Approximate Counting of k-Paths: Simpler, Deterministic and in Polynomial Space*

Andreas Björklund† Daniel Lokshtanov‡ Saket Saurabh§ Meirav Zehavi¶

Abstract

Recently, Brand et al. [STOC 2018] gave a *randomized* $\mathcal{O}(4^k m\epsilon^{-2})$-time exponential-space algorithm to approximately compute the number of paths on k vertices in a graph G up to a multiplicative error of $1\pm\epsilon$, based on exterior algebra. Prior to our work, this has been the state-of-the-art. In this article, we revisit the algorithm by Alon and Gutner [IWPEC 2009, TALG 2010], and obtain the following results.

- We present a *deterministic* $4^{k+\mathcal{O}(\sqrt{k}(\log^2 k+\log^2 \epsilon^{-1}))} m \log n$-time *polynomial-space* algorithm. This *matches* the running time of the best known deterministic polynomial-space algorithm for *deciding* whether a given graph G has a path on k vertices.
- Additionally, we present a *randomized* $4^{k+\mathcal{O}(\log k(\log k+\log \epsilon^{-1}))} m \log n$-time *polynomial-space* algorithm. Our algorithm is simple—we only make elementary use of the probabilistic method.

Here, n and m are the number of vertices and the number of edges, respectively. Additionally, our approach extends to approximate counting of other patterns of small size (such as q-dimensional p-matchings).

*A preliminary version of this paper appeared in the proceedings of ICALP 2019.

†Lund University, Lund, Sweden. andreas.bjorklund@cs.lth.se

‡University of California, Santa Barbara, USA. daniello@ucsb.edu

§The Institute of Mathematical Sciences, HBNI, Chennai, India. saket@imsc.res.in

¶Ben-Gurion University, Beersheba, Israel. meiravze@bgu.ac.il

1 Introduction

The objective of the #k-Path problem is to compute the number of k-paths—that is, (simple) paths on k vertices—in a given graph G. Unfortunately, this problem is #W[1]-hard [21], which means that it is unlikely to be solvable in time $f(k)n^{\mathcal{O}(1)}$ for any computable function f of k. Nevertheless, this problem is long known to admit an FPT-approximation scheme (FPT-AS), that is, an $f(k, \epsilon^{-1})n^{\mathcal{O}(1)}$-time algorithm that approximately computes the number of k-paths in a given graph G up to a multiplicative error of $1 \pm \epsilon$. More than 15 years ago, Arvind and Raman [7] utilized the classic method of *color coding* [6] to design a *randomized* exponential-space FPT-AS for #k-Path with running time $k^{\mathcal{O}(k)}n^{\mathcal{O}(1)}$ whenever $\epsilon^{-1} \leq k^{\mathcal{O}(k)}$. A few years afterwards, the development and use of applications in computational biology to detect and analyze *network motifs* have already become common practice [36, 39, 38, 20, 26]. Roughly speaking, a network motif is a small pattern whose number of occurrences in a given network is substantially larger than its number of occurrences in a random network. Due to their tight relation to network motifs, #k-Path and other cases of the #Subgraph Isomorphism problem became highly relevant to the study of gene transcription networks, protein-protein interaction (PPI) networks, neural networks and social networks [33]. In light of these developments, Alon et al. [2] revisited the method of color coding to attain a running time whose dependency on k is *single-exponential* rather than *slightly super-exponential*. Specifically, they designed a *simple randomized* $\mathcal{O}((2e)^k m\epsilon^{-2})$-time exponential-space FPT-AS for #k-Path, which they employed to analyze PPI networks of unicellular organisms. In particular, their algorithm has running time $2^{\mathcal{O}(k)}m$ whenever $\epsilon^{-1} \leq 2^{\mathcal{O}(k)}$. Here, n and m are the number of vertices and the number of edges, respectively.

The first *deterministic* FPT-AS for #k-Path was found in 2007 by Alon and Gutner [4]; this algorithm has an exponential space complexity and running time $2^{\mathcal{O}(k \log\log k)}m \log n$ whenever $\epsilon^{-1} = 2^{o(\log k)}$. Shortly afterwards, Alon and Gutner [3] improved upon their previous work, and designed a deterministic exponential-space FPT-AS for #k-Path with running time $(2e)^{k+\mathcal{O}(\log^3 k)}m \log n$ whenever $\epsilon^{-1} = k^{\mathcal{O}(1)}$. For close to a decade, this algorithm has remained the state-of-the-art. In contrast, during this decade, the k-Path problem (the decision version of #k-Path) has seen several improvements that were considered to be breakthroughs at their time [16, 28, 9, 11, 23]. In 2016, Koutis and Williams [29] conjectured that #k-Path admits an FPT-AS with running time $2^k n^{\mathcal{O}(1)}$. Recently, at the cost of reintroducing randomization, Brand et al. [14] provided a speed-up towards the resolution of this conjecture. Specifically, they gave an algebraic *randomized* $\mathcal{O}(4^k m\epsilon^{-2})$-time exponential-space algorithm. In the context of Parameterized Complexity in general, and the k-Path problem in particular, the power of randomization is an issue of wide interest [1]. Specifically for the k-Path problem, an algebraic randomized $2^k n^{\mathcal{O}(1)}$-time algorithm has been found already a decade ago [41], and since then, the existence of a deterministic algorithm that exhibits the same time complexity has been repeatedly posed as a major open problem in the field. Both Koutis and Williams conjectured this question to have an affirmative answer in several venues [41, 30, 29]. Clearly, this question is simpler than the one of the design of a deterministic FPT-AS for #k-Path with running time $2^k n^{\mathcal{O}(1)}$.

In this article, we modify the foundation of the work of Alon and Gutner [4, 3], and with a novel twist, obtain the following results (see Theorem 6.1 and Corollary 4.1).

- First, we present a *randomized* $4^{k+\mathcal{O}(\log k(\log k+\log \epsilon^{-1}))}m \log n$-time *polynomial-space* algorithm. For this purpose, we only make elementary use of the probabilistic method.

- Additionally, we present a *deterministic* $4^{k+\mathcal{O}(\sqrt{k}(\log^2 k+\log^2 \epsilon^{-1}))}m \log n$-time *polynomial-space* algorithm. In particular, *without compromising time complexity, we attain both the properties of having a polynomial space complexity and being deterministic simultaneously.*

In fact, even though we deal with #k-Path, the running time of our algorithm *matches* the best known running time of a deterministic polynomial-space algorithm for k-Path (the decision version of #k-Path) [16].

Thus, the algorithm by Brand et al. [14] runs in time $4^{k+o(k)}m$ whenever $\epsilon^{-1} = 2^{o(k)}$, while our deterministic and randomized algorithms run in time $4^{k+o(k)}m \log n$ whenever $\epsilon^{-1} = 2^{o(k^{\frac{1}{4}})}$ and $\epsilon^{-1} = 2^{o(\frac{k}{\log k})}$, respectively.

The design of polynomial-space parameterized algorithms is an active research area in Parameterized Complexity. Even (sometimes) at a notable compromise of time complexity, the property of having polynomial space complexity is sought (see, e.g., [22, 32, 31, 8, 25]). Indeed, algorithms with high space complexity are in practice more constrained because the amount of memory is not easily scaled beyond hardware constraints whereas time complexity can be alleviated by allowing for more time for the algorithm to finish. Furthermore, algorithms with low space complexity are typically easier to parallelize and more cache-friendly.

Additionally, we remark that our approach is embeddable in the classic framework of divide-and-color, hence it immediately extends to approximate counting of graphs of bounded treewidth; in comparison, Brand et al. [14] note that their approach is limited to graphs of bounded pathwidth. Similarly, we can approximately count various other objects such as q-dimensional p-matchings, q-set p-packings, graph motifs, and more:

Theorem 1.1. *The following problems admit deterministic* $4^{k+\mathcal{O}(\sqrt{k}(\log^2 k+\log^2 \frac{1}{\epsilon}))}n^{\mathcal{O}(1)}$*-time (resp. randomized* $4^{k+\mathcal{O}(\log^2 k)}(\frac{1}{\epsilon})^{\mathcal{O}(\log k)}n^{\mathcal{O}(1)}$*-time) FPT-ASs with polynomial space complexity:* (i) #Subgraph Isomorphism *for k-vertex subgraphs of treewidth $\mathcal{O}(1)$;* (ii) #q-Dimensional p-Matching *with $k = (q-1)p$;* (iii) #q-Set p-Packing *with $k = qp$;* (iv) #Graph Motif *and* #Module Motif *with $k = 2p$ where p is the motif size;* (v) #p-Internal Out-Branching *with $k = 2p$;* (vi) #Partial Cover *for k-element solutions.*[1]

Towards the design of our algorithms, our first conceptual contribution is the introduction of the notion of an *approximate parsimonious splitter*. While a randomized construction of such an object is simple, we do not know how (or whether it is even possible) to compute it deterministically within the size and time bounds that we require. We believe that this gap in knowledge of derandomization is the main reason why, for close to a decade, no progress has been made upon the result by Alon and Gutner [4, 3]. Here, our second conceptual contribution comes into play. We show that for recursive procedures, a weaker object that can only split so-called nice sets suffices, since the recursion itself can keep track on the "niceness" of sets. We believe that both the concept of approximate parsimonious splitters as well as our approach of how to weaken a randomized object (to efficiently compute it deterministically) at the cost of simple bookkeeping might find further applications in the future.

Related Work. The algorithms by Alon et al. [2] and Alon and Gutner [4, 3], just like our algorithms, extend to approximate counting of graphs of bounded treewidth. (This remark is also made by Alon and Gutner [4, 3].) In what follows, we briefly review works related to exact counting and decision from the viewpoint of Parameterized Complexity. Since these topics are not the focus of our work, the survey is illustrative rather than comprehensive.

The problem of counting the number of subgraphs of a graph G that are isomorphic to a graph H—that is, #Subgraph Isomorphism with Pattern H—admits a dichotomy: If the vertex cover number of H is bounded, then it is FPT [42], and otherwise it is #W[1]-hard [18]. The #W[1]-hardness of #k-Path, originally shown by Flum and Grohe [21], follows from this

[1]For problems (i) and (iv), the basis 4 is replaced by the basis 4.001 (or, more precisely, $4 + \delta$ for any fixed constant $\delta > 0$).

Table 1: State-of-the-art of $\#k$-Path and k-Path.

Ref.	Time	Counting	Deterministic	Poly. Space	Extension
[16]	$4^{k+o(k)}n^{\mathcal{O}(1)}$	No	Yes	Yes	Treewidth $\mathcal{O}(1)$
[40]	$2.554^k n^{\mathcal{O}(1)}$	No	Yes	No	Treewidth $\mathcal{O}(1)$
[41]	$2^k n^{\mathcal{O}(1)}$	No	No	Yes	Treewidth $\mathcal{O}(1)$
[11]	$1.657^k n^{\mathcal{O}(1)}$	No	No	Yes	No Extension
[3]	$(2e)^{k+o(k)}n^{\mathcal{O}(1)}$	Yes	Yes	No	Treewidth $\mathcal{O}(1)$
[14]	$4^k n^{\mathcal{O}(1)}$	Yes	No	No	Pathwidth $\mathcal{O}(1)$
This Paper	$4^{k+o(k)}n^{\mathcal{O}(1)}$	Yes	Yes	Yes	Treewidth $\mathcal{O}(1)$

dichotomy. By using the "meet in the middle" approach, the $\#k$-Path problem and, more generally, #Subgraph Isomorphism with Pattern H where H has bounded *pathwidth* and k vertices, was shown to admit an $n^{\frac{k}{2}+\mathcal{O}(1)}$-time algorithm [10]. Later, Björklund et al. [13] showed that $\frac{k}{2}$ is not a barrier (which was considered to be the case at that time) by designing an $n^{0.455k+\mathcal{O}(1)}$-time algorithm. Recently, a breakthrough that resulted in substantially faster running times took place: Curticapean et al. [17] showed that #Subgraph Isomorphism with Pattern H is solvable in time $\ell^{\mathcal{O}(\ell)}n^{0.174\ell}$ where ℓ is the number of edges in H; in particular, this algorithm solves $\#k$-Path in time $k^{\mathcal{O}(k)}n^{0.174k}$.

The k-Path problem (on both directed and undirected graphs) is among the most extensively studied parameterized problems [19, 24]. After a long sequence of works in the past three decades, the current best known parameterized algorithms for k-Path have running times $1.657^k n^{\mathcal{O}(1)}$ (randomized, polynomial space, undirected only) [11, 9] (extended in [12]), $2^k n^{\mathcal{O}(1)}$ (randomized, polynomial space) [41], $2.597^k n^{\mathcal{O}(1)}$ (deterministic, exponential space) [43, 23, 37], and $4^{k+o(k)}n^{\mathcal{O}(1)}$ (deterministic, polynomial space) [16]. The $1.657^k n^{\mathcal{O}(1)}$-time algorithm of Björklund et al. [11, 9] crucially relies on the symmetric structure of undirected k-paths. However, all other algorithms above directly extend to the detection of subgraphs of bounded treewidth. In particular, if the running time of the algorithm is $c^k n^{\mathcal{O}(1)}$, then the running time of the extension is $c^k n^{t+\mathcal{O}(1)}$ where t is the treewidth of the sought graph. To ensure that the constant c remains essentially the same[2] when dealing with the two deterministic algorithms (of [43, 23, 37] and [16]), the "division into small trees" trick by Fomin et al. [23] can be used; for the randomized algorithm (of [41]), no trick is required. For the sake of clarity, we explain how this is done in our case in Appendix A.

2 Preliminaries

For the sake of readability, we ignore ceiling and floor signs. Given a graph G, we let $V(G)$ and $E(G)$ denote the vertex set and edge set of G, respectively. For a positive integer k, a *k-path* in G is a (simple) path on k vertices in G; in case G is directed, the path is directed as well. We let $n = |V(G)|$ and $m = |E(G)|$. For a subset $U \subseteq V(G)$, $G[U]$ denotes the subgraph of G induced by U, $G - U = G[V(G) \setminus U]$, and $N_G(U)$ denoted the open neighborhood of U in G. We extend these notations to also consider sets U that contain entities that do not belong to $V(G)$—for such sets U, $G[U]$ denotes the subgraph of G induced by $U \cap V(G)$, and still $G - U = G[V(G) \setminus U]$. Given $u, v \in V(G)$ and $k \in \mathbb{N}$, let $\mathcal{P}_{u,v}^{G,k}$ denote the set of k-paths in G with endpoints u and v.

[2]More precisely, the constant c becomes a new constant c' that can be made arbitrarily close to c (at the cost of a higher degree of the polynomial factor).

Treewidth is a measure of how "treelike" is a graph, which is formally defined as follows. Here, if the given graph is directed, we refer to its underlying undirected graph.

Definition 2.1 (Treewidth)**.** *A* tree decomposition *of a graph G is a pair (T, β) of a tree T and a function $\beta : V(T) \to 2^{V(G)}$, such that*

1. *for any edge $\{x, y\} \in E(G)$ there exists a node $v \in V(T)$ such that $x, y \in \beta(v)$, and*
2. *for any vertex $x \in V(G)$, the subgraph of T induced by the set $T_x = \{v \in V(T) : x \in \beta(v)\}$ is a non-empty tree.*

The width *of (T, β) is $\max_{v \in V(T)}\{|\beta(v)|\} - 1$. The* treewidth *of G is the minimum width over all tree decompositions of G.*

Given a tree decomposition (T, β) of a graph G, for every $v \in V(T)$, the set $\beta(v)$ is called the *bag* of v. A *path decomposition* of a graph G is a tree decomposition (T, β) of G where T is restricted to be a path. Accordingly, the *pathwidth* of G is the minimum width over all path decompositions of G. We define the treewidth of a directed graph as the treewidth of its underlying undirected graph.

For a function $f : A \to B$ and subsets $A' \subseteq A$ and $B' \subseteq B$, define $f(A') = \{f(a) : a \in A'\}$ and $f^{-1}(B') = \{a \in A : f(a) \in B'\}$. For two functions $f : A \to B$ and $g : B \to C$, the notation $g \circ f : A \to C$ refers to function composition. For two tuples $X = (x_1, x_2, \ldots, x_p)$ and $Y = (y_1, y_2, \ldots, y_q)$, denote their concatenation by $X + Y = (x_1, x_2, \ldots, x_p, y_1, y_2, \ldots, y_q)$. By standard Chernoff bounds, we have the following bounds.

Proposition 2.1 ([34])**.** *Let $X_1, \ldots, X_n$ be independent random variables, each assigned a value in $\{0, 1\}$. Let $X = \sum_{i=1}^{n} X_i$, and let $\mu = E[X]$ denote the expected value of X. Then, for any $0 \leq \delta \leq 1$, it holds that* (i) $\Pr(X \leq (1-\delta)\mu) \leq e^{-\frac{\delta^2\mu}{2}}$*, and* (ii) $\Pr(X \geq (1+\delta)\mu) \leq e^{-\frac{\delta^2\mu}{3}}$.

Universal Families. For any $k \in \mathbb{N}$, a *k-set* is a set of size k. Given a universe U, denote $\binom{U}{k} = \{S \subseteq U : |S| = k\}$. Given a family $\mathscr{F}$ over U and two subsets $A, B \subseteq U$, denote $\mathscr{F}[A, B] = \{F \in \mathscr{F} : A \subseteq F, B \cap F = \emptyset\}$. Next, we present the definition of a universal family.

Definition 2.2 (Universal Family [35, 23])**.** *Let $n, p, q \in \mathbb{N}$. A family $\mathscr{F}$ of sets over a universe U of size n is an* (n, p, q)-universal family *if for each pair of disjoint sets $A \in \binom{U}{p}$ and $B \in \binom{U}{q}$, there is a set $F \in \mathscr{F}$ that contains A and is disjoint from B, that is, $\mathscr{F}[A, B] \neq \emptyset$.*

In the classic setting by Naor et al. [35], $p = q$. However, as shown by Fomin et al. [23], cases where $p \neq q$ are also of interest. Specifically, the following well-known proposition asserts that small representative families can be computed efficiently.

Proposition 2.2 ([35, 23])**.** *Let $n, p, q \in \mathbb{N}$, and $k = p + q$. Let U be a universe of size n. Then, an (n, p, q)-universal family $\mathscr{F}$ of sets over U of size $\mathcal{O}(\binom{k}{p} \log n)$ can be computed with success probability $1 - 1/n$ in time $\mathcal{O}(\binom{k}{p} n \log n)$. Additionally, an (n, p, q)-universal family $\mathscr{F}$ of sets over U of size $\binom{k}{p} 2^{o(k)} \log n$ can be computed (deterministically) in time $\binom{k}{p} 2^{o(k)} n \log n$. Both computations can enumerate the sets in $\mathscr{F}$ with polynomial delay.*

Observe that the constructions above are essentially optimal since any (n, p, q)-universal family must be of size at least $\binom{k}{p}$. We later define a notion of "approximate parsimonious universal families" that extends Definition 2.2 to be approximately parsimonious (so that, for all A and B, $|\mathscr{F}[A, B]|$ will be roughly the same), and present a computation for approximate parsimonious universal families.

3 Approx. Parsimonious Universal Family: Randomized Construction

For any pair of disjoint sets $A \in \binom{U}{p}$ and $B \in \binom{U}{q}$, Definition 2.2 guarantees that $\mathscr{F}[A,B] \neq \emptyset$. However, the number of sets in $\mathscr{F}[A,B]$ can be arbitrary. In our applications, the number of sets in $\mathscr{F}[A,B]$ will be tightly linked to the number of solutions whose "first half" is in A and whose "second half" is in B; thus, to avoid over-counting some solutions, we need all families $\mathscr{F}[\cdot,\cdot]$ to be *roughly* of the same size. For this purpose, let us first extend Definition 2.2 to be approximately parsimonious.

Definition 3.1 (δ-Parsimonious Universal Family)**.** *Let $n,p,q \in \mathbb{N}$ and $0 < \delta < 1$. Denote $k = p+q$. A family $\mathscr{F}$ of sets over a universe U of size (at most) n is a* δ-parsimonious (n,p,q)-universal family *if there exists $T = T(n,p,q,\delta) > 0$ such that for each pair of disjoint sets $A \in \binom{U}{p}$ and $B \in \binom{U}{q}$, it holds that $(1-\delta)\cdot T \leq |\mathscr{F}[A,B]| \leq (1+\delta)\cdot T$.*

We call the value T above a *correction factor*, and suppose it to be given along with the family $\mathscr{F}$. Our randomized computation of a δ-parsimonious (n,p,q)-universal family is based on the probabilistic method, inspired by [35, 23]. Specifically, we prove the following.

Theorem 3.1. *Let $n,p,q \in \mathbb{N}$ and $0 < \delta < 1$, and denote $k = p+q$. Let U be a universe of size n. A δ-parsimonious (n,p,q)-universal family $\mathscr{F}$ of sets over U of size $t = \mathcal{O}\left(\frac{k^k}{p^p q^q} \cdot k \log n \cdot \frac{1}{\delta^2}\right)$,*[3] *can be computed with success probability at least $1 - 1/n^{100k}$ in time $\mathcal{O}(t \cdot n)$. In particular, the sets in $\mathscr{F}$ can be enumerated with delay $\mathcal{O}(n)$.*

We note that the choice of 100 is arbitrary; it can be replaced by the choice of any fixed constant c. Crucially, we gain the extra property of being δ-parsimonious while essentially having the same time complexity and upper bound on the size of the output as in the non-parsimonious construction. We note that our computation is the first proof that approximate parsimonious of small size universal families *exist*.

Towards the proof of Theorem 3.1, we state a simple observation by [23]. For the sake of completeness, we also present the proof.

Observation 3.1. *Let $n,p,q \in \mathbb{N}$, and denote $k = p+q$. Let U be a universe of size n. In addition, let $A \in \binom{U}{p}$ and $B \in \binom{U}{q}$ be two disjoint subsets of U of sizes p and q, respectively. Then, for a set $F \subseteq U$ constructed by inserting (independently at random) each element $u \in U$ into F with probability p/k, the probability that $A \subseteq F$ and $F \cap B = \emptyset$ is $\frac{p^p q^q}{k^k}$.*

Proof. For every element $u \in A$, the probability that u is inserted into F is p/k. Moreover, for every element $u \in B$, the probability that u is not inserted into F is $1-(p/k) = q/k$. Since $|A| = p$ and $|B| = q$, the probability that $A \subseteq F$ and $F \cap B = \emptyset$ is $(p/k)^p(q/k)^q$. Because $(p/k)^p(q/k)^q = \frac{p^p q^q}{k^k}$, the observation follows. □

We are now ready to prove Theorem 3.1.

Proof of Theorem 3.1. Denote $T = 3(100k+k+1)\ln n \cdot \frac{1}{\delta^2}$. For $t = \frac{k^k}{p^p q^q} \cdot T$, we construct a family $\mathscr{F} = \{F_1, F_2, \ldots, F_t\}$ as follows. For $i = 1,2,\ldots,t$, we construct the set F_i by inserting each element $u \in U$ into F with probability p/k. Distinct elements are inserted (or not) into F_i independently, and the construction of the different sets in $\mathscr{F}$ is also independent. This

[3] Note that as $p+q=k$, the value $\frac{k^k}{p^p q^q}$ is upper bounded by 2^k rather than being of the magnitude of k^k.

completes the construction. Clearly, the sets in $\mathscr{F}$ can be enumerated with delay $\mathcal{O}(n)$, and the total time spent is $\mathcal{O}(t \cdot n)$.

In what follows, we show that with success probability at least $1 - 1/n^{100k}$, $\mathscr{F}$ is a δ-parsimonious (n, p, q)-universal family. To this end, choose (arbitrarily) two disjoint sets $A \in \binom{U}{p}$ and $B \in \binom{U}{q}$. By Observation 3.1, for any $i \in \{1, 2, \ldots, t\}$, the probability that $A \subseteq F_i$ and $F_i \cap B = \emptyset$ is $p^p q^q / k^k$. Since the construction of the different sets in $\mathscr{F}$ is also independent, the linearity of expectation implies that the expected number of sets in $\mathscr{F}$ that contain A and are disjoint from B is $t \cdot \frac{p^p q^q}{k^k} = T$, that is, $E[|\mathscr{F}[A, B]|] = T$. By Proposition 2.1, we have that

$$\Pr(|\mathscr{F}[A, B]| \leq (1 - \delta)T) \leq e^{-\frac{\delta^2 T}{2}} \leq n^{-(100k+k+1)}$$
$$\Pr(|\mathscr{F}[A, B]| \geq (1 + \delta)T) \leq e^{-\frac{\delta^2 T}{3}} = n^{-(100k+k+1)}.$$

Thus, the probability that either $|\mathscr{F}[A, B]| \leq (1-\delta)T$ or $|\mathscr{F}[A, B]| \geq (1+\delta)T$ is upper bounded by $2n^{-(100k+k+1)} \leq n^{-(100k+k)}$. For the last inequality, we implicitly assumed that $2/n \leq 1$, since otherwise the proof is trivial.

Recall that the choice of A and B above was arbitrary. This means that by union bound, the probability that there exist disjoint sets $A \in \binom{U}{p}$ and $B \in \binom{U}{q}$ such that either $|\mathscr{F}[A, B]| \leq (1 - \delta)T$ or $|\mathscr{F}[A, B]| \geq (1 + \delta)T$ is upper bounded by

$$\binom{n}{p} \cdot \binom{n}{q} \cdot n^{-(100k+k)} < n^k \cdot n^{-(100k+k)} = 1/n^{100k}.$$

Thus, the success probability is at least $1 - 1/n^{100k}$ as required. This completes the proof. □

4 Randomized Algorithms for #k-Path

For the sake of simplicity, we suppose that k is a power of 2. Otherwise, it is required to use ceiling and floor notations in appropriate places, which obfuscate the proofs.

4.1 Warm Up Application: Simple Randomized FPT-AS for #k-Path

Before we delve into more technical and less intuitive definitions related to our deterministic construction, we find it important to understand the relation between Definition 3.1 and #k-Path. For this purpose, we present a simple randomized polynomial-space FPT-AS for #k-Path. The dependency of the time complexity on n is made almost linear in Section 4.2). While the improved algorithm is still short and simple, it is somewhat less intuitive and hence presented separately later. For the sake of illustration, suppose that G is undirected.

Algorithm. Let $\xi = \ln(1 + \epsilon)/(k - 1)$. Our algorithm, denoted by $\mathcal{A}$, is recursive. Each call to $\mathcal{A}$ is of the form $\mathcal{A}(G', k')$ where G' is an induced subgraph of G and $k' \in \{1, \ldots, k\}$. For all $u, v \in V(G')$, the call $\mathcal{A}(G', k')$ should output an integer $a_{u,v}$ that approximates the number of k'-paths with endpoints u and v in G'. The initial call to the algorithm is with $G' = G$ and $k' = k$, and the final output is $(\sum_{u,v \in V(G)} a_{u,v})/2$.

We turn to describe a call $\mathcal{A}(G', k')$. In the basis, where $k' = 1$, we return $a_{v,v} = 1$ for all $v \in V(G')$, and $a_{u,v} = 0$ for all $u, v \in V(G')$ (with $u \neq v$).

Now, suppose that $k' \geq 2$. By Theorem 3.1, for a ξ-parsimonious $(n, k'/2, k'/2)$-universal family $\mathscr{F}$ of sets over $V(G)$, we can enumerate the sets $F \in \mathscr{F}$ with delay $\mathcal{O}(n)$. For each set $F \in \mathscr{F}$, we proceed as follows. We first perform two recursive calls: *(i)* we call $\mathcal{A}$ with $(G'[F], k'/2)$; *(ii)* we call $\mathcal{A}$ with $(G' - F, k'/2)$. For any $u, v \in F \cap V(G')$, let $b^F_{u,v}$ denote

the number returned by the first call. Similarly, for any $u, v \in V(G') \setminus F$, let $c^F_{u,v}$ denote the number returned by the second call. Then, for all $u \in F$ and $v \in V(G') \setminus F$, define $a^F_{u,v} = \sum_{\substack{\{p,q\} \in E(G') \\ \text{s.t. } p \in F, q \notin F}} b^F_{u,p} \cdot c^F_{q,v}$.

Let T be the correction factor of $\mathscr{F}$. After all sets $F \in \mathscr{F}$ were enumerated, for all $u, v \in V(G')$, we output $a_{u,v}$ calculated as follows: $a_{u,v} = \frac{1}{T} \cdot \sum_{\substack{F \in \mathscr{F} \\ \text{s.t. } u \in F, v \notin F}} a^F_{u,v}$. Note that we do not store all the values $a^F_{u,v}$ simultaneously, but we merely store one such value at a time and delete it immediately after $a^F_{u,v}/T$ is added. This completes the description of $\mathcal{A}$. The pseudocode of the algorithm is given in Algorithm 1.

Algorithm 1 Simple randomized FPT-AS for #k-PATH.

```
1: let ξ = ln(1 + ε)/(k − 1);
2: call SimpleRand(G, k): let a_{u,v} be the output for u, v ∈ V(G);
3: return Σ_{u,v∈V(G)} a_{u,v}/2;
4: function SimpleRand(G', k')
5:     if k' = 1 then
6:         return a_{v,v} = 1 for v ∈ V(G'), and a_{v,u} = 0 for u, v ∈ V(G') where u ≠ v;
7:     end if
8:     initialize a_{u,v} = 0 for u, v ∈ V(G');
9:     let I be an iterator of an ξ-parsimonious (n, k'/2, k'/2)-universal family F over V(G)
       with correction factor T, given by Theorem 3.1;
10:    for all F ∈ F (iterate with I) do
11:        call SimpleRand(G'[F], k'/2): let b^F_{u,v} be the output for u, v ∈ F ∩ V(G');
12:        call SimpleRand(G' − F, k'/2): let c^F_{u,v} be the output for u, v ∈ V(G') \ F;
13:        let a^F_{u,v} = Σ_{{p,q}∈E(G'): p∈F, q∉F} b^F_{u,p} · c^F_{q,v} for u ∈ F and v ∈ V(G') \ F;
14:        update a_{u,v} = a_{u,v} + a^F_{u,v}/T for u ∈ F and v ∈ V(G') \ F;
15:    end for
16:    return a_{u,v} for u, v ∈ V(G');
17: end function
```

Analysis. The main part of the analysis is done in the proof of the following lemma.

Lemma 4.1. *For some fixed constant $\eta > 0$, any call $\mathcal{A}(G', k')$ has polynomial space complexity and running time $\eta^{\log k'} 4^{k'} k'^{\log k'} (\log n)^{\log k'} mn^2 (\frac{1}{\xi^2})^{\log k'}$. Additionally, if all constructions of approximate universal families were successful, then for all $u, v \in V(G')$, the number $a_{u,v}$ returned by $\mathcal{A}(G', k')$ satisfies $(1-\xi)^{k'-1} x^{G',k'}_{u,v} \leq a_{u,v} \leq (1+\xi)^{k'-1} x^{G',k'}_{u,v}$ where $x^{G',k'}_{u,v} = |\mathcal{P}^{G',k'}_{u,v}|$.*

Proof. Let λ be a fixed constant that bounds from above those hidden by the $\mathcal{O}$-notations in Theorem 3.1. Let η be a fixed constant such that SimpleRand(G'', k'') for any G'' and k'' it may be called with, with the exception of the recursive calls that it makes, can be executed in time $\tau \cdot |\mathscr{F}| \cdot mn^2$.[4] We remark that the dependency on n and m is such due to Statement (i.e., Line) 13 in the pseudocode. We choose $\eta = 10 \max\{\lambda, \tau\}$.[5]

[4] This bound does not depend on G'' and k'', which is why we use n (being $|V(G)|$) and m (being $|E(G)|$) rather than $|V(G')|$ and $|E(G')|$.

[5] We choose 10 just because it is a large enough constant so that the last inequality that concerns time complexity holds.

Let $k' = k/2^d$. The proof is by backwards induction of d. In the basis ($k' = 1$), the claim is trivial. Now, let $d \leq \log_2 k - 1$, and suppose that the claim holds for $d+1$. Clearly, $\mathcal{A}(G', k')$ has a polynomial space complexity. By Theorem 3.1 and the choice of λ, we have that

$$|\mathscr{F}| \leq \lambda \cdot \frac{{k'}^{k'}}{(k'/2)^{k'/2}(k'/2)^{k'/2}} \cdot k' \log n \cdot \frac{1}{\xi^2} = \lambda \cdot 2^{k'} \cdot k' \log n \cdot \frac{1}{\xi^2}.$$

Moreover, by the inductive hypothesis, for a fixed constant $\tau > 0$, the running time of $\mathcal{A}(G', k')$ is upper bounded by $|\mathscr{F}| \cdot \left(2 \cdot \eta^{\log \frac{k'}{2}} 4^{\frac{k'}{2}} (\frac{k'}{2})^{\log \frac{k'}{2}} (\log n)^{\log \frac{k'}{2}} mn^2 (\frac{1}{\xi^2})^{\log \frac{k'}{2}} + \tau mn^2\right)$. Note that τ is independent of η. Because $\eta = 10 \max\{\lambda, \tau\}$, this means that the running time of $\mathcal{A}(G', k')$ is upper bounded by

$$\begin{aligned}
& |\mathscr{F}| \cdot \left(2 \cdot \eta^{\log \frac{k'}{2}} 4^{\frac{k'}{2}} (\frac{k'}{2})^{\log \frac{k'}{2}} (\log n)^{\log \frac{k'}{2}} mn^2 (\frac{1}{\xi^2})^{\log \frac{k'}{2}} + \tau mn^2\right) \\
\leq \; & \frac{\eta}{10} 2^{k'} k' \log n \frac{1}{\xi^2} \cdot \left(2 \cdot \eta^{\log k' - 1} 2^{k'} {k'}^{\log k' - 1} (\log n)^{\log k' - 1} mn^2 (\frac{1}{\xi^2})^{\log k' - 1} + \frac{\eta}{10} mn^2\right) \\
\leq \; & \eta^{\log k'} 4^{k'} {k'}^{\log k'} (\log n)^{\log k'} mn^2 (\frac{1}{\xi^2})^{\log k'}.
\end{aligned}$$

This completes the proof of the first item of the claim.

Towards the proof of the second item of the claim, suppose that all constructions of approximate universal families were successful, and consider some $u, v \in V(G')$. By the inductive hypothesis, we have that

$$\begin{aligned}
a_{u,v} &= \frac{1}{T} \cdot \sum_{\substack{F \in \mathscr{F} \\ \text{s.t. } u \in F, v \notin F}} a^F_{u,v} = \frac{1}{T} \cdot \sum_{\substack{F \in \mathscr{F} \\ \text{s.t. } u \in F, v \notin F}} \left(\sum_{\substack{\{p,q\} \in E(G') \\ \text{s.t. } p \in F, q \notin F}} b^F_{u,p} \cdot c^F_{q,v} \right) \\
&\leq \frac{1}{T} \cdot \sum_{\substack{F \in \mathscr{F} \\ \text{s.t. } u \in F, v \notin F}} \left(\sum_{\substack{\{p,q\} \in E(G') \\ \text{s.t. } p \in F, q \notin F}} (1+\xi)^{\frac{k'}{2}-1} x^{G'[F], \frac{k'}{2}}_{u,p} \cdot (1+\xi)^{\frac{k'}{2}-1} x^{G'-F, \frac{k'}{2}}_{q,v} \right) \\
&= \frac{1}{T} \cdot (1+\xi)^{k'-2} \cdot \sum_{\substack{F \in \mathscr{F} \\ \text{s.t. } u \in F, v \notin F}} \left(\sum_{\substack{\{p,q\} \in E(G') \\ \text{s.t. } p \in F, q \notin F}} x^{G'[F], \frac{k'}{2}}_{u,p} \cdot x^{G'-F, \frac{k'}{2}}_{q,v} \right)
\end{aligned}$$

For any subset $F \subseteq V(G')$, let $\mathcal{P}^{G',k'}_{u,v}[F]$ denote the set of paths $P \in \mathcal{P}^{G',k'}_{u,v}$ such that $F \in \mathscr{F}[A, B]$ where A (resp. B) is the set of $k'/2$ vertices on P closest to u (resp. v) including u (resp. v). Thus, the last expression above is equal to $(1+\xi)^{k'-2} \cdot \frac{\sum_{F \in \mathscr{F}} |\mathcal{P}^{G',k'}_{u,v}[F]|}{T}$, which implies that

$$a_{u,v} \leq (1+\xi)^{k'-2} \cdot \frac{\sum_{F \in \mathscr{F}} |\mathcal{P}^{G',k'}_{u,v}[F]|}{T}.$$

Since $\mathscr{F}$ is an ξ-parsimonious $(n, k'/2, k'/2)$-universal family, for any path $P \in \mathcal{P}^{G',k'}_{u,v}$ it holds that the number of sets $F \in \mathscr{F}$ such that $P \in \mathcal{P}^{G',k'}_{u,v}[F]$ is upper bounded by $(1+\xi)T$. Thus,

$$a_{u,v} \leq (1+\xi)^{k'-2} \cdot \frac{(1+\xi)T|\mathcal{P}^{G',k'}_{u,v}|}{T} = (1+\xi)^{k'-1} x^{G',k'}_{u,v}.$$

Symmetrically, we derive that $(1-\xi)^{k'-1} x^{G',k'}_{u,v} \leq a_{u,v}$. This completes the proof. □

We now conclude the following theorem.

Theorem 4.1. *There is a randomized* $(4^{k+o(k)}mn^2+mn^{2+o(1)})(\frac{1}{\epsilon})^{\mathcal{O}(\log k)}$*-time polynomial-space algorithm that, given a graph* G*, a positive integer* k *and an accuracy value* $0<\epsilon<1$*, outputs a number* y *that (with high probability, say, at least* $9/10$*) satisfies* $(1-\epsilon)x \leq y \leq (1+\epsilon)x$ *where* x *is the number of* k*-paths in* G*. In particular, if* $\frac{1}{\epsilon} = 2^{o(k/\log k)}$*, then the running time is* $4^{k+o(k)}mn^2 + mn^{2+o(1)}$*.*

Proof. By Lemma 4.1 with $G' = G$ and $k' = k$, we know that the total running time of $\mathcal{A}(G,k)$ is bounded by $4^{k+\mathcal{O}(\log^2 k)}(\log n)^{\log k}mn^2(\frac{1}{\xi})^{\log k}$ and uses polynomial space. Additionally, if all constructions of approximate universal families were successful, then for all $u,v \in V(G)$, the number $a_{u,v}$ computed by $\mathcal{A}(G,k)$ satisfies $(1-\xi)^{k-1}x^{G,k}_{u,v} \leq a_{u,v} \leq (1+\xi)^{k-1}x^{G,k}_{u,v}$.

If $\log n \leq 2^{\sqrt{k}}$, then $(\log n)^{\log k} \leq 2^{o(k)}$. Otherwise, when $\log n > 2^{\sqrt{k}}$, it holds that $k < \log^2 \log n$. It follows that $4^{k+\mathcal{O}(\log^2 k)}(\log n)^{\log k} \leq 4^{\log^2 \log n+\mathcal{O}(\log\log\log n)}(\log n)^{2\log\log\log n} \leq n^{\mathcal{O}(\frac{\log^2\log n}{\log n})} \leq n^{o(1)}$. In addition, by Taylor series $\ln(1+x) = \sum_{n=1}^{\infty}(-1)^{n+1}\frac{x^n}{n}$, it follows that $\epsilon/2 \leq \epsilon - \epsilon^2/2 \leq \ln(1+\epsilon) \leq \epsilon$, which means that $(\frac{1}{\xi})^{\log k} = 2^{\mathcal{O}(\log^2 k)}(\frac{1}{\epsilon})^{\mathcal{O}(\log k)}$. Thus, $4^{k+\mathcal{O}(\log^2 k)}(\log n)^{\log k}mn^2(\frac{1}{\xi})^{\log k} = (4^{k+o(k)}mn^2 + mn^{2+o(1)})(\frac{1}{\epsilon})^{\mathcal{O}(\log k)}$.

We now claim that with high probability, all constructions of approximate universal families were successful. By Theorem 3.1, the probability that a single construction is successful is at least $1 - 1/n^{100k}$. Thus, the probability that all constructions are successful is at least $(1-1/n^{100k})^{\mu}$ where μ is the number of constructions. Clearly, the number of constructions is upper bounded by the running time of $\mathcal{A}$. In turn, we can assume w.l.o.g. that the upper bound proven on this running time is, in itself, upper bounded by n^k, since otherwise the problem can be solved exactly by brute force within it. Thus, $\mu \leq n^k$. From this, we know that the probability that all constructions are successful is at least $(1-1/n^{100k})^{n^k}$. As n grows larger, this value approaches 1. In particular, the success probability can be assumed to be at least $9/10$ (otherwise n is a fixed constant), which proves our claim.

Thus, we know that for all $u,v \in V(G)$, it holds that $(1-\xi)^{k-1}x^{G,k}_{u,v} \leq a_{u,v} \leq (1+\xi)^{k-1}x^{G,k}_{u,v}$. Substituting ξ by $\frac{\ln(1+\epsilon)}{k-1}$, we have that for all $u,v \in V(G)$, it holds that $(1-\ln(1+\epsilon))x^{G,k}_{u,v} \leq (1-\frac{\ln(1+\epsilon)}{k-1})^{k-1}x^{G,k}_{u,v} \leq a_{u,v} \leq (1+\frac{\ln(1+\epsilon)}{k-1})^{k-1}x^{G,k}_{u,v} \leq e^{\ln(1+\epsilon)}x^{G,k}_{u,v}$. Since $(1-\epsilon) \leq (1-\ln(1+\epsilon))$ and $e^{\ln(1+\epsilon)} = (1+\epsilon)$, we have that for all $u,v \in V(G)$, it holds that $(1-\epsilon)x^{G,k}_{u,v} \leq a_{u,v} \leq (1+\epsilon)x^{G,k}_{u,v}$. Thus,

$$y = \left(\sum_{u,v\in V(G)} a_{u,v}\right)/2 \leq \left(\sum_{u,v\in V(G)} (1+\epsilon)x^{G,k}_{u,v}\right)/2 = (1+\epsilon)\left(\sum_{u,v\in V(G)} x^{G,k}_{u,v}\right)/2 = (1+\epsilon)x.$$

Symmetrically, we obtain that $(1-\epsilon)x \leq y$. This completes the proof. □

4.2 Improved Randomized FPT-AS for #k-Path

As our improved randomized FPT-AS is less intuitive, we first discuss the intuition behind it. Here, in addition to G' and k', every call to the recursive algorithm $\mathcal{A}$ is given an assignment $\beta : V(G) \setminus V(G') \to \mathbb{N}_0$ of a non-negative integer to each vertex outside G'. Roughly speaking, for each vertex $v \in V(G) \setminus V(G')$, the value $\beta(v)$ is an approximation of the number of $\widehat{k}$-paths that end at v and are completely contained in $G - U$ for a certain integer $\widehat{k} \in \{1,2,\ldots,k-k'\}$ and a subset $U \subseteq V(G)$ that contains $V(G')$.[6] In particular, given that now the goal of each call is to output such an assignment for $G - (U \setminus V(G'))$ (a precise definition of the goal of a call is given in the formal description of the algorithm), we do not need to consider every pair of

[6]To be more precise, when we reach a recursive call, some part of the graph G and of k has already been processed, some part is to be processed in the current call, and some part is to be processed in future calls to be made by ancestor calls of the current one; then, U and $\widehat{k}$ are the vertex set and part of k already processed.

vertices $u, v \in V(G')$ and compute a value $a_{u,v}$; instead, we only compute one value per vertex. Additionally, recall that in the previous algorithm in order to compute $a_{u,v}$, we considered every edge $\{p, q\} \in E(G')$ while computing $a^F_{u,v}$ and hence divided our task into the computation of $k'/2$-paths between u and p in one recursive call and $k'/2$-paths between q and u in the other. Here, we do not store the two endpoints of paths, but their "middle". More precisely, the flow of information differs: to compute the assignment we need to output in the current call, we perform one recursive call to which the assignment β is given as input; this call will return an assignment that "handles" the first $\widehat{k} + k'/2$ vertices on the paths being counted, and be sent as input to the second recursive call to handle the next $k'/2$ vertices.

Algorithm. Let $\xi = \ln(1+\epsilon)/(k-1)$. We add a new vertex s to G and connect it to all vertices in G. Thus, rather than counting the number of k-paths in the former graph G, we can count the number of $(k+1)$-paths with s as an endpoint in the new graph G. In what follows, we focus on this goal.

Our algorithm, denoted by $\mathcal{A}$, is recursive. Each call to $\mathcal{A}$ is of the form $\mathcal{A}(G', k', \beta)$ where G' is an induced subgraph of G, $k' \in \{1, \ldots, k\}$, and $\beta : V(G) \setminus V(G') \to \mathbb{N}_0$. The call $\mathcal{A}(G', k', \beta)$ should output an assignment $\alpha : V(G') \to \mathbb{N}_0$ with the following property: For each vertex $v \in V(G')$, it holds that $\alpha(v)$ approximates the following number:

$$\sum_{\substack{\{p,q\} \in E(G) \\ \text{s.t. } p \notin V(G'), q \in V(G')}} \beta(p) \cdot x^{G',k'}_{q,v},$$

where $x^{G',k'}_{q,v} = |\mathcal{P}^{G',k'}_{q,v}|$.

The initial call to the algorithm is with $G' = G - \{s\}$, $k' = k$, and $\beta(s) = 1$. The final output is $\sum_{v \in V(G) \setminus \{s\}} \alpha(v)$.

We turn to describe a call $\mathcal{A}(G', k', \beta)$. In the basis, where $k' = 1$, we return an assignment $\alpha : V(G') \to \mathbb{N}_0$ defined as follows: For each vertex $v \in V(G')$, define

$$\alpha(v) = \sum_{\substack{u \notin V(G') \\ \text{s.t. } \{u,v\} \in E(G)}} \beta(u).$$

Intuitively, the meaning of this sum is to extend each path "considered by β" by one vertex, so the number of paths that end at v is the sum, over every neighbor u of v, of number of paths "considered by β" that end at u (and so it only makes sense to consider the case where $u \notin V(G')$).

Now, suppose that $k' \geq 2$. By Theorem 3.1, for a ξ-parsimonious $(n, k'/2, k'/2)$-universal family $\mathscr{F}$ of sets over $V(G)$, we can enumerate the sets $F \in \mathscr{F}$ with delay $\mathcal{O}(n)$. For each set $F \in \mathscr{F}$, we proceed as follows. We first recursively call $\mathcal{A}$ with $(G'[F], k'/2, \beta_F)$ where β_F is the extension of β that assigns 0 to every vertex in $V(G') \setminus F$. Let γ_F be the output of this call, and extend it to assign 0 to every vertex in $V(G) \setminus V(G')$. Then, we recursively call $\mathcal{A}$ with $(G' - F, k'/2, \gamma_F)$. Let α_F be the output of this recursive call.

Let T be the correction factor of $\mathscr{F}$. After all sets $F \in \mathscr{F}$ were enumerated, the output $\alpha : V(G') \to \mathbb{N}_0$ is computed as follows. For all $v \in V(G')$, we calculate

$$\alpha(v) = \left(\sum_{\substack{F \in \mathscr{F} \\ \text{s.t. } v \notin F}} \alpha_F(v) \right) / T.$$

Note that we do not store all the assignments α_F simultaneously, but we merely store one such assignment at a time and delete it immediately after $\alpha_F(v)/T$, for every $v \in V(G')$, is added. This completes the description of $\mathcal{A}$. The pseudocode is given in Algorithm 2.

Algorithm 2 Improved randomized FPT-AS for #k-PATH.

let $\xi = \ln(1+\epsilon)/(k-1)$;
add a new vertex s to G, and connect s to all vertices in G;
let $\beta : \{s\} \to \mathbb{N}_0$ assign 1 to s;
call ImprovedRand$(G - \{s\}, k+1, \beta)$: let α be the output;
return $\sum_{v \in V(G) \setminus \{s\}} \alpha(v)$;
function ImprovedRand(G', k', β)
    if $k' = 1$ **then**
        return $\alpha : V(G') \to \mathbb{N}_0$ where for $v \in V(G')$, $\alpha(v) = \sum_{\substack{u \notin V(G'): \\ \{u,v\} \in E(G)}} \beta(u)$;
    end if
    initialize $\alpha : V(G') \to \mathbb{N}_0$ to assign 0 to each $v \in V(G')$;
    let I be an iterator of an ξ-parsimonious $(n, k'/2, k'/2)$-universal family $\mathscr{F}$ over $V(G)$ with correction factor T, given by Theorem 3.1;
    for all $F \in \mathscr{F}$ (iterate with I) **do**
        let β_F be the extension of β that assigns 0 to each $v \in V(G') \setminus F$;
        call ImprovedRand$(G'[F], k'/2, \beta_F)$: let γ_F be the output;
        extend γ_F to assign 0 to each $v \in V(G) \setminus V(G')$;
        call ImprovedRand$(G' - F, k'/2, \gamma_F)$: let α_F be the output;
        update $\alpha(v) = \alpha(v) + \alpha_F(v)/T$ for $v \in V(G') \setminus F$;
    end for
    return α;
end function

Correctness. The proof of correctness of our algorithm roughly follows the same lines as the proof of correctness of Theorem 4.1. The main part of the analysis is done in the proof of the following lemma.

Lemma 4.2. *For some fixed constant $\eta > 0$, any call $\mathcal{A}(G', k', \beta)$ satisfies the conditions below.*

- *$\mathcal{A}(G', k', \beta)$ takes time $\eta^{\log k'} 4^{k'} k'^{\log k'} (\log n)^{\log k'} m(\frac{1}{\xi^2})^{\log k'}$ and polynomial space.*
- *If all constructions of approximate universal families were successful, then for all $v \in V(G')$, the number $\alpha(v)$ assigned to v by the assignment α returned by $\mathcal{A}(G', k', \beta)$ satisfies the following inequalities:*

$$(1-\xi)^{k'-1} \cdot \left(\sum_{\substack{\{p,q\} \in E(G) \\ \text{s.t. } p \notin V(G'), q \in V(G')}} \beta(p) \cdot x_{q,v}^{G',k'} \right) \leq \alpha(v) \leq (1+\xi)^{k'-1} \cdot \left(\sum_{\substack{\{p,q\} \in E(G) \\ \text{s.t. } p \notin V(G'), q \in V(G')}} \beta(p) \cdot x_{q,v}^{G',k'} \right),$$

where $x_{q,v}^{G',k'} = |\mathcal{P}_{q,v}^{G',k'}|$.

Proof. Let λ be a fixed constant that bounds from above those hidden by the $\mathcal{O}$-notations in Theorem 3.1. Let η be a fixed constant such that ImprovedRand(G'', k'') for any G'' and k'' it may be called with, with the exception of the recursive calls that it makes and the basis (that takes time $\mathcal{O}(m)$), can be executed in time $\tau \cdot |\mathscr{F}| \cdot n$.

Let $k' = k/2^d$. The proof is by backwards induction of d. In the basis, where $k' = 1$, the claim trivially holds.

Now, let $d \leq \log_2 k - 1$, and suppose that the claim holds for $d+1$. Clearly, $\mathcal{A}(G', k', \beta)$ has a polynomial space complexity. By Theorem 3.1,we have that

$$|\mathscr{F}| \leq \lambda \cdot 2^{k'} \cdot k' \log n \cdot \frac{1}{\xi^2}.$$

By the inductive hypothesis, the running time of $\mathcal{A}(G', k', \beta)$ is upper bounded by

$$|\mathscr{F}| \cdot \left(2 \cdot \eta^{\log \frac{k'}{2}} 4^{\frac{k'}{2}} \left(\frac{k'}{2}\right)^{\log \frac{k'}{2}} (\log n)^{\log \frac{k'}{2}} m \left(\frac{1}{\xi^2}\right)^{\log \frac{k'}{2}} + \tau n \right).$$

As in the proof of Theorem 4.2, because $\eta = 10 \max\{\lambda, \tau\}$, we obtain that the running time of $\mathcal{A}(G', k', \beta)$ is upper bounded by

$$\eta^{\log k'} 4^{k'} k'^{\log k'} (\log n)^{\log k'} m \left(\frac{1}{\xi^2}\right)^{\log k'}.$$

Towards the proof of the second item of the claim, suppose that all constructions of approximate universal families were successful, and consider some $v \in V(G')$. By the inductive hypothesis, we have that

$$\begin{aligned}
\alpha(v) &= \frac{1}{T} \cdot \sum_{\substack{F \in \mathscr{F} \\ \text{s.t. } v \notin F}} \alpha_F(v) \\
&\leq \frac{1}{T} \cdot \sum_{\substack{F \in \mathscr{F} \\ \text{s.t. } v \notin F}} \left((1+\xi)^{\frac{k'}{2}-1} \cdot \sum_{\substack{\{a,b\} \in E(G) \\ \text{s.t. } a \notin V(G'-F), b \in V(G'-F)}} \gamma_F(a) \cdot x_{b,v}^{G'-F, \frac{k'}{2}} \right) \\
&\leq (1+\xi)^{\frac{k'}{2}-1} \cdot \frac{1}{T} \cdot \sum_{\substack{F \in \mathscr{F} \\ \text{s.t. } v \notin F}} \left(\sum_{\substack{\{a,b\} \in E(G) \\ \text{s.t. } a \notin V(G'-F), b \in V(G'-F)}} \left((1+\xi)^{\frac{k'}{2}-1} \sum_{\substack{\{p,q\} \in E(G) \\ \text{s.t. } p \notin V(G'), q \in F}} \beta(p) \cdot x_{q,a}^{G'[F], \frac{k'}{2}} \right) \cdot x_{b,v}^{G'-F, \frac{k'}{2}} \right) \\
&\leq (1+\xi)^{k'-2} \cdot \frac{1}{T} \cdot \sum_{\substack{F \in \mathscr{F} \\ \text{s.t. } v \notin F}} \left(\sum_{\substack{\{p,q\} \in E(G) \\ \text{s.t. } p \notin V(G'), q \in F}} \sum_{\substack{\{a,b\} \in E(G) \\ \text{s.t. } a \notin V(G'-F), b \in V(G'-F)}} \beta(p) \cdot x_{q,a}^{G'[F], \frac{k'}{2}} \cdot x_{b,v}^{G'-F, \frac{k'}{2}} \right).
\end{aligned}$$

In addition, for any subset $F \subseteq V(G')$, let $\mathcal{P}_{q,v}^{G',k'}[F]$ denote the set of paths $P \in \mathcal{P}_{q,v}^{G',k'}$ such that $F \in \mathscr{F}[A, B]$ where A (resp. B) is the set of $k'/2$ vertices on P closest to q (resp. v) including q (resp. v). Thus, with the last expression above being equal to the one below (that bounds $\alpha(v)$ from above), we have that

$$\begin{aligned}
\alpha(v) &\leq (1+\xi)^{k'-2} \cdot \frac{1}{T} \cdot \sum_{\substack{F \in \mathscr{F} \\ \text{s.t. } v \notin F}} \left(\sum_{\substack{\{p,q\} \in E(G) \\ \text{s.t. } p \notin V(G'), q \in F}} \beta(p) \cdot |\mathcal{P}_{q,v}^{G',k'}[F]| \right) \\
&= (1+\xi)^{k'-2} \cdot \frac{1}{T} \cdot \sum_{\substack{\{p,q\} \in E(G) \\ \text{s.t. } p \notin V(G'), q \in V(G')}} \left(\beta(p) \cdot \sum_{\substack{F \in \mathscr{F} \\ \text{s.t. } v \notin F, q \in F}} |\mathcal{P}_{q,v}^{G',k'}[F]| \right).
\end{aligned}$$

Since $\mathscr{F}$ is a ξ-parsimonious $(n, k'/2, k'/2)$-universal family, for any vertex $q \in V(G')$ and path $P \in \mathcal{P}_{q,v}^{G',k'}$, the number of sets $F \in \mathscr{F}$ such that $P \in \mathcal{P}_{q,v}^{G',k'}[F]$ is upper bounded by $(1+\xi)T$. Thus, we have that

$$\begin{aligned}
\alpha(v) &\leq (1+\xi)^{k'-2} \cdot \frac{1}{T} \cdot \sum_{\substack{\{p,q\} \in E(G) \\ \text{s.t. } p \notin V(G'), q \in V(G')}} \beta(p) \cdot (1+\xi) T |\mathcal{P}_{q,v}^{G',k'}| \\
&= (1+\xi)^{k'-1} \cdot \left(\sum_{\substack{\{p,q\} \in E(G) \\ \text{s.t. } p \notin V(G'), q \in V(G')}} \beta(p) \cdot x_{q,v}^{G',k'} \right).
\end{aligned}$$

Symmetrically, we derive that $(1-\xi)^{k'-1} \cdot \left(\sum_{\substack{\{p,q\} \in E(G) \\ \text{s.t. } p \notin V(G'), q \in V(G')}} \beta(p) \cdot x_{q,v}^{G',k'} \right) \leq \alpha(v)$. This completes the proof. □

We now conclude the proof of Theorem 4.2.

Theorem 4.2. *There is a randomized $(4^{k+o(k)}m + mn^{o(1)})(\frac{1}{\epsilon})^{\mathcal{O}(\log k)}$-time polynomial-space algorithm that, given a graph G, a positive integer k and an accuracy value $0 < \epsilon < 1$, outputs a number y that (with high probability) satisfies $(1-\epsilon)x \leq y/2 \leq (1+\epsilon)x$ where x is the number of k-paths in G. In particular, if $\frac{1}{\epsilon} = 2^{o(k/\log k)}$, then the running time is $4^{k+o(k)}mn^{o(1)}$.*

Proof. For the sake of simplicity, we let G denote the graph obtained after the addition of s (rather than the input graph G). Then, it is sufficient to show the inequalities $(1-\epsilon)x \leq y \leq (1+\epsilon)x$ where x is the number of $(k+1)$-paths in G with s as an endpoint. (To see this, observe that the number of $(k+1)$-path in G with s as an endpoint is exactly twice the number of k-paths in the original graph.)

Let β_{init} be the assignment $\beta_{\text{init}} : \{s\} \to \mathbb{N}_0$ that satisfies $\beta_{\text{init}}(s) = 1$. Observe that for all $v \in V(G) \setminus \{s\}$, it holds that $\sum_{\substack{\{p,q\} \in E(G) \\ \text{s.t. } p \notin V(G-\{s\}), q \in V(G-\{s\})}} \beta_{\text{init}}(p) \cdot x_{q,v}^{G-\{s\},k}$ is simply equal to $x_{s,v}^{G,k+1}$. Thus, by Lemma 4.2 with $G' = G$, $k' = k$ and $\beta = \beta_{\text{init}}$, we know that

- $\mathcal{A}(G, k, \beta_{\text{init}})$ runs in time $4^{k+\mathcal{O}(\log^2 k)}(\log n)^{\log k} m (\frac{1}{\xi})^{\log k}$ and uses polynomial space.
- If all constructions of approximate universal families were successful, then for all $v \in V(G) \setminus \{s\}$, the number $\alpha(v)$ assigned to v by the assignment α returned by $\mathcal{A}(G, k, \beta_{\text{init}})$ satisfies $(1-\xi)^{k-1} x_{s,v}^{G,k+1} \leq \alpha(v) \leq (1+\xi)^{k-1} x_{s,v}^{G,k+1}$.

As in the proof of Theorem 4.1, $4^{k+\mathcal{O}(\log^2 k)}(\log n)^{\log k} m (\frac{1}{\xi})^{\log k} = (4^{k+o(k)}m + mn^{o(1)})(\frac{1}{\epsilon})^{\mathcal{O}(\log k)}$. Moreover, as in the proof of Theorem 4.1, with high probability (say, higher than 9/10), all constructions of approximate universal families were successful. Thus, we know that for all $v \in V(G) \setminus \{s\}$, it holds that $(1-\xi)^{k-1} x_{s,v}^{G,k+1} \leq \alpha(v) \leq (1+\xi)^{k-1} x_{s,v}^{G,k+1}$. As in the proof of Theorem 4.1, for all $v \in V(G) \setminus \{s\}$, it follows that $(1-\epsilon) x_{s,v}^{G,k+1} \leq \alpha(v) \leq (1+\epsilon) x_{s,v}^{G,k+1}$.

We thus obtain that

$$y = \sum_{v \in V(G) \setminus \{s\}} \alpha(v) \leq \sum_{v \in V(G) \setminus \{s\}} (1+\epsilon) x_{s,v}^{G,k+1} = (1+\epsilon) \sum_{v \in V(G) \setminus \{s\}} x_{s,v}^{G,k+1} = (1+\epsilon)x.$$

Symmetrically, we obtain that $(1-\epsilon)x \leq y$. This completes the proof. □

The time complexity bound above can be easily reduced to $4^{k+\mathcal{O}(\log^2 k)} m \log n (\frac{1}{\epsilon})^{\mathcal{O}(\log k)}$. Indeed, it is only required to utilize Proposition 5.2 to reduce the universe size from n to k^2 before calling the algorithm in Theorem 4.2; then, the term that gives rise to $n^{o(1)}$ above is subsumed by $4^{\mathcal{O}(\log^2 k)}$. Since this is precisely what our deterministic algorithm does to obtain such dependency, we do not repeat these details. Here, we directly restate Corollary 4.1.

Corollary 4.1. *There is a randomized $4^{k+\mathcal{O}(\log^2 k)} m \log n (\frac{1}{\epsilon})^{\mathcal{O}(\log k)}$-time polynomial-space algorithm that, given a graph G, a positive integer k and an accuracy value $0 < \epsilon < 1$,'outputs a number y that (with high probability) satisfies $(1-\epsilon)x \leq y/2 \leq (1+\epsilon)x$ where x is the number of k-paths in G. In particular, if $\frac{1}{\epsilon} = 2^{o(k/\log k)}$, then the running time is $4^{k+o(k)} m \log n$.*

5 Approx. Parsimonious Universal Family: Deterministic Construction

5.1 Definitions and Statements

We do not know how to deterministically construct small δ-parsimonious universal families. Indeed, the best construction that we are aware of is the one based on bipartite Paley graphs (see Theorem 11.9 in the book by Jukna [27] and the historical notes behind the result). This construction leads to families of size $4^{k+o(k)}$ for $p = q = \frac{k}{2}$, whereas we would like size $2^{k+o(k)}$. Instead, we provide an efficient deterministic computation of a small δ-parsimonious universal family that is suitable for handling so-called "nice pairs". The crucial point is that with respect to our applications, this relaxed construction suffices. In this section, we present the definition of this relaxation, its construction as well a main property that this definition satisfies (Lemma 5.2). To allow the reader to see the "big picture", we defer the proofs of the two lemmas and the theorem stated in this subsection to the next subsections.

To simplify the following definitions, we introduce the following notation. To see the intuition behind this notation in the context of applications, throughout this section h can be thought of as a function that reduces the size of the universe from n to z, by coloring each of the n elements of the universe in one of z colors. These z colors are then partitioned into t parts, and we indicate the parts of the partition as $f^{-1}(i)$, $i \in \{1, 2, \ldots, t\}$. We will pick f to partition the reduced universe evenly into parts of size k/t. For bookkeeping purposes, we write down the sizes of intersections of "partial solutions" with each part in a vector $\mathbf{p} = (\mathbf{p}(1), \ldots, \mathbf{p}(t))$, with $0 \leq \mathbf{p}(i) \leq k/t$.

Definition 5.1. *Let $n, p, q, t, z \in \mathbb{N}$, and $k = p + q$. Let U be a universe of size n. A t-dimensional vector $\mathbf{p} = (\mathbf{p}(1), \mathbf{p}(2), \ldots, \mathbf{p}(t))$ such that $\mathbf{p}(i) \in \{0, 1, \ldots, k/t\}$ for each $i \in \{1, 2, \ldots, t\}$ and $\sum_{i=1}^{t} p_i = p$ is called* (p, q, t)-compatible. *When $\mathbf{p}$ is clear from context, for each $i \in \{1, 2, \ldots, t\}$, denote $p_i = \mathbf{p}(i)$ and $q_i = (k/t) - p_i$.*

A triple $(h, f, \mathbf{p})$ is called (n, p, q, t, z)-compatible *if $h : U \to \{1, 2, \ldots, z\}$, $f : \{1, 2, \ldots, z\} \to \{1, 2, \ldots, t\}$, and $\mathbf{p}$ is (p, q, t)-compatible. (The universe U will be clear from context.)*

We begin by defining what is a nice pair. Intuitively, it is a pair of disjoint sets (A, B) whose union is reduced "properly" by h so it does not map two elements in the union to the same element in the reduced universe, and which "comply" with $\mathbf{p}$, where A is thought of as the set of elements we "currently use" to construct partial solutions, and B is thought as the set of elements we will "use next" to extend them.

Definition 5.2 (Nice Pair)**.** *Let $n, p, q, t, z \in \mathbb{N}$. Let U be a universe of size n. Let $(h, f, \mathbf{p})$ be (n, p, q, t, z)-compatible. A pair (A, B) is* nice *(with respect to $(h, f, \mathbf{p})$) if $A \in \binom{U}{p}$ and $B \in \binom{U}{q}$ are disjoint sets, and the following conditions hold.*

1. *The function h is injective when restricted to $A \cup B$.*
2. *For each $i \in \{1, 2, \ldots, t\}$, it holds that $|\{u \in A : f(h(u)) = i\}| = p_i$ and $|\{u \in B : f(h(u)) = i\}| = (k/t) - p_i$.*

Towards the definition of a δ-parsimonious universal family for nice pairs, we first present a weaker definition of this notion where we have a triple $(h, f, \mathbf{p})$ at hand. Essentially, this definition can be thought of as a restriction of the definition of a parsimonious universal family that works only for nice pairs.

Definition 5.3 (Specific δ-Parsimonious Universal Family for Nice Pairs)**.** *Let $n, p, q, t, z \in \mathbb{N}$. Let U be a universe of size n. Let $(h, f, \mathbf{p})$ be (n, p, q, t, z)-compatible. Let $0 < \delta < 1$. A family*

$\mathscr{F}$ of sets over $\{1,\ldots,z\}$ is a δ-parsimonious $(h,f,\mathbf{p})$-universal family (for nice pairs) *if there exists $T = T(h,f,\mathbf{p},\delta) > 0$ such that for every nice pair (A,B), it holds that $(1-\delta)\cdot T \leq |\mathscr{F}[h(A),h(B)]| \leq (1+\delta)\cdot T$.*

Before we show how to extend Definition 5.3 to the notion useful for applications, we argue that small δ-parsimonious $(h,f,\mathbf{p})$-universal families can be computed "efficiently".

Lemma 5.1. *Let $p,q,t,z \in \mathbb{N}$, and denote $k = p+q$ and $s = k/t$. Let $(h,f,\mathbf{p})$ be (n,p,q,t,z)-compatible. Let $0 < \delta < 1$. A δ-parsimonious $(h,f,\mathbf{p})$-universal family $\mathscr{F}$ of sets over $\{1,\ldots,z\}$ of size $\ell = \mathcal{O}\left(\binom{k}{p}\cdot(k\cdot\log z\cdot\frac{\mathcal{O}(1)}{\delta})^{2t}\right)$ can be computed in time $\ell\cdot z^{s+1}s^{\mathcal{O}(1)}t$. In particular, the sets in $\mathscr{F}$ can be enumerated with delay $z^{s+1}s^{\mathcal{O}(1)}t$.*

Towards the definition of our general construction, we need to present the definitions of a balanced splitter and a balanced hash family. Constructions of such a splitter and a family were given by Alon and Gutner [4, 3].

Definition 5.4 (Definition 2.2 [4])**.** *Suppose that $1 \leq \ell \leq k \leq n$ and $0 < \epsilon < 1$, and let $\mathscr{S}$ be a family of functions from $\{1,\ldots,n\}$ to $\{1,\ldots,\ell\}$. For a set $S \in \binom{\{1,\ldots,n\}}{k}$, let $\texttt{split}_{\mathscr{S}}(S)$ denote the number of functions $f \in \mathscr{S}$ that split $\mathscr{S}$ into equal size parts, that is, $|f^{-1}(i)\cap S| = k/\ell$. Then, $\mathscr{S}$ is an* ϵ-balanced (n,k,ℓ)-splitter *if there exists $T = T(n,k,\ell,\epsilon) > 0$ such that for every set $S \in \binom{\{1,\ldots,n\}}{k}$, we have $(1-\epsilon)T \leq \texttt{split}_{\mathscr{S}}(S) \leq (1-\epsilon)T$.*

Definition 5.5 (Definition 2.1 [4])**.** *Suppose that $1 \leq k \leq \ell \leq n$ and $0 < \epsilon < 1$. A family $\mathscr{H}$ of functions from $\{1,\ldots,n\}$ to $\{1,\ldots,\ell\}$ is an* (ϵ,k)-balanced family of hash functions *if there exists $T = T(n,k,\ell,\epsilon) > 0$ such that for every set $S \in \binom{\{1,\ldots,n\}}{k}$, the number of functions in $\mathscr{H}$ that are injective when restricted to S is between $(1-\epsilon)T$ and $(1+\epsilon)T$.*

We are now ready to define our general derandomization tool. Here, instead of considering a single triple $(h,f,\mathbf{p})$ as in Definition 5.3, we have two families to generate all such triples that are relevant to us, and store a parsimonious universal family with respect to each of them.

Definition 5.6 ((General) δ-Parsimonious Universal Family for Nice Pairs)**.** *Let $n,p,q \in \mathbb{N}$ and $0 < \delta < 1$, and denote $k = p+q$, $z = \frac{2k^2}{\epsilon}$, $t = \sqrt{k}$, $s = k/t = \sqrt{k}$, and $\epsilon = \delta/3$. Let U be a universe of size n. A* δ-parsimonious (n,p,q)-universal tuple (for nice pairs) *is a tuple $(\mathscr{H},\mathscr{S},\{\mathscr{F}^{h,f,\mathbf{p}}\}|_{h\in\mathscr{H},f\in\mathscr{S},\mathbf{p}})$, where the enumeration is over every (p,q,t)-compatible $\mathbf{p}$, that satisfies the following conditions.*

- *$\mathscr{H}$ is an (ϵ,k)-balanced family of hash functions from $\{1,\ldots,n\}$ to $\{1,\ldots,z\}$ (with correction factor $T_{\mathscr{H}}$).*
- *$\mathscr{S}$ is an ϵ-balanced (z,k,t)-splitter (with correction factor $T_{\mathscr{S}}$).*
- *For every hash function $h \in \mathscr{H}$, splitter $f \in \mathscr{S}$ and (p,q,t)-compatible function $\mathbf{p}$, it holds that $\mathscr{F}^{h,f,\mathbf{p}}$ is a δ-parsimonious $(h,f,\mathbf{p})$-universal family (with correction factor $T_{\mathbf{p}}$).*

We define the *collection of quadruples of* $(\mathscr{H},\mathscr{S},\{\mathscr{F}^{h,f,\mathbf{p}}\}|_{h\in\mathscr{H},f\in\mathscr{S},\mathbf{p}})$ as the collection of every quadruple $(h,f,\mathbf{p},F)$ such that $h \in \mathscr{H}$, $f \in \mathscr{S}$ and $F \in \mathscr{F}^{h,f,\mathbf{p}}$. By *enumerating the quadruples of* $(\mathscr{H},\mathscr{S},\{\mathscr{F}^{h,f,\mathbf{p}}\}|_{h\in\mathscr{H},f\in\mathscr{S},\mathbf{p}})$, we refer to the enumeration of every quadruple in this collection. We remark that below, for the sake of brevity, when we write $k,z,t,s,\epsilon,T_{\mathscr{H}},T_{\mathscr{S}}$ and $T_{\mathbf{p}}$, we refer to the notations given in Definition 5.6. Let us now state our construction.

Theorem 5.1. *Let $n,p,q \in \mathbb{N}$ and $0 < \delta < 1$. Denote $k = p+q$. Let U be a universe of size n. A δ-parsimonious (n,p,q)-universal tuple $(\mathscr{H},\mathscr{S},\{\mathscr{F}^{h,f,\mathbf{p}}\}|_{h\in\mathscr{H},f\in\mathscr{S},\mathbf{p}})$ whose collection of quadruples is of size ℓ can be computed in time $\frac{k^{\mathcal{O}(1)}n\log n}{\delta^{\mathcal{O}(1)}}+\ell\cdot\Delta$. In particular, after preprocessing*

time $\frac{k^{\mathcal{O}(1)} n \log n}{\delta^{\mathcal{O}(1)}}$*, the quadruples of* $(\mathscr{H}, \mathscr{S}, \{\mathscr{F}^{h,f,\mathbf{p}}\}|_{h\in\mathscr{H}, f\in\mathscr{S},\mathbf{p}})$ *can be enumerated with delay* Δ*. Here,*

$$\begin{aligned}\ell &= \binom{k}{p} \cdot 2^{\mathcal{O}(\sqrt{k}(\log^2 k + \log^2 \frac{1}{\delta}))} \cdot \log n, \text{ and}\\ \Delta &= 2^{\mathcal{O}(\sqrt{k}(\log k + \log \frac{1}{\delta}))}.\end{aligned}$$

In order to state the property of a δ-parsimonious (n,p,q)-universal tuple that makes it useful for applications, we need one last definition.

Definition 5.7. *Let* $n,p,q \in \mathbb{N}$ *and* $0 < \delta < 1$*. Let* U *be a universe of size* n*. Furthermore, let* $(\mathscr{H}, \mathscr{S}, \{\mathscr{F}^{h,f,\mathbf{p}}\}|_{h\in\mathscr{H}, f\in\mathscr{S},\mathbf{p}})$ *be a* δ*-parsimonious* (n,p,q)*-universal tuple. Finally, let* $A \in \binom{U}{p}$ *and* $B \in \binom{U}{q}$ *be disjoint sets. We say that the pair* (A,B) fits *a quadruple* $(h,f,\mathbf{p},F)$ *of* $(\mathscr{H}, \mathscr{S}, \{\mathscr{F}^{h,f,\mathbf{p}}\}|_{h\in\mathscr{H}, f\in\mathscr{S},\mathbf{p}})$ *if* (A,B) *is nice with respect to* $(h,f,\mathbf{p})$*, and* $F \in \mathscr{F}[h(A), h(B)]$*.*

Finally, we state the promised property.

Lemma 5.2. *Let* $n,p,q \in \mathbb{N}$ *and* $0 < \delta < 1$*. Let* U *be a universe of size* n*. Furthermore, let* $(\mathscr{H}, \mathscr{S}, \{\mathscr{F}^{h,f,\mathbf{p}}\}|_{h\in\mathscr{H}, f\in\mathscr{S},\mathbf{p}})$ *be a* δ*-parsimonious* (n,p,q)*-universal tuple. Then, there exist* $T = T(n,p,q,\delta) > 0$ *and for every* $\mathbf{p}$ *that is* (p,q,t)*-compatible,* $T_{\mathbf{p}} = T_{\mathbf{p}}(n,p,q,\delta) > 0$*, such that for any* $A \in \binom{U}{p}$ *and* $B \in \binom{U}{q}$ *that are disjoint, the following conditions hold.*

1. *The number of triples* $(h,f,\mathbf{p})$ *with respect to which* (A,B) *is nice, where* $h \in \mathscr{H}$*,* $f \in \mathscr{S}$ *and* $\mathbf{p}$ *is* (p,q,t)*-compatible, is between* $(1-\delta)T$ *and* $(1+\delta)T$*.*
2. *For any triple* $(h,f,\mathbf{p})$ *with respect to which* (A,B) *is nice, where* $h \in \mathscr{H}$*,* $f \in \mathscr{S}$ *and* $\mathbf{p}$ *is* (p,q,t)*-compatible, the number of quadruples* $(h,f,\mathbf{p},F)$ *of* $(\mathscr{H}, \mathscr{S}, \{\mathscr{F}^{h,f,\mathbf{p}}\}|_{h\in\mathscr{H}, f\in\mathscr{S},\mathbf{p}})$ *that fit* (A,B) *is between* $(1-\delta)T_{\mathbf{p}}$ *and* $(1+\delta)T_{\mathbf{p}}$*.*

5.2 Proof of Lemma 5.1

Since we will reduce n to $\mathcal{O}(k^2/\epsilon)$ later on, our computation is allowed to be inefficient for large n, and we will now present such a procedure. To this end, we use an adaptation of the proof of Theorem 4.1 in [4] (based on the method of conditional probabilities). For the sake of completeness, we give the full details of this proof below.

Lemma 5.3. *Let* $n,p,q \in \mathbb{N}$ *and* $0 < \delta < 1$*, and denote* $k = p+q$*. Let* U *be a universe of size* n*. A* δ*-parsimonious* (n,p,q)*-universal family* $\mathscr{F}$ *of sets over* U *of size* $\ell = \mathcal{O}(\frac{k^k}{p^p q^q} \cdot k \log n \cdot \frac{1}{\delta^2})$ *can be computed in time* $\ell \cdot n^{k+1} k^{\mathcal{O}(1)}$*. In particular, the sets in* $\mathscr{F}$ *can be enumerated with* $n^{k+1}k^{\mathcal{O}(1)}$ *delay.*

Proof. Denote $t = \frac{p^p q^q}{k^k}$, $M = \frac{2(k \ln n + 1)}{t\delta^2}$ and $\lambda = \delta/2$. Consider a choice of M independent random sets $F_1, F_2, \ldots, F_M \subseteq U$: Every set F_i is obtained by inserting each element $u \in U$ into F with probability p/k. This choice will be derandomized in the course of the algorithm. For every pair (A,B) of disjoint sets $A \in \binom{U}{p}$ and $B \in \binom{U}{q}$, define $X_{(A,B)} = \sum_{i=1}^{M} X_{(A,B),i}$, where $X_{(A,B),i}$ is the indicator random variable that is equal to 1 if and only if the i-th set F_i contains A and is disjoint from B. Consider the following potential function:

$$\Phi = \sum_{A\in\binom{U}{p}} \sum_{B\in\binom{U\setminus A}{q}} \left(e^{\lambda(X_{(A,B)} - tM)} + e^{\lambda(tM - X_{(A,B)})}\right).$$

By Observation 3.1, its expectation can be calculated as follows:

$$\begin{aligned} E[\Phi] &= \binom{n}{p}\binom{n-p}{q}(e^{-\lambda tM}\prod_{i=1}^{M}E[e^{\lambda X_{(A,B),i}}] + e^{\lambda tM}\prod_{i=1}^{M}E[e^{-\lambda X_{(A,B),i}}]) \\ &= \binom{n}{p}\binom{n-p}{q}(e^{-\lambda tM}[te^{\lambda}+(1-t)]^{M} + e^{\lambda tM}[te^{-\lambda}+(1-t)]^{M}). \end{aligned}$$

We now give an upper bound for $E[\Phi]$. Since $1+u \leq e^u$ for all u, and $e^{-u} \leq 1-u+u^2/2$ for all $u \geq 0$, we get that $te^{-\lambda}+(1-t) \leq e^{te^{-\lambda}-1} \leq e^{t(-\lambda+\lambda^2/2)}$. Define $\epsilon = e^{\lambda}-1$, that is $\lambda = \ln(1+\epsilon)$. Thus $te^{\lambda}+(1-t) = 1+\epsilon t \leq e^{\epsilon t}$. This implies that

$$E[\Phi] \leq n^k((\frac{e^{\epsilon}}{1+\epsilon})^{tM} + e^{\lambda^2 tM/2}).$$

Since $e^u \leq 1+u+u^2$ for all $0 \leq u \leq 1$, we have that $\frac{e^{\epsilon}}{1+\epsilon} = e^{e^{\lambda}-1-\lambda} \leq e^{\lambda^2}$. We conclude that

$$E[\Phi] \leq 2n^k e^{\lambda^2 tM} \leq e^{2(k\ln n+1)}.$$

We now describe a deterministic algorithm for finding M sets so that $E[\Phi]$ will still obey the last upper bound. This is performed using the method of conditional probabilities (c.f., e.g., [5]). The algorithm will have M phases, where each phase will consist of n steps. In step i of phase j, the algorithm will determine whether the i-th element in U is inserted into F_i. Out of the two possible options (inserting the element or not inserting the element), we greedily choose the value that will decrease $E[\Phi]$ as much as possible. We note that at any specific step of the algorithm, the exact value of the conditional expectation of the potential function can be easily computed in time $n^k k^{\mathcal{O}(1)}$.

After all the M functions have been determined, every pair (A,B) of disjoint sets $A \in \binom{U}{p}$ and $B \in \binom{U}{q}$ satisfies the following:

$$e^{\lambda(X_{(A,B)}-tM)} + e^{\lambda(tM-X_{(A,B)})} \leq e^{2(k\ln n+1)}.$$

This implies that

$$-2(k\ln n+1) \leq \lambda(X_{(A,B)}-tM) \leq 2(k\ln n+1).$$

Recall that $\lambda = \delta/2$, and therefore

$$(1-\frac{4(k\ln n+1)}{\delta tM})tM \leq X_{(A,B)} \leq (1+\frac{4(k\ln n+1)}{\delta tM})tM.$$

Plugging in the value of M and t, we get the desired result

$$(1-\delta)tM \leq X_{(A,B)} \leq (1+\delta)tM.$$

This completes the proof. □

Having Lemma 5.3 at hand, we turn to present the efficient computation required to prove Lemma 5.1.

Lemma 5.1. *Let $p,q,t,z \in \mathbb{N}$, and denote $k = p+q$ and $s = k/t$. Let $(h,f,\mathbf{p})$ be (n,p,q,t,z)-compatible. Let $0<\delta<1$. A δ-parsimonious $(h,f,\mathbf{p})$-universal family $\mathscr{F}$ of sets over $\{1,\ldots,z\}$ of size $\ell = \mathcal{O}\left(\binom{k}{p}\cdot(k\cdot\log z\cdot\frac{\mathcal{O}(1)}{\delta})^{2t}\right)$ can be computed in time $\ell\cdot z^{s+1}s^{\mathcal{O}(1)}t$. In particular, the sets in $\mathscr{F}$ can be enumerated with delay $z^{s+1}s^{\mathcal{O}(1)}t$.*

Proof. Denote $Z = \{1, 2, \ldots, z\}$. Define the partition $(Z_1, Z_2, \ldots, Z_t)$ of Z as follows: for each $i \in \{1, 2, \ldots, t\}$, $Z_i = \{u \in Z : f(u) = i\}$. By Lemma 5.3, for any choice of $\widehat{p} \in \{1, 2, \ldots, s\}$ and $\widehat{q} = s - \widehat{p}$, we compute a $\frac{\ln(1+\delta)}{t}$-parsimonious $(z, \widehat{p}, \widehat{q})$-universal family $\mathscr{F}_{\widehat{p},\widehat{q}}$ of sets over Z of size $\ell_{\widehat{p},\widehat{q}} = \mathcal{O}(\frac{s^s}{\widehat{p}^{\widehat{p}}\widehat{q}^{\widehat{q}}} \cdot s \log z \cdot \frac{1}{(\frac{\ln(1+\delta)}{t})^2})$ with correction factor $T_{\widehat{p},\widehat{q}}$. In particular, we need not compute $\mathscr{F}_{\widehat{p},\widehat{q}}$ at once, but we can enumerate its sets with delay $z^{s+1}s^{\mathcal{O}(1)}$.

Let us now construct the family $\mathscr{F}$. For every choice of $F_1 \in \mathscr{F}_{p_1,q_1}, F_2 \in \mathscr{F}_{p_2,q_2}, \ldots, F_t \in \mathscr{F}_{p_t,q_t}$, we insert the set $(F_1 \cap Z_1) \cup (F_2 \cap Z_2) \cup \cdots \cup (F_t \cap Z_t)$ into $\mathscr{F}$. Observe that the sets in $\mathscr{F}$ can be enumerated with delay $z^{s+1}s^{\mathcal{O}(1)}t$. Since $s \cdot t = k$, the size of $\mathscr{F}$, denoted by ℓ, is upper bounded as follows.

$$\begin{aligned}
\ell &= \textstyle\prod_{i=1}^{t} \ell_{p_i,q_i} \\
&= \mathcal{O}\left(\left(\prod_{i=1}^{t} \frac{s^s}{p_i^{p_i} q_i^{q_i}}\right) \cdot \left(s \cdot \log z \cdot \frac{\mathcal{O}(1)}{(\frac{\ln(1+\delta)}{t})^2}\right)^t\right) \\
&= \mathcal{O}\left(\left(\prod_{i=1}^{t} \binom{s}{p_i}\right) \cdot \left(s \cdot \log z \cdot \frac{t^2 \cdot \mathcal{O}(1)}{\ln^2(1+\delta)}\right)^t\right) \\
&= \mathcal{O}\left(\binom{k}{p} \cdot \left(k \cdot \log z \cdot \frac{\mathcal{O}(1)}{\ln(1+\delta)}\right)^{2t}\right) \\
&= \mathcal{O}\left(\binom{k}{p} \cdot \left(k \cdot \log z \cdot \frac{\mathcal{O}(1)}{\delta}\right)^{2t}\right).
\end{aligned}$$

The last bound follows from Taylor series $\ln(1+x) = \sum_{n=1}^{\infty}(-1)^{n+1}\frac{x^n}{n}$, which implies that $\ln(1+\delta) \geq \delta - (\delta^2/2) \geq \delta/2$.

Define $T = \prod_{i=1}^{t} T_{p_i,q_i}$. To conclude the proof, it remains to prove that $\mathscr{F}$ is a δ-parsimonious $(h, f, \mathbf{p})$-universal family. To this end, we let (A, B) be an arbitrary nice pair, and show that $(1-\delta)\cdot T \leq |\mathscr{F}[h(A), h(B)]| \leq (1+\delta)\cdot T$. By the construction of $\mathscr{F}$ and because $(Z_1, Z_2, \ldots, Z_t)$ is a partition of Z, it holds that $|\mathscr{F}[h(A), h(B)]| = \prod_{i=1}^{t} |\mathscr{F}_{p_i,q_i}[h(A) \cap Z_i, h(B) \cap Z_i]|$. Moreover, since h is injective when restricted $A \cup B$ as well as $|\{u \in A : f(h(u)) = i\}| = p_i$ and $|\{u \in B : f(h(u)) = i\}| = (k/t) - p_i$ for every $i \in \{1, 2, \ldots, t\}$ (because (A, B) is a nice pair), the construction of $\mathscr{F}_{p_i,q_i}$ implies that $(1 - \frac{\ln(1+\delta)}{t}) \cdot T_{p_i,q_i} \leq |\mathscr{F}_{p_i,q_i}[h(A) \cap Z_i, h(B) \cap Z_i]| \leq (1 + \frac{\ln(1+\delta)}{t}) \cdot T_{p_i,q_i}$. We thus derive that

$$\begin{aligned}
|\mathscr{F}[h(A), h(B)]| &\leq \prod_{i=1}^{t}(1 + \frac{\ln(1+\delta)}{t}) \cdot T_{p_i,q_i} \\
&= \left(1 + \frac{\ln(1+\delta)}{t}\right)^t \cdot \prod_{i=1}^{t} T_{p_i,q_i} \\
&\leq e^{\ln(1+\delta)} \cdot \prod_{i=1}^{t} T_{p_i,q_i} \\
&= (1+\delta) \cdot T.
\end{aligned}$$

In addition, it holds that

$$
\begin{aligned}
|\mathscr{F}[h(A), h(B)]| &\geq \prod_{i=1}^{t}(1 - \frac{\ln(1+\delta)}{t}) \cdot T_{p_i,q_i} \\
&= \left(1 - \frac{\ln(1+\delta)}{t}\right)^t \cdot \prod_{i=1}^{t} T_{p_i,q_i} \\
&\geq (1 - \ln(1+\delta)) \cdot \prod_{i=1}^{t} T_{p_i,q_i} \\
&\geq (1-\delta) \cdot T.
\end{aligned}
$$

The last inequality follows from Taylor series $\ln(1+x) = \sum_{n=1}^{\infty}(-1)^{n+1}\frac{x^n}{n}$, which implies that $\ln(1+\delta) \leq \delta$ and therefore $1 - \ln(1+\delta) \geq 1 - \delta$. This completes the proof. □

5.3 Proof of Theorem 5.1

In order to prove Theorem 5.1, we make use of the following propositions.

Proposition 5.1 (Theorem 4 [3]). *For any $0 < \epsilon < 1$, an ϵ-balanced (z, k, t)-splitter $\mathscr{S}$ of size $\left(\frac{zk^t}{\epsilon}\right)^{\mathcal{O}(\log z)}$ can be constructed in time $\left(\frac{zk^t}{\epsilon}\right)^{\mathcal{O}(\log z)}$. Moreover, the functions in $\mathscr{S}$ can be enumerated with polynomial delay.*

Proposition 5.2 (Theorem 4.2 [4]). *For any $0 < \epsilon < 1$, an (ϵ, k)-balanced family of hash functions from $\{1, \ldots, n\}$ to $\{1, \ldots, \ell\}$, where $\ell = \frac{2k^2}{\epsilon}$, of size $\frac{k^{\mathcal{O}(1)}\log n}{\epsilon^{\mathcal{O}(1)}}$ can be constructed in time $\frac{k^{\mathcal{O}(1)} n \log n}{\epsilon^{\mathcal{O}(1)}}$.*

We are now ready to prove Theorem 5.1.

Theorem 5.1. *Let $n, p, q \in \mathbb{N}$ and $0 < \delta < 1$. Denote $k = p + q$. Let U be a universe of size n. A δ-parsimonious (n, p, q)-universal tuple $(\mathscr{H}, \mathscr{S}, \{\mathscr{F}^{h,f,\mathbf{p}}\}|_{h\in\mathscr{H}, f\in\mathscr{S}, \mathbf{p}})$ whose collection of quadruples is of size ℓ can be computed in time $\frac{k^{\mathcal{O}(1)} n \log n}{\delta^{\mathcal{O}(1)}} + \ell \cdot \Delta$. In particular, after preprocessing time $\frac{k^{\mathcal{O}(1)} n \log n}{\delta^{\mathcal{O}(1)}}$, the quadruples of $(\mathscr{H}, \mathscr{S}, \{\mathscr{F}^{h,f,\mathbf{p}}\}|_{h\in\mathscr{H}, f\in\mathscr{S}, \mathbf{p}})$ can be enumerated with delay Δ. Here,*

$$
\begin{aligned}
\ell &= \binom{k}{p} \cdot 2^{\mathcal{O}(\sqrt{k}(\log^2 k + \log^2 \frac{1}{\delta}))} \cdot \log n, \text{ and} \\
\Delta &= 2^{\mathcal{O}(\sqrt{k}(\log k + \log \frac{1}{\delta}))}.
\end{aligned}
$$

Proof. We construct $\mathscr{H}$ and $\mathscr{S}$ by making use of Propositions 5.1 and 5.2, respectively. For every $h \in \mathscr{H}$, $f \in \mathscr{S}$ and $\mathbf{p}$, we construct $\mathscr{F}^{h,f,\mathbf{p}}$ by making use of Lemma 5.1. By Proposition5.1, we compute $\mathscr{H}$ in time $\frac{k^{\mathcal{O}(1)} n \log n}{\delta^{\mathcal{O}(1)}}$. We need not construct $\mathscr{S}$ and $\{\mathscr{F}^{h,f,\mathbf{p}}\}|_{h\in\mathscr{H}, f\in\mathscr{S}, \mathbf{p}}$ explicitly, but we can enumerate the quadruples of $(\mathscr{H}, \mathscr{S}, \{\mathscr{F}^{h,f,\mathbf{p}}\}|_{h\in\mathscr{H}, f\in\mathscr{S}, \mathbf{p}})$ with delay Δ, where Δ is upper bounded as follows.

- By Proposition 5.2, the functions in $\mathscr{S}$ can be enumerated with delay polynomial in k and $\frac{1}{\delta}$.
- All (p, q, t)-compatible vectors $\mathbf{p}$ can be enumerated with delay polynomial in k.
- By Lemma 5.1, for every $h \in \mathscr{H}$, $f \in \mathscr{S}$ and $\mathbf{p}$, the sets in $\mathscr{F}$ can be enumerated with delay $z^{s+1}s^{\mathcal{O}(1)}t = \left(\frac{6k^2}{\delta}\right)^{\sqrt{k}+1}(\sqrt{k})^{\mathcal{O}(1)} = 2^{\mathcal{O}(\sqrt{k}(\log k + \log \frac{1}{\delta}))}$.

Thus, $\Delta = 2^{\mathcal{O}(\sqrt{k}(\log k + \log \frac{1}{\delta}))}$.

The assertion that $(\mathscr{H}, \mathscr{S}, \{\mathscr{F}^{h,f,\mathbf{p}}\}|_{h\in\mathscr{H}, f\in\mathscr{S},\mathbf{p}})$ is indeed a δ-parsimonious (n,p,q)-universal tuple follows directly from Propositions 5.1 and 5.2 and Lemma 5.1. Moreover, by these propositions and lemma, it holds that:

- $|\mathscr{H}| = \dfrac{k^{\mathcal{O}(1)}\log n}{\epsilon^{\mathcal{O}(1)}} = \dfrac{k^{\mathcal{O}(1)}\log n}{\delta^{\mathcal{O}(1)}}$;
- $|\mathscr{S}| = \left(\dfrac{zk^t}{\epsilon}\right)^{\mathcal{O}(\log z)} = \left(\dfrac{\frac{6k^2}{\delta}k^{\sqrt{k}}}{\delta/3}\right)^{\mathcal{O}(\log \frac{6k^2}{\delta})} = \left(\dfrac{k^{\sqrt{k}}}{\delta}\right)^{\mathcal{O}(\log \frac{k}{\delta})} = 2^{\mathcal{O}\left(\sqrt{k}(\log^2 k + \log^2 \frac{1}{\delta})\right)}$;
- the number of (p,q,t)-compatible vectors $\mathbf{p}$ is equal to

$$\binom{p+t-1}{t-1} \leq \binom{k+\sqrt{k}}{\sqrt{k}} \leq (k+\sqrt{k})^{\sqrt{k}} = 2^{\mathcal{O}(\sqrt{k}\log k)};$$

- for every $h \in \mathscr{H}$, $f \in \mathscr{S}$ and $\mathbf{p}$, it holds that

$$\begin{aligned} |\mathscr{F}^{h,f,\mathbf{p}}| &= \mathcal{O}\left(\binom{k}{p}\cdot\left(k\cdot\log z\cdot\frac{\mathcal{O}(1)}{\delta}\right)^{2t}\right) \\ &= \mathcal{O}\left(\binom{k}{p}\cdot\left(k\cdot\log\left(\frac{6k^2}{\delta}\right)\cdot\frac{\mathcal{O}(1)}{\delta}\right)^{2\sqrt{k}}\right) = \binom{k}{p}\cdot 2^{\mathcal{O}(\sqrt{k}(\log k+\log\frac{1}{\delta}))}. \end{aligned}$$

Thus, the number of quadruples of $(\mathscr{H}, \mathscr{S}, \{\mathscr{F}^{h,f,\mathbf{p}}\}|_{h\in\mathscr{S}, f\in\mathscr{S},\mathbf{p}})$, denoted by ℓ, is upper bounded as follows.

$$\begin{aligned} \ell &= \frac{k^{\mathcal{O}(1)}\log n}{\delta^{\mathcal{O}(1)}}\cdot 2^{\mathcal{O}(\sqrt{k}(\log^2 k+\log^2\frac{1}{\delta}))}\cdot\binom{k}{p}\cdot 2^{\mathcal{O}(\sqrt{k}(\log k+\log\frac{1}{\delta}))} \\ &= \binom{k}{p}\cdot 2^{\mathcal{O}(\sqrt{k}(\log^2 k+\log^2\frac{1}{\delta}))}\cdot\log n. \end{aligned}$$

This completes the proof. □

5.4 Proof of Lemma 5.2

Finally, we prove Lemma 5.2.

Lemma 5.2. *Let $n,p,q \in \mathbb{N}$ and $0<\delta<1$. Let U be a universe of size n. Furthermore, let $(\mathscr{H}, \mathscr{S}, \{\mathscr{F}^{h,f,\mathbf{p}}\}|_{h\in\mathscr{H}, f\in\mathscr{S},\mathbf{p}})$ be a δ-parsimonious (n,p,q)-universal tuple. Then, there exist $T = T(n,p,q,\delta) > 0$ and for every $\mathbf{p}$ that is (p,q,t)-compatible, $T_{\mathbf{p}} = T_{\mathbf{p}}(n,p,q,\delta) > 0$, such that for any $A \in \binom{U}{p}$ and $B \in \binom{U}{q}$ that are disjoint, the following conditions hold.*

1. *The number of triples $(h,f,\mathbf{p})$ with respect to which (A,B) is nice, where $h\in\mathscr{H}$, $f\in\mathscr{S}$ and $\mathbf{p}$ is (p,q,t)-compatible, is between $(1-\delta)T$ and $(1+\delta)T$.*
2. *For any triple $(h,f,\mathbf{p})$ with respect to which (A,B) is nice, where $h\in\mathscr{H}$, $f\in\mathscr{S}$ and $\mathbf{p}$ is (p,q,t)-compatible, the number of quadruples $(h,f,\mathbf{p},F)$ of $(\mathscr{H}, \mathscr{S}, \{\mathscr{F}^{h,f,\mathbf{p}}\}|_{h\in\mathscr{H}, f\in\mathscr{S},\mathbf{p}})$ that fit (A,B) is between $(1-\delta)T_{\mathbf{p}}$ and $(1+\delta)T_{\mathbf{p}}$.*

Proof. Define $T = T_{\mathscr{H}}\cdot T_{\mathscr{S}}$, and for every $\mathbf{p}$ that is (p,q,t)-compatible, let the definition of $T_{\mathbf{p}}$ be given by Definition 5.6. Let us consider some $A \in \binom{U}{p}$ and $B \in \binom{U}{q}$ that are disjoint. Then, we know that

- the number of hash functions $h \in \mathscr{H}$ that are injective when restricted to $A\cup B$ is between $(1-\epsilon)T_{\mathscr{H}}$ and $(1+\epsilon)T_{\mathscr{H}}$,

- for every hash function $h \in \mathscr{H}$ that is injective when restricted to $A \cup B$, the number of splitting functions $f \in \mathscr{S}$ such that for every $i \in \{1, 2, \ldots, t\}$, $|\{u \in h(A \cup B) : f(u) = i\}| = s$, is between $(1-\epsilon)T_{\mathscr{S}}$ and $(1+\epsilon)T_{\mathscr{S}}$, and
- for every pair (h, f) of a hash function $h \in \mathscr{H}$ that is injective when restricted to $A \cup B$ and a splitting function $f \in \mathscr{S}$ such that for every $i \in \{1, 2, \ldots, t\}$, $|\{u \in h(A \cup B) : f(u) = i\}| = s$, the number of vectors $\mathbf{p}$ such that for every $i \in \{1, 2, \ldots, t\}$, $|\{u \in h(A) : f(u) = i\}| = \mathbf{p}(i)$, is exactly 1.

Thus, the number of triples $(h, f, \mathbf{p})$ with respect to which (A, B) is nice, where $h \in \mathscr{H}$, $f \in \mathscr{S}$ and $\mathbf{p}$ is (p, q, t)-compatible, is between $(1-\epsilon)T_{\mathscr{H}} \cdot (1-\epsilon)T_{\mathscr{S}}$ and $(1+\epsilon)T_{\mathscr{H}} \cdot (1+\epsilon)T_{\mathscr{S}}$. Recall that $\epsilon = \delta/3$, and note that $\delta^2 < \delta$. Therefore, $(1-\epsilon)^2 = 1 - (2\epsilon - \epsilon^2) = 1 - (\frac{2}{3}\delta - \frac{1}{9}\delta^2) \geq 1 - \delta$, and $(1+\epsilon)^2 = 1 + (2\epsilon + \epsilon^2) = 1 + (\frac{2}{3}\delta + \frac{1}{9}\delta^2) \leq 1 + \delta$. However, this implies that Condition 1 is satisfied.

Now, we know that for every hash function $h \in \mathscr{H}$, splitter $f \in \mathscr{S}$ and (p, q, t)-compatible function $\mathbf{p}$, it holds that $\mathscr{F}^{h,f,\mathbf{p}}$ is a δ-parsimonious $(h, f, \mathbf{p})$-universal family with correction factor $T_{\mathbf{p}}$. This means that the number of quadruples $(h, f, \mathbf{p}, F)$ of $(\mathscr{H}, \mathscr{S}, \{\mathscr{F}^{h,f,\mathbf{p}}\}|_{h \in \mathscr{H}, f \in \mathscr{S}, \mathbf{p}})$ that fit (A, B) is between $(1-\delta)T_{\mathbf{p}}$ and $(1+\delta)T_{\mathbf{p}}$. Thus, Condition 2 is satisfied as well. Since the choice of A and B was arbitrary, the proof is complete. □

6 Deterministic FPT-AS for #k-Path

Our deterministic FPT-AS builds upon the scheme of our second randomized FPT-AS, but it is more technical. We first discuss the main idea that underlies the design of this algorithm. Like our previous algorithm, this algorithm (denoted by $\mathcal{A}$) is recursive. However, in addition to G', k' and β, every call to $\mathcal{A}$ is also given two tuples $\mathcal{R}$ and $\mathcal{W}$. The number of elements in $\mathcal{R}$ and $\mathcal{W}$ equals the depth d of the current recursive call in the recursion tree.

Roughly speaking, every element in $\mathcal{R}$ is a quadruple $(h_i, f_i, \mathbf{p}_i, \sigma_i)$ where *(i)* the triple $(h_i, f_i, \mathbf{p}_i)$ corresponds to the interpretation preceding Definition 5.1, and *(ii)* $\sigma_i \in \{\texttt{left}, \texttt{right}\}$ indicates whether we should count paths that consist of $\mathbf{p}_i(j)$ (in case $\sigma_i = \texttt{left}$) or $s_i - \mathbf{p}_i(j)$ (in case $\sigma_i = \texttt{right}$) vertices of the j-th part of the reduced universe split by f_i. Thus, we "keep track" of all triples considered along the current recursion branch. The reason why we have to store this information is to ensure that, in the current recursive call, we only count paths P whose vertex set has the following property: when we will return to the i-th recursive call, the partition (A, B) of $V(P)$ where A consists of the first $\widehat{k}$ vertices of P (for a certain $\widehat{k} \in \{1, 2, \ldots, k\}$ that depends on the location of this i-th call in the recursion tree) is nice with respect to $(h_i, f_i, \mathbf{p}_i)$, see Definition 5.2. This simple (though perhaps slightly tedious) bookkeeping sidesteps the fact that Lemma 5.2 only suits nice pairs.

The tuple $\mathcal{W}$ is meant to keep track of how many vertices the paths that we currently count have used "so far" from the j-th part of the universe split by f_i for every choice of i and j. For this purpose, $\mathcal{W}$ is defined to have the form $(\mathbf{w}_1, \mathbf{w}_2, \ldots, \mathbf{w}_d)$ such that for each $i \in \{1, 2, \ldots, d\}$, the following condition holds: For each $j \in \{1, 2, \ldots, t_i\}$, if $\sigma_i = \texttt{left}$ then $\mathbf{w}_i(j) \leq \mathbf{p}_i(j)$, and otherwise $\mathbf{w}_i(j) \leq s_i - \mathbf{p}_i(j)$. Here, $s_i = \sqrt{(k/2^i)}$ is the number of vertices the paths that we currently count should use (in total) from each part split by f_i.

Accordingly, the value returned by a call $\mathcal{A}(G', k', \beta, \mathcal{R}, \mathcal{W})$ is an assignment $\alpha : V(G') \to \mathbb{N}_0$ with the following property: For each vertex $v \in V(G')$, it holds that $\alpha(v)$ approximates $\sum_{\substack{\{p,q\} \in E(G) \\ \text{s.t. } p \notin V(G'), q \in V(G')}} \beta(p) \cdot |\mathcal{P}_{q,v}^{G',k',\mathcal{R},\mathcal{W}}|$. Roughly speaking, $\mathcal{P}_{q,v}^{G',k',\mathcal{R},\mathcal{W}}$ is the collection of all k'-paths in G' with endpoints q and v that "comply" (in a sense that will be made formal later) with the constraints imposed by $\mathcal{R}$ and $\mathcal{W}$.

6.1 Algorithm Specification

Let $\xi = \frac{\ln(1+\epsilon)}{2(k-1)}$. As in Section 4.2, we add a new vertex s to G and connect it to all vertices in G. Thus, rather than counting the number of k-paths in the former graph G, we can count the number of $(k+1)$-paths with s as an endpoint in the new graph G. In what follows, we focus on this goal.

Our algorithm, denoted by $\mathcal{A}$, is recursive. We first specify the arguments with which $\mathcal{A}$ is called. Then, we specify the value returned by a recursive call. Afterwards, we note how $\mathcal{A}$ is initially called, and then we turn to describe the computation executed by a recursive call.

Arguments. Each call to $\mathcal{A}$ is of the form $\mathcal{A}(G', k', \beta, \mathcal{R}, \mathcal{W})$ where G' is an induced subgraph of G, $k' = k/2^d$ for some $d \in \{0, 1, \ldots, \log_2 k\}$, $\beta : V(G) \setminus V(G') \to \mathbb{N}_0$, and $\mathcal{R} = ((h_1, f_1, \mathbf{p}_1, \sigma_1), (h_2, f_2, \mathbf{p}_2, \sigma_2), \ldots, (h_d, f_d, \mathbf{p}_d, \sigma_d))$. (The argument $\mathcal{W}$ is defined later.) For each $i \in \{1, 2, \ldots, d\}$, it holds that

- $k_i := \frac{k}{2^i}$, $z_i := \frac{2{k_i}^2}{\xi}$, $t_i := \sqrt{k_i}$ and $s_i := k_i/t_i$.
- h_i is a function from $\{1, 2, \ldots, z_{i-1}\}$ to $\{1, 2, \ldots, z_i\}$ if $i \geq 2$, and from $V(G)$ to $\{1, 2, \ldots, z_1\}$ otherwise,
- f_i is a function from $\{1, 2 \ldots, z_i\}$ to $\{1, 2, \ldots, t_i\}$,
- $\mathbf{p}_i$ is a function from $\{1, 2, \ldots, t_i\}$ to $\{1, 2, \ldots, s_i\}$ such that $\sum_{j=1}^{t_i} \mathbf{p}_i(j) = k_i/2$, and
- $\sigma_i \in \{\texttt{left}, \texttt{right}\}$.

Additionally, define $z_0 = n$. Roughly speaking, the tuple $\mathcal{R}$ stores the sequence of parsimonious universal tuples that were considered along the branch of the recursion tree that leads from the initial call to $\mathcal{A}$ to the current call to $\mathcal{A}$. We need to keep track of this information in order to ensure that the paths we count in the current call behave "nicely" with respect to this entire sequence of parsimonious universal tuples—indeed each of these parsimonious universal tuples has the behavior we want only with respect to nice pairs (see Definition 5.2 and Lemma 5.2).

Let us now explain what is the argument $\mathcal{W}$. To this end, we need the following definition.

Definition 6.1. *For the arguments G', k' and $\mathcal{R}$ of a call to $\mathcal{A}$, the* collection of relevant tuples $\mathsf{W}_{G',k',\mathcal{R}}$ *is the collection of all tuples $\mathcal{W} = (\mathbf{w}_1, \mathbf{w}_2, \ldots, \mathbf{w}_d)$ such that for each $i \in \{1, 2, \ldots, d\}$, the following condition holds:*

- *For each $j \in \{1, 2, \ldots, t_i\}$, if $\sigma_i = \texttt{left}$ then $\mathbf{w}_i(j) \leq \mathbf{p}_i(j)$, and otherwise $\mathbf{w}_i(j) \leq s_i - \mathbf{p}_i(j)$.*

In case $d = 0$ (that is, the initial call), $\mathsf{W}_{G,k,()} = \{()\}$.

Having this definition at hand, we note that the argument $\mathcal{W}$ in a call $\mathcal{A}(G', k', \beta, \mathcal{R}, \mathcal{W})$ is a tuple from $\mathsf{W}_{G',k',\mathcal{R}}$. Intuitively, the reason why we need such a tuple $\mathcal{W}$ can be roughly explained as follows. Each parsimonious universal tuple $(h_i, f_i, \mathbf{p}_i, \sigma_i)$ in $\mathcal{R}$ can be thought of as a "demand" to make sure that each of the paths counted when we go back to the i-th call on the current branch tree (mentioned earlier) uses *exactly* $\mathbf{p}_i(j)$ (or $s_i - \mathbf{p}_i(j)$) vertices from a particular set for each $j \in \{1, \ldots, t_i\}$. However, in the current call, we do not need to exhaust all of this budget of $\mathbf{p}_i(j)$ vertices—it should only be exhausted later, when we eventually return to the i-th call; instead, we need to keep track of every possibility of how this budget should be exhausted. The argument $\mathcal{W}$ is meant to serve this purpose.

Output Value. Before we can state the value returned by a call, we need another definition. Here, we make the definition of which paths "comply" with all of the demands imposed by $\mathcal{R}$ and $\mathcal{W}$ precise. Since our parsimonious universal tuples only work properly for nice sets, it will be crucial that we only count paths that fit this definition.

Definition 6.2. *For a call $\mathcal{A}(G', k', \beta, \mathcal{R}, \mathcal{W})$ with $\mathcal{W} = (\mathbf{w}_1, \mathbf{w}_2, \ldots, \mathbf{w}_d) \in \mathsf{W}_{G',k',\mathcal{R}}$, and vertices $u, v \in V(G')$, the* collection of relevant paths $\mathcal{P}_{u,v}^{G',k',\mathcal{R},\mathcal{W}}$ *is the collection of all k'-paths P in G' with endpoints u and v such that the following conditions holds:*

- *For each $i \in \{1, 2, \ldots, d\}$, denote $h_i^\star := (h_i \circ \cdots \circ (h_3 \circ (h_2 \circ h_1)))$. Then, the function $h_d^\star$ is injective when restricted to $V(P)$.*
- *For each $i \in \{1, 2, \ldots, d\}$, denote $f_i^\star := f_i \circ h_i^\star$. Then, for all $i \in \{1, 2, \ldots, d\}$ and $j \in \{1, 2, \ldots, t_i\}$, it holds that $|\{v \in V(P) : f_i^\star(v) = j\}| = \mathbf{w}_i(j)$.*

Note that for $\mathcal{W} = ()$, $\mathcal{P}_{u,v}^{G',k',\mathcal{R},\mathcal{W}}$ is the collection of all k'-paths P in G' with endpoints u and v. (The two conditions are vacuously satisfied.)

The value returned by each call $\mathcal{A}(G', k', \beta, \mathcal{R}, \mathcal{W})$ is an assignment $\alpha : V(G') \to \mathbb{N}_0$ with the following property: For each vertex $v \in V(G')$, it holds that $\alpha(v)$ approximates the following number:

$$\sum_{\substack{\{p,q\} \in E(G) \\ \text{s.t. } p \notin V(G'), q \in V(G')}} \beta(p) \cdot |\mathcal{P}_{q,v}^{G',k',\mathcal{R},\mathcal{W}}|.$$

Initial Call. The initial call is $\mathcal{A}(G - \{s\}, k, \beta_{\text{init}}, (), ())$ (that is, $\mathcal{R}$ and $\mathcal{W}$ are 0-length tuples), where $\beta_{\text{init}}(s) = 1$. The final output, returned by the initial call, is $\sum_{v \in V(G) \setminus \{s\}} \alpha(v)$.

Computation: Basis. Now, we turn to describe a call $\mathcal{A}(G', k', \beta, \mathcal{R}, \mathcal{W})$. In the basis, where $k' = 1$, we compute the integer $\alpha(v)$ for all $v \in V(G')$ as follows.

- If for all $i \in \{1, 2, \ldots, d\}$ and $j \in \{1, 2, \ldots, t_i\}$, $|\{v\} \cap \{r \in V(G') : f_i^\star(r) = j\}| = \mathbf{w}_i(j)$,[7] then
$$\alpha(v) = \sum_{\substack{u \notin V(G') \\ \text{s.t. } \{u,v\} \in E(G)}} \beta(u).$$
- Otherwise, $\alpha(v) = 0$.

Computation: Step. Next, suppose that $k' = k/2^d \geq 2$. Denote $\mathcal{R} = ((h_1, f_1, \mathbf{p}_1, \sigma_1), (h_2, f_2, \mathbf{p}_2, \sigma_2), \ldots, (h_d, f_d, \mathbf{p}_d, \sigma_d))$ and $\mathcal{W} = (\mathbf{w}_1, \mathbf{w}_2, \ldots, \mathbf{w}_d)$. By Theorem 5.1, for an ξ-parsimonious $(z_d, k'/2, k'/2)$-universal tuple $(\mathscr{H}, \mathscr{S}, \{\mathscr{F}^{h,f,\mathbf{p}}\}|_{h \in \mathscr{H}, f \in \mathscr{S}, \mathbf{p}})$ with universe $U = \{1, 2, \ldots, z_d\}$ (for $d = 0$, we mean $U = V(G)$), after spending preprocessing time $\frac{{k'}^{\mathcal{O}(1)} z_d \log z_d}{\xi^{\mathcal{O}(1)}}$, we enumerate the ℓ quadruples $Q = (h, f, \mathbf{p}, F) \in (\mathscr{H}, \mathscr{S}, \{\mathscr{F}^{h,f,\mathbf{p}}\}|_{h \in \mathscr{H}, f \in \mathscr{S}, \mathbf{p}})$ with delay Δ, where

$$\begin{aligned} \ell &= 2^{k' + \mathcal{O}(\sqrt{k'}(\log^2 k' + \log^2 \frac{1}{\xi}))} \log z_d, \text{ and} \\ \Delta &= 2^{\mathcal{O}(\sqrt{k'}(\log^2 k' + \log^2 \frac{1}{\xi}))}. \end{aligned}$$

For each quadruple $Q = (h, f, \mathbf{p}, F)$, we proceed as follows.

[7] That is, for each $i \in \{1, 2, \ldots, d\}$ there is a single $j \in \{1, 2, \ldots, t_i\}$ such that $\mathbf{w}_i(j) = 1$ (for any other $j' \in \{1, 2, \ldots, t_i\}$, $\mathbf{w}_i(j') = 0$), and for this j it holds that $f_i^\star(v) = j$.

- Denote $F^\star = (h \circ h_d^\star)^{-1}(F) \cap V(G')$, that is, $F^\star$ is the set of vertices in $V(G')$ that $h \circ h_d^\star$ maps to integers in F.
- Denote $\mathcal{R}_{\mathtt{left}} = \mathcal{R} + ((h, f, \mathbf{p}, \mathtt{left}))$.
- Denote $\mathcal{R}_{\mathtt{right}} = \mathcal{R} + ((h, f, \mathbf{p}, \mathtt{right}))$.
- For all $\mathcal{W}_{\mathtt{left}} = (\mathbf{w}_1^{\mathtt{left}}, \mathbf{w}_2^{\mathtt{left}}, \ldots, \mathbf{w}_{d+1}^{\mathtt{left}}) \in \mathsf{W}_{G'[F^\star], k'/2, \mathcal{R}_{\mathtt{left}}}$ and $\mathcal{W}_{\mathtt{right}} = (\mathbf{w}_1^{\mathtt{right}}, \mathbf{w}_2^{\mathtt{right}}, \ldots, \mathbf{w}_{d+1}^{\mathtt{right}}) \in \mathsf{W}_{G'-F^\star, k'/2, \mathcal{R}_{\mathtt{right}}}$, we say that $(\mathcal{W}_{\mathtt{left}}, \mathcal{W}_{\mathtt{right}})$ *fits* $\mathcal{W}$ if two conditions hold:
  1. for all $i \in \{1, \ldots, d\}$ and $j \in \{1, \ldots, t_i\}$, it holds that $\mathbf{w}_i(j) = \mathbf{w}_i^{\mathtt{left}}(j) + \mathbf{w}_i^{\mathtt{right}}(j)$;
  2. for all $j \in \{1, \ldots, t_{d+1}\}$, it holds that $\mathbf{p}(j) = \mathbf{w}_{d+1}^{\mathtt{left}}(j)$ and $s_{d+1} - \mathbf{p}(j) = \mathbf{w}_{d+1}^{\mathtt{right}}(j)$.

  Note that if $\mathcal{W} = ()$, then the first condition is vacuously satisfied.

Having established the notations above, for every $\mathcal{W}_{\mathtt{left}} \in \mathsf{W}_{G'[F^\star], k'/2, \mathcal{R}_{\mathtt{left}}}$, we perform two recursive calls as follows:

1. First, we call $\mathcal{A}$ with $(G'[F^\star], k'/2, \beta_{Q,\mathcal{W}_{\mathtt{left}}}, \mathcal{R}_{\mathtt{left}}, \mathcal{W}_{\mathtt{left}})$ where $\beta_{Q,\mathcal{W}_{\mathtt{left}}}$ is the extension of β that assigns 0 to every vertex in $V(G') \setminus F^\star$. Let $\alpha_{Q,\mathcal{W}_{\mathtt{left}}}$ be the output of this call, and extend it to assign 0 to every vertex in $V(G) \setminus V(G')$.
2. Second, we call $\mathcal{A}$ with $(G' - F^\star, k'/2, \alpha_{Q,\mathcal{W}_{\mathtt{left}}}, \mathcal{R}_{\mathtt{right}}, \mathcal{W}_{\mathtt{right}})$ where $\mathcal{W}_{\mathtt{right}}$ is the unique tuple in $\mathsf{W}_{G'-F^\star, k'/2, \mathcal{R}_{\mathtt{right}}}$ such that $(\mathcal{W}_{\mathtt{left}}, \mathcal{W}_{\mathtt{right}})$ fits $\mathcal{W}$. Let $\alpha_{Q,\mathcal{W}_{\mathtt{right}}}$ be the output of this recursive call.

After all quadruples Q were enumerated, the output $\alpha : V(G') \to \mathbb{N}_0$ is computed as follows. First, for all $v \in V(G')$ and a quadruple $Q = (h, f, \mathbf{p}, F)$, we define

$$\alpha_Q(v) = \sum_{(\mathcal{W}_{\mathtt{left}}, \mathcal{W}_{\mathtt{right}}) \text{ fits } \mathcal{W}} \alpha_{Q,\mathcal{W}_{\mathtt{right}}}(v).$$

We stress that we do not store all the assignments $\alpha_{Q,\mathcal{W}_{\mathtt{left}}}$, $\alpha_{Q,\mathcal{W}_{\mathtt{right}}}$ and α_Q simultaneously, but we only need to store one such assignment at a time in order to compute the assignment mentioned below "on the go".

Let $T > 0$ and for every $\mathbf{p}$, $T_{\mathbf{p}} > 0$, be the correction factors of $(\mathscr{H}, \mathscr{S}, \{\mathscr{F}^{h,f,\mathbf{p}}\}|_{h \in \mathscr{H}, f \in \mathscr{S}, \mathbf{p}})$ (see Lemma 5.2). For all $v \in V(G')$, we calculate

$$\alpha(v) = \sum_{Q=(h,f,\mathbf{p},F) \in (\mathscr{H}, \mathscr{S}, \{\mathscr{F}^{h,f,\mathbf{p}}\}|_{h \in \mathscr{H}, f \in \mathscr{S}, \mathbf{p}})} \frac{\alpha_Q(v)}{T \cdot T_{\mathbf{p}}}.$$

This completes the description of $\mathcal{A}$. (The pseudocode of $\mathcal{A}$ is given in Algorithm 3.)

6.2 Analysis

The main part of the analysis is done in the proof of the following lemma.

Lemma 6.1. *There exists a fixed constant $\eta > 0$ such that for any call $\mathcal{A}(G', k', \beta, \mathcal{R}, \mathcal{W})$ where $k' = k/2^d$, the following conditions hold.*

- *$\mathcal{A}(G', k', \beta, \mathcal{R}, \mathcal{W})$ runs in time*

$$4^{k' + \eta\sqrt{k'}(\log^2 k' + \log^2 \frac{1}{\xi})} \cdot 8^{\log k' \cdot \sqrt{k} \log k} \cdot M,$$

  where $M = m$ if $k' < k$ and $M = m \log n$ otherwise (i.e., $k' = k$). Moreover, $\mathcal{A}(G', k', \beta, \mathcal{R}, \mathcal{W})$ has a polynomial space complexity.

Algorithm 3 Deterministic FPT-AS for $\#k$-PATH.

let $\xi = \frac{\ln(1+\epsilon)}{2(k-1)}$;
add a new vertex s to G, and connect s to all vertices in G;
let $\beta : \{s\} \to \mathbb{N}_0$ assign 1 to s;
call $\textsc{Det}(G - \{s\}, k+1, \beta, (), ())$: let α be the output;
return $\sum_{v \in V(G) \setminus \{s\}} \alpha(v)$;
function $\textsc{Det}(G', k', \beta, \mathcal{R}, \mathcal{W})$
    if $k' = 1$ **then**
        return $\alpha : V(G') \to \mathbb{N}_0$ where for $v \in V(G')$: if for all $i \in \{1, \ldots, d\}$ and $j \in \{1, \ldots, t_i\}$, $|\{v\} \cap \{r \in V(G') : f_i^\star(r) = j\}| = \mathbf{w}_i(j)$, then $\alpha(v) = \sum_{\substack{u \notin V(G'): \\ \{u,v\} \in E(G)}} \beta(u)$; otherwise, $\alpha(v) = 0$; ▷ See notation in Arguments and Definition 6.2
    end if
    initialize $\alpha : V(G') \to \mathbb{N}_0$ to assign 0 to each $v \in V(G')$;
    let I be an iterator of a ξ-parsimonious $(z_d, k'/2, k'/2)$-universal tuple $(\mathscr{H}, \mathscr{S}, \{\mathscr{F}^{h,f,\mathbf{p}}\}|_{h \in \mathscr{H}, f \in \mathscr{S}, \mathbf{p}})$ over $U = \{1, 2, \ldots, z_d\}$ with correction factor T and for $\mathbf{p}$, $T_{\mathbf{p}} > 0$, given by Theorem 5.1;
    for all $Q = (h, f, \mathbf{p}, F) \in (\mathscr{H}, \mathscr{S}, \{\mathscr{F}^{h,f,\mathbf{p}}\}|_{h \in \mathscr{H}, f \in \mathscr{S}, \mathbf{p}})$ (iterate with I) **do**
        let $F^\star = (h \circ h_d^\star)^{-1}(F) \cap V(G')$, $\mathcal{R}_{\texttt{left}} = \mathcal{R} + ((h, f, \mathbf{p}, \texttt{left}))$, and $\mathcal{R}_{\texttt{right}} = \mathcal{R} + ((h, f, \mathbf{p}, \texttt{right}))$; ▷ See Definition 6.2
        initialize $\alpha_Q : V(G') \to \mathbb{N}_0$ to assign 0 to each $v \in V(G')$;
        for all $\mathcal{W}_{\texttt{left}} \in \mathsf{W}_{G'[F^\star], k'/2, \mathcal{R}_{\texttt{left}}}$ **do** ▷ See Definition 6.1
            let $\beta_{Q, \mathcal{W}_{\texttt{left}}}$ be the extension of β that assigns 0 to each $v \in V(G') \setminus F^\star$;
            call $\textsc{Det}(G'[F^\star], k'/2, \beta_{Q, \mathcal{W}_{\texttt{left}}}, \mathcal{R}_{\texttt{left}}, \mathcal{W}_{\texttt{left}})$: let $\alpha_{Q, \mathcal{W}_{\texttt{left}}}$ be the output;
            extend $\alpha_{Q, \mathcal{W}_{\texttt{left}}}$ to assign 0 to each $v \in V(G) \setminus V(G')$;
            let $\mathcal{W}_{\texttt{right}}$ is the unique tuple in $\mathsf{W}_{G' - F^\star, k'/2, \mathcal{R}_{\texttt{right}}}$ such that $(\mathcal{W}_{\texttt{left}}, \mathcal{W}_{\texttt{right}})$ fits $\mathcal{W}$; ▷ See Conditions 1 and 2 in Computation: Step
            call $\textsc{Det}(G' - F^\star, k'/2, \alpha_{Q, \mathcal{W}_{\texttt{left}}}, \mathcal{R}_{\texttt{right}}, \mathcal{W}_{\texttt{right}})$: let $\alpha_{Q, \mathcal{W}_{\texttt{right}}}$ be the output;
            update $\alpha_Q(v) = \alpha_Q(v) + \alpha_{Q, \mathcal{W}_{\texttt{right}}}(v)$ for $v \in V(G') \setminus F^\star$;
        end for
        update $\alpha(v) = \alpha(v) + \alpha_Q(v)/(T \cdot T_{\mathbf{p}})$ for $v \in V(G')$;
    end for
    return α;
end function

- *For all $v \in V(G')$, the number $\alpha(v)$ assigned to v by the assignment α returned by $\mathcal{A}(G', k', \beta, \mathcal{R}, \mathcal{W})$ satisfies the following inequalities:*

$$\alpha(v) \geq (1-\xi)^{2(k'-1)} \sum_{\substack{\{p,q\} \in E(G) \\ \text{s.t. } p \notin V(G'), q \in V(G')}} \beta(p) \cdot |\mathcal{P}_{q,v}^{G',k',\mathcal{R},\mathcal{W}}|.$$
$$\alpha(v) \leq (1+\xi)^{2(k'-1)} \sum_{\substack{\{p,q\} \in E(G) \\ \text{s.t. } p \notin V(G'), q \in V(G')}} \beta(p) \cdot |\mathcal{P}_{q,v}^{G',k',\mathcal{R},\mathcal{W}}|.$$

Proof. The proof is by induction of d. In the basis, where $k' = 1$, the claim easily holds. Now, let $d \leq \log_2 k - 1$, and suppose that the claim holds for $d+1$.

Time and Space Complexities. It is clear that $\mathcal{A}(G', k', \beta, \mathcal{R}, \mathcal{W})$ has a polynomial space complexity. Note that $z_d = n$ if $d = 0$, and $z_d = \mathcal{O}(k')$ otherwise. Denote $N = n$ if $d = 0$, and $N = 2$ (so $\log N = 1$) otherwise. Then, after preprocessing time bounded by $\frac{{k'}^{\mathcal{O}(1)} N \log N}{\xi^{\mathcal{O}(1)}}$, for some fixed constant $\lambda > 0$, the ℓ quadruples $Q = (h, f, \mathbf{p}, F) \in (\mathscr{H}, \mathscr{S}, \{\mathscr{F}^{h,f,\mathbf{p}}\}|_{h \in \mathscr{H}, f \in \mathscr{S}, \mathbf{p}})$ are enumerated with delay Δ, where

$$\begin{aligned} \ell &= 2^{k' + \lambda\sqrt{k'}(\log^2 k' + \log^2 \frac{1}{\xi})} \log N, \text{ and} \\ \Delta &= 2^{\lambda\sqrt{k'}(\log k' + \log \frac{1}{\xi})}. \end{aligned}$$

The number of pairs $(\mathcal{W}_{\texttt{left}}, \mathcal{W}_{\texttt{right}})$ that fit $\mathcal{W}$, which equals $|\mathsf{W}_{G'[F^\star], k'/2, \mathcal{R}_{\texttt{left}}}|$, is upper bounded by

$$r := \prod_{i=1}^{d} (s_i + 1)^{t_i} = \prod_{i=1}^{d} (\sqrt{\frac{k}{2^i}} + 1)^{\sqrt{\frac{k}{2^i}}} \leq \prod_{i=1}^{\log k} (\sqrt{k})^{\sqrt{\frac{k}{2^i}}} < 2^{\sqrt{k} \log k \cdot \sum_{i=1}^{\log k} (\frac{1}{\sqrt{2}})^i} < 8^{\sqrt{k} \log k}.$$

Thus, by the inductive hypothesis, for fixed constant $\zeta > 0$ (that does not depend on the inductive hypothesis), the running time of $\mathcal{A}(G', k', \beta, \mathcal{R}, \mathcal{W})$ is upper bounded by

$$\begin{aligned} & \ell \cdot \left(\zeta \cdot r \cdot 4^{\frac{k'}{2} + \eta\sqrt{\frac{k'}{2}}(\log^2 \frac{k'}{2} + \log^2 \frac{1}{\xi})} 8^{\log \frac{k'}{2} \cdot \sqrt{k} \log k} m \right) \\ \leq \; & 2^{k' + \lambda\sqrt{k'}(\log^2 k' + \log^2 \frac{1}{\xi})} \log N \cdot \zeta \cdot 8^{\sqrt{k} \log k} \cdot 4^{\frac{k'}{2} + \eta\sqrt{\frac{k'}{2}}(\log^2 k' + \log^2 \frac{1}{\xi})} 8^{(\log k' - 1)\sqrt{k} \log k} m \\ = \; & \zeta 4^{k' + \lambda\sqrt{k'}(\log^2 k' + \log^2 \frac{1}{\xi}) + \frac{\eta}{\sqrt{2}}\sqrt{k'}(\log^2 \frac{k'}{2} + \log^2 \frac{1}{\xi})} \cdot 8^{\log k' \cdot \sqrt{k} \log k} \cdot M. \end{aligned}$$

Thus, by choosing η to be a large enough constant that depends only on ζ and λ (say, $\eta = 10 \max\{\zeta, \lambda\}$), the expression above is upper bounded by

$$4^{k' + \eta\sqrt{k'}(\log^2 k' + \log^2 \frac{1}{\xi})} \cdot 8^{\log k' \cdot \sqrt{k} \log k} \cdot M.$$

Accuracy. Towards the proof of the second item of the claim, consider some vertex $v \in V(G')$. By definition, we have that

$$\alpha(v) = \sum_{Q=(h,f,\mathbf{p},F) \in (\mathscr{H}, \mathscr{S}, \{\mathscr{F}^{h,f,\mathbf{p}}\}|_{h \in \mathscr{H}, f \in \mathscr{S}, \mathbf{p}})} \; \sum_{(\mathcal{W}_{\texttt{left}}, \mathcal{W}_{\texttt{right}}) \text{ fits } \mathcal{W}} \frac{\alpha_{Q, \mathcal{W}_{\texttt{right}}}(v)}{T \cdot T_{\mathbf{p}}}.$$

By the inductive hypothesis, for each term $\alpha_{Q,\mathcal{W}_{\texttt{right}}}(v)$ in the sum above, we have that

$$\alpha_{Q,\mathcal{W}_{\texttt{right}}}(v) \leq (1+\xi)^{k'-2} \sum_{\substack{\{a,b\} \in E(G) \\ \text{s.t. } a \in F^\star, b \in V(G') \setminus F^\star}} \alpha_{Q,\mathcal{W}_{\texttt{left}}}(a) \cdot |\mathcal{P}_{b,v}^{G'-F^\star, k'/2, \mathcal{R}_{\texttt{right}}, \mathcal{W}_{\texttt{right}}}|.$$

By applying the inductive hypothesis again, we have that

$$\alpha_{Q,\mathcal{W}_{\mathtt{right}}}(v) \leq (1+\xi)^{2(k'-2)} \sum_{\substack{\{p,q\}\in E(G) \\ \text{s.t. } p\notin V(G'),q\in F^\star}} \sum_{\substack{\{a,b\}\in E(G) \\ \text{s.t. } a\in F^\star,b\in V(G')\setminus F^\star}} \beta(p)\cdot|\mathcal{P}_{q,a}^{G'[F^\star],k'/2,\mathcal{R}_{\mathtt{left}},\mathcal{W}_{\mathtt{left}}}|\cdot|\mathcal{P}_{b,v}^{G'-F^\star,k'/2,\mathcal{R}_{\mathtt{right}},\mathcal{W}_{\mathtt{right}}}|.$$

For each quadruple $Q=(h,f,\mathbf{p},F)\in(\mathscr{H},\mathscr{S},\{\mathscr{F}^{h,f,\mathbf{p}}\})|_{h\in\mathscr{H},f\in\mathscr{S},\mathbf{p}}$, we let $\mathcal{P}_{q,v,Q}^{G',k',\mathcal{R},\mathcal{W}}$ denote the collection of paths $P\in\mathcal{P}_{q,v}^{G',k',\mathcal{R},\mathcal{W}}$ that satisfy the following conditions:

1. $h^\star_{d+1}:=h\circ h^\star_d$ is injective when restricted to $V(P)$;
2. for all $j\in\{1,2,\ldots,t_{d+1}\}$, it holds that $|V(P)\cap\{v\in V(G'):(f\circ h^\star_{d+1})(v)=j\}|=\mathbf{p}(j)$;
3. the first $k'/2$ (closest to q) vertices of P belong $F^\star$, and the last (closest to v) vertices of P belong $F^\star$.

To proceed, observe that for each quadruple $Q=(h,f,\mathbf{p},F)\in(\mathscr{H},\mathscr{S},\{\mathscr{F}^{h,f,\mathbf{p}}\})|_{h\in\mathscr{H},f\in\mathscr{S},\mathbf{p}}$, it holds that

$$\sum_{(\mathcal{W}_{\mathtt{left}},\mathcal{W}_{\mathtt{right}})\text{ fits }\mathcal{W}} \sum_{\substack{\{a,b\}\in E(G) \\ \text{s.t. } a\in F^\star,b\in V(G')\setminus F^\star}} |\mathcal{P}_{q,a}^{G'[F^\star],k'/2,\mathcal{R}_{\mathtt{left}},\mathcal{W}_{\mathtt{left}}}|\cdot|\mathcal{P}_{b,v}^{G'-F^\star,k'/2,\mathcal{R}_{\mathtt{right}},\mathcal{W}_{\mathtt{right}}}| = |\mathcal{P}_{q,v,Q}^{G',k',\mathcal{R},\mathcal{W}}|.$$

Thus, we have that

$$\alpha(v)\leq(1+\xi)^{2(k'-2)}\sum_{\substack{\{p,q\}\in E(G) \\ \text{s.t. } p\notin V(G'),q\in V(G')}}\left(\beta(p)\sum_{Q=(h,f,\mathbf{p},F)\in(\mathscr{H},\mathscr{S},\{\mathscr{F}^{h,f,\mathbf{p}}\}|_{h\in\mathscr{H},f\in\mathscr{S},\mathbf{p}})}\frac{|\mathcal{P}_{q,v,Q}^{G',k',\mathcal{R},\mathcal{W}}|}{T\cdot T_{\mathbf{p}}}\right).$$

By Lemma 5.2, for each path $P\in\mathcal{P}_{q,v}^{G',k',\mathcal{R},\mathcal{W}}$, it holds that

- the number of distinct triples $(h,f,\mathbf{p})$ (from which the quadruples consist) with respect to which P satisfies Properties 1 and 2 above is upper bounded by $(1+xi)\cdot T$, and
- for each triple $(h,f,\mathbf{p})$ with respect to which P satisfies Properties 1 and 2 above, the number of sets $F\in\mathscr{F}^{h,f,\mathbf{p}}$ with respect to which P satisfies Property 3 above is upper bounded by $(1+\xi)\cdot T_{\mathbf{p}}$.

Thus, we have that

$$\sum_{Q=(h,f,\mathbf{p},F)\in(\mathscr{H},\mathscr{S},\{\mathscr{F}^{h,f,\mathbf{p}}\}|_{h\in\mathscr{H},f\in\mathscr{S},\mathbf{p}})}\frac{[P\in\mathcal{P}_{q,v,Q}^{G',k',\mathcal{R},\mathcal{W}}]}{T\cdot T_{\mathbf{p}}}\leq(1+\xi)^2,$$

where $[P\in\mathcal{P}_{q,v,Q}^{G',k',\mathcal{R},\mathcal{W}}]$ is 1 if $P\in\mathcal{P}_{q,v,Q}^{G',k',\mathcal{R},\mathcal{W}}$, and 0 otherwise. Since the choice of P was arbitrary, we get that

$$\begin{aligned}\alpha(v) &\leq (1+\xi)^{2(k'-2)}\sum_{\substack{\{p,q\}\in E(G) \\ \text{s.t. } p\notin V(G'),q\in V(G')}}\beta(p)(1+\xi)^2|\mathcal{P}_{q,v}^{G',k',\mathcal{R},\mathcal{W}}| \\ &= (1+\xi)^{2(k'-1)}\sum_{\substack{\{p,q\}\in E(G) \\ \text{s.t. } p\notin V(G'),q\in V(G')}}\beta(p)\cdot|\mathcal{P}_{q,v}^{G',k',\mathcal{R},\mathcal{W}}|.\end{aligned}$$

Symmetrically, we derive that

$$\alpha(v)\geq(1-\xi)^{2(k'-1)}\sum_{\substack{\{p,q\}\in E(G) \\ \text{s.t. } p\notin V(G'),q\in V(G')}}\beta(p)\cdot|\mathcal{P}_{q,v}^{G',k',\mathcal{R},\mathcal{W}}|.$$

This completes the proof. □

Finally, we conclude the proof of Theorem 6.1.

Theorem 6.1. *There is a deterministic $4^{k+\mathcal{O}(\sqrt{k}(\log^2 k+\log^2 \frac{1}{\epsilon}))}m\log n$-time polynomial-space algorithm that, given a graph G, a positive integer k and an accuracy value $0<\epsilon<1$, outputs a number y that satisfies $(1-\epsilon)x \leq y/2 \leq (1+\epsilon)x$ where x is the number of k-paths in G. In particular, if $\frac{1}{\epsilon}=2^{o(k^{\frac{1}{4}})}$, then the running time is $4^{k+o(k)}m\log n$.*

Proof. As in the proof of Theorem 4.2, for the sake of simplicity, we let G denote the graph obtained after the addition of s (rather than the input graph G). Then, it is sufficient to show the inequalities $(1-\epsilon)x \leq y \leq (1+\epsilon)x$ where x is the number of $(k+1)$-paths in G with s as an endpoint.

Moreover, observe that for all $v \in V(G)\setminus\{s\}$, it holds that

$$\sum_{\substack{\{p,q\}\in E(G)\\ \text{s.t. } p\notin V(G-\{s\}),q\in V(G-\{s\})}} \beta_{\text{init}}(p)\cdot|\mathcal{P}^{G,k,(),()}_{q,v}|$$

is equal to $|\mathcal{P}^{G,k+1}_{s,v}|$ denoted by $x^{G,k+1}_{s,v}$. Thus, by Lemma 6.1 with $G'=G$, $k'=k$ and $\beta=\beta_{\text{init}}$, we know that

- $\mathcal{A}(G,k,\beta_{\text{init}},(),())$ runs in time

  $$4^{k+\mathcal{O}(\sqrt{k}(\log^2 k+\log^2 \frac{1}{\xi}))}m\log n.$$

  Moreover, $\mathcal{A}(G,k,\beta_{\text{init}},(),())$ has a polynomial space complexity.

- For all $v\in V(G)$, the number $\alpha(v)$ assigned to v by the assignment α returned by $\mathcal{A}(G,k,\beta_{\text{init}},(),())$ satisfies the following inequalities:

  $$(1-\epsilon')^{2(k-1)}x^{G,k+1}_{s,v} \leq \alpha(v) \leq (1+\epsilon')^{2(k-1)}x^{G,k+1}_{s,v}.$$

First, by substituting ϵ' and since $\frac{\epsilon}{2}\leq\ln(1+\epsilon)\leq\epsilon$, we have that $\mathcal{A}(G,k,\beta_{\text{init}},(),())$ runs in time

$$4^{k+\mathcal{O}(\sqrt{k}(\log^2 k+\log^2 \frac{2(k-1)}{\ln(1+\epsilon)}))}m\log n \leq 4^{k+\mathcal{O}(\sqrt{k}(\log^2 k+\log^2 \frac{1}{\epsilon}))}m\log n.$$

Thus, we obtain the claimed running time.

For the accuracy, substituting ϵ' by ϵ, we have that for all $v\in V(G)$, it holds that

$$(1-\epsilon)x^{G,k+1}_{s,v} \leq (1-\frac{\ln(1+\epsilon)}{2(k-1)})^{2(k-1)}x^{G,k+1}_{s,v} \leq \alpha(v) \leq (1+\frac{\ln(1+\epsilon)}{2(k-1)})^{2(k-1)}x^{G,k+1}_{s,v} \leq (1+\epsilon)x^{G,k+1}_{s,v}.$$

Having these inequalities at hand, as in the proof of Theorem 4.2, we obtain that

$$y=\sum_{v\in V(G)\setminus\{s\}}\alpha(v) \leq \sum_{v\in V(G)\setminus\{s\}}(1+\epsilon)x^{G,k+1}_{s,v} = (1+\epsilon)\sum_{v\in V(G)\setminus\{s\}}x^{G,k+1}_{s,v} = (1+\epsilon)x.$$

Symmetrically, we obtain that $(1-\epsilon)x\leq y$. This completes the proof. □

7 Extensions and Other Applications

Finally, we briefly discuss extensions and other applications of our work. For the sake of illustration, we also present the proof of one of the claimed applications in detail (in Section 7.1).

Extensions. The framework of divide-and-color [16] is well known to solve (the decision version of) Subgraph Isomorphism where the pattern graph is a graph of bounded (by a fixed constant) pathwidth. In particular, our algorithms immediately extend to provide the following theorem.

Theorem 7.1. *Let $\epsilon > 0$. There is a deterministic $4^{k+\mathcal{O}(\sqrt{k}(\log^2 k+\log^2 \frac{1}{\epsilon}))}n^{t+\mathcal{O}(1)}$-time (resp. randomized $4^{k+\mathcal{O}(\log^2 k)}(\frac{1}{\epsilon})^{\mathcal{O}(\log k)}n^{t+\mathcal{O}(1)}$-time) $n^{t+\mathcal{O}(1)}$-space algorithm that, given a graph G and a graph H of pathwidth t on k vertices, outputs a number y that satisfies $(1-\epsilon)x \leq y \leq (1+\epsilon)x$ where x is the number of subgraphs of G isomorphic to H.*

When we deal with k-vertex paths (resp. k-vertex graphs of pathwidth t), we know that there exists a vertex (resp. $t+1$ vertices) whose removal results in two paths (resp. graphs of pathwidth t) on at most $k/2$ (resp. $k/2+t+1$) vertices. However, when we deal with trees (or, more generally, graphs of treewidth t), $k/2$ is replaced by $2k/3$. In turn, this means that at a recursive step, k is ensured to decrease by $k/3$ but not by $k/2$, hence the recursive formula to upper bound the running time of an algorithm yields a worse result. However, the "division into small trees" trick introduced by Fomin et al. [23] to solve the k-Tree problem (that is, the extension of k-Path where the pattern graph is a tree) easily resolves this issue and thereby extends divide-and-color to solve Subgraph Isomorphism where the pattern graph is a graph of bounded (by a fixed constant) treewidth with a negligible loss in time complexity. Roughly speaking, the idea behind this trick is as follows. Having a tree (or forest) T at hand, we do not remove only a single vertex, but $c = \mathcal{O}(1)$ vertices that result in the partition of T into $\mathcal{O}(1)$ small trees. Then, we can group these small trees together into two sets that correspond to two forests on "almost" $k/2$ vertices. The largest c is, the closer to $k/2$ these sizes are, yet additional bookkeeping (to remember which c vertices were selected) is required. For the sake of clarity, we briefly sketch the details of how this trick can be used to extend our improved randomized algorithm for #k-Path from Section 4.2 to solve #k-Tree (or, more generally, patterns of bounded treewidth) in Appendix A. By applying this trick to our deterministic algorithm for #k-Path from Section 6 in the same manner, we obtain the following theorem.

Theorem 7.2. *Let $\epsilon > 0$. There is a deterministic $4.001^{k+\mathcal{O}(\sqrt{k}(\log^2 \frac{1}{\epsilon}))}n^{t+\mathcal{O}(1)}$-time (resp. randomized $4.001^{k}(\frac{1}{\epsilon})^{\mathcal{O}(\log k)}n^{t+\mathcal{O}(1)}$-time)$n^{t+\mathcal{O}(1)}$-space algorithm that, given a graph G and a graph H of treewidth t on k vertices, outputs a number y that satisfies $(1-\epsilon)x \leq y \leq (1+\epsilon)x$ where x is the number of subgraphs of G isomorphic to H.*

The technical details of the extensions are those of completely standard bookkeeping on graphs of bounded treewidth (see, e.g., the Chapters on treewidth in [19], which teach dynamic programming over tree decompositions). Indeed, Alon and Gutner [4, 3] have skipped them altogether. Still, we provide some more details in Appendix A.

Other Applications. As mentioned earlier, the approach employed in Section 4.2 and Section 6 readily works for (essentially) all problems amenable to the framework of divide-and-color [15]. Thus, we obtain Theorem 1.1 (restated below) where the list of problems stated in it is illustrative rather than comprehensive.

Theorem 1.1. *The following problems admit deterministic $4^{k+\mathcal{O}(\sqrt{k}(\log^2 k+\log^2 \frac{1}{\epsilon}))}n^{\mathcal{O}(1)}$-time (resp. randomized $4^{k+\mathcal{O}(\log^2 k)}(\frac{1}{\epsilon})^{\mathcal{O}(\log k)}n^{\mathcal{O}(1)}$-time) FPT-ASs with polynomial space complexity:* (i) #Subgraph Isomorphism *for k-vertex subgraphs of treewidth $\mathcal{O}(1)$;* (ii) #q-Dimensional p-Matching *with $k = (q-1)p$;* (iii) #q-Set p-Packing *with $k = qp$;* (iv) #Graph Motif *and* #Module Motif *with $k = 2p$ where p is the motif size;* (v) #p-Internal Out-Branching *with $k = 2p$;* (vi) #Partial Cover *for k-element solutions.*[8]

[8]For problems (i) and (iv), the basis 4 is replaced by the basis 4.001 (or, more precisely, $4+\delta$ for any fixed constant $\delta > 0$).

7.1 Algorithms for #q-Dimensional p-Matching

In this section, we exemplify the additional applications of our approach by developing FPT-ASs for a problem on set families rather than graphs, called #q-Dimensional p-Matching, whose decision version has been extensively studied in Parameterized Complexity (see [19]). Here, the input consists of positive integers p and q, a partition $(U_1, U_2, \ldots, U_q)$ of a universe U, a family $\mathcal{S}$ of sets over U where $|\{S \cap U_i\}| = 1$ for every $S \in \mathcal{S}$ and $i \in \{1, 2, \ldots, q\}$, and an accuracy parameter $\epsilon > 0$. In this context, a p-packing in $\mathcal{S}$ is a subfamily of $\mathcal{S}$ that consists of p pairwise disjoint sets. Our goal is to approximately count the number of p-packings in $\mathcal{S}$, that is, if x is the (unknown) number of p-packings in $\mathcal{S}$, then the goal is to return a number y such that $(1-\epsilon)x \leq y \leq (1+\epsilon)x$.

In what follows, we choose an arbitrary order on U_1, hence comparison between elements in U_1 will be well defined. Additionally, we suppose w.l.o.g. that the smallest element in U_1, denoted by s, does not belong to any set in $\mathcal{S}$. (If this is not the case, we can add such an element as a dummy element and hence ensure this property). Let $n = |U|$ and $m = |\mathcal{S}|$.

7.1.1 Randomized FPT-AS for #q-Dimensional p-Matching

Algorithm. Let $k = (q-1)p$ and $\xi = \ln(1+\epsilon)/(k-1)$. Our algorithm, denoted by $\mathcal{A}$, is recursive. Every call to $\mathcal{A}$ has the form $\mathcal{A}(U', \mathcal{S}', p', \beta)$ where $U' \subseteq U$ and $U_1 \subseteq U'$, $\mathcal{S}'$ is a subfamily of $\mathcal{S}$ over U', $p' \in \{1, \ldots, p\}$ and $\beta : U_1 \to \mathbb{N}_0$.[9]

The output of a call $\mathcal{A}(U', \mathcal{S}', p', \beta)$ should be an assignment $\alpha : U_1 \to \mathbb{N}_0$ with the property that for each element $v \in U_1$, it holds that $\alpha(v)$ approximates the following number:

$$\sum_{\substack{u \in U_1 \\ \text{s.t. } u < v}} \beta(u) \cdot x_{u,v},$$

where $x_{u,v}$ is the number of p'-packings $\mathcal{H}$ in $\mathcal{S}'$ where the largest element in $(\bigcup \mathcal{H}) \cap U_1$ is v and the smallest element in $(\bigcup \mathcal{H}) \cap U_1$ is larger than u.

The initial call to the algorithm is with $U' = U$, $\mathcal{S}' = \mathcal{S}$, $p' = p$, and β that assigns 1 to s and 0 to all other elements in U_1. The final output is $\sum_{v \in U_1 \setminus \{s\}} \alpha(v)$.

We turn to describe a call $\mathcal{A}(U', \mathcal{S}', p', \beta)$. In the basis, where $p' = 1$, we return an assignment $\alpha : U_1 \to \mathbb{N}_0$ defined as follows: For each element $v \in U_1$, define

$$\alpha(v) = \sum_{\substack{u \in U_1 \\ \text{s.t. } u < v}} \beta(u) \cdot x_v.$$

Here, x_v is simply the number of sets in $\mathcal{S}'$ whose intersection with U_1 is $\{v\}$.

Now, suppose that $p' \geq 2$. Denote $k' = (q-1)p'$. By Theorem 3.1, for a ξ-parsimonious $(n, k'/2, k'/2)$-universal family $\mathscr{F}$ of sets over U, we can enumerate the sets $F \in \mathscr{F}$ with delay $\mathcal{O}(n)$. For each set $F \in \mathscr{F}$, we proceed as follows. We first recursively call $\mathcal{A}$ with $((U' \cap F) \cup U_1, \mathcal{S}'[F], p'/2, \beta)$ where $\mathcal{S}'[F] = \{S \in \mathcal{S}' : S \setminus U_1 \subseteq F\}$. Let γ_F be the output of this call. Then, we recursively call $\mathcal{A}$ with $((U' \setminus F) \cup U_1, \mathcal{S}'[U' \setminus F], p'/2, \gamma_F)$ where $\mathcal{S}'[U' \setminus F] = \{S \in \mathcal{S}' : (S \setminus U_1) \cap F = \emptyset\}$. Let α_F be the output of this recursive call.

Let T be the correction factor of $\mathscr{F}$. After all sets $F \in \mathscr{F}$ were enumerated, the output $\alpha : U_1 \to \mathbb{N}_0$ is computed as follows. For all $v \in U_1$, we calculate

$$\alpha(v) = \left(\sum_{F \in \mathscr{F}} \alpha_F(v) \right) / T.$$

[9] Roughly speaking, for each element $v \in U_1$, the value $\beta(v)$ would be an approximation of the number of $\widehat{p}$-packings $\mathcal{H}$ in $\widehat{\mathcal{S}}$ where the largest element in $(\bigcup \mathcal{H}) \cap U_1$ is v, for a certain integer $\widehat{p} \in \{1, 2, \ldots, p - p'\}$ and a subfamily $\widehat{\mathcal{S}} \subseteq \mathcal{S}$ that satisfies $(\bigcup \widehat{\mathcal{S}}) \cap (\bigcup \mathcal{S}') \subseteq U_1$.

Note that we do not store all the assignments α_F simultaneously, but we merely store one such assignment at a time and delete it immediately after $\alpha_F(v)/T$, for every $v \in U_1$, is added. This completes the description of $\mathcal{A}$.

Analysis. The main part of the analysis is done in the proof of the following lemma.

Lemma 7.1. *For some fixed constant $\eta > 0$, any call $\mathcal{A}(U', \mathcal{S}', p', \beta)$ satisfies the conditions below . Here, $k' = (q-1)p'$.*

- *$\mathcal{A}(U', \mathcal{S}', p', \beta)$ takes time $\eta^{\log k'} 4^{k'} k'^{\log k'} (\log n)^{\log k'} (n+qm)(\frac{1}{\xi^2})^{\log k'}$ and polynomial space.*
- *If all constructions of approximate universal families were successful, then for all $v \in U_1$, the number $\alpha(v)$ assigned to v by the assignment α returned by $\mathcal{A}(U', \mathcal{S}', p', \beta)$ satisfies the following inequalities:*

$$(1-\xi)^{k'-1} \cdot \left(\sum_{\substack{u \in U_1 \\ \text{s.t. } u<v}} \beta(u) \cdot x_{u,v} \right) \leq \alpha(v) \leq (1+\xi)^{k'-1} \cdot \left(\sum_{\substack{u \in U_1 \\ \text{s.t. } u<v}} \beta(u) \cdot x_{u,v} \right),$$

where $x_{u,v}$ is the number of p'-packings $\mathcal{H}$ in $\mathcal{S}'$ such that the largest element in $(\bigcup \mathcal{H}) \cap U_1$ is v and the smallest element in $(\bigcup \mathcal{H}) \cap U_1$ is larger than u.

Proof. Let $p' = p/2^d$. The proof is by backwards induction of d. In the basis, where $p' = 1$, the part of the claim that concerns the space complexity and the second condition trivially holds. For the running time bound, notice that the computation can be performed in time $\mathcal{O}(n+m)$. Indeed, to this end, we first compute x_v for all $v \in U_1$ in time $\mathcal{O}(n+m)$. Then, we iterate over U_1 from its smallest element to its largest element: when we consider an element v after we have just considered an element v', we can compute $\alpha(v)$ easily by observing that the computation stated in the specification of the algorithm is equivalent to $\alpha(v) = x_v \cdot (\beta(v') + \alpha(v')/x_{v'})$ in time $\mathcal{O}(1)$.

Now, let $d \leq \log_2 p - 1$, and suppose that the claim holds for $d+1$. Clearly, $\mathcal{A}(U', \mathcal{S}', p', \beta)$ has a polynomial space complexity. By Theorem 3.1, for some fixed constant $\lambda > 0$, we have that

$$|\mathscr{F}| \quad \leq \lambda \cdot 2^{k'} \cdot k' \log n \cdot \frac{1}{\xi^2}.$$

By the inductive hypothesis, for a fixed constant $\tau > 0$, the running time of $\mathcal{A}(U', \mathcal{S}', p', \beta)$ is upper bounded by

$$|\mathscr{F}| \cdot \left(2 \cdot \eta^{\log \frac{k'}{2}} 4^{\frac{k'}{2}} \left(\frac{k'}{2}\right)^{\log \frac{k'}{2}} (\log n)^{\log \frac{k'}{2}} (n+qm) \left(\frac{1}{\xi^2}\right)^{\log \frac{k'}{2}} + \tau(n+qm) \right).$$

In particular, the term $\tau(n+qm)$ above subsumes the computation of a single family $\mathcal{S}'[F]$ and a single family $\mathcal{S}'[U' \setminus F]$. Similarly to the proof of Theorem 4.2, by choosing $\eta = 10 \max\{\lambda, \tau\}$, we obtain that the running time of $\mathcal{A}(U', \mathcal{S}', p', \beta)$ is upper bounded by

$$\eta^{\log k'} 4^{k'} k'^{\log k'} (\log n)^{\log k'} (n+qm) \left(\frac{1}{\xi^2}\right)^{\log k'}.$$

Towards the proof of the second item of the claim, suppose that all constructions of approximate universal families were successful, and consider some $v \in U'$. For a subfamily $\widehat{S} \subseteq \mathcal{S}$ and $a, b \in U_1$, let $x_{a,b}^{\widehat{\mathcal{S}}}$ denote the number of $p'/2$-packings $\mathcal{H}$ in $\widehat{\mathcal{S}}$ where the largest element

in $(\bigcup \mathcal{H}) \cap U_1$ is v and the smallest element in $(\bigcup \mathcal{H}) \cap U_1$ is larger than u. By the inductive hypothesis, we have that

$$\begin{aligned}
\alpha(v) &= \frac{1}{T} \cdot \sum_{F \in \mathscr{F}} \alpha_F(v) \\
&\leq \frac{1}{T} \cdot \sum_{F \in \mathscr{F}} \left((1+\xi)^{\frac{k'}{2}-1} \cdot \sum_{\substack{b \in U_1 \\ \text{s.t. } b<v}} \gamma_F(b) \cdot x_{b,v}^{\mathcal{S}'[U' \setminus F]} \right) \\
&\leq (1+\xi)^{\frac{k'}{2}-1} \cdot \frac{1}{T} \cdot \sum_{F \in \mathscr{F}} \left(\sum_{\substack{b \in U_1 \\ \text{s.t. } b<v}} \left((1+\xi)^{\frac{k'}{2}-1} \sum_{\substack{a \in U_1 \\ \text{s.t. } a<b}} \beta(a) \cdot x_{a,b}^{\mathcal{S}'[F]} \right) \cdot x_{b,v}^{\mathcal{S}'[U' \setminus F]} \right) \\
&\leq (1+\xi)^{k'-2} \cdot \frac{1}{T} \cdot \sum_{F \in \mathscr{F}} \left(\sum_{\substack{a,b \in U_1 \\ \text{s.t. } a<b<v}} \beta(a) \cdot x_{a,b}^{\mathcal{S}'[F]} \cdot x_{b,v}^{\mathcal{S}'[U' \setminus F]} \right).
\end{aligned}$$

For any element $a \in U_1$, let $\mathcal{P}_{a,v}$ denote the set of p'-packings $\mathcal{H}$ in $\mathcal{S}'$ such that the largest element in $(\bigcup \mathcal{H}) \cap U_1$ is v and the smallest element in $(\bigcup \mathcal{H}) \cap U_1$ is larger than u. In addition, for any subset $F \subseteq U'$, let $\mathcal{P}_{a,v}[F]$ denote the set of packings $\mathcal{H} \in \mathcal{P}_{a,v}$ such that the $p'/2$ sets in $\mathcal{H}$ whose elements in U_1 are smallest have the property that their intersection with $U' \setminus U_1$ belongs to F, and the other $p'/2$ sets in $\mathcal{H}$ (whose elements in U_1 are largest) have the property that their intersection with $U' \setminus U_1$ has no element that belongs to F. Thus, we have that

$$\begin{aligned}
\alpha(v) &\leq (1+\xi)^{k'-2} \cdot \frac{1}{T} \cdot \sum_{F \in \mathscr{F}} \left(\sum_{\substack{a \in U_1 \\ \text{s.t. } a<v}} \beta(a) \cdot |\mathcal{P}_{a,v}[F]| \right) \\
&= (1+\xi)^{k'-2} \cdot \frac{1}{T} \cdot \sum_{\substack{a \in U_1 \\ \text{s.t. } a<v}} \left(\beta(a) \cdot \sum_{F \in \mathscr{F}} |\mathcal{P}_{a,v}[F]| \right).
\end{aligned}$$

Since $\mathscr{F}$ is a ξ-parsimonious $(n, k'/2, k'/2)$-universal family, for any element $a \in U_1$ and packing $\mathcal{H} \in \mathcal{P}_{a,v}$, the number of sets $F \in \mathscr{F}$ such that $\mathcal{H} \in \mathcal{P}_{a,v}[F]$ is upper bounded by $(1+\xi)T$. Thus, we have that

$$\begin{aligned}
\alpha(v) &\leq (1+\xi)^{k'-2} \cdot \frac{1}{T} \cdot \sum_{\substack{a \in U_1 \\ \text{s.t. } a<v}} \beta(a) \cdot (1+\xi) T |\mathcal{P}_{a,v}| \\
&= (1+\xi)^{k'-1} \cdot \left(\sum_{\substack{a \in U_1 \\ \text{s.t. } a<v}} \beta(a) \cdot x_{a,v} \right).
\end{aligned}$$

Symmetrically, we derive that $(1-\xi)^{k'-1} \cdot \left(\sum_{\substack{a \in U_1 \\ \text{s.t. } a<v}} \beta(a) \cdot x_{a,v} \right) \leq \alpha(v)$. This completes the proof. □

Now, we conclude the proof of correctness of our algorithm.

Theorem 7.3. *There is a randomized $(4^{k+o(k)}(n+m)+(n+m)n^{o(1)})(\frac{1}{\epsilon})^{\mathcal{O}(\log k)}$-time polynomial-space algorithm that, given positive integers p and q where $k = (q-1)p$, a partition $(U_1, U_2, \ldots, U_q)$ of a universe U, a family $\mathcal{S}$ of sets over U where $|\{S \cap U_i\}| = 1$ for every $S \in \mathcal{S}$ and*

$i \in \{1, 2, \ldots, q\}$, and an accuracy value $0 < \epsilon < 1$, outputs a number y that (with high probability) satisfies $(1-\epsilon)x \leq y \leq (1+\epsilon)x$ where x is the number of p-packings in $\mathcal{S}$. In particular, if $\frac{1}{\epsilon} = 2^{o(k/\log k)}$, then the running time is $4^{k+o(k)}(n+m)n^{o(1)}$.

Proof. Let β_{init} be the assignment $\beta_{\text{init}} : \{s\} \to \mathbb{N}_0$ that assigns 1 to s and 0 to all other elements in U_1. Observe that for all $v \in U_1 \setminus \{s\}$, it holds that $\sum_{\substack{u \in U_1 \\ \text{s.t. } u<v}} \beta_{\text{init}}(u) \cdot x_{u,v}$ is equal to $x_{s,v}$. Here, for any two elements $a, b \in U_1$, $x_{a,b}$ is the number of p-packings $\mathcal{H}$ in $\mathcal{S}$ such that the largest element in $(\bigcup \mathcal{H}) \cap U_1$ is a and the smallest element in $(\bigcup \mathcal{H}) \cap U_1$ is larger than b. In particular, by the choice of s, $x_{s,v}$ is the number of p-packings $\mathcal{H}$ in $\mathcal{S}$ such that the largest element in $(\bigcup \mathcal{H}) \cap U_1$ is v. Thus, by Lemma 7.1 with $U' = U$, $\mathcal{S}' = \mathcal{S}$, $p' = p$ and $\beta = \beta_{\text{init}}$, we know that

- $\mathcal{A}(U, \mathcal{S}, p, \beta_{\text{init}})$ takes time $4^{k+\mathcal{O}(\log^2 k)}(\log n)^{\log k}(n+m)(\frac{1}{\xi})^{\log k}$ and polynomial space.
- If all constructions of approximate universal families were successful, then for all $v \in U_1$, the number $\alpha(v)$ assigned to v by the assignment α returned by $\mathcal{A}(U, \mathcal{S}, p, \beta_{\text{init}})$ satisfies $(1-\xi)^{k-1}x_{s,v} \leq \alpha(v) \leq (1+\xi)^{k-1}x_{s,v}$.

As in the proof of Theorem 4.1, $4^{k+\mathcal{O}(\log^2 k)}(\log n)^{\log k}(n+m)(\frac{1}{\xi})^{\log k} = (4^{k+o(k)}(n+m) + (n+m)n^{o(1)})(\frac{1}{\epsilon})^{\mathcal{O}(\log k)}$. Moreover, as in the proof of Theorem 4.1, with high probability (say, higher than 9/10), all constructions of approximate universal families were successful. Thus, we know that for all $v \in U_1$, it holds that $(1-\xi)^{k-1}x_{s,v} \leq \alpha(v) \leq (1+\xi)^{k-1}x_{s,v}$. By substituting ξ as in the proof of Theorem 4.1, for all $v \in U_1$, it follows that $(1-\epsilon)x_{s,v} \leq \alpha(v) \leq (1+\epsilon)x_{s,v}$.

We thus obtain that

$$y = \sum_{v \in U_1} \alpha(v) \leq \sum_{v \in U_1} (1+\epsilon)x_{s,v} = (1+\epsilon) \sum_{v \in U_1} x_{s,v} = (1+\epsilon)x.$$

Symmetrically, we obtain that $(1-\epsilon)x \leq y$. This completes the proof. □

The time complexity bound above can be easily reduced to $4^{k+\mathcal{O}(\log^2 k)}(n+m)\log n(\frac{1}{\epsilon})^{\mathcal{O}(\log k)}$. Indeed, it is only required to utilize Proposition 5.2 to reduce the universe size from n to k^2 before calling the algorithm in Theorem 7.3; then, the term that gives rise to $n^{o(1)}$ above is subsumed by $4^{\mathcal{O}(\log^2 k)}$. Since this is precisely what our deterministic algorithm does to obtain such dependency, we do not repeat these details and directly state the result.

Corollary 7.1. *There is a randomized $4^{k+\mathcal{O}(\log^2 k)}(n+m)\log n(\frac{1}{\epsilon})^{\mathcal{O}(\log k)}$-time polynomial-space algorithm that, given positive integers p and q where $k = (q-1)p$, a partition $(U_1, U_2, \ldots, U_q)$ of a universe U, a family $\mathcal{S}$ of sets over U where $|\{S \cap U_i\}| = 1$ for every $S \in \mathcal{S}$ and $i \in \{1, 2, \ldots, q\}$ and an accuracy value $0 < \epsilon < 1$, outputs a number y that (with high probability) satisfies $(1-\epsilon)x \leq y \leq (1+\epsilon)x$ where x is the number of p-packings in $\mathcal{S}$. In particular, if $\frac{1}{\epsilon} = 2^{o(k/\log k)}$, then the running time is $4^{k+o(k)}(n+m)\log n$.*

7.1.2 Deterministic FPT-AS for #q-Dimensional p-Matching

Algorithm Specification. Let $k = (q-1)p$ and $\xi = \frac{\ln(1+\epsilon)}{2(k-1)}$. Our algorithm, denoted by $\mathcal{A}$, is recursive. We first specify the arguments with which $\mathcal{A}$ is called. Then, we specify the goal of a recursive call. Afterwards, we note how $\mathcal{A}$ is initially called, and then we describe the computation that is executes.

Arguments. Each call to $\mathcal{A}$ is of the form $\mathcal{A}(U', \mathcal{S}', p', \beta, \mathcal{R}, \mathcal{W})$ where $U' \subseteq U$ and $U_1 \subseteq U'$, $\mathcal{S}'$ is a subfamily of $\mathcal{S}$ over U', $p' = p/2^d$ for some $d \in \{0, 1, \ldots, \log_2 k\}$, $\beta : U_1 \to \mathbb{N}_0$, and $\mathcal{R} = ((h_1, f_1, \mathbf{p}_1, \sigma_1), (h_2, f_2, \mathbf{p}_2, \sigma_2), \ldots, (h_d, f_d, \mathbf{p}_d, \sigma_d))$. (The argument $\mathcal{W}$ is defined later.) For each $i \in \{1, 2, \ldots, d\}$, it holds that

- $k_i := \frac{k}{2^i}$, $z_i := \frac{2{k_i}^2}{\xi}$, $t_i := \sqrt{k_i}$ and $s_i := k_i/t_i$.
- h_i is a function from $\{1, 2, \ldots, z_{i-1}\}$ to $\{1, 2, \ldots, z_i\}$ if $i \geq 2$, and from $U \setminus U_1$ to $\{1, 2, \ldots, z_1\}$ otherwise,
- f_i is a function from $\{1, 2 \ldots, z_i\}$ to $\{1, 2, \ldots, t_i\}$,
- $\mathbf{p}_i$ is a function from $\{1, 2, \ldots, t_i\}$ to $\{1, 2, \ldots, s_i\}$ such that $\sum_{j=1}^{t_i} \mathbf{p}_i(j) = k_i/2$, and
- $\sigma_i \in \{\texttt{left}, \texttt{right}\}$.

Additionally, define $z_0 = n$.

Towards the description of $\mathcal{W}$, we introduce the following definition.

Definition 7.1. *For the arguments $U', \mathcal{S}', p'$ and $\mathcal{R}$ of a call to $\mathcal{A}$, the* collection of relevant tuples $\mathsf{W}_{U',\mathcal{S}',p',\mathcal{R}}$ *is the collection of all tuples $\mathcal{W} = (\mathbf{w}_1, \mathbf{w}_2, \ldots, \mathbf{w}_d)$ such that for each $i \in \{1, 2, \ldots, d\}$, the following condition holds:*

- *For each $j \in \{1, 2, \ldots, t_i\}$, if $\sigma_i = \texttt{left}$ then $\mathbf{w}_i(j) \leq \mathbf{p}_i(j)$, and otherwise $\mathbf{w}_i(j) \leq s_i - \mathbf{p}_i(j)$.*

In case $d = 0$ (that is, the initial call), $\mathsf{W}_{U,\mathcal{S},p,()} = \{()\}$.

Having this definition at hand, we note that the argument $\mathcal{W}$ in a call $\mathcal{A}(U', \mathcal{S}', p', \beta, \mathcal{R}, \mathcal{W})$ is a tuple from $\mathsf{W}_{U',\mathcal{S}',p',\mathcal{R}}$.

Output Value. Before we state the value returned by a call, we define which packings "comply" with the demands imposed by $\mathcal{R}$ and $\mathcal{W}$ precise. Since our parsimonious universal tuples only work properly for nice sets, it will be crucial that we only count paths that fit this definition.

Definition 7.2. *For a call $\mathcal{A}(U', \mathcal{S}', p', \beta, \mathcal{R}, \mathcal{W})$ with $\mathcal{W} = (\mathbf{w}_1, \mathbf{w}_2, \ldots, \mathbf{w}_d) \in \mathsf{W}_{U',\mathcal{S}',p',\mathcal{R}}$, and elements $u, v \in U_1$, the* collection of relevant packings $\mathcal{P}_{u,v}^{U',\mathcal{S}',p',\mathcal{R},\mathcal{W}}$ *is the collection of all p'-packings $\mathcal{H}$ in $\mathcal{S}'$ where the largest element in $(\bigcup \mathcal{H}) \cap U_1$ is v and the smallest element in $(\bigcup \mathcal{H}) \cap U_1$ is larger than u, such that the following conditions holds:*

- *For each $i \in \{1, 2, \ldots, d\}$, denote $h_i^\star := (h_i \circ \cdots \circ (h_3 \circ (h_2 \circ h_1)))$. Then, the function $h_d^\star$ is injective when restricted to $(\bigcup \mathcal{H}) \setminus U_1$.*
- *For each $i \in \{1, 2, \ldots, d\}$, denote $f_i^\star := f_i \circ h_i^\star$. Then, for all $i \in \{1, 2, \ldots, d\}$ and $j \in \{1, 2, \ldots, t_i\}$, it holds that $|\{v \in (\bigcup \mathcal{H}) \setminus U_1 : f_i^\star(v) = j\}| = \mathbf{w}_i(j)$.*

Note that for $\mathcal{W} = ()$, $\mathcal{P}_{u,v}^{U',\mathcal{S}',p',\mathcal{R},\mathcal{W}}$ is the collection of all p'-packings $\mathcal{H}$ in $\mathcal{S}'$ where the largest element in $(\bigcup \mathcal{H}) \cap U_1$ is v and the smallest element in $(\bigcup \mathcal{H}) \cap U_1$ is larger than u.

The value returned by each call $\mathcal{A}(U', \mathcal{S}', p', \beta, \mathcal{R}, \mathcal{W})$ is an assignment $\alpha : U_1 \to \mathbb{N}_0$ with the following property: For each vertex $v \in U_1$, it holds that $\alpha(v)$ approximates the following number:

$$\sum_{\substack{u \in U_1 \\ \text{s.t. } u < v}} \beta(u) \cdot |\mathcal{P}_{u,v}^{U',\mathcal{S}',p',\mathcal{R},\mathcal{W}}|.$$

Initial Call. The initial call is $\mathcal{A}(U, \mathcal{S}, p, \beta_{\text{init}}, (), ())$ (that is, $\mathcal{R}$ and $\mathcal{W}$ are 0-length tuples), where β_{init} assigns 1 to s and 0 to every other vertex in U_1. The final output, returned by the initial call, is $\sum_{v \in U_1} \alpha(v)$.

Computation: Basis. Now, we turn to describe a call $\mathcal{A}(U', \mathcal{S}', p', \beta, \mathcal{R}, \mathcal{W})$. In the basis, where $p' = 1$, for every element $v \in U_1$, we compute the integer $\alpha(v)$ as follows. First, we compute $\mathcal{P}_v^{U',\mathcal{S}',\mathcal{R},\mathcal{W}}$, defined to be the collection of sets $S \in \mathcal{S}'$ that satisfy the following conditions: **(i)** $S \cap U_1 = \{v\}$; **(ii)** $h_d^\star$ is injective when restricted to $S \setminus U_1$; **(iii)** for all $i \in \{1, 2, \ldots, d\}$ and $j \in \{1, 2, \ldots, t_i\}$, it holds that $|\{v \in S \setminus U_1 : f_i^\star(v) = j\}| = \mathbf{w}_i(j)$. Then, we set $\alpha(v) = |\mathcal{P}_v^{U',\mathcal{S}',\mathcal{R},\mathcal{W}}|$.

Computation: Step. Next, suppose that $p' = p/2^d \geq 2$. Denote $\mathcal{R} = ((h_1, f_1, \mathbf{p}_1, \sigma_1), (h_2, f_2, \mathbf{p}_2, \sigma_2), \ldots, (h_d, f_d, \mathbf{p}_d, \sigma_d))$ and $\mathcal{W} = (\mathbf{w}_1, \mathbf{w}_2, \ldots, \mathbf{w}_d)$. Let $k' = (q-1)p'$. By Theorem 5.1, for an ξ-parsimonious $(z_d, k'/2, k'/2)$-universal tuple $(\mathscr{H}, \mathscr{S}, \{\mathscr{F}^{h,f,\mathbf{p}}\}|_{h\in\mathscr{H}, f\in\mathscr{S}, \mathbf{p}})$ with universe $\{1, 2, \ldots, z_d\}$,[10] after spending preprocessing time $\frac{{k'}^{\mathcal{O}(1)} z_d \log z_d}{\xi^{\mathcal{O}(1)}}$, we enumerate the ℓ quadruples $Q = (h, f, \mathbf{p}, F) \in (\mathscr{H}, \mathscr{S}, \{\mathscr{F}^{h,f,\mathbf{p}}\}|_{h\in\mathscr{H}, f\in\mathscr{S}, \mathbf{p}})$ with delay Δ, where

$$\begin{aligned} \ell &= 2^{k' + \mathcal{O}(\sqrt{k'}(\log^2 k' + \log^2 \frac{1}{\xi}))} \log z_d, \text{ and} \\ \Delta &= 2^{\mathcal{O}(\sqrt{k'}(\log^2 k' + \log^2 \frac{1}{\xi}))}. \end{aligned}$$

For each quadruple $Q = (h, f, \mathbf{p}, F)$, we proceed as follows.

- Denote $F^\star = (h \circ h_d^\star)^{-1}(F) \cap (U' \setminus U_1)$, that is, $F^\star$ is the set of elements in $U' \setminus U_1$ that $h \circ h_d^\star$ maps to integers in F.
- Denote $\mathcal{R}_{\mathtt{left}} = \mathcal{R} + ((h, f, \mathbf{p}, \mathtt{left}))$, and $\mathcal{R}_{\mathtt{right}} = \mathcal{R} + ((h, f, \mathbf{p}, \mathtt{right}))$.
- Denote $\mathcal{S}'[F^\star] = \{S \in \mathcal{S}' : S \setminus U_1 \subseteq F^\star\}$, and $\mathcal{S}'[U' \setminus F^\star] = \{S \in \mathcal{S}' : (S \setminus U_1) \cap F^\star = \emptyset\}$.
- For all $\mathcal{W}_{\mathtt{left}} = (\mathbf{w}_1^{\mathtt{left}}, \mathbf{w}_2^{\mathtt{left}}, \ldots, \mathbf{w}_{d+1}^{\mathtt{left}}) \in \mathsf{W}_{F^\star \cup U_1, \mathcal{S}'[F^\star], p'/2, \mathcal{R}_{\mathtt{left}}}$ and $\mathcal{W}_{\mathtt{right}} = (\mathbf{w}_1^{\mathtt{right}}, \mathbf{w}_2^{\mathtt{right}}, \ldots, \mathbf{w}_{d+1}^{\mathtt{right}}) \in \mathsf{W}_{U' \setminus F^\star, \mathcal{S}'[U' \setminus F^\star], p'/2, \mathcal{R}_{\mathtt{right}}}$, we say that $(\mathcal{W}_{\mathtt{left}}, \mathcal{W}_{\mathtt{right}})$ *fits* $\mathcal{W}$ if two conditions hold:
  1. for all $i \in \{1, \ldots, d\}$ and $j \in \{1, \ldots, t_i\}$, it holds that $\mathbf{w}_i(j) = \mathbf{w}_i^{\mathtt{left}}(j) + \mathbf{w}_i^{\mathtt{right}}(j)$;
  2. for all $j \in \{1, \ldots, t_{d+1}\}$, it holds that $\mathbf{p}(j) = \mathbf{w}_{d+1}^{\mathtt{left}}(j)$ and $s_{d+1} - \mathbf{p}(j) = \mathbf{w}_{d+1}^{\mathtt{right}}(j)$.

  Note that if $\mathcal{W} = ()$, then the first condition is vacuously satisfied.

Having established the notations above, for every $\mathcal{W}_{\mathtt{left}} \in \mathsf{W}_{F^\star \cup U_1, \mathcal{S}'[F^\star], p'/2, \mathcal{R}_{\mathtt{left}}}$, we perform two recursive calls as follows:

1. First, we call $\mathcal{A}$ with $(F^\star \cup U_1, \mathcal{S}'[F^\star], p'/2, \beta, \mathcal{R}_{\mathtt{left}}, \mathcal{W}_{\mathtt{left}})$. Let $\alpha_{Q,\mathcal{W}_{\mathtt{left}}}$ be the output of this call.
2. Second, we call $\mathcal{A}$ with $(U' \setminus F^\star, \mathcal{S}'[U' \setminus F^\star], p'/2, \alpha_{Q,\mathcal{W}_{\mathtt{left}}}, \mathcal{R}_{\mathtt{right}}, \mathcal{W}_{\mathtt{right}})$ where $\mathcal{W}_{\mathtt{right}}$ is the unique tuple in $\mathsf{W}_{U' \setminus F^\star, \mathcal{S}'[U' \setminus F^\star], p'/2, \mathcal{R}_{\mathtt{right}}}$ such that $(\mathcal{W}_{\mathtt{left}}, \mathcal{W}_{\mathtt{right}})$ fits $\mathcal{W}$. Let $\alpha_{Q,\mathcal{W}_{\mathtt{right}}}$ be the output of this recursive call.

After all quadruples Q were enumerated, the output $\alpha : U_1 \to \mathbb{N}_0$ is computed as follows. First, for all $v \in U_1$ and a quadruple $Q = (h, f, \mathbf{p}, F)$, we define

$$\alpha_Q(v) = \sum_{(\mathcal{W}_{\mathtt{left}}, \mathcal{W}_{\mathtt{right}}) \text{ fits } \mathcal{W}} \alpha_{Q,\mathcal{W}_{\mathtt{right}}}(v).$$

Note that we do not store all the assignments $\alpha_{Q,\mathcal{W}_{\mathtt{left}}}, \alpha_{Q,\mathcal{W}_{\mathtt{right}}}$ and α_Q simultaneously, but we only need to store one such assignment at a time in order to compute the assignment mentioned below "on the go".

[10] For $d = 0$, we refer to the universe $U \setminus U_1$.

Let $T > 0$ and for every $\mathbf{p}$, $T_{\mathbf{p}} > 0$, be the correction factors of $(\mathscr{H}, \mathscr{S}, \{\mathscr{F}^{h,f,\mathbf{p}}\}|_{h\in\mathscr{H}, f\in\mathscr{S},\mathbf{p}})$ (see Lemma 5.2). For all $v \in U_1$, we calculate

$$\alpha(v) = \sum_{Q=(h,f,\mathbf{p},F)\in(\mathscr{H},\mathscr{S},\{\mathscr{F}^{h,f,\mathbf{p}}\}|_{h\in\mathscr{H},f\in\mathscr{S},\mathbf{p}})} \frac{\alpha_Q(v)}{T\cdot T_{\mathbf{p}}}.$$

This completes the description of $\mathcal{A}$.

Analysis. The main part of the analysis is done in the proof of the following lemma.

Lemma 7.2. *There exists a fixed constant $\eta > 0$ such that for any call $\mathcal{A}(U', \mathcal{S}', p', \beta, \mathcal{R}, \mathcal{W})$ where $(q-1)p' = k' = k/2^d$, the following conditions hold.*

- *$\mathcal{A}(U', \mathcal{S}', p', \beta, \mathcal{R}, \mathcal{W})$ runs in time*

$$4^{k'+\eta\sqrt{k'}(\log^2 k'+\log^2\frac{1}{\xi})}\cdot 8^{\log k'\cdot\sqrt{k}\log k}\cdot M,$$

  where $M = (n+qm)$ if $k' < k$ and $M = (n+qm)\log n$ otherwise (i.e., $k' = k$). Moreover, $\mathcal{A}(U', \mathcal{S}', p', \beta, \mathcal{R}, \mathcal{W})$ has a polynomial space complexity.
- *For all $v \in U_1$, the number $\alpha(v)$ assigned to v by the assignment α returned by $\mathcal{A}(U', \mathcal{S}', p', \beta, \mathcal{R}, \mathcal{W})$ satisfies the following inequalities:*

$$\alpha(v) \geq (1-\xi)^{2(k'-1)} \sum_{\substack{u\in U_1\\ \text{s.t. } u<v}} \beta(u)\cdot|\mathcal{P}_{u,v}^{U',\mathcal{S}',p',\mathcal{R},\mathcal{W}}|.$$
$$\alpha(v) \leq (1+\xi)^{2(k'-1)} \sum_{\substack{u\in U_1\\ \text{s.t. } u<v}} \beta(u)\cdot|\mathcal{P}_{u,v}^{U',\mathcal{S}',p',\mathcal{R},\mathcal{W}}|.$$

Proof. The proof is by induction of d. In the basis, where $p' = 1$, the claim easily holds. Now, let $d \leq \log_2 p - 1$, and suppose that the claim holds for $d+1$.

Time and Space Complexities. It is clear that $\mathcal{A}(U', \mathcal{S}', p', \beta, \mathcal{R}, \mathcal{W})$ has a polynomial space complexity. Note that $z_d = \mathcal{O}(n)$ if $d = 0$, and $z_d = \mathcal{O}(k')$ otherwise. Denote $N = n$ if $d = 0$, and $N = 2$ (so $\log N = 1$) otherwise. Then, after preprocessing time bounded by $\frac{k'^{\mathcal{O}(1)} N\log N}{\xi^{\mathcal{O}(1)}}$, for some fixed constant $\lambda > 0$, the ℓ quadruples $Q = (h, f, \mathbf{p}, F) \in (\mathscr{H}, \mathscr{S}, \{\mathscr{F}^{h,f,\mathbf{p}}\}|_{h\in\mathscr{H},f\in\mathscr{S},\mathbf{p}})$ are enumerated with delay Δ, where

$$\begin{aligned}\ell &= 2^{k'+\lambda\sqrt{k'}(\log^2 k'+\log^2\frac{1}{\xi})}\log N, \text{ and}\\ \Delta &= 2^{\lambda\sqrt{k'}(\log k'+\log\frac{1}{\xi})}.\end{aligned}$$

The number of pairs $(\mathcal{W}_{\texttt{left}}, \mathcal{W}_{\texttt{right}})$ that fit $\mathcal{W}$, which equals $|\mathsf{W}_{F^\star\cup U_1,\mathcal{S}'[F^\star],p'/2,\mathcal{R}_{\texttt{left}}}|$, is upper bounded by

$$r := \prod_{i=1}^{d}(s_i+1)^{t_i} = \prod_{i=1}^{d}(\sqrt{\frac{k}{2^i}}+1)^{\sqrt{\frac{k}{2^i}}} \leq \prod_{i=1}^{\log k}(\sqrt{k})^{\sqrt{\frac{k}{2^i}}} < 2^{\sqrt{k}\log k\cdot\sum_{i=1}^{\log k}(\frac{1}{\sqrt{2}})^i} < 8^{\sqrt{k}\log k}.$$

Thus, by the inductive hypothesis, for some fixed constant $\zeta > 0$, the running time of $\mathcal{A}(U', \mathcal{S}', p', \beta, \mathcal{R}, \mathcal{W})$ is upper bounded by

$$\begin{aligned}&\ell\cdot\left(\zeta\cdot r\cdot 4^{\frac{k'}{2}+\eta\sqrt{\frac{k'}{2}}(\log^2\frac{k'}{2}+\log^2\frac{1}{\xi})}8^{\log\frac{k'}{2}\cdot\sqrt{k}\log k}(n+qm)\right)\\ \leq\ & 2^{k'+\lambda\sqrt{k'}(\log^2 k'+\log^2\frac{1}{\xi})}\log N\cdot\zeta\cdot 8^{\sqrt{k}\log k}\cdot 4^{\frac{k'}{2}+\eta\sqrt{\frac{k'}{2}}(\log^2 k'+\log^2\frac{1}{\xi})}8^{(\log k'-1)\sqrt{k}\log k}(n+qm)\\ =\ & \zeta 4^{k'+\lambda\sqrt{k'}(\log^2 k'+\log^2\frac{1}{\xi})+\frac{\eta}{\sqrt{2}}\sqrt{k'}(\log^2\frac{k'}{2}+\log^2\frac{1}{\xi})}\cdot 8^{\log k'\cdot\sqrt{k}\log k}\cdot M.\end{aligned}$$

Thus, by choosing η to be a large enough constant that depends only on ζ and λ (say, $\eta = 10\max\{\zeta,\lambda\}$), the expression above is upper bounded by

$$4^{k'+\eta\sqrt{k'}(\log^2 k'+\log^2\frac{1}{\xi})}\cdot 8^{\log k'\cdot\sqrt{k}\log k}\cdot M.$$

Accuracy. Towards the proof of the second item of the claim, consider some vertex $v\in U_1$. By definition, we have that

$$\alpha(v)=\sum_{Q=(h,f,\mathbf{p},F)\in(\mathscr{H},\mathscr{S},\{\mathscr{F}^{h,f,\mathbf{p}}\}|_{h\in\mathscr{H},f\in\mathscr{S},\mathbf{p}})}\ \sum_{(\mathcal{W}_{\mathtt{left}},\mathcal{W}_{\mathtt{right}})\text{ fits }\mathcal{W}}\frac{\alpha_{Q,\mathcal{W}_{\mathtt{right}}}(v)}{T\cdot T_{\mathbf{p}}}.$$

By the inductive hypothesis, for each term $\alpha^{Q,\mathcal{W}_{\mathtt{right}}}(v)$ in the sum above, we have that

$$\alpha^{Q,\mathcal{W}_{\mathtt{right}}}(v)\leq(1+\xi)^{k'-2}\sum_{\substack{b\in U_1\\ \text{s.t. } b<v}}\alpha^{Q,\mathcal{W}_{\mathtt{left}}}(b)\cdot|\mathcal{P}_{b,v}^{U'\setminus F^\star,\mathcal{S}'[U'\setminus F^\star],p'/2,\mathcal{R}_{\mathtt{right}},\mathcal{W}_{\mathtt{right}}}|.$$

By applying the inductive hypothesis again, we have that

$$\alpha^{Q,\mathcal{W}_{\mathtt{right}}}(v)\leq$$
$$(1+\xi)^{2(k'-2)}\sum_{\substack{a,b\in U_1\\ \text{s.t. } a<b<v}}\beta(a)\cdot|\mathcal{P}_{a,b}^{F^\star\cup U_1,\mathcal{S}'[F^\star],p'/2,\mathcal{R}_{\mathtt{left}},\mathcal{W}_{\mathtt{left}}}|\cdot|\mathcal{P}_{b,v}^{U'\setminus F^\star,\mathcal{S}'[U'\setminus F^\star],p'/2,\mathcal{R}_{\mathtt{right}},\mathcal{W}_{\mathtt{right}}}|.$$

For each quadruple $Q=(h,f,\mathbf{p},F)\in(\mathscr{H},\mathscr{S},\{\mathscr{F}^{h,f,\mathbf{p}}\})|_{h\in\mathscr{H},f\in\mathscr{S},\mathbf{p}}$, we let $\mathcal{P}_{a,v,Q}^{U',\mathcal{S}',p',\mathcal{R},\mathcal{W}}$ denote the collection of packings $\mathcal{H}\in\mathcal{P}_{a,v}^{U',\mathcal{S}',p',\mathcal{R},\mathcal{W}}$ that satisfy the following conditions:

1. $h_{d+1}^\star:=h\circ h_d^\star$ is injective when restricted to $(\bigcup\mathcal{H})\setminus U_1$;
2. for all $j\in\{1,2,\ldots,t_{d+1}\}$, it holds that $|\{v\in(\bigcup\mathcal{H})\setminus U_1:(f\circ h_{d+1}^\star)(v)=j\}|=\mathbf{p}(j)$;
3. the $p'/2$ sets in $\mathcal{H}$ whose elements in U_1 are smallest have the property that their intersection with $U'\setminus U_1$ belongs to $F^\star$, and the other $p'/2$ sets in $\mathcal{H}$ (whose elements in U_1 are largest) have the property that their intersection with $U'\setminus U_1$ has no element that belongs to $F^\star$.

To proceed, observe that for each quadruple $Q=(h,f,\mathbf{p},F)\in(\mathscr{H},\mathscr{S},\{\mathscr{F}^{h,f,\mathbf{p}}\})|_{h\in\mathscr{H},f\in\mathscr{S},\mathbf{p}}$, it holds that

$$\sum_{(\mathcal{W}_{\mathtt{left}},\mathcal{W}_{\mathtt{right}})\text{ fits }\mathcal{W}}\ \sum_{\substack{b\in U_1\\ \text{s.t. } a<b<v}}|\mathcal{P}_{a,b}^{F^\star\cup U_1,\mathcal{S}'[F^\star],p'/2,\mathcal{R}_{\mathtt{left}},\mathcal{W}_{\mathtt{left}}}|\cdot|\mathcal{P}_{b,v}^{U'\setminus F^\star,\mathcal{S}'[U'\setminus F^\star],p'/2,\mathcal{R}_{\mathtt{right}},\mathcal{W}_{\mathtt{right}}}|$$
$$=|\mathcal{P}_{a,v,Q}^{U',\mathcal{S}',p',\mathcal{R},\mathcal{W}}|.$$

Thus, we have that

$$\alpha(v)\leq(1+\xi)^{2(k'-2)}\sum_{\substack{a\in U_1\\ \text{s.t. } a<v}}\left(\beta(a)\sum_{Q=(h,f,\mathbf{p},F)\in(\mathscr{H},\mathscr{S},\{\mathscr{F}^{h,f,\mathbf{p}}\}|_{h\in\mathscr{H},f\in\mathscr{S},\mathbf{p}})}\frac{|\mathcal{P}_{a,v,Q}^{U',\mathcal{S}',p',\mathcal{R},\mathcal{W}}|}{T\cdot T_{\mathbf{p}}}\right).$$

By Lemma 5.2, for each packing $\mathcal{H}\in\mathcal{P}_{a,v}^{U',\mathcal{S}',p',\mathcal{R},\mathcal{W}}$, it holds that

- the number of distinct triples $(h,f,\mathbf{p})$ (from which the quadruples consist) with respect to which $\mathcal{H}$ satisfies Properties 1 and 2 above is upper bounded by $(1+\xi)\cdot T$, and

- for each triple $(h, f, \mathbf{p})$ with respect to which $\mathcal{H}$ satisfies Properties 1 and 2 above, the number of sets $F \in \mathscr{F}^{h,f,\mathbf{p}}$ with respect to which $\mathcal{H}$ satisfies Property 3 above is upper bounded by $(1+\xi) \cdot T_{\mathbf{p}}$.

Thus, we have that

$$\sum_{Q=(h,f,\mathbf{p},F)\in(\mathscr{H},\mathscr{S},\{\mathscr{F}^{h,f,\mathbf{p}}\}|_{h\in\mathscr{H},f\in\mathscr{S},\mathbf{p}})} \frac{[\mathcal{H} \in \mathcal{P}_{a,v,Q}^{U',\mathcal{S}',p',\mathcal{R},\mathcal{W}}]}{T \cdot T_{\mathbf{p}}} \leq (1+\xi)^2,$$

where $[\mathcal{H} \in \mathcal{P}_{a,v,Q}^{U',\mathcal{S}',p',\mathcal{R},\mathcal{W}}]$ is 1 if $\mathcal{H} \in \mathcal{P}_{a,v,Q}^{U',\mathcal{S}',p',\mathcal{R},\mathcal{W}}$, and 0 otherwise. Since the choice of $\mathcal{H}$ was arbitrary, we get that

$$\begin{aligned}\alpha(v) &\leq (1+\xi)^{2(k'-2)} \sum_{\substack{a\in U_1 \\ \text{s.t. } a<v}} \beta(a)(1+\xi)^2 |\mathcal{P}_{a,v}^{U',\mathcal{S}',p',\mathcal{R},\mathcal{W}}| \\ &= (1+\xi)^{2(k'-1)} \sum_{\substack{a\in U_1 \\ \text{s.t. } a<v}} \beta(a) |\mathcal{P}_{a,v}^{U',\mathcal{S}',p',\mathcal{R},\mathcal{W}}|.\end{aligned}$$

Symmetrically, we derive that

$$\alpha(v) \geq (1-\xi)^{2(k'-1)} \sum_{\substack{a\in U_1 \\ \text{s.t. } a<v}} \beta(a) |\mathcal{P}_{a,v}^{U',\mathcal{S}',p',\mathcal{R},\mathcal{W}}|.$$

This completes the proof. □

Finally, we conclude the proof of correctness of our algorithm.

Theorem 7.4. *There is a deterministic $4^{k+\mathcal{O}(\sqrt{k}(\log^2 k+\log^2 \frac{1}{\epsilon}))}(n+m)\log n$-time polynomial-space algorithm that, given positive integers p and q where $k=(q-1)p$, a partition $(U_1, U_2, \ldots, U_q)$ of a universe U, a family $\mathcal{S}$ of sets over U where $|\{S \cap U_i\}| = 1$ for every $S \in \mathcal{S}$ and $i \in \{1, 2, \ldots, q\}$, and an accuracy value $0<\epsilon<1$, outputs a number y that satisfies $(1-\epsilon)x \leq y \leq (1+\epsilon)x$ where x is the number of p-packings in $\mathcal{S}$. In particular, if $\frac{1}{\epsilon} = 2^{o(k^{\frac{1}{4}})}$, then the running time is $4^{k+o(k)}(n+m)\log n$.*

Proof. Observe that for all $v \in U_1$, it holds that $\sum_{\substack{u\in U_1 \\ \text{s.t. } u<v}} \beta_{\text{init}}(u) \cdot |\mathcal{P}_{u,v}^{U,\mathcal{S},p,(),()}|$ is equal to the number of p-packings in $\mathcal{S}$ such that the largest element they contain from U_1 is v; we denote this number by x_v. Thus, by Lemma 6.1 with $U'=U$, $\mathcal{S}'=\mathcal{S}$, $p'=p$ and $\beta=\beta_{\text{init}}$, we know that

- $\mathcal{A}(U, \mathcal{S}, p, \beta_{\text{init}}, (), ())$ runs in time $4^{k+\mathcal{O}(\sqrt{k}(\log^2 k+\log^2 \frac{1}{\xi}))}(n+m)\log n$. Moreover, $\mathcal{A}(U, \mathcal{S}, p, \beta_{\text{init}}, (), ())$ has a polynomial space complexity.
- For all $v \in U_1$, the number $\alpha(v)$ assigned to v by the assignment α returned by $\mathcal{A}(U, \mathcal{S}, p, \beta_{\text{init}}, (), ())$ satisfies the following inequalities:

$$(1-\xi)^{2(k-1)} x_v \leq \alpha(v) \leq (1+\xi)^{2(k-1)} x_v.$$

First, by substituting ξ and since $\frac{\epsilon}{2} \leq \ln(1+\epsilon) \leq \epsilon$, we have that $\mathcal{A}(U, \mathcal{S}, p, \beta_{\text{init}}, (), ())$ runs in time

$$4^{k+\mathcal{O}(\sqrt{k}(\log^2 k+\log^2 \frac{2(k-1)}{\ln(1+\epsilon)}))}(n+m)\log n \leq 4^{k+\mathcal{O}(\sqrt{k}(\log^2 k+\log^2 \frac{1}{\epsilon}))}(n+m)\log n.$$

Thus, we obtain the claimed running time.

For the accuracy, substituting ξ by ϵ, we have that for all $v \in U_1$, it holds that

$$(1-\epsilon)x_v \leq (1-\frac{\ln(1+\epsilon)}{2(k-1)})^{2(k-1)}x_v \leq \alpha(v) \leq (1+\frac{\ln(1+\epsilon)}{2(k-1)})^{2(k-1)}x_v \leq (1+\epsilon)x_v.$$

Having these inequalities at hand, as in the proof of Theorem 4.2, we obtain that

$$y = \sum_{v \in U_1} \alpha(v) \leq \sum_{v \in U_1} (1+\epsilon)x_v = (1+\epsilon)\sum_{v \in U_1} x_v = (1+\epsilon)x.$$

Symmetrically, we obtain that $(1-\epsilon)x \leq y$. This completes the proof. □

References

[1] *Randomization in parameterized complexity.* `www.dagstuhl.de/de/programm/kalender/semhp/?semnr=17041`.

[2] N. Alon, P. Dao, I. Hajirasouliha, F. Hormozdiari, and S. C. Sahinalp, *Biomolecular network motif counting and discovery by color coding*, in Proceedings 16th International Conference on Intelligent Systems for Molecular Biology (ISMB), Toronto, Canada, July 19-23, 2008, 2008, pp. 241–249.

[3] N. Alon and S. Gutner, *Balanced hashing, color coding and approximate counting*, in Parameterized and Exact Computation, 4th International Workshop, IWPEC 2009, Copenhagen, Denmark, September 10-11, 2009, Revised Selected Papers, 2009, pp. 1–16.

[4] ——, *Balanced families of perfect hash functions and their applications*, ACM Trans. Algorithms, 6 (2010), pp. 54:1–54:12.

[5] N. Alon and J. H. Spencer, *The Probabilistic Method*, Wiley Publishing, 4th ed., 2016.

[6] N. Alon, R. Yuster, and U. Zwick, *Color-coding*, J. ACM, 42 (1995), pp. 844–856.

[7] V. Arvind and V. Raman, *Approximation algorithms for some parameterized counting problems*, in Algorithms and Computation, 13th International Symposium, ISAAC 2002 Vancouver, BC, Canada, November 21-23, 2002, Proceedings, 2002, pp. 453–464.

[8] A. Berger, L. Kozma, M. Mnich, and R. Vincze, *A time- and space-optimal algorithm for the many-visits TSP*, in Proceedings of the Thirtieth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2019, San Diego, California, USA, January 6-9, 2019, 2019, pp. 1770–1782.

[9] A. Björklund, *Determinant sums for undirected hamiltonicity*, SIAM J. Comput., 43 (2014), pp. 280–299.

[10] A. Björklund, T. Husfeldt, P. Kaski, and M. Koivisto, *Counting paths and packings in halves*, in Algorithms - ESA 2009, 17th Annual European Symposium, Copenhagen, Denmark, September 7-9, 2009. Proceedings, 2009, pp. 578–586.

[11] ——, *Narrow sieves for parameterized paths and packings*, J. Comput. Syst. Sci., 87 (2017), pp. 119–139.

[12] A. Björklund, V. Kamat, L. Kowalik, and M. Zehavi, *Spotting trees with few leaves*, SIAM J. Discrete Math., 31 (2017), pp. 687–713.

[13] A. Björklund, P. Kaski, and L. Kowalik, *Counting thin subgraphs via packings faster than meet-in-the-middle time*, ACM Trans. Algorithms, 13 (2017), pp. 48:1–48:26.

[14] C. Brand, H. Dell, and T. Husfeldt, *Extensor-coding*, in Proceedings of the 50th Annual ACM SIGACT Symposium on Theory of Computing, STOC 2018, Los Angeles, CA, USA, June 25-29, 2018, 2018, pp. 151–164.

[15] J. Chen, J. Kneis, S. Lu, D. Molle, S. Richter, P. Rossmanith, S.-H. Sze, and F. Zhang, *Randomized divide-and-conquer: Improved path, matching, and packing algorithms*, SIAM Journal on Computing, 38 (2009), pp. 2526—2547.

[16] J. Chen, J. Kneis, S. Lu, D. Mölle, S. Richter, P. Rossmanith, S. Sze, and F. Zhang, *Randomized divide-and-conquer: Improved path, matching, and packing algorithms*, SIAM Journal on Computing, 38 (2009), pp. 2526–2547.

[17] R. Curticapean, H. Dell, and D. Marx, *Homomorphisms are a good basis for counting small subgraphs*, in Proceedings of the 49th Annual ACM SIGACT Symposium on Theory of Computing, STOC 2017, New York, NY, USA, 2017, ACM, pp. 210–223.

[18] R. Curticapean and D. Marx, *Complexity of counting subgraphs: Only the boundedness of the vertex-cover number counts*, in 55th IEEE Annual Symposium on Foundations of Computer Science, FOCS 2014, Philadelphia, PA, USA, October 18-21, 2014, 2014, pp. 130–139.

[19] M. Cygan, F. V. Fomin, L. Kowalik, D. Lokshtanov, D. Marx, M. Pilipczuk, M. Pilipczuk, and S. Saurabh, *Parameterized Algorithms*, Springer, 2015.

[20] B. Dost, T. Shlomi, N. Gupta, E. Ruppin, V. Bafna, and R. Sharan, *Qnet: A tool for querying protein interaction networks*, Journal of Computational Biology, 15 (2008), pp. 913–925.

[21] J. Flum and M. Grohe, *The parameterized complexity of counting problems*, SIAM J. Comput., 33 (2004), pp. 892–922.

[22] F. V. Fomin, P. Kaski, D. Lokshtanov, F. Panolan, and S. Saurabh, *Parameterized single-exponential time polynomial space algorithm for steiner tree*, in Automata, Languages, and Programming - 42nd International Colloquium, ICALP 2015, Kyoto, Japan, July 6-10, 2015, Proceedings, Part I, 2015, pp. 494–505.

[23] F. V. Fomin, D. Lokshtanov, F. Panolan, and S. Saurabh, *Efficient computation of representative families with applications in parameterized and exact algorithms*, J. ACM, 63 (2016), pp. 29:1–29:60.

[24] F. V. Fomin, D. Lokshtanov, S. Saurabh, and M. Zehavi, *Kernelization: Theory of Parameterized Preprocessing*, Cambridge University Press, 2019.

[25] G. Z. Gutin, F. Reidl, M. Wahlström, and M. Zehavi, *Designing deterministic polynomial-space algorithms by color-coding multivariate polynomials*, J. Comput. Syst. Sci., 95 (2018), pp. 69–85.

[26] F. Hüffner, S. Wernicke, and T. Zichner, *Algorithm engineering for color-coding with applications to signaling pathway detection*, Algorithmica, 52 (2008), pp. 114–132.

[27] S. Jukna, *Extremal Combinatorics: With Applications in Computer Science*, Springer Publishing Company, Incorporated, 1st ed., 2010.

[28] I. Koutis, *Faster algebraic algorithms for path and packing problems*, in Automata, Languages and Programming, 35th International Colloquium, ICALP 2008, Reykjavik, Iceland, July 7-11, 2008, Proceedings, Part I: Tack A: Algorithms, Automata, Complexity, and Games, 2008, pp. 575–586.

[29] I. Koutis and R. Williams, *Algebraic fingerprints for faster algorithms*, Commun. ACM, 59 (2016), pp. 98–105.

[30] ——, *LIMITS and applications of group algebras for parameterized problems*, ACM Trans. Algorithms, 12 (2016), pp. 31:1–31:18.

[31] D. Lokshtanov, M. Mnich, and S. Saurabh, *Planar k-path in subexponential time and polynomial space*, in Graph-Theoretic Concepts in Computer Science - 37th International Workshop, WG 2011, Teplá Monastery, Czech Republic, June 21-24, 2011. Revised Papers, 2011, pp. 262–270.

[32] D. Lokshtanov and J. Nederlof, *Saving space by algebraization*, in Proceedings of the 42nd ACM Symposium on Theory of Computing, STOC 2010, Cambridge, Massachusetts, USA, 5-8 June 2010, 2010, pp. 321–330.

[33] R. Milo, S. Shen-Orr, S. Itzkovitz, N. Kashtan, D. Chklovskii, and U. Alon, *Network motifs: Simple building blocks of complex networks*, Science, 298 (2002), pp. 824–827.

[34] M. Mitzenmacher and E. Upfal, *Probability and computing - randomized algorithms and probabilistic analysis*, Cambridge University Press, 2005.

[35] M. Naor, L. J. Schulman, and A. Srinivasan, *Splitters and near-optimal derandomization*, in 36th Annual Symposium on Foundations of Computer Science, Milwaukee, Wisconsin, 23-25 October 1995, 1995, pp. 182–191.

[36] J. Scott, T. Ideker, R. M. Karp, and R. Sharan, *Efficient algorithms for detecting signaling pathways in protein interaction networks*, Journal of Computational Biology, 13 (2006), pp. 133–144.

[37] H. Shachnai and M. Zehavi, *Representative families: A unified tradeoff-based approach*, J. Comput. Syst. Sci., 82 (2016), pp. 488–502.

[38] R. Sharan and T. Ideker, *Modeling cellular machinery through biological network comparison. nat. biotechnol. 24, 427-433*, Nature biotechnology, 24 (2006), pp. 427–33.

[39] T. Shlomi, D. Segal, E. Ruppin, and R. Sharan, *Qpath: a method for querying pathways in a protein-protein interaction network*, BMC Bioinformatics, 7 (2006), p. 199.

[40] D. Tsur, *Faster deterministic parameterized algorithm for* k*-path*, Theor. Comput. Sci., 790 (2019), pp. 96–104.

[41] R. Williams, *Finding paths of length k in $o^*(2^k)$ time*, Inf. Process. Lett., 109 (2009), pp. 315–318.

[42] V. V. Williams and R. Williams, *Finding, minimizing, and counting weighted subgraphs*, SIAM J. Comput., 42 (2013), pp. 831–854.

[43] M. Zehavi, *Mixing color coding-related techniques*, in Algorithms - ESA 2015 - 23rd Annual European Symposium, Patras, Greece, September 14-16, 2015, Proceedings, 2015, pp. 1037–1049.

A Extension to Trees (and Bounded Treewidth Graphs)

We will make use of the following notation. Given a tree T and a vertex $p \in V(T)$, we let T_p denote the subtree of T rooted at p, and let $\overline{T_p}$ be the tree obtained when T_p is removed from T. Our algorithm for $\#k$-Tree will rely on two parts. The first is an extension of the algorithm from Section 4.2 to solve $\#k$-Tree in time $6.75^{k+o(k)}n^{\mathcal{O}(1)}(\frac{1}{\epsilon})^{\mathcal{O}(\log k)}$. To this end, we will make use of the following well-known (folklore) proposition about separators in trees.

Proposition A.1 (Folklore)**.** *Let T be a t-vertex tree. Then, there exists a vertex $p \in V(T)$ such that both $|V(T_p)| \leq \frac{2}{3}t$ and $|V(\overline{T_p})| \leq \frac{2}{3}t$. Further, such a vertex p can be computed in polynomial time.*

So, we will consider the following lemma in Appendix A.2. Here, we solve a slightly more general problem than $\#k$-Tree, where the mapping of some vertices is fixed, as this generalization will come in handy for our main algorithm. Clearly, the success probability can be boosted by repetitions.

Lemma A.1. *There is a randomized $6.75^{k+o(k)}n^{\mathcal{O}(1)}(\frac{1}{\epsilon})^{\mathcal{O}(\log k)}$-time polynomial-space algorithm that, given a graph G on n vertices, a tree T on k vertices, an injection $f : U \to V(G)$ for some $U \subseteq V(T)$, and an accuracy value $0 < \epsilon < 1$, outputs a number y that (with high probability, say, at least $9/10$) satisfies $(1-\epsilon)x \leq y \leq (1+\epsilon)x$ where x is the number of isomorphisms between T and all subtrees of G such that every vertex $p \in U$ is mapped to $f(p)$.*

We will only call this algorithm with a parameter k' that is extremely small (yet linear in k), and so the fact that the constant is 6.75 rather than 4 will essentially have no consequence (as we eventually strive to derive the base 4.001 rather than exactly 4). This will be done by our main algorithm, where we apply the "division into small trees" trick, and which is an extension of the algorithm from Section 4.1. This trick is based on the following result.

Proposition A.2 ([23])**.** *For any $c \geq 1$ and t-vertex tree T, there exists a subset $W \subseteq V(T)$ of size $\mathcal{O}(c)$ such that each tree R in $T-W$ contains at most $\mathcal{O}(t/c)$ vertices and $|N_T(V(R)) \cap W| \leq 2$. Moreover, such a set W can be computed in polynomial time.*

For us, c will be some fixed constant. So, we will consider the following theorem in Appendix A.1.

Theorem A.1. *There is a randomized $4.001^k n^{\mathcal{O}(1)}(\frac{1}{\epsilon})^{\mathcal{O}(\log k)}$-time polynomial-space algorithm that, given a graph G on n vertices, a tree T on k vertices and an accuracy value $0 < \epsilon < 1$, outputs a number y that (with high probability, say, at least $9/10$) satisfies $(1-\epsilon)x \leq y \leq (1+\epsilon)x$ where x is the number of isomorphisms between T and all subtrees of G.*

We will skip formal proofs of correctness (which are essentially the same as the proofs in Sections 4.1 and 4.2), but we will fully describe the algorithms in the next two subsections. We remark that if we would like to count the number of subtrees of G that are isomorphic to T, then the number output by our algorithms should be divided by the number of automorphisms of T.

A.1 Proof of Lemma A.1

We root T at some arbitrarily chosen vertex, and in what follows, we treat T accordingly as a rooted tree. For a vertex $p \in V(T)$, we let $\mathsf{parent}(p)$ denote the parent of p in T. For a subtree T' of T, we let $\mathsf{root}(T')$ denote the root of T', which is the (unique) vertex of T' that is closest (in T) to the root of T. Additionally, we let $\mathsf{children}(T')$ denote the set of vertices in $N_T(V(T'))$ that are descendants of $\mathsf{root}(T')$ in T (these are all vertices in $N_T(V(T'))$ expect for at most one, which is the parent of $\mathsf{root}(T')$ in T, unless $\mathsf{root}(T') = \mathsf{root}(T)$).

Algorithm. Let $\xi = \ln(1+\epsilon)/(k-1)$. Our algorithm, denoted by $\mathcal{A}$, is recursive. Each call to $\mathcal{A}$ is of the form $\mathcal{A}(G', T', B)$ where G' is an induced subgraph of G, T' is a subtree of T, and $B = \{\beta_p : p \in \mathsf{children}(T')\}$ where for every $p \in \mathsf{children}(T')$, $\beta_p : V(G) \setminus V(G') \to \mathbb{N}_0$. (The supports of different functions β_p can be assumed to be disjoint.) To see the correspondence to the algorithm in Section 4.2, let $k' = |V(T')|$.

The call $\mathcal{A}(G', T', B)$ should output an assignment $\alpha : V(G') \to \mathbb{N}_0$ with the following property: For each vertex $v \in V(G')$, it holds that $\alpha(v)$ approximates the following number:

$$\sum_{g:\mathsf{children}(T') \to V(G) \setminus V(G')} \left(\left(\prod_{p \in \mathsf{children}(T')} \beta_p(g(p)) \right) \cdot x_{g,v}^{G',T'} \right),$$

where $x_{g,v}^{G',T'}$ is the number of isomorphisms h between T' and all subtrees of G' such that:

1. For all $p \in V(T') \cap \mathsf{domain}(f)$, $h(p) = f(p)$.
2. For all $p \in \mathsf{children}(T')$, $h(\mathsf{parent}(p))$ is a neighbor (in G) of $g(p)$.

We note that if $\mathsf{children}(T') = \emptyset$, then the above expression only corresponds to g with empty domain, and then we just mean that $\alpha(v)$ approximates $x_{g,v}^{G',T'}$ (where g is the function with empty domain), which is the number of isomorphisms h between T' and all subtrees of G' such that for all $p \in V(T') \cap \mathsf{domain}(f)$, $h(p) = f(p)$.

The initial call to the algorithm is with $G' = G$, $T' = T$, and $B = \emptyset$. The final output is $\sum_{v \in V(G)} \alpha(v)$.

We turn to describe a call $\mathcal{A}(G', T', B)$. In the basis, where $k' = 1$, we return an assignment $\alpha : V(G') \to \mathbb{N}_0$ defined as follows: If $\mathsf{root}(T') \notin \mathsf{domain}(f)$, then for each vertex $v \in V(G')$, define

$$\alpha(v) = \sum_{g:\mathsf{children}(T') \to N_G(v) \setminus V(G')} \left(\prod_{p \in \mathsf{children}(T')} \beta_p(g(p)) \right).$$

This expression is computed in polynomial time using dynamic programming. To this end, we denote $\mathsf{children}(T') = \{q_1, q_2, \ldots, q_t\}$ where $t = |\mathsf{children}(T')|$, and observe that the aforementioned expression is equal to:

$$\sum_{v_t \in N_G(v) \setminus V(G')} \left(\beta_{q_t}(v_t) \cdot \sum_{v_{t-1} \in N_G(v) \setminus V(G')} \left(\beta_{q_{t-1}}(v_{t-1}) \cdots \left(\sum_{q_1 \in N_G(v) \setminus V(G')} \beta_{q_1}(v_1) \right) \cdots \right) \right)$$

So, it can be compute using a table with an entry for every q_i, $i \in \{1, 2, \ldots, t\}$.

If $\mathsf{root}(T') \in \mathsf{domain}(f)$, then we use the aforementioned expression only when $v = f(\mathsf{root}(T'))$, and for every other $v \in V(G')$, we set $\alpha(v) = 0$.

Now, suppose that $k' \geq 2$. We use the algorithm in Proposition A.1 to obtain a vertex p. By Theorem 3.1, for a ξ-parsimonious $(n, |V(T'_p)|, |V(\overline{T'_p})|)$-universal family $\mathscr{F}$ of sets over $V(G)$, we can enumerate the sets $F \in \mathscr{F}$ with delay $\mathcal{O}(n)$. For each set $F \in \mathscr{F}$, we proceed as follows.

1. We first recursively call $\mathcal{A}$ with $(G'[F], T'_p, B^1_F)$ where B^1_F is obtained from B by taking only the functions β_q where $q \in \mathsf{children}(T'_p)$, and extending each such function to assign 0 to every vertex in $V(G') \setminus F$. Let $\beta_{p,F}$ be the output of this call, and extend it to assign 0 to every vertex in $V(G) \setminus V(G')$.
2. Then, we recursively call $\mathcal{A}$ with $(G' - F, \overline{T'_p}, B^2_F)$ where B^2_F is obtained from B by taking only the functions β_q where $q \in \mathsf{children}(\overline{T'_p}) \setminus \{p\}$, and extending each such function to assign 0 to every vertex in F, as well as inserting $\beta_{p,F}$. Let α_F be the output of this recursive call.

Let T be the correction factor of $\mathscr{F}$. After all sets $F \in \mathscr{F}$ were enumerated, the output $\alpha : V(G') \to \mathbb{N}_0$ is computed as follows. For all $v \in V(G')$, we calculate

$$\alpha(v) = \left(\sum_{\substack{F \in \mathscr{F} \\ \text{s.t. } v \notin F}} \alpha_F(v) \right) / T.$$

Note that we do not store all the assignments α_F simultaneously, but we merely store one such assignment at a time and delete it immediately after $\alpha_F(v)/T$, for every $v \in V(G')$, is added. This completes the description of $\mathcal{A}$.

Analysis. It should be clear that the space complexity is polynomial. Now, let us briefly explain why we get the base 6.75. Let $\mathsf{T}(k')$ denote the time complexity of the algorithm when called with T' such that $|V(T')| = k'$. Observe that by Proposition A.1 and Theorem 3.1, we can bound $\mathsf{T}(k')$ as follows:

$$\begin{aligned}
\mathsf{T}(k') &= \max_{\frac{1}{3}k' \le t \le \frac{2}{3}k'} \binom{k'}{t} \cdot (\mathsf{T}(t) + \mathsf{T}(k'-t)) \cdot n^{\mathcal{O}(1)} \\
&= \binom{k'}{\frac{2}{3}k'} \cdot \mathsf{T}(\frac{2}{3}k') \cdot n^{\mathcal{O}(1)} \\
&= (\frac{3}{2^{\frac{2}{3}}})^{k'} \cdot \mathsf{T}(\frac{2}{3}k') \cdot n^{\mathcal{O}(1)} \\
&= (\frac{3}{2^{\frac{2}{3}}})^{\sum_{i=0}^{\infty}(\frac{2}{3})^i k'} \cdot n^{\mathcal{O}(1)} \\
&= (\frac{3}{2^{\frac{2}{3}}})^{3k'} \cdot n^{\mathcal{O}(1)} = (\frac{3^3}{2^2})^{k'} \cdot n^{\mathcal{O}(1)} = 6.75^{k'} \cdot n^{\mathcal{O}(1)}.
\end{aligned}$$

A.2 Proof of Theorem A.1

Algorithm. Let $\xi = \ln(1+\epsilon)/(k-1)$. In preprocessing, we use the algorithm in Proposition A.2 to obtain W where the constant $c \ge 1$ will be determined later. Let $\mathcal{C}$ be the set of connected components (which are trees) of $T - W$. For every injective function $f : W \to V(G)$, we make a call to our recursive algorithm, denoted by $\mathcal{A}$. Each call to $\mathcal{A}$ is of the form $\mathcal{A}(G', \mathcal{C}', f)$ where G' is an induced subgraph of G (that contains the image of f) and $\mathcal{C}' \subseteq \mathcal{C}$. The call $\mathcal{A}(G', \mathcal{C}')$ should output an integer a that approximates the number of isomorphisms between the forest $T_{W,\mathcal{C}'} = T[W \cup \bigcup_{C \in \mathcal{C}'} V(C)]$ and all subforests of G' such that for all $p \in W$, p is mapped to $f(p)$. The algorithm $\mathcal{A}$ is initially called with $G' = G$ and $\mathcal{C}' = \mathcal{C}$. After all initial recursive calls to $\mathcal{A}$ (corresponding to each choice of f) have been made, the algorithm returns the sum of the integers a that they return.

Let d be a constant that upper bounds those hidden in the $\mathcal{O}$ notations in Proposition A.2 (which is independent of c). We now turn to describe a call $\mathcal{A}(G', \mathcal{C}', f)$. In the basis, where $|\bigcup_{C \in \mathcal{C}'} V(C)| \le 3dk/c$, we call the algorithm from Lemma A.1 on graph H, tree R and function $f_{H,R}$, defined as follows. Let $\mathcal{D}$ be the set of connected components (trees) of the forest $T_{W,\mathcal{C}'}$. Let H be the graph obtained from G' be adding a new vertex v and making it adjacent to all vertices in G'. Let R be the tree obtained from $T_{W,\mathcal{C}'}$ by adding a new vertex p and making it adjacent to exactly one vertex from each tree in $\mathcal{D}$ (which is chosen arbitrarily). Lastly, let $f_{H,R}$ be the extension of f that assigns v to p.

Now, suppose that $|\bigcup_{C \in \mathcal{C}'} V(C)| > 3dk/c$. Then, $|\mathcal{C}'| \ge 3$. Partition $\mathcal{C}'$ into two sets, $\mathcal{C}_1$ and $\mathcal{C}_2$, such that $|\bigcup_{C \in \mathcal{C}_1} V(C)| - |\bigcup_{C \in \mathcal{C}_2} V(C)|$ is minimized (this can be done in polynomial time using dynamic programming). Notice that $|\bigcup_{C \in \mathcal{C}_1} V(C)| - |\bigcup_{C \in \mathcal{C}_2} V(C)| \le dk/c$,

$|\bigcup_{C\in\mathcal{C}_1} V(C)| \geq dk/c$ and $|\bigcup_{C\in\mathcal{C}_2} V(C)| \geq dk/c$. By Theorem 3.1, for a ξ-parsimonious $(n, |\bigcup_{C\in\mathcal{C}_1} V(C)|, |\bigcup_{C\in\mathcal{C}_2} V(C)|)$-universal family $\mathscr{F}$ of sets over $V(G)$, we can enumerate the sets $F \in \mathscr{F}$ with delay $\mathcal{O}(n)$. For each set $F \in \mathscr{F}$, we proceed as follows. We perform two recursive calls:

1. We call $\mathcal{A}$ with $(G'[F \cup \mathsf{image}(f)], \mathcal{C}_1, f)$. Let x_F denote its output.
2. We call $\mathcal{A}$ with $(G' - (F \setminus \mathsf{image}(f)), \mathcal{C}_2, f)$. Let y_F denote its output.

Let T be the correction factor of $\mathscr{F}$. After all sets $F \in \mathscr{F}$ were enumerated, we output a calculated as follows: $a = \frac{1}{T} \cdot \sum_{F\in\mathscr{F}} x_F \cdot y_F$. This completes the description of $\mathcal{A}$.

Analysis. It should be clear that the space complexity is polynomial. Now, let us briefly explain why we get the base 4.001. Let $\mathsf{T}(k')$ denote the time complexity of the algorithm when called with $\mathcal{C}'$ such that $|\bigcup_{C\in\mathcal{C}'} V(C)| = k'$. Observe that, by Lemma A.1, when $k' \leq 3dk/c$, we can bound $\mathsf{T}(k') = 6.75^{3dk/c} \cdot n^{\mathcal{O}(1)}$. Moreover, by Theorem 3.1 when $k' > 3.dk/c$, we can bound $\mathsf{T}(k')$ as follows:

$$\begin{aligned}\mathsf{T}(k') &= \max_{(\frac{1}{2}-\frac{d}{c})k' \leq t \leq (\frac{1}{2}+\frac{d}{c})k'} \binom{k'}{t} \cdot (\mathsf{T}(t) + \mathsf{T}(k'-t)) \cdot n^{\mathcal{O}(1)} \\ &= \binom{k'}{(\frac{1}{2}+\frac{d}{c})k'} \cdot \mathsf{T}((\frac{1}{2}+\frac{d}{c})k') \cdot n^{\mathcal{O}(1)}.\end{aligned}$$

Notice that the larger c is, the smaller $6.75^{3dk/c}$ is, and the closer is $\binom{k'}{(\frac{1}{2}+\frac{d}{c})k'} \cdot \mathsf{T}((\frac{1}{2}+\frac{d}{c})k') \cdot n^{\mathcal{O}(1)}$ to $\binom{k'}{\frac{1}{2}k'} \cdot \mathsf{T}(\frac{1}{2}k') \cdot n^{\mathcal{O}(1)}$. So, by picking a larger c, then time complexity can be bounded by $q^k \cdot n^{\mathcal{O}(1)}$ where q can be made arbitrarily close to 4 (though the polynomial factor $n^{\mathcal{O}(1)}$ becomes larger because $\mathcal{O}(1)$ hides c).

Patterns of Bounded Treewidth. We briefly remark that in order to extend our algorithms to patterns of bounded treewidth, firstly, the set of vertices W obtained by applying Proposition A.2 (as well as the vertices obtained when applying Proposition A.1) are nodes of the tree T of the tree decomposition (T, β) of the input pattern graph. Further, when we map them to G (when we choose f), we need to map all the vertices that belong to the sets assigned to them by β so that existing adjacencies between them are preserved. Several similar modifications should be made. In particular, when we consider the algorithm in Appendix A.2, instead of considering vertices v, we need to consider sets of vertices (of size at most the treewidth) that preserve existing adjacencies, and in the basis, existing adjacencies between all vertices mapped to the root of T' and all vertices mapped to its children should be verified.