneous pairs" introduced by Chvátal and Sbihi [20] can be counted in polynomial time. Homogeneous pairs can be expressed as an $M$-partitions problem for a certain $6 \times 6$ matrix, with cardinality constraints on the parts.

## 7.2 Preliminaries

For a positive integer $k$, we write $[k]$ for the set $\{1, \ldots, k\}$. If $\mathcal{S}$ is a set of sets then we use $\bigcap \mathcal{S}$ to denote the intersection of all sets in $\mathcal{S}$. The vertex set of a graph $G$ is denoted $V(G)$ and its edge set is $E(G)$. We write $\{0, 1, *\}^D$ for the set of all functions $\sigma \colon D \to \{0, 1, *\}$ and $\{0, 1, *\}^{D \times D'}$ for the set of all matrices $M = (M_{i,j})_{i \in D, j \in D'}$, where each $M_{i,j} \in \{0, 1, *\}$.

We always use the term "$M$-partition" when talking about a partition of the vertices of a graph according to a $\{0, 1, *\}$-matrix $M$. When we use the term "partition" without referring to a matrix, we mean it in the conventional sense of partitioning a set $X$ into disjoint subsets $X_1, \ldots, X_k$ with $X_1 \cup \cdots \cup X_k = X$.

## 7.3 Counting list $M$-partition problems and counting CSPs

Toward the development of our algorithms and the proof of our dichotomy, we study a special case of the problem #$\mathcal{L}$-$M$-PARTITIONS, in which $\mathcal{L}$ is $M$-purifying and subset-closed. For such $\mathcal{L}$ and $M$, we show that the problem #$\mathcal{L}$-$M$-PARTITIONS is polynomial-time Turing-equivalent to a counting constraint satisfaction problem (#CSP). To give the equivalence, we introduce the notation needed to specify #CSPs.

A *constraint language* is a finite set $\Gamma$ of named relations over some set $D$. For such a language, we define the counting problem #CSP($\Gamma$) as follows.

**Problem 7.16.** *Name.* #CSP($\Gamma$).

*Parameter.* A constraint language $\Gamma$.

*Input.* A set $V$ of variables and a set $C$ of constraints of the form $\langle (v_1, \ldots, v_k), R \rangle$, where $(v_1, \ldots, v_k) \in V^k$ and $R$ is an arity-$k$ relation in $\Gamma$.

*Output.* The number of assignments $\sigma \colon V \to D$ such that

$$(\sigma(v_1), \ldots, \sigma(v_k)) \in R \text{ for all } \langle (v_1, \ldots, v_k), R \rangle \in C\,. \tag{7.1}$$

The tuple of variables $v_1, \ldots, v_k$ in a constraint is referred to as the constraint's *scope*. The assignments $\sigma\colon V \to D$ for which (7.1) holds are called the *satisfying assignments* of the instance $(V, C)$. Note that a unary constraint $\langle v, R\rangle$ has the same effect as a list: it directly restricts the possible values of the variable $v$. As before, we allow the possibility that $\emptyset \in \Gamma$; any instance that includes a constraint $\langle (v_1, \ldots, v_k), \emptyset\rangle$ has no satisfying assignments.

**Definition 7.17.** Let $M$ be a symmetric matrix in $\{0, 1, *\}^{D\times D}$ and let $\mathcal{L}$ be a subset-closed $M$-purifying set. Define the constraint language

$$\Gamma'_{\mathcal{L},M} = \{H^M_{X,Y} \mid X, Y \in \mathcal{L}\}$$

and let $\Gamma_{\mathcal{L},M} = \Gamma'_{\mathcal{L},M} \cup \mathcal{P}(D)$, where $\mathcal{P}(D)$ represents the set of all unary relations on $D$.

The unary constraints in $\Gamma_{\mathcal{L},M}$ will be useful in our study of the complexity of the dichotomy criterion, in Section 7.6. First, we define a convenient restriction on instances of #CSP($\Gamma_{\mathcal{L},M}$).

**Definition 7.18.** An instance of #CSP($\Gamma_{\mathcal{L},M}$) is *simple* if:

1. there is exactly one unary constraint $\langle v, X_v\rangle$ for each variable $v \in V$,
2. there are no binary constraints $\langle (v, v), R\rangle$, and
3. each pair $u$, $v$ of distinct variables appears in at most one constraint of the form $\langle (u, v), R\rangle$ or $\langle (v, u), R\rangle$.

**Lemma 7.19.** *For every instance $(V, C)$ of #CSP($\Gamma_{\mathcal{L},M}$), there is a simple instance $(V, C')$ such that an assignment $\sigma\colon V \to D$ satisfies $(V, C)$ if and only if it satisfies $(V, C')$. Further, such an instance can be computed in polynomial time.*

*Proof.* Observe that the set of binary relations in $\Gamma_{\mathcal{L},M}$ is closed under intersections: $H^M_{X,Y} \cap H^M_{X',Y'} = H^M_{X\cap X',Y\cap Y'}$ and this relation is in $\Gamma_{\mathcal{L},M}$ because $\mathcal{L}$ is subset-closed. The binary part of $\Gamma_{\mathcal{L},M}$ is also closed under relational inverse because $M$ is symmetric, so

$$\left(H^M_{X,Y}\right)^{-1} = \{(b, a) \mid (a, b) \in H^M_{X,Y}\} = H^M_{Y,X} \in \Gamma_{\mathcal{L},M}\,.$$

Since $\mathcal{P}(D) \subseteq \Gamma_{\mathcal{L},M}$, the set of unary relations is also closed under intersections.

We construct $C'$ as follows, starting with $C$. Any binary constraint $\langle (v, v), R\rangle$ can be replaced by the unary constraint $\langle v, \{d \mid (d, d) \in R\}\rangle$. All the binary constraints between distinct variables $u$ and $v$ can be replaced by the single constraint

$$\left\langle (u, v), \bigcap\{R \mid \langle (u, v), R\rangle \in C \text{ or } \langle (v, u), R^{-1}\rangle \in C\}\right\rangle\,.$$

Let the set of constraints produced so far be $C''$. For each variable $v$ in turn, if there are no unary constraints applied to $v$ in $C''$, add the constraint $\langle v, D\rangle$; otherwise, replace all the unary constraints involving $v$ in $C''$ with the single constraint

$$\left\langle v, \bigcap\{R \mid \langle v, R\rangle \in C''\}\right\rangle .$$

$C'$ is the resulting constraint set. The closure properties established above guarantee that $(V, C')$ is a #CSP$(\Gamma_{\mathcal{L},M})$ instance. It is clear that it has the same satisfying assignments as $(V, C)$ and that it can be produced in polynomial time. $\square$

Our main result connecting the counting list $M$-partitions problem with counting CSPs is the following.

**Proposition 7.20.** *For any symmetric $M \in \{0, 1, *\}^{D\times D}$ and any subset-closed, $M$-purifying set $\mathcal{L}$, the problem #$\mathcal{L}$-$M$-*PARTITIONS *is polynomial-time Turing-equivalent to* #CSP$(\Gamma_{\mathcal{L},M})$.

Because of its length, we split the proof of the proposition into two lemmas.

**Lemma 7.21.** *For any symmetric matrix $M \in \{0, 1, *\}^{D\times D}$ and any subset-closed, $M$-purifying set $\mathcal{L}$, the problem* #CSP$(\Gamma_{\mathcal{L},M})$ *is polynomial-time Turing-reducible to* #$\mathcal{L}$-$M$-PARTITIONS.

*Proof.* Consider an input $(V, C)$ to #CSP$(\Gamma_{\mathcal{L},M})$, which we may assume to be simple. Each variable appears in exactly one unary constraint, $\langle v, X_v\rangle \in C$. Any variable $v$ that is not used in a binary constraint can take any value in $X_v$ so just introduces a multiplicative factor of $|X_v|$ to the output of the counting CSP. Thus, we will assume without loss of generality that every variable is used in at least one constraint with a relation from $\Gamma'_{\mathcal{L},M}$ and, by simplicity, there are no constraints of the form $\langle (v, v), R\rangle$.

We now define a corresponding instance $(G, L)$ of #$\mathcal{L}$-$M$-PARTITIONS. The vertices of $G$ are the variables $V$ of the #CSP instance. For each variable $v \in V$, set

$$L(v) = X_v \cap \bigcap \left\{X \mid \text{for some } u \text{ and } Y,\ \langle (v, u), H^M_{X,Y}\rangle \in C \text{ or } \langle (u, v), H^M_{Y,X}\rangle \in C\right\}.$$

The edges $E(G)$ of our instance are the unordered pairs $\{u, v\}$ that satisfy one of the following conditions:

1. there is a constraint between $u$ and $v$ in $C$ and $M|_{L(u)\times L(v)}$ has a 0 entry, or

2. there is no constraint between $u$ and $v$ in $C$ and $M|_{L(u)\times L(v)}$ has a 1 entry.

Since every vertex $v$ is used in at least one constraint with a relation $H^M_{X,Y}$ where, by definition, $X$ and $Y$ are in $\mathcal{L}$, every set $L(v)$ is a subset of some set $W \in \mathcal{L}$. $\mathcal{L}$ is subset-closed so $L(v) \in \mathcal{L}$ for all $v \in V$, as required.

We claim that a function $\sigma\colon V \to D$ is a satisfying assignment of $(V, C)$ if and only if it is an $M$-partition of $G$ that respects $L$. Note that, since $\mathcal{L}$ is $M$-purifying, no submatrix $M|_{X\times Y}$ $(X, Y \in \mathcal{L})$ contains both 0s and 1s.

First, suppose that $\sigma$ is a satisfying assignment of $(V, C)$. For each variable $v$, $\sigma$ satisfies all the constraints $\langle v, X_v \rangle$, $\langle (v, u), H^M_{X,Y} \rangle$ and $\langle (u, v), H^M_{Y,X} \rangle$ containing $v$. Therefore, $\sigma(v) \in X_v$ and $\sigma(v) \in X$ for each binary constraint $\langle (v, u), H^M_{X,Y} \rangle$ or $\langle (u, v), H^M_{Y,X} \rangle$, so $\sigma$ satisfies all the list requirements.

To show that $\sigma$ is an $M$-partition of $G$, consider any pair of distinct vertices $u, v \in V$. If there is a constraint $\langle (u, v), H^M_{X,Y} \rangle \in C$, then $\sigma$ satisfies this constraint so $M_{\sigma(u),\sigma(v)} = *$ and $u$ and $v$ cannot stop $\sigma$ being an $M$-partition. Conversely, suppose there is no constraint between $u$ and $v$ in $C$. If $M|_{L(u)\times L(v)}$ contains a 0, there is no edge $(u, v) \in E(G)$ by construction; otherwise, if $M|_{L(u)\times L(v)}$ contains a 1, there is an edge $(u, v) \in E(G)$ by construction; otherwise, $M_{x,y} = *$ for all $x \in L(u)$, $y \in L(v)$. In all three cases, the assignment to $u$ and $v$ is consistent with $\sigma$ being an $M$-partition.

Conversely, suppose that $\sigma$ is not a satisfying assignment of $(V, C)$. If $\sigma$ does not satisfy some unary constraint $\langle v, X \rangle$ then $\sigma(v) \notin L(v)$ so $\sigma$ does not respect $\mathcal{L}$. If $\sigma$ does not satisfy some binary constraint $\langle (u, v), H^M_{X,Y} \rangle$ where $u$ and $v$ are distinct then, by definition of the relation $H^M_{X,Y}$, $M_{\sigma(u),\sigma(v)} \neq *$. If $M_{\sigma(u),\sigma(v)} = 0$, there is an edge $(u, v) \in E(G)$ by construction, which is forbidden in $M$-partitions; if $M_{\sigma(u),\sigma(v)} = 1$, there is no edge $(u, v) \in E(G)$ but this edge is required in $M$-partitions. Hence, $\sigma$ is not an $M$-partition. □

**Lemma 7.22.** *For any symmetric $M \in \{0, 1, *\}^{D\times D}$ and any $M$-purifying set $\mathcal{L}$ that is subset-closed, the problem #$\mathcal{L}$-$M$-*PARTITIONS* is polynomial-time Turing-reducible to* #CSP($\Gamma_{\mathcal{L},M}$).

*Proof.* We now essentially reverse the construction of the previous lemma to give a reduction from #$\mathcal{L}$-$M$-PARTITIONS to #CSP($\Gamma_{\mathcal{L},M}$). For any instance $(G, L)$ of #$\mathcal{L}$-$M$-PARTITIONS, we construct a corresponding instance $(V, C)$ of #CSP($\Gamma_{\mathcal{L},M}$) as follows. The set of variables $V$ is $V(G)$. The set of constraints $C$ consists of a constraint $\langle v, L(v) \rangle$ for each vertex $v \in V(G)$ and a constraint $\langle (u, v), H^M_{L(u),L(v)} \rangle$ for every pair of distinct vertices $u$, $v$ such that:

1. $(u, v) \in E(G)$ and $M|_{L(u)\times L(v)}$ has a 0 entry, or

2. $(u, v) \notin E(G)$ and $M|_{L(u)\times L(v)}$ has a 1 entry.

We show that a function $\sigma\colon V \to D$ is a satisfying assignment of $(V, C)$ if and only if it is an $M$-partition of $G$ that respects $L$. It is clear that $\sigma$ satisfies the unary constraints if and only if it respects $L$.

If $\sigma$ satisfies $(V, C)$, then consider any pair of distinct vertices $u, v \in V$. If there is a binary constraint involving $u$ and $v$, then $M_{\sigma(u),\sigma(v)} = M_{\sigma(v),\sigma(u)} = *$ so the existence or non-existence of the edge $(u, v)$ of $G$ does not affect whether $\sigma$ is an $M$-partition. If there is no binary constraint involving $u$ and $v$, then either there is an edge $(u, v) \in E(G)$ and $M_{\sigma(u),\sigma(v)} \neq 0$ or there is no edge $(u, v)$ and $M_{\sigma(u),\sigma(v)} \neq 1$. In all three cases, $\sigma$ maps $u$ and $v$ consistently with it being an $M$-partition.

**Algorithm 1** The algorithm for computing arc-consistent domains for a simple $\#\mathrm{CSP}(\Gamma_{\mathcal{L},M})$ instance $(V, C)$ where, for each $v \in V$, $\langle v, X_v\rangle \in C$ is the unary constraint involving $v$.

```
for $v \in V$ do
    $D_v \leftarrow X_v$
repeat
    for $v \in V$ do
        $D'_v \leftarrow D_v$
    for $\langle (u,v), R\rangle \in C$ do
        $D_u \leftarrow \{d \in D_u \mid \text{for some } d' \in D_v,\ (d,d') \in R\}$
        $D_v \leftarrow \{d \in D_v \mid \text{for some } d' \in D_u,\ (d',d) \in R\}$
until $\forall v \in V,\ D_v = D'_v$
return $(D_v)_{v\in V}$
```

Conversely, if $\sigma$ does not satisfy $(V, C)$, either it fails to satisfy a unary constraint, in which case it does not respect $L$, or it satisfies all unary constraints (so it respects $L$), but it fails to satisfy a binary constraint $\langle (u, v), H^M_{L(u),L(v)}\rangle$. In the latter case, by construction, $M_{\sigma(u),\sigma(v)} \neq *$ so either $M_{\sigma(u),\sigma(v)} = 0$ but there is an edge $(u, v) \in E(G)$, or $M_{\sigma(u),\sigma(v)} = 1$ and there is no edge $(u, v) \in E(G)$. In either case, $\sigma$ is not an $M$-partition of $G$. □

## 7.4 An arc-consistency based algorithm for the constraint satisfaction problem $\#\mathrm{CSP}(\Gamma_{\mathcal{L},M})$

In the previous section, we showed that a class of $\#\mathcal{L}$-$M$-PARTITIONS problems is equivalent to a certain class of counting CSPs, where the constraint language consists of binary relations and all unary relations over the domain $D$. We now investigate the complexity of such #CSPs.

Arc-consistency is a standard solution technique for constraint satisfaction problems [58]. It is, essentially, a local search method which initially assumes that each variable may take any value in the domain and iteratively reduces the range of values that can be assigned to each variable, based on the constraints applied to it and the values that can be taken by other variables in the scopes of those constraints.

For any simple $\#\mathrm{CSP}(\Gamma_{\mathcal{L},M})$ instance $(V, C)$, define the vector of *arc-consistent domains* $(D_v)_{v\in V}$ by the procedure in Algorithm 1. At no point in the execution of the algorithm can any domain $D_v$ increase in size so, for fixed $D$, the running time of the algorithm is at most a polynomial in $|V| + |C|$.

It is clear that, if $(D_v)_{v\in V}$ is the vector of arc-consistent domains for a simple $\#\mathrm{CSP}(\Gamma_{\mathcal{L},M})$ instance $(V, C)$, then every satisfying assignment $\sigma$ for that instance must have $\sigma(v) \in D_v$ for each variable $v$. In particular, if some $D_v = \emptyset$, then the instance is unsatisfiable. (Note, though, that the converse does not hold. If $D = \{0, 1\}$ and

**Algorithm 2** The algorithm for factoring a simple #CSP($\Gamma_{\mathcal{L},M}$) instance $(V, C)$ with respect to a vector $(D_v)_{v\in V}$ of arc-consistent domains. $F$ is the set of factored constraints.

```
$F \leftarrow C$
**for** $\langle (u,v), R\rangle \in C$ **do**
    **if** $R \cap (D_u \times D_v)$ is a Cartesian product $D'_u \times D'_v$ **then**
        Let $\langle u, X_u\rangle$ and $\langle v, X_v\rangle$ be the unary constraints involving $u$ and $v$ in $F$.
        $F \leftarrow (F \cup \{\langle u, X_u \cap D'_u\rangle, \langle v, X_u \cap D'_v\rangle\}) \setminus \{\langle (u,v), R\rangle, \langle u, X_u\rangle, \langle v, X_v\rangle\}$
**return** $F$
```

$R = \{(0,1),(1,0)\}$, the instance with constraints $\langle x, D\rangle$, $\langle y, D\rangle$, $\langle z, D\rangle$, $\langle (x,y), R\rangle$, $\langle (y,z), R\rangle$ and $\langle (z,x), R\rangle$ is unsatisfiable but arc-consistency assigns $D_x = D_y = D_z = \{0,1\}$.)

The arc-consistent domains computed for a simple instance $(V, C)$ can yield further simplification of the constraint structure, which we refer to as *factoring.* The factoring applies when the arc-consistent domains restrict a binary relation to a Cartesian product. In this case, the binary relation can be replaced with corresponding unary relations. Algorithm 2 factors a simple instance with respect to a vector $(D_v)_{v\in V}$ of arc-consistent domains, producing a set $F$ of factored constraints. Recall that there is at most one constraint in $C$ between distinct variables and there are no binary constraints $\langle (v,v), R\rangle$ because the instance is simple. Note also that, if $|D_u| \leq 1$ or $|D_v| \leq 1$, then $R \cap (D_u \times D_v)$ is necessarily a Cartesian product. It is easy to see that the result of factoring a simple instance is simple, that Algorithm 2 runs in polynomial time and that the instance $(V, F)$ has the same satisfying assignments as $(V, C)$.

The *constraint graph* of a CSP instance $(V, C)$ (in any constraint language) is the undirected graph with vertex set $V$ that contains an edge between every pair of distinct variables that appear together in the scope of some constraint.

Algorithm 3 uses arc-consistency to count the satisfying assignments of simple #CSP($\Gamma_{\mathcal{L},M}$) instances. It is straightforward to see that the algorithm terminates, since each recursive call is either on an instance with strictly fewer variables or on one in which at least one variable has had its unary constraint reduced to a singleton and no variable's unary constraint has increased. For general inputs, the algorithm may take exponential time to run but, in Lemma 7.23 we show that the running time is polynomial for the inputs we are interested in.

We first argue that the algorithm is correct. By Lemma 7.19, we may assume that the given instance $(V, C)$ is simple. Every satisfying assignment $\sigma\colon V \to D$ satisfies $\sigma(v) \in D_v$ for all $v \in V$ so restricting our attention to arc-consistent domains does not alter the output. Factoring the constraints also does not change the number of satisfying assignments: it merely replaces some binary constraints with equivalent unary ones. The constraints are factored, so any variable $v$ with $|D_v| = 1$ must, in fact, be an isolated vertex in the constraint graph because, as noted above, any binary

**Algorithm 3** The arc-consistency based algorithm for counting satisfying assignments to simple instances of #CSP($\Gamma_{\mathcal{L},M}$). The input is a simple instance $(V, C)$ of #CSP($\Gamma_{\mathcal{L},M}$).

```
function AC(variable set V, constraint set C)
    Use Algorithm 1 to compute the vector of arc-consistent domains $(D_v)_{v \in V}$
    Use Algorithm 2 to construct the set $F$ of factored constraints
    if $D_v = \emptyset$ for some $v \in V$ then
        return 0
    Compute the constraint graph $H$ of $(V, F)$
    Let $H_1, \dots, H_\kappa$ be the components of $H$ with $V_i = V(H_i)$
    Let $F_i$ be the set of constraints in $F$ involving variables in $V_i$
    for $i \in [\kappa]$ do
        if $|D_w| = 1$ for some $w \in V_i$ then
            $Z_i \leftarrow 1$
        else
            Choose $w_i \in V_i$
            Let $\theta_i$ be the unary constraint involving $w_i$ in $F_i$
            for $d \in D_{w_i}$ do
                $F'_{i,d} \leftarrow (F_i \cup \{\langle w_i, \{d\}\rangle\}) \setminus \{\theta_i\}$
            $Z_i \leftarrow \sum_{d \in D_{w_i}} \mathrm{AC}(V_i, F'_{i,d})$
    return $\prod_{i=1}^{\kappa} Z_i$
```

constraint involving it has been replaced by unary constraints. Therefore, if a component $H_i$ contains a variable $v$ with $|D_v| = 1$, that component is the single vertex $v$, which is constrained to take a single value, so the number of satisfying assignments for this component, which we denote $Z_i$, is equal to 1. (So we have now shown that the if branch in the for loop is correct.) For components that contain more than one variable, it is clear that we can choose one of those variables, $w_i$, and group the set of $M$-partitions $\sigma$ according to the value of $\sigma(w_i)$. (So we have now shown that the else branch is correct.) Because there are no constraints between variables in different components of the constraint graph, the number of satisfying assignments factorises as $\prod_{i=1}^{\kappa} Z_i$.

For a binary relation $R$, we write

$$\pi_1(R) = \{a \mid (a, b) \in R \text{ for some } b\}$$
$$\pi_2(R) = \{b \mid (a, b) \in R \text{ for some } a\}.$$

For the following proof, we will also need the observation of Dyer and Richerby [25, Lemma 1] that any rectangular relation $R \subseteq \pi_1(R) \times \pi_2(R)$ can be written as $(A_1 \times B_1) \cup \dots \cup (A_\lambda \times B_\lambda)$, where the $A_i$ and $B_i$ partition $\pi_1(R)$ and $\pi_2(R)$, respectively. The subrelations $A_i \times B_i$ are referred to as *blocks*. A rectangular relation $R \neq \pi_1(R) \times \pi_2(R)$ must have at least two blocks.

**Lemma 7.23.** *Suppose that $\mathcal{L}$ is subset-closed and $M$-purifying. If there is no $\mathcal{L}$-$M$-*

*derectangularising sequence, then Algorithm 3 runs in polynomial time.*

*Proof.* We will argue that the number of recursive calls made by the function AC in Algorithm 3 is bounded above by a polynomial in $|V|$. This suffices, since every other step of the procedure is obviously polynomial.

Consider a run of the algorithm on instance $(V, C)$ which, by Lemma 7.19, we may assume to be simple. Suppose the run makes a recursive call with input $(V_i, F'_{i,d})$. For each $v \in V_i$, let $D'_v$ denote the arc-consistent domain for $v$ that is computed during the recursive call. We will show below that $D'_v \subset D_v$ for every variable $v \in V_i$. This implies that the recursion depth is at most $|D|$. As a crude bound, it follows that the number of recursive calls is at most $(|V| \cdot |D|)^{|D|}$, since each recursive call that is made is nested below a sequence of at most $|D|$ previous calls, each of which chose a vertex $v \in V$ and "pinned" it to a domain element $d \in D$ (i.e., introduced the constraint $\langle v, \{d\}\rangle$).

Towards showing that the domains of all variables decrease at each recursive call, suppose that we are computing $\mathrm{AC}(V, C)$ and the arc-consistent domains are $(D_v)_{v \in V}$. As observed above, for any component $H_i$ of the constraint graph on which a recursive call is made, we must have $|D_v| > 1$ for every $v \in V_i$. Fix such a component and, for each $v \in V_i$, let $D'_v$ be the arc-consistent domain calculated for $v$ in the recursive call on $H_i$. It is clear that $D'_v \subseteq D_v$; we will show that $D'_v \subset D_v$.

Consider a path $v_1 \dots v_\ell$ in $H_i$, where $v_1 = w_i$ and $v_\ell = v$. For each $j \in [\ell - 1]$, there is exactly one binary constraint in $F_i$ involving $v_j$ and $v_{j+1}$. This is either $\langle (v_j, v_{j+1}), R_j\rangle$ or $\langle (v_{j+1}, v_j), R_j^{-1}\rangle$ and, without loss of generality, we may assume that it is the former. For $j \in [\ell - 1]$, let $R'_j = R_j \cap (D_{v_j} \times D_{v_{j+1}}) = H^M_{D_{v_j}, D_{v_{j+1}}}$. The relation $R'_j$ is pure because $D_{v_j}$ and $D_{v_{j+1}}$ are in the subset-closed set $\mathcal{L}$ and, since $\mathcal{L}$ is $M$-purifying, so is $\{D_{v_j}, D_{v_{j+1}}\}$. These two domains do not form a derectangularising sequence by the hypothesis of the lemma, so $H^M_{D_{v_j}, D_{v_{j+1}}}$ is rectangular. If some $R_j = \emptyset$ then $D_{v_j} = D_{v_{j+1}} = \emptyset$ by arc-consistency, contradicting the fact that $|D_v| > 1$ for all $v \in V_i$. If some $R'_j$ has just one block, $R_j \cap (D_{v_j} \times D_{v_{j+1}})$ is a Cartesian product, contradicting the fact that $F$ is a factored set of constraints. Thus, every $R'_j$ has at least two blocks.

For $j \in [\ell - 1]$, let $\Phi_j = R'_1 \circ \dots \circ R'_j$. As above, note that $\{D_{v_1}, \dots, D_{v_{j+1}}\}$ is $M$-purifying and the sequence $D_{v_1}, \dots, D_{v_{j+1}}$ is not derectangularising, so $\Phi_j$ is rectangular. We will show by induction on $j$ that $\pi_1(\Phi_j) = D_{v_1}$, $\pi_2(\Phi_j) = D_{v_{j+1}}$ and $\Phi_j$ has at least two blocks. Therefore, since the recursive call constrains $\sigma(w_i)$ to be $d$ and $d \in A$ for some block $A \times B \subset \Phi_\ell$, we have $D'_v \subseteq B \subset D_v$, which is what we set out to prove.

For the base case of the induction, take $j = 1$ so $\Phi_1 = R'_1$. We showed above that $R'_1$ has at least two blocks and that $R'_1 = H^M_{D_{v_1}, D_{v_2}}$. By arc-consistency, $\pi_1(R'_1) = D_{v_1}$ and $\pi_2(R'_1) = D_{v_2}$.

For the inductive step, take $j \in [\ell - 2]$. Suppose that $\pi_1(\Phi_j) = D_{v_1}$, $\pi_2(\Phi_j) = D_{v_{j+1}}$ and $\Phi_j = \bigcup_{s=1}^{\lambda} (A_s \times A'_s)$ has at least two blocks. We have $\Phi_{j+1} = \Phi_j \circ R'_{j+1}$ and

$R'_{j+1} = \bigcup_{t=1}^{\mu}(B_t \times B'_t)$ for some $\mu \geq 2$.

For every $d \in D_{v_1}$, there is a $d' \in D_{v_{j+1}}$ such that $(d, d') \in \Phi_j$ by the inductive hypothesis, and a $d'' \in D_{v_{j+1}}$ such that $(d', d'') \in D_{v_{j+2}}$, by arc-consistency. Therefore, $\pi_1(\Phi_{j+1}) = D_{v_1}$; a similar argument shows that $\pi_2(\Phi_{j+1}) = D_{v_{j+2}}$.

Suppose, towards a contradiction, that $\Phi_{j+1} = D_{v_1} \times D_{v_{j+2}}$. For this to be the case, we must have $A'_s \cap B_t \neq \emptyset$ for every $s \in \{1,2\}$ and $t \in [\mu]$. Now, let $D^*_{v_{j+1}} = D_{v_{j+1}} \setminus (A'_2 \cap B_2)$ and consider the relation

$$R = \{(d_1, d_3) \mid \text{for some } d_2 \in D^*_{v_{j+1}},\ (d_1, d_2) \in \Phi_j \text{ and } (d_2, d_3) \in R'_{j+1}\ \}.$$

Since $A'_1 \subseteq D^*_{v_{j+1}}$ the non-empty sets $A'_1 \cap B_1$ and $A'_1 \cap B_2$ are both subsets of $D^*_{v_{j+1}}$ so $A_1 \times B'_1 \subseteq R$ and $A_1 \times B'_2 \subseteq R$. Similarly, $B_1 \subseteq D^*_{v_{j+1}}$, so $A'_2 \cap B_1 \subseteq D^*_{v_{j+1}}$ so $A_2 \times B'_1 \subseteq R$. However, $(A_2 \times B'_2) \cap R = \emptyset$, so $R$ is not rectangular. We will now derive a contradiction by showing that $R$ is rectangular. Note that

$$R = H^M_{D_{v_1}, D_{v_2}} \circ \cdots \circ H^M_{D_{v_{j-1}}, D_{v_j}} \circ H^M_{D_{v_j}, D^*_{v_{j+1}}} \circ H^M_{D^*_{v_{j+1}}, D_{v_{j+2}}}$$

but this relation is rectangular because the hypothesis of the lemma guarantees that the sequence

$$D_{v_1}, \ldots, D_{v_j}, D^*_{v_{j+1}}, D_{v_{j+2}}$$

is not an $\mathcal{L}$-$M$-derectangularising sequence and all of the elements of this sequence are in $\mathcal{L}$, and $\{D_{v_1}, \ldots, D_{v_j}, D^*_{v_{j+1}}, D_{v_{j+2}}\}$ is $M$-purifying. □

## 7.5 Polynomial-time algorithms and the dichotomy theorem

Bulatov [10] showed that every problem of the form #CSP(Γ) is either in FP or #P-complete. Together with Proposition 7.20, his result immediately shows that a similar dichotomy exists for the special case of the problem #$\mathcal{L}$-$M$-PARTITIONS in which $\mathcal{L}$ is $M$-purifying and is closed under subsets. Our algorithmic work in Section 7.4 can be combined with Dyer and Richerby's explicit dichotomy for #CSP to obtain an explicit dichotomy for this special case of #$\mathcal{L}$-$M$-PARTITIONS. In particular, Lemma 7.23 gives a polynomial-time algorithm for the case in which there is no $\mathcal{L}$-$M$-derectangularising sequence. When there is such a sequence, $\Gamma_{\mathcal{L},M}$ is not "strongly rectangular" in the sense of [25]. It follows immediately that #CSP($\Gamma_{\mathcal{L},M}$) is #P-complete [25, Lemma 24] so #$\mathcal{L}$-$M$-PARTITIONS is also #P-complete by Proposition 7.20. In fact, the dichotomy for this special case does not require the full generality of Dyer and Richerby's dichotomy. If there is an $\mathcal{L}$-$M$-derectangularising sequence then it follows immediately from work of Bulatov and Dalmau [11, Theorem 2 and Corollary 3] that #CSP($\Gamma_{\mathcal{L},M}$) is #P-complete.

In this section we will move beyond the case in which $\mathcal{L}$ is $M$-purifying to provide a full dichotomy for the problem #$\mathcal{L}$-$M$-PARTITIONS. We will use two data structures: *sparse-dense partitions* and a representation of the set of *splits* of a bipartite graph. Similar data structures were used by Hell et al. [51] in their dichotomy for the #$M$-PARTITIONS problem for matrices of size at most 3-by-3.

### 7.5.1 Data Structures

We use two types of graph partition. The first is a special case of a sparse-dense partition [37] which is also called an $(a, b)$-graph with $a = b = 2$.

**Definition 7.24.** A bipartite–cobipartite partition of a graph $G$ is a partition $(B, C)$ of $V(G)$ such that $B$ induces a bipartite graph and $C$ induces the complement of a bipartite graph.

**Lemma 7.25.** *[37, Theorem 3.1; see also the remarks on $(a, b)$-graphs.] There is a polynomial-time algorithm for finding all bipartite–cobipartite partitions of a graph $G$.*

The second decomposition is based on certain sub-hypercubes called subcubes. For any finite set $U$, a *subcube* of $\{0, 1\}^U$ is a subset of $\{0, 1\}^U$ that is a Cartesian product of the form $\prod_{u\in U} S_u$ where $S_u \in \{\{0\}, \{1\}, \{0, 1\}\}$ for each $u \in U$. We can also associate a subcube $\prod_{u\in U} S_u$ with the set of assignments $\sigma\colon U \to \{0, 1\}$ such that $\sigma(u) \in S_u$ for all $u \in U$. Subcubes can be represented efficiently by listing the projections $S_u$.

**Definition 7.26.** Let $G = (U, U', E)$ be a bipartite graph, where $U$ and $U'$ are disjoint vertex sets, and $E \subseteq U \times U'$. A *subcube decomposition* of $G$ is a list $U_1, \ldots, U_k$ of subcubes of $\{0, 1\}^U$ and a list $U'_1, \ldots, U'_k$ of subcubes of $\{0, 1\}^{U'}$ such that the following hold.

1. The union $(U_1 \times U'_1) \cup \cdots \cup (U_k \times U'_k)$ is the set of assignments $\sigma\colon U \cup U' \to \{0, 1\}$ such that:

$$\text{no edge } (u, u') \in E \text{ has } \sigma(u) = \sigma(u') = 0 \text{ and} \tag{7.2}$$

$$\text{no pair } (u, u') \in (U \times U') \setminus E \text{ has } \sigma(u) = \sigma(u') = 1. \tag{7.3}$$

2. For distinct $i, j \in [k]$, $U_i \times U'_i$ and $U_j \times U'_j$ are disjoint.
3. For each $i \in [k]$, either $|U_i| = 1$ or $|U'_i| = 1$ (or both).

Note that, although we require $U_i \times U'_i$ and $U_j \times U'_j$ to be disjoint for distinct $i, j \in [k]$, we allow $U_i \cap U_j \neq \emptyset$ as long as $U'_i$ and $U'_j$ are disjoint, and vice-versa. It is even possible that $U_i = U_j$, and indeed this will happen in our constructions below.

**Lemma 7.27.** *A subcube decomposition of a bipartite graph $G = (U, U', E)$ can be computed in polynomial time, with the subcubes represented by their projections.*

*Proof.* For a vertex $x$ in a bipartite graph, let $\Gamma(x)$ be its set of neighbours and let $\overline{\Gamma}(x)$ be its set of non-neighbours on the other side of the graph. Thus, for $x \in U$, $\overline{\Gamma}(x) = U' \setminus \Gamma(x)$ and, for $x \in U'$, $\overline{\Gamma}(x) = U \setminus \Gamma(x)$.

Observe that we can write $\{0,1\}^n \setminus \{0\}^n$ as the disjoint union of $n$ subcubes $\{0,1\}^{k-1} \times \{1\}^1 \times \{0,1\}^{n-k}$ with $1 \leq k \leq n$, and similarly for any other cube minus a single point.

We first deal with two base cases. If $G$ has no edges, then the set of assignments $\sigma \colon U \cup U' \to \{0,1\}$ satisfying (7.2) and (7.3) is the disjoint union of

$$\{0\}^U \times \{0\}^{U'}, \quad (\{0,1\}^U \setminus \{0\}^U) \times \{0\}^{U'} \quad \text{and} \quad \{0\}^U \times (\{0,1\}^{U'} \setminus \{0\}^{U'}).$$

The second and third terms can be decomposed into subcubes as described above to produce the output. Similarly, if $G$ is a complete bipartite graph, then the set of assignments satisfying (7.2) and (7.3) is the disjoint union of

$$\{1\}^U \times \{1\}^{U'}, \quad (\{0,1\}^U \setminus \{1\}^U) \times \{1\}^{U'} \quad \text{and} \quad \{1\}^U \times (\{0,1\}^{U'} \setminus \{1\}^{U'}).$$

If neither of these cases occurs then there is a vertex $x$ such that neither $\Gamma(x)$ nor $\overline{\Gamma}(x)$ is empty. If possible, choose $x \in U$; otherwise, choose $x \in U'$. To simplify the description of the algorithm, we assume that $x \in U$; the other case is symmetric. We consider separately the assignments where $\sigma(x) = 0$ and those where $\sigma(x) = 1$. Note that, for any assignment, if $\sigma(y) = 0$ for some vertex $y$, then $\sigma(z) = 1$ for all $z \in \Gamma(y)$ and, if $\sigma(y) = 1$, then $\sigma(z) = 0$ for all $z \in \overline{\Gamma}(y)$. Applying this iteratively, setting $\sigma(x) = c$ for $c \in \{0,1\}$ also determines the value of $\sigma$ on some set $S_{x=c} \subseteq U \cup U'$ of vertices.

Thus, we can compute a subcube decomposition for $G$ recursively. First, compute $S_{x=0}$ and $S_{x=1}$. Then, recursively compute subcube decompositions of $G - S_{x=0}$ (the graph formed from $G$ by deleting the vertices in $S_{x=0}$) and $G - S_{x=1}$. Translate these subcube decompositions into a subcube decomposition of $G$ by extending each subcube $(U_i \times U_i')$ of $G - S_{x=c}$ to a subcube $(V_i \times V_i')$ of $G$ whose restriction to $G - S_{x=c}$ is $(U_i \times U_i')$ and whose restriction to $S_{x=c}$ is an assignment $\sigma$ with $\sigma(x) = c$ (in fact, all assignments that set $x$ to $c$ agree on the set $S_{x=c}$, by construction).

It remains to show that the algorithm runs in polynomial time. The base cases are clearly computable in polynomial time, as are the individual steps in the recursive cases, so we only need to show that the number of recursive calls is polynomially bounded. At the recursive step, we only choose $x \in U'$ when $E(G) = U'' \times U'$ for some proper subset $\emptyset \subset U'' \subset U$ and, in this case, the two recursive calls are to base cases. Since each recursive call when $x \in U$ splits $U'$ into disjoint subsets, there can be at most $|U'| - 1$ such recursive calls, so the total number of recursive calls is linear in $|V(G)|$. $\square$

### 7.5.2 Reduction to a problem with $M$-purifying lists

Our algorithm for counting list $M$-partitions uses the data structures from Section 7.5.1 to reduce problems where $\mathcal{L}$ is not $M$-purifying to problems where it is (which we already know how to solve from Sections 7.3 and 7.4). The algorithm is defined recursively on the set $\mathcal{L}$ of allowed lists. The algorithm for parameters $\mathcal{L}$ and $M$ calls the algorithm for $\mathcal{L}_i$ and $M$ where $\mathcal{L}_i$ is a subset of $\mathcal{L}$. The base case arises when $\mathcal{L}_i$ is $M$-purifying.

We will use the following computational problem to reduce #$\mathcal{L}$-$M$-PARTITIONS to a collection of problems #$\mathcal{L}'$-$M$-PARTITIONS that are, in a sense, disjoint.

**Problem 7.28.** *Name.* #$\mathcal{L}$-$M$-PURIFY.

*Parameter.* A symmetric matrix $M \in \{0, 1, *\}^{D\times D}$ and a subset $\mathcal{L}$ of $\mathcal{P}(D)$.

*Input.* A graph $G$ and a function $L\colon V(G) \to \mathcal{L}$.

*Output.* Functions $L_1, \dots, L_t\colon V(G) \to \mathcal{L}$ such that

1. for each $i \in [t]$, the set $\{L_i(v) \mid v \in V(G)\}$ is $M$-purifying,
2. for each $i \in [t]$ and $v \in V(G)$, $L_i(v) \subseteq L(v)$, and
3. each $M$-partition of $G$ that respects $L$ respects exactly one of the functions $L_1, \dots, L_t$.

We will give an algorithm for solving the problem #$\mathcal{L}$-$M$-PURIFY in polynomial time when there is no $\mathcal{L}$-$M$-derectangularising sequence of length exactly 2. The following computational problem will be central to the inductive step.

**Problem 7.29.** *Name.* #$\mathcal{L}$-$M$-PURIFY-STEP.

*Parameter.* A symmetric matrix $M \in \{0, 1, *\}^{D\times D}$ and a subset $\mathcal{L}$ of $\mathcal{P}(D)$.

*Input.* A graph $G$ and a function $L\colon V(G) \to \mathcal{L}$.

*Output.* Functions $L_1, \dots, L_k\colon V(G) \to \mathcal{L}$ such that

1. for each $i \in [k]$ and $v \in V(G)$, $L_i(v) \subseteq L(v)$,
2. every $M$-partition of $G$ that respects $L$ respects exactly one of $L_1, \dots, L_k$, and
3. for each $i \in [k]$, there is a $W \in \mathcal{L}$ which is inclusion-maximal in $\mathcal{L}$ but does not occur in the image of $L_i$.

Note that we can trivially produce a solution to the problem #$\mathcal{L}$-$M$-PURIFY-STEP by letting $L_1, \dots, L_k$ be an enumeration of all possible functions such that all lists $L_i(v)$ have size 1 and satisfy $L_i(v) \subseteq L(v)$. Such a function $L_i$ corresponds to an

**Algorithm 4** A polynomial-time algorithm for the problem $\#\mathcal{L}$-$M$-PURIFY-STEP when $\mathcal{L} \subseteq \mathcal{P}(D)$ is subset-closed, $\mathcal{L}$ is not $M$-purifying and there is no length-2 $\mathcal{L}$-$M$-derectangularising sequence. The input is a pair $(G, L)$ with $V(G) = \{v_1, \ldots, v_n\}$.

```
function #ℒ-M-PURIFY-STEP(G,L)
    if there is a $v_i \in V(G)$ with $L(v_i) = \emptyset$ then return the empty sequence
    else if there are $X, Y \in \mathcal{L}$, $a, b \in X$, and $d \in Y$
              such that $M_{a,d} = 0$ and $M_{b,d} = 1$ then
        Run Algorithm 5    /* Case 1 */
    else if there is an $X \in \mathcal{L}$ such that $M|_{X\times X}$ is not pure then
        Run Algorithm 6    /* Case 2 */
    else
        Run Algorithm 7    /* Case 3 */
```

assignment of vertices to parts so there is either exactly one $L_i$-respecting $M$-partition or none, which means that every $L$-respecting $M$-partition is $L_i$-respecting for exactly one $i$. However, this solution is exponentially large in $|V(G)|$ and we are interested in solutions that can be produced in polynomial time. Also, if $L(v) = \emptyset$ for some vertex $v$, the algorithm is entitled to output an empty list, since no $M$-partition respects $L$.

The following definition extends rectangularity to $\{0, 1, *\}$-matrices and is used in our proof.

**Definition 7.30.** A matrix $M \in \{0, 1, *\}^{X\times Y}$ is *$*$-rectangular* if the relation $H^M_{X,Y}$ is rectangular.

Thus, $M$ is $*$-rectangular if and only if $M_{x,y} = M_{x',y} = M_{x,y'} = *$ implies that $M_{x',y'} = *$ for all $x, x' \in X$ and all $y, y' \in Y$.

We will show in Lemma 7.31 that the function $\#\mathcal{L}$-$M$-PURIFY-STEP from Algorithm 4 is a polynomial-time algorithm for the problem $\#\mathcal{L}$-$M$-PURIFY-STEP whenever $\mathcal{L}$ is not $M$-purifying and there is no length-2 $\mathcal{L}$-$M$-derectangularising sequence. Note that a length-2 $\mathcal{L}$-$M$-derectangularising sequence is a pair $X, Y \in \mathcal{L}$ such that $M|_{X\times Y}$, $M|_{X\times X}$ and $M|_{Y\times Y}$ are pure and $M|_{X\times Y}$ is not $*$-rectangular. If $\mathcal{L} \neq \mathcal{P}(D)$, it is possible that a matrix that is not $*$-rectangular has no length-2 $\mathcal{L}$-$M$-derectangularising sequence. For example, let $D = \{1, 2, 3\}$ and $\mathcal{L} = \mathcal{P}(\{1, 2\})$ and let $M_{3,3} = 0$ and $M_{i,j} = *$ for every other pair $(i, j) \in D^2$. $M$ is not $*$-rectangular but this fact is not witnessed by any submatrix $M|_{X\times Y}$ for $X, Y \in \mathcal{L}$.

**Lemma 7.31.** *Let $M$ be a symmetric matrix in $\{0, 1, *\}^{D\times D}$ and let $\mathcal{L} \subseteq \mathcal{P}(D)$ be subset-closed. If $\mathcal{L}$ is not $M$-purifying and there is no length-2 $\mathcal{L}$-$M$-derectangularising sequence, then Algorithm 4 is a polynomial-time algorithm for the problem $\#\mathcal{L}$-$M$-*PURIFY-STEP.

*Proof.* Let $(G, L)$ be an instance of the problem $\#\mathcal{L}$-$M$-PURIFY-STEP with $V(G) = \{v_1, \ldots, v_n\}$. If there is a $v_i \in V(G)$ with $L(v_i) = \emptyset$ then no $M$-partition of $G$ respects $L$, so the output is correct. Otherwise, we consider the three cases that can occur in the execution of the algorithm.

**Algorithm 5** Case 1 in Algorithm 4.

```
Choose $X, Y \in \mathcal{L}$, $a, b \in X$, and $d \in Y$
such that $M_{a,d} = 0$, $M_{b,d} = 1$ and $X$ and $Y$ are inclusion-maximal in $\mathcal{L}$
for $i \in [n]$ do
    $L_i(v_i) \leftarrow L(v_i) \cap \{d\}$
    for $j < i$ do
        if $(v_i, v_j) \in E(G)$ then
            $L_i(v_j) \leftarrow \{d' \in L(v_j) \mid d' \neq d$ and $M_{d,d'} \neq 0\}$
        else
            $L_i(v_j) \leftarrow \{d' \in L(v_j) \mid d' \neq d$ and $M_{d,d'} \neq 1\}$
    for $j > i$ do
        if $(v_i, v_j) \in E(G)$ then
            $L_i(v_j) \leftarrow \{d' \in L(v_j) \mid M_{d,d'} \neq 0\}$
        else
            $L_i(v_j) \leftarrow \{d' \in L(v_j) \mid M_{d,d'} \neq 1\}$
    $L_{n+1}(v_i) \leftarrow L(v_i) \setminus \{d\}$
return $L_1, \dots, L_{n+1}$ (of course, if we have $L_i(v) = \emptyset$ for any $i$ and $v$ then $L_i$ can
be omitted from the output)
```

**Case 1** In this case column $d$ of $M|_{X \times Y}$ contains both a zero and a one. Equivalently, row $d$ of $M|_{Y \times X}$ does. Algorithm 5 groups the set of $M$-partitions of $G$ that respect $L$, based on the first vertex that is placed in part $d$. For $i \in [n]$, $L_i$ requires that $v_i$ is placed in part $d$ and $v_1, \dots, v_{i-1}$ are not in part $d$; $L_{n+1}$ requires that part $d$ is empty. Thus, no $M$-partition can respect more than one of the $L_i$. Now consider an $L$-respecting $M$-partition $\sigma\colon V(G) \to D$ and suppose that $i$ is minimal such that $\sigma(v_i) = d$. We claim that $\sigma$ respects $L_i$. We have $\sigma(v_i) = d$, as required. For $j \neq i$, we must have $\sigma(v_j) \in L(v_j)$ since $\sigma$ respects $L$ and we must have $M_{d,\sigma(v_j)} \neq 1$ if $(v_i, v_j) \notin E(G)$ and $M_{d,\sigma(v_j)} \neq 0$ if $(v_i, v_j) \in E(G)$, since $\sigma$ is an $M$-partition. In addition, by construction, $\sigma(v_j) \neq d$ if $j < i$. Therefore, $\sigma$ respects $L_i$. A similar argument shows that $\sigma$ respects $L_{n+1}$ if $\sigma(v) \neq d$ for all $v \in V(G)$. Hence, any $M$-partition that respects $L$ respects exactly one of the $L_i$.

Finally, we show that, for each $i \in [n+1]$, there is a set $W$ which is inclusion-maximal in $\mathcal{L}$ and is not in the image of $L_i$. For $i \in [n]$, we cannot have both $a$ and $b$ in $L_i(v_j)$ for any $v_j$, so $X$ is not in the image of $L_i$. $Y$ contains $d$, so $Y$ is not in the image of $L_{n+1}$.

**Case 2** In this case, every row of $M|_{X_0 \times X}$ contains a 0, while every row of $M|_{X_1 \times X}$ fails to contain a zero. Since $M|_{X \times X}$ is not pure, but no row of $M|_{X \times X}$ contains both a zero and a one (since we are not in Case 1), $X_0$ and $X_1$ are non-empty. Note that $M|_{X_0 \times X_0}$ and $M|_{X_1 \times X_1}$ are both pure, but every entry of $M|_{X_0 \times X_1}$ is a $*$.

If $V_X = \emptyset$ then $X$ is an inclusion-maximal member of $\mathcal{L}$ that is not in the image of $L$, so the output of Algorithm 6 is correct. Otherwise, $(B_1, C_1), \dots, (B_k, C_k)$ is the list containing all partitions $(B, C)$ of $V_X$ such that $B$ induces a bipartite graph in $G$

**Algorithm 6** Case 2 in Algorithm 4.

```
Choose $X \in \mathcal{L}$ such that $M|_{X\times X}$ is not pure and $X$ is inclusion-maximal in $\mathcal{L}$
Let $X_0 \subseteq X$ be the set of rows of $M|_{X\times X}$ that contain a 0
$X_1 \leftarrow X \setminus X_0$
$V_X \leftarrow \{v_j \in V(G) \mid L(v_j) = X\}$
if $V_X = \emptyset$ then return $L$
else
    Use the algorithm promised in Lemma 7.25 to compute the list
        $(B_1, C_1), \dots, (B_k, C_k)$ of all bipartite–cobipartite partitions of $G[V_X]$
    for $i \in [k], j \in [n]$ do
        if $v_j \notin V_X$ then
            $L_i(v_j) \leftarrow L(v_j)$
        else if $v_j \in B_i$ then
            $L_i(v_j) \leftarrow X_0$
        else   /* $v_j \in C_i$ */
            $L_i(v_j) \leftarrow X_1$
    return $L_1, \dots, L_k$
```

and $C$ induces the complement of a bipartite graph. The algorithm returns $L_1, \dots, L_k$. $X$ is not in the image of any $L_i$ so, to show that $\{L_1, \dots, L_k\}$ is a correct output for the problem #$\mathcal{L}$-$M$-PURIFY-STEP, we just need to show that every $M$-partition of $G$ that respects $L$ respects exactly one of $L_1, \dots, L_k$. For $i \neq i'$, $(B_i, C_i) \neq (B_{i'}, C_{i'})$ so there is at least one vertex $v_j$ such that $L_i(v_j) = X_0$ and $L_{i'}(v_j) = X_1$ or vice-versa. Since $X_0$ and $X_1$ are disjoint, no $M$-partition can simultaneously respect $L_i$ and $L_{i'}$. It remains to show that every $M$-partition respects at least one of $L_1, \dots, L_k$. To do this, we deduce two structural properties of $M|_{X\times X}$.

First, we show that $M|_{X\times X}$ has no $*$ on its diagonal. Suppose towards a contradiction that $M_{d,d} = *$ for some $d \in X$. If $d \in X_0$, then, for each $d' \in X_1$, $M_{d,d'} = M_{d',d} = *$ because, as noted above, every entry of $M|_{X_0\times X_1}$ is a $*$. Therefore, the $2 \times 2$ matrix $M' = M|_{\{d,d'\}\times\{d,d'\}}$ contains at least three $*$s so it is pure. $\{d, d'\} \subseteq X \in \mathcal{L}$ so, by the hypothesis of the lemma, the length-2 sequence $\{d, d'\}, \{d, d'\}$ is not $\mathcal{L}$-$M$-derectangularising, so $M'$ must be $*$-rectangular, so $M_{d',d'} = *$ for all $d' \in X_1$. Similarly, if $M_{d',d'} = *$ for some $d' \in X_1$, then $M_{d,d} = *$ for all $d \in X_0$. Therefore, if $M|_{X\times X}$ has a $*$ on its diagonal, every entry on the diagonal is $*$. But $M$ contains a 0, say $M_{i,j} = 0$ with $i, j \in X_0$. For any $k \in X_1$,

$$M|_{\{i,j\}\times\{j,k\}} = \begin{bmatrix} 0 & * \\ * & * \end{bmatrix},$$

so the length-2 sequence $\{i, j\}, \{j, k\}$ is $\mathcal{L}$-$M$-derectangularising, contradicting the hypothesis of the lemma (note that $\{i, j\}, \{j, k\} \subseteq X \in \mathcal{L}$).

Second, we show that there is no sequence $d_1, \dots, d_\ell \in X_0$ of odd length such that

$$M_{d_1,d_2} = M_{d_2,d_3} = \cdots = M_{d_{\ell-1},d_\ell} = M_{d_\ell,d_1} = *\,.$$

**Algorithm 7** Case 3 in Algorithm 4.

```
Choose inclusion-maximal $X$ and $Y$ in $\mathcal{L}$ so that $M|_{X\times Y}$ is not pure
Let $X_0 \subseteq X$ be the set of rows of $M|_{X\times Y}$ that contain a 0
$X_1 \leftarrow X \setminus X_0$
Let $Y_0 \subseteq Y$ be the set of columns of $M|_{X\times Y}$ that contain a 0
$Y_1 \leftarrow Y \setminus Y_0$
$V_X \leftarrow \{v_j \in V(G) \mid L(v_j) = X\}$
$V_Y \leftarrow \{v_j \in V(G) \mid L(v_j) = Y\}$
if $V_X = \emptyset$ or $V_Y = \emptyset$ then return $L$
else
    Let $E$ be the set of edges of $G$ between $V_X$ and $V_Y$
    Use the algorithm promised in Lemma 7.27 to produce a subcube decomposition
        $(U_1, U_1'), \dots, (U_k, U_k')$ of $(V_X, V_Y, E)$
    for $i \in [k], j \in [n]$ do
        if $v_j \in V_X$ and the projection of $U_i$ on $v_j$ is $\{0\}$ then
            $L_i(v_j) \leftarrow X_0$
        else if $v_j \in V_X$ and the projection of $U_i$ on $v_j$ is $\{1\}$ then
            $L_i(v_j) \leftarrow X_1$
        else if $v_j \in V_Y$ and the projection of $U_i'$ on $v_j$ is $\{0\}$ then
            $L_i(v_j) \leftarrow Y_0$
        else if $v_j \in V_Y$ and the projection of $U_i'$ on $v_j$ is $\{1\}$ then
            $L_i(v_j) \leftarrow Y_1$
        else
            $L_i(v_j) \leftarrow L(v_j)$
return $L_1, \dots, L_k$
```

Suppose for a contradiction that such a sequence exists. Note that $M|_{X_0\times X_0}$ is $*$-rectangular since $X_0, X_0$ is not an $\mathcal{L}$-$M$-derectangularising sequence and $M|_{X_0\times X_0}$ is pure since Case 1 does not apply. We will show by induction that for every non-negative integer $\kappa \leq (\ell-3)/2$, $M_{d_1,d_{\ell-2\kappa-2}} = *$. This gives a contradiction by taking $\kappa = (\ell-3)/2$ since $M_{d_1,d_1} = *$ and we have already shown that $M|_{X_0\times X_0}$ has no $*$ on its diagonal. For every $\kappa$, the argument follows by considering the matrix $M_\kappa = M|_{\{d_1,d_{\ell-2\kappa-1}\}\times\{d_{\ell-2\kappa-2},d_{\ell-2\kappa}\}}$. The definition of the sequence $d_1,\dots,d_\ell$ together with the symmetry of $M$ guarantees that both entries in row $d_{\ell-2\kappa-1}$ of $M_\kappa$ are $*$. It is also true that $M_{d_1,d_{\ell-2\kappa}} = *$: If $\kappa = 0$ then this follows from the definition of the sequence; otherwise it follows by induction. The fact that $M_{d_1,d_{\ell-2\kappa-2}} = *$ then follows by $*$-rectangularity.

This second structural property implies that, for any $M|_{X\times X}$-partition of $G[V_X]$, the graph induced by vertices assigned to $X_0$ has no odd cycles, and is therefore bipartite. Similarly, the vertices assigned to $X_1$ induce the complement of a bipartite graph. Therefore, any $M$-partition of $G$ that respects $L$ must respect at least one of the $L_1,\dots,L_k$, so it respects exactly one of them, as required.

**Case 3** Since Cases 1 and 2 do not apply and $\mathcal{L}$ is not $M$-purifying, there are distinct $X, Y \in \mathcal{L}$ such that $X$ and $Y$ are inclusion-maximal in $\mathcal{L}$ and $M|_{X\times Y}$ is not pure. As

**Algorithm 8** A trivial algorithm for the problem $\#\mathcal{L}$-$M$-PURIFY for the case in which $\mathcal{L}$ is $M$-purifying.

```
  function #$\mathcal{L}$-$M$-PURIFY($G$,$L$) return $L$
```

in the previous case, the sets $X_0$, $X_1$, $Y_0$ and $Y_1$ are all non-empty.

If either $V_X$ or $V_Y$ is empty then either $X$ or $Y$ is an inclusion-maximal set in $\mathcal{L}$ that is not in the image of $L$ so the output of Algorithm 7 is correct. Otherwise, $(U_1, U_1'), \ldots, (U_k, U_k')$ is a subcube decomposition of the bipartite subgraph $(V_X, V_Y, E)$. The $U_i$s are subcubes of $\{0,1\}^{V_X}$ and the $U_i'$s are subcubes of $\{0,1\}^{V_Y}$. The algorithm returns $L_1, \ldots, L_k$.

Note that if $|U_i'| = 1$ then $Y$ is not in the image of $L_i$. Similarly, if $|U_i'| > 1$ but $|U_i| = 1$ then $X$ is not in the image of $L_i$. The definition of subcube decompositions guarantees that, for every $i$, at least one of these is the case. To show this definition of $L_1, \ldots, L_k$ is a correct output for the problem $\#\mathcal{L}$-$M$-PURIFY-STEP, we must show that any $M$-partition of $G$ that respects $L$ also respects exactly one $L_i$. Since the sets in $\{U_i \times U_i' \mid i \in [k]\}$ are disjoint subsets of $\{0,1\}^{V_X \cup V_Y}$, any $M$-partition of $G$ that respects $L$ respects at most one $L_i$ so it remains to show that every $M$-partition of $G$ respects at least one $L_i$. To do this, we deduce two structural properties of $M|_{X\times Y}$.

First, we show that every entry of $M|_{X_0\times Y_0}$ is 0. The definition of $X_0$ guarantees that every row of $M|_{X_0\times Y_0}$ contains a 0. Since Case 1 does not apply, and $M$ is symmetric, every entry of $M|_{X_0\times Y_0}$ is either 0 or $*$. Suppose for a contradiction that $M_{i,j} = *$ for some $(i,j) \in X_0 \times Y_0$. Pick $i' \in X_1$. For any $j' \in Y_0 \setminus \{j\}$ we have $M_{i,j} = M_{i',j} = M_{i',j'} = *$, so by $*$-rectangularity of $M|_{X\times Y_0}$ we have $M_{i,j'} = *$. Thus, every entry of $M|_{\{i\}\times Y_0}$ is $*$, so there is a $*$ in every $Y_0$-indexed column of $M$. By the same argument, swapping the roles of $X$ and $Y$, every entry in $M|_{X_0\times Y_0}$ is $*$, contradicting the fact that $M|_{X\times Y}$ contains a 0 since $M|_{X\times Y}$ is not pure.

Second, a similar argument shows that every entry of $M|_{X_1\times Y_1}$ is 1.

Thus for all $M$-partitions $\sigma$ of $G$ respecting $L$, for all $x \in V_X$ and $y \in V_Y$, if $(x,y) \in E$ then $(\sigma(x), \sigma(y)) \notin X_0 \times Y_0$ while if $(x,y) \notin E$ then $(\sigma(x), \sigma(y)) \notin X_1 \times Y_1$. Using the definition of subcube decompositions, this shows that any $M$-partition of $G$ respecting $L$ respects some $L_i$. □

We can now give an algorithm for the problem $\#\mathcal{L}$-$M$-PURIFY. The algorithm consists of the function $\#\mathcal{L}$-$M$-PURIFY, which is defined in Algorithm 8 for the trivial case in which $\mathcal{L}$ is $M$-purifying and in Algorithm 9 for the case in which it is not. Note that for any fixed $\mathcal{L}$ and $M$ the algorithm is defined either in Algorithm 8 or in Algorithm 9 and the function $\#\mathcal{L}$-$M$-PURIFY is not recursive. However, the *definition* is recursive, so the function $\#\mathcal{L}$-$M$-PURIFY defined in Algorithm 9 does make a call to a function $\#\mathcal{L}_i$-$M$-PURIFY for some $\mathcal{L}_i$ which is smaller than $\mathcal{L}$. The function $\#\mathcal{L}_i$-$M$-PURIFY is in turn defined in Algorithm 8 or Algorithm 9. The correctness of the algorithm follows from the definition of the problem. The following lemma bounds the

**Algorithm 9** A polynomial-time algorithm for the problem $\#\mathcal{L}$-$M$-PURIFY when $\mathcal{L} \subseteq \mathcal{P}(D)$ is subset-closed and is not $M$-purifying and there is no length-2 $\mathcal{L}$-$M$-derectangularising sequence. This algorithm calls the function $\#\mathcal{L}$-$M$-PURIFY-STEP from Algorithm 4. It also calls the function $\#\mathcal{L}_i$-$M$-PURIFY for various lists $\mathcal{L}_i$ which are shorter than $\mathcal{L}$. These functions are defined inductively in Algorithm 8 and here.

```
function $\#\mathcal{L}$-$M$-PURIFY($G$,$L$)
    // $\emptyset \in \mathcal{L}$ since $\mathcal{L}$ is subset-closed. Since $\mathcal{L}$ is not $M$-purifying, $\mathcal{L} \neq \{\emptyset\}$,
    // hence $|\mathcal{L}| > 1$.
    Let $B$ be the empty sequence of list functions
    $L_1, \ldots, L_k \leftarrow \#\mathcal{L}$-$M$-PURIFY-STEP$(G, L)$
    for $i \in [k]$ do
        $\mathcal{L}_i \leftarrow \bigcup_{v \in V(G)} \mathcal{P}(L_i(v))$
        $L'_1, \ldots, L'_j \leftarrow \#\mathcal{L}_i$-$M$-PURIFY$(G, L_i)$
        Add $L'_1, \ldots, L'_j$ to $B$
    return $B$
```

running time.

**Lemma 7.32.** *Let $M \in \{0, 1, *\}^{D \times D}$ be a symmetric matrix and let $\mathcal{L} \subseteq \mathcal{P}(D)$ be subset-closed. If there is no length-2 $\mathcal{L}$-$M$-derectangularising sequence, then the function $\#\mathcal{L}$-$M$-PURIFY as defined in Algorithms 8 and 9 is a polynomial-time algorithm for the problem $\#\mathcal{L}$-$M$-PURIFY.*

*Proof.* Note that $\mathcal{L}$ is a fixed parameter of the problem $\#\mathcal{L}$-$M$-PURIFY — it is not part of the input. The proof is by induction on $|\mathcal{L}|$. If $|\mathcal{L}| = 1$ then $\mathcal{L} = \{\emptyset\}$ so it is $M$-purifying. In this case, function $\#\mathcal{L}$-$M$-PURIFY is defined in Algorithm 8. It is clear that it is a polynomial-time algorithm for the problem $\#\mathcal{L}$-$M$-PURIFY.

For the inductive step suppose that $|\mathcal{L}| > 1$. If $\mathcal{L}$ is $M$-purifying then function $\#\mathcal{L}$-$M$-PURIFY is defined in Algorithm 8 and again the result is trivial. Otherwise, function $\#\mathcal{L}$-$M$-PURIFY is defined in Algorithm 9. Note that $\mathcal{L} \subseteq \mathcal{P}(D)$ is subset-closed and there is no length-2 $\mathcal{L}$-$M$-derectangularising sequence. From this, we can conclude that, for any subset-closed subset $\mathcal{L}'$ of $\mathcal{L}$, there is no length-2 $\mathcal{L}'$-$M$-derectangularising sequence. So we can assume by the inductive hypothesis that for all subset-closed $\mathcal{L}' \subset \mathcal{L}$, the function $\#\mathcal{L}'$-$M$-PURIFY runs in polynomial time.

The result now follows from the fact that the function $\#\mathcal{L}$-$M$-PURIFY-STEP runs in polynomial time (as guaranteed by Lemma 7.31) and from the fact that each $\mathcal{L}_i$ is a strict subset of $\mathcal{L}$, which follows from the definition of problem $\#\mathcal{L}$-$M$-PURIFY-STEP. Each $M$-partition that respects $L$ respects exactly one of $L_1, \ldots, L_k$ and, hence, it respects exactly one of the list functions that is returned. □

### 7.5.3 Algorithm for $\#\mathcal{L}$-$M$-PARTITIONS and proof of the dichotomy

We can now present our algorithm for the problem $\#\mathcal{L}$-$M$-PARTITIONS. The algorithm consists of the function $\#\mathcal{L}$-$M$-PARTITIONS which is defined in Algorithm 10 for the

**Algorithm 10** A polynomial-time algorithm for the problem #$\mathcal{L}$-$M$-PARTITIONS when $\mathcal{L}$ is subset-closed and $M$-purifying and there is no $\mathcal{L}$-$M$-derectangularising sequence.

```
function #$\mathcal{L}$-$M$-PARTITIONS($G$,$L$)
    $(V, C) \leftarrow$ the instance of #CSP($\Gamma_{\mathcal{L},M}$) obtained by applying the polynomial-
        time Turing reduction from Proposition 7.20 to the input $(G, L)$
    return AC($V, C$) where AC is the function from Algorithm 3
```

**Algorithm 11** A polynomial-time algorithm for the problem #$\mathcal{L}$-$M$-PARTITIONS when $\mathcal{L}$ is subset-closed and not $M$-purifying and there is no $\mathcal{L}$-$M$-derectangularising sequence. The algorithm calls the function #$\mathcal{L}$-$M$-PURIFY$(G, L)$ from Algorithm 9.

```
function #$\mathcal{L}$-$M$-PARTITIONS($G$,$L$)
    $L_1, \ldots, L_t \leftarrow$ #$\mathcal{L}$-$M$-PURIFY$(G, L)$
    $Z \leftarrow 0$
    for $i \in [t]$ do
        $\mathcal{L}_i \leftarrow \bigcup_{v \in V(G)} \mathcal{P}(L_i(v))$
        $(V, C_i) \leftarrow$ the instance of #CSP($\Gamma_{\mathcal{L}_i,M}$) obtained by applying the
            polynomial-time Turing reduction from Proposition 7.20 to the input $(G, L_i)$
        $Z_i \leftarrow$ AC($V, C_i$) where AC is the function from Algorithm 3
        $Z \leftarrow Z + Z_i$
    return $Z$
```

case in which $\mathcal{L}$ is $M$-purifying and in Algorithm 11 when it is not.

**Lemma 7.33.** *Let $M \in \{0, 1, *\}^{D \times D}$ be a symmetric matrix and let $\mathcal{L} \subseteq \mathcal{P}(D)$ be subset-closed. If there is no $\mathcal{L}$-$M$-derectangularising sequence, then the function #$\mathcal{L}$-$M$-PARTITIONS as defined in Algorithms 10 and 11 is a polynomial-time algorithm for the problem #$\mathcal{L}$-$M$-PARTITIONS.*

*Proof.* If $\mathcal{L}$ is $M$-purifying then the function #$\mathcal{L}$-$M$-PARTITIONS is defined in Algorithm 10. Proposition 7.20 shows that the reduction in Algorithm 10 to a CSP instance is correct and takes polynomial time. The CSP instance can be solved by the function AC in Algorithm 3, whose running time is shown to be polynomial in Lemma 7.23.

If $\mathcal{L}$ is not $M$-purifying then the function #$\mathcal{L}$-$M$-PARTITIONS is defined in Algorithm 11. Lemma 7.32 guarantees that the function #$\mathcal{L}$-$M$-PURIFY is a polynomial-time algorithm for the problem #$\mathcal{L}$-$M$-PURIFY. If the list $L_1, \ldots, L_t$ is empty then there is no $M$-partition of $G$ that respects $L$ so it is correct that the function #$\mathcal{L}$-$M$-PARTITIONS returns 0. Otherwise, we know from the definition of the problem #$\mathcal{L}$-$M$-PURIFY that

1. functions $L_1, \ldots, L_t$ are from $V(G)$ to $\mathcal{L}$,
2. for each $i \in [t]$, the set $\{L_i(v) \mid v \in V(G)\}$ is $M$-purifying,
3. for each $i \in [t]$ and $v \in V(G)$, $L_i(v) \subseteq L(v)$, and

4. each $M$-partition of $G$ that respects $L$ respects exactly one of $L_1, \dots, L_t$.

The desired result is now the sum, over all $i \in [t]$, of the number of $M$-partitions of $G$ that respect $L_i$. Since the list $L_1, \dots, L_t$ is generated in polynomial time, $t$ is bounded by some polynomial in $|V(G)|$.

Now, for each $i \in [t]$, $\mathcal{L}_i$ is a subset-closed subset of $\mathcal{L}$. Since there is no $\mathcal{L}$-$M$-derectangularising sequence, there is also no $\mathcal{L}_i$-$M$-derectangularising sequence. Also, $\mathcal{L}_i$ is $M$-purifying. Thus, the argument that we gave for the purifying case shows that $Z_i$ is the desired quantity. □

We can now combine our results to establish our dichotomy for the problem #$\mathcal{L}$-$M$-PARTITIONS.

**Theorem 7.11.** *Let $M$ be a symmetric matrix in $\{0, 1, *\}^{D \times D}$ and let $\mathcal{L} \subseteq \mathcal{P}(D)$ be subset-closed. If there is an $\mathcal{L}$-$M$-derectangularising sequence, then the problem #$\mathcal{L}$-$M$-PARTITIONS is #P-complete. Otherwise, it is in FP.*

*Proof.* Suppose that there is an $\mathcal{L}$-$M$-derectangularising sequence $D_1, \dots, D_k$. Recall (from Definition 7.4) the definition of the subset-closure $\mathcal{S}(\mathcal{L}'')$ of a set $\mathcal{L}'' \subseteq \mathcal{P}(D)$. Let

$$\mathcal{L}' = \mathcal{S}(\{D_1, \dots, D_k\})\,.$$

Since $\{D_1, \dots, D_k\}$ is $M$-purifying, so is $\mathcal{L}'$, which is also subset-closed. It follows that $\Gamma_{\mathcal{L}',M}$ is well defined (see Definition 7.17) and contains each of the relations $H^M_{D_1,D_2}, \dots, H^M_{D_{k-1},D_k}$ (and possibly others). Since $H^M_{D_1,D_2} \circ H^M_{D_2,D_3} \circ \cdots \circ H^M_{D_{k-1},D_k}$ is not rectangular, #CSP($\Gamma_{\mathcal{L}',M}$) is #P-complete [11, Theorem 2 and Corollary 3] (see also [25, Lemma 24]). By Proposition 7.20, the problem #$\mathcal{L}'$-$M$-PARTITIONS is #P-complete so the more general problem #$\mathcal{L}$-$M$-PARTITIONS is also #P-complete. On the other hand, if there is no $\mathcal{L}$-$M$-derectangularising sequence, then the result follows from Lemma 7.33. □

## 7.6 Complexity of the dichotomy criterion

The dichotomy established in Theorem 7.11 shows that, if there exists an $\mathcal{L}$-$M$-derectangularising sequence, then the problem #$\mathcal{L}$-$M$-PARTITIONS is #P-complete; otherwise, it is in FP. This section addresses the computational problem of determining which is the case, given $\mathcal{L}$ and $M$.

The following lemma will allow us to show that the problem EXISTSDERECTSEQ (the problem of determining whether there is an $\mathcal{S}(\mathcal{L})$-$M$-derectangularising sequence, given $\mathcal{L}$ and $M$) and the related problem MATRIXHASDERECTSEQ (the problem of determining whether there is a $\mathcal{P}(D)$-$M$-derectangularising sequence, given $M$) are both in NP. Note that, for this "meta-problem", $\mathcal{L}$ and $M$ are the inputs whereas, previously, we have regarded them as fixed parameters.

**Lemma 7.35.** *Let $M \in \{0,1,*\}^{D\times D}$ be symmetric, and let $\mathcal{L} \subseteq \mathcal{P}(D)$ be subset-closed. If there is an $\mathcal{L}$-$M$-derectangularising sequence, then there is one of length at most* $512(|D|^3+1)$.

*Proof.* Pick an $\mathcal{L}$-$M$-derectangularising sequence $D_1, \dots, D_k$ with $k$ minimal; we will show that $k \leq 512(|D|^3+1)$. Define

$$R = H^M_{D_1,D_2} \circ H^M_{D_2,D_3} \circ \dots \circ H^M_{D_{k-1},D_k}.$$

Note that $R \subseteq D_1 \times D_k$. By the definition of derectangularising sequence, there are $a, a' \in D_1$ and $b, b' \in D_k$ such that $(a,b)$, $(a',b)$ and $(a,b')$ are all in $R$ but $(a',b') \notin R$. So there exist

$$(x_1, \dots, x_k), (y_1, \dots, y_k), (z_1, \dots, z_k) \in D_1 \times \dots \times D_k$$

with $(x_1,x_k) = (a,b)$, $(y_1,y_k) = (a',b)$ and $(z_1,z_k) = (a,b')$ such that $M_{x_i,x_{i+1}} = M_{y_i,y_{i+1}} = M_{z_i,z_{i+1}} = *$ for every $i \in [k-1]$ but, for any $(w_1, \dots, w_k) \in D_1 \times \dots \times D_k$ with $(w_1,w_k) = (a',b')$, there is an $i \in [k-1]$ such that $M_{w_i,w_{i+1}} \neq *$.

Setting $D'_i = \{x_i, y_i, z_i\}$ for each $i$ gives an $\mathcal{L}$-$M$-derectangularising sequence $D'_1, \dots,$ $D'_k$ with $|D'_i| \leq 3$ for each $1 \leq i \leq k$. (Note that any submatrix of a pure matrix is pure.) For all $1 \leq s < t \leq k$ define

$$R_{s,t} = H^M_{D'_s,D'_{s+1}} \circ H^M_{D'_{s+1},D'_{s+2}} \circ \dots \circ H^M_{D'_{t-1},D'_t}.$$

Since $D'_1, \dots, D'_k$ is $\mathcal{L}$-$M$-derectangularising, $R_{1,k}$ is not rectangular but, by the minimality of $k$, every other $R_{s,t}$ is rectangular. Note also that no $R_{s,t} = \emptyset$ since, if that were the case, we would have $R_{1,k} = \emptyset$, which is rectangular.

Suppose for a contradiction that $k > 512(|D|^3+1)$. There are at most $|D|^3+1$ subsets of $D$ with size at most three, so there are indices $1 \leq i_0 < i_1 < i_2 < \dots < i_{512} \leq k$ such that $D'_{i_0} = \dots = D'_{i_{512}}$. There are at most $2^{|D'_{i_0}|^2} - 1 \leq 2^9 - 1 = 511$ non-empty binary relations on $D'_{i_0}$, so $R_{i_0,i_m} = R_{i_0,i_n}$ for some $1 \leq m < n \leq 512$. Since $R_{1,k}$ is not rectangular,

$$R_{1,k} = R_{1,i_0} \circ R_{i_0,i_n} \circ R_{i_n,k} = R_{1,i_0} \circ R_{i_0,i_m} \circ R_{i_n,k} = R_{1,i_m} \circ R_{i_n,k}$$

is not rectangular. Therefore, $D'_1, D'_2, \dots, D'_{i_m}, D'_{1+i_n}, D'_{2+i_n}, \dots, D'_k$ is an $\mathcal{L}$-$M$- derectangularising sequence of length less than $k$, which contradicts the minimality of $k$. □

Now that we have membership in NP, we can prove completeness.

**Theorem 7.13.** ExistsDerectSeq *is* NP*-complete under polynomial-time many-one reductions.*

*Proof.* We first show that ExistsDerectSeq is in NP. Given $D$, a symmetric matrix $M \in \{0, 1, *\}^{D \times D}$ and $\mathcal{L} \subseteq \mathcal{P}(D)$, a non-deterministic polynomial time algorithm for ExistsDerectSeq first "guesses" an $\mathcal{S}(\mathcal{L})$-$M$-derectangularising sequence $D_1, \ldots, D_k$ with $k \leq 512(|D|^3 + 1)$. Lemma 7.35 guarantees that such a sequence exists if the output should be "yes". The algorithm then verifies that each $D_i$ is a subset of a set in $\mathcal{L}$, that $\{D_1, \ldots, D_k\}$ is $M$-purifying, and that the relation $H^M_{D_1,D_2} \circ H^M_{D_2,D_3} \circ \cdots \circ H^M_{D_{k-1},D_k}$ is not rectangular. All of these can be checked in polynomial time without explicitly constructing $\mathcal{S}(\mathcal{L})$.

To show that ExistsDerectSeq is NP-hard, we give a polynomial-time reduction from the well-known NP-hard problem of determining whether a graph $G$ has an independent set of size $k$.

Let $G$ and $k$ be an input to the independent set problem. Let $V(G) = [n]$ and assume without loss of generality that $k \in [n]$. Setting $D = [n] \times [k] \times [3]$, we construct a $D \times D$ matrix $M$ and a set $\mathcal{L}$ of lists such that there is an $\mathcal{S}(\mathcal{L})$-$M$-derectangularising sequence if and only if $G$ has an independent set of size $k$.

$M$ will be a block matrix, constructed using the following $3 \times 3$ symmetric matrices. Note that each is pure, apart from Id.

$$M_{\text{start}} = \begin{bmatrix} * & * & 0 \\ * & * & 0 \\ 0 & 0 & * \end{bmatrix} \qquad M_{\text{end}} = \begin{bmatrix} * & 0 & 0 \\ 0 & * & * \\ 0 & * & * \end{bmatrix} \qquad M_{\text{bij}} = \begin{bmatrix} * & 0 & 0 \\ 0 & * & 0 \\ 0 & 0 & * \end{bmatrix}$$

$$\mathbf{0} = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \qquad \text{Id} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}.$$

For $v \in [n]$ and $j \in [k]$, let $D[v, j] = \{(v, j, c) \mid c \in [3]\}$. Below, when we say that $M|_{D[v,j] \times D[v',j']} = N$ for some $3 \times 3$ matrix $N$, we mean more specifically that $M_{(v,j,c),(v',j',c')} = N_{c,c'}$ for all $c, c' \in [3]$. $M$ is constructed as follows.

1. For all $v \in [n]$, $M|_{D[v,1] \times D[v,1]} = M_{\text{start}}$ and $M|_{D[v,k] \times D[v,k]} = M_{\text{end}}$.

2. For all $v \in [n]$ and all $j \in \{2, \ldots, k-1\}$, $M|_{D[v,j] \times D[v,j]} = M_{\text{bij}}$.

3. If $v \neq v'$, $(v, v') \notin E(G)$ and $j < k$, then

   (a) $M|_{D[v,j] \times D[v',j+1]} = M|_{D[v',j+1] \times D[v,j]} = M_{\text{bij}}$ and

   (b) $M|_{D[v,j] \times D[v',j']} = M|_{D[v',j'] \times D[v,j]} = \mathbf{0}$ for all $j' > j + 1$.

4. For all $v, v' \in [n]$ and $j, j' \in [k]$ not covered above, $M|_{D[v,j] \times D[v',j']} = \text{Id}$.

To complete the construction, let $\mathcal{L} = \{D[v, j] \mid v \in [n], j \in [k]\}$. We will show that $G$ has an independent set of size $k$ if and only if there is an $\mathcal{S}(\mathcal{L})$-$M$-derectangularising sequence.

For the forward direction of the proof, suppose that $G$ has an independent set $I = \{v_1, \ldots, v_k\}$ of size $k$. We will show that

$$D[v_1, 1], D[v_1, 1], D[v_2, 2], D[v_3, 3], \ldots, D[v_{k-1}, k-1], D[v_k, k], D[v_k, k]$$

(where the first and last elements are repeated and the others are not) is $\mathcal{S}(\mathcal{L})$-$M$-derectangularising. Since there is no edge $(v_i, v_{i'}) \in E(G)$ for $i, i' \in [k]$, the matrix $M|_{D[v_i,i]\times D[v_{i'},i']}$ is always one of $M_{\text{start}}$, $M_{\text{end}}$, $M_{\text{bij}}$ and $\mathbf{0}$, so it is always pure. Therefore, $\{D[v_1, 1], \ldots, D[v_k, k]\}$ is $M$-purifying. It remains to show that the relation

$$R = H^M_{D[v_1,1],D[v_1,1]} \circ H^M_{D[v_1,1],D[v_2,2]} \circ \cdots \circ H^M_{D[v_{k-1,k-1}],D[v_k,k]} \circ H^M_{D[v_k,k],D[v_k,k]}$$

is not rectangular.

Consider $i \in [k-1]$. Since $(v_i, v_{i+1}) \notin E(G)$, $M|_{D[v_i,i]\times D[v_{i+1},i+1]} = M_{\text{bij}}$ so $H^M_{D[v_i,i],D[v_{i+1},i+1]}$ is the bijection that associates $(v_i, i, c)$ with $(v_{i+1}, i+1, c)$ for each $c \in [3]$. Therefore,

$$H^M_{D[v_1,1],D[v_1,2]} \circ \cdots \circ H^M_{D[v_{k-1},k-1],D[v_k,k]}$$

is the bijection that associates $(v_1, 1, c)$ with $(v_k, k, c)$ for each $c \in [3]$. We have $M|_{D[v_1,1]\times D[v_1,1]} = M_{\text{start}}$ and $M|_{D[v_k,k]\times D[v_k,k]} = M_{\text{end}}$ so

$$\begin{aligned} H^M_{D[v_1,1],D[v_1,1]} &= \{((v_1, 1, c), (v_1, 1, c')) \mid c, c' \in [2]\} \cup \{((v_1, 1, 3), (v_1, 1, 3))\} \\ H^M_{D[v_k,k],D[v_k,k]} &= \{((v_k, k, 1), (v_k, k, 1))\} \cup \{((v_k, k, c), (v_k, k, c')) \mid c, c' \in \{2, 3\}\}, \end{aligned}$$

and, therefore,

$$R = \{((v_1, 1, c), (v_k, k, c')) \mid c, c' \in [3]\} \setminus \{((v_1, 1, 3), (v_k, k, 1))\},$$

which is not rectangular, as required.

For the reverse direction of the proof, suppose that there is an $\mathcal{S}(\mathcal{L})$-$M$-derectangularising sequence $D_1, \ldots, D_m$. The fact that the sequence is derectangularising implies that $|D_i| \geq 2$ for each $i \in [m]$ — see the remarks following Definition 7.10. Each set in the sequence is a subset of some $D[v, j]$ in $\mathcal{L}$ so for every $i \in [m]$ let $v_i$ denote the vertex in $[n]$ and let $j_i$ denote the index in $[k]$ such that $D_i \subseteq D[v_i, j_i]$. Clearly, it is possible to have $(v_i, j_i) = (v_{i'}, j_{i'})$ for distinct $i$ and $i'$ in $[m]$.

We will finish the proof by showing that $G$ has a size-$k$ independent set. Let

$$R = H^M_{D_1,D_2} \circ \cdots \circ H^M_{D_{m-1},D_m},$$

which is not rectangular because the sequence is $\mathcal{S}(\mathcal{L})$-$M$-derectangularising. Since $\{D_1, \ldots, D_m\}$ is $M$-purifying, and any submatrix of Id with at least two rows and at least two columns is impure, every pair $(i, i') \in [m]^2$ satisfies $M|_{D[v_i,j_i]\times D[v_{i'},j_{i'}]} \neq \text{Id}$.

This means that we cannot have $(v_i, v_{i'}) \in E(G)$ for any pair $(i, i') \in [m]^2$ so the set $I = \{v_1, \dots, v_m\}$ is independent in $G$. It remains to show that $|I| \geq k$.

Observe that, if $v_i = v_{i'}$, we must have $j_i = j_{i'}$ since, otherwise, the construction ensures that

$$M|_{D[v_i, j_i] \times D[v_{i'}, j_{i'}]} = M|_{D[v_i, j_i] \times D[v_i, j_{i'}]} = \mathrm{Id}\,,$$

which we already ruled out. Therefore, $|I| \geq |\{j_1, \dots, j_m\}|$.

We must have $|j_i - j_{i+1}| \leq 1$ for each $i \in [m-1]$ as, otherwise, we would have $M|_{D[v_i, j_i] \times D[v_{i+1}, j_{i+1}]} = \mathbf{0}$, which implies that $R = \emptyset$, which is rectangular. There must be at least one $i \in [m-1]$ such that $v_i = v_{i+1}$ and $j_i = j_{i+1} = 1$, so $M|_{D[v_i, j_i] \times D[v_{i+1}, j_{i+1}]} = M_{\mathrm{start}}$. If not, $R$ is a composition of relations corresponding to $M_{\mathrm{bij}}$ and $M_{\mathrm{end}}$ and any such relation is either a bijection, or of the form of $M_{\mathrm{end}}$, so it is rectangular. Similarly, there must be at least one $i$ such that $v_i = v_{i+1}$ and $j_i = j_{i+1} = k$, giving $M|_{D[v_i, j_i] \times D[v_{i+1}, j_{i+1}]} = M_{\mathrm{end}}$. Therefore, the sequence $j_1, \dots, j_m$ contains 1 and $k$. Since $|j_i - j_{i+1}| \leq 1$ for all $i \in [m-1]$, it follows that $[k] \subseteq \{j_1, \dots, j_m\}$, so $|I| \geq k$, as required. In fact, $\{j_1, \dots, j_m\} = [k]$ since each $j_i \in [k]$ by construction. □

We defined the problem ExistsDerectSeq using a concise input representation: $\mathcal{S}(\mathcal{L})$ does not need to be written out in full. Instead, the instance is a subset $\mathcal{L}$ containing the maximal elements of $\mathcal{S}(\mathcal{L})$. For example, when the instance is $\mathcal{L} = \{D\}$, we have $\mathcal{S}(\mathcal{L}) = \mathcal{P}(D)$. It is important to note that the $\mathsf{NP}$-completeness of ExistsDerectSeq is not an artifact of this concise input coding. The elements of the list $\mathcal{L}$ constructed in the NP-hardness proof have length at most three, so the list $\mathcal{S}(\mathcal{L})$ could also be constructed explicitly in polynomial time.

Lemma 7.35 has the following immediate corollary for the complexity of the dichotomy criterion of the general #List-$M$-partitions problem. Recall that, in this version of the meta-problem, the input is just the matrix $M$.

**Corollary 7.15.** MatrixHasDerectSeq *is in* $\mathsf{NP}$.

*Proof.* Take $\mathcal{L} = \{D\}$ in Lemma 7.35. □

## 7.7 Cardinality constraints

Finally, we show how lists can be used to implement cardinality constraints of the kind that often appear in counting problems in combinatorics.

Feder, Hell, Klein and Motwani [37] point out that lists can be used to determine whether there are $M$-partitions that obey simple cardinality constraints. For example, it is natural to require some or all of the parts to be non-empty or, more generally, to contain at least some constant number of vertices. Given a $D \times D$ matrix $M$, we represent such cardinality constraints as a function $C\colon D \to \mathbb{N}$. We say that an $M$-partition $\sigma$ of a graph $G$ *satisfies* the constraint if, for each $d \in D$, $|\{v \in V(G) \mid \sigma(v) = d\}| \geq C(d)$. Given a cardinality constraint $C$, we write $|C| = \sum_{d \in D} C(d)$.

We can determine whether there is an $M$-partition of $G = (V, E)$ that satisfies the cardinality constraint $C$ by making at most $|V|^{|C|}$ queries to an oracle for the list $M$-partitions problem, as follows. Let $L_C$ be the set of list functions $L\colon V \to \mathcal{P}(D)$ such that:

1. for all $v \in V$, either $L(v) = D$ or $|L(v)| = 1$, and
2. for all $d \in D$, there are exactly $C(d)$ vertices $v$ with $L(v) = \{d\}$.

There are at most $|V|^{|C|}$ such list functions and it is clear that $G$ has an $M$-partition satisfying $C$ if, and only if, it has a list $M$-partition that respects at least one $L \in L_C$. The number of queries is polynomial in $|V|$ as long as the cardinality constraint $C$ is independent of $G$.

For counting, the situation is a little more complicated, as we must avoid double-counting. The solution is to count all $M$-partitions of the input graph and subtract off those that fail to satisfy the cardinality constraint. We formally define the problem #$C$-$M$-PARTITIONS as follows, parameterized by a $D \times D$ matrix $M$ and a cardinality constraint function $C\colon D \to \mathbb{N}$.

**Problem 7.38.** *Name.* #$C$-$M$-PARTITIONS.

*Parameter.* A symmetric matrix $M$ in $\{0, 1, *\}^{D\times D}$ and $C\colon D \to \mathbb{N}$.

*Input.* A graph $G$.

*Output.* The number of $M$-partitions of $G$ that satisfy $C$.

**Proposition 7.39.** *#$C$-$M$-PARTITIONS reduces to #LIST-$M$-PARTITIONS under polynomial-time Turing reductions.*

*Proof.* Given the cardinality constraint function $C$, let $R = \{d \in D \mid C(d) > 0\}$: that is, $R$ is the set of parts that have a non-trivial cardinality constraint. For any set $P \subseteq R$, say that an $M$-partition $\sigma$ of a graph $G = (V, E)$ *fails on* $P$ if $|\{v \in V \mid \sigma(v) = d\}| < C(d)$ for all $d \in P$. That is, if $\sigma$ violates the cardinality constraints on all parts in $P$ (and possibly others, too). Let $\Sigma$ be the set of all $M$-partitions of our given input graph $G$. For $i \in R$, let $A_i = \{\sigma \in \Sigma \mid \sigma \text{ fails on } \{i\}\}$ and let $A = \bigcup_{i\in R} A_i$. By inclusion-exclusion,

$$\begin{aligned}|A| &= -\sum_{\emptyset\subset P\subseteq R} (-1)^{|P|}\Big|\bigcap_{i\in P} A_i\Big| \\ &= -\sum_{\emptyset\subset P\subseteq R} (-1)^{|P|}\Big|\{\sigma \in \Sigma \mid \sigma \text{ fails on } P\}\Big|.\end{aligned}$$

We wish to compute

$$\begin{aligned}\Big|\{\sigma \in \Sigma \mid \sigma \text{ satisfies } C\}\Big| &= \Big|\Sigma\Big| - |A| \\ &= \Big|\Sigma\Big| + \sum_{\emptyset \subset P \subseteq R} (-1)^{|P|} \Big|\{\sigma \in \Sigma \mid \sigma \text{ fails on } P\}\Big| .\end{aligned}$$

Therefore, it suffices to show that we can use lists to count the $M$-partitions that fail on each non-empty $P \subseteq R$. For such a set $P$, let $L_P$ be the set of list functions $L$ such that

1. for all $v \in V$, either $L(v) = D \setminus P$ or $L(v) = \{p\}$ for some $p \in P$, and
2. for all $p \in P$, $\Big|\Big\{v \in V \mid L(v) = \{p\}\Big\}\Big| < C(p)$.

Thus, the set of $M$-partitions that respect some $L \in L_P$ is precisely the set of $M$-partitions that fail on $P$. Also, for distinct $L$ and $L'$ in $L_P$, the set of $M$-partitions that respect $L$ is disjoint from the set of $M$-partitions that respect $L'$. So we can compute $\Big|\{\sigma \in \Sigma \mid \sigma \text{ fails on } P\}\Big|$ by making $|L_P|$ calls to #LIST-$M$-PARTITIONS, noting that $|L_P| \le |V|^{|C|}$. □

As an example of a combinatorial structure that can be represented as an $M$-partition problem with cardinality constraints, consider the *homogeneous pairs* introduced by Chvátal and Sbihi [20]. A homogeneous pair in a graph $G = (V, E)$ is a partition of $V$ into sets $U$, $W_1$ and $W_2$ such that:

1. $|U| \ge 2$;
2. $|W_1| \ge 2$ or $|W_2| \ge 2$ (or both);
3. for every vertex $v \in U$, $v$ is either adjacent to every vertex in $W_1$ or to none of them; and
4. for every vertex $v \in U$, $v$ is either adjacent to every vertex in $W_2$ or to none of them.

Feder et al. [37] observe that the problem of determining whether a graph has a homogeneous pair can be represented as the problem of determining whether it has an $M_{\text{hp}}$-partition satisfying certain constraints, where $D = \{1, \dots, 6\}$ and

$$M_{\text{hp}} = \begin{bmatrix} * & * & 1 & 0 & 1 & 0 \\ * & * & 1 & 1 & 0 & 0 \\ 1 & 1 & * & * & * & * \\ 0 & 1 & * & * & * & * \\ 1 & 0 & * & * & * & * \\ 0 & 0 & * & * & * & * \end{bmatrix} .$$

$W_1$ corresponds to the set of vertices mapped to part 1 (row 1 of $M_{\text{hp}}$), $W_2$ corresponds to the set of vertices mapped to part 2 (row 2 of $M_{\text{hp}}$), and $U$ corresponds to the set of vertices mapped to parts 3–6.

In fact, there is a one-to-one correspondence between the homogeneous pairs of $G$ in which $W_1$ and $W_2$ are non-empty and the $M_{\text{hp}}$-partitions $\sigma$ of $G$ that satisfy the following additional constraints. For $d \in D$, let $N_\sigma(d) = |\{v \in V(G) \mid \sigma(v) = d\}|$ be the number of vertices that $\sigma$ maps to part $d$. We require that

1. $N_\sigma(3) + N_\sigma(4) + N_\sigma(5) + N_\sigma(6) \geq 2$,

2. $N_\sigma(1) > 0$ and $N_\sigma(2) > 0$, and

3. at least one $N_\sigma(1)$ and $N_\sigma(2)$ is at least 2.

To see this, consider a homogeneous pair $(U, W_1, W_2)$ in which $W_1$ and $W_2$ are non-empty. Note that there is exactly one $M_{\text{hp}}$-partition of $G$ in which vertices in $W_1$ are mapped to part 1 and vertices in $W_2$ are mapped to part 2 and vertices in $U$ are mapped to parts 3–6. There is exactly one part available to each $v \in U$ since $v$ has an edge or non-edge to $W_1$ (but not both!) ruling out exactly two parts and $v$ has an edge or non-edge to $W_2$ ruling out an additional part. Going the other way, an $M_{\text{hp}}$-partition that satisfies the constraints includes a homogeneous pair.

Now let

$$M_{\text{hs}} = \begin{bmatrix} * & 0 & 1 \\ 0 & * & * \\ 1 & * & * \end{bmatrix}.$$

There is a one-to-one correspondence between the homogeneous pairs of $G$ in which $W_2$ is empty and the $M_{\text{hs}}$-partitions of $G$ that satisfy the following additional constraints.

1. At least two vertices are mapped to parts 2–3 (vertices in these parts are in $U$).

2. At least two vertices are mapped to part 1 (vertices in this part are in $W_1$).

Symmetrically, there is also a one-to-one correspondence between the homogeneous pairs of $G$ in which $W_1$ is empty and the $M_{\text{hs}}$-partitions of $G$ that satisfy the above constraints. (Partitions according to $M_{\text{hs}}$ correspond to so-called “homogeneous sets” but we do not need the details of these.)

It is known from [28] that, in deterministic polynomial time, it is possible to determine whether a graph contains a homogeneous pair and, if so, to find one. We show that the homogeneous pairs in a graph can also be counted in polynomial time. We start by considering the relevant list-partition counting problems.

**Theorem 7.40.** *There are polynomial-time algorithms for* #LIST-$M_{\text{hp}}$-PARTITIONS *and* #LIST-$M_{\text{hs}}$-PARTITIONS.

*Proof.* We first show that there is a polynomial-time algorithm for the problem #List-$M_{\mathrm{hp}}$-partitions. The most natural way to do this would be to show that there is no $\mathcal{P}(D)$-$M_{\mathrm{hp}}$-derectangularising sequence and then apply Theorem 7.11. In theory, we could show that there is no $\mathcal{P}(D)$-$M_{\mathrm{hp}}$-derectangularising sequence by brute force since $|D| = 6$, but the number of possibilities is too large to make this feasible. Instead, we argue non-constructively.

First, if there is no $\mathcal{P}(D)$-$M_{\mathrm{hp}}$-derectangularising sequence, the result follows from Theorem 7.11.

Conversely, suppose that $D_1, \ldots, D_k$ is a $\mathcal{P}(D)$-$M_{\mathrm{hp}}$-derectangularising sequence. Let $M$ be the matrix such that $M_{i,j} = 0$ if $(M_{\mathrm{hp}})_{i,j} = 1$ and $M_{i,j} = (M_{\mathrm{hp}})_{i,j}$, otherwise. $D_1, \ldots, D_k$ is also a $\mathcal{P}(D)$-$M$-derectangularising sequence, since $H^{M}_{X,Y} = H^{M_{\mathrm{hp}}}_{X,Y}$ for any $X, Y \subseteq D$ and any sequence $D_1, \ldots, D_k$ is $M$-purifying because $M$ is already pure. Therefore, by Theorem 7.11, counting list $M$-partitions is $\#\mathsf{P}$-complete.

However, counting the list $M$-partitions of a graph $G$ corresponds to counting list homomorphisms from $G$ to the 6-vertex graph $H$ whose two components are an edge and a 4-clique, and which has self-loops on all six vertices. There is a very straightforward polynomial-time algorithm for this problem (a simple modification of the version without lists in [27]). Thus, $\#\mathsf{P} = \mathsf{FP}$ so, in particular, there is a polynomial-time algorithm for counting list $M_{\mathrm{hp}}$-partitions.

The proof that there is a polynomial-time algorithm for #List-$M_{\mathrm{hs}}$-partitions is similar. □

**Corollary 7.41.** *There is a polynomial-time algorithm for counting the homogeneous pairs in a graph.*

*Proof.* We are given a graph $G = (V, E)$ and we wish to compute the number of homogeneous pairs that it contains. By the one-to-one correspondence given earlier, it suffices to show how to count $M_{\mathrm{hp}}$-partitions and $M_{\mathrm{hs}}$-partitions of $G$ satisfying additional constraints. We start with the first of these. Recall the constraints on the $M_{\mathrm{hp}}$-partitions $\sigma$ that we wish to count:

1. $N_\sigma(3) + N_\sigma(4) + N_\sigma(5) + N_\sigma(6) \geq 2$,
2. $N_\sigma(1) > 0$ and $N_\sigma(2) > 0$, and
3. at least one $N_\sigma(1)$ and $N_\sigma(2)$ is at least 2.

Define three subsets $\Sigma_1$, $\Sigma_2$ and $\Sigma_{1,2}$ of the set of $M_{\mathrm{hp}}$-partitions of $G$ that satisfy the constraints. In the definition of each of $\Sigma_1$, $\Sigma_2$ and $\Sigma_{1,2}$, we will require that parts 1 and 2 are non-empty and parts 3–6 contain a total of at least two vertices. In $\Sigma_1$, part 1 must contain at least two vertices; in $\Sigma_2$, part 2 must contain at least two vertices; in $\Sigma_{1,2}$, both parts 1 and 2 must contain at least two vertices. The number of suitable $M_{\mathrm{hp}}$-partitions of $G$ is $|\Sigma_1| + |\Sigma_2| - |\Sigma_{1,2}|$.

Each of $|\Sigma_1|$, $|\Sigma_2|$ and $|\Sigma_{1,2}|$ can be computed by counting the $M_{\mathrm{hp}}$-partitions of $G$ that satisfy appropriate cardinality constraints. Parts 1 and 2 are trivially dealt with. The requirement that parts 3–6 must contain at least two vertices between them is equivalent to saying that at least one of them must contain at least two vertices or at least two must contain at least one vertex. This can be expressed with a sequence of cardinality constraint functions and using inclusion–exclusion to eliminate double-counting.

Counting constrained $M_{\mathrm{hs}}$-partitions of $G$ is similar (but simpler). $\square$

# Part III

# Evolutionary dynamics

# Chapter 8

# The Moran process on superstars

The results of this chapter are published in the paper "Amplifiers for the Moran Process" co-authored with Andreas Galanis, Leslie Goldberg, John Lapinskas and David Richerby [41].

## 8.1 Introduction

In this chapter we study the extinction probability of the Moran algorithm on a family of directed graphs called the superstars. Recall from Section 1.4 that the Moran algorithm has a parameter $r$ which is the fitness of "mutants". All non-mutants have fitness 1. The algorithm runs on a directed graph. In the initial state, one vertex is chosen uniformly at random to become a mutant. After this, the algorithm runs in discrete steps as follows. At each step, a vertex is selected at random, with probability proportional to its fitness. Suppose that this is vertex $v$. Next, an out-neighbour $w$ of $v$ is selected uniformly at random. Finally, the state of vertex $v$ (mutant or non-mutant) is copied to vertex $w$. If the graph is finite and strongly connected then with probability 1, the algorithm will either reach the state where there are only mutants (known as *fixation*) or it will reach the state where there are only non-mutants (known as *extinction*).

To state our main result for this chapter, recall the definition of superstars.

**Definition 1.21.** Let $k$, $\ell$ and $m$ be positive integers. The $(k,\ell,m)$-*superstar* is the directed graph $\mathcal{S}_{k,\ell,m}$ defined as follows. (See Figure 1.2, page 18.) The vertex set $V(\mathcal{S}_{k,\ell,m})$ of $\mathcal{S}_{k,\ell,m}$ is the disjoint union of $\ell$ size-$m$ sets $R_1,\dots,R_\ell$ (called *reservoirs*) together with $k\ell$ vertices $v_{1,1}, v_{1,2},\dots,v_{\ell,k}$ and a single centre vertex $v^*$. The edge set of $\mathcal{S}_{k,\ell,m}$ is given by

$$E(\mathcal{S}_{k,\ell,m}) = \bigcup_{i=1}^{\ell}\left((\{v^*\}\times R_i)\cup(R_i\times\{v_{i,1}\})\cup\{(v_{i,j},v_{i,j+1}) \mid j\in[k-1]\}\cup\{(v_{i,k},v^*)\}\right).$$

We prove the following theorem about superstars.

**Theorem 1.22.** *Let $\zeta(r,n)$ be any function such that, for any $r > 1$,*

$$\lim_{n\to\infty} \zeta(r,n)(n\log n)^{1/3} = 0.$$

*Then there is no infinite family of superstars that is up-to-$\zeta$ fixating.*

### 8.1.1 Proof techniques

As we have seen in Section 1.4, it is easy to study the Moran process on a $d$-regular graph by considering the transition matrix of the corresponding Markov chain (which looks like a one-dimensional random walk). Highly symmetric graphs such as undirected stars can also be handled in a straightforward matter, by directly analysing the transition matrix. Superstars are more complicated and the number of mutant-configurations is exponential, so instead we resort to dividing the process into phases, as is typical in the study of stochastic processes.

An essential and common trick in the area of stochastic processes (for example, in work on the voter model) is moving to continuous time. Instead of directly studying the discrete-time Moran process, one could consider the following natural continuous-time model which was studied in [24]: Given a set of mutants at time $t$, each vertex waits an amount of time before reproducing. For each vertex, the period of time is chosen according to the exponential distribution with parameter equal to the vertex's fitness, independently of the other vertices. If the first vertex to reproduce is $v$ at time $t + \tau$ then, as in the standard, discrete-time version of the process, one of its out-neighbours $w$ is chosen uniformly at random, the individual at $w$ is replaced by a copy of the one at $v$, and the time at which $w$ will next reproduce is exponentially distributed with parameter given by its new fitness. The discrete-time process is recovered by taking the sequence of configurations each time a vertex reproduces. Thus, the fixation probability of the discrete-time process is *exactly the same* as the fixation probability of the continuous-time process. So moving to the continuous-time model causes no harm. As [24] explains, analysis can be easier in the continuous-time model because certain natural stochastic domination techniques apply in the continuous-time setting but not in the discrete-time setting.

It turns out that moving to the model of [24] does not suffice for our purposes. A major problem in our proofs is dealing with dependencies. In order to make this feasible, we instead study a continuous-time model (see "the clock process" in Section 8.3.1) in which every edge of the underlying graph $G$ is equipped with two Poisson processes, one of which is called a *mutant clock* and the other of which is called a *non-mutant clock*. The clock process is a stochastic process in which all of these clocks run independently. The continuous-time Moran process (Definition 8.7) can be recovered as a function of the times at which these clocks trigger.

Having all of these clocks available still does not give us the flexibility that we need.

We say that a vertex $u$ "spawns a mutant" in the Moran process if, at some point in time, $u$ is a mutant, and it is selected for reproduction. We wish to be able to discuss events such as the event that the vertex $u$ does not spawn a mutant until it has already been a mutant for some particular amount of time. In order to express such events in a clean way, making all conditioning explicit, we define additional stochastic processes called "star-clocks" (see Section 8.3.3). All of the star-clocks run independently in the star-clock process.

In Section 8.3.4 we provide a coupling[1] of the star-clock process with the Moran process. The coupling is valid in the sense that the two projections are correct — the projection onto the Moran process runs according to the correct distribution and so does the projection onto the star-clock process. The point of the coupling is that the different star-clocks can be viewed as having their own "local" times. In particular, there is a star-clock $M^*_{(u,v)}$ which controls reproductions from vertex $u$ onto vertex $v$ *during the time that $u$ is a mutant.* The coupling enables us to focus on relevant parts of the stochastic process, making all conditioning explicit.

The processes that we have described so far are all that we need to derive our upper bound on the fixation probability of superstars (Section 8.4).

### 8.1.2 Organisation

In Section 8.2, we define some notation and state some well-known probabilistic bounds (Chernoff bounds and analysis of gambler's ruin) which will be used in the proof. In Section 8.3 we define several stochastic processes which we use to study the Moran process. Finally, in Section 8.4 we give an upper bound on the fixation probability of superstars. The main result of the section is Theorem 8.13, which immediately implies Theorem 1.22.

## 8.2 Definitions and preliminaries

### 8.2.1 Notation

We use $N^-(v)$ to refer to the set of in-neighbours of a vertex $v$ and $N^+(v)$ to refer to the set of out-neighbours of $v$. We use $d^-(v) = |N^-(v)|$ and $d^+(v) = |N^+(v)|$.

We refer to the Lebesgue measure of a (measurable) subset $S \subseteq \mathbb{R}$ as the *measure* of that set, and denote it by $\text{len}(S)$.

We use base $e$ for logarithms unless the base is given explicitly.

We write $\mathbb{Z}_{\geq 0} = \{0, 1, 2, \dots\}$, $\mathbb{Z}_{\geq 1} = \{1, 2, \dots\}$, and $[n] = \{1, 2, \dots, n\}$.

If $b < a$, we consider the interval $[a, b]$ to be well-defined but empty. Likewise if $b \leq a$, we consider the intervals $(a, b)$, $(a, b]$ and $[a, b)$ to be well-defined but empty.

[1]For a definition of a coupling see e.g. [63, Definition 11.3]

We define empty sums, products, unions etc. to be the identities of the corresponding operations. For example , $\prod_{i=1}^{0} i = 1$ and $\bigcup_{i=1}^{0} A_i = \emptyset$.

Throughout the chapter, we use lower case $t$'s to denote fixed times and upper case $T$'s to denote stopping times.

### 8.2.2 Chernoff bounds

We often use the following simple bound which applies to any real number $x \in [0, 1]$.

$$x/2 \leq 1 - e^{-x} \leq x. \tag{8.1}$$

We will require the following well-known Chernoff bounds. The first appears in [63, Theorem 5.4].

**Lemma 8.3.** *Let $Y$ be a Poisson random variable with parameter $\rho \geq 0$. If $y > \rho$ and $z < \rho$, then*

$$\mathbb{P}(Y \geq y) \leq \frac{e^{-\rho}(e\rho)^y}{y^y} \qquad \text{and} \qquad \mathbb{P}(Y \leq z) \leq \frac{e^{-\rho}(e\rho)^z}{z^z}.$$

**Corollary 8.4.** *Let $Y$ be a Poisson random variable with parameter $\rho \geq 0$. Then $\mathbb{P}(Y \geq 2\rho) \leq e^{-\rho/3}$ and $\mathbb{P}(Y \leq 2\rho/3) \leq e^{-\rho/16}$.*

*Proof.* Lemma 8.3 applied with $y = 2\rho$ and $z = 2\rho/3$ implies that

$$\mathbb{P}(Y \geq 2\rho) \leq \frac{e^{-\rho}(e\rho)^{2\rho}}{(2\rho)^{2\rho}} = e^{-\rho}\left(\frac{e^2}{4}\right)^{\rho} = e^{(1-\log 4)\rho} \leq e^{-\rho/3},$$

$$\mathbb{P}(Y \leq 2\rho/3) \leq \frac{e^{-\rho}(e\rho)^z}{z^z} = e^{-\rho}\left(\frac{3e}{2}\right)^{2\rho/3} = e^{-(1-2/3-2\log(3/2)/3)\rho} \leq e^{-\rho/16}. \qquad \square$$

**Corollary 8.5.** *Let $s$ be a positive integer and let $Y$ be the sum of $s$ i.i.d. exponential random variables, each with parameter $\lambda$. Then, for any $j \geq 3s/(2\lambda)$, $\mathbb{P}(Y < j) \geq 1 - e^{-\lambda j/16}$.*

*Proof.* First, note that $\mathbb{P}(Y < j) = \mathbb{P}(Y \leq j)$ since $\mathbb{P}(Y = j) = 0$. Then $\mathbb{P}(Y \leq j)$ is equal to the probability that a Poisson process with parameter $\lambda$ triggers at least $s$ times in the interval $[0, j]$. This is the same as the probability that a Poisson random variable with parameter $\lambda j$ is at least $s$. Since $s \leq 2\lambda j/3$, we can now use Corollary 8.4. $\square$

The following is in [56, Corollary 2.4].

**Lemma 8.6.** *Suppose that $Y$ follows the binomial distribution with $n$ Bernoulli trials, each with success probability $p \in (0, 1)$ and that $c > 1$. Then, for all $y \geq cnp$, $\mathbb{P}(Y \geq y) \leq e^{-\varphi(c)y}$, where $\varphi(c) = \log c - 1 + 1/c$. Note that $\varphi(2) > 1/6$ and $\varphi(7) > 1$.*

## 8.3 Stochastic processes

We will be concerned with the discrete-time Moran process [64], as adapted by Lieberman, Hauert and Nowak [59] and described in the introduction of this chapter. This is a discrete model of evolution on an underlying directed graph $G$ where the reproduction rate of mutants is a parameter $r > 0$ called the "fitness".

In this thesis, we consider the situation $r > 1$, which corresponds to the situation in which a mutation is advantageous. The fitness $r$ is a parameter of all of our processes. Our results apply to any fixed $r > 1$. Since the value of $r$ is fixed, we simplify the presentation by not including it in the explicit notation and terminology. Thus, from now on, we say "Moran process" to signify "Moran process with fitness $r$".

Following [24] we will simplify our proofs by studying a continuous-time version of the Moran process. The continuous-time version is also parametrised by $G$ and $r$ and it has the same fixation probability as the discrete-time version, so our results will carry over immediately to the discrete process.

In order to deal with conditioning in the proofs we will in fact define several general stochastic processes, all of which depend on $G$ and $r$ — one of these will be equivalent to the continuous-time Moran process and others will be useful for dominations.

All of the processes that we study evolve over time. For any process $P$, we use $\mathcal{F}(P)$ to denote the filtration of $P$ so $\mathcal{F}_t(P)$ captures the history of the process $P$ up to and including time $t$.

### 8.3.1 The clock process

For each edge $e = (u, v)$ of $G$ we define two Poisson processes — a Poisson process $M_e$ with parameter $r/d^+(u)$ and a Poisson process $N_e$ with parameter $1/d^+(u)$. We refer to these processes as *clocks*, and when an event occurs in one of them, we say that the relevant clock *triggers*. We refer to $M_e$ as a *mutant clock with source $u$ and target $v$* and $N_e$ as a *non-mutant clock with source $u$ and target $v$*.

We use $\mathcal{C}(G)$ to denote the set of all clocks so $\mathcal{C}(G) = \bigcup_{e\in E(G)}\{M_e, N_e\}$. We use $P(G)$ to denote the Cartesian product of all process in $\mathcal{C}(G)$. $P(G)$ is the stochastic process in which all clocks in $\mathcal{C}(G)$ evolve simultaneously and independently, starting at time 0.

With probability 1, the clocks trigger a countably infinite number of times and these can be indexed by an increasing sequence $\tau_1, \tau_2, \ldots$. Also, no clocks trigger simultaneously and the clocks trigger for an infinitely long period — that is, for every clock and every $t$, the clock triggers at some $\tau_i > t$. For convenience, we take $\tau_0 = 0$. We will use the random variables $\tau_0, \tau_1, \ldots$ (which depend on the process $P(G)$) in our arguments.

### 8.3.2 The Moran process

**Definition 8.7 (the Moran process).** The (continuous-time) Moran process $X$ has an underlying graph $G(X) = G$ and an initial state $X_0 = \{x_0\}$, where $x_0 \in V(G)$. At every time $t$, the state $X_t$ is a subset of $V(G(X))$, which we sometimes refer to as the "set of mutants" at time $t$. Recall that, for every positive integer $i$, a clock $C \in \mathcal{C}(G)$ triggers at $\tau_i$. For $t \in (\tau_{i-1}, \tau_i)$, we set $X_t = X_{\tau_{i-1}}$. Then we define $X_{\tau_i}$ as follows.

(i) If $C = M_{(u,v)}$ for some $(u, v) \in E(G)$ and $u \in X_{\tau_{i-1}}$ then $X_{\tau_i} = X_{\tau_{i-1}} \cup \{v\}$.

(ii) If $C = N_{(u,v)}$ for some $(u, v) \in E(G)$ and $u \notin X_{\tau_{i-1}}$ then $X_{\tau_i} = X_{\tau_{i-1}} \setminus \{v\}$.

(iii) Otherwise, $X_{\tau_i} = X_{\tau_{i-1}}$.

Considering the positive integers $i$ in order, this completes the definition of the Moran process $X_t$.

We define some terminology associated with the Moran process $X$.

- If the clock $M_{(u,v)}$ triggers at time $t$ and $u \in X_t$ we say that $u$ *spawns a mutant onto $v$ at time* $t$ and that $X$ *spawns a mutant* onto $v$ at time $\tau_i$.
- If the clock $N_{(u,v)}$ triggers at time $t$ and $u \notin X_t$ we say that $u$ *spawns a non-mutant onto $v$ at time $t$*. We say that $X$ *spawns a non-mutant* onto $v$ at time $\tau_i$.
- If $v \in X_{\tau_i}$ and $v \notin X_{\tau_{i-1}}$ we say that $v$ *becomes a mutant at time $\tau_i$*.
- If $v \in X_{\tau_{i-1}}$ and $v \notin X_{\tau_i}$ we say that $v$ *becomes a non-mutant at time* $\tau_i$ or that $v$ *dies at time* $\tau_i$.

Note that $v$ does not necessarily become a mutant at time $\tau_i$ when some $u$ spawns a mutant onto $v$ at time $\tau_i$ since $v$ may already be a mutant at that time.

For convenience, we include the filtration $\mathcal{F}_t(P(G))$ in the filtration $\mathcal{F}_t(X)$ of the Moran process so the sequence of trigger-times $\tau_0, \tau_1, \ldots$ up to time $t$ can be determined from $\mathcal{F}_t(X)$.

**Definition 8.8.** We say that the Moran process is *extinct by time* $t$ if, for all $t' \geq t$, $X_{t'} = \emptyset$. We say that it *fixates by time* $t$ if, for all $t' \geq t$, $X_{t'} = V(G(X))$. We say that it *absorbs by time* $t$ if it is extinct by time $t$ or it fixates by time $t$. The *fixation probability* is the probability that, for some $t$, it fixates by time $t$. The *extinction probability* is the probability that, for some $t$, it is extinct by time $t$.

**Remark 8.9.** In Definition 8.7, say that $\tau_i$ is a "relevant trigger time" if (i) or (ii) occurs rather than (iii). The discrete-time Moran process [64], as adapted by Lieberman, Hauert and Nowak [59] is the Markov chain $X_{\tau_0}, X_{\tau_{i_1}}, X_{\tau_{i_2}}, \ldots$, where $\tau_{i_1}, \tau_{i_2}, \ldots$

is the increasing sequence of relevant trigger times. Note that the fixation probability of the discrete-time Moran process is the same as the fixation probability of the continuous-time process $X_t$, so we will study the process $X_t$ in this chapter.

**Remark 8.10.** The Moran process $X_t$ is extinct by time $t$ if $X_t = \emptyset$ and fixates by time $t$ if $X_t = V(G(X))$. If $G$ is strongly connected then the fixation probability and the extinction probability sum to 1.

**Definition 8.11.** For the Moran Process $X$, any vertex $u \in V(G(X))$, and any $t \geq 0$, we define $i_{\mathsf{m}}(X, u, t)$ to be the measure of the set $\{t' \leq t \mid u \in X_{t'}\}$. Similarly, we define $i_{\mathsf{n}}(X, u, t)$ to be the measure of the set $\{t' \leq t \mid u \notin X_{t'}\}$.

The subscript "m" stands for "mutant" since $i_{\mathsf{m}}(X, u, t)$ is the amount of time that $u$ is a mutant in $X$, up until time $t$. Similarly, the subscript "n" stands for non-mutant. The random variables $i_{\mathsf{m}}(X, u, t)$ and $i_{\mathsf{n}}(X, u, t)$ are determined by $\mathcal{F}_t(X)$. Also, $i_{\mathsf{m}}(X, u, t) + i_{\mathsf{n}}(X, u, t) = t$.

### 8.3.3 The star-clock process

Consider the Moran process process $X$. We wish to be able to discuss events such as the event that a vertex $u$ does not spawn a mutant until it has been a mutant for time $t$. In order to express such events in a clean way, making all conditioning explicit, we define additional stochastic processes.

For each edge $e = (u, v)$ of $G$ we define four further Poisson processes — Poisson processes $M_e^*$ and $\overline{M}_e^*$ each with parameter $r/d^+(u)$ and Poisson processes $N_e^*$ and $\overline{N}_e^*$ each with parameter $1/d^+(u)$. We refer to these processes as *star-clocks*. We identify sources and targets of star-clocks in the same way that we did for clocks. For example, the star-clock $M_{(u,v)}^*$ has source $u$ and target $v$.

We use $\mathcal{C}^*_{\text{mut}}(G)$ to denote the set $\mathcal{C}^*_{\text{mut}}(G) = \bigcup_{e \in E(G)} \{M_e^*, \overline{N}_e^*\}$. We use $\mathcal{C}^*_{\text{nmut}}(G)$ to denote the set $\mathcal{C}^*_{\text{nmut}}(G) = \bigcup_{e \in E(G)} \{N_e^*, \overline{M}_e^*\}$.

The star-clock process $P^*(G)$ is the stochastic process where all star-clocks in $\mathcal{C}^*_{\text{mut}}(G) \cup \mathcal{C}^*_{\text{nmut}}(G)$ evolve simultaneously and independently, starting at time 0.

### 8.3.4 A coupled process

Given the Moran process $X$, let $G = G(X)$. We will now define a stochastic process $\Psi(X)$ which is a coupling of $X$ (which includes the clock process $P(G)$) with the newly-defined star-clock process $P^*(G)$. Intuitively, the idea of the coupling is that each clock $M_{(u,v)}$ in $P(G)$ will evolve following $M^*_{(u,v)}$ when $u$ is a mutant and following $\overline{M}^*_{(u,v)}$ when $u$ is a non-mutant. Similarly, $N_{(u,v)}$ will evolve following $N^*_{(u,v)}$ when $u$ is a non-mutant and $\overline{N}^*_{(u,v)}$ when $u$ is a mutant. In the coupling, we pause the star-clocks in $\mathcal{C}^*_{\text{mut}}(G) \cup \mathcal{C}^*_{\text{nmut}}(G)$ while they are not being used to drive clocks in $\mathcal{C}(G)$, so that, e.g., the "local time" of a clock $M^*_{(u,v)}$ at global time $t$ is $i_{\mathsf{m}}(X, u, t)$.

We will be able to deduce both $\mathcal{F}_t(X)$ and $\mathcal{F}_t(P^*(G))$ from the filtration $\mathcal{F}_T(\Psi(X))$ of the coupled process at an appropriate stopping time $T$ — the details are given below. The fact that the coupling is valid (which we will show below) will ensure that both of the marginal processes, $X$ and $P^*(G)$, evolve according to their correct distributions.

To construct the coupling we start with a copy of the star-clock process $P^*(G)$ and with the initial state $X_0$ of the Moran process $X$. We define $\tau_0 = 0$ (so we have implicitly defined $\mathcal{F}_{\tau_0}(X)$).

Suppose that, for some non-negative integer $j$, we have defined $\mathcal{F}_{\tau_j}(X)$. Given this and the evolution of the star-clock process $P^*(G)$, we will show how to define $\tau_{j+1}$ and $\mathcal{F}_{\tau_{j+1}}(P(G))$ which determine $\mathcal{F}_{\tau_{j+1}}(X)$. To do this, let $t_j$ be the minimum $t > 0$ such that one of the following occurs.

- For some $u \in X_{\tau_j}$, a star-clock in $\mathcal{C}^*_{\text{mut}}(G)$ with source $u$ triggers at time $i_{\mathsf{m}}(X, u, \tau_j) + t$, or
- for some $u \notin X_{\tau_j}$, a star-clock in $\mathcal{C}^*_{\text{nmut}}(G)$ with source $u$ triggers at time $i_{\mathsf{n}}(X, u, \tau_j) + t$.

We define $\tau_{j+1} = \tau_j + t_j$. No clocks in $\mathcal{C}(G)$ trigger in the interval $(\tau_j, \tau_{j+1})$. We now determine which clock from $\mathcal{C}(G)$ triggers at time $\tau_{j+1}$ by reconsidering each case.

- If $u \in X_{\tau_j}$ and $M^*_{(u,v)}$ triggers at time $i_{\mathsf{m}}(X, u, \tau_j) + t_j$ then $M_{(u,v)}$ triggers at time $\tau_{j+1}$.
- If $u \in X_{\tau_j}$ and $\overline{N}^*_{(u,v)}$ triggers at time $i_{\mathsf{m}}(X, u, \tau_j) + t_j$ then $N_{(u,v)}$ triggers at time $\tau_{j+1}$.
- If $u \notin X_{\tau_j}$, and $N^*_{(u,v)}$ triggers at time $i_{\mathsf{n}}(X, u, \tau_j) + t_j$ then $N_{(u,v)}$ triggers at time $\tau_{j+1}$.
- If $u \notin X_{\tau_j}$ and $\overline{M}^*_{(u,v)}$ triggers at time $i_{\mathsf{n}}(X, u, \tau_j) + t_j$ then $M_{(u,v)}$ triggers at time $\tau_{j+1}$.

This fully defines $\mathcal{F}_{\tau_{j+1}}(P(G))$ and hence $\mathcal{F}_{\tau_{j+1}}(X)$. So we have fully defined the coupling and therefore the process $\Psi(X)$.

Before showing that the coupling is valid, it will be helpful to state exactly what information is contained in $\mathcal{F}_t(\Psi(X))$. Certainly this includes $\mathcal{F}_t(X)$ which itself includes $\mathcal{F}_t(P(G))$. Also, $\mathcal{F}_t(P(G))$ defines a non-negative integer $j$ so that $t \in [\tau_j, \tau_{j+1})$. We will use $j$ to state the information that $\mathcal{F}_t(\Psi(X))$ contains for the evolution of $P^*(G)$.

- For each star-clock $C \in \mathcal{C}^*_{\text{mut}}(G)$ with source $u \in X_{\tau_j}$, $\mathcal{F}_t(\Psi(X))$ includes a list of the times in $[0, i_{\mathsf{m}}(X, u, \tau_j) + t - \tau_j]$ when $C$ triggers.
- For each star-clock $C \in \mathcal{C}^*_{\text{mut}}(G)$ with source $u \notin X_{\tau_j}$, $\mathcal{F}_t(\Psi(X))$ includes a list of the times in $[0, i_{\mathsf{m}}(X, u, \tau_j)]$ when $C$ triggers.

- For each star-clock $C \in \mathcal{C}^*_{\text{nmut}}(G)$ with source $u \notin X_{\tau_j}$, $\mathcal{F}_t(\Psi(X))$ includes a list of the times in $[0, i_{\mathsf{n}}(X, u, \tau_j) + t - \tau_j]$ when $C$ triggers.
- For each star-clock $C \in \mathcal{C}^*_{\text{nmut}}(G)$ with source $u \in X_{\tau_j}$, $\mathcal{F}_t(\Psi(X))$ includes a list of the times in $[0, i_{\mathsf{n}}(X, u, \tau_j)]$ when $C$ triggers.

To show that the coupling is valid we must show that both of the marginal processes, $X$ and $P^*(G)$, evolve according to their correct distributions. The fact that $P^*(G)$ does so is by construction. To show that $X$ does so, it suffices to prove that for all $j \in \mathbb{Z}_{\geq 0}$ and all possible values $f_j$ of $\mathcal{F}_{\tau_j}(X)$, the distribution of $\mathcal{F}_{\tau_{j+1}}(X)$ conditioned on $\mathcal{F}_{\tau_j}(X) = f_j$ is correct. Note that the only information contained in $\mathcal{F}_{\tau_{j+1}}(X)$ but not $\mathcal{F}_{\tau_j}(X)$ is the value of $\tau_{j+1}$ and the identity of the clock in $\mathcal{C}(G)$ that triggers at time $\tau_{j+1}$.

Let $f'_j$ be an arbitrary possible value of $\mathcal{F}_{\tau_j}(\Psi(X))$ that is consistent with the event $\mathcal{F}_{\tau_j}(X) = f_j$, in the sense that the intersection of the events $\mathcal{F}_{\tau_j}(\Psi(X)) = f'_j$ and $\mathcal{F}_{\tau_j}(X) = f_j$ is non-empty. Recall from the definition of $\Psi(X)$ that, conditioned on $\mathcal{F}_{\tau_j}(\Psi(X)) = f_j$, $\mathcal{F}_{\tau_{j+1}}(X)$ depends only on particular star-clocks in particular intervals, as follows.

- For each $u \in X_{\tau_j}$, it depends on the evolution of each star-clock in $\mathcal{C}^*_{\text{mut}}(G)$ with source $u$ only during the interval $(i_{\mathsf{m}}(X, u, \tau_j), \infty)$. It does not depend on the evolution of star-clocks in $\mathcal{C}^*_{\text{nmut}}(G)$ with source $u$.
- For each $u \notin X_{\tau_j}$, it depends on the evolution of each star-clock in $\mathcal{C}^*_{\text{nmut}}(G)$ with source $u$ only during the interval $(i_{\mathsf{n}}(X, u, \tau_j), \infty)$. It does not depend on the evolution of star-clocks in $\mathcal{C}^*_{\text{mut}}(G)$ with source $u$.

For each star-clock, these intervals are disjoint from the intervals exposed in $f'_j$, and the start of each interval is determined by $f'_j$. Moreover, in the interval $(\tau_j, \tau_{j+1}]$, each clock in $\mathcal{C}(G)$ is triggered by a unique clock in $\mathcal{C}^*_{\text{mut}}(G) \cup \mathcal{C}^*_{\text{nmut}}(G)$ with the same rate. Thus all clocks in $\mathcal{C}(G)$ trigger with the correct rates in this period and they are independent of each other (since all of the star-clocks in $P^*(G)$ evolve independently). We conclude that $\mathcal{F}_{\tau_{j+1}}(P(G))$, and hence $\mathcal{F}_{\tau_{j+1}}(X)$, has the appropriate distribution. The coupling is therefore valid.

By construction, we have the following observation.

**Observation 8.12.** Let $X$ be the Moran process and consider $\Psi(X)$. Let $(u, v)$ be an edge of $G(X)$. Given $t > 0$, let $j$ be the maximum integer such that $\tau_j < t$. Then the following are true.

- $M_{(u,v)}$ triggers at time $t$ if and only if either $u \in X_{\tau_j}$ and $M^*_{(u,v)}$ triggers at time $i_{\mathsf{m}}(X, u, t)$ or $u \notin X_{\tau_j}$ and $\overline{M}^*_{(u,v)}$ triggers at time $i_{\mathsf{n}}(X, u, t)$.
- $N_{(u,v)}$ triggers at time $t$ if and only if either $u \notin X_{\tau_j}$ and $N^*_{(u,v)}$ triggers at time $i_{\mathsf{n}}(X, u, t)$ or $u \in X_{\tau_j}$ and $\overline{N}^*_{(u,v)}$ triggers at time $i_{\mathsf{m}}(X, u, t)$.

## 8.4 An upper bound on the fixation probability of superstars

Recall the definition of a $(k, \ell, m)$-superstar (Definition 1.21). We use $n = \ell(k+m)+1$ to denote the number of vertices of a $(k, \ell, m)$-superstar.

Given any $i \in [\ell]$, we say that $v_{i,1}v_{i,2} \dots v_{i,k}$ is the *path associated with* the reservoir $R_i$. We will often consider the case that the initial mutant $x_0$ is in a reservoir. When it is possible, we simplify the notation by dropping the index $i$ of the reservoir. Thus, we write $R$ for the reservoir containing $x_0$ and we write $v_1 \dots v_k$ for the path associated with $R$. So if $R = R_i$ then for each $j \in [k]$, we write $v_j$ as a synonym for $v_{i,j}$. The main result of this section is the following upper bound on the fixation probability of the superstar.

**Theorem 8.13.** *Let $r > 1$. Then there exists a constant $c_r > 0$ (depending on $r$) such that the following holds for all positive integers $k$, $\ell$ and $m$. Choose $x_0$ uniformly at random from $V(\mathcal{S}_{k,\ell,m})$. Let $X$ be the Moran process (with fitness $r$) with $G(X) = \mathcal{S}_{k,\ell,m}$ and $X_0 = \{x_0\}$. Then the probability that $X$ goes extinct is at least $1/(c_r(n \log n)^{1/3})$.*

### 8.4.1 Proof Sketch

In this Section, we give an informal sketch of the proof of Theorem 8.13. The presentation of the proof itself does not depend upon the sketch so the reader may prefer to skip directly to the proof. In all of our proof sketches, we use the word "likely" to mean "sufficiently likely". We leave the details of "how likely" to the actual proofs.

If $m$ is small relative to $k$ (in particular, if $m < k(n \log n)^{1/3}$) then the initial mutant $x_0$ is likely to be placed in a path, rather than in a reservoir. If this happens, then it is likely to go extinct. This easy case is dealt with in Lemma 8.14 and corresponds to Case 2 in the proof of Theorem 8.13. (Case 1 is the trivial case where $n < n_0$.)

Another easy case arises if $\ell$ is sufficiently small relative to $n$ (in particular, if $\ell = O((n \log n)^{1/3})$). This case is dealt with in Lemma 8.15 and corresponds to Case 3 in the proof of Theorem 8.13. In this case, even when $x_0$ is placed in a reservoir $R$, it is still likely that $x_0$ dies before $v_2$ ever becomes a mutant. This is because it takes roughly $\Theta(m)$ time for the mutation to spread from $v_1$ to $v_2$ since a mutant at $v_1$ has only probability $\Theta(1/m)$ of spawning a mutant before it dies. On the other hand, since $\ell$ is small, $x_0$ is sufficiently likely to die in $\Theta(m)$ time. For details, see the proof of Lemma 8.15.

The remaining case, Case 4 in the proof of Theorem 8.13, is deemed the "difficult regime" and is dealt with in Section 8.4.3. In this case, it is easy to show that $\ell = \Omega(\kappa \log n)$ and $m = \Omega(\kappa)$ where $\kappa = \max\{3k, 70r^4 \log n\}$.

It is likely that the initial mutant $x_0$ is placed in a reservoir $R$, and the key lemma, showing that it is sufficiently likely to go extinct, is Lemma 8.17.

At a very high level, the argument proceeds as follows. Suppose that $v^*$ does not spawn a mutant before $x_0$ dies. Then it is very easy to see that, after $x_0$ dies, the path of reservoir $R$ is likely to go extinct quickly.

Thus, the crux of the argument is to show that $x_0$ is likely to die before $v^*$ spawns a mutant. Each time $v^*$ becomes a mutant it has an $O(1/\ell)$ chance of spawning a mutant before dying, so roughly our goal is to show that $x_0$ is sufficiently likely to die before $v^*$ becomes a mutant $\Omega(\ell)$ times.

Very roughly, our high-level approach is to partition time into intervals of length $\kappa = O(m)$. In each block of $O(m/\kappa)$ such intervals, $v_2$ is likely to become a mutant $O(1)$ times. Each time this happens, it is likely that $R$'s path will again fill with non-mutants within $O(\kappa)$ time, so it is likely that $v_k$ is a mutant for at most $O(\kappa)$ time during the block and it is likely that $v^*$ becomes a mutant at most $O(\kappa)$ times during the block. Combining $O(\ell/\kappa)$ blocks, it is likely that $v^*$ becomes a mutant at most $O(\ell)$ times by time $\ell m/\kappa$. Since $N_{(v^*,x_0)}$ has rate $1/(\ell m)$, it is also likely that $x_0$ dies by time $\ell m/\kappa$.

In more detail, the proof of Lemma 8.17 shows that $x_0$ dies before $v^*$ spawns a mutant as long as certain events called $\mathcal{P}_1$–$\mathcal{P}_5$ occur. These events are defined in the statement of Lemma 8.23. They formalise the high-level approach that we have just described. It is important that most of these events are defined in terms of clock-triggers so that we can get good upper bounds on the probability that they fail and thus prove (in Lemma 8.23) that they are likely to occur simultaneously.

The proof of Lemma 8.17 tracks a quantity $\sigma(t)$ which is the number of times that $v_k$ (the end of the path of the reservoir containing $x_0$) spawns a mutant onto the centre vertex $v^*$ by time $t$. The proof uses $\mathcal{P}_1$–$\mathcal{P}_5$ to show that $\sigma(t)$ stays $O(\ell)$ up to a fixed time $t_{x_0} = O(\ell m/\kappa)$. As we noted, the analysis divides the period up to time $t_{x_0}$ into intervals of size $\kappa$. Event $\mathcal{P}_5$ ensures that during most such intervals, non-mutant clocks with target $v_1$ and mutant clocks with targets $v_1$ and $v_2$ behave appropriately so that, if $x_0$ is the only mutant in $R$ during the interval, then $v_2$ does not become a mutant during the interval. The fact that $x_0$ is indeed the only mutant in $R$ follows from event $\mathcal{P}_1$ which ensures that $v^*$ does not spawn a mutant while $\sigma(t)$ is small. Then since $v_2$ does not become a mutant during the interval, event $\mathcal{P}_3$ ensures that the clocks along the path trigger in such a way that (unless $v_1$ or $v^*$ spawn a mutant) the only mutants remaining at the end of the interval are in $\{x_0, v_1\}$. This ensures that $\sigma(t)$ stays small through another interval. Event $\mathcal{P}_5$ only ensures the above during "most such intervals" but event $\mathcal{P}_4$ ensures that the mutant clock with source $v_k$ does not trigger too often, so the remaining intervals are not too problematic. Thus, events $\mathcal{P}_1$, $\mathcal{P}_3$, $\mathcal{P}_4$ and $\mathcal{P}_5$, taken together, ensure that $\sigma(t_{x_0})$ is $O(\ell)$.

Given that $\sigma(t_{x_0})$ is $O(\ell)$, it is easy to show that the initial mutant goes extinct during the next two intervals (beyond time $t_{x_0}$). Event $\mathcal{P}_1$ ensures that $v^*$ doesn't spawn any mutants. Event $\mathcal{P}_2$ ensures that the initial mutant $x_0$ has already died by time $t_{x_0}$. Finally, event $\mathcal{P}_3$ ensures that any remaining mutants die in the path during the next two intervals.

The difficult part of the proof is defining events $\mathcal{P}_1$–$\mathcal{P}_5$ in such a way that we can show (in Lemma 8.23) that they are likely to occur simultaneously. It turns out (Lemma 8.27, Corollary 8.29 and Lemma 8.30) that events $\mathcal{P}_3$–$\mathcal{P}_5$ are so unlikely to fail that we bound this probability with a simple union bound, avoiding any complicating conditioning. (Of course, for this it was necessary to express these events in terms of clocks rather than in terms of the underlying Moran process.) In order to simplify the presentation, we deal with $\mathcal{P}_1$ and $\mathcal{P}_2$ together, in Lemma 8.26. Roughly, they correspond to the event that, as long as $\sigma(t) = O(\ell)$ then $v^*$ does not spawn a mutant at time $t$ and, for $t = t_{x_0}$, $x_0$ dies by time $t$. This event is implied by the conjunction of three further events.

- $\mathcal{E}_1$ is the event that no star-clock $M^*_{(v^*,v)}$ (for any $v$) triggers in $[0, 1/r]$.
- $\mathcal{E}_2$ is the event that the star-clock $N^*_{(v^*,x_0)}$ triggers in $[0, t_{x_0} - 1]$.
- $\mathcal{E}_3$ corresponds informally to the event that $v^*$ is a mutant for a period of time shorter than $1/r$ during the first $O(\ell)$ times that it becomes a mutant (though the formal definition is expressed in terms of clocks, and is a little more complicated). Note the intention, though, which is to ensure that $v^*$ is a mutant for a period of time shorter than $1/r$, which makes $\mathcal{E}_1$ relevant.

Lemma 8.25 shows that $\mathcal{E}_3$ is very likely to hold. In the proof of Lemma 8.26, it is observed that $\mathcal{E}_1$ and $\mathcal{E}_2$ are independent (by the definition of the star-clocks) and that $\mathbb{P}(\mathcal{E}_1) = 1/e$. The proof demonstrates that $\mathcal{E}_2$ is sufficiently likely, giving the desired bound.

### 8.4.2 The easy regimes

**Lemma 8.14.** *Choose $x_0$ uniformly at random from $V(\mathcal{S}_{k,\ell,m})$. Let $X$ be the Moran process with $G(X) = \mathcal{S}_{k,\ell,m}$ and $X_0 = \{x_0\}$. The extinction probability of $X$ is at least $k/(2r(m+k))$.*

*Proof.* We have

$$\mathbb{P}(x_0 \notin R_1 \cup \cdots \cup R_\ell) = 1 - \frac{\ell m}{\ell(m+k)+1} \geq 1 - \frac{m}{m+k} = \frac{k}{m+k}.$$

Moreover, if $x_0 \notin R_1 \cup \cdots \cup R_\ell$, then $x_0$ has an in-neighbour of out-degree 1 so, with probability at least $1/(1+r) \geq 1/(2r)$, $x_0$ dies before spawning a mutant. The result therefore follows. □

**Lemma 8.15.** *Suppose $m \geq 12r$ and $x_0 \in R_1 \cup \cdots \cup R_\ell$. Let $X$ be the Moran process with $G(X) = \mathcal{S}_{k,\ell,m}$ and $X_0 = \{x_0\}$. The extinction probability of $X$ is at least $1/(26r^2\ell)$.*

*Proof.* Let $R$ be the reservoir containing $x_0$, and let $v_1 \dots v_k$ be the path associated with $R$. Let $\xi = \lfloor m/(2r) \rfloor$, $t^* = m/(4r^2)$ and $J = [0, t^*]$. For all $t \geq 0$, let $\mathcal{E}^1$, $\mathcal{E}^2$ and $\mathcal{E}^3_t$ be events defined as follows.

$\mathcal{E}^1$: $N_{(v^*,x_0)}$ triggers in $J$.

$\mathcal{E}^2$: $M_{(x_0,v_1)}$ triggers at most $\xi$ times in $J$.

$\mathcal{E}^3_t$**:** $\min\{t' > t \mid \text{for some } v \neq x_0,\ N_{(v,v_1)} \text{ triggers at } t'\}$
$< \min\{t' > t \mid M_{(v_1,v_2)} \text{ triggers at } t'\}$.

Finally, let $T^i_{v_1}$ be the $i$'th time at which the clock $M_{(x_0,v_1)}$ triggers and define $\mathcal{E}^3 = \bigcap_{i=1}^{\xi} \mathcal{E}^3_{T^i_{v_1}}$.

Suppose that events $\mathcal{E}^1$, $\mathcal{E}^2$ and $\mathcal{E}^3$ occur. We will show that $X$ goes extinct. Let $\xi'$ be the number of times that $v_1$ becomes a mutant in $J$. By $\mathcal{E}^2$, $\xi' \leq \xi$. By $\mathcal{E}^3$, for each of the first $\xi'$ times that $v_1$ becomes a mutant, it dies before spawning a mutant. Thus, for all $t \in J$, $X_t \subseteq \{x_0, v_1\}$. Also, by $\mathcal{E}^1$, $x_0$ dies in $J$. As soon as $x_0$ dies, and $v_1$ dies for the $(\xi')$'th time, $X$ is extinct.

We bound $\mathbb{P}(\mathcal{E}^1 \cap \mathcal{E}^2 \cap \mathcal{E}^3)$ below. Since $N_{(v^*,x_0)}$ has rate $1/(\ell m)$, we have

$$\mathbb{P}(\mathcal{E}^1) = 1 - e^{-t^*/(\ell m)} = 1 - e^{-m/(4r^2\ell m)} \geq 1 - \left(1 - \frac{1}{8r^2\ell}\right) = \frac{1}{8r^2\ell}. \tag{8.2}$$

Here the inequality follows by (8.1). Moreover, since $M_{(x_0,v_1)}$ has rate $r$, by Corollary 8.4 we have

$$\mathbb{P}(\mathcal{E}^2) \geq 1 - e^{-m/(12r)} \geq 1 - e^{-1}. \tag{8.3}$$

For any $t \in J$, let $f$ be a possible value of $\mathcal{F}_t(X)$. Let $\Phi$ be the random variable containing the list of times in $J$ at which $N_{(v^*,x_0)}$ and $M_{(x_0,v_1)}$ trigger. Let $\varphi$ be a possible value of $\Phi$ which is consistent with the events $\mathcal{F}_t(X) = f$ and $\mathcal{E}^1 \cap \mathcal{E}^2$. Note that $\varphi$ determines $\mathcal{E}^1 \cap \mathcal{E}^2$. By memorylessness and independence of clocks in $\mathcal{C}(\mathcal{S}_{k,\ell,m})$, we have

$$\mathbb{P}\left(\mathcal{E}^3_t \mid \mathcal{F}_t(X) = f, \Phi = \varphi\right) = \frac{m-1}{m-1+r} = 1 - \frac{r}{m-1+r} \geq 1 - \frac{r}{m}.$$

Thus for all $i \in [\xi]$, $\mathbb{P}\left(\mathcal{E}^3_{T^i_{v_1}} \mid \mathcal{E}^1 \cap \mathcal{E}^2\right) \geq 1 - r/m$. It follows by a union bound that

$$\mathbb{P}\left(\mathcal{E}^3 \mid \mathcal{E}^1 \cap \mathcal{E}^2\right) \geq 1 - \xi\left(\frac{r}{m}\right) \geq \frac{1}{2}. \tag{8.4}$$

Since $\mathcal{E}^1$ and $\mathcal{E}^2$ depend entirely on distinct clocks in $\mathcal{C}(\mathcal{S}_{k,\ell,m})$ in fixed intervals, the two events are independent. Thus by (8.2)–(8.4), we have

$$\mathbb{P}\left(\mathcal{E}^1 \cap \mathcal{E}^2 \cap \mathcal{E}^3\right) = \mathbb{P}\left(\mathcal{E}^1\right) \mathbb{P}\left(\mathcal{E}^2\right) \mathbb{P}\left(\mathcal{E}^3 \mid \mathcal{E}^1 \cap \mathcal{E}^2\right) \geq \left(\frac{1}{8r^2\ell}\right)\left(1 - \frac{1}{e}\right)\frac{1}{2} \geq \frac{1}{26r^2\ell},$$

and the result follows. ☐

### 8.4.3 The difficult regime

**Definition 8.16.** Let $K = 70$ and $\kappa = \max\{3k, Kr^4 \log n\}$.

**Lemma 8.17.** *Consider any $r > 1$. There is an $n_0$, depending on $r$, such that the following holds. Suppose that $\ell \geq Kr^4\kappa \log n$, $m \geq 6r^2\kappa$ and $n \geq n_0$. Fix $x_0 \in R_1 \cup \cdots \cup R_\ell$. Let $X$ be the Moran process with $G(X) = \mathcal{S}_{k,\ell,m}$ and $X_0 = \{x_0\}$. Then the extinction probability of $X$ is at least $1/(7Kr^4\kappa)$.*

The following corollary, which applies to the regime in which $\kappa = 3k$, is immediate.

**Corollary 8.18.** *Consider any $r > 1$. There is an $n_0$, depending on $r$, such that the following holds. Suppose that $k \geq (K/3)r^4 \log n$, $\ell \geq 3Kr^4k \log n$, $m \geq 18r^2k$ and $n \geq n_0$. Fix $x_0 \in R_1 \cup \cdots \cup R_\ell$. Let $X$ be the Moran process with $G(X) = \mathcal{S}_{k,\ell,m}$ and $X_0 = \{x_0\}$. Then the extinction probability of $X$ is at least $1/(21Kr^4k)$.*

The crux of our proof of Lemma 8.17 is Lemma 8.23. In order to state this lemma, we require the following additional definitions.

**Definition 8.19.** Let $t_{x_0} = \ell m/(Kr^4\kappa)$, and $t_{\mathsf{max}} = 2\ell m$. For all $i \in \mathbb{Z}_{\geq 0}$, let $I_i = (i\kappa, (i+1)\kappa]$. For all $t \in [0, t_{\mathsf{max}}]$, let

$$\sigma(t) = \left|\{t' \leq t \mid v_k \text{ spawns a mutant onto } v^* \text{ at time } t'\}\right|.$$

The reason that we give $t_{x_0}$ its name is that we will be most concerned with the case in which $x_0$ dies in the interval $[0, t_{x_0}]$.

**Definition 8.20.** Let $I \subseteq [0, \infty)$ be an interval, let $R \in \{R_1, \ldots, R_\ell\}$, suppose $x_0 \in R$, and let $v_1 \ldots v_k$ be the path associated with $R$. We say that $v_1$ *clears before spawning a mutant within* $I$ if at least one of the following statements holds:

(i) $M_{(v_1,v_2)}$ does not trigger in $I$, or

(ii) for some $v \neq x_0$, $N_{(v,v_1)}$ triggers in $I$ before $M_{(v_1,v_2)}$ first triggers in $I$.

**Definition 8.21.** Let $i \in \mathbb{Z}_{\geq 0}$, let $R \in \{R_1, \ldots, R_\ell\}$, suppose $x_0 \in R$, and let $v_1 \ldots v_k$ be the path associated with $R$. We say that $v_2$ *is protected in* $I_i$ if both of the following properties hold.

(i) $v_1$ clears before spawning a mutant within $I_i$.

(ii) For all $t \in I_i$ such that $M_{(x_0,v_1)}$ triggers at time $t$, $v_1$ clears before spawning a mutant within $(t, (i+1)\kappa]$.

In particular, suppose that $v_2$ is protected in $I_i$ and that $x_0$ is the only mutant in $R$ for the duration of $I_i$. Then as we will see in the proof of Lemma 8.17, $v_2$ does not become a mutant in $I_i$.

**Definition 8.22.** Let $i \in \mathbb{Z}_{\geq 0}$, let $R \in \{R_1, \dots, R_\ell\}$, suppose $x_0 \in R$, and let $v_1 \dots v_k$ be the path associated with $R$. We say that $v_1 \dots v_k$ *clears within* $I_i$ if there exist $v_0 \in R \setminus \{x_0\}$ and times $i\kappa < t_1 < \cdots < t_{k+1} \leq (i+1)\kappa$ satisfying both of the following properties.

(i) For all $j \in [k]$, $N_{(v_{j-1}, v_j)}$ triggers at time $t_j$, and $N_{(v_k, v^*)}$ triggers at time $t_{k+1}$.

(ii) $M_{(x_0, v_1)}$ does not trigger in the interval $[t_1, t_2]$.

To see the purpose of this definition consider the following. Suppose that $X_{i\kappa} \subseteq \{x_0, v_1, \dots, v_k, v^*\}$, that neither $v_1$ nor $v^*$ spawns a mutant within $I_i$, and that $v_1 \dots v_k$ clears within $I_i$. Then, as we will see in the proof of Lemma 8.17, we will have $X_{(i+1)\kappa} \subseteq \{x_0, v_1\}$.

Our main task will be to prove the following lemma.

**Lemma 8.23.** *Consider any $r > 1$. There is an $n_0$, depending on $r$, such that the following holds. Suppose $\ell \geq Kr^4\kappa \log n$, $m \geq 6r^2\kappa$ and $n \geq n_0$. Let $R \in \{R_1, \dots, R_\ell\}$, and let $v_1 \dots v_k$ be the path associated with $R$. Fix $x_0 \in R$. Let $X$ be the Moran process with $G(X) = \mathcal{S}_{k,\ell,m}$ and $X_0 = \{x_0\}$. Then, with probability at least $1/(7Kr^4\kappa)$, the following events occur simultaneously.*

$\mathcal{P}_1$: *$\forall t \leq t_{\mathsf{max}}$, $\sigma(t) \geq \lfloor \ell/(2r) \rfloor + 1$ or $v^*$ does not spawn a mutant at time $t$.*

$\mathcal{P}_2$: *$\sigma(t_{x_0}) \geq \lfloor \ell/(2r) \rfloor + 1$ or $x_0 \notin X_{t_{x_0}}$.*

$\mathcal{P}_3$: *For all integers $i$ with $0 \leq i \leq t_{x_0}$, $v_1 \dots v_k$ clears within $I_i$.*

$\mathcal{P}_4$: *For all integers $i$ with $0 \leq i \leq t_{x_0}$, the clock $M_{(v_k, v^*)}$ triggers at most $\lfloor 2r\kappa \rfloor$ times within $I_i$.*

$\mathcal{P}_5$: *For all but at most $8r^2 t_{x_0}/m$ integers $i$ with $0 \leq i \leq t_{x_0}/\kappa$, $v_2$ is protected in $I_i$.*

Note that the definition of $\mathcal{P}_5$ considers $i$ up to $t_{x_0}/\kappa$, because it corresponds to at most $t_{x_0}/\kappa$ intervals of length $\kappa$. The definitions of $\mathcal{P}_3$ and $\mathcal{P}_4$ consider larger values of $i$. In fact, it is only necessary to take $i$ up to $t_{x_0}/\kappa + 2$ in $\mathcal{P}_3$ and $\mathcal{P}_4$ but we state the lemma as we did to avoid clutter. As a first step towards proving Lemma 8.23, we prove Lemmas 8.25 and 8.26 which give a lower bound on the probability that $\mathcal{P}_1$ and $\mathcal{P}_2$ hold.

**Definition 8.24.** Let $R \in \{R_1, \dots, R_\ell\}$, let $v_1 \dots v_k$ be the path associated with $R$, and suppose $x_0 \in R$. Let $X$ be the Moran process with $G(X) = \mathcal{S}_{k,\ell,m}$ and $X_0 = \{x_0\}$. We give two mutual recurrences to define stopping times $T_{\mathsf{n}}^h$ for all $h \in \mathbb{Z}_{\geq 0}$ and $T_{\mathsf{m}}^h$ for all $h \in \mathbb{Z}_{\geq 1}$. The subscript "$\mathsf{n}$" stands for "non-mutant" and the subscript "$\mathsf{m}$" stands for “mutant”.

$$T_{\mathsf{n}}^h = \begin{cases} 0, & \text{if } h = 0, \\ \min\left\{ t > T_{\mathsf{m}}^h \;\middle|\; \begin{array}{l} t = t_{\mathsf{max}} \text{ or some clock } N_{(v,v^*)} \\ \text{with } v \neq v_k \text{ triggers at } t \end{array} \right\}, & \text{otherwise.} \end{cases}$$

$$T_{\mathsf{m}}^h = \min\{t > T_{\mathsf{n}}^{h-1} \mid t = t_{\mathsf{max}} \text{ or } v_k \text{ spawns a mutant onto } v^* \text{ at } t\}.$$

Finally, for all $h \in \mathbb{Z}_{\geq 1}$, let $Y_h = T_{\mathsf{n}}^h - T_{\mathsf{m}}^h$.

**Lemma 8.25.** *Consider any $r > 1$. There is an $n_0$, depending on $r$, such that the following holds. Suppose $\ell \geq K r^4 \kappa \log n$ and $n \geq n_0$. Let $R \in \{R_1, \ldots, R_\ell\}$ and let $v_1 \ldots v_k$ be the path associated with $R$. Let $x_0 \in R$. Let $X$ be the Moran process with $G(X) = \mathcal{S}_{k,\ell,m}$ and $X_0 = \{x_0\}$. Then*

$$\mathbb{P}\left( \sum_{i=1}^{\lfloor \ell/(2r) \rfloor} Y_i < \frac{1}{r} \right) \geq 1 - \frac{1}{n^2}.$$

*Proof.* Let $n_0$ be an integer which is sufficiently large with respect to $r$. We claim that $Y_1, \ldots, Y_{\lfloor \ell/(2r) \rfloor}$ are stochastically dominated above by $\lfloor \ell/(2r) \rfloor$ independent exponential variables, each with parameter $\ell - 1$. To see this claim, fix $i \in \mathbb{Z}_{\geq 1}$, $t \geq 0$, and $y_1, \ldots, y_{i-1}, t_{\mathsf{m}}^i > 0$. Let $f_i$ be a possible value of $\mathcal{F}_{t_{\mathsf{m}}^i}(X)$. Suppose that the events $Y_1 = y_1, \ldots, Y_{i-1} = y_{i-1}$, $T_{\mathsf{m}}^i = t_{\mathsf{m}}^i$ and $\mathcal{F}_{t_{\mathsf{m}}^i}(X) = f_i$ are consistent, and note that in this case $\mathcal{F}_{t_{\mathsf{m}}^i}(X) = f_i$ determines the other events.

If $t \geq 0$ satisfies $t_{\mathsf{m}}^i + t \geq t_{\mathsf{max}}$, it follows that $t_{\mathsf{m}}^i + t \geq T_{\mathsf{n}}^i$ and hence if $\mathcal{F}_{t_{\mathsf{m}}^i}(X) = f_i$ then $Y_i = T_{\mathsf{n}}^i - t_{\mathsf{m}}^i \leq t$. Hence, for all such $t$,

$$\mathbb{P}\Big(Y_i \leq t \mid \mathcal{F}_{t_{\mathsf{m}}^i}(X) = f_i\Big) = 1 \geq 1 - e^{-(\ell-1)t}. \tag{8.5}$$

Suppose instead that $t_{\mathsf{m}}^i + t < t_{\mathsf{max}}$. If $\mathcal{F}_{t_{\mathsf{m}}^i}(X) = f_i$ then $Y_i \leq t$ if and only if some clock $N_{(v,v^*)}$ with $v \neq v_k$ triggers in the interval $(t_{\mathsf{m}}^i, t_{\mathsf{m}}^i + t]$. These clocks have total rate $\ell - 1$, and so by memorylessness we have

$$\mathbb{P}\Big(Y_i \leq t \mid \mathcal{F}_{t_{\mathsf{m}}^i}(X) = f_i\Big) = 1 - e^{-(\ell-1)t}. \tag{8.6}$$

Since (8.5) and (8.6) apply to every value of $f_i$ consistent with $Y_1 = y_1, \ldots, Y_{i-1} = y_{i-1}$ and $T_{\mathsf{m}}^i = t_{\mathsf{m}}^i$, it follows that

$$\mathbb{P}\Big(Y_i \leq t \mid Y_1 = y_1, \ldots, Y_{i-1} = y_{i-1}\Big) \geq 1 - e^{-(\ell-1)t}.$$

Thus $\sum_{i=1}^{\lfloor \ell/(2r) \rfloor} Y_i$ is stochastically dominated above by a sum $S$ of $\lfloor \ell/(2r) \rfloor$ i.i.d. exponential variables with parameter $\ell - 1$. Corollary 8.5 applies since

$$\frac{1}{r} \geq \frac{3\lfloor \ell/(2r) \rfloor}{2(\ell-1)},$$

so we have

$$\mathbb{P}\left(\sum_{i=1}^{\lfloor \ell/(2r) \rfloor} Y_i < \frac{1}{r}\right) \geq \mathbb{P}\left(S < \frac{1}{r}\right) \geq 1 - e^{-(\ell-1)/(16r)} \geq 1 - \frac{1}{n^2},$$

as required. □

We are now in a position to prove that $\mathcal{P}_1$ and $\mathcal{P}_2$ occur with reasonable probability.

**Lemma 8.26.** *Consider any $r > 1$. There is an $n_0$, depending on $r$, such that the following holds. Suppose $\ell \geq Kr^4\kappa \log n$, $m \geq 2$ and $n \geq n_0$. Let $R \in \{R_1, \dots, R_\ell\}$ and let $v_1 \dots v_k$ be the path associated with $R$. Let $x_0 \in R$. Let $X$ be the Moran process with $G(X) = \mathcal{S}_{k,\ell,m}$ and $X_0 = \{x_0\}$. Then $\mathbb{P}(\mathcal{P}_1 \cap \mathcal{P}_2) \geq 1/(6Kr^4\kappa)$.*

*Proof.* Let $n_0$ be an integer which is sufficiently large with respect to $r$. Consider the process $\Psi(X)$. Define the following three events.

$\mathcal{E}_1$: no star-clock $M^*_{(v^*,v)}$ (for any $v$) triggers in $[0, 1/r]$.

$\mathcal{E}_2$: the star-clock $N^*_{(v^*,x_0)}$ triggers in $[0, t_{x_0} - 1]$.

$\mathcal{E}_3$: $\sum_{i=1}^{\lfloor \ell/(2r) \rfloor} Y_i < 1/r$.

We will prove that $\mathbb{P}(\mathcal{E}_1 \cap \mathcal{E}_2 \cap \mathcal{E}_3) \geq 1/(6Kr^4\kappa)$, and that if $\mathcal{E}_1 \cap \mathcal{E}_2 \cap \mathcal{E}_3$ occurs then so does $\mathcal{P}_1$ and $\mathcal{P}_2$.

We first bound $\mathbb{P}(\mathcal{E}_1 \cap \mathcal{E}_2 \cap \mathcal{E}_3)$ below. The sum of the parameters of the star-clocks in $\{M^*_{(v^*,v)}\}$ is $r$, so $\mathbb{P}(\mathcal{E}_1) = e^{-1}$. We have $t_{x_0} = \ell m/(Kr^4\kappa) \geq m \log n \geq 2 \log n_0$ by hypothesis so, by choice of $n_0$, we may assume $t_{x_0} \geq 25$. The parameter of the star-clock $N^*_{(v^*,x_0)}$ is $1/(\ell m)$, so using (8.1) we have

$$\mathbb{P}(\mathcal{E}_2) = 1 - e^{-(t_{x_0}-1)/(\ell m)} \geq 1 - e^{-24t_{x_0}/(25\ell m)} \geq \frac{12t_{x_0}}{25\ell m} = \frac{12}{25Kr^4\kappa}.$$

Note that $\mathcal{E}_1$ and $\mathcal{E}_2$ are independent of each other by the definition of the star-clock process $P^*(\mathcal{S}_{k,\ell,m})$ and the fact that the intervals in the definitions of $\mathcal{E}_1$ and $\mathcal{E}_2$ are fixed: $t_{x_0} = \ell m/(Kr^4\kappa)$ does not depend on the evolution of $\Psi(X)$. So we have $\mathbb{P}(\mathcal{E}_1 \cap \mathcal{E}_2) \geq 12/(25eKr^4\kappa)$. Finally, by Lemma 8.25 together with the fact that $\kappa \leq n$, it follows that

$$\mathbb{P}(\mathcal{E}_1 \cap \mathcal{E}_2 \cap \mathcal{E}_3) \geq \mathbb{P}(\mathcal{E}_1 \cap \mathcal{E}_2) - \mathbb{P}(\overline{\mathcal{E}_3}) \geq \frac{12}{25eKr^4\kappa} - \frac{1}{n^2} \geq \frac{1}{6Kr^4\kappa}. \tag{8.7}$$

We next show that $\mathcal{E}_1$ and $\mathcal{E}_3$ together imply that $\mathcal{P}_1$ occurs. If $v^*$ does not spawn a mutant before time $t_{\mathsf{max}}$ then $\mathcal{P}_1$ occurs, so suppose instead that $v^*$ spawns a mutant for the first time at some time $t_{\mathsf{sp}} \leq t_{\mathsf{max}}$. ( The "s" subscript in $t_{\mathsf{sp}}$ stands for "spawn".) We must show that $\sigma(t_{\mathsf{sp}}) \geq \lfloor \ell/(2r) \rfloor + 1$. This will ensure that $\mathcal{P}_1$ occurs since $\sigma(t)$ is monotonically increasing.

Since $v^*$ spawns no mutants before time $t_{\mathsf{sp}}$, we have $X_t \subseteq \{x_0, v_1, \ldots, v_k, v^*\}$ for all $t < t_{\mathsf{sp}}$, and so (recalling Definition 8.11)

$$i_{\mathsf{m}}(X, v^*, t_{\mathsf{sp}}) \leq \sum_{\substack{i \geq 1, \\ T_{\mathsf{m}}^i < t_{\mathsf{sp}}}} Y_i. \tag{8.8}$$

Since $\mathcal{E}_3$ occurs, we have

$$\sum_{\substack{1 \leq i \leq \lfloor \ell/(2r) \rfloor, \\ T_{\mathsf{m}}^i < t_{\mathsf{sp}}}} Y_i < \frac{1}{r}. \tag{8.9}$$

However, since $\mathcal{E}_1$ occurs and $v^*$ spawns a mutant at $t_{\mathsf{sp}}$, we have $i_{\mathsf{m}}(X, v^*, t_{\mathsf{sp}}) \geq 1/r$. Therefore, by (8.8) and (8.9),

$$\sum_{\substack{i \geq 1, \\ T_{\mathsf{m}}^i < t_{\mathsf{sp}}}} Y_i > \sum_{\substack{1 \leq i \leq \lfloor \ell/(2r) \rfloor, \\ T_{\mathsf{m}}^i < t_{\mathsf{sp}}}} Y_i.$$

Hence $T_{\mathsf{m}}^{\lfloor \ell/(2r) \rfloor + 1} < t_{\mathsf{sp}}$. Thus, $\sigma(t_{\mathsf{sp}}) \geq \lfloor \ell/(2r) \rfloor + 1$, and $\mathcal{P}_1$ occurs.

Finally, we show that $\mathcal{P}_1$, $\mathcal{E}_2$ and $\mathcal{E}_3$ together imply that $\mathcal{P}_2$ occurs. Suppose $\sigma(t_{x_0}) \leq \lfloor \ell/(2r) \rfloor$. We have $t_{x_0} \leq \ell m \leq t_{\mathsf{max}}$, so by $\mathcal{P}_1$, $v^*$ spawns no mutants in $[0, t_{x_0}]$. Hence as in (8.8), we have

$$i_{\mathsf{m}}(X, v^*, t_{x_0}) \quad \leq \sum_{\substack{i \geq 1, \\ T_{\mathsf{m}}^i < t_{x_0}}} Y_i = \sum_{\substack{i \geq 1, \\ T_{\mathsf{m}}^i \leq t_{x_0}}} Y_i.$$

Since $\sigma(t_{x_0}) \leq \lfloor \ell/(2r) \rfloor$, it follows by $\mathcal{E}_3$ that $i_{\mathsf{m}}(X, v^*, t_{x_0}) < 1/r$ and therefore $i_{\mathsf{n}}(X, v^*, t_{x_0}) \geq t_{x_0} - 1/r > t_{x_0} - 1$. Since $\mathcal{E}_2$ occurs, it follows that $v^*$ spawns a non-mutant onto $x_0$ at some time $t \leq t_{x_0}$. Since $v^*$ spawns no mutants in $[0, t_{x_0}]$, $x_0$ cannot become a mutant in $(t, t_{x_0}]$, so $x_0 \notin X_{t_{x_0}}$ and $\mathcal{P}_2$ occurs.

Thus $\mathcal{E}_1 \cap \mathcal{E}_2 \cap \mathcal{E}_3$ implies that $\mathcal{P}_1$ and $\mathcal{P}_2$ occur, and so the result follows from (8.7).

□

Lower bounds for the probabilities that properties $\mathcal{P}_3$–$\mathcal{P}_5$ hold follow from Chernoff bounds without too much difficulty.

**Lemma 8.27.** *Consider any $r > 1$. There is an $n_0$, depending on $r$, such that the following holds. Suppose $\ell \geq K r^4 \kappa \log n$, $m \geq 2$ and $n \geq n_0$. Let $R \in \{R_1, \ldots, R_\ell\}$ and let $v_1 \ldots v_k$ be the path associated with $R$. Let $x_0 \in R$. Let $X$ be the Moran process with $G(X) = \mathcal{S}_{k,\ell,m}$ and $X_0 = \{x_0\}$. Then $\mathbb{P}(\mathcal{P}_3) \geq 1 - 1/n$.*

*Proof.* Let $n_0$ be an integer which is sufficiently large with respect to $r$. Fix $i \in \mathbb{Z}_{\geq 0}$ — we will bound the probability that $v_1 \ldots v_k$ clears within $I_i$ (as in Definition 8.22).

Let $v_0 \in R \setminus \{x_0\}$ be arbitrary. For all $h \in \mathbb{Z}_{\geq 0}$, let

$$\begin{aligned}
T_{1,h} &= \min\{t \geq i\kappa + h \mid t = i\kappa + h + 1/2 \text{ or } N_{(v_0,v_1)} \text{ triggers at } t\}\\
T_{2,h} &= \min\{t > T_{1,h} \mid \ t = i\kappa + h + 1 \text{ or } (M_{(x_0,v_1)} \text{ does not trigger in } [T_{1,h}, t)\\
&\qquad\qquad\qquad\qquad\qquad\qquad \text{and } N_{(v_1,v_2)} \text{ triggers at } t)\}.
\end{aligned}$$

Let $\mathcal{E}_h$ be the event that $T_{1,h} < i\kappa + h + 1/2$ and $T_{2,h} < i\kappa + h + 1$.

The probability that the clock $N_{(v_0,v_1)}$ triggers in $[i\kappa+h, i\kappa+h+1/2)$ is $1-e^{-1/2}$. For any $t_1 \in [i\kappa+h, i\kappa+h+1/2)$ the probability that there is a $t_2 \in (t_1, i\kappa+h+1)$ such that $N_{(v_1,v_2)}$ triggers at $t_2$, and $M_{(x_0,v_1)}$ does not trigger in $[t_1, t_2]$ is $(1-e^{-(r+1)(i\kappa+h+1-t_1)})/(r+1)$. To see this, note that the $1-e^{-(r+1)(i\kappa+h+1-t_1)}$ factor corresponds to the probability that either $N_{(v_1,v_2)}$ or $M_{(x_0,v_1)}$ triggers in the relevant interval (together, they correspond to a Poisson process with rate $r+1$). The $1/(r+1)$ factor corresponds to the probability that it is actually $N_{(v_1,v_2)}$ rather than $M_{(x_0,v_1)}$ that triggers first. Since the relevant interval has length at least $1/2$, the product of these two factors is at least $(1 - e^{-(r+1)/2})/(r + 1)$. So

$$\mathbb{P}(\mathcal{E}_h) \geq \left(1-e^{-1/2}\right)\left(1-e^{-(r+1)/2}\right)/\left(r+1\right) \geq (1-e^{-1/2})(1-e^{-1})/(r+1) \geq \frac{1}{5(r+1)} \geq \frac{1}{10r}\,.$$

Moreover, the events $\{\mathcal{E}_h \mid h \in \mathbb{Z}_{\geq 0}\}$ are mutually independent, as they depend only on the behaviour of clocks in $\mathcal{C}(\mathcal{S}_{k,\ell,m})$ in fixed disjoint intervals. Thus

$$\begin{aligned}
\mathbb{P}\Big(\mathcal{E}_h \text{ holds for some } 0 \leq h \leq (Kr^4 \log n)/3\Big) &\geq 1 - \left(1 - \frac{1}{10r}\right)^{\lfloor (Kr^4 \log n)/3 \rfloor + 1}\\
&\geq 1 - \left(1 - \frac{1}{10r}\right)^{(Kr^4 \log n)/3}\\
&\geq 1 - e^{-(Kr^3 \log n)/30}\\
&\geq 1 - e^{-2\log n} = 1 - \frac{1}{n^2}\,. \qquad (8.10)
\end{aligned}$$

Now, for all integers $j$ with $3 \leq j \leq k+1$, define

$$T_j = \begin{cases} \min\{t > i\kappa + \kappa/2 \mid N_{(v_2,v_3)} \text{ triggers at } t\} & \text{if } j = 3,\\ \min\{t > T_{j-1} \mid N_{(v_{j-1},v_j)} \text{ triggers at } t\} & \text{if } 4 \leq j \leq k,\\ \min\{t > T_k \mid N_{(v_k,v^*)} \text{ triggers at } t\} & \text{if } j = k+1. \end{cases}$$

Note that if $T_{k+1} < (i+1)\kappa$, then $T_3, \ldots, T_{k+1} \in I_i$. By memorylessness and independence of distinct clocks in $\mathcal{C}(\mathcal{S}_{k,\ell,m})$, the random variables $T_3 - (i\kappa + \kappa/2), T_4 - T_3, \ldots, T_{k+1} - T_k$ are $k-1$ i.i.d. exponential variables with rate 1, and $T_{k+1} - (i\kappa + \kappa/2)$

is their sum. Corollary 8.5 applies because $\kappa/2 \geq 3(k-1)/2$, so

$$\begin{aligned}\mathbb{P}\Big(T_{k+1} < (i+1)\kappa\Big) &= \mathbb{P}\Big(T_{k+1} - (i\kappa + \kappa/2) < \kappa/2\Big) \\ &\geq 1 - e^{-\kappa/32} \geq 1 - e^{-(Kr^4 \log n)/32} \geq 1 - \frac{1}{n^2}. \qquad (8.11)\end{aligned}$$

Suppose $\mathcal{E}_h$ holds for some $0 \leq h \leq (Kr^4 \log n)/3$. Note that $T_{2,h} \leq i\kappa + \kappa/2$. Suppose further that $T_{k+1} < (i+1)\kappa$. Then setting $t_1 = T_{1,h}$, $t_2 = T_{2,h}$ and $t_j = T_j$ for all $j \in \{3, \dots, k+1\}$, we see that $t_1, \dots, t_{k+1}$ satisfy the requirements of Definition 8.22 and so $v_1 \dots v_k$ clears within $I_i$. It therefore follows by (8.10), (8.11), and a union bound that

$$\mathbb{P}(v_1 \dots v_k \text{ clears within } I_i) \geq 1 - \frac{2}{n^2}.$$

Hence by a union bound over all integers $i$ with $0 \leq i \leq t_{x_0}$ and by $t_{x_0} \leq n/4$ it follows that

$$\mathbb{P}(\mathcal{P}_3) \geq 1 - 2t_{x_0} \cdot \frac{2}{n^2} \geq 1 - \frac{n}{2} \cdot \frac{2}{n^2} \geq 1 - \frac{1}{n}. \qquad \square$$

**Lemma 8.28.** *Consider any $r > 1$. There is an $n_0$, depending on $r$, such that the following holds. Suppose $n \geq n_0$. Fix $x_0 \in R_1 \cup \dots \cup R_\ell$. Let $X$ be the Moran process with $G(X) = \mathcal{S}_{k,\ell,m}$ and $X_0 = \{x_0\}$. With probability at least $1 - 1/n^2$, for all integers $i$ with $0 \leq i \leq t_{x_0}$, every clock in $\mathcal{C}(\mathcal{S}_{k,\ell,m})$ triggers at most $\lfloor 2r\kappa \rfloor$ times in $I_i$.*

*Proof.* Let $n_0$ be an integer which is sufficiently large with respect to $r$. Fix a given clock $C \in \mathcal{C}(\mathcal{S}_{k,\ell,m})$, fix $i \in \mathbb{Z}_{\geq 0}$ with $i \leq t_{x_0}$, and write $a \leq r$ for the rate of $\mathcal{C}(\mathcal{S}_{k,\ell,m})$. The number of times that $C$ triggers in $I_i$ follows the Poisson distribution with parameter $a\kappa$. Since the number of triggers is an integer and $2r\kappa \geq 2a\kappa$, by Corollary 8.4 we have

$$\begin{aligned}\mathbb{P}(C \text{ triggers at most } \lfloor 2r\kappa \rfloor \text{ times in } I_i) &= \mathbb{P}(C \text{ triggers at most } 2r\kappa \text{ times in } I_i) \\ &\geq \mathbb{P}(C \text{ triggers at most } 2a\kappa \text{ times in } I_i) \\ &\geq 1 - e^{-a\kappa/3} \geq 1 - e^{-(Kr^4 \log n)/3}.\end{aligned}$$

There are at most $2n^2$ clocks in $\mathcal{C}(\mathcal{S}_{k,\ell,m})$ and at most $t_{x_0} + 1 \leq 2n^2$ choices of $i$. Thus by a union bound, with probability at least $1 - 4n^4 e^{-Kr^4 \log n/3} \geq 1 - 1/n^2$, no single clock in $\mathcal{C}(\mathcal{S}_{k,\ell,m})$ triggers more than $\lfloor 2r\kappa \rfloor$ times in any interval $I_i$ with $0 \leq i \leq t_{x_0}$. $\square$

The following corollary follows immediately from Lemma 8.28. Of course, the probability bound in the corollary can be strengthened to $1 - 1/n^2$, but we state what we will later use.

**Corollary 8.29.** *Consider any $r > 1$. There is an $n_0$, depending on $r$, such that the following holds. Suppose $n \geq n_0$. Fix $x_0 \in R_1 \cup \dots \cup R_\ell$. Let $X$ be the Moran process with $G(X) = \mathcal{S}_{k,\ell,m}$ and $X_0 = \{x_0\}$. Then $\mathbb{P}(\mathcal{P}_4) \geq 1 - 1/n$.* $\square$

The following lemma gives a lower bound on the probability that $\mathcal{P}_5$ occurs. In this lemma, we require that $m \geq 6r^2\kappa$, rather than $m \geq 2$, which we have so far been assuming.

**Lemma 8.30.** *Consider any $r > 1$. There is an $n_0$, depending on $r$, such that the following holds. Suppose $\ell \geq Kr^4\kappa \log n$, $m \geq 6r^2\kappa$ and $n \geq n_0$. Let $R \in \{R_1, \dots, R_\ell\}$ and let $v_1 \dots v_k$ be the path associated with $R$. Fix $x_0 \in R$. Let $X$ be the Moran process with $G(X) = \mathcal{S}_{k,\ell,m}$ and $X_0 = \{x_0\}$. Then $\mathbb{P}(\mathcal{P}_5) \geq 1 - 1/n$.*

*Proof.* Let $n_0$ be an integer which is sufficiently large with respect to $r$. Fix $i \in \mathbb{Z}_{\geq 0}$. For all $t \in I_i$, define the following events $\mathcal{E}_t^1$ and $\mathcal{E}_i^2$.

$\mathcal{E}_t^1$: $\min\{t' > t \mid \text{for some } v \neq x_0,\ N_{(v,v_1)} \text{ triggers at } t'\}$
$< \min\{t' > t \mid M_{(v_1,v_2)} \text{ triggers at } t'\}$.

$\mathcal{E}_i^2$: $M_{(x_0,v_1)}$ triggers at most $\lfloor 2r\kappa \rfloor$ times in $I_i$.

Thus $\mathcal{E}_t^1$ occurs if and only if $v_1$ clears before spawning a mutant within $(t, \infty)$. Let $T_{i,0} = i\kappa$, and let $T_{i,h}$ be the $h$'th time in $I_i$ at which the clock $M_{(x_0,v_1)}$ triggers, or $(i+1)\kappa$ if no such time exists. Note that if $\bigcap_{h=0}^{\lfloor 2r\kappa \rfloor} \mathcal{E}_{T_{i,h}}^1 \cap \mathcal{E}_i^2$ occurs, then $v_2$ is protected in $I_i$.

Now consider any $t \in I_i$ and let $f_t$ be a possible value of $\mathcal{F}_t(X)$. By memorylessness, we have

$$\mathbb{P}\Big(\mathcal{E}_t^1 \mid \mathcal{F}_t(X) = f_t\Big) = \frac{m-1}{m-1+r} = 1 - \frac{r}{m-1+r} \geq 1 - \frac{r}{m}.$$

In particular, since the event $T_{i,h} = t$ is determined by $\mathcal{F}_t(X)$, it follows by a union bound that

$$\mathbb{P}\left(\bigcap_{h=0}^{\lfloor 2r\kappa \rfloor} \mathcal{E}_{T_{i,h}}^1\right) \geq 1 - \frac{(\lfloor 2r\kappa \rfloor + 1)r}{m} \geq 1 - \frac{3r^2\kappa}{m}.$$

By Lemma 8.28 we have $\mathbb{P}(\mathcal{E}_i^2) \geq 1 - 1/n^2$, so it follows by a union bound that

$$\mathbb{P}(v_2 \text{ is protected in } I_i) \geq \mathbb{P}\left(\bigcap_{h=0}^{\lfloor 2r\kappa \rfloor} \mathcal{E}_{T_{i,h}}^1 \cap \mathcal{E}_i^2\right) \geq 1 - \frac{3r^2\kappa}{m} - \frac{1}{n^2}. \tag{8.12}$$

Since $I_0, I_1, \dots$ are disjoint intervals, the events that $v_2$ is or is not protected in these intervals are independent by memorylessness. Thus the number of intervals $I_i$ with $0 \leq i \leq t_{x_0}/\kappa$ in which $v_2$ is not protected is stochastically dominated above by a binomial distribution consisting of $\lfloor t_{x_0}/\kappa \rfloor + 1$ Bernoulli trials, each with success probability $3r^2\kappa/m + 1/n^2$. This distribution has expectation

$$\left(\left\lfloor \frac{t_{x_0}}{\kappa} \right\rfloor + 1\right)\left(\frac{3r^2\kappa}{m} + \frac{1}{n^2}\right) \leq \frac{4r^2 t_{x_0}}{m} = \frac{4r^2\ell}{Kr^4\kappa},$$

so by Lemma 8.6 we have

$$\mathbb{P}(\overline{\mathcal{P}_5}) \leq e^{-(1/6)8r^2 t_{x_0}/m} = e^{-(4/3)r^2\ell/(Kr^4\kappa)} \leq e^{-(4/3)r^2 \log n} \leq \frac{1}{n}.$$

Here the penultimate inequality follows since $\ell \geq Kr^4\kappa \log n$ by hypothesis. The result therefore follows. □

Now that we have proved lower bounds on the probability that each of $\mathcal{P}_1$–$\mathcal{P}_5$ occur, Lemma 8.23 follows easily.

**Lemma 8.23.** *Consider any $r > 1$. There is an $n_0$, depending on $r$, such that the following holds. Suppose $\ell \geq Kr^4\kappa \log n$, $m \geq 6r^2\kappa$ and $n \geq n_0$. Let $R \in \{R_1, \ldots, R_\ell\}$, and let $v_1 \ldots v_k$ be the path associated with $R$. Fix $x_0 \in R$. Let $X$ be the Moran process with $G(X) = \mathcal{S}_{k,\ell,m}$ and $X_0 = \{x_0\}$. Then, with probability at least $1/(7Kr^4\kappa)$, the following events occur simultaneously.*

*$\mathcal{P}_1$: $\forall t \leq t_{\mathsf{max}}$, $\sigma(t) \geq \lfloor \ell/(2r) \rfloor + 1$ or $v^*$ does not spawn a mutant at time $t$.*

*$\mathcal{P}_2$: $\sigma(t_{x_0}) \geq \lfloor \ell/(2r) \rfloor + 1$ or $x_0 \notin X_{t_{x_0}}$.*

*$\mathcal{P}_3$: For all integers $i$ with $0 \leq i \leq t_{x_0}$, $v_1 \ldots v_k$ clears within $I_i$.*

*$\mathcal{P}_4$: For all integers $i$ with $0 \leq i \leq t_{x_0}$, the clock $M_{(v_k,v^*)}$ triggers at most $\lfloor 2r\kappa \rfloor$ times within $I_i$.*

*$\mathcal{P}_5$: For all but at most $8r^2 t_{x_0}/m$ integers $i$ with $0 \leq i \leq t_{x_0}/\kappa$, $v_2$ is protected in $I_i$.*

*Proof.* $\mathbb{P}(\mathcal{P}_1 \cap \cdots \cap \mathcal{P}_5) \geq \mathbb{P}(\mathcal{P}_1 \cap \mathcal{P}_2) - \mathbb{P}(\overline{\mathcal{P}_3}) - \mathbb{P}(\overline{\mathcal{P}_4}) - \mathbb{P}(\overline{\mathcal{P}_5})$. Let $n_0$ be an integer which is sufficiently large with respect to $r$. Then we bound each term on the right-hand side by applying (in order) Lemma 8.26, Lemma 8.27, Corollary 8.29 and Lemma 8.30 to obtain

$$\mathbb{P}(\mathcal{P}_1 \cap \cdots \cap \mathcal{P}_5) \geq \frac{1}{6Kr^4\kappa} - \frac{3}{n} \geq \frac{1}{7Kr^4\kappa},$$

as required. The final inequality follows since, by hypothesis, $\kappa \leq \ell/(Kr^4 \log n) \leq n/\log n$. □

We are now at last in a position to prove Lemma 8.17, which we will then use to prove Theorem 8.13.

**Lemma 8.17.** *Consider any $r > 1$. There is an $n_0$, depending on $r$, such that the following holds. Suppose that $\ell \geq Kr^4\kappa \log n$, $m \geq 6r^2\kappa$ and $n \geq n_0$. Fix $x_0 \in R_1 \cup \cdots \cup R_\ell$. Let $X$ be the Moran process with $G(X) = \mathcal{S}_{k,\ell,m}$ and $X_0 = \{x_0\}$. Then the extinction probability of $X$ is at least $1/(7Kr^4\kappa)$.*

*Proof.* Let $n_0$ be an integer which is sufficiently large with respect to $r$. Let $R$ be a reservoir in $\{R_1, \ldots, R_\ell\}$ and let $v_1 \ldots v_k$ be the path associated with $R$. Suppose $x_0 \in R$. By Lemma 8.23, it suffices to assume that $\mathcal{P}_1 - \mathcal{P}_5$ occur and to prove that $X$ goes extinct.

Recall the definition of $\sigma(t)$ from Definition 8.19. Note that $\sigma(0) = 0$ and $\sigma(t)$ is monotonically increasing in $t$. We will first bound $\sigma(t_{x_0})$ from above (assuming that

$\mathcal{P}_1$–$\mathcal{P}_5$ occur). Consider an interval $I_i$ with $0 \le i \le t_{x_0}$ (technically, we need only consider $0 \le i \le t_{x_0}/\kappa$, but the extra generality does no harm and we will later need to consider slightly larger $i$). Note that $I_i \subseteq [0, t_{\mathsf{max}}]$ since $(i+1)\kappa \le 2t_{x_0}\kappa = t_{\mathsf{max}}/(Kr^4)$. Suppose that $\sigma(i\kappa) \le \lfloor \ell/(2r) \rfloor - \lfloor 2r\kappa \rfloor$. We will derive an upper bound on $\sigma((i+1)\kappa)$ by splitting into cases.

**Case 1: $\boldsymbol{i > 0}$ and $\boldsymbol{v_2}$ is protected in $\boldsymbol{I_{i-1}}$ and $\boldsymbol{I_i}$.** First note that since $\mathcal{P}_1$ occurs and $\sigma(i\kappa) \le \lfloor \ell/(2r) \rfloor$, $v^*$ does not spawn a mutant over the course of $[0, i\kappa]$ and so

$$X_t \subseteq \{x_0, v_1, \ldots, v_k, v^*\} \text{ for all } t \in [0, i\kappa].$$

Now, suppose for a contradiction that $v_2$ becomes a mutant at some time $\widehat{t}_2 \in I_{i-1}$. Then $v_1$ must have become a mutant beforehand. Let $\widehat{t}_1$ be the latest time in $[0, \widehat{t}_2]$ at which this occurs, and note that $M_{(x_0,v_1)}$ must have triggered at time $\widehat{t}_1$. Since $v_2$ is protected in $I_{i-1}$, if it were the case that $\widehat{t}_1 \in I_{i-1}$, then $v_1$ would clear before spawning within $(\widehat{t}_1, i\kappa]$ and so $v_1$ would die in $(\widehat{t}_1, \widehat{t}_2)$. This is impossible since $v_1$ spawns a mutant at time $\widehat{t}_2$, and $v_1$ does not become a mutant in $(\widehat{t}_1, \widehat{t}_2]$ by the definition of $\widehat{t}_1$. We therefore have $\widehat{t}_1 \notin I_{i-1}$, so $\widehat{t}_1 \le (i-1)\kappa$. Since $v_2$ is protected in $I_{i-1}$, $v_1$ clears before spawning a mutant within $I_{i-1}$, so $v_1$ dies in $((i-1)\kappa, \widehat{t}_2)$ — again contradicting the fact that $v_1$ spawns a mutant at time $\widehat{t}_2$. Thus we can conclude that $v_2$ does not become a mutant in $I_{i-1}$.

Since $\mathcal{P}_3$ occurs, $v_1 \ldots v_k$ clears within $I_{i-1}$. Let $v_0 \in R \setminus \{x_0\}$ and $t_1, \ldots, t_{k+1} \in I_{i-1}$ be as in Definition 8.22. Since $N_{(v_0,v_1)}$ triggers at time $t_1$, it follows that $v_1 \notin X_{t_1}$. Since $M_{(x_0,v_1)}$ does not trigger in $[t_1, t_2]$, $v_1$ does not become a mutant in $[t_1, t_2]$ and so $v_1 \notin X_{t_2}$. Since $N_{(v_1,v_2)}$ triggers at time $t_2$, it follows that $v_2 \notin X_{t_2}$. We have already seen that $v_2$ does not become a mutant in $I_{i-1}$, so it follows that $v_2 \notin X_t$ for all $t \in [t_2, i\kappa]$.

Now, $v_3 \notin X_{t_3}$ since $N_{(v_2,v_3)}$ triggers at time $t_3 \in [t_2, i\kappa]$. Since $v_2$ is a non-mutant throughout $[t_2, i\kappa]$, it follows that $v_3 \notin X_t$ for all $t \in [t_3, i\kappa]$. Repeating the argument for $t_4, \ldots, t_{k+1}$, we see that $v_2, \ldots, v_k, v^* \notin X_{i\kappa}$ and hence $X_{i\kappa} \subseteq \{x_0, v_1\}$.

Since $X_{i\kappa} \subseteq \{x_0, v_1\}$, no mutants can be spawned in $I_i$ until $v_2$ next becomes a mutant. However, by the same argument as above, the fact that $v_2$ is protected in $I_i$ implies that $v_2$ does not become a mutant in $I_i$. Hence $X_t \subseteq \{x_0, v_1\}$ for all $t \in I_i$, and in particular $v_k$ does not spawn a mutant onto $v^*$ in $I_i$. Thus $\sigma((i+1)\kappa) = \sigma(i\kappa)$ This gives the desired upper bound on $\sigma((i+1)\kappa)$.

**Case 2: Case 1 does not hold.** Suppose for a contradiction that $\sigma((i+1)\kappa) \ge \sigma(i\kappa) + \lfloor 2r\kappa \rfloor + 1$. Then $v_k$ spawns a mutant onto $v^*$ at least $\lfloor 2r\kappa \rfloor + 1$ times in $I_i$, contradicting $\mathcal{P}_4$. Thus $\sigma((i+1)\kappa) \le \sigma(i\kappa) + \lfloor 2r\kappa \rfloor$. Again, we have the desired upper bound on $\sigma((i+1)\kappa)$.

Combining Cases 1 and 2, we have proved that whenever $0 \le i \le t_{x_0}$ and $\sigma(i\kappa) \le$

$\lfloor \ell/(2r)\rfloor - \lfloor 2r\kappa\rfloor$,

$$\begin{aligned}&\sigma((i+1)\kappa) = \sigma(i\kappa), \text{ if } i>0 \text{ and } v_2 \text{ is protected in } I_{i-1} \text{ and } I_i, \text{ and}\\ &\sigma((i+1)\kappa) \le \sigma(i\kappa) + \lfloor 2r\kappa\rfloor, \text{ otherwise.}\end{aligned} \tag{8.13}$$

Since $\mathcal{P}_5$ occurs and $\ell \ge Kr^4\kappa\log n$, the number of intervals $I_i$ such that $0 \le i \le \lfloor t_{x_0}/\kappa\rfloor$ and Case 1 does not hold is at most

$$\begin{aligned}1 + 2\left|\left\{i \in \mathbb{Z}_{\ge 0} \;\middle|\; i \le \left\lfloor \frac{t_{x_0}}{\kappa}\right\rfloor, v_2 \text{ is not protected in } I_i\right\}\right| &\le 1 + 2\cdot\frac{8r^2 t_{x_0}}{m}\\ &= 1 + \frac{16r^2\ell}{Kr^4\kappa} \le \frac{17r^2\ell}{Kr^4\kappa}.\end{aligned}$$

Moreover, again using the fact that $\ell \ge Kr^4\kappa\log n$,

$$\lfloor 2r\kappa\rfloor \cdot \frac{17r^2\ell}{Kr^4\kappa} \le \frac{34r^3\ell}{Kr^4} = \frac{34\ell}{Kr} \le \left\lfloor\frac{\ell}{2r}\right\rfloor - \lfloor 2r\kappa\rfloor.$$

Since $\sigma(0) = 0$, it therefore follows by repeated application of (8.13) that

$$\sigma(t_{x_0}) \le \sigma\left(\left\lfloor\frac{t_{x_0}}{\kappa} + 1\right\rfloor\kappa\right) \le \left\lfloor\frac{\ell}{2r}\right\rfloor - \lfloor 2r\kappa\rfloor. \tag{8.14}$$

Now consider the behaviour of the process in the interval $(t_{x_0}, \lfloor t_{x_0}/\kappa + 2\rfloor\kappa]$. From (8.14), we have that $\sigma(\lfloor t_{x_0}/\kappa + 1\rfloor\kappa) \le \lfloor \ell/(2r)\rfloor - \lfloor 2r\kappa\rfloor$, so by (8.13) it follows that $\sigma(\lfloor t_{x_0}/\kappa + 2\rfloor\kappa) \le \lfloor \ell/(2r)\rfloor$ and so, since $\mathcal{P}_1$ occurs, $v^*$ does not spawn a mutant in the interval $[0, \lfloor t_{x_0}/\kappa + 2\rfloor\kappa]$.

Since $\mathcal{P}_2$ occurs and (8.14) holds, we have $x_0 \notin X_{t_{x_0}}$, so for all $t \in (t_{x_0}, \lfloor t_{x_0}/\kappa+2\rfloor\kappa]$, we have $X_t \subseteq \{v_1, \dots, v_k, v^*\}$. Since $\mathcal{P}_3$ occurs, $v_1 \dots v_k$ clears within $I_{\lfloor t_{x_0}/\kappa+1\rfloor}$. Let $v_0 \in R\setminus\{x_0\}$ and the sequence of times $t_1, \dots, t_{k+1} \in I_{\lfloor t_{x_0}/\kappa+1\rfloor}$ be as in Definition 8.22. Then for all $i \in [k]$, $N_{(v_{i-1},v_i)}$ triggers at time $t_i$ and so $v_i \notin X_t$ for all $t \in [t_i, \lfloor t_{x_0}/\kappa+2\rfloor]\kappa$. Likewise, $v^* \notin X_t$ for all $t \in [t_{k+1}, \lfloor t_{x_0}/\kappa + 2\rfloor]\kappa$. In particular, $X_{\lfloor t_{x_0}/\kappa+2\rfloor\kappa} = \emptyset$, so $X$ goes extinct and the result holds. □

## 8.4.4 Proving the main theorem (Theorem 8.13)

We now have everything we need to prove Theorem 8.13, which follows relatively easily from Lemmas 8.14, 8.15 and 8.17.

**Theorem 8.13.** *Let $r > 1$. Then there exists a constant $c_r > 0$ (depending on $r$) such that the following holds for all positive integers $k$, $\ell$ and $m$. Choose $x_0$ uniformly at random from $V(\mathcal{S}_{k,\ell,m})$. Let $X$ be the Moran process (with fitness $r$) with $G(X) = \mathcal{S}_{k,\ell,m}$ and $X_0 = \{x_0\}$. Then the probability that $X$ goes extinct is at least $1/(c_r(n\log n)^{1/3})$.*

*Proof.* Fix $r > 1$ as in the statement of the theorem. Recall from Definitions 8.16 and 8.19 that $K = 70$ and $\kappa = \max\{3k, Kr^4\log n\}$. Let $n_0$ be the smallest integer such that, for $n \geq n_0$, Lemma 8.17 and Lemma 8.26 applies and also

$$(n \log n)^{1/3} \geq n^{1/3} \geq \max\{18r^2, 6Kr^6 \log n, Kr^4(\log n)^2\}\,. \tag{8.15}$$

We split into cases depending on the values of $k$, $\ell$, $m$ and $n$. We show that in each case, the statement of the theorem holds, provided $c_r \geq \max\{2rn_0, 156r^6K\}$.

**Case 1: $\boldsymbol{n < n_0}$.** We show that with probability at least $1/2rn_0$, $x_0$ dies before spawning a single mutant. Indeed, at the start of the process $x_0$ spawns a mutant with rate $r$, and every choice of $x_0 \in V(\mathcal{S}_{k,\ell,m})$ has an in-neighbour so $x_0$ dies with rate at least $1/n$. Thus $X$ goes extinct with probability at least

$$\frac{\frac{1}{n}}{\frac{1}{n}+r} \geq \frac{1}{2rn} \geq \frac{1}{2rn_0},$$

so the result follows since $c_r \geq 2rn_0$.

**Case 2: $\boldsymbol{n \geq n_0}$, $\boldsymbol{m < k(n\log n)^{1/3}}$.** By Lemma 8.14, $X$ goes extinct with probability at least

$$\frac{k}{2r(m+k)} \geq \frac{k}{2r(k(n\log n)^{1/3}+k)} \geq \frac{1}{4r(n\log n)^{1/3}},$$

where the final inequality holds since $(n\log n)^{1/3} \geq 1$. The result follows since $c_r \geq 4r$.

**Case 3: $\boldsymbol{n \geq n_0}$, $\boldsymbol{m \geq k(n\log n)^{1/3}}$ and $\boldsymbol{\ell < 3Kr^4(n\log n)^{1/3}}$.** Note that

$$\mathbb{P}(x_0 \in R_1 \cup \cdots \cup R_\ell) = \frac{\ell m}{\ell(m+k)+1} \geq \frac{m}{m+k+1} \geq \frac{1}{2}, \tag{8.16}$$

where the final inequality is valid since $m \geq k(n\log n)^{1/3} \geq 2k$. We will therefore condition on $x_0 \in R_1 \cup \cdots \cup R_\ell$. Moreover, we have $m \geq k(n\log n)^{1/3} \geq 12r$. Thus by Lemma 8.15 and (8.16), $X$ goes extinct with probability at least

$$\frac{1}{2}\cdot\frac{1}{26r^2\ell} \geq \frac{1}{156r^2Kr^4(n\log n)^{1/3}},$$

so the statement holds since $c_r \geq 156Kr^6$.

**Case 4: $\boldsymbol{n \geq n_0}$, $\boldsymbol{m \geq k(n\log n)^{1/3}}$ and $\boldsymbol{\ell \geq 3Kr^4(n\log n)^{1/3}}$.** Note that

$$m \geq k(n\log n)^{1/3} \geq 6r^2 \max\{3k, Kr^4\log n\} = 6r^2\kappa.$$

We will also show that $\ell \geq Kr^4\kappa\log n$, in order to apply Lemma 8.17. Since $\ell \geq 3Kr^4(n\log n)^{1/3}$ and $n = \ell(m+k)+1 \geq \ell m$, we have $m \leq n/\ell \leq (n^2/\log n)^{1/3}$. It is also immediate from (8.15) and the hypothesis on $\ell$ that

$$\ell \geq K^2r^8(\log n)^2. \tag{8.17}$$

Therefore,

$$3k \leq \frac{3m}{(n \log n)^{1/3}} \leq \frac{3n^{1/3}}{(\log n)^{2/3}} = \frac{3Kr^4(n \log n)^{1/3}}{Kr^4 \log n} \leq \frac{\ell}{Kr^4 \log n}. \tag{8.18}$$

It therefore follows from (8.17), (8.18) and the definition of $\kappa$ (Definition 8.16) that $\ell \geq Kr^4 \kappa \log n$, and so we may apply Lemma 8.17.

As in Case 3, (8.16) holds. Thus by (8.16) and Lemma 8.17, $X$ goes extinct with probability at least $1/(14Kr^4\kappa)$. By (8.18) we have $3k \leq 3n^{1/3}$, and so

$$\frac{1}{14Kr^4\kappa} = \frac{1}{14Kr^4 \max\{3k, Kr^4 \log n\}} \geq \frac{1}{14Kr^4 \max\{3n^{1/3}, Kr^4 \log n\}} = \frac{1}{42Kr^4 n^{1/3}},$$

and the result follows since $c_r \geq 156Kr^6$. □

# Chapter 9

# Future directions

Recall the conjecture of Faben and Jerrum.

**Conjecture 1.7** (Faben and Jerrum)**.** Let $H$ be a graph. If its involution-free reduction $H^*$ has at most one vertex, then $\#_2$HomsTo$H$ is in $\mathsf{P}$; otherwise, $\#_2$HomsTo$H$ is $\#_2\,\mathsf{P}$-complete.

Our results show that this conjecture is true when $H$ is a cactus graph or a square-free graph. To prove this conjecture for all graphs $H$, one has to show that for all involution-free graphs that contain a 4-cycle and contain at least two cycles that share at least one edge, $\#_2$HomsTo$H$ is $\#\,\mathsf{P}$-complete.

We do not believe that the restriction to square-free graphs is fundamental, since our results on pinning apply to all involution-free graphs for counting modulo 2 (Section 2.4) and neither our definition of hardness gadgets (Definition 5.3) nor our proof that the existence of a $(0, 2)$-gadget for $H$ implies that $\#_2$HomsTo$H$ is $\#_2\,\mathsf{P}$-complete (Corollary 4.5) requires $H$ to be square-free. However, all the actual $(0, 2)$-gadgets that we find for the target graphs $H$ do rely on the absence of 4-cycles, as discussed in Section 5.3.2, and removing this restriction seems technically challenging. We note that dealing with 4-cycles also caused significant difficulties in cactus graphs (Chapter 4).

It is interesting to see whether a dichotomy can be proven for $\#_p$HomsTo$H$ for all primes $p$. Theorem 1.10 shows that for all asymmetric trees the polynomial-time cases are the ones described in Corollary 2.9. By adapting the techniques presented in Chapters 4 and 5 for locating gadgets in graphs we could use Corollary 3.12 to show that the dichotomy of $\#_p$HomsTo$H$ holds for other families of asymmetric graphs besides trees. Our pinning approach though requires the "orbit compatibility" property (Definition 2.22), which can only be guaranteed when either $p = 2$ or $H$ is an asymmetric graph. To overcome this obstacle a more powerful pinning technique is needed. When considering the problem $\#_k$HomsTo$H$ for general integers $k$ and not just primes, the polynomial-time algorithm does not hold. Furthermore, as we show in Section 3.6, techniques available to the more general framework of the constraint satisfaction problem, that reduce the complexity of counting modulo an integer to counting modulo its prime factors, do not apply to graph homomorphism problems.

Another important open problem in the area of modular counting is extending the dichotomy of Guo et al. [49] for $\#_k$CSP to include functions of arbitrary domain size. A dichotomy for $\#_k$CSP of arbitrary domain size would imply a dichotomy for $\#_k$HomsTo$H$.

Further interesting question in the area is to consider the problem of counting (or counting modulo an integer) graph homomorphisms from restricted inputs, such as planar graphs. This would be of particular interest in the modular counting setting, as the problems that have different tractability depending on the value of the modulus are, in their majority, planar problems —like Valiant's famous restricted version of 3-SAT [72].

Another way of restricting the input would be to introduce lists, as we did with matrix partitions. The input for counting list homomorphisms modulo an integer is a graph $G$ and a list function $L$ from the vertices of $G$ to sets of vertices of the target graph $H$ and the output is the number (modulo an integer) of homomorphisms from $G$ to $H$ that respect $L$. Lists could potentially break symmetries in modular counting, in the following way. Assume we are interested in counting (modulo 2) the number of homomorphisms of a graph $G$ to a graph $H$ that respect a list function $L$. Further assume that $H$ has an involution exchanging $v \in V(H)$ with $u \in V(H)$. Unless $L(v) = L(u)$, we cannot reduce $H$ by involutions by removing $u$ and $v$ from $H$, which is what we did in the unlisted version of the problem.

For counting matrix partitions of graphs, the unlisted version of the problem for large matrices (of size at least $5\times5$) remains open. The lists enable us to pin for free and that is what allowed us to prove hardness for the listed version of problem. Dyer, Goldberg and Richerby [26] conjectured that the dichotomy criterion we proved for the listed problem is the same for the unlisted problem. Studying the listed or unlisted version of the problem in the modular counting setting is also interesting. For the unlisted version, one could take advantage of the cancellations and develop a pinning approach that could give an interesting dichotomy theorem.

# Appendix A

# The program used in the proof of Lemma 6.18

In this appendix we have included the source code and the output file of the program we used for proving Lemma 6.18. Lemma 6.18 identifies values of the parameters $\gamma$, $\lambda$ and $p$, where $p < 100$, for which $\#_p Z_{\gamma,\lambda}$ is $\#_p\,\mathsf{P}$-hard.

The program, written in Python, cycles over all values of the parameters $\gamma$, $\lambda$, and $p$ for $p < 100$. For each parameter value the program then searches for a suitable gadget, by computing and comparing $Z^{\tau(J),0}_{\gamma,\lambda}(G(J), x)$ and $Z^{\tau(J),1}_{\gamma,\lambda}(G(J), x)$ for a set of partially pinned graphs with distinguished vertices as given in the proof of Lemma 6.18.

The only value of the parameters it fails to find a gadget is $p = 41$, $\gamma = 18$ and $\lambda = 6$. In fact Lemma 6.18 can be strengthened to include the rest of the values of the parameters for $p = 41$.

## Source Code

```
#!/path/to/interpreter

#The following program searches for possible gadgets that will give hardness for the problem
#of computing the partition function of a two-spin system with external field
#on a multigraph modulo a prime p.
import math

#Compute n choose r.
def nCr(n,r):
        f = math.factorial
        if n<r: g=0
        else: g=f(n) / f(r) / f(n-r)

        return g

#Compute the partition function of the (k+1)-clique when the distinguished vertex is mapped to 0.
#g corresponds to \gamma and l to \lambda.
#The l factor is taken away from the 0-partition and will be added later.
def z_out(k,g,l):
        return sum(nCr(k,i)*pow(l,i)*(pow(g,nCr(k-i,2))) for i in range(k+1))

#Compute the partition function of the (k+1)-clique when the distinguished vertex is mapped to 1.
#g corresponds to \gamma and l to \lambda.
def z_in(k,g,l):
        return sum(nCr(k,i)*(pow(l,i))*(pow(g,nCr(k-i,2)))*(pow(g,(k-i))) for i in range(k+1))


#A simple primality testing.
```

```
def is_prime(n):
        if n % 2 == 0 and n > 2:
                return False
        return all(n % i for i in range(3, int(math.sqrt(n)) + 1, 2))


#Make the list containing the primes.
primes=[]

for i in range(4,100): #The program starts with p=5 as for p=3 we already know the complexity.
        if is_prime(i): primes.append(i)

print primes


#Check if a list of lists is all 1.
#This will be used to check if we found all gadgets for a prime p.
def allones(c):
        flag=1
        for i in range((len(c))):
                for j in range(len(c[i])):
                        if c[i][j]!=1: flag=0
        return flag


#Find the non-zero elements of a matrix and return them in a list.
#This is used to identify the values g,l,p for which we haven't found a gadget yet.
def unsolved(c):
        u=[]
        flag=1
        for i in range((len(c))):
                for j in range(len(c[i])):
                        if c[i][j]!=1: u.append([i+2,j+2])#Add 2 as g,l start from 2.
        return u


#The first gadget we try is the pinned-vertex-parallel-edges gadget,
#together with a clique.
#Compute z_in and z_out and store it in a list to speed up computations.
#k[1] is the size of the clique.
def precomp(k,g,l):
        z=[0,0]
        z[0]=l*z_out(k[1],g,l)
        z[1]=z_in(k[1],g,l)
        return z


#This function checks wether the above gadget works.
#k[0] is the number of edges incident to the pinned vertex.
#We check if the 0-partition is non-zero and equivalent to the 1-partition modulo p.
def clique(k,z,g,p):
        if z[0]%p!=0 and z[0]%p==(pow(g,k[0])*z[1])%p:
                return 1
        else:
                return 0


f1=open('output_gadgets1.txt','w') #Open the output file.

u=[] #List containing the unsolved values of k,g,l.


#In this routine we check for gadgets.
#We will increment the number of parallel edges k[0] and the size of the clique k[1]
#In nested loops.

f1.write('Search␣for␣gadgets␣by␣increasing␣the␣size␣of␣one␣clique.')
f1.write('\n')
for p in primes:
        c=[[0 for i in range(2,p-1)] for j in range(2,p)] #c[g][l]=1 when we have a gadget.
        for g in range(2,p-1): #Loop over values of g. Excluded g=0,1,-1.
                for l in range(2,p): #Loop over values of l. Excluded l=0,1.
                        flag=0
                        k=[0 for i in range(2)]
                        for k[1] in range(100): #Loop over the clique size.
                                z=precomp(k,g,l)
                                for k[0] in range(p): #Loop over the number of edges.
                                        flag=clique(k,z,g,p)
                                        if flag==1: break
                                if flag==1:
                                        c[l-2][g-2]=1 #Indicate in c that we have a gadget.
                                        break
```

```
            if flag==0: u.append([p,[g,l]]) #Add the unsolved parameters to u.
    solved=allones(c) #Check if we covered all g,l for this prime.
    print('For',p,'covered␣all␣is␣',solved==1)
    f1.write('For␣p=')
    p_txt=str(p)
    f1.write(p_txt)
    f1.write('␣covered␣all␣g,l␣is␣') #Write to the file if gadgets have been found for all g,l.
    solv_txt=str(solved==1)
    f1.write(solv_txt)
    f1.write('\n')




print u #The list u contains the values [p,[g,l]] for which he have not found gadgets yet.
b=str(u)
f1.write('\n')
f1.write('Unresolved␣values␣of␣[p,[g,l]]␣after␣first␣search:')
f1.write('\n')
f1.write(b)
f1.write('\n')


# We will now search for gadgets for the elements of u.

#Create the matrix containing z_in and z_out for every gadget we try.
#The vector x contains the number of each gadget we combine.
#The gadgets are a pinned vertex with x[0] parallel edges,
#a 2-path, a 3-path and cliques of increasing size.
#In x, the number of P_2 and P_3 we include come before the cliques.
#in contrast to as it is written in the proof of the Lemma.
def precompute(x,g,l):
    z=[[0 for i in range(len(x))] for j in range(2)]
    z[0][0]=l
    z[1][0]=g
    z[0][1]=pow(l,2)+2*l+g #The 0 partition of a 2-path /l.
    z[1][1]=pow(l,2)+l*g+l+pow(g,2) #The 1 partition of a 2-path.
    z[0][2]=pow(l,3)+3*pow(l,2)+ 2*l*g+l+pow(g,2) #The 0 parition of a 3-path /l.
    z[1][2]=pow(l,3)+2*pow(l,2)+pow(l,2)*g+2*l*g+l*pow(g,2)+pow(g,3)
                                                     #The 1 parition of a 3-path /l.
    for j in range(3,len(x)):
        z[0][j]=z_out(j-2,g,l) #The 0-partition of the clique.
        z[1][j]=z_in(j-2,g,l)  #The 1-partition of the clique.
    return z


#This function checks wether the gadget works.
#x is the vector containing the clique sizes.
#Compute the 0-partition and 1-partition #accodring to observation 6.15 in text.
#Then check that the 0-partition is non-zero and equivalent to the 1-partition modulo p.
def gadget(x,z,p):
    w_out=z[0][0]
    w_in=pow(z[1][0],x[0]) #The partition of the pinned vertex parallel-edges gadget.
    for i in range(1,len(x)):
        w_out=w_out*pow(z[0][i],x[i])
        w_in=w_in*pow(z[1][i],x[i])
    if w_out%p!=0 and w_in%p==w_out%p: #Check wether the gadget works.
        return 1
    else:
        return 0


#Cycle over the possible values of k_i, where k_i is the number of cliques of size i.
#Increase number of the smallest not maxed out set of cliqes.
#The numbers k-i are stored in a vector x.
#We only check up to p-1 copies of each gadget --due to modulo arithmetic.
def inc(x,p):
    k=0
    while x[k]==p-1:
        x[k]=0
        k=k+1
    x[k]=x[k]+1
    return x


f1.write('\n')
f1.write('Finding␣gadgets␣for␣the␣remaining␣values.')
f1.write('\n')


#Test a combination of paths and cliques as constructed from precompute
#for possible gadgets in the remaining values.
```

```
w=[]#This is the list of values for which the next search for gadgets fails.

for i in range(len(u)): #Loop over the elements of u.
        p=u[i][0]
        g=u[i][1][0]
        l=u[i][1][1]
        print (p,g,l)
        x=[0 for j in range(5)] #The initial vector of number of gadgets of each type.
        y=[p-1 for j in range(5)] #The end vector of number of gadgets of each type.
        z=precompute(x,g,l)
        flag=0
        while flag==0 and x!=y:
                inc(x,p) #Increment the vector of number of gadgets.
                flag=gadget(x,z,p) #Check if the gadget works.
                if flag==1:
                        print x #Return the gadget vector that works.
                        a=str(u[i])
                        b=str(x)
                        f1.write(a)
                        f1.write('␣gadget␣found␣with␣x=')
                        f1.write(b)
                        f1.write('\n')
        if flag==0: w.append(u[i]) #If not solved add the values [p,[g,l]] to the list w.


# Return the values for which we don't have a hardness proof.
print w

f1.write('\n')
w_txt=str(w)
f1.write('Unresolved␣values␣of␣[p,[g,l]]␣after␣final␣search:')
f1.write('\n')
f1.write(w_txt)
f1.close()
```

# Output File

```
Search for gadgets by increasing the size of one clique.
For p=5 covered all g,l=True
For p=7 covered all g,l=True
For p=11 covered all g,l=True
For p=13 covered all g,l=True
For p=17 covered all g,l=True
For p=19 covered all g,l=True
For p=23 covered all g,l=True
For p=29 covered all g,l=True
For p=31 covered all g,l=False
For p=37 covered all g,l=True
For p=41 covered all g,l=False
For p=43 covered all g,l=False
For p=47 covered all g,l=True
For p=53 covered all g,l=True
For p=59 covered all g,l=True
For p=61 covered all g,l=False
For p=67 covered all g,l=False
For p=71 covered all g,l=False
For p=73 covered all g,l=False
For p=79 covered all g,l=False
For p=83 covered all g,l=True
For p=89 covered all g,l=False
For p=97 covered all g,l=False

First search unresolved values of [p,[g,l]]:
[[31, [5, 13]], [31, [5, 17]], [31, [5, 23]], [31, [25, 3]], [31, [25, 23]], [41, [18, 6]],
[43, [6, 2]], [43, [36, 4]], [43, [36, 9]], [43, [36, 11]], [43, [36, 12]], [43, [36, 15]],
[43, [36, 23]], [43, [36, 24]], [61, [13, 31]], [61, [13, 37]], [61, [13, 50]], [61, [13, 51]],
[61, [47, 40]], [61, [47, 51]], [67, [29, 2]], [67, [29, 7]], [67, [29, 34]], [67, [29, 39]],
[67, [29, 43]], [67, [29, 48]], [67, [29, 52]], [67, [29, 53]], [67, [29, 58]], [67, [29, 59]],
[67, [37, 36]], [67, [37, 41]], [67, [37, 63]], [71, [5, 40]], [73, [8, 14]], [73, [8, 15]],
[73, [27, 47]], [73, [27, 60]], [73, [27, 69]], [73, [46, 28]], [73, [46, 59]], [73, [46, 69]],
[73, [64, 18]], [73, [64, 36]], [73, [64, 39]], [73, [64, 41]], [73, [64, 47]], [73, [64, 57]],
[73, [64, 69]], [73, [64, 71]], [79, [23, 57]], [79, [23, 58]], [79, [55, 30]], [79, [55, 57]],
[79, [55, 70]], [89, [34, 22]], [89, [34, 67]], [89, [55, 36]], [89, [55, 53]], [97, [22, 5]],
[97, [22, 12]], [97, [22, 15]], [97, [22, 28]], [97, [22, 45]], [97, [35, 10]], [97, [35, 17]],
[97, [35, 25]], [97, [35, 44]], [97, [35, 52]], [97, [35, 65]], [97, [35, 67]], [97, [35, 71]],
[97, [36, 76]], [97, [61, 2]], [97, [61, 13]], [97, [61, 15]], [97, [61, 17]], [97, [61, 40]],
[97, [61, 42]], [97, [61, 47]], [97, [61, 59]], [97, [61, 60]], [97, [61, 67]], [97, [61, 74]],
[97, [61, 85]], [97, [75, 13]], [97, [75, 52]], [97, [75, 69]]]
```

```
Finding gadgets for the remaining values.
[31, [5, 13]] gadget found with x=[1, 1, 0, 0, 0]
[31, [5, 17]] gadget found with x=[0, 1, 4, 0, 0]
[31, [5, 23]] gadget found with x=[0, 1, 0, 0, 0]
[31, [25, 3]] gadget found with x=[0, 7, 0, 0, 0]
[31, [25, 23]] gadget found with x=[0, 9, 0, 0, 0]
[43, [6, 2]] gadget found with x=[1, 2, 0, 0, 1]
[43, [36, 4]] gadget found with x=[1, 4, 0, 0, 0]
[43, [36, 9]] gadget found with x=[1, 10, 0, 0, 0]
[43, [36, 11]] gadget found with x=[0, 2, 0, 0, 0]
[43, [36, 12]] gadget found with x=[2, 3, 0, 0, 0]
[43, [36, 15]] gadget found with x=[1, 4, 0, 0, 0]
[43, [36, 23]] gadget found with x=[0, 2, 0, 0, 0]
[43, [36, 24]] gadget found with x=[1, 2, 0, 0, 0]
[61, [13, 31]] gadget found with x=[2, 4, 1, 0, 0]
[61, [13, 37]] gadget found with x=[0, 1, 3, 0, 1]
[61, [13, 50]] gadget found with x=[0, 5, 0, 0, 0]
[61, [13, 51]] gadget found with x=[1, 2, 3, 0, 0]
[61, [47, 40]] gadget found with x=[1, 3, 0, 1, 0]
[61, [47, 51]] gadget found with x=[0, 4, 1, 0, 0]
[67, [29, 2]] gadget found with x=[2, 21, 0, 0, 0]
[67, [29, 7]] gadget found with x=[2, 13, 0, 0, 0]
[67, [29, 34]] gadget found with x=[1, 9, 0, 0, 0]
[67, [29, 39]] gadget found with x=[2, 9, 0, 0, 0]
[67, [29, 43]] gadget found with x=[0, 0, 9, 0, 0]
[67, [29, 48]] gadget found with x=[1, 3, 1, 0, 0]
[67, [29, 52]] gadget found with x=[0, 5, 0, 0, 0]
[67, [29, 53]] gadget found with x=[0, 1, 4, 0, 0]
[67, [29, 58]] gadget found with x=[1, 3, 1, 0, 0]
[67, [29, 59]] gadget found with x=[1, 8, 0, 0, 0]
[67, [37, 36]] gadget found with x=[2, 2, 0, 0, 0]
[67, [37, 41]] gadget found with x=[0, 1, 1, 0, 0]
[67, [37, 63]] gadget found with x=[0, 1, 1, 0, 0]
[71, [5, 40]] gadget found with x=[3, 0, 3, 0, 0]
[73, [8, 14]] gadget found with x=[0, 5, 3, 0, 0]
[73, [8, 15]] gadget found with x=[0, 1, 1, 2, 0]
[73, [27, 47]] gadget found with x=[2, 1, 2, 0, 0]
[73, [27, 60]] gadget found with x=[0, 5, 1, 0, 0]
[73, [27, 69]] gadget found with x=[2, 5, 0, 0, 0]
[73, [46, 28]] gadget found with x=[2, 11, 0, 0, 0]
[73, [46, 59]] gadget found with x=[0, 5, 0, 0, 0]
[73, [46, 69]] gadget found with x=[0, 7, 0, 0, 0]
[73, [64, 18]] gadget found with x=[1, 0, 0, 2, 0]
[73, [64, 36]] gadget found with x=[2, 4, 0, 0, 0]
[73, [64, 39]] gadget found with x=[0, 5, 1, 0, 0]
[73, [64, 41]] gadget found with x=[1, 20, 0, 0, 0]
[73, [64, 47]] gadget found with x=[0, 6, 1, 0, 0]
[73, [64, 57]] gadget found with x=[1, 10, 0, 0, 0]
[73, [64, 69]] gadget found with x=[1, 4, 2, 0, 0]
[73, [64, 71]] gadget found with x=[2, 20, 0, 0, 0]
[79, [23, 57]] gadget found with x=[0, 1, 6, 0, 0]
[79, [23, 58]] gadget found with x=[0, 9, 0, 1, 0]
[79, [55, 30]] gadget found with x=[0, 11, 0, 0, 0]
[79, [55, 57]] gadget found with x=[2, 10, 1, 0, 0]
[79, [55, 70]] gadget found with x=[2, 10, 0, 1, 0]
[89, [34, 22]] gadget found with x=[2, 1, 0, 0, 0]
[89, [34, 67]] gadget found with x=[3, 10, 0, 0, 0]
[89, [55, 36]] gadget found with x=[1, 8, 0, 0, 0]
[89, [55, 53]] gadget found with x=[1, 20, 0, 0, 0]
[97, [22, 5]] gadget found with x=[2, 0, 5, 0, 0]
[97, [22, 12]] gadget found with x=[0, 9, 0, 0, 0]
[97, [22, 15]] gadget found with x=[1, 2, 7, 0, 0]
[97, [22, 28]] gadget found with x=[0, 15, 0, 0, 0]
[97, [22, 45]] gadget found with x=[3, 3, 0, 0, 0]
[97, [35, 10]] gadget found with x=[1, 3, 0, 0, 0]
[97, [35, 17]] gadget found with x=[1, 13, 0, 0, 0]
[97, [35, 25]] gadget found with x=[0, 10, 0, 0, 0]
[97, [35, 44]] gadget found with x=[2, 3, 0, 0, 0]
[97, [35, 52]] gadget found with x=[0, 2, 2, 1, 0]
[97, [35, 65]] gadget found with x=[2, 22, 0, 0, 0]
[97, [35, 67]] gadget found with x=[1, 10, 1, 0, 0]
[97, [35, 71]] gadget found with x=[2, 0, 1, 0, 0]
[97, [36, 76]] gadget found with x=[4, 11, 0, 0, 0]
[97, [61, 2]] gadget found with x=[1, 6, 0, 0, 0]
[97, [61, 13]] gadget found with x=[1, 15, 0, 0, 0]
[97, [61, 15]] gadget found with x=[2, 31, 0, 0, 0]
[97, [61, 17]] gadget found with x=[2, 9, 0, 0, 0]
[97, [61, 40]] gadget found with x=[0, 6, 0, 1, 0]
[97, [61, 42]] gadget found with x=[0, 11, 0, 0, 0]
[97, [61, 47]] gadget found with x=[1, 20, 0, 0, 0]
[97, [61, 59]] gadget found with x=[1, 11, 0, 0, 0]
[97, [61, 60]] gadget found with x=[0, 13, 0, 0, 0]
```

```
[97, [61, 67]] gadget found with x=[1, 1, 1, 0, 0]
[97, [61, 74]] gadget found with x=[2, 15, 0, 0, 0]
[97, [61, 85]] gadget found with x=[1, 0, 1, 0, 3]
[97, [75, 13]] gadget found with x=[3, 1, 0, 0, 0]
[97, [75, 52]] gadget found with x=[1, 21, 0, 0, 0]
[97, [75, 69]] gadget found with x=[0, 21, 0, 0, 0]

Final unresolved values of [p,[g,l]] are:
[[41, [18, 6]]]
```

# Bibliography

[1] F. Arends, J. Ouaknine, and C. W. Wampler. On searching for small Kochen–Specker vector systems. In *Proceedings of the 37th International Workshop on Graph-Theoretic Concepts in Computer Science*, pages 23–34, 2011.

[2] M. A. Armstrong. *Groups and Symmetry.* Springer-Verlag, 1988.

[3] R. Beigel and J. Gill. Counting classes: Thresholds, parity, mods, and fewness. *Theoretical Computer Science*, 103(1):3–23, 1992.

[4] B. Ben-Moshe, B. K. Bhattacharya, Q. Shi, and A. Tamir. Efficient algorithms for center problems in cactus networks. *Theoretical Computer Science*, 378(3):237–252, 2007.

[5] B. Ben-Moshe, A. Dvir, M. Segal, and A. Tamir. Centdian computation in cactus graphs. *Journal of Graph Algorithms and Applications*, 16(2):199–224, 2012.

[6] B. Bollobás and A. Thomason. The structure of hereditary properties and colourings of random graphs. *Combinatorica*, 20(2):173–202, 2000.

[7] A. Brandstädt. Partitions of graphs into one or two independent stable sets and cliques. *Discrete Mathematics*, 152(2):47–54, 1996.

[8] G. R. Brightwell and P. Winkler. Graph homomorphisms and phase transitions. *Journal of Combinatorial Theory, Series B*, 77(2):221 – 262, 1999.

[9] M. Broom and J. Rychtár. An analysis of the fixation probability of a mutant on special classes of non-directed graphs. *ProcRoyal Society A*, 464(2098):2609–2627, 2008.

[10] A. Bulatov. The complexity of the counting constraint satisfaction problem. In *Proceedings of the 35th International Colloquium on Automata, Languages and Programming*, pages 646–661, 2008.

[11] A. Bulatov and V. Dalmau. Towards a dichotomy theorem for the counting constraint satisfaction problem. *Information and Computation*, 205(5):651–678, 2007.

[12] A. A. Bulatov, M. E. Dyer, L. A. Goldberg, M. Jalsenius, and D. Richerby. The complexity of weighted Boolean #CSP with mixed signs. *Theoretical Computer Science*, 410(38-40):3949–3961, 2009.

[13] A. A. Bulatov and M. Grohe. The complexity of partition functions. *Theoretical Computer Science*, 348(2–3):148–186, 2005.

[14] J.-Y. Cai and X. Chen. Complexity of counting CSP with complex weights. In *Proceedings of the 44th Symposium on Theory of Computing*, pages 909–920, 2012.

[15] J.-Y. Cai, X. Chen, and P. Lu. Non-negatively weighted #CSP: An effective complexity dichotomy. In *Proceedings of the IEEE Conference on Computational Complexity*, pages 45–54, 2011.

[16] J.-Y. Cai, X. Chen, and P. Lu. Graph homomorphisms with complex values: A dichotomy theorem. *SIAM Journal on Computing*, 42(3):924–1029, 2013.

[17] J.-Y. Cai and L. A. Hemachandra. On the power of parity polynomial time. *Mathematical Systems Theory*, 23(2):95–106, 1990.

[18] J.-Y. Cai and P. Lu. Holographic algorithms: From art to science. *Journal of Computer and System Sciences*, 77(1):41–61, 2011.

[19] M. Chudnovsky, N. Robertson, P. Seymour, and R. Thomas. The strong perfect graph theorem. *Annals of Mathematics. Second Series*, 164(1):51–229, 2006.

[20] V. Chvátal and N. Sbihi. Bull-free Berge graphs are perfect. *Graphs and Combinatorics*, 3(1):127–139, 1987.

[21] M. Conforti, G. Cornuéjols, and K. Vušković. Square-free perfect graphs. *Journal of Combinatorial Theory, Series B*, 90(2):257–307, 2004.

[22] J. Díaz, L. A. Goldberg, G. B. Mertzios, D. Richerby, M. J. Serna, and P. G. Spirakis. On the fixation probability of superstars. *Proceedings of the Royal Society A*, 469(2156):1–11, 2013.

[23] J. Díaz, L. A. Goldberg, G. B. Mertzios, D. Richerby, M. J. Serna, and P. G. Spirakis. Approximating fixation probabilities in the generalised Moran process. *Algorithmica*, 69(1):78–91, 2014.

[24] J. Díaz, L. A. Goldberg, D. Richerby, and M. J. Serna. Absorption time of the Moran process. *Random Structures and Algorithms*, 49(1):137–159, 2016.

[25] M. Dyer and D. Richerby. An effective dichotomy for the counting constraint satisfaction problem. *SIAM Journal on Computing*, 42(3):1245–1274, 2013.

[26] M. E. Dyer, L. A. Goldberg, and D. Richerby. Counting $4 \times 4$ matrix partitions of graphs. *Discrete Applied Mathematics*, 213:76–92, 2016.

[27] M. E. Dyer and C. S. Greenhill. The complexity of counting graph homomorphisms. *Random Structures and Algorithms*, 17(3-4):260–289, 2000.

[28] H. Everett, S. Klein, and B. Reed. An algorithm for finding homogeneous pairs. *Discrete Applied Mathematics*, 72(3):209–218, 1997.

[29] H. Everett, S. Klein, and B. Reed. An optimal algorithm for finding clique-cross partitions. In *Proceedings of the 29th Southeastern International Conference on Combinatorics, Graph Theory and Computing*, pages 171–177, 1998.

[30] J. Faben. The complexity of counting solutions to generalised satisfiability problems modulo k. *CoRR*, abs/0809.1836, 2008.

[31] J. Faben. *The Complexity of Modular Counting in Constraint Satisfaction Problems.* PhD thesis, Queen Mary, University of London, 2012.

[32] J. Faben and M. Jerrum. The complexity of parity graph homomorphism: an initial investigation. *Theory of Computing*, 11:35–57, 2015.

[33] T. Feder and P. Hell. List homomorphisms to reflexive graphs. *Journal of Combinatorial Theory, Series B*, 72(2):236–250, 1998.

[34] T. Feder and P. Hell. Full constraint satisfaction problems. *SIAM Journal on Computing*, 36(1):230–246, 2006.

[35] T. Feder, P. Hell, and J. Huang. List homomorphisms and circular arc graphs. *Combinatorica*, 19(4):487–505, 1999.

[36] T. Feder, P. Hell, and J. Huang. Bi-arc graphs and the complexity of list homomorphisms. *Journal of Graph Theory*, 42(1):61–80, 2003.

[37] T. Feder, P. Hell, S. Klein, and R. Motwani. List partitions. *SIAM Journal on Discrete Mathematics*, 16(3):449–478, 2003.

[38] T. Feder and M. Vardi. The computational structure of monotone monadic SNP and constraint satisfaction: a study through Datalog and group theory. *SIAM Journal on Computing*, 28(1):57–104, 1999.

[39] W. Feller. *An Introduction to Probability Theory and its Applications*, volume I. Wiley, 3rd edition, 1968.

[40] G. Fontaine. Why is it hard to obtain a dichotomy for consistent query answering? *ACM Transactions on Computational Logic*, 16(1):7:1–7:24, 2015.

[41] A. Galanis, A. Göbel, L. A. Goldberg, J. Lapinskas, and D. Richerby. Amplifiers for the Moran process. *Journal of the ACM*, to appear. `http://arxiv.org/abs/1512.05632`.

[42] A. Göbel, L. A. Goldberg, C. McQuillan, D. Richerby, and T. Yamakami. Counting list matrix partitions of graphs. *SIAM Journal on Computing*, 44(4):1089–1118, 2015.

[43] A. Göbel, L. A. Goldberg, and D. Richerby. The complexity of counting homomorphisms to cactus graphs modulo 2. *ACM Transactions on Computation Theory*, 6(4):article 17, 2014.

[44] A. Göbel, L. A. Goldberg, and D. Richerby. Counting homomorphisms to square-free graphs, modulo 2. *ACM Transactions on Computation Theory,*, 8(3):12, 2016.

[45] L. A. Goldberg, M. Grohe, M. Jerrum, and M. Thurley. A complexity dichotomy for partition functions with mixed signs. *SIAM Journal on Computing*, 39(7):3336–3402, 2010.

[46] L. A. Goldberg, R. Gysel, and J. Lapinskas. Approximately counting locally-optimal structures. *Journal of Computer and System Sciences*, 82(6):1144–1160, 2016.

[47] L. M. Goldschlager and I. Parberry. On the construction of parallel computers from various bases of Boolean functions. *Theoretical Computer Science*, 43:43–58, 1986.

[48] M. Golumbic. *Algorithmic Graph Theory and Perfect Graphs.* Elsevier Science, 2nd edition, 2004.

[49] H. Guo, S. Huang, P. Lu, and M. Xia. The complexity of weighted Boolean #CSP modulo k. In *Proceedings of the 28th International Symposium on Theoretical Aspects of Computer Science*, pages 249–260, 2011.

[50] F. Harary and G. E. Uhlenbeck. On the number of Husimi trees. I. *Proceedings of the National Academy of Sciences of the United States of America*, 39:315–322, 1953.

[51] P. Hell, M. Hermann, and M. Nevisi. Counting partitions of graphs. In *Proceedings of the 23rd International Symposium on Algorithms and Computation*, pages 227–236, 2012.

[52] P. Hell and J. Nešetřil. On the complexity of $H$-coloring. *Journal of Combinatorial Theory, Series B*, 48(1):92–110, 1990.

[53] P. Hell and J. Nešetřil. *Graphs and Homomorphisms*. Oxford University Press, 2004.

[54] U. Hertrampf. Relations among mod-classes. *Theoretical Computer Science*, 74(3):325–328, 1990.

[55] A. Jamieson-Lane and C. Hauert. Fixation probabilities on superstars, revisited and revised. *Journal of Theoretical Biology*, 382:44–56, 2015.

[56] S. Janson, T. Łuczak, and A. Rucinski. *Random Graphs*. Wiley, 2000.

[57] R. E. Ladner. On the structure of polynomial time reducibility. *Journal of the ACM*, 22(1):155–171, 1975.

[58] C. Lecoutre. *Constraint Networks: Techniques and Algorithms*. Wiley–IEEE Press, 2009.

[59] E. Lieberman, C. Hauert, and M. A. Nowak. Evolutionary dynamics on graphs. *Nature*, 433(7023):312–316, 2005. Supplementary material available at `http://www.nature.com/nature/journal/v433/n7023/full/nature03204.html`.

[60] L. Lovász. Operations with structures. *Acta Mathematica Academiae Scientiarum Hungaricae*, 18(3–4):321–328, 1967.

[61] L. Lovász. On the cancellation law among finite relational structures. *Periodica Mathematica Hungarica*, 1(2):145–156, 1971.

[62] L. Lovász. Normal hypergraphs and the perfect graph conjecture. *Discrete Mathematics*, 2(3):253–267, 1972.

[63] M. Mitzenmacher and E. Upfal. *Probability and Computing: Randomized Algorithms and Probabilistic Analysis*. Cambridge University Press, 2005.

[64] P. A. P. Moran. Random processes in genetics. *Proceedings of the Cambridge Philosophical Society*, 54(1):60–71, 1958.

[65] C. H. Papadimitriou and S. Zachos. Two remarks on the power of counting. In *Proceedings of the 6th GI-Conference on Theoretical Computer Science*, pages 269–276, 1982.

[66] B. Paten, M. Diekhans, D. Earl, J. St. John, J. Ma, B. B. Suh, and D. Haussler. Cactus graphs for genome comparisons. *Journal of Computational Biology*, 18(3):469–481, 2011.

[67] S. J. Russell and P. Norvig. *Artificial Intelligence - A Modern Approach (3. internat. ed.)*. Pearson Education, 2010.

[68] T. J. Schaefer. The complexity of satisfiability problems. In *Proceedings of the 11h Annual ACM Symposium on Theory of Computing*, pages 216–226, 1978.

[69] V. Shoup. *A Computational Introduction to Number Theory and Algebra.* Cambridge University Press, 2006.

[70] S. Toda. PP is as hard as the polynomial-time hierarchy. *SIAM Journal on Computing*, 20(5):865–877, 1991.

[71] L. G. Valiant. The complexity of computing the permanent. *Theoretical Computer Science*, 8(2):189–201, 1979.

[72] L. G. Valiant. Accidental algorthims. In *Proceedings of the 47th Annual IEEE Symposium on Foundations of Computer Science*, pages 509–517, 2006.

[73] M. Xia, P. Zhang, and W. Zhao. Computational complexity of counting problems on 3-regular planar graphs. *Theoretical Computer Science*, 384(1):111–125, 2007.