Approximately counting and sampling small witnesses using a colourful decision oracle[*]

Holger Dell[†] John Lapinskas[‡] Kitty Meeks[§]

Abstract

In this paper, we prove "black box" results for turning algorithms which decide whether or not a witness exists into algorithms to approximately count the number of witnesses, or to sample from the set of witnesses approximately uniformly, with essentially the same running time. We do so by extending the framework of Dell and Lapinskas (STOC 2018), which covers decision problems that can be expressed as edge detection in bipartite graphs given limited oracle access; our framework covers problems which can be expressed as edge detection in arbitrary k-hypergraphs given limited oracle access. (Simulating this oracle generally corresponds to invoking a decision algorithm.) This includes many key problems in both the fine-grained setting (such as k-SUM, k-OV and weighted k-Clique) and the parameterised setting (such as induced subgraphs of size k or weight-k solutions to CSPs). From an algorithmic standpoint, our results will make the development of new approximate counting algorithms substantially easier; indeed, it already yields a new state-of-the-art algorithm for approximately counting graph motifs, improving on Jerrum and Meeks (JCSS 2015) unless the input graph is very dense and the desired motif very small. Our k-hypergraph reduction framework generalises and strengthens results in the graph oracle literature due to Beame et al. (ITCS 2018) and Bhattacharya et al. (CoRR abs/1808.00691).

1 Introduction

Many decision problems reduce to the question: Does a witness exist? Such problems admit a natural counting version: How many witnesses exist? For example, one may ask whether a bipartite graph contains a perfect matching, or how many perfect matchings it contains. As one might expect, the counting version is never easier than the decision version, and is often substantially harder; for example, deciding whether a bipartite graph contains a perfect matching is easy, and counting the number of such matchings is #P-complete [41]. However, even when the counting version of a problem is hard, it is often easy to approximate well. For example, Jerrum, Sinclair and Vigoda [31] gave a polynomial-time approximation algorithm for the number of perfect matchings in a bipartite graph. The study of approximate counting has seen amazing progress over the last two decades, particularly in the realm of trichotomy results for general problem frameworks such as constraint satisfaction problems, and is now a major field of study in its own right [17, 18, 24, 27, 28]. In this paper, we explore the question of when approximating the counting version of a problem is not merely fast, but essentially as fast as solving the decision version.

We first recall the standard notion of approximation in the field: For all real $x, y > 0$ and $0 < \varepsilon < 1$, we say that x is an *ε-approximation* to y if $|x - y| < \varepsilon y$. Note in particular that any ε-approximation to zero is itself zero, so computing an ε-approximation to N is always at least as hard as deciding whether $N > 0$ holds. For example, it is at least as hard to approximately count the number of satisfying assignments of a CNF formula (i.e. to ε-approximate #SAT) as it is to decide whether it is satisfiable at all (i.e. to solve SAT).

Perhaps surprisingly, in many cases, the converse is also true. For example, Valiant and Vazirani [42] proved that any polynomial-time algorithm to decide SAT can be bootstrapped into a polynomial-time ε-approximation algorithm for #SAT, or, more formally, that a size-n instance of any problem in #P can be ε-approximated in time poly(n, ε^{-1}) using an NP-oracle. A similar result holds in the parameterised setting, where Müller [39] proved that a size-n instance of any problem in #W$[i]$ with parameter k can be ε-approximated in time $g(k) \cdot$ poly(n, ε^{-1}) using a W$[i]$-oracle for some computable function $g\colon \mathbb{N} \to \mathbb{N}$. Another such result holds in the subexponential setting, where Dell and Lapinskas [14] proved that the (randomised) Exponential Time Hypothesis is equivalent to the statement: There is no ε-approximation algorithm for #3-SAT which runs on an n-variable instance in time $\varepsilon^{-2}2^{o(n)}$.

We now consider the fine-grained setting, which is the focus of this paper. Here, we are concerned with the exact running time of an algorithm, rather than broad categories such as polynomial time, FPT time or subexponential time.

[*] The research leading to these results has received funding from the European Research Council under the European Union's Seventh Framework Programme (FP7/2007-2013) ERC grant agreement no. 334828. The paper reflects only the authors' views and not the views of the ERC or the European Commission. The European Union is not liable for any use that may be made of the information contained therein. The research was also supported by a Royal Society of Edinburgh Personal Research Fellowship, funded by the Scottish Government. The collaboration begun at Dagstuhl Seminar 17341.

[†] IT University of Copenhagen and BARC, Copenhagen, Denmark.

[‡] University of Bristol, Bristol, UK.

[§] University of Glasgow, Glasgow, UK.

The above reductions all introduce significant overhead, so they are not fine-grained. Here only one general result is known, again due to Dell and Lapinskas [14]. Informally, if the decision problem reduces "naturally" to deciding whether an n-vertex bipartite graph contains an edge, then any algorithm for the decision version can be bootstrapped into an ε-approximation algorithm for the counting version with only $\mathcal{O}(\varepsilon^{-2}\text{polylog}(n))$ overhead. (See Section 1.1 for more details.)

The reduction of [14] is general enough to cover core problems in fine-grained complexity such as ORTHOGONAL VECTORS, 3SUM and NEGATIVE-WEIGHT TRIANGLE, but it is not universal. In this paper, we substantially generalise it to cover any problem which can be "naturally" formulated as deciding whether a k-partite k-hypergraph contains an edge; thus we essentially recover the original result on taking $k = 2$. For any problem which satisfies this property, our result implies that any new decision algorithm will automatically lead to a new approximate counting algorithm whose running time is at most a factor of $\log^{\mathcal{O}(k)} n$ larger. Our framework covers several reduction targets in fine-grained complexity not covered by [14], including k-ORTHOGONAL VECTORS, k-SUM and EXACT-WEIGHT k-CLIQUE, as well as some key problems in parameterised complexity including weight-k CSPs and size-k induced subgraph problems. (Note that the overhead of $\log^{\mathcal{O}(k)} n$ can be re-expressed as $k^{2k}n^{o(1)}$ using a standard trick, so an FPT decision algorithm is transformed into an FPT approximate counting algorithm; see Section 1.3.)

In fact, we get more than fast approximate counting algorithms — we also prove that any problem in this framework has an algorithm for approximately-uniform sampling, again with $\log^{\mathcal{O}(k)} n$ overhead over decision. There is a well-known reduction between the two for self-reducible problems due to Jerrum, Valiant and Vazirani [32], but it does not apply in our setting since it adds polynomial overhead.

In the parameterised setting, our results have interesting implications. Here, the requirement that the hypergraph be k-partite typically corresponds to considering the "colourful" or "multicolour" version of the decision problem, so our result implies that uncoloured approximate counting is essentially equivalent to multicolour decision. We believe that our results motivate considerable further study of the relationship between multicolour parameterised decision problems and their uncoloured counterparts.

Finally, we note that the applications of our results are not just complexity-theoretic in nature, but also algorithmic. They give a "black box" argument that any decision algorithm in our framework, including fast ones, can be converted into an approximate counting or sampling algorithm with minimal overhead. Concretely, we obtain new algorithms for approximately counting and/or sampling zero-weight subgraphs, graph motifs, and satisfying assignments for first-order models, and our framework is sufficiently general that we believe new applications will be forthcoming.

In Section 1.1, we set out our main results in detail as Theorems 1 and 2, and discuss our edge-counting reduction framework (which is of independent interest). We describe the applications of Theorems 1 and 2 to fine-grained complexity in Section 1.2, and their applications to parameterised complexity in Section 1.3.

1.1 The k-hypergraph framework Given a k-hypergraph $G = (V, E)$, write $e(G) = |E|$, and let

$$\begin{aligned}&\mathcal{C}(G) :=\\ &\quad \{(X_1, \dots, X_k)\colon X_1, \dots, X_k \text{ are disjoint subsets of } V\}.\end{aligned}$$

For any $(X_1, \dots, X_k) \in \mathcal{C}(G)$, we write $G[X_1, \dots, X_k]$ for the k-partite k-hypergraph on $X_1 \cup \dots \cup X_k$ whose edge set is $\{e \in E(G)\colon |e \cap X_i| = 1 \text{ for all } i \in [k]\}$. We define the *coloured independence oracle* of G to be the function $\text{cIND}_G\colon \mathcal{C}(G) \to \{0, 1\}$ such that $\text{cIND}_G(X_1, \dots, X_k) = 1$ if the k-partite k-hypergraph on $G[X_1, \dots, X_k]$, the k-partite k-hypergraph on $X_1 \cup \dots \cup X_k$ whose edge set is $\{e \in E(G)\colon |e \cap X_i| = 1 \text{ for all } i \in [k]\}$,

$G[X_1, \dots, X_k]$ has no edges, and $\text{cIND}_G(X_1, \dots, X_k) = 0$ otherwise. Informally, we think of elements of $\mathcal{C}(G)$ as representing k-colourings of induced subgraphs of G, with X_i being the i'th colour class; thus given a vertex colouring of an induced subgraph of G, the coloured independence oracle outputs 1 if and only if no colourful edge is present. We consider a computation model where the algorithm is given access to V and k, but can only access E via cIND_G. We say that such an algorithm has *coloured oracle access* to G, and for legibility we write it to have G as an input. Our main result is as follows.

THEOREM 1. *There is a randomised algorithm* $\texttt{Count}(G, \varepsilon, \delta)$ *with the following behaviour. Suppose G is an n-vertex k-hypergraph, and that* $\texttt{Count}$ *has coloured oracle access to G. Suppose ε and δ are rational with* $0 < \varepsilon, \delta < 1$. *Then, writing* $T = \log(1/\delta)\varepsilon^{-2}k^{6k}\log^{4k+7} n$: *in time $\mathcal{O}(nT)$, and using at most $\mathcal{O}(T)$ queries to* cIND_G, $\texttt{Count}(G, \varepsilon, \delta)$ *outputs a rational number $\hat{e}$. With probability at least* $1 - \delta$, *we have* $\hat{e} \in (1 \pm \varepsilon)e(G)$.

We note that an analogue of Theorem 1 in a more abstract setting was obtained in subsequent independent work by Bhattacharya, Bishnu, Ghosh and Mishra [9]; our result achieves better running time in terms of ε (ε^{-2} as compared with ε^{-4} in [9]).

As an example of how Theorem 1 applies to approximate counting problems, consider the problem #k-CLIQUE of counting the number of cliques in an n-vertex graph H of size k. We take G to be the k-hypergraph on vertex set $V(H)$ whose hyperedges are precisely those size-k sets which span cliques in G. Thus ε-approximating the number of k-cliques in H corresponds to ε-approximating the number of hyperedges in G. We may use a decision algorithm for k-Clique with running time $f(n,k)$ to evaluate cIND$_G$ in time $f(n,k)$, by applying it to an appropriate subgraph of G (in which we delete all edges within each colour class X_i). Thus Theorem 1 gives us an algorithm for ε-approximating the number of k-cliques in H in time $\mathcal{O}(nT + Tf(n,k))$. Any decision algorithm for k-Clique must read a constant proportion of its input, so we have $f(n,k) = \Omega(n)$ and our overall running time is $\mathcal{O}(Tf(n,k))$. It follows that any decision algorithm for k-clique yields an ε-approximation algorithm for #k-Clique with overhead only $T = \varepsilon^{-2}(k \log n)^{\mathcal{O}(k)}$.

The polynomial dependence on ε in Theorem 1 is not surprising, as by taking $\varepsilon < 1/2n^k$ and rounding we can obtain the number of edges of G exactly. Thus if the dependence on ε were subpolynomial, Theorem 1 would essentially imply a fine-grained reduction from exact counting to decision. This is impossible under SETH in our setting; see [14, Theorem 3] for a more detailed discussion.

We extend Theorem 1 to approximately-uniform sampling as follows.

THEOREM 2. *There is a randomised algorithm* `Sample`(G,ε) *which, given a rational number ε with $0 < \varepsilon < 1$ and coloured oracle access to an n-vertex k-hypergraph G containing at least one edge, outputs either a random edge $f \in E(G)$ or* `Fail`. *For all $f \in E(G)$,* `Sample`(G,ε) *outputs f with probability $(1 \pm \varepsilon)/e(G)$; in particular, it outputs* `Fail` *with probability at most ε. Moreover, writing $T = \varepsilon^{-2}k^{7k}\log^{4k+11} n$,* `Sample`$(G,\varepsilon)$ *runs in time $\mathcal{O}(nT)$ and uses at most $\mathcal{O}(T)$ queries to* cIND$_G$.

We call the output of this algorithm an *ε-approximate sample*. Note that there is a standard trick using rejection sampling which, given an algorithm of the above form, replaces the ε^{-2} factor in the running time by a polylog(ε^{-1}) factor; see [32]. Unfortunately, it does not apply to Theorem 2, as we do not have a fast way to compute the true distribution of `Sample`'s output.

By the same argument as above, Theorem 2 may be used to sample a size-k clique from a distribution with total variation distance at most ε from uniformity with overhead only $T = \varepsilon^{-2}(k \log n)^{\mathcal{O}(k)}$ over decision. (We also note that it is easy to extend Theorems 1 and 2 to cover the case where the original decision algorithm is randomised, at the cost of an extra factor of $k \log n$ in the number of oracle uses; we discuss this further in the full version.)

Theorems 1 and 2 are also of independent interest, generalising known results in the graph oracle literature. Our colourful independence oracles are a natural generalisation of the bipartite independent set (BIS) oracles of Beame et al. [6] to a hypergraph setting, and when $k = 2$ the two notions coincide. Their main result [6, Theorem 4.9] says that given BIS oracle access to an n-vertex graph G, one can ε-approximate the number of edges of G using $\mathcal{O}(\varepsilon^{-4}\log^{14} n)$ BIS queries (which they take as their measure of running time). The $k = 2$ case of Theorem 1 gives a total of $\mathcal{O}(\varepsilon^{-2}\log^{19} n)$ queries used, improving their running time for most values of ε, and Theorem 2 extends their algorithm to approximately-uniform sampling.

When $k = 3$, our colourful independence oracles are similar to the tripartite independent set (TIS) oracles of Bhattacharya et al. [8]. (These oracles ask whether a 3-coloured graph H contains a colourful triangle, rather than whether a 3-coloured 3-hypergraph G contains a colourful edge. But if G is taken to be the 3-hypergraph whose edges are the triangles of H, then the two notions coincide exactly.) Their main result, Theorem 1, says that given TIS oracle access to an n-vertex graph G in which every edge belongs to at most d triangles, one can ε-approximate the number of triangles in G using at most $\mathcal{O}(\varepsilon^{-12}d^{12}\log^{25} n)$ TIS queries. Our Theorem 1 gives an algorithm which requires only $\mathcal{O}(\varepsilon^{-2}\log^{22} n)$ TIS queries, with no dependence on d, and which also generalises to approximately counting k-cliques for all fixed k. Again, Theorem 2 extends the result to approximately-uniform sampling.

We note in passing that the main result of [14] doesn't quite fit into this setting, as it also makes unrestricted use of edge existence queries. It resembles a version of Theorem 1 restricted to $k = 2$ and with slightly lower overhead in n.

1.2 Corollaries in fine-grained complexity

In [14], fine-grained reductions from approximate counting to decision were shown for the problems ORTHOGONAL VECTORS, 3SUM and NEGATIVE-WEIGHT TRIANGLE (among others). The approximate counting procedure for k-uniform hypergraphs in Theorem 1 allows us to generalize these reductions to k-OV, k-SUM, ZERO-WEIGHT k-CLIQUE, and other subgraph isomorphism problems. They also apply to model checking of first-order formulas with k variables. In each case, Theorem 2 yields a corresponding result for approximate sampling of witnesses.

1.2.1 First-order Formulas on Sparse Structures and k Orthogonal Vectors

We consider first-order formulas φ, that is, formulas of the form: $Q_1 x_{\ell+1} Q_2 x_{\ell+2} \ldots Q_{k-\ell} x_k \,.\, \psi(x_1,\ldots,x_k)$. The variables $x_1,\ldots,x_\ell$ are the free variables of φ, each Q_i is a quantifier from $\{\exists,\forall\}$, and ψ is a quantifier-free Boolean formula over the variables $x_1,\ldots,x_k$. We consider first-order formulas in prenex-normal form with $\ell \in \{0,\ldots,k\}$ free variables and quantifier-rank at most $k-\ell$; let k-FO denote the set of all such formulas. The *property testing problem for k-FO* is, given a formula and a structure (e.g., the edge relation of a graph), to decide whether the formula is satisfiable in the structure, that is, whether there is an assignment to the free variables that makes the formula true. Correspondingly, the *property counting problem* is to count all satisfying assignments.

Model checking and property testing are important problems in logic and database theory, and have recently been studied in the context of fine-grained complexity [15, 25, 44]: Gao et al. [25] devise an algorithm for the property testing problem for k-FO that runs in time $m^{k-1}/2^{\Theta(\sqrt{\log m})}$, where m is the number of distinct tuples in the input relations. This improves upon an already slightly non-trivial $\widetilde{\mathcal{O}}(m^{k-1})$ algorithm.[1] By using this improved decision algorithm as a black box, we obtain new algorithms for approximate counting (via Theorem 1) and approximate sampling (via Theorem 2). Note all our approximate counting algorithms work with probability at least $2/3$; this can easily be increased to $1-\delta$ in the usual way, i.e. running them $\mathcal{O}(\log(1/\delta))$ times and taking the median result.

COROLLARY 3. *Fix $k \in \mathbb{Z}_{\geq 0}$, suppose an instance of property testing for k-FO can be solved in time $T(n, m) = \mathcal{O}((m+n)^k)$, where n is the size of the universe and m is the number of tuples in the structure, and write $\mathcal{S}$ for the set of satisfying assignments. Then there is a randomised algorithm to ε-approximate $|\mathcal{S}|$, or draw an ε-approximate sample from $\mathcal{S}$, in time $\varepsilon^{-2} \cdot \widetilde{\mathcal{O}}(T(n, m))$.*

In combination with the algorithm of Gao et al. [25], we can thus ε-approximately sample from the set of satisfying assignments to any k-FO-property in time $\varepsilon^{-2}m^{k-1}/2^{\Theta(\sqrt{\log m})}$. For example, this algorithm can be used to sample an approximately uniformly random solution tuple to a conjunctive query.

The k-ORTHOGONAL VECTORS (k-OV) problem is a specific example of a property testing problem, and has connections to central conjectures in fine-grained complexity theory [1, 25]. The problem asks, given k sets $X_1, \ldots, X_k \subseteq \{0,1\}^D$ of Boolean vectors, whether there exist $x_1 \in X_1$, $\ldots$, $x_k \in X_k$ such that $\sum_{j=1}^{D} \prod_{i=1}^{k} x_{ij} = 0$. (The sum and product are the usual arithmetic operations over $\mathbb{Z}$.) When $x_1, \ldots, x_k$ are viewed as representing subsets of $[D]$ in the canonical manner, this condition is equivalent to requiring they have an empty intersection; when $k = 2$, it is equivalent to x_1 and x_2 being orthogonal. Any tuple $(x_1, \ldots, x_k)$ satisfying the condition is called a *witness*. Clearly, k-OV can be solved in time $O(N^k D)$ using exhaustive search. Gao et al. [25] stated the Moderate-Dimension k-OV Conjecture, which says that k-OV cannot be solved in time $O(N^{k-\varepsilon}\operatorname{poly}(D))$ time for any $\varepsilon > 0$. We show that any reasonable-sized improvement over exhaustive search carries over to approximate counting and sampling.

COROLLARY 4. *Fix $k \geq 2$, suppose an N-vector D-dimension instance of k-OV can be solved in time $T(N, D)$, and write W for the set of witnesses. Then there is a randomised algorithm to ε-approximate $|W|$, or draw an ε-approximate sample from W, in time $\varepsilon^{-2} \cdot \widetilde{\mathcal{O}}(T(N, D))$.*

Note that such an improvement is already known for 2-OV, which has an $N^{2-1/\mathcal{O}(\log(D/\log N))}$-time algorithm [3], although Chan and Williams [12] already generalised this to an exact counting algorithm.

1.2.2 k-SUM

The k-SUM problem has been studied since the 1990s as it arises naturally in the context of computational geometry, see for example [23], and it has become an important problem in fine-grained complexity theory [45]. For all integers $k \geq 3$, the k-SUM problem asks, given a set of integers, whether some k of them sum to zero. Each k-subset of integers that does sum to zero is called a *witness*. While Kane, Lovett, and Moran [33] very recently developed almost linear-size linear decision trees for k-SUM, the fastest known algorithm for this problem still runs in time $\widetilde{\mathcal{O}}(n^{\lceil k/2 \rceil})$, and $n^{o(k)}$ as $k \to \infty$ is ruled out under the exponential-time hypothesis [40]. We prove that any sufficiently non-trivial improvement over the best known decision algorithm carries over to approximate counting and witness sampling.

COROLLARY 5. *Fix $k \geq 3$, suppose an n-integer instance of k-SUM can be solved in time $T(n)$, and write W for the set of witnesses. Then there is a randomised algorithm to ε-approximate $|W|$, or draw an ε-approximate sample from W, in time $\varepsilon^{-2} \cdot \widetilde{\mathcal{O}}(T(n))$.*

1.2.3 EXACT-WEIGHT k-CLIQUE and Other Subgraph Problems

Recall that Theorem 1 applies to the problem #k-CLIQUE. This observation generalizes to other subgraph problems as well. We consider weighted graph problems, where we are given a graph G with an edge-weight function $w : E(G) \to \mathbb{Z}$. The weight of a clique X in G is the sum $\sum_e w(e)$ over all edges $e \in E(G)$ with $e \subseteq X$. The EXACT-WEIGHT k-CLIQUE problem is to decide whether there is a k-clique X of weight exactly 0. It has been conjectured [1] that there is no real $\varepsilon > 0$ and integer $k \geq 3$ such that the EXACT-WEIGHT k-CLIQUE problem on n-vertex graphs and with edge-weights in $\{-M, \ldots, M\}$ can be solved in time $\mathcal{O}(n^{(1-\varepsilon)k}\operatorname{polylog}(M))$. (For the closely related MIN-WEIGHT k-CLIQUE problem, a subpolynomial-time improvement over the exhaustive search algorithm is known [1, 43, 12], with running time $n^k/\exp(\Omega(\sqrt{\log n}))$.) Theorems 1 and 2 imply that any sufficiently non-trivial improvement on the running time of an EXACT-WEIGHT k-CLIQUE algorithm will carry over to the approximate counting and sampling versions of the problem.

COROLLARY 6. *Fix $k \geq 3$, suppose an n-vertex m-edge instance of* EXACT-WEIGHT k-CLIQUE *with weights in*

[1] The notation $\widetilde{\mathcal{O}}(f(n, m))$ means $f(n, m) \cdot \operatorname{polylog}(n + m)$.

$[-M, M]$ can be solved in time $T(n, m, M)$, and write $\mathcal{C}$ for the set of zero-weight k-cliques. Then there is a randomised algorithm to ε-approximate $|\mathcal{C}|$, or draw an ε-approximate sample from $\mathcal{C}$, in time $\varepsilon^{-2} \cdot \widetilde{\mathcal{O}}(T(n, m, M))$.

There is a more general version of EXACT-WEIGHT k-CLIQUE which takes as input an edge-weighted d-hypergraph and asks whether it contains a zero-weight k-clique. A similar conjecture exists for this version of the problem [1], and Theorems 1 and 2 yield a result analogous to Corollary 6.

Our framework also applies to subgraphs more general than cliques. The EXACT-WEIGHT-H problem asks, given an edge-weighted graph G, whether there exists a subgraph of G that has weight zero and is isomorphic to H. We say H is a *core* if every homomorphism from H to H is also an automorphism. Cores are a rich class of graphs, including cliques, odd cycles, and (with high probability) any binomial random graph $G(n, p)$ with edge probability $n^{-1/3}\log^2 n < p < 1 - n^{1/3}\log^2 n$ (see [11, Theorem 2]). Corollary 6 generalises to EXACT-WEIGHT-H whenever H is a core. In particular, Abboud and Lewi [2, Corollary 5] prove that EXACT-WEIGHT-H can be solved in time $\widetilde{\mathcal{O}}(n^{\gamma(H)})$, where $\gamma(H) \geq 1$ is a graph parameter that is small whenever H has a balanced separator, so we obtain the following result.

COROLLARY 7. *Let H be a core, let G be an n-vertex graph, and let $H(G)$ be the set of zero-weight H-subgraphs in G. There is an algorithm to draw an ε-approximate sample from $H(G)$ in time $\widetilde{\mathcal{O}}(\varepsilon^{-2} n^{\gamma(H)})$.*

Our framework also applies to colourful subgraphs. The COLOURFUL-H problem asks, given a graph G and a vertex colouring $c\colon V(G) \to \{1, \dots, |V(H)|\}$, whether G contains a colourful copy of H — that is, a subgraph isomorphic to H containing one vertex from each colour class.

COROLLARY 8. *Let H be a fixed graph, suppose an n-vertex m-edge instance of* COLOURFUL-H *can be solved in time $T(m, n)$, and write $\mathcal{H}$ for the set of colourful H-subgraphs. Then there is a randomised algorithm to ε-approximate $|\mathcal{H}|$, or draw an ε-approximate sample from $\mathcal{H}$, in time $\varepsilon^{-2} \cdot \widetilde{\mathcal{O}}(T(m, n))$.*

Daz, Serna, and Thilikos [16] show using dynamic programming that #COLOURFUL-H can be solved exactly in time $\widetilde{\mathcal{O}}(n^{t+1})$, where t is the treewidth of H. Marx [36] asks whether it is possible to detect colourful subgraphs in time $n^{o(t)}$, and proves that $n^{o(t/\log t)}$ is impossible under the exponential-time hypothesis (ETH). Our result shows that any algorithm to detect colourful subgraphs in time $n^{o(t)}$ would essentially also have to approximately count these subgraphs — a more difficult task.

1.3 Corollaries in parameterised complexity

When considering approximation algorithms for parameterised counting problems, an "efficient" approximation scheme is an FPTRAS (fixed parameter tractable randomised approximation scheme), as introduced by Arvind and Raman [5]; this is the analogue of an FPRAS in the parameterised setting. An FPTRAS for a parameterised counting problem Π with parameter k is an algorithm that takes an instance I of Π (with $|I| = n$) and a rational number $\varepsilon > 0$, and in time $f(k) \cdot \text{poly}(n, 1/\varepsilon)$ (where f is some computable function) outputs a rational number z such that

$$\mathbb{P}[(1-\varepsilon)\Pi(I) \leq z \leq (1+\varepsilon)\Pi(I)] \geq 2/3.$$

Note that this definition is equivalent to that given in [5] which requires the failure probability to be at most δ, where δ is part of the input; repeating the process above $\mathcal{O}(\log(1/\delta))$ times and returning the median solution allows us to reduce the error probability from $1/3$ to δ.

As mentioned above, a large number of well-studied problems in parameterised complexity fall within our k-hypergraph framework; for standard notions in parameterised (counting) complexity we refer the reader to [21]. Observe that we can rewrite our overhead of $\log^{\mathcal{O}(k)} n$ in the form $k^{2k} n^{o(1)}$: if $k \leq \log n/(\log\log n)^2$ then $\log^{\mathcal{O}(k)} n = e^{\mathcal{O}(\log n/\log\log n)} = n^{o(1)}$, and if $k \geq \log n/(\log\log n)^2$ then $\log^{\mathcal{O}(k)} n = \mathcal{O}(k^{2k})$. Thus we can consider this to be a "fine-grained FPT overhead".

Theorems 1 and 2 can therefore be applied immediately to any *self-contained k-witness problem* (see [38]); that is, any problem with integer parameter k in which we are interested in the existence of witnesses consisting of k-element subsets of some given universe, and we have the ability to quickly test whether any given k-element set is such a witness. Examples include weight-k solutions to CSPs, size-k solutions to database queries, and sets of k vertices in a (weighted) graph or hypergraph which induce a sub(hyper)graph with specific properties. This last example encompasses many of the best-studied problems in parameterised counting complexity, including the problem #SUB(H, G) (with parameter $|V(H)|$) which asks for the number of subgraphs of G isomorphic to H; the well-studied problems of counting k-vertex paths, cycles and cliques are all special cases. More generally, we can consider the problem #INDUCED SUBGRAPH WITH PROPERTY(Φ) (#ISWP(Φ)), introduced by Jerrum and Meeks [30], for any property Φ.

However, our coloured independence oracle doesn't quite correspond to deciding whether a witness exists: it needs to solve a *multicolour* version of the decision problem. The multicolour decision version of a self-contained k-witness problem takes as input a universe U together with a k-colouring of the elements of U, and asks whether there exists a witness which contains precisely one element of each colour. The following result is immediate from Theorems 1 and 2 on taking the vertex set of the hypergraph to be U, the edges to be the k-witnesses, and simulating the coloured independence oracle by invoking a multicolour decision algorithm.

THEOREM 9. *Let Π be a self-contained k-witness decision problem, and suppose that the multicolour version of Π can be solved in time $T(n,k)$ when the universe U has size n. Let $c\colon U \to [k]$ be a colouring, let W be the set of (uncoloured) witnesses of Π, and let W^c be the set of multicolour witnesses of Π with respect to c. Then given U and c, in time $\varepsilon^{-2}k^{2k}n^{o(1)}T(n,k)$, there is a randomised algorithm to ε-approximate $|W|$ or $|W^c|$, or draw an ε-approximate sample from W or W^c.*

Such multicolour problems have been studied before in the literature, including #MISWP(Φ), the multicolour version of #ISWP(Φ); see [37] for a survey of results relating the complexity of multicolour and uncoloured problems in this setting. In many cases, the multicolour decision problem reduces straightforwardly to the original decision problem — for example, if our witnesses are k-vertex cliques in a graph. But this is not true in general; if our witnesses are k-vertex cliques *and* k-vertex independent sets, then the uncoloured decision problem admits a trivial FPT algorithm by Ramsey's theorem [5], but the W[1]-complete problem k-CLIQUE reduces to the multicolour version [37]. In the restricted setting of SUB(H,G), it is straightforward to verify that the multicoloured and uncoloured versions of the problem are equivalent when the graph H is a core, but this is not known for general H. In fact, a proof of equivalence would imply the long-standing dichotomy conjecture for the parameterised embedding problem (see [13] for recent progress on this conjecture). We believe that Theorem 9 motivates substantial further research into the complexity relationship between multicoloured problems and their uncoloured counterparts.

One consequence of Theorem 9 is that if MISWP(Φ) admits an FPT decision algorithm, then we obtain FPTRASes for both #MISWP(Φ) and #ISWP(Φ) with roughly the same running time as the original decision algorithm. This generalises a previous result of Meeks [37, Corollaries 4.8 and 4.10] which states that subject to standard complexity-theoretic assumptions, if we restrict our attention to properties Φ that are preserved under adding edges, there is an FPTRAS for the counting problems #MISWP(Φ) and #ISWP(Φ) if and only if there is an FPT decision algorithm for MISWP(Φ). Theorem 9 strengthens this result in two ways. Firstly, we no longer need the restriction that the property is preserved under adding edges, as we can now consider an arbitrary property Φ. Secondly, we demonstrate a close relationship between the running-times for decision and approximate counting, meaning that any improvement in a decision algorithm immediately translates to an improved algorithm for approximate counting.

One example where Theorem 9 already gives an improvement (in almost all settings) to the previously best-known algorithm for approximate counting is the GRAPH MOTIF problem, introduced by Lacroix, Fernandes and Sagot [35] in the context of metabolic networks. This problem takes as input an n-vertex m-edge graph with a (not necessarily proper) vertex-colouring, together with a multiset M of colours, and a solution is a subset U of $|M| = k$ vertices such that the subset induced by U is connected and the colour multiset of U is exactly M; M is called a *motif*, and we call U a *motif witness* for M.

There has been substantial progress in recent years on improving the running-time of decision algorithms for GRAPH MOTIF [7, 10, 19, 26, 34], with the fastest randomised algorithm [10] (based on constrained multilinear detection) running in time $\mathcal{O}(2^k k^3 m)$. For the counting version, Guillemot and Sikora [26] addressed the related problem of counting k-vertex sub*trees* of a graph whose vertex set has colour multiset M (which counts motif witnesses U for M weighted by the the number of trees spanned by U). They demonstrated that this problem admits an FPT algorithm for exact counting when M is a set, but is #W[1]-hard otherwise. Subsequently, Jerrum and Meeks [30] addressed the more natural counting analogue of GRAPH MOTIF in which the goal is to count motif witnesses for M without weights. They demonstrated that this problem is #W[1]-hard to solve exactly even if M is a set, but gave an FPTRAS to solve it approximately. By using this FPTRAS together with Theorems 1 and 2, we prove the following.

COROLLARY 10. *Given an n-vertex instance of GRAPH MOTIF with parameter k and $0 < \varepsilon < 1$, there is a randomised algorithm to ε-approximate the number of motif witnesses or to draw an ε-approximate sample from the set of motif witnesses in time $\mathcal{O}(\varepsilon^{-2}k^{8k}m\log^{4k+8} n)$.*

Theorem 9 also generalises a known relationship between the complexity of uncoloured approximate counting and multicolour decision in the special case of SUB(H,G). In this restricted setting, multicolour decision is actually equivalent to multicolour exact counting; there is an FPT algorithm to exactly count the number of multicolour solutions whenever the treewidth of H is bounded by a constant, with essentially the same running time as the best-known decision algorithm [4]. On the other hand, even the multicolour decision problem is W[1]-hard if H is restricted to any class of graphs with unbounded treewidth [37]. Alon et al. [29] essentially give a fine-grained reduction from uncoloured approximate counting to multicolour exact counting, giving an algorithm with running time matching the best-known algorithm for multicolour decision. (Note that their running time is slightly better than that obtained by applying Theorem 9, and that uncoloured exact counting is #W[1]-hard even when H is a path or cycle [22].)

However, in general it is not true that multicolour exact counting is equivalent to multicolour decision — indeed, there are natural examples (such as counting k-vertex subsets that induce connected subgraphs) in which the counting is #W[1]-hard but the decision is FPT [30]. Theorem 9 therefore strengthens [29], in the sense that if a faster multicolour decision algorithm is discovered then the improvement to the running time will immediately be carried over to uncoloured approximate counting, whether or not the new algorithm generalises to exact multicolour counting.

In this specific case, the existing decision algorithm turns out to already give an algorithm for exact counting with the same asymptotic complexity; however, there is no theoretical reason why the constant in the exponent could not be improved, and our results mean that any such improvement in a decision algorithm could immediately be translated to a faster algorithm for approximate counting.

Organisation. In Section 2, we set out our notation. We sketch the proof of Theorem 1 in Section 3, using a weaker approximation algorithm which we set out in Section 4. We sketch the proof of Theorem 2 (using Theorem 1) in Section 5.

2 Notation

Let $k \geq 2$ and let $G = (V, E)$ be a k-hypergraph, so that each edge in E has size exactly k. We write $e(G) = |E|$. For all $U \subseteq V$, we write $G[U]$ for the subgraph induced by U. For all $S \subseteq V$, we write $d_H(S) = |\{e \in E(G)\colon S \subseteq e\}|$ for the degree of S in H. If $S = \{v_1, \dots, v_{|S|}\}$, then we will sometimes write $d_H(v_1, \dots, v_{|S|}) = d_H(S)$.

For all positive integers t, we write $[t] = \{1, \dots, t\}$. We write $\ln$ for the natural logarithm, and $\log$ for the base-2 logarithm. Given real numbers $x, y \geq 0$ and $0 < \varepsilon < 1$, we say that x is an *ε-approximation* to y if $(1-\varepsilon)x < y < (1+\varepsilon)x$, and write $y \in (1 \pm \varepsilon)x$. We extend this notation to other operations in the natural way, so that (for example) $y \in xe^{\pm\varepsilon}/(2 \mp \varepsilon)$ means that $xe^{-\varepsilon}/(2+\varepsilon) \leq y \leq xe^{\varepsilon}/(2-\varepsilon)$.

When stating bounds on running times of algorithms, we assume the standard randomised word-RAM machine model with logarithmic-sized words; thus given an input of size N, we can perform arithmetic operations on $\mathcal{O}(\log N)$-bit words and generate uniformly random $\mathcal{O}(\log N)$-bit words in $\mathcal{O}(1)$ time.

Recall the definitions of $\mathcal{C}(G)$ and the coloured independence oracle of G, and coloured oracle access from Section 1.1. Note that for all $X \subseteq V(G)$, cIND$_{G[X]}$ is a restriction of cIND$_G$. Thus an algorithm with coloured oracle access to G can safely call a subroutine that requires coloured oracle access to $G[X]$.

3 The main algorithm

In this section we sketch the proof of our main approximate counting result, Theorem 1. We will make use of an algorithm with a weaker approximation guarantee. We state its properties in the following lemma, whose proof we will sketch in Section 4.

LEMMA 11. *There is a randomised algorithm* `Coarse`(G, δ) *with the following behaviour. Suppose G is an n-vertex k-hypergraph to which* `Coarse` *has (only) coloured oracle access, where n is a power of two, and suppose $0 < \delta < 1$. Then in time $\mathcal{O}(\log(1/\delta)k^{3k}n\log^{2k+2}n)$, and using at most $\mathcal{O}(\log(1/\delta)k^{3k}\log^{2k+2}n)$ queries to* cIND$_G$, `Coarse`(G,δ) *outputs a rational number $\hat{e}$. Moreover, with probability at least $1-\delta$,*

$$\frac{e(G)}{2(4k\log n)^k} \leq \hat{e} \leq e(G) \cdot 2(4k\log n)^k.$$

Write $n = 2^\ell$ for some integer ℓ. We first set out a toy algorithm for the purpose of illustration. Let t be a suitably large integer, and take independent uniformly random subsets $X_1, \dots, X_t \subseteq V(G)$ subject to $|X_i| = 2^{\ell-1}$ for all $i \in [t]$. It is not hard to show that $\mathbb{E}(e(G[X_i])) \approx e(G)/2^k$ for all i. Thus, using Hoeffding's inequality, we can show that the total number of edges $\sum_{i=1}^t e(G[X_i])$ is concentrated around its mean of roughly $te(G)/2^k$. It follows that, with high probability, $(2^k/t)\sum_{i=1}^t e(G[X_i]) \approx e(G)$.

Repeating this expansion procedure yields the following (bad) algorithm. We maintain a list L of pairs (w, X), where $w \in \mathbb{Q}$ is positive and $X \subseteq V(G)$, and we preserve the invariant $\sum_{(w,X)\in L} we(G[X]) \approx e(G)$ with high probability. (We expect the quality of approximation to degrade as the algorithm runs, but we ignore this subtlety in our sketch.) Initially, we take $L = (1, V(G))$, which clearly satisfies this invariant. At each stage, for each pair $(w, X) \in L$, we independently choose t uniformly random subsets $X_1, \dots, X_t \subseteq X$ subject to $|X_i| = |X|/2$ for all i, as above. We then delete (w, X) from L and replace it by $(2^k w/t, X_1), \dots, (2^k w/t, X_t)$. Thus, as we proceed, L grows, but the sets X in L's entries become smaller, and the invariant $\sum_{(w,X)\in L} we(G[X]) \approx e(G)$ is maintained. Eventually, the entries of L become so small that for all $(w, X) \in L$, we can use cIND$_G$ to count $e(G[X])$ quickly by brute force, and we are done.

The problem with the algorithm described above is that in order to maintain the invariant with high probability, we must take $t = \Omega(\varepsilon^{-2}\log n)$, and to bring the vertex sets in L down to a manageable size we require $\Omega(\log n)$ expansion operations. Thus our final list will have length $(\varepsilon^{-2}\log n)^{\Omega(\log n)}$, resulting in an algorithm with superpolynomial running time. We avoid this problem by exploiting a statistical technique called importance sampling, previously applied to the $k = 2$ case by Beame et al. [6]. Given a coarse estimate of each $e(G[X_i])$, as found by `Coarse`, this technique allows us to prune L to a manageable length in $\mathcal{O}(|L|)$ time, while maintaining the invariant $\sum_{(w,X)\in L} we(G[X]) \approx e(G)$ with high probability. We set out our algorithm for this, `Trim`, in the full version; it gives a substantially shorter list than the algorithm used in [6], thereby improving our running time.

Unlike [6], we also use the output of `Coarse` to improve the efficiency of our expansion procedure. The algorithm described above treats all pairs $(w, X) \in L$ equally, expanding each one into t smaller pairs. Thus L grows by a factor of t in a single expansion step. Our real algorithm will work differently. For each pair (w_i, X_i), we will choose the number t_i of replacement pairs according to our coarse estimate of $w_i e(G[X_i])$. We will take t_i to be large if (w_i, X_i) accounts for a large proportion of $\sum_{(w,X)\in L} we(G[X])$, and small otherwise; thus we only spend a lot of time processing a pair if it is "important". This optimisation, together with the improved importance sampling procedure discussed above, drops our running time by a factor of roughly ε^{-2}. We therefore improve the results of [6] even when $k = 2$. In the full version, we set out our expansion procedure as `Halve`.

Overall, a sketch implementation of `Count` is as follows. Let $I = \log n - \lceil \log(2k^2) \rceil$, and let $\delta = 1/3(2I+1)$. Initially, we take $L = (1, V(G), \texttt{Coarse}(G, \delta))$, and we maintain the invariants that

$$\sum_{(w,S,\hat{e})\in L} we(G[S]) \approx e(G),$$

$$\hat{e} = \texttt{Coarse}(G[S], \delta) \text{ for all } (w, S, \hat{e}) \in L.$$

Then we update $L \leftarrow \texttt{Halve}(\texttt{Trim}(L))$ a total of I times. Each invocation of `Trim` reduces the length of L to $\varepsilon^{-2} \log^{\mathcal{O}(k)} n$, and after the i'th invocation of `Halve` we have $|S| = n/2^i$ for all $(w, S, \hat{e}) \in L$. Then, for all $(w, S, \hat{e}) \in L$, we calculate $e(G[S])$ by brute force using cIND_G; this is fast since `Halve` guarantees that $|S| = \mathcal{O}(k^2)$, and since `Trim` guarantees that L is short. We then output $\sum_{(w,S,\hat{e})\in L} we(G[S]) \approx e(G)$. Note we have glossed over several technical details, such as some degradation of our invariant with repeated invocations of `Trim` and `Halve`, which we cover in detail in the full version.

4 Coarse approximate counting

The heart of our proof for Lemma 11 is a subroutine to solve the following simpler "gap-version" of the approximation problem. Given a k-partite k-hypergraph G, to which we have (only) coloured oracle access, and a guess $M \geq 0$, we ask: Does G have more than M edges? We wish to answer correctly with high probability provided that either G has at least M edges, or G has significantly fewer than M edges, namely at most γM edges with $\gamma = 1/(2^{3k+1} k^{2k} \log^k n)$.

Suppose we can solve this problem probabilistically, perhaps outputting `Yes` with probability at least $1/50$ if $e(G) \geq M$ (which we call *completeness*) and outputting `Yes` with probability at most $1/100$ if $e(G) \leq \gamma M$ (which we call *soundness*). We then apply probability amplification to substantially reduce the failure probability, and use binary search to find the least M such that our output is `Yes` — with high probability, this will approximate $e(G)$ when our input k-hypergraph is k-partite. We then generalise our algorithm from k-partite inputs to arbitrary inputs using random colour-coding. These parts of the algorithm are fairly standard, so in this section we will only sketch our solution to the gap-problem.

Let G be a k-partite k-hypergraph with vertex classes $X_1, \ldots, X_k$, and for simplicity suppose $n = |V(G)|$ is a power of two. The basic idea of the algorithm is to randomly remove vertices from G to form a new graph H in such a way that each edge survives with probability roughly $1/M$, and then query the coloured independence oracle and output `Yes` if and only if at least one edge remains. If G has at most γM edges, then a union bound implies we are likely to output `No` (soundness); if G has at least M edges, then in expectation at least one edge survives the removal, so we hope to output `Yes` (completeness). Unfortunately, the number of edges remaining in H need not be concentrated around its expectation
a single vertex v — so we must be very careful if this hope is to be realised.

Suppose for the moment that $k = 2$, so that G is a bipartite graph with vertex classes X_1 and X_2. Then we will form $X_1' \subseteq X_1$ by including each vertex independently with some probability p_1, and $X_2' \subseteq X_2$ by including each vertex independently with some probability p_2. Each edge survives with probability $p_1 p_2$, so we require $p_1 p_2 \leq 1/M$ to ensure soundness. To ensure completeness, we would then like to choose p_1 and p_2 such that $G[X_1', X_2']$ is likely to contain an edge whenever $e(G) \geq M$.

To see that such a pair (p_1, p_2) exists, we first partition the vertices in X_1 according to their degree: For $1 \leq d \leq \log n$, let X_1^d be the set of vertices v with $2^{d-1} \leq d(v) < 2^d$. By the pigeonhole principle, there exists some D such that X_1^D is incident to at least $e(G)/\log n$ edges. We take $p_1 = 2^D/M$ and $p_2 = 1/2^D$. We certainly have $p_1 p_2 \leq 1/M$. Suppose $e(G) \geq M$. Since X_1^D is incident to at least $e(G)/\log n$ edges, we have $|X_1^D| \geq M/2^D \log n$, so with reasonable probability X_1' contains a vertex $v_1 \in X_1^D$. Then v_1 has degree roughly 2^D in X_2, so again with reasonable probability X_2' contains a vertex adjacent to it.

There is one remaining obstacle: Since we only have coloured oracle access to G, we do not know what D is! Fortunately, since there are only $\mathcal{O}(\log n)$ possibilities, we can simply try them all in turn, and output `Yes` if any one of them yields a pair X_1', X_2' such that $G[X_1', X_2']$ contains an edge. (It is not hard to tune the parameters so that this doesn't affect soundness.) This is essentially the argument used by Beame et al. [6].

When we try to generalise this approach to k-hypergraphs, we hit a problem. For illustration, take $k = 3$ and suppose $e(G) \geq M$. Then we wish to guess a vector (p_1, p_2, p_3) such that $p_1 p_2 p_3 \leq 1/M$ and, with reasonable probability, $G[X_1', X_2', X_3']$ contains an edge. As in the $k = 2$ case, we can guess an integer $0 \leq D \leq 2 \log n$ such that a large proportion of G's edges are incident to a vertex in X_1 of degree roughly 2^D. Also, if we take $p_1 = 2^D/M$ then it is reasonably likely that X_1' will contain a vertex of degree roughly 2^D, say v_1. But we cannot iterate this process — the structure of $G[v_1, X_2, X_3]$, and hence the "correct" value of p_2, depends very heavily on v_1. So for example, when we test the two guesses $(2^D/M, 1/2^D, 1)$ and $(2^D/M, 1, 1/2^D)$, we wish to ensure that the value of v_1 is the same in each test. This is the reason for step (C1) in the following algorithm; it is important that we do not choose new random subsets of $X_1, \dots, X_k$ independently with each iteration of step (C2).

Algorithm `VerifyGuess`$(G, M, X_1, \dots, X_k)$**.**

Input: G is an n-vertex k-hypergraph to which `VerifyGuess` has (only) coloured oracle access. n and M are positive powers of two, and $X_1, \dots, X_k \subseteq V(G)$ are disjoint.

Behaviour: Let $p_{\mathsf{out}} = (8k \log n)^{-k}$, and let $\gamma = p_{\mathsf{out}}/2(k \log n)^k$.
Completeness: If $e(G[X_1, \dots, X_k]) \geq M$, then $\mathbb{P}($`VerifyGuess` outputs `Yes`$) \geq p_{\mathsf{out}}$.
Soundness: If $e(G[X_1, \dots, X_k]) < \gamma M$, then $\mathbb{P}($`VerifyGuess` outputs `Yes`$) \leq p_{\mathsf{out}}/2$.

(C1) For each $i \in [k]$ and each $0 \leq j \leq k \log n$, construct a subset $Y_{i,j}$ of X_i by including each vertex independently with probability $1/2^j$. Construct the finite set A of all tuples $(a_1, \dots, a_k)$ with $0 \leq a_1, \dots, a_k \leq k \log n$ and $a_1 + \dots + a_k \geq \log M$.

(C2) For each tuple $(a_1, \dots, a_k) \in A$: If $\mathrm{cIND}_G(Y_{1,a_1}, \dots, Y_{k,a_k}) = 0$, then halt and output `Yes`.

(C3) We have not halted yet, but do so now and output `No`.

In the full version, we formalise the above sketch to prove that `VerifyGuess` behaves correctly, and use it to prove Lemma 11. It is not hard to show that `VerifyGuess` runs in time $\mathcal{O}(k^{3k} n \log^{2k+2} n)$ and requires $\mathcal{O}(k^{3k} \log^{2k+2} n)$ oracle queries.

5 Approximately uniform sampling

In this section we sketch the proof of Theorem 2 from Theorem 1. Suppose for the moment that we are given an *exact* counting algorithm `Count`(G) which, given coloured oracle access to an n-vertex k-hypergraph G, returns $e(G)$. The standard approach for reducing sampling to counting, as used in [32], would essentially be to choose an arbitrary vertex $v \in V(G)$, and then to include v in the output edge with probability `Count`$(G - v)/$`Count`(G). If v is not included, then update $G \mapsto G - v$; if v is included, then update G to the $(k-1)$-hypergraph with vertex set $G - v$ and edge set $\{e \setminus \{v\} \colon v \in e \in E(G)\}$. Then repeat the process until we have a k-element output edge. This approach has two problems in our setting. First, our problems are not self-reducible in this sense; in general, we cannot efficiently simulate the coloured independence oracle of the $(k-1)$-hypergraph described above. And second, in general this approach requires $\Omega(n)$ invocations of the coloured independence oracle, which is far more than the statement of Theorem 2 allows.

Suppose for simplicity that n and k are both powers of two. Then our approach is as follows. Let $X_1 = V(G)$. We sample a uniformly-random size-$(n/2)$ subset $X \subset V(G)$. With probability `Count`$(G[X])/$`Count`$(G[X_1])$, we "accept" this set and let $X_2 = X$. Otherwise, we "reject" it and resample X, repeating the process until we have chosen X_2. This is an example of rejection sampling (see e.g. Florescu [20, Proposition 3.3]), so for all size-$(n/2)$ sets $X \subset V(G)$, the probability that $X_2 = X$ is proportional to $e(G[X])$. Moreover, one can show that $\mathcal{O}(2^k)$ samples are required in expectation before accepting a set X_2. We then repeat the process to find a size-$(n/4)$ set $X_3 \subset V(G)$ such that $\mathbb{P}(X_3 = X)$ is proportional to $e(G[X_3])$, and so on until reaching a size-k set $X_{\log n - \log k + 1}$. By our invariant that $\mathbb{P}(X_r = X)$ is proportional to $e(G[X_r])$ for all r, this set will be a uniformly-chosen edge. The running time and oracle usage of our algorithm is then dominated by our $\mathcal{O}(2^k \log n)$ calls to `Count`.

Of course, Theorem 1 does not actually give us an exact counting algorithm `Count`(G), but an approximate counting algorithm `Count`(G, ε, δ) whose output has relative error ε and which fails with probability δ; in particular, `Count` may output zero when the correct answer is non-zero or vice versa. This makes the analysis more technical, in ways we defer to the full version, but the idea remains the same.

References

[1] Amir Abboud, Karl Bringmann, Holger Dell, and Jesper Nederlof. More consequences of falsifying SETH and the orthogonal vectors conjecture. In Ilias Diakonikolas, David Kempe, and Monika Henzinger, editors, *Proceedings of the 50th Annual ACM SIGACT Symposium on Theory of Computing, STOC 2018, Los Angeles, CA, USA, June 25-29, 2018*, pages 253–266. ACM, 2018.

[2] Amir Abboud and Kevin Lewi. Exact weight subgraphs and the k-sum conjecture. In Fedor V. Fomin, Rusins Freivalds, Marta Z. Kwiatkowska, and David Peleg, editors, *Automata, Languages, and Programming - 40th International Colloquium, ICALP 2013, Riga, Latvia, July 8-12, 2013, Proceedings, Part I*, volume 7965 of *Lecture Notes in Computer Science*, pages 1–12. Springer, 2013.

[3] Amir Abboud, Richard Ryan Williams, and Huacheng Yu. More applications of the polynomial method to algorithm design. In Piotr Indyk, editor, *Proceedings of the Twenty-Sixth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2015, San Diego, CA, USA, January 4-6, 2015*, pages 218–230. SIAM, 2015.

[4] Noga Alon, Raphael Yuster, and Uri Zwick. Color-coding. *J. ACM*, 42(4):844–856, July 1995.

[5] V. Arvind and Venkatesh Raman. Approximation algorithms for some parameterized counting problems. In P. Bose and P. Morin, editors, *ISAAC 2002*, volume 2518 of *LNCS*, pages 453–464. Springer-Verlag Berlin Heidelberg, 2002.

[6] Paul Beame, Sariel Har-Peled, Sivaramakrishnan Natarajan Ramamoorthy, Cyrus Rashtchian, and Makrand Sinha. Edge estimation with independent set oracles. In *9th Innovations in Theoretical Computer Science Conference, ITCS 2018, January 11-14, 2018, Cambridge, MA, USA*, pages 38:1–38:21, 2018.

[7] Nadja Betzler, René van Bevern, Michael Fellows, Christian Komusiewicz, and Rolf Niedermeier. Parameterized algorithmics for finding connected motifs in biological networks. *IEEE/ACM Trans. Comput. Biology Bioinform.*, 8(5):1296–1308, 2011.

[8] Anup Bhattacharya, Arijit Bishnu, Arijit Ghosh, and Gopinath Mishra. Triangle estimation using polylogarithmic queries. *CoRR*, abs/1808.00691, 2018.

[9] Anup Bhattacharya, Arijit Bishnu, Arijit Ghosh, and Gopinath Mishra. Triangle estimation using polylogarithmic queries. *CoRR*, abs/1908.04196, 2019.

[10] Andreas Björklund, Petteri Kaski, and Łukasz Kowalik. Constrained multilinear detection and generalized graph motifs. *Algorithmica*, 74(2):947–967, Feb 2016.

[11] Anthony Bonato and Pawel Prałat. The good, the bad, and the great: Homomorphisms and cores of random graphs. *Discrete Mathematics*, 309(18):5535–5539, 2009.

[12] Timothy M. Chan and Ryan Williams. Deterministic apsp, orthogonal vectors, and more: Quickly derandomizing razborov-smolensky. In Robert Krauthgamer, editor, *Proceedings of the Twenty-Seventh Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2016, Arlington, VA, USA, January 10-12, 2016*, pages 1246–1255. SIAM, 2016.

[13] Yijia Chen, Martin Grohe, and Bingkai Lin. The hardness of embedding grids and walls. In Hans L. Bodlaender and Gerhard J. Woeginger, editors, *Graph-Theoretic Concepts in Computer Science*, pages 180–192, Cham, 2017. Springer International Publishing.

[14] Holger Dell and John Lapinskas. Fine-grained reductions from approximate counting to decision. In *Proceedings of the 50th Annual ACM SIGACT Symposium on Theory of Computing, STOC 2018, Los Angeles, CA, USA, June 25-29, 2018*, pages 281–288, 2018.

[15] Holger Dell, Marc Roth, and Philip Wellnitz. Counting answers to existential questions. In Christel Baier, Ioannis Chatzigiannakis, Paola Flocchini, and Stefano Leonardi, editors, *46th International Colloquium on Automata, Languages, and Programming, ICALP 2019, July 9-12, 2019, Patras, Greece.*, volume 132 of *LIPIcs*, pages 113:1–113:15. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, 2019.

[16] Josep Díaz, Maria J. Serna, and Dimitrios M. Thilikos. Counting h-colorings of partial k-trees. *Theor. Comput. Sci.*, 281(1-2):291–309, 2002.

[17] Martin Dyer, Leslie Ann Goldberg, Catherine Greenhill, and Mark Jerrum. The relative complexity of approximate counting problems. *Algorithmica*, 38(3):471–500, 2004.

[18] Martin E. Dyer, Leslie Ann Goldberg, and Mark Jerrum. An approximation trichotomy for boolean #CSP. *J. Comput. Syst. Sci.*, 76(3-4):267–277, 2010.

[19] Michael Fellows, Guillaume Fertin, Danny Hermelin, and Stéphane Vialette. Sharp tractability borderlines for finding connected motifs in vertex-colored graphs. In *Automata, Languages and Programming, 34th International Colloquium, ICALP 2007, Wroclaw, Poland, July 9-13, 2007, Proceedings*, pages 340–351, 2007.

[20] Ionuţ Florescu. *Probability and Stochastic Processes*. Wiley-Blackwell, 2014.

[21] J. Flum and M. Grohe. *Parameterized Complexity Theory*. Springer, 2006.

[22] Jörg Flum and Martin Grohe. The parameterized complexity of counting problems. *SIAM J. Comput.*, 33(4):892–922, April 2004.

[23] Anka Gajentaan and Mark H. Overmars. On a class of o(n^2) problems in computational geometry. *Comput. Geom.*, 45(4):140–152, 2012.

[24] Andreas Galanis, Leslie Ann Goldberg, and Mark Jerrum. A complexity trichotomy for approximately counting list H-colorings. *TOCT*, 9(2):9:1–9:22, 2017.

[25] Jiawei Gao, Russell Impagliazzo, Antonina Kolokolova, and Ryan Williams. Completeness for first-order properties on sparse structures with algorithmic applications. In *Proceedings of the Twenty-Eighth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2017, Barcelona, Spain, Hotel Porta Fira, January 16-19*, pages 2162–2181, 2017.

[26] Sylvain Guillemot and Florian Sikora. Finding and counting vertex-colored subtrees. *Algorithmica*, 65(4):828–844, Apr 2013.

[27] Heng Guo, Chao Liao, Pinyan Lu, and Chihao Zhang. Counting hypergraph colourings in the local lemma regime. In *Proceedings of the 50th Annual ACM SIGACT Symposium on Theory of Computing, STOC 2018, Los Angeles, CA, USA, June 25-29, 2018*, pages 926–939, 2018.

[28] Heng Guo, Chao Liao, Pinyan Lu, and Chihao Zhang. Zeros of holant problems: locations and algorithms. In *Proceedings of the Thirtieth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2019, San Diego, California, USA, January 6-9, 2019*, pages 2262–2278, 2019.

[29] Fereydoun Hormozdiari, Iman Hajirasouliha, Noga Alon, Phuong Dao, and S. Cenk Sahinalp. Biomolecular network motif counting and discovery by color coding. *Bioinformatics*, 24(13):i241–i249, 07 2008.

[30] Mark Jerrum and Kitty Meeks. The parameterised complexity of counting connected subgraphs and graph motifs. *Journal of Computer and System Sciences*, 81(4):702 – 716, 2015.

[31] Mark Jerrum, Alistair Sinclair, and Eric Vigoda. A polynomial-time approximation algorithm for the permanent of a matrix with nonnegative entries. *J. ACM*, 51(4):671–697, 2004.

[32] Mark Jerrum, Leslie G. Valiant, and Vijay V. Vazirani.

Random generation of combinatorial structures from a uniform distribution. *Theor. Comput. Sci.*, 43:169–188, 1986.

[33] Daniel M. Kane, Shachar Lovett, and Shay Moran. Near-optimal linear decision trees for k-sum and related problems. *J. ACM*, 66(3):16:1–16:18, 2019.

[34] Ioannis Koutis. Constrained multilinear detection for faster functional motif discovery. *Inf. Process. Lett.*, 112(22):889–892, 2012.

[35] Vincent Lacroix, Cristina G. Fernandes, and Marie-France Sagot. Motif search in graphs: Application to metabolic networks. *IEEE/ACM Trans. Comput. Biol. Bioinformatics*, 3(4):360–368, October 2006.

[36] Dániel Marx. Can you beat treewidth? *Theory of Computing*, 6(1):85–112, 2010.

[37] Kitty Meeks. The challenges of unbounded treewidth in parameterised subgraph counting problems. *Discrete Applied Mathematics*, 198:170 – 194, 2016.

[38] Kitty Meeks. Randomised enumeration of small witnesses using a decision oracle. *Algorithmica*, 81(2):519–540, Feb 2019.

[39] Moritz Müller. Randomized approximations of parameterized counting problems. In *Parameterized and Exact Computation: Second International Workshop, IWPEC 2006, Zürich, Switzerland, September 13-15, 2006. Proceedings*, pages 50–59, 2006.

[40] Mihai Patrascu and Ryan Williams. On the possibility of faster SAT algorithms. In Moses Charikar, editor, *Proceedings of the Twenty-First Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2010, Austin, Texas, USA, January 17-19, 2010*, pages 1065–1075. SIAM, 2010.

[41] Leslie G. Valiant. The complexity of computing the permanent. *Theor. Comput. Sci.*, 8:189–201, 1979.

[42] Leslie G. Valiant and Vijay V. Vazirani. NP is as easy as detecting unique solutions. *Theor. Comput. Sci.*, 47:85–93, 1986.

[43] R. Ryan Williams. Faster all-pairs shortest paths via circuit complexity. *SIAM J. Comput.*, 47(5):1965–1985, 2018.

[44] Ryan Williams. Faster decision of first-order graph properties. In *Joint Meeting of the Twenty-Third EACSL Annual Conference on Computer Science Logic (CSL) and the Twenty-Ninth Annual ACM/IEEE Symposium on Logic in Computer Science (LICS), CSL-LICS '14, Vienna, Austria, July 14 - 18, 2014*, pages 80:1–80:6, 2014.

[45] Virginia Vassilevska Williams. Hardness of easy problems: Basing hardness on popular conjectures such as the strong exponential time hypothesis (invited talk). In Thore Husfeldt and Iyad A. Kanj, editors, *10th International Symposium on Parameterized and Exact Computation, IPEC 2015, September 16-18, 2015, Patras, Greece*, volume 43 of *LIPIcs*, pages 17–29. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, 2015.