Weighted Counting of Matchings in Unbounded-Treewidth Graph Families

Antoine Amarilli
LTCI, Télécom Paris, Institut Polytechnique de Paris, France

Mikaël Monet
Univ. Lille, Inria, CNRS, Centrale Lille, UMR 9189 CRIStAL, F-59000 Lille, France

Abstract

We consider a weighted counting problem on matchings, denoted PrMatching($\mathcal{G}$), on an arbitrary fixed graph family $\mathcal{G}$. The input consists of a graph $G \in \mathcal{G}$ and of rational probabilities of existence on every edge of G, assuming independence. The output is the probability of obtaining a *matching* of G in the resulting distribution, i.e., a set of edges that are pairwise disjoint. It is known that, if $\mathcal{G}$ has bounded *treewidth*, then PrMatching($\mathcal{G}$) can be solved in polynomial time. In this paper we show that, under some assumptions, bounded treewidth in fact *characterizes* the tractable graph families for this problem. More precisely, we show intractability for all graph families $\mathcal{G}$ satisfying the following *treewidth-constructibility* requirement: given an integer k in unary, we can construct in polynomial time a graph $G \in \mathcal{G}$ with treewidth at least k. Our hardness result is then the following: for *any* treewidth-constructible graph family $\mathcal{G}$, the problem PrMatching($\mathcal{G}$) is intractable. This generalizes known hardness results for weighted matching counting under some restrictions that do not bound treewidth, e.g., being planar, 3-regular, or bipartite; it also answers a question left open in [1]. We also obtain a similar lower bound for the weighted counting of edge covers.

2012 ACM Subject Classification Mathematics of computing → Matchings and factors

Keywords and phrases Treewidth, counting complexity, matchings, Fibonacci sequence

Digital Object Identifier 10.4230/LIPIcs.MFCS.2022.9

Related Version *Full Version*: `https://arxiv.org/abs/2205.00851` [4]

Supplementary Material *Software (Computer calculations)*:
`https://gitlab.com/Gruyere/supplementary-material-for-Weighted-Counting-of-Matchings`, archived at `swh:1:dir:b01ffb42985d203c5cc1510dad134c9696e2d1f4`

Funding *Antoine Amarilli*: Partially supported by the ANR project EQUUS ANR-19-CE48-0019 and by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) – 431183758.

Acknowledgements We thank Florent Capelli and Sébastien Tavenas for discussions on the problem, in particular on how to prove Proposition 5.2.

1 Introduction

Many complexity results on computational problems rely on a study of fundamental graph patterns such as independent sets, vertex covers, edge covers, matchings, cliques, etc. In this paper we specifically study *counting problems* for such patterns, and for the most part focus on counting the *matchings*: given an input graph G, we wish to count how many edge subsets of G are a matching, i.e., each vertex has at most one incident edge.

Our goal is to address an apparent gap between the existing intractability and tractability results for counting matchings and similar patterns. On the one hand, counting the matchings is known to be #P-hard, and hardness is known even when the input graph is restricted in certain ways, e.g., being planar, being 3-regular, or being bipartite [18, 14, 27, 26]. On

the other hand, some restrictions can make the problem tractable, e.g., imposing that the input graphs have *bounded treewidth* [5, 1], because matchings can be described in monadic second-order logic. But this does not settle the complexity of the problem; could there be other restrictions on graphs that makes it tractable to count matchings or other patterns?

This paper answers this question in the negative, for a *weighted* version of counting problems: we show that, at least for matchings and edge covers, and under a technical assumption on the graph family, the weighted counting problem is intractable if we do not bound the treewidth of the input graphs. Thus, treewidth is the right parameter to ensure tractability. Our weighted counting problems are of the following form: we fix a graph family $\mathcal{G}$ (e.g., 3-regular graphs, graphs of treewidth $\leqslant 2$), we are given as input a graph G of $\mathcal{G}$ along with an independent probability of existence for each edge, and the goal is to compute the probability in this distribution of the subsets of edges of G which have a certain property, e.g., they are a matching, they are an edge cover. Note that the class $\mathcal{G}$ restricts the shape of the graphs, but the edge probabilities are arbitrary – and indeed there are known tractability results when we restrict the graphs and probabilities to be symmetric [6]. Our paper shows the hardness of these problems when $\mathcal{G}$ is not of bounded treewidth; the specific technical assumption on $\mathcal{G}$ is that one can effectively construct graphs of $\mathcal{G}$ having arbitrarily high treewidth, i.e., the *treewidth-constructible* requirement from [1] (cf. Definition 2.2):

▶ **Result 1.** *Let $\mathcal{G}$ be an arbitrary family of graphs which is treewidth-constructible. Then the problem, given a graph $G = (V, E)$ of $\mathcal{G}$ and rational probability values $\pi(e)$ for every edge of G, of computing the probability of a matching in G under π, is #P-hard under ZPP reductions.*

We obtain an analogous result for edge covers. Thus, as bounded-treewidth makes the problems tractable, our results imply that treewidth characterizes the tractable graph families for these problems – for weighted counting, and assuming treewidth-constructibility. We leave open the complexity of unweighted counting, and of weighted counting on graph families that have unbounded treewidth but satisfy weaker requirements than treewidth-constructibility, e.g., being strongly unbounded poly-logarithmically [16, 13].

The paper is devoted to showing Result 1. Because of the page limit, the full proofs are deferred to the full version [4]. At a high level, we use the standard technique of reducing from the #P-hard problem of counting matchings on a 3-regular planar graph G [26], using the randomized polynomial-time grid minor extraction result of [10] as in [1]. However, the big technical challenge is to reduce the counting of matchings of G to the problem of computing the probability of a matching on the arbitrary subdivision G' of G that we extract. For this, we use the classical interpolation method, where we design a linear equation system relating the matchings to the result of polynomially many oracle calls on G', with different probability assignments; and we argue that the matrix is invertible. After the preliminaries (Section 2), we present this proof, first in the case where G' is a 6-subdivision of G (Section 3), and then when it is a n-subdivision, i.e., when all edges are subdivided to the same length n (Section 4). These special cases already pose some difficulties, most of which are solved by adapting techniques by Dalvi and Suciu [12]; e.g., to show invertibility, we study the Jacobian determinant of the mapping associating edge probabilities to the probability of matchings on paths with fixed endpoints, and we borrow a technique from [12] to effectively construct suitable rational edge probabilities.

The main novelties of this work are in Section 5, where we extend the proof to the general case: G' is a subdivision of G, and different edges of G may be subdivided in G' to different lengths. To obtain the equation system, we show that we can assign probabilities on short paths so that they "behave" like long paths. Proving this stand-alone *emulation result*

(Proposition 5.2) was the main technical obstacle; the proof is by solving a system of equations involving the Fibonacci sequence. It also introduces further complications, e.g., dealing with numerical error (because the resulting probabilities are irrational), and distinguishing even-length and odd-length subdivisions. After concluding the proof of Result 1 in Section 5, we adapt it in Section 6 to edge covers.

Related work. Our work follows a line of results that show the intractability of some problems on any "sufficiently constructible" unbounded-treewidth graph family. Kreutzer and Tazari [16] (see also [13]) show that there are formulas in an expressive formalism (MSO_2) that are intractable to check on any subgraph-closed unbounded treewidth graph family that is closed under taking subgraphs and satisfies a requirement of being *strongly unbounded poly-logarithmically*. This was extended in [1] to the weighted counting problem, this time for a query in first-order logic, with a different hardness notion (#P-hardness under randomized reductions), and under the stronger requirement of treewidth-constructibility. Our focus here is to show that the hardness of weighted counting already holds for natural and well-studied graph properties, e.g., "being a matching"; this was left as an open problem in [1].

For such weak patterns, lower bounds were shown in [1] and [3] on the *size of tractable representations*: for any graph G of bounded degree having treewidth k, any so-called *d-SDNNF* circuit representing the matchings (or edge covers) of G must have exponential size in k. However, this does not imply that the problems are intractable, as some tractable counting algorithms do not work via such circuit representations (e.g., the one in [12]). Thus, our hardness result does not follow from this size bound, but rather complements it.

The necessity of bounded treewidth has also been studied for graphical models [9] and Bayesian networks [17]. Specifically, [17] shows the intractability of inference in a Bayesian network as a function of the treewidth (but without otherwise restricting the class of network), and [9] restricts the shape of the graphical model but allows arbitrary "potential functions" (whereas we assume independence across edges). There are also necessity results on treewidth for the problem of *counting the homomorphisms* between two structures in the CSP context [11]; but this has no clear relationship to our problems, where we do (weighted) *counting of the substructures* that have a certain form (e.g., are matchings).

Note that, unlike our problem of weighted counting of matchings, the problem of *finding* a matching of maximal weight in a weighted graph is tractable on arbitrary graphs, using Edmond's blossom algorithm [21].

2 Preliminaries

We write $\mathbb{N}^+$ for $\mathbb{N}\backslash\{0\}$, and for $n \in \mathbb{N}^+$ we write $[n]$ the set $\{0, \ldots, n-1\}$. We write $\mathbb{R}$ the real numbers and $\mathbb{Q}$ the rational numbers. Recall that *decimal fractions* are rational numbers that can be written as a fraction $a/10^k$ of an integer a and a power of ten 10^k.

Reductions and complexity classes. Recall that #P is the class of counting problems that count the number of accepting paths of a nondeterministic polynomial-time Turing machine. A problem P_1 is #P-*hard* if every problem P_2 of #P reduces to P_1 in polynomial time; following Valiant [19, 20], we use here the notion of *Turing reductions*, i.e., P_2 can be solved in polynomial time with an oracle for P_1. We specifically study what we call #P-*hardness under zero-error probabilistic polynomial-time (ZPP) reductions*. To define these, we define a *randomized algorithm* as an algorithm that has access to an additional random tape. We say that a decision problem is in *ZPP* if there is a randomized algorithm

that (always) runs in polynomial time on the input instance, and returns the correct answer on the instance (i.e., accepting or rejecting) with some constant probability, and otherwise returns a special failure value. The probabilities are taken over the draws of the contents of the random tape. The exact value of the acceptance probability is not important, because we can make it exponentially small by simply repeating the algorithm polynomially many times. Going beyond decision problems, a *ZPP algorithm* is a randomized algorithm that runs in polynomial time but may return a special failure value with some constant probability. A *ZPP (Turing) reduction* from a problem P_1 to a problem P_2 is then a ZPP algorithm having access to an oracle for P_2 that takes an instance of problem P_1, runs in polynomial time, returns the correct output (for P_1) with some constant probability, and returns the special failure value otherwise. Again, the failure probability can be made arbitrarily small by invoking the reduction multiple times. A problem P_2 is then said to be *#P-hard under ZPP reductions* if any #P-hard problem P_1 has a ZPP reduction to it. We will implicitly rely on the fact that we can show #P-hardness under ZPP reductions by reducing in ZPP from any problem which is #P-hard (under Turing reductions); see the full version [4] for details.

Graphs and problem studied. A finite undirected *graph* $G = (V, E)$ consists of a finite set V of *vertices* (or *nodes*) and of a set E of *edges* of the form $\{x, y\}$ for $x, y \in V$ with $x \neq y$. A *graph family* $\mathcal{F}$ is a (possibly infinite) set of graphs. For $v \in V$, we write $\mathcal{E}_G(v)$ for the set of edges that are incident to v. Recall that a *matching* of G is a set of edges $M \subseteq E$ that do not share any vertices, i.e., for every $e, e' \in M$ with $e \neq e'$ we have $e \cap e' = \varnothing$; or equivalently, we have $|\{\mathcal{E}_G(v) \cap M\}| \leqslant 1$ for all $v \in V$. For a graph family $\mathcal{F}$, we write #Matching($\mathcal{F}$) the problem of counting the matchings for graphs in $\mathcal{F}$: the input is a graph $G \in \mathcal{F}$, and the output is the number of matchings of G, written #Matching(G).

We study a weighted version of #Matching, defined on *probabilistic graphs*. A *probabilistic graph* is a pair (G, π) where $G = (V, E)$ is a graph and $\pi \colon E \to [0, 1]$ maps every edge e of H to a probability value $\pi(e)$. The probabilistic graph (G, π) defines a probability distribution on the set of subsets E' of E, where each edge $e \in E$ is in E' with probability $\pi(e)$, assuming independence across edges. Formally, the probability of each subset E' is:

$$\Pr_{G,\pi}(E') := \prod_{e \in E'} \pi(e) \times \prod_{e \in E \setminus E'} (1 - \pi(e)).$$

Given a probabilistic graph (G, π), the *probability of a matching in G under pi*, denoted $\Pr_{\text{matching}}(G, \pi)$, is the probability of obtaining a matching in the distribution. Formally:

$$\Pr_{\text{matching}}(G, \pi) := \sum_{\text{matching } M \text{ of } G} \Pr_{G,\pi}(M). \tag{1}$$

In particular, if π maps every edge to the probability 1/2, then we have $\Pr_{\text{matching}}(G, \pi) = \#\text{Matching}(G)/2^{|E|}$. For a graph family $\mathcal{F}$, we will study the problem PrMatching($\mathcal{F}$) of computing the probability of a matching: the input is a probabilistic graph (G, π) where $G \in \mathcal{F}$ and π is an arbitrary function with rational probability values, and the output is $\Pr_{\text{matching}}(G, \pi)$. Note that $\mathcal{F}$ only specifies the graph G and not the probabilities π, in particular π can give probability 0 to edges, which amounts to removing them.

Treewidth and topological minors. Treewidth is a parameter mapping any graph G to a number $\text{tw}(G)$ intuitively describing how far G is from being a tree. We omit the formal definition of treewidth (see [25]), as we only rely on the following *extraction result*: given any *planar graph H of maximum degree* 3, and a graph G of *sufficiently high* treewidth, it is

possible (in randomized polynomial time) to find H as a *topological minor* of G. We now define this.

The *degree* of a node v in $H = (V_H, E_H)$ is simply $|\mathcal{E}_G(v)|$. We say that H is 3-*regular* if every vertex has degree 3, and call H *planar* if it can be drawn on the plane without edge crossings, in the usual sense [24]. Given H and $\eta\colon E_H \to \mathbb{N}^+$, the *$\eta$-subdivision of H*, written $\mathrm{Sub}(H, \eta)$, is the graph obtained from H by replacing every edge $e = \{x, y\}$ by a path of length $\eta(e)$, whose end vertices are identified with x and y, all intermediate vertices being fresh across all edges. We abuse notation and write $\mathrm{Sub}(H, i)$ for $i \in \mathbb{N}_+$ to mean $\mathrm{Sub}(H, \eta_i)$ for η_i the constant-i function. Note that $\mathrm{Sub}(H, 1) = H$. A *subgraph* of a graph $G = (V_G, E_G)$ is a graph (V'_G, E'_G) where $E'_G \subseteq E_G$ and $V'_G \subseteq V_G$ such that $e \subseteq V'_G$ for each edge $e \in E'_G$. The graph $H = (V_H, E_H)$ is a *topological minor* of the graph $G = (V_G, E_G)$ if there is a function $\eta\colon E_H \to \mathbb{N}^+$ such that there is an isomorphism f from the subdivision $\mathrm{Sub}(H, \eta) = (V'_H, E'_H)$ to some subgraph $G' = (V'_G, E'_G)$ of G, i.e., a bijection $f\colon V'_H \to V'_G$ such that for every $x, y \in V'_H$ we have $\{x, y\} \in E'_H$ if and only if $\{f(x), f(y)\} \in E'_G$.

We can now state the extraction result that we use, which follows from the work of Chekuri and Chuzhoy [10]:

▶ **Theorem 2.1** (Direct consequence of [10], see, e.g., [1], Lemma 4.4)**.** *There exists $c \in \mathbb{N}$ and a ZPP algorithm*[1] *that, given as input a planar graph $H = (V_H, E_H)$ of maximum degree 3 and another graph G with $\mathrm{tw}(G) \geqslant |V_H|^c$, computes a subgraph G' of G, a function $\eta : V_H \to \mathbb{N}^+$, and an isomorphism from $\mathrm{Sub}(H, \eta)$ to G' (witnessing that H is a topological minor of G).*

Our intractability result will apply to graph families where large treewidth graphs can be efficiently found, which we formalize as *treewidth-constructibility* like in [1]:

▶ **Definition 2.2.** *A graph family $\mathcal{F}$ is* treewidth-constructible *if there is a polynomial-time algorithm that, given an integer k written in unary*[2]*, outputs a graph $G \in \mathcal{F}$ with $\mathrm{tw}(G) \geqslant k$.*

Kronecker products and Vandermonde matrices. To simplify notation, we will work with matrices indexed with arbitrary finite sets (not necessarily ordered). Given two finite sets I, J of same cardinality, we write $\mathbb{R}^{I,J}$ (resp., $\mathbb{Q}^{I,J}$) the set of matrices with real values (resp., rational values) whose rows are indexed by I and columns by J. When $\boldsymbol{A} \in \mathbb{R}^{I,J}$ and $(i, j) \in I \times J$, we write $a_{i,j}$ the corresponding entry. We recall that the inverse of an invertible matrix M with entries in $\mathbb{Q}$ also has entries in $\mathbb{Q}$ and can be computed in polynomial time in the encoding size of M.

Given two matrices $\boldsymbol{A} \in \mathbb{R}^{I,J}$ and $\boldsymbol{B} \in \mathbb{R}^{K,L}$, the *Kronecker product of $\boldsymbol{A}$ and $\boldsymbol{B}$*, denoted $\boldsymbol{A} \otimes \boldsymbol{B}$, is the matrix $\boldsymbol{C} \in \mathbb{R}^{I\times K,\, J\times L}$ defined by $c_{(i,k),(j,l)} := a_{i,j} \times b_{k,l}$ for $(i, j, k, l) \in I \times J \times K \times L$. Recall that $\boldsymbol{A} \otimes \boldsymbol{B}$ is invertible if and only if both $\boldsymbol{A}$ and $\boldsymbol{B}$ are. For $n \in \mathbb{N}^+$ and $(p_0, \ldots, p_{n-1}) \in \mathbb{R}^n$, we denote by $\mathcal{V}(p_0, \ldots, p_{n-1})$ the *Vandermonde matrix with coefficients* $(p_0, \ldots, p_{n-1})$, i.e., the matrix in $\mathbb{R}^{[n],[n]}$ whose (i, j)-th entry is p_i^j. Recall that this matrix is invertible if and only if the $p_0, \ldots, p_{n-1}$ are pairwise distinct.

[1] The randomized algorithm from [10] is indeed a ZPP algorithm because the output that it returns (namely, a prospective embedding of a grid as a topological minor of the input graph) can be verified in (deterministic) polynomial time. Hence, we can always detect when the algorithm has failed, and then return the special failure value.

[2] Note that the existence of such an algorithm for k written in unary would be implied by the same claim but with k given in binary. In other words, the existence of an algorithm for k given in unary is a weaker requirement. This is simply because, given an integer in unary, we can convert it in PTIME to an integer in binary.

3 Proof When Every Subdivision Has Length 6

Towards showing our main result (Result 1), we first show in this section a much simpler result: counting the matchings of a graph G reduces to counting the probability of a matching on the graph where each edge is subdivided into a path of length 6. We use similar techniques to previous work, in particular Greenhill [14] and Dalvi and Suciu [12], but present them in detail because we will adapt them in the rest of the paper. Formally, in this section, we show:

▶ **Proposition 3.1.** *For any graph family $\mathcal{F}$, the problem #Matching($\mathcal{F}$) reduces in polynomial time to PrMatching($\mathcal{G}$) where $\mathcal{G} = \{Sub(H, 6) \mid H \in \mathcal{F}\}$.*

Let $H = (V, E)$ be a graph in $\mathcal{F}$ for which we wish to count the number of matchings, with $m := |E|$. Let us start by fixing for the remainder of this section an arbitrary orientation $\overrightarrow{H}$ of H obtained by choosing some orientation of the edges, i.e., $\overrightarrow{H} = (V, \overrightarrow{E})$ is a *directed* graph where for every edge $\{x, y\} \in E$ we add exactly one of (x, y) or (y, x) in $\overrightarrow{E}$. The high-level idea of the reduction is then the following. First, using $\overrightarrow{H}$, we define some sets S_τ, based on 4-tuples $\tau \in [m+1]^4$, such that the number of matchings of H can be computed from the cardinalities $|S_\tau|$. Second, we argue that these cardinalities can be connected to the results of oracle calls for the PrMatching problem by a system of linear equations. Third, we argue that the matrix of this system can be made invertible. We now detail these three steps.

Step 1: Defining the sets S_τ and linking them to matchings. We define a *selection function* of the graph H as a function μ that maps each vertex $x \in V$ to at most one incident edge, i.e., to a subset of $\mathcal{E}_H(x)$ of size at most one. We will partition the set of selection functions by counting the number of edges of each *type* that each selection function has, as defined next. Given a selection function μ, consider each edge $e = (x, y)$ of $\overrightarrow{H}$. The edge e can have one of four types: letting b be 1 if $\mu(x)$ selects e (i.e., $\mu(x) = \{\{x, y\}\}$) and 0 otherwise (i.e., $\{x, y\} \notin \mu(x)$), and letting b' be 1 if $\mu(y)$ selects e and 0 otherwise, we say that e *has type bb' with respect to (w.r.t.) μ*. We now define the sets S_τ as follows.

▶ **Definition 3.2.** *For a 4-tuple $\tau \in [m+1]^4$, indexed in binary, let $S_\tau \subseteq S$ be the set of the selection functions μ such that, for all $b, b' \in \{0, 1\}$, precisely $\tau_{bb'}$ edges have type bb' w.r.t. μ.*

Observe that S_τ is empty unless $\tau_{00} + \tau_{01} + \tau_{10} + \tau_{11} = m$. We can then easily connect the cardinalities $|S_\tau|$ to the number of matchings of H as follows (see the full version [4]):

▶ **Fact 3.3.** *We have that #Matching(H) $= \sum_{\substack{\tau \in [m+1]^4 \\ \tau_{01} = \tau_{10} = 0}} |S_\tau|$.*

Step 2: Recovering the $|S_\tau|$ from oracle calls. We now explain how to use the oracle for PrMatching($\mathcal{G}$) to compute in polynomial time all the values $|S_\tau|$, allowing us to conclude via Fact 3.3. We will invoke the oracle on $(m+1)^4$ probabilistic graphs, denoted $H_6(\kappa)$ for $\kappa \in [m+1]^4$, as defined next. To this end, let us consider $(m+1)^4$ 4-tuples of probability values, written $\rho_\kappa = (\rho_{\kappa,00}, \rho_{\kappa,01}, \rho_{\kappa,10}, \rho_{\kappa,11}) \in [0, 1]^4$ for $\kappa \in [m+1]^4$; the precise choice of these values will be explained in Step 3. For $\kappa \in [m+1]^4$, we then define $H_6(\kappa)$ to be the probabilistic graph (H_6, π_κ) where $H_6 := \mathrm{Sub}(H, 6)$ is the 6-subdivision of H and the probabilities π_κ are defined as follows. For every directed edge (x, y) of $\overrightarrow{H}$, the subdivision H_6 contains an (undirected) path between x and y, and we define π_κ on this path as follows:

$$x \xrightarrow{1/2} v_1 \xrightarrow{\rho_{\kappa,00}} v_2 \xrightarrow{\rho_{\kappa,01}} v_3 \xrightarrow{\rho_{\kappa,10}} v_4 \xrightarrow{\rho_{\kappa,11}} v_5 \xrightarrow{1/2} y$$

We now introduce some notation for the probability of matchings in paths of length 4. We write $\Pi_4(\rho_\kappa)$ the probability of having a matching in the 4-edge path with successive probabilities $\rho_{\kappa,00}, \rho_{\kappa,01}, \rho_{\kappa,10}, \rho_{\kappa,11}$. The value can be explicitly computed as a polynomial in the values $\rho_{\kappa,bb'}$, e.g., using Equation (1). Accordingly, we will also use Π_4 as a polynomial with real variables, i.e., $\Pi_4(\chi)$ for a 4-tuple χ of real values (which may not be in $[0,1]$). We also define variants of these definitions that account for the two surrounding edges, i.e., those with probability 1/2: for $b, b' \in \{0,1\}$, write $\Pi_4^{bb'}(\rho_\kappa)$ to denote the probability of having a matching in the same 4-edge path but when adding an edge incident to the first vertex with probability 1 if $b = 1$, and adding an edge incident to the last vertex with probability 1 if $b' = 1$. Equivalently, $\Pi_4^{bb'}(\rho_\kappa)$ is the probability of obtaining a matching where we further require if $b = 1$ that the edge with probability $\rho_{\kappa,00}$ is *not* taken, and if $b' = 1$ that the edge with probability $\rho_{\kappa,11}$ is *not* taken. The values $\Pi_4^{bb'}(\rho_\kappa)$ are also explicitly computable as polynomials, and again we also see $\Pi_4^{bb'}$ as a polynomial with real variables. To simplify notation, for $b, b' \in \{0,1\}$ and $\kappa \in [m+1]^4$ let us write $\Lambda_{\kappa,bb'} := \Pi_4^{bb'}(\rho_\kappa)$.

We then show that the probability of a matching in the subdivided graph H_6 can be obtained by first summing over the possible edge type cardinalities τ, and then regrouping the edges of the same type by noticing that the matchings corresponding to the selection functions in the set S_τ all have the same probability. Namely, we show (see the full version [4]):

▶ **Fact 3.4.** *For each $\kappa \in [m+1]^4$, we have:*

$$2^{2m} \times \Pr_{matching}(H_6(\kappa)) \;=\; \sum_{\tau \in [m+1]^4} |S_\tau| \times (\Lambda_{\kappa,00})^{\tau_{00}} \times (\Lambda_{\kappa,01})^{\tau_{01}} \times (\Lambda_{\kappa,10})^{\tau_{10}} \times (\Lambda_{\kappa,11})^{\tau_{11}}.$$

Now, let us write $c_\kappa := \Pr_{\text{matching}}(H_6(\kappa))$ the value returned by the oracle call on $H_6(\kappa)$, and let $\boldsymbol{C}$ be the vector of these oracle answers. Let $\boldsymbol{S}$ be the vector $|S_\tau|$ of the values that we wish to compute. Both these vectors are indexed by $[m+1]^4$. Observe that the equation above defines a system of linear equations $\boldsymbol{V}\boldsymbol{S} = \boldsymbol{C}$ with $\boldsymbol{V} \in \mathbb{R}^{[m+1]^4,[m+1]^4}$ defined by

$$v_{\kappa,\tau} \;:=\; 2^{-2m} \times (\Lambda_{\kappa,00})^{\tau_{00}} \times (\Lambda_{\kappa,01})^{\tau_{01}} \times (\Lambda_{\kappa,10})^{\tau_{10}} \times (\Lambda_{\kappa,11})^{\tau_{11}}.$$

Therefore, if we can choose 4-tuples of probability values ρ_κ that make $\boldsymbol{V}$ invertible, we would be able to recover all $|S_\tau|$ values from the oracle answers $\boldsymbol{C}$, from which we could compute the number of matchings of H using Fact 3.3. This is what we do next.

Step 3: Making $\boldsymbol{V}$ invertible. We now explain how to choose in polynomial time $(m+1)^4$ 4-tuples ρ_κ of rational probability values, for $\kappa \in [m+1]^4$, such that $\boldsymbol{V}$ is invertible. To this end, consider the matrix $\boldsymbol{U}$ defined like $\boldsymbol{V}$ except that each 4-tuple ρ_κ is replaced by a 4-tuple of variables $\chi_\kappa = (\chi_{\kappa,00}, \chi_{\kappa,01}, \chi_{\kappa,10}, \chi_{\kappa,11})$. Each cell $m_{\kappa,\tau}$ of $\boldsymbol{U}$ is then a polynomial P_τ in the 4 variables $\chi_{\kappa,bb'}$ for $b, b' \in \{0,1\}$; in particular, note that the polynomial only depends on the column τ, whereas the variables $\chi_{\kappa,bb'}$ only depend on the row κ. We can then find suitable values ρ_κ using a technique introduced by Dalvi and Suciu [12] (see the full version [4]):

▶ **Proposition 3.5** (From Proposition 8.44 of [12]). *Fix $k \in \mathbb{N}$, let $(x_i)_{i \in I}$ be k-tuples of real variables indexed by a finite set I, let $(P_j)_{j \in J}$ be polynomials in k variables indexed by a finite set J, and consider the matrix $\boldsymbol{M}$ indexed by $I \times J$ such that $m_{i,j} = P_j(x_i)$ for all $(i,j) \in I \times J$. Assume that $\det(\boldsymbol{M})$ is not the null polynomial. There is an algorithm that runs in polynomial time in $\boldsymbol{M}$ and finds $|I|$ k-tuples of decimal fractions $(a_i)_{i \in I}$ with values in $[0,1]$ such that the matrix obtained by substituting each x_i by a_i in $\boldsymbol{M}$ is invertible.*

If $\det(\boldsymbol{U})$ is not the null polynomial, we can invoke this result with $k = 4$ and $I = J = [m+1]^4$ on the matrix $\boldsymbol{U}$, which gives us in polynomial time the desired rational probability values ρ_κ (namely, the a_i from the proposition) and concludes the proof of Proposition 3.1.

Hence, the only remaining point is to argue that $\det(\boldsymbol{U})$ is not the null polynomial (in the χ_κ). To this end, let us study the mapping $\xi : \mathbb{R}^4 \to \mathbb{R}^4$, defined as follows, with χ denoting a 4-tuple of real variables: $\xi(\chi) := \big(\Pi_4^{00}(\chi), \Pi_4^{01}(\chi), \Pi_4^{10}(\chi), \Pi_4^{11}(\chi)\big)$. For a 4-tuple of reals ρ, we call the mapping ξ *invertible* around point ρ if there is $\epsilon > 0$ such that the *ϵ-neighborhood around* $\xi(\rho)$, i.e., the set $\{\alpha \in \mathbb{R}^4 \mid |\alpha_{bb'} - \xi(\rho)_{bb'}| \leqslant \epsilon \text{ for each } b, b' \in \{0,1\}\}$, is included in the image of ξ. We conclude by showing two claims:

▶ **Fact 3.6.** *The mapping ξ is invertible around some point.*

Proof. By the inverse function theorem [23], if the *Jacobian determinant* of ξ at a point is not null, then ξ is invertible around that point. Recall that the Jacobian determinant of ξ is the determinant of the *Jacobian matrix* of ξ, which is the 4×4 matrix $\boldsymbol{J}_\xi$ whose entry at cell $((b_1, b_2), (b_1', b_2'))$ is $\frac{\partial \Lambda_{\chi, b_1 b_2}}{\partial \chi_{b_1' b_2'}}$. We explicitly compute $\det(\boldsymbol{J})$ with the help of SageMath, showing that it is not the null polynomial (see the full version [4]). ◀

▶ **Fact 3.7.** *If ξ is invertible around some point ρ, then $\det(\boldsymbol{U})$ is not the null polynomial.*

Proof. The invertibility of ξ around ρ implies that there exist, for each $b, b' \in \{0,1\}$, a set of $m+1$ distinct values $\Psi_{bb'} := \{\psi_{bb',0}, \ldots, \psi_{bb',m-1}\}$ such that the Cartesian product $\Psi := \bigtimes_{b,b' \in \{0,1\}} \Psi_{bb'}$ is included in the ϵ-neighborhood of $\xi(\rho)$. Let us index the $(m+1)^4$ 4-tuples of Ψ as ψ_κ for $\kappa \in [m+1]^4$, i.e., $\psi_\kappa = (\psi_{00,\kappa_{00}}, \psi_{01,\kappa_{01}}, \psi_{10,\kappa_{10}}, \psi_{11,\kappa_{11}})$. Using invertibility, let α_κ be a preimage of each ψ_κ, i.e., $\xi(\alpha_\kappa) = \psi_\kappa$ for all $\kappa \in [m+1]^4$. But then observe that, for this choice of χ_κ (i.e., substituting the χ_κ by the α_κ), each cell $u_{\kappa,\tau}$ of the matrix $\boldsymbol{U}$ becomes:

$$u_{\kappa,\tau} = 2^{-2m} \times (\psi_{00,\kappa_{00}})^{\tau_{00}} \times (\psi_{01,\kappa_{01}})^{\tau_{01}} \times (\psi_{10,\kappa_{10}})^{\tau_{10}} \times (\psi_{11,\kappa_{11}})^{\tau_{11}}.$$

Thus, $\boldsymbol{U}$ is the Kronecker product of four Vandermonde matrices $\boldsymbol{U}_{bb'}$ for $b, b' \in \{0,1\}$, where $\boldsymbol{U}_{bb'}$ is $\mathcal{V}(\psi_{bb',0}, \ldots, \psi_{bb',m-1})$. As the $\Psi_{bb'}$ consist of pairwise distinct values, these Vandermonde matrices are invertible, and their Kronecker product $\boldsymbol{U}$ also is. ◀

4 Proof When All Subdivisions Have the Same Length $\geqslant 7$

We now prove a variant of Proposition 3.1 where all edges of the initial graph are subdivided the same number of times (at least 7). Given a graph H and integer $K > 0$, we write G_K to mean $\mathrm{Sub}(H, K)$. In this section we show:

▶ **Proposition 4.1.** *Fix an integer $K \geqslant 7$. Then, for any graph family $\mathcal{F}$, the problem #Matching($\mathcal{F}$) reduces in polynomial time to PrMatching($\mathcal{G}$), where $\mathcal{G} = \{H_K \mid H \in \mathcal{F}\}$.*

To prove this, we follow the same strategy as for Proposition 3.1. The first step – the definition of the S_τ – is strictly identical; for m the number of edges of H, we fix again an orientation $\overrightarrow{H}$ of H, and denote S_τ for $\tau \in [m+1]^4$ the $(m+1)^4$ sets of selection functions defined from $\overrightarrow{H}$ as in Definition 3.2. In particular, Fact 3.3 still holds. Now, we will again construct $(m+1)^4$ probabilistic graphs, denoted $H_K(\kappa)$ for $\kappa \in [m+1]^4$, such that, letting $c_\kappa := \Pr_{\text{matching}}(H_K(\kappa))$, the $|S_\tau|$ and the c_κ form a linear system of equations $\boldsymbol{V}\boldsymbol{S} = \boldsymbol{C}$. We will then again use the Jacobian technique to argue that the determinant of this matrix is not the null polynomial, and complete the proof using Proposition 3.5 to compute

in polynomial time rational values that make $\boldsymbol{V}$ have rational entries and be invertible. The difference with Section 3 is in the construction of the probabilistic graphs $H_K(\kappa)$, and in the Jacobian determinant. Before we start, we need to extend the notation from Section 3.

Probabilistic path graphs. For $n \in \mathbb{N}^+$ we denote by P_n the *path of length* n, i.e., $P_n = (\{v_0, \ldots, v_n\}, E)$ where $E = \{\{v_i, v_{i+1}\} \mid 0 \leqslant i \leqslant n-1\}$. For $\rho \in [0,1]^n$, we let $P_n(\rho)$ be the probabilistic graph where each edge $\{v_i, v_{i+1}\}$ of P_n has probability ρ_i. We write $\Pi_n(\rho)$ the probability of a matching in $P_n(\rho)$. For $b, b' \in \{0,1\}$, we write $\Pi_n^{bb'}(\rho)$ to denote $\Pi_{n+2}(b, \rho, b')$, i.e., the probability of a matching in $P_n(\rho)$ where we add an edge to the left if $b = 1$ and add an edge to the right if $b' = 1$. In particular $\Pi_n^{00}(\rho) = \Pi_n(\rho)$. We call the quadruple of values $\Pi_n^{bb'}(\rho)$ for $b, b' \in \{0,1\}$ the *behavior* of the path $P_n(\rho)$. Each $\Pi_n^{bb'}(\rho)$ is a polynomial in the probabilities ρ, and thus we also see $\Pi_n^{bb'}$ as a polynomial with real variables as in Section 3. We will use the following two lemmas. The first one expresses the behavior of the concatenation of two paths as a function of the behavior of each path (see the full version [4]):

▶ **Lemma 4.2.** *Let $n, n' \in \mathbb{N}^+$ and $\rho \in [0,1]^n$, $\rho' \in [0,1]^{n'}$ be tuples of probability values. Then, for every $b, b' \in \{0,1\}$, we have:*

$$\Pi_{n+n'}^{bb'}(\rho, \rho') = (\Pi_n^{b0}(\rho) \times \Pi_{n'}^{1b'}(\rho')) + (\Pi_n^{b1}(\rho) \times \Pi_{n'}^{0b'}(\rho')) - (\Pi_n^{b1}(\rho) \times \Pi_{n'}^{1b'}(\rho')).$$

The second lemma expresses the values $\Pi_n^{bb'}(1/2, \ldots, 1/2)$ in terms of the *Fibonacci sequence*. Recall that this is the integer sequence defined by $f_0 := 0$, $f_1 := 1$, and $f_n := f_{n-1} + f_{n-2}$ for all $n \in \mathbb{N}^+$, and that this sequence satisfies *Cassini's identity* [22], which says that $f_n^2 = f_{n+1} f_{n-1} + (-1)^{n+1}$ for every $n \in \mathbb{N}^+$. We have (see the full version [4]):

▶ **Lemma 4.3.** *For all $n \in \mathbb{N}^+$, $b, b' \in \{0,1\}$, we have $\Pi_n^{bb'}(1/2, \ldots, 1/2) = \frac{f_{n+2-b-b'}}{2^n}$.*

Proving Proposition 4.1. Let us now build the graphs $H_K(\kappa)$. As before, consider $(m+1)^4$ 4-tuples of probability values $\rho_\kappa = (\rho_{\kappa,00}, \rho_{\kappa,01}, \rho_{\kappa,10}, \rho_{\kappa,11})$ for $\kappa \in [m+1]^4$, to be chosen later. Each graph $H_K(\kappa)$ has H_K as its underlying graph, and for every directed edge $(x, y) \in \overrightarrow{H}$, we set the probabilities on the corresponding undirected path in H_K as follows:

$$x \overset{1/2}{\text{---}} v_1 \overset{\rho_{\kappa,00}}{\text{---}} v_2 \overset{\rho_{\kappa,01}}{\text{---}} v_3 \overset{\rho_{\kappa,10}}{\text{---}} v_4 \overset{\rho_{\kappa,11}}{\text{---}} v_5 \overset{1/2}{\text{---}} v_6 \overset{1/2}{\text{---}} \cdots \overset{1/2}{\text{---}} v_{K-1} \overset{1/2}{\text{---}} y$$

Note that this is like in Section 3, but giving probability 1/2 to the $N := K - 6$ extra edges on the path. For $b, b' \in \{0,1\}$ we write again $\Lambda_{\kappa,bb'} := \Pi_4^{bb'}(\rho_\kappa)$ the behavior of the 4-path with probabilities ρ_κ, and we define the behavior $\Upsilon_{\kappa,bb'} := \Pi_{K-2}^{bb'}(\rho_\kappa, 1/2, \ldots, 1/2)$ of the path depicted above without the first and last edges. Note that with Lemma 4.2 and Lemma 4.3, we can then express the $\Upsilon_{\kappa,bb'}$ as a function of the $\Lambda_{\kappa,bb'}$ and of the Fibonacci numbers:

▶ **Fact 4.4.** *We have $\Upsilon_{\kappa,bb'} = 2^{-N} \times (\Lambda_{\kappa,b0} \times f_{N+1-b'} + \Lambda_{\kappa,b1} \times f_{N-b'})$ for $b, b' \in \{0,1\}$.*

Studying the graphs $H_K(\kappa)$, by the same reasoning as for Fact 3.4, we can easily show:

$$2^{2m} \times \Pr_{\text{matching}}(H_K(\kappa)) = \sum_{\tau \in [m+1]^4} |S_\tau| \times (\Upsilon_{\kappa,00})^{\tau_{00}} \times (\Upsilon_{\kappa,01})^{\tau_{01}} \times (\Upsilon_{\kappa,10})^{\tau_{10}} \times (\Upsilon_{\kappa,11})^{\tau_{11}}. \quad (2)$$

This is again a system of linear equations $\boldsymbol{V}\boldsymbol{S} = \boldsymbol{C}$ with $\boldsymbol{V} \in \mathbb{R}^{[m+1]^4, [m+1]^4}$, where $v_{\kappa,\tau} := 2^{-2m} \times (\Upsilon_{\kappa,00})^{\tau_{00}} \times (\Upsilon_{\kappa,01})^{\tau_{01}} \times (\Upsilon_{\kappa,10})^{\tau_{10}} \times (\Upsilon_{\kappa,11})^{\tau_{11}}$. To show that we can compute in polynomial time 4-tuples of rational probability values ρ_κ for $\kappa \in [m+1]^4$ that make $\boldsymbol{V}$ have rational entries and be invertible, we reason as in Section 3. Specifically, we study the Jacobian determinant of the mapping $\xi_N : \chi \mapsto (\Pi_{K-2}^{00}(\chi, 1/2, \ldots, 1/2), \Pi_{K-2}^{01}(\chi, 1/2, \ldots, 1/2),$

$\Pi^{10}_{K-2}(\chi, 1/2, \ldots, 1/2)$, $\Pi^{11}_{K-2}(\chi, 1/2, \ldots, 1/2)\big)$, where χ is a 4-tuple of real variables. We show that this determinant is not the null polynomial. To do this, starting from the Jacobian $\boldsymbol{J}_\xi$ of Section 3, using Fact 4.4 and Cassini's identity, and using the fact that the determinant is multilinear and alternating, we obtain (see the full version [4]):

▶ **Fact 4.5.** *We have:* $\det(\boldsymbol{J}_{\xi_N}) = 2^{-4N} \times \det(\boldsymbol{J}_\xi)$.

Hence, $\det(\boldsymbol{J}_{\xi_N})$ is not the null polynomial and, as in Section 3, we can use Proposition 3.5 to complete the proof of Proposition 4.1 (see the full version [4]).

5 Proof for Arbitrary Subdivisions

In this section we finally prove our main result (Result 1), which we re-state here:

▶ **Theorem 5.1.** *Let $\mathcal{G}$ be an arbitrary family of graphs which is treewidth-constructible. Then PrMatching($\mathcal{G}$) is #P-hard under ZPP reductions.*

We will reduce from the problem of counting matchings in 3-regular planar graphs of, which is #P-hard[3] by [26]. Our reduction will be similar to that of Section 4, with the major issue that the various edges of the input graph can now be subdivided to different lengths.

The proof consists of five steps. In step 1, we show a general result allowing us to assign probabilities to a path of length 4 so as to "emulate" the behavior of any long path of *even* length. We then revisit the proof of the previous section. Step 2 extracts the input graph H from the treewidth-constructible family. Step 3 relates the number of matchings of H to cardinalities similar to those of the previous section, but taking the parities of the subdivisions into account. Step 4 then explains how to conclude using emulation. Last, step 5 works around the issue that the probabilities of Step 1 could be irrational, by explaining how we can conclude with sufficiently precise approximations. We now detail these steps.

Step 1: Emulating long even paths. We start by presenting the main technical tool, namely, how to emulate long paths of even length by paths of length 4.

▶ **Proposition 5.2** (Emulation result). *There exist closed-form expressions, denoted $p(i), q(i), r(i), s(i)$, such that for every even integer $i \geqslant 4$ the following hold:*

(A) *the expressions evaluate to well-defined probability values, i.e., we have $0 \leqslant p(i), q(i), r(i), s(i) \leqslant 1$; and*

(B) *the path of length 4 with probabilities $p(i), q(i), r(i), s(i)$ behaves like a path of length i with probabilities $1/2$, i.e., $\Pi_4^{bb'}(p(i), q(i), r(i), s(i)) = \Pi_i^{bb'}(1/2, \ldots, 1/2)$ for all $b, b' \in \{0, 1\}$.*

Further, each of these expressions is of the form $\frac{P \pm \sqrt{Q}}{R}$ where P, Q, R are polynomials in the Fibonacci numbers f_{i-1} and f_{i-2} and in 2^{-i}, with rational coefficients.

Proof sketch. The result is simple to state, but we did not find an elegant way to show it. Our proof consists of four steps: (i) rewriting condition (B) into a simpler equivalent system of equations (using Lemma 4.3), (ii) proving that any solution of that system must be in $(0, 1)^4$, (iii) exhibiting closed-form expressions that satisfy the system, found with the help of SageMath; and (iv) verifying that these expressions are well-defined. See the full version [4]. ◀

[3] Note that, in holographic literature, graphs may be multigraphs (i.e., can have multiple edges between two nodes) – see [15]. However, inspecting the proof of [26], we see that the graphs are in fact simple.

▶ Remark 5.3. As $\Pi_i^{bb'}(1/2, \ldots, 1/2)$ is symmetric, one would expect the closed-form expressions to satisfy $p(i) = s(i)$ and $q(i) = r(i)$. However, surprisingly, numerical evaluation (already for $i = 6$) shows that our solution does not have this property.

▶ Remark 5.4. It is necessary to require that i is even, as otherwise Proposition 5.2 demonstrably does not hold. In fact, we can prove that, more generally, the behavior of a probabilistic path inherently depends on the parity of its length (see the full version [4]). This is why we will distinguish even-length and odd-length subdivisions in the sequel.

Step 2: Choosing the graph in $\mathcal{G}$. Let $H = (V, E)$ be the input to the reduction, i.e., the 3-regular planar graph for which we want to compute #Matching(H), and let $m := |E|$. We first build the graph $H_{10} = \mathrm{Sub}(H, 10)$, writing $H_{10} = (V_{10}, E_{10})$ and we compute $k := |V_{10}|^c$ where c is the constant from Theorem 2.1. Notice that H_{10} is a planar graph of maximum degree 3, and that the size of k in unary is polynomial in (the encoding size of) H. Intuitively, this initial subdivision in 10 will ensure that we have enough room for our probabilistic gadgets. Now, we use the treewidth-constructibility of $\mathcal{G}$ to build in polynomial time a graph $G = (V_G, E_G) \in \mathcal{G}$ such that $\mathrm{tw}(G) \geqslant k$, and using Theorem 2.1 we compute in ZPP a subgraph G' of G with a subdivision $\eta_{10} : E_{10} \to \mathbb{N}^+$ of H_{10} and an isomorphism from $\mathrm{Sub}(H_{10}, \eta_{10})$ to G'. This gives us a subdivision $\eta : E \to \mathbb{N}^+$ of H and an isomorphism f from $\mathrm{Sub}(H, \eta)$ to G', with the initial subdivision ensuring that $\eta(e) \geqslant 10$ for each $e \in E$.

Step 3: Defining the new sets $S_{\tau,\tau'}$ and linking them to matchings. As before, fix an orientation $\overrightarrow{H}$ of H. We call an edge e of H *even* if $\eta(e)$ is even, and *odd* otherwise. For $\tau, \tau' \in [m+1]^4$, both indexed in binary, we define $S_{\tau,\tau'}$ to be the set of selection functions μ of H such that, for $b, b' \in \{0, 1\}$, precisely $\tau_{bb'}$ even edges e of H have type bb' w.r.t. μ, and precisely $\tau'_{bb'}$ odd edges e of H have type bb' w.r.t. μ. Then, as in Section 3, we have:

$$\#\mathrm{Matching}(H) = \sum_{\substack{\tau,\tau' \in [m+1]^4 \\ \tau_{01}=\tau_{10}=\tau'_{01}=\tau'_{10}=0}} |S_{\tau,\tau'}|. \tag{3}$$

Step 4: Describing the probabilistic graphs and obtaining the system. To complete the definition of the reduction, let us build the $(m+1)^8$ probabilistic graphs on which we want to invoke the oracle, denoted $G(\kappa, \kappa')$ for $\kappa, \kappa' \in [m+1]^4$. Let $K := \max_{\substack{e \in E \\ \eta(e) \text{ is even}}}(\eta(e))$ and $K' := \max_{\substack{e \in E \\ \eta(e) \text{ is odd}}}(\eta(e))$ and $N := K - 6$ and $N' := K' - 6$. The underlying graph of $G(\kappa, \kappa')$ is G, every edge $e \in E_G$ that is not in G' is assigned probability zero, and we explain next what is the probability associated to the edges that are in G'. Consider $2 \times (m+1)^4$ 4-tuples of probability values $\rho_\kappa = (\rho_{\kappa,00}, \rho_{\kappa,01}, \rho_{\kappa,10}, \rho_{\kappa,11})$ and $\rho'_\kappa = (\rho'_{\kappa',00}, \rho'_{\kappa',01}, \rho'_{\kappa',10}, \rho'_{\kappa',11})$ for $\kappa, \kappa' \in [m+1]^4$, to be chosen later. For every directed edge $(x, y) \in \overrightarrow{H}$, let $\gamma := \eta(\{x, y\})$ be the length to which it is subdivided in G'. Letting $f(x), v_1, \ldots, v_{\gamma-1}, f(y)$ be the corresponding path in G', we set the probabilities of the γ edges along that path as follows:

- If γ is even (illustrated in Figure 1):
  - $1/2, \rho_{\kappa,00}, \rho_{\kappa,01}, \rho_{\kappa,10}, \rho_{\kappa,11}$ for the first 5 edges,
  - $p(N - \gamma + 10), q(N - \gamma + 10), r(N - \gamma + 10), s(N - \gamma + 10)$ for the next four edges,
  - $1/2$ for the remaining $\gamma - 9$ edges.
- If γ is odd:
  - $1/2, \rho'_{\kappa',00}, \rho'_{\kappa',01}, \rho'_{\kappa',10}, \rho'_{\kappa',11}$ for the first 5 edges,
  - $p(N' - \gamma + 10), q(N' - \gamma + 10), r(N' - \gamma + 10), s(N' - \gamma + 10)$ for the next four edges,
  - $1/2$ for the remaining $\gamma - 9$ edges.

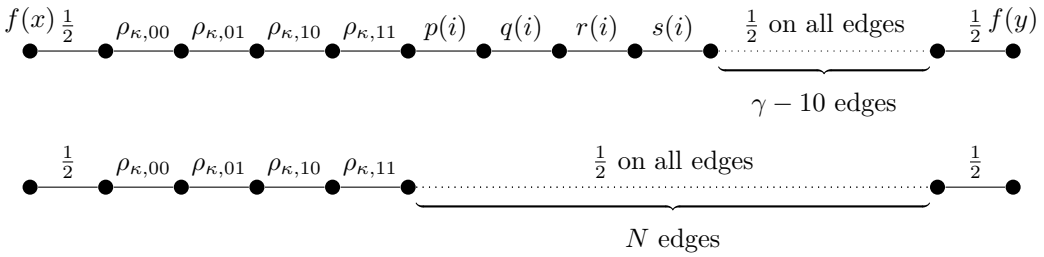

Figure 1 The upper path depicts how we set the probabilities along a path $f(x), v_1, \ldots, v_{\gamma-1}, f(y)$ corresponding to an edge $(x, y) \in \overrightarrow{H}$ such that $\gamma := \eta(\{x, y\})$ is even. We write $i := N - \gamma + 10$. By Lemma 4.2 and Proposition 5.2, this path has exactly the same behavior as the lower path.

We know that $N - \gamma + 10$ (resp., $N' - \gamma + 10$) is an even integer when γ is even (resp., when γ is odd); and it is $\geqslant 4$ by definition of K (resp., of K'). Thus, using Proposition 5.2 and then Lemma 4.2, we know that the path that we defined behaves exactly like the path $P_K(1/2, \rho_{\kappa,00}, \rho_{\kappa,01}, \rho_{\kappa,10}, \rho_{\kappa,11}, 1/2, \ldots, 1/2)$ if γ is even, and exactly like the path $P_{K'}(1/2, \rho'_{\kappa',00}, \rho'_{\kappa',01}, \rho'_{\kappa',10}, \rho'_{\kappa',11}, 1/2, \ldots, 1/2)$ if γ is odd (see again Figure 1).

We have now managed to ensure that all paths for even edges (resp., for odd edges) behave as if they had been subdivided to length K (resp., to length K'). We continue the proof as in the previous section, except that we distinguish odd and even edges. Specifically, for $b, b' \in \{0, 1\}$, we write as in the previous section $\Upsilon_{\kappa,bb'} := \Pi^{bb'}_{K-2}(\rho_\kappa, 1/2, \ldots, 1/2)$ and $\Upsilon'_{\kappa',bb'} := \Pi^{bb'}_{K'-2}(\rho'_{\kappa'}, 1/2, \ldots, 1/2)$. Using the same reasoning as for Equation 2, we obtain:

$$2^{2m} \times \Pr_{\text{matching}}(G(\kappa, \kappa')) = \sum_{\tau,\tau' \in [m+1]^4} |S_{\tau,\tau'}| \times (\Upsilon_{\kappa,00})^{\tau_{00}} \times (\Upsilon_{\kappa,01})^{\tau_{01}} \times (\Upsilon_{\kappa,10})^{\tau_{10}} \times (\Upsilon_{\kappa,11})^{\tau_{11}} \times (\Upsilon'_{\kappa',00})^{\tau'_{00}} \times (\Upsilon'_{\kappa',01})^{\tau'_{01}} \times (\Upsilon'_{\kappa',10})^{\tau'_{10}} \times (\Upsilon'_{\kappa',11})^{\tau'_{11}}, \quad (4)$$

i.e., we obtain a system of linear equations $\boldsymbol{\Gamma S} = \boldsymbol{C}$ with $\boldsymbol{S}$ the vector of the desired values $|S_{\tau,\tau'}|$, with $\boldsymbol{C}$ the vector of the oracle answers $\Pr_{\text{matching}}(G(\kappa, \kappa'))$, and with $\boldsymbol{\Gamma} \in \mathbb{R}^{[m+1]^8,[m+1]^8}$, whose entries are given according to the above equation. But notice that we have $\boldsymbol{\Gamma} = \boldsymbol{V} \otimes \boldsymbol{V'}$, with $v_{\kappa,\tau} := 2^{-m} \times (\Upsilon_{\kappa,00})^{\tau_{00}} \times (\Upsilon_{\kappa,01})^{\tau_{01}} \times (\Upsilon_{\kappa,10})^{\tau_{10}} \times (\Upsilon_{\kappa,11})^{\tau_{11}}$ and $v'_{\kappa',\tau'} := 2^{-m} \times (\Upsilon'_{\kappa',00})^{\tau'_{00}} \times (\Upsilon'_{\kappa',01})^{\tau'_{01}} \times (\Upsilon'_{\kappa',10})^{\tau'_{10}} \times (\Upsilon'_{\kappa',11})^{\tau'_{11}}$. Since $\boldsymbol{V}$ and $\boldsymbol{V'}$ share no variables and are identical up to renaming variables, to argue that there exist 4-tuples of probabilistic values ρ_κ and $\rho'_{\kappa'}$ for $\kappa, \kappa' \in [m+1]^4$ that make $\boldsymbol{\Gamma}$ invertible, it is enough to know that the Jacobian determinant of the mapping ξ_N is not identically null, as we showed in the previous section (Fact 4.5). Thus, we can again use Proposition 3.5 to compute in polynomial time $2 \times (m+1)^4$ 4-tuples of rational probability values ρ_κ and $\rho'_{\kappa'}$ such that the matrices $\boldsymbol{V}$ and $\boldsymbol{V'}$, hence $\boldsymbol{\Gamma}$, are invertible (see the full version [4]). By Equation 4, $\boldsymbol{\Gamma}$ has rational entries, and its inverse $\boldsymbol{\Gamma}^{-1}$ also does and is computable in polynomial time.

Step 5: Using decimal fractions approximations. The last issue is that we cannot really obtain $\boldsymbol{C}$ via oracle calls, because the graphs $G(\kappa, \kappa')$ may have irrational edge probabilities, namely, the $p(i), q(i), r(i), s(i)$. We now argue that we can still recover the $\boldsymbol{C}$, so that we can compute $\boldsymbol{S} = \boldsymbol{\Gamma}^{-1}\boldsymbol{C}$ and conclude. To do this, we first observe that $\boldsymbol{C}$ is in fact a vector of decimal fractions, as the graphs $G(\kappa, \kappa')$ emulate a graph where the probabilities are decimal fractions; further, we can bound the number of decimal places of its values to $[m \times (\max(N, N') + 10)] \times z$, with z the maximal number of decimal places of a decimal fraction in $\rho_\kappa, \rho'_\kappa$ Second, we show how to compute decimal fraction *approximations* $\widehat{p(i)}, \widehat{q(i)}, \widehat{r(i)}, \widehat{s(i)}$

of the $p(i), q(i), r(i), s(i)$, in polynomial time in the desired number of places, using the form that they have according to Proposition 5.2. Third, we argue that when invoking the oracles on the graphs where we replace $p(i), q(i), r(i), s(i)$ by $\widehat{p(i)}, \widehat{q(i)}, \widehat{r(i)}, \widehat{s(i)}$, then the error on the answer is bounded as a function of that of the approximations, so that we can recover $\boldsymbol{C}$ exactly if the approximations were sufficiently precise. See the full version [4] for detailed proofs.

6 Result for Edge Covers

Having shown Result 1, we now explain how to adapt its proof to obtain our analogous results for edge covers. We only sketch the argument, and refer to the full version [4] for more details. Recall that an *edge cover* of a graph $G = (V, E)$ is a set of edges $S \subseteq E$ such that $V = \bigcup_{e \in S} e$. Given a probabilistic graph (G, π), we define $\mathrm{Pr}_{\mathrm{edgeCover}}(G, \pi)$ to be the sum of the probabilities of all edge covers in the probability distribution induced by π, and define PrEdgeCover($\mathcal{F}$) for a graph family $\mathcal{F}$ to be the corresponding computational problem. We first note that, in this context, the strict analogue of Result 1 does not hold. Indeed, take some treewidth-constructible graph family $\mathcal{G}$, and consider the graph family $\mathcal{G}'$ obtained from $\mathcal{G}$ as follows: for every graph $G \in \mathcal{G}$, we add to $\mathcal{G}'$ the graph that is obtained from G by attaching a dangling edge with a fresh vertex to every node of G. The family $\mathcal{G}'$ is still treewidth-constructible, but PrEdgeCover($\mathcal{G}'$) is now tractable as it is easy to see that the edge covers of a graph in $\mathcal{G}'$ are precisely the edge subsets where all dangling edges are kept.

To avoid this, let us assume that $\mathcal{G}$ is closed under taking subgraphs, i.e., if $G \in \mathcal{G}$ and G' is a subgraph of G, then $G' \in \mathcal{G}$. We then have:

▶ **Theorem 6.1.** *Let $\mathcal{G}$ be an arbitrary family of graphs which is treewidth-constructible and closed under taking subgraphs. Then PrEdgeCover($\mathcal{G}$) is #P-hard under ZPP reductions.*

This is proved like Result 1, with the following modifications. We reduce from counting edge covers (instead of matchings) on 3-regular planar graphs: this is hard by [8], even on simple graphs [2, Appendix D]. We now define a selection function μ to map each vertex $x \in V$ to *at least* one incident edge, and we define the types and the sets $S_{\tau,\tau'}$ as before, via an arbitrary orientation of the graph H. We obtain the number of edge covers of H from the quantities $|S_{\tau,\tau'}|$ exactly as in Equation 3. We redefine $\Pi_n^{bb'}(\rho)$ to be the probability of an edge cover in a path of length n with probabilities ρ on the edges and with endpoint constraints given by b, b' as before. Lemma 4.3 then becomes $\Pi_n^{bb'}(1/2, \ldots, 1/2) = \frac{f_{n+b+b'}}{2^n}$, i.e., the role of b, b' is "reversed". Analogous versions of Lemma 4.2 and of Proposition 5.2 still hold, so the relevant Jacobian determinants are still non-identically null. We take the graph $G \in \mathcal{G}$ again via the topological minor extraction result, but this time directly extracting $\mathrm{Sub}(H, \eta) \in \mathcal{G}$ as $\mathcal{G}$ is subgraph-closed. The rest of the proof is identical.

We point out that the situation is different for *perfect* matchings. Indeed, using a weighted variant of the FKT algorithm [7, Chapter 4], the weighted counting of perfect matchings is polynomial-time over the class of planar graphs, which is treewidth-constructible.

We conclude by leaving open two directions for future work. The first one would be to obtain the same kind of lower bounds when the probabilities annotate the *nodes* instead of the edges, that is, studying the corresponding weighted counting problems for, e.g., independent sets, vertex covers, or cliques. We believe that the corresponding result should hold and do not expect any surprises. The second question would be to show our hardness results in the *unweighted* case, e.g., unweighted counting of matchings, assuming that the graph family is subgraph-closed. This appears to be much more challenging, as our current proof crucially relies on the ability to use arbitrary probability values.

References

1 Antoine Amarilli, Pierre Bourhis, and Pierre Senellart. Tractable lineages on treelike instances: Limits and extensions. In *PODS*, pages 355–370, 2016. arXiv:1604.02761.

2 Antoine Amarilli, Mikaël Monet, and Pierre Senellart. Conjunctive queries on probabilistic graphs: Combined complexity. In *PODS*, pages 217–232, 2017. arXiv:1703.03201.

3 Antoine Amarilli, Mikaël Monet, and Pierre Senellart. Connecting width and structure in knowledge compilation. In *ICDT*, volume 98, pages 6:1–6:17, 2018. arXiv:1709.06188.

4 Antoine Amarilli and Mikaël Monet. Weighted counting of matchings in unbounded-treewidth graph families. In *MFCS*, 2022. Full version with proofs. arXiv:2205.00851.

5 Stefan Arnborg, Jens Lagergren, and Detlef Seese. Easy problems for tree-decomposable graphs. *Journal of Algorithms*, 12(2):308–340, 1991. URL: https://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.12.2544&rep=rep1&type=pdf.

6 Paul Beame, Guy Van den Broeck, Eric Gribkoff, and Dan Suciu. Symmetric weighted first-order model counting. In *PODS*, pages 313–328, 2015. arXiv:1412.1505.

7 Jin-Yi Cai and Xi Chen. *Complexity Dichotomies for Counting Problems: Volume 1, Boolean Domain*. Cambridge University Press, 2017.

8 Jin-Yi Cai, Pinyan Lu, and Mingji Xia. Holographic reduction, interpolation and hardness. *Computational Complexity*, 21(4):573–604, 2012. URL: https://pages.cs.wisc.edu/~jyc/papers/interpolation08.pdf.

9 Venkat Chandrasekaran, Nathan Srebro, and Prahladh Harsha. Complexity of Inference in Graphical Models. In *UAI*, pages 70–78, 2008. URL: https://dslpitt.org/uai/displayArticleDetails.jsp?mmnu=1&smnu=2&article_id=1343&proceeding_id=24.

10 Chandra Chekuri and Julia Chuzhoy. Polynomial bounds for the grid-minor theorem. *Journal of the ACM*, 63(5):1–65, 2016. arXiv:1305.6577.

11 Víctor Dalmau and Peter Jonsson. The complexity of counting homomorphisms seen from the other side. *Theor. Comput. Sci.*, 329(1):315–323, 2004. URL: https://core.ac.uk/reader/81145200.

12 Nilesh N. Dalvi and Dan Suciu. The dichotomy of probabilistic inference for unions of conjunctive queries. *Journal of the ACM*, 59(6):30, 2012. URL: https://homes.cs.washington.edu/~suciu/jacm-dichotomy.pdf.

13 Robert Ganian, Petr Hliněný, Alexander Langer, Jan Obdržálek, Peter Rossmanith, and Somnath Sikdar. Lower bounds on the complexity of MSO1 model-checking. *JCSS*, 1(80):180–194, 2014. arXiv:1109.5804.

14 Catherine Greenhill. The complexity of counting colourings and independent sets in sparse graphs and hypergraphs. *computational complexity*, 9(1):52–72, 2000. URL: http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.467.3100&rep=rep1&type=pdf.

15 M.Monet (https://cstheory.stackexchange.com/users/38111/m monet). Holant problems and holographic reduction: simple graphs or multigraphs? Theoretical Computer Science Stack Exchange. URL: https://cstheory.stackexchange.com/q/43912 (version: 2019-05-18).

16 Stephan Kreutzer and Siamak Tazari. Lower bounds for the complexity of monadic second-order logic. In *LICS*, pages 189–198, 2010. arXiv:1001.5019.

17 Johan Kwisthout, Hans L. Bodlaender, and Linda C. van der Gaag. The necessity of bounded treewidth for efficient inference in Bayesian networks. In *ECAI*, pages 237–242, 2010. URL: http://www.booksonline.iospress.nl/Content/View.aspx?piid=17749.

18 Salil P Vadhan. The complexity of counting in sparse, regular, and planar graphs. *SIAM Journal on Computing*, 31(2):398–427, 2001. URL: https://epubs.siam.org/doi/pdf/10.1137/S0097539797321602.

19 Leslie G Valiant. The complexity of computing the permanent. *Theoretical computer science*, 8(2):189–201, 1979. URL: https://core.ac.uk/download/pdf/82500417.pdf.

20 Leslie G. Valiant. The complexity of enumeration and reliability problems. *SIAM J. Comput.*, 8(3):410–421, 1979. URL: https://www.math.cmu.edu/~af1p/Teaching/MCC17/Papers/enumerate.pdf.

21 Wikipedia. Blossom algorithm, 2022. URL: `https://en.wikipedia.org/wiki/Blossom_algorithm`.

22 Wikipedia. Cassini and Catalan identities, 2022. URL: `https://en.wikipedia.org/wiki/Cassini_and_Catalan_identities`.

23 Wikipedia. Inverse function theorem, 2022. URL: `https://en.wikipedia.org/wiki/Inverse_function_theorem`.

24 Wikipedia. Planar graphs, 2022. URL: `https://en.wikipedia.org/wiki/Planar_graphs`.

25 Wikipedia. Treewidth, 2022. URL: `https://en.wikipedia.org/wiki/Treewidth`.

26 Mingji Xia, Peng Zhang, and Wenbo Zhao. Computational complexity of counting problems on 3-regular planar graphs. *Theor. Comput. Sci.*, 384(1):111–125, 2007. URL: `https://core.ac.uk/download/pdf/82063901.pdf`.

27 Mingji Xia and Wenbo Zhao. #3-regular bipartite planar vertex cover is #P-complete. In *International Conference on Theory and Applications of Models of Computation*, pages 356–364. Springer, 2006.