A Fixed-Parameter Perspective on #BIS

Radu Curticapean[1] · Holger Dell[2] · Fedor Fomin[3] · Leslie Ann Goldberg[4] · John Lapinskas[4]

Abstract

The problem of (approximately) counting the independent sets of a bipartite graph (#BIS) is the canonical approximate counting problem that is complete in the intermediate complexity class #RHΠ_1. It is believed that #BIS does not have an efficient approximation algorithm but also that it is not NP-hard. We study the robustness of the intermediate complexity of #BIS by considering variants of the problem parameterised by the size of the independent set. We map the complexity landscape for three problems, with respect to exact computation and approximation and with respect to conventional and parameterised complexity. The three problems are counting independent sets of a given size, counting independent sets with a given number of vertices in one vertex class and counting maximum independent sets amongst those with a given number of vertices in one vertex class. Among other things, we show that all of these problems are NP-hard to approximate within any polynomial ratio. (This is surprising because the corresponding problems without the size parameter are complete in #RHΠ_1, and hence are not believed to be NP-hard.) We also show that the first problem is #W[1]-hard to solve exactly but admits an FPTRAS, whereas the other two are W[1]-hard to approximate even within any polynomial ratio. Finally, we show that, when restricted to graphs of bounded degree, all three problems have efficient exact fixed-parameter algorithms.

Keywords Approximate counting · Parameterised complexity · Independent sets

1 Introduction

The problem of (approximately) counting the independent sets of a bipartite graph, called #BIS, is one of the most important problems in the field of approximate counting. This problem is known to be complete in the intermediate complexity class #RHΠ_1 [8]. Many approximate counting problems are equivalent in difficulty to #BIS, including those that arise in spin-system problems [12,14] and in other domains. These problems are not believed to have efficient approximation algorithms, but they are also not believed to be NP-hard.

In this paper we study the robustness of the intermediate complexity of #BIS by considering relevant fixed parameters. It is already known that the complexity of #BIS is unchanged when the *degree* of the input graph is restricted (even if it is restricted to be at most 6) [2] but there is an efficient approximation algorithm when a stronger degree restriction (degree at most 5) is applied, even to the vertices in just one of the parts of the vertex partition of the bipartite graph [17].

We consider variants of the problem parameterised by the *size* of the independent set. We first show that all of the following problems are #P-hard to solve exactly and NP-hard to approximate *within any polynomial factor*.

- #Size-BIS: Given a bipartite graph G and a non-negative integer k, count the size-k independent sets of G.
- #Size-Left-BIS: Given a bipartite graph G with vertex partition (U, V) and a non-negative integer k, count the independent sets of G that have k vertices in U, and
- #Size-Left-Max-BIS: Given a bipartite graph G with vertex partition (U, V) and a non-negative integer k, count the maximum independent sets amongst all independent sets of G with k vertices in U.

The NP-hardness of these approximate counting problems is surprising given that the corresponding problems without the parameter k (that is, the problem #BIS and also the problem #Max-BIS, which is the problem of counting the *maximum* independent sets of a bipartite graph) are both complete in #RHΠ_1, and hence are not believed to be NP-hard. Therefore, it is the introduction of the parameter k that causes the hardness.

To gain a more refined perspective on these problems, we also study them from the perspective of parameterised complexity, taking the number of vertices, n, as the size of the input and k as the fixed parameter. Our results are summarised in Table 1, and stated in detail later in the paper. Rows 1 and 3 of the table correspond to the conventional (exact and approximate) setting that we have already discussed. Rows 2 and 4 correspond to the parameterised complexity setting, which we discuss next. As is apparent from the table, we have mapped the complexity landscape for the three problems in all four settings.

In parameterised complexity, the central goal is to determine whether computational problems have fixed-parameter tractable (FPT) algorithms, that is, algorithms that run in time $f(k) \cdot n^{O(1)}$ for some computable function f. Hardness results are presented using the W-hierarchy [10], and in particular using the complexity classes W[1] and W[2], which constitute the first two levels of the hierarchy. It is known (see [10]) that FPT $\subseteq$ W[1] $\subseteq$ W[2] and these classes are widely believed to be distinct from

Table 1 A summary of our results

	#Size-BIS	#Size-Left-BIS	#Size-Left-Max-BIS
Exact poly	#P-complete even in graphs of max-degree 3. (Theorem 1)	#P-complete even in graphs of max-degree 3. (Theorem 1)	#P-hard even in graphs of max-degree 3. (Theorem 2)
Exact FPT	#W[1]-complete. (Theorem 4)	#W[2]-hard. (Theorem 5)	W[1]-hard. (Theorem 6)
	FPT for bounded-degree graphs. (Theorem 14(i))	FPT for bounded-degree graphs. (Theorem 14(ii))	FPT for bounded-degree graphs. (Theorem 14(iii))
Approx poly	NP-hard to approximate within any polynomial factor. (Theorem 9)	NP-hard to approximate within any polynomial factor. (Theorem 7)	NP-hard to approximate within any polynomial factor. (Theorem 6)
Approx FPT	Has FPTRAS. (Theorem 11)	W[1]-hard to approximate within any polynomial factor. (Theorem 7)	W[1]-hard to approximate within any polynomial factor. (Theorem 6)

Each column corresponds to one of the three problems that we consider (#Size-BIS, #Size-Left-BIS and #Size-Left-Max-BIS), and each row corresponds to one of the four settings we consider (exact polynomial-time, exact FPT-time, approximate polynomial-time, and approximate FPT-time)

each other. It is also known [6, Chapter 14] that the Exponential Time Hypothesis (see [15]) implies FPT $\neq$ W[1]. Analogous classes #W[1] and #W[2] exist for counting problems [9].

As can be seen from the table, we prove that all of our problems are at least W[1]-hard to solve exactly, which indicates (subject to the complexity assumptions in the previous paragraph) that they cannot be solved by FPT algorithms. Moreover, #Size-Left-BIS and #Size-Left-Max-BIS are W[1]-hard to solve even approximately. It is known [19] that each parameterised counting problem in the class #W[i] has a randomised FPT approximation algorithm using a W[i] oracle, so W[i]-hardness is the appropriate hardness notion for parameterised approximate counting problems. By contrast, we show that #Size-BIS *can* be solved approximately in FPT time. In fact, it has an FPT randomized approximation scheme (FPTRAS).

Motivated by the fact that #BIS is known to be #P-complete to solve exactly even on graphs of degree 3 [24], we also consider the case where the input graph has bounded degree. While the conventional problems remain intractable in this setting (row one of the table), we prove that all three of our problems admit linear-time fixed-parameter algorithms for bounded-degree instances (row two). Note that Theorem 14(i) is also implicit in independent work by Patel and Regts [20].

2 Preliminaries

For a positive integer n, we let $[n]$ denote the set $\{1, \ldots, n\}$. We consider graphs G to be undirected. For a vertex set $X \subseteq V(G)$, denote by $G[X]$ the subgraph induced by X. For a vertex $v \in V(G)$, we write $\Gamma(v)$ for its open neighbourhood (that is, excluding v itself).

Given a graph G, we denote the size of a maximum independent set of G by $\mu(G)$. We denote the number of all independent sets of G by $\text{IS}(G)$, the number of size-k independent sets of G by $\text{IS}_k(G)$, and the number of size-$\mu(G)$ independent sets of G by $\text{MIS}(G)$. A bipartite graph G is presented as a triple (U, V, E) in which (U, V) is a partition of the vertices of G into *vertex classes*, and E is a subset of $U \times V$. If $G = (U, V, E)$ is a bipartite graph then an independent set S of G is said to be an "ℓ-left independent set of G" if $|S \cap U| = \ell$. The size of a *maximum ℓ-left independent set* of G is denoted by $\mu_{\ell\text{-left}}(G)$. An ℓ-left independent set of G is said to be "ℓ-left-maximum" if and only if it has size $\mu_{\ell\text{-left}}(G)$. Finally, $\text{IS}_{\ell\text{-left}}(G)$ denotes the number of ℓ-left independent sets of G and $\text{IS}_{\ell\text{-left-max}}(G)$ denotes the number of ℓ-left-maximum independent sets of G. Using these definitions, we now give formal definitions of #BIS and of the three problems that we study.

Name: #BIS.
Input: A bipartite graph G.
Output: $\text{IS}(G)$.

Name: #Size-BIS.
Input: A bipartite graph G and a non-negative integer k.
Output: $\text{IS}_k(G)$.
Parameter: k.

Name: #Size-Left-BIS.
Input: A bipartite graph G and a non-negative integer ℓ.
Output: $\text{IS}_{\ell\text{-left}}(G)$.
Parameter: ℓ.

Name: #Size-Left-Max-BIS.
Input: A bipartite graph G and a non-negative integer ℓ.
Output: $\text{IS}_{\ell\text{-left-max}}(G)$.
Parameter: ℓ.

For each of our computational problems, we add "[Δ]" to the end of the name of the problem to indicate that the input graph G has degree at most Δ. For example, #BIS[Δ] is the problem defined as follows.

Name: #BIS[Δ].
Input: A bipartite graph G with degree at most Δ.
Output: $\text{IS}(G)$.

When stating quantitative bounds on running times of algorithms, we assume the standard word-RAM machine model with logarithmic-sized words.

3 Exact Computation: Hardness Results

In this section, we prove the hardness results presented in the first two rows of Table 1.

3.1 Polynomial-Time Computation

We prove that all three problems are #P-hard, even when the input graphs are restricted to have degree at most 3.

Theorem 1 #Size-BIS[3] *and* #Size-Left-BIS[3] *are both* #P *-complete.*

Proof The problems #Size-BIS[3] and #Size-Left-BIS[3] are in #P, which can be deduced from their definitions. We show that the problems are #P-hard. Xia, Zhang and Zhao [24, Theorem 9] show that #BIS[3] is #P-hard, even under the additional restriction that the input graph is planar and 3-regular.

There is a straightforward reduction from #BIS[3] to #Size-BIS[3]. Suppose that G is an n-vertex input to #BIS[3]. Then $IS(G) = \sum_{k=0}^{n} IS_k(G)$. Using an oracle for #Size-BIS[3] (with the graph G and each $k \in \{0, \dots, n\}$) one can therefore compute $IS(G)$, as desired.

Similarly, there is a straightforward reduction from #BIS[3] to #Size-Left-BIS[3], using the fact that $IS(G) = \sum_{\ell=0}^{n} IS_{\ell\text{-left}}(G)$. Thus, the problems #Size-BIS[3] and #Size-Left-BIS[3] are both #P-hard.   □

Theorem 2 #Size-Left-Max-BIS[3] *is* #P-*hard.*

Proof Vadhan has shown [23, Corollary 4.2(1)] that #Max-BIS[3] is #P-complete. We now reduce #Max-BIS[3] to #Size-Left-Max-BIS[3]. Let $G = (U, V, E)$ be an instance of #Max-BIS[3] and let $s = |U|$. For each $j \in \{0, \dots, s\}$, let x_j be the number of size-$\mu(G)$ $(s - j)$-left independent sets of G. We wish to compute $MIS(G) = \sum_{j=0}^{s} x_j$, so it suffices to show how to compute the vector $(x_0, \dots, x_s)$ in polynomial time using an oracle for #Size-Left-Max-BIS[3]—this is what we do in the remainder of the proof.

For every non-negative integer i, let $G_i = (U_i, V_i, E_i)$ be the graph formed from G by adding a disjoint matching of size $s + i$. Note that $\mu(G_i) = \mu(G) + s + i$. Also, G_i has an s-left independent set of size $\mu(G_i)$ (to see this, consider any size-$\mu(G)$ independent set of G, say one that is a-left for some $a \in \{0, \dots, s\}$, and augment this with $s - a$ matching vertices from U_i and $i + a$ matching vertices from V_i). Let w_i be the number of size-$(\mu(G_i))$ s-left independent sets of G_i and note that $IS_{s\text{-left-max}}(G_i) = w_i$. Since G_i has degree at most 3, w_i can be computed using an oracle for #Size-Left-Max-BIS[3] (using the input graph G_i and setting the input ℓ equal to s).

From the definitions of x_j and w_i, we have

$$w_i = \sum_{j=0}^{s} x_j \binom{s+i}{j}. \tag{1}$$

Now let M be the matrix whose rows and columns are indexed by $\{0, \dots, s\}$ and whose entry $M_{i,j}$ is $\binom{s+i}{j}$. Let $\boldsymbol{w}$ be the transpose of the row vector $(w_0, \dots, w_s)$ and $\boldsymbol{x}$ be the transpose of the row vector $(x_0, \dots, x_s)$. Then Equation (1) can be re-written as $\boldsymbol{w} = M\boldsymbol{x}$.

Now [13, Corollary 2] shows that M is invertible (taking $k = s + 1$, $a_i = s + i - 1$ and $b_j = j - 1$ for $1 \leq i, j \leq k$, in the language of the corollary), so the vector $\boldsymbol{x}$ can be computed as $\boldsymbol{x} = M^{-1}\boldsymbol{w}$. Since it suffices to compute $\boldsymbol{x}$, and the vector $\boldsymbol{w}$ can be computed using the oracle, this completes the reduction. □

3.2 Fixed-Parameter Intractability

We first define the parameterised complexity classes relevant in this paper, namely, the class W[1] of decision problems, and the counting classes #W[1] and #W[2]. For simplicity, we do so in terms of complete problems and reductions. The following definitions are taken from Flum and Grohe [10].

Definition 3 Let F and G be parameterised problems. For any instance x of F, write $k(x)$ for the parameter of F and $|x|$ for the size of x. For any instance y of G, write $\ell(y)$ for the parameter of y. An *FPT Turing reduction* from F to G is an algorithm with an oracle for G that, for some computable functions $f, g : \mathbb{N} \to \mathbb{N}$ and for some constant $c \in \mathbb{N}$, solves any instance x of F in time at most $f(k(x)) \cdot |x|^c$ in such a way that for all oracle queries the instances y of G satisfy $\ell(y) \leq g(k(x))$.

Now, write $\mathcal{F}$ for the set of all instances of F, and for all $x \in \mathcal{F}$ write $F(x)$ for the desired output given input x. Likewise, write $\mathcal{G}$ for the set of all instances of G, and for all $y \in \mathcal{G}$ write $G(y)$ for the desired output given input y. Suppose $R : \mathcal{F} \to \mathcal{G}$ is a function satisfying the following properties: for all $x \in \mathcal{F}$, $F(x) = G(R(x))$; there exists a computable function $f : \mathbb{N} \to \mathbb{N}$ and a constant $c \in \mathbb{N}$ such that for all $x \in \mathcal{F}$, $R(x)$ is computable in time at most $f(k(x)) \cdot |x|^c$; there exists a computable function $g : \mathbb{N} \to \mathbb{N}$ such that for all $x \in \mathcal{F}$, $\ell(R(x)) \leq g(k(x))$. If F and G are decision problems, we call R an *FPT many-one reduction* from F to G; if F and G are counting problems, we call R an *FPT parsimonious reduction* from F to G.

We define W[1] in terms of the following problem.

Name: Size-Clique.
Input: A graph G and a positive integer k.
Output: True if G contains a k-clique, false otherwise.
Parameter: k.

W[1] is the set of all parameterised decision problems with an FPT many-one reduction to Size-Clique. We say a problem F is W[1]-hard if there is an FPT Turing reduction from Size-Clique to F. For a proof that this is equivalent to the standard definition of W[1], see e.g., Downey and Fellows [7, Theorem 21.3.4].

We define #W[1] in terms of the following problem.

Name: #Size-Clique.
Input: A graph G and a positive integer k.
Output: The number of k-cliques in G.
Parameter: k.

#W[1] is the set of all parameterised counting problems with an FPT parsimonious reduction to #Size-Clique. We say a problem F is #W[1]-*hard* if there is an FPT Turing reduction from #Size-Clique to F. For a proof that this is equivalent to the standard definition of #W[1], see e.g., [10, Theorem 14.18].

Recall that a set $D \subseteq V(G)$ is called a *dominating set* of a graph G if every vertex $v \in V(G)$ is either contained in D or adjacent to a vertex of D. We define #W[2] in terms of the following problem.

Name: #Size-Dominating-Set.
Input: A graph $G = (U, E)$ and a positive integer k.
Output: The number of dominating sets of G of size k.
Parameter: k.

#W[2] is the set of all parameterised counting problems with an FPT parsimonious reduction to #Size-Dominating-Set. We say a problem F is #W[2]-hard if there is an FPT Turing reduction from #Size-Dominating-Set to F. For a proof that this is equivalent to the standard definition of #W[2], see [9, Theorem 19]).

In order to prove our exact fixed-parameter hardness results, we consider the following problem.

Name: #Size-Partitioned-Biclique.
Input: An integer t, a $2t$-coloured graph $\mathcal{G}$, and a $2t$-coloured balanced biclique $\mathcal{H}$ on $2t$ vertices (i.e. a $2t$-coloured copy of $K_{t,t}$) in which every colour appears exactly once.
Output: The number $\mathrm{Sub}_\mathcal{G}(\mathcal{H})$ of subgraphs $\mathcal{K} \subseteq \mathcal{G}$ with $\mathcal{K} \simeq \mathcal{H}$.
Parameter: t.

Theorem 4 #Size-BIS *is* #W[1] *-complete.*

Proof We will prove first easiness, then hardness.

#Size-BIS is in #W[1]: We give an FPT parsimonious reduction to #Size-Clique. Indeed, given an instance (G, k) of #Size-BIS with $G = (U, V, E)$, let $V' = U \cup V$, $E' = \{\{u, v\} \mid u, v \in V', (u, v) \notin E\}$, and $G' = (V', E')$. Then the size-k independent sets of G correspond exactly to the size-k cliques of G', as required.

#Size-BIS is #W[1]-hard: We give an FPT Turing reduction from #Size-Partitioned-Biclique. Note that the class $\{K_{t,t} : t \geq 1\}$ of all balanced bicliques is recursively enumerable and contains graphs of arbitrarily high treewidths, so #Size-Partitioned-Biclique is #W[1]-hard by a result of Curticapean and Marx [5, Theorem II.8].

Let $(t, \mathcal{G}, \mathcal{H})$ be an instance of #Size-Partitioned-Biclique. Write $\mathcal{G} = ((V, E), c)$. Without loss of generality, suppose the colours $\{1, \ldots, t\}$ appear in one vertex class of $\mathcal{H}$ and the colours $\{t + 1, \ldots, 2t\}$ appear in the other. Let

$$E' = \{\{u, v\} \mid u, v \in V,\ c(u) \in [t],\ c(v) \in [2t] \setminus [t],\ \{u, v\} \notin E\}.$$

Define a coloured graph $\mathcal{G}' = ((V, E'), c)$. Then each copy of $\mathcal{H}$ in $\mathcal{G}$ spans an independent set in $\mathcal{G}'$ in which every colour appears exactly once and vice versa, so $\mathrm{Sub}_\mathcal{G}(\mathcal{H})$ is precisely the number of such independent sets in $\mathcal{G}'$.

For any set $S \subseteq [2t]$, let $\mathcal{I}_S$ be the set of size-$2t$ independent sets in $\mathcal{G}'$ which contain no vertices with colours in S. By the inclusion-exclusion principle,

$$\text{Sub}_G(\mathcal{H}) = \left| \mathcal{I}_\emptyset \setminus \bigcup_{i=1}^{2t} \mathcal{I}_{\{i\}} \right| = |\mathcal{I}_\emptyset| - \sum_{\emptyset \neq S \subseteq [2t]} (-1)^{|S|-1} |\mathcal{I}_S|. \tag{2}$$

Moreover, for any set $S \subseteq [2t]$, let G_S be the bipartite graph (U_S, V_S, E_S) defined by

$$U_S = \{v \in V \mid c(v) \in [t] \setminus S\},$$
$$V_S = \{v \in V \mid c(v) \in [2t] \setminus ([t] \cup S)\},$$
$$E_S = \{(u, v) \mid u \in U_S, v \in V_S, \{u, v\} \in E'\}.$$

Then $\mathcal{I}_S$ is precisely the set of size-$2t$ independent sets in G_S. Our algorithm therefore determines each $|\mathcal{I}_S|$ by calling a #Size-BIS oracle with input $(G_S, 2t)$, then uses (2) to compute $\text{Sub}_G(\mathcal{H})$. $\qquad \square$

Next, we turn to the exact parameterised complexity of #Size-Left-BIS. The hardness result we obtain for this problem is a bit stronger than for #Size-BIS: we prove that it is #W[2]-hard.

Theorem 5 #Size-Left-BIS *is #W[2]-hard.*

Proof We reduce from the dominating set problem. Let $G = (U, E)$ and k be given as input for #Size-Dominating-Set where $U = \{u_1, \ldots, u_n\}$. The reduction computes the bipartite split graph of G; formally, let $V = \{v_1, \ldots, v_n\}$, let $E' = \{(u_a, v_b) \mid a = b \text{ or } \{u_a, u_b\} \in E\}$, and let $G' = (U, V, E')$.

For non-negative integers ℓ and r, we define an (ℓ, r)-*set of* G' to be a size-ℓ subset X of U that has exactly r neighbours in V. Let $Z_{\ell,r}$ be the number of (ℓ, r)-sets of G'. Note that a size-k subset X of U is a dominating set of G if and only if it is a (k, n)-set of G', so there are precisely $Z_{k,n}$ size-k dominating sets of G.

The algorithm applies polynomial interpolation to determine $Z_{k,r}$ for all $r \in \{0, \ldots, n\}$. We use a special case of the cloning construction from the proof of Theorem 4. For every positive integer i, let $V_i = V \times [i]$, let $E_i' = \{(u, (v, b)) \in U \times V_i \mid (u, v) \in E'\}$, and let $G_i' = (U, V_i, E_i')$. For each (k, r)-set X of G', there are exactly $2^{i(n-r)}$ k-left independent sets S of G_i' with $S \cap U = X$. Thus for all $i \in [n+1]$,

$$\text{IS}_{k\text{-left}}(G_i') = \sum_{r=0}^{n} 2^{i(n-r)} Z_{k,r}. \tag{3}$$

Let M be the $(n+1) \times (n+1)$ matrix whose rows are indexed by $[n+1]$ and columns are indexed by $\{0, \ldots, n\}$ such that $M_{i,r} = 2^{i(n-r)}$ holds. Then (3) can be viewed as a linear equation system $\boldsymbol{w} = M\boldsymbol{z}$, where $\boldsymbol{w} = (\text{IS}_{k\text{-left}}(G_1'), \ldots, \text{IS}_{k\text{-left}}(G_{n+1}'))^T$ and $\boldsymbol{z} = (Z_{k,0}, \ldots, Z_{k,n})^T$. The oracle for #Size-Left-BIS can be used to compute $\boldsymbol{w}$, and M is invertible since it is a (transposed) Vandermonde matrix. Thus the reduction can compute $\boldsymbol{z}$, and in particular $Z_{k,n}$, as required. $\qquad \square$

We defer the proof of the W[1]-hardness of #Size-Left-Max-BIS to the next section, as it is implied by the corresponding approximation hardness result.

4 Approximate Computation: Hardness Results

In this section, we prove the hardness results in rows 3 and 4 of Table 1. Note that the reductions from Sect. 3 cannot be used here, since #BIS is not known to be NP-hard to approximate. In order to state our hardness results formally, we introduce approximation versions of the problems that we consider.

Name: Deg-c-#ApxSizeLeftMaxBIS.
Input: A bipartite graph G on n vertices and a non-negative integer ℓ.
Output: A number z such that $n^{-c} \cdot \mathrm{IS}_{\ell\text{-left-max}}(G) \leq z \leq n^c \cdot \mathrm{IS}_{\ell\text{-left-max}}(G)$.
Parameter: ℓ.

Name: Deg-c-#ApxSizeLeftBIS.
Input: A bipartite graph G on n vertices and a non-negative integer ℓ.
Output: A number z such that $n^{-c} \cdot \mathrm{IS}_{\ell\text{-left}}(G) \leq z \leq n^c \cdot \mathrm{IS}_{\ell\text{-left}}(G)$.
Parameter: ℓ.

Name: Deg-c-#ApxSizeBIS.
Input: A bipartite graph G on n vertices and a non-negative integer k.
Output: A number z such that $n^{-c} \cdot \mathrm{IS}_k(G) \leq z \leq n^c \cdot \mathrm{IS}_k(G)$.
Parameter: k.

We first prove the results in the last column of Table 1 and establish the others by reduction.

Theorem 6 *For all $c \geq 0$, Deg-c-#ApxSizeLeftMaxBIS is both* NP-*hard and* W[1]-*hard.*

Proof Let c be any non-negative integer. We will give a reduction from Size-Clique to Deg-c-#ApxSizeLeftMaxBIS which is both an FPT Turing reduction and a polynomial-time Turing reduction. The theorem then follows from the fact that Size-Clique is both NP-hard [22, Theorem 7.32] and W[1]-hard [7, Theorem 21.3.4].

Let (G, k) be an instance of Size-Clique with $G = (V, E)$ and $n = |V|$. We use a standard powering construction to produce an intermediate instance (G', k) of Size-Clique with $G' = (V', E')$. More precisely, let $t = n^{2c}$, let C be a set of k new vertices, and let $V' = (V \times [t]) \cup C$. We define E' such that

$$E' = \big\{ \{(u, i), (v, j)\} \mid \{u, v\} \in E, i, j \in [t] \big\} \cup \big\{ \{u, v\} \mid u, v \in C, u \neq v \big\}.$$

From (G', k), we construct an instance (G'', ℓ) of Deg-c-#ApxSizeLeftMaxBIS with $G'' = (U, V', E'')$ and $\ell = \binom{k}{2}$. For this, let $U = \{u_e \mid e \in E'\}$ be a set of vertices and let $E'' = \{(u_e, v) \mid e \in E', v \in e\}$. The reduction queries the oracle for

(G'', ℓ), which yields an approximate value z for the number $\mathrm{IS}_{\ell\text{-left-max}}(G'')$. If $z \leq n^c$, the reduction returns 'no', there is no k-clique in G, and otherwise it returns 'yes'. It is obvious that the reduction runs in polynomial time.

It remains to prove the correctness of the reduction. Let $\mathrm{CL}_k(G)$ be the number of k-cliques in G. The ℓ-left-maximum independent sets X of G'' correspond bijectively to the size-ℓ edge sets $\{e \mid u_e \in X \cap U\}$ of G' which span a minimum number of vertices. Note that any set of $\ell = \binom{k}{2}$ edges must span at least k vertices, with equality only in the case of a k-clique. Since G' contains at least one k-clique (induced by C), we have $\mathrm{IS}_{\ell\text{-left-max}}(G'') = \mathrm{CL}_k(G')$. Moreover, each k-clique X in G corresponds to a size-t^k family of k-cliques in G'. Each k-clique in the family consists of exactly one vertex from each set $\{x\} \times [t]$ such that $x \in V(X)$. This accounts for all k-cliques in G' except $G'[C]$. Thus

$$\mathrm{IS}_{\ell\text{-left-max}}(G'') = \mathrm{CL}_k(G') = t^k \mathrm{CL}_k(G) + 1. \tag{4}$$

Let z be the result of applying our oracle to (G'', ℓ). If G contains no k-cliques, then by (4) we have $z \leq n^c \cdot \mathrm{IS}_{\ell\text{-left-max}}(G'') = n^c$ and the reduction returns 'no'. Otherwise, we have $z \geq n^{-c} \cdot \mathrm{IS}_{\ell\text{-left-max}}(G'') \geq n^{-c}(t^k + 1) > n^c$ and the reduction returns 'yes'. Thus the reduction is correct and the theorem follows. $\qquad\square$

Theorem 7 *For all $c \geq 0$, Deg-c-#ApxSizeLeftBIS is both* NP *-hard and* W[1] *-hard.*

Proof Let c be any non-negative integer. We will give a reduction from the problem Deg-$(c + 1)$-#ApxSizeLeftMaxBIS to the problem Deg-c-#ApxSizeLeftBIS which is both an FPT Turing reduction and a polynomial-time Turing reduction. The result then follows by Theorem 6.

Let (G, ℓ) be an instance of Deg-c-#ApxSizeLeftMaxBIS. Write $G = (U, V, E)$, let $n = |V(G)|$, and let $t = 6n$. Without loss of generality, suppose $n \geq 5$ and that n is sufficiently large that $n^c 2^{-n} \leq 1$. Let $V' = V \times [t]$, let $E' = \{(u, (v, i)) \mid (u, v) \in E, i \in [t]\}$, and let $G' = (U, V', E')$. Let $\mu = \mu_{\ell\text{-left}}(G)$, and let z be the result of applying our oracle to (G', ℓ).

For any non-negative integers i and j, we define $\mathrm{IS}_{i,j}(G)$ to be the number of independent sets $X \subseteq V(G)$ with $|X \cap U| = i$ and $|X \cap V| = j$. Each ℓ-left independent set X of G corresponds to the family of ℓ-left independent sets of G' consisting of $X \cap U$ together with at least one vertex from each set $\{x\} \times [t]$ such that $x \in X \cap V$. Thus by the definition of μ,

$$\mathrm{IS}_{\ell\text{-left}}(G') = \sum_{r=0}^{\mu-\ell} \mathrm{IS}_{\ell,r}(G)(2^t - 1)^r. \tag{5}$$

Since G contains at most 2^n independent sets and $\mathrm{IS}_{\ell,\mu-\ell}(G) \geq 1$, we have

$$(2^t - 1)^{\mu-\ell} \leq \mathrm{IS}_{\ell\text{-left}}(G') \leq 2^n (2^t - 1)^{\mu-\ell}.$$

Since $n^c \leq 2^n \leq (2^t - 1)^{1/5}$, it follows that $(2^t - 1)^{\mu-\ell-1/5} \leq z \leq (2^t - 1)^{\mu-\ell+2/5}$, and hence the algorithm can obtain μ by rounding $\ell + \lg(z)/\lg(2^t - 1)$ to the nearest integer. Moreover, by (5) we have

$$\mathrm{IS}_{\ell\text{-left}}(G') \leq \mathrm{IS}_{\ell,\, \mu-\ell}(G)(2^t - 1)^{\mu-\ell} + 2^n (2^t - 1)^{\mu-\ell-1} \leq 2\mathrm{IS}_{\ell,\, \mu-\ell}(G)(2^t - 1)^{\mu-\ell}.$$

It follows that $\mathrm{IS}_{\ell,\, \mu-\ell}(G) \leq \mathrm{IS}_{\ell\text{-left}}(G')/(2^t - 1)^{\mu-\ell} \leq 2\mathrm{IS}_{\ell,\, \mu-\ell}(G)$, and hence that

$$n^{-c-1}\mathrm{IS}_{\ell,\, \mu-\ell}(G) \leq z/(2^t - 1)^{\mu-\ell} \leq n^{c+1}\mathrm{IS}_{\ell,\, \mu-\ell}(G).$$

The algorithm therefore outputs $z/(2^t - 1)^{\mu-\ell}$.   □

The following well-known Chernoff bound appears in e.g., Janson et al. [16, Corollary 2.3].

Lemma 8 *If $X \sim \mathrm{Bin}\,(n,\, p)$ is a binomial variable and $0 < \varepsilon \leq 3/2$, then*

$$\mathbb{P}(|X - \mathbb{E}(X)| \geq \varepsilon\mathbb{E}(X)) \leq 2e^{-\varepsilon^2\mathbb{E}(X)/3}.$$

□

Theorem 9 *For all $c \geq 0$,* Deg-c-#ApxSizeBIS *is NP-hard.*

Proof For all $c \geq 0$, we give a polynomial-time Turing reduction from the problem Deg-$(c + 1)$-#ApxSizeLeftBIS to the problem Deg-c-#ApxSizeBIS. The result then follows from Theorem 7.

Fix $c \geq 0$ and let (G, ℓ) be an instance of Deg-$(c + 1)$-#ApxSizeLeftBIS. Suppose that $G = (U, V, E)$ where $U = \{u_1, \ldots, u_p\}$. Note from the problem definition that $n = |U \cup V|$ and suppose without loss of generality that $\ell \in [p]$ and that $n \geq 40$ (otherwise, (G, ℓ) is an easy instance of Deg-$(c + 1)$-#ApxSizeLeftBIS, so the answer can be computed, even without using the oracle).

Let $s = 2n^6$ and $t = \lfloor s \log_2 3 \rfloor - s$. For each $i \in [p]$, let U_i, V_i and U_i' be disjoint sets of vertices with $|U_i'| = |V_i| = s$ and $|U_i| = t$. Write $U_i' = \{u_{i,1}, \ldots, u_{i,s}\}$ and $V_i = \{v_{i,1}, \ldots, v_{i,s}\}$. Then let $U' = \bigcup_{i\in[p]}(U_i \cup U_i')$, $V' = \bigcup_{i\in[p]} V_i \cup V$, and

$$E' = \bigcup_{i\in[p]} \Big((U_i \times V_i) \cup \{(u_{i,j}, v_{i,j}) \mid j \in [s]\}\Big) \cup \bigcup_{(u_j,v)\in E(G)} (U_j \times \{v\}).$$

Let $G' = (U', V', E')$, as depicted in Fig. 1.

Intuitively, the proof will proceed as follows. We will map independent sets X' of G' to independent sets X of G by taking $X \cap V = X' \cap V$ and adding each $u_i \in U$ to X if and only if $U_i \cap X' \neq \emptyset$. We will show that roughly half the independent
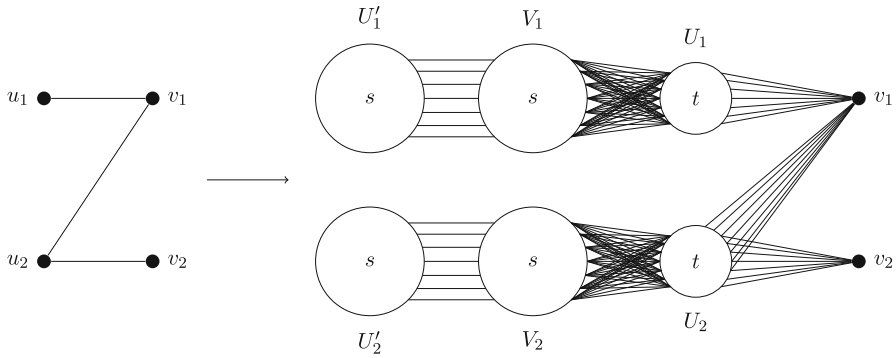
Fig. 1 An example of the reduction from Deg-$(c + 1)$-#ApxSizeLeftBIS to Deg-c-#ApxSizeLeftBIS used in the proof of Theorem 9 when $G = P_3$. Each vertex $u_i \in U$ is replaced by three vertex sets U_i', V_i and U_i in the resulting graph G'. Note that G' does not depend on the input parameter ℓ

sets of each gadget $U_i' \cup V_i \cup U_i$ have this form. We will also show that within each gadget, almost all independent sets with vertices in U_i have size roughly $(s + t)/2$, and almost all others have size roughly $2s/3$. Thus the independent sets in G with ℓ vertices in U roughly correspond to the independent sets in G' of size roughly $\ell \cdot (s + t)/2 + (p - \ell) \cdot 2s/3$, which we count using a #Size-BIS oracle.

We start by defining disjoint sets of independent sets of G'. For $x \in \{0, \ldots, p\}$, let $E(x) = \frac{2s}{3}(p - x) + \frac{s+t}{2}x$ and let

$$\mathcal{A}_x = \left\{ X' \subseteq V(G') \mid X' \text{ is an independent set of } G' \text{ and } \left| |X'| - E(x) \right| \leq \frac{s}{20} + n \right\}.$$

Note that since $n \geq 3$, we have $t > 17s/30$ and $120n \leq s$. Thus, if $x' > x$,

$$E(x') - E(x) = \left(\frac{t}{2} - \frac{s}{6} \right)(x' - x) > \left(\frac{17}{60} - \frac{1}{6} \right)s = \frac{s}{10} + \frac{s}{60} \geq \frac{s}{10} + 2n.$$

We conclude that the sets $\mathcal{A}_0, \ldots, \mathcal{A}_p$ are disjoint.

Next, we connect the independent sets of G' with those of G. Each independent set X' of G' projects onto the independent set $(X' \cap V) \cup \{u_i \mid X' \cap U_i \neq \emptyset\}$ of G. Given an independent set X of G, let $\varphi(X)$ be the set of independent sets X' of G' which project onto X. If $u_i \in X$, then there are $2^t - 1$ possibilities for $X' \cap U_i$ and 2^s possibilities for $X' \cap U_i'$, but $X' \cap V_i$ is empty. If $u_i \notin X$, then $X' \cap U_i$ is empty and there are 3^s possibilities for $X' \cap (U_i' \cup V_i)$. For $x \in \{0, \ldots, p\}$, let $F(x) = (2^{s+t} - 2^s)^x \cdot 3^{(p-x)s}$. It follows that, for any x-left independent set X of G, $|\varphi(X)| = F(x)$, which establishes the first of the following claims.

Claim 1. For any ℓ-left independent set X of G, $|\varphi(X) \cap \mathcal{A}_\ell| \leq F(\ell)$.
Claim 2. For any ℓ-left independent set X of G, $|\varphi(X) \cap \mathcal{A}_\ell| \geq F(\ell)/2$.
Claim 3. For any $x \in \{0, \ldots, p\} \setminus \{\ell\}$ and any x-left independent set X of G, $|\varphi(X) \cap \mathcal{A}_\ell| \leq F(\ell)/2^n$.

The proofs of Claims 2 and 3 are mere calculation, so before proving them we use the claims to complete the proof of the lemma. Recall that (G, ℓ) is an instance of Deg-$(c + 1)$-#ApxSizeLeftBIS with $\ell \in [p]$ and $n \geq 2$. Together, the claims imply

$$\frac{F(\ell)}{2}\mathrm{IS}_{\ell\text{-left}}(G) \leq |\mathcal{A}_\ell| \leq F(\ell)\mathrm{IS}_{\ell\text{-left}}(G) + F(\ell), \tag{6}$$

where the final $F(\ell)$ comes from the contribution to $|\mathcal{A}_\ell|$ corresponding to the (at most 2^n) independent sets of G that are not ℓ-left independent sets. Since $\ell \in [p]$, the quantity $\mathrm{IS}_{\ell\text{-left}}(G)$ is at least 1, which means that the right-hand side of (6) is at most $2F(\ell)\mathrm{IS}_{\ell\text{-left}}(G)$. Also, $F(\ell) > 0$. Thus, (6) implies

$$\frac{\mathrm{IS}_{\ell\text{-left}}(G)}{2} \leq \frac{|\mathcal{A}_\ell|}{F(\ell)} \leq 2\mathrm{IS}_{\ell\text{-left}}(G).$$

The oracle for Deg-c-#ApxSizeBIS can be used to compute a number z such that $n^{-c}|\mathcal{A}_\ell| \leq z \leq n^c|\mathcal{A}_\ell|$. (To do this, just call the oracle repeatedly with input G' and with every non-negative integer k such that $|k - E(\ell)| \leq \frac{s}{20} + n$, adding the results.) Thus,

$$n^{-c}\frac{\mathrm{IS}_{\ell\text{-left}}(G)}{2} \leq n^{-c}\frac{|\mathcal{A}_\ell|}{F(\ell)} \leq \frac{z}{F(\ell)} \leq n^c\frac{|\mathcal{A}_\ell|}{F(\ell)} \leq 2n^c\mathrm{IS}_{\ell\text{-left}}(G),$$

so the desired approximation of $\mathrm{IS}_{\ell\text{-left}}(G)$ can be achieved by dividing z by $F(\ell)$. We now complete the proof by proving Claims 2 and 3.

Proof of Claim 2:   Consider any $x \in \{0, \ldots, p\}$ and let X be an x-left independent set of G. We will show $|\varphi(X) \cap \mathcal{A}_x| \geq F(x)/2$, which implies the claim by taking $\ell = x$. In fact, we will establish the much stronger inequality

$$|\varphi(X) \cap \mathcal{A}_x| \geq (1 - 3ne^{-n^2})F(x), \tag{7}$$

which will also be useful in the proof of Claim 3. To establish Equation (7) we will show that the probability that a random element Y of $\varphi(X)$ satisfies $\left||Y| - E(x)\right| \leq \frac{s}{20} + n$ is at least $1 - 3ne^{-n^2}$.

So let Y be a uniformly random element of $\varphi(X)$. We will show that, with probability at least $1 - 3ne^{-n^2}$, the following bullet points hold.

- For all $i \in [p]$ with $u_i \notin X$, we have $\left||Y \cap (U_i \cup V_i \cup U_i')| - \frac{2s}{3}\right| \leq \frac{s}{n^2}$, and
- for all $i \in [p]$ with $u_i \in X$, we have $\left||Y \cap (U_i \cup V_i \cup U_i')| - \frac{s+t}{2}\right| \leq \frac{s+t}{n^2}$,

Since $n \geq 40$, we have $(p-x)s/n^2 + x(s+t)/n^2 \leq 2ps/n^2 \leq s/20$ and $|Y \cap V| \leq n$, so the claim follows. To obtain the desired failure probability, we will show that, for any $i \in [p]$, the probability that the relevant bullet point fails to hold is at most $3e^{-n^2}$ (so the total failure probability is at most $3ne^{-n^2}$, by a union bound).

First, consider any $i \in [p]$ with $u_i \notin X$. In this case, $Y \cap (U_i \cup V_i \cup U_i')$ is generated by including (independently for each $j \in [s]$) one of three possibilities: (i) $u_{i,j}$ but not $v_{i,j}$, (ii) $v_{i,j}$ but not $u_{i,j}$, or (iii) neither $u_{i,j}$ nor $v_{i,j}$. Each of the three choices is

equally likely. Thus $|Y \cap (U_i \cup V_i \cup U_i')|$ is distributed binomially with mean $2s/3$, so by a Chernoff bound (Lemma 8), the probability that the first bullet point fails for i is at most $2e^{-s/2n^4} < 3e^{-n^2}$, as desired.

Second, consider any $i \in [p]$ with $u_i \in X$. In this case, $Y \cap (U_i \cup V_i \cup U_i')$ is chosen uniformly from all subsets of $U_i \cup U_i'$ that contain at least one element of U_i. The total variation distance between the uniform distribution on these subsets and the uniform distribution on all subsets of $U_i \cup U_i'$ is at most 2^{-t}. Also, by a Chernoff bound (Lemma 8), the probability that a uniformly-random subset of $U_i \cup U_i'$ has a size that differs from its mean, $(s + t)/2$, by at least $(s + t)/n^2$ is at most $2e^{-2(s+t)/(3n^4)}$. Thus, the probability that the second bullet point fails for i is at most $2^{-t} + 2e^{-2(s+t)/(3n^4)} \leq 3e^{-n^2}$, as desired.

Proof of Claim 3: Suppose that $x \in \{0, \ldots, p\} \setminus \{\ell\}$ and that X is an x-left independent set of G. We know from Equation (7) that $|\varphi(X) \cap \mathcal{A}_\ell| \leq 3ne^{-n^2} F(x)$. We wish to show that this is at most $F(\ell)/2^n$. Note that $t \geq 1$ and $3^{s-1} \leq 2^{s+t} \leq 3^s$, so for all $y \in \{0, \ldots, p\}$,

$$F(y) = (2^{s+t} - 2^s)^y \cdot 3^{ps-ys} \leq 2^{y(s+t)} \cdot 3^{ps-ys} \leq 3^{ps}, \text{ and}$$
$$F(y) \geq 2^{y(s+t)-y} \cdot 3^{ps-ys} \geq 3^{ps-2y} \geq 3^{ps-2n}.$$

The claim follows from $F(x) \leq 3^{ps} \leq 3^{2n} F(\ell)$ and from the fact that $n \geq 40$. □

5 Algorithms

In this final section, we give our algorithmic results: An FPT randomized approximation scheme (FPTRAS) for #Size-BIS, and an exact FPT-algorithm for all three problems in bounded-degree graphs. We define an FPTRAS of #Size-BIS as in Arvind and Raman [1].

Definition 10 An *FPTRAS* for #Size-BIS is a randomised algorithm that takes as input a bipartite graph G, a non-negative integer k, and a real number $\varepsilon \in (0, 1)$ and outputs a real number z. With probability at least $2/3$, the output z must satisfy $(1 - \varepsilon)\mathrm{IS}_k(G) \leq z \leq (1 + \varepsilon)\mathrm{IS}_k(G)$. Furthermore, there is a function $f : \mathbb{R} \to \mathbb{R}$ and a polynomial p such that the running time of the algorithm is at most $f(k)\, p(|V(G)|, 1/\varepsilon)$.

Theorem 11 *There is an FPTRAS for* #Size-BIS *with time complexity* $O\left(2^k \cdot k^2/\varepsilon^2\right)$ *for input graphs with n vertices and m edges.*

Proof Let (G, k) be an instance of #Size-BIS with $G = (U, V, E)$ and $n = |V(G)|$. Let $\varepsilon > 0$ be the other input of the FPTRAS. Let $t = 10\lceil 2^k/\varepsilon^2 \rceil$. The FPTRAS independently samples t uniformly-random size-k subsets of $U \cup V$. Let X be the number of independent sets among the samples. The output z of the FPTRAS is $z = X \cdot \binom{n}{k}/t$.

Note that $\mathbb{E}(X) = t \cdot \mathrm{IS}_k(G)/\binom{n}{k}$. We now show that with probability at least $2/3$,

$$(1 - \varepsilon)\mathrm{IS}_k(G) \leq z \leq (1 + \varepsilon)\mathrm{IS}_k(G).$$

Since each sample lies entirely within U or entirely within V with probability at least 2^{-k}, we have $\mathbb{E}(X) \geq t2^{-k} \geq 10/\varepsilon^2$. By Lemma 8, we have

$$\mathbb{P}\Big(|X - \mathbb{E}(X)| \geq \varepsilon\mathbb{E}(X)\Big) \leq 2e^{-10/3} < 1/3.$$

Thus, with probability at least $2/3$, we have $|X - \mathbb{E}(X)| \leq \varepsilon\mathbb{E}(X)$, and so $|z - \mathrm{IS}_k(G)| \leq \varepsilon\mathrm{IS}_k(G)$ holds as required.

Recall that we use the word-RAM model, in which operations on $O(\log n)$-sized words take $O(1)$ time. Thus for each of the t samples, the algorithm generates the sample in $O(k)$ time and makes $\binom{k}{2}$ queries to the graph to check that the selected set is an independent set. The running time is therefore as claimed.   □

We now turn to our algorithms for bounded-degree graphs. We require the following definitions. For any positive integer s, an *s-coloured graph* is a tuple (G, c) where G is a graph and $c : V(G) \to [s]$ is a map. Suppose $\mathcal{G} = (G, c)$ and $\mathcal{G}' = (G', c')$ are coloured graphs with $G = (V, E)$ and $G' = (V', E')$.

We say a map $\phi : V \to V'$ is a *homomorphism* from $\mathcal{G}$ to $\mathcal{G}'$ if ϕ is a homomorphism from G to G' and, for all $v \in V$, $c(v) = c'(\phi(v))$. If ϕ is also bijective, we say ϕ is an *isomorphism* from $\mathcal{G}$ to $\mathcal{G}'$, that $\mathcal{G}$ and $\mathcal{G}'$ are *isomorphic*, and write $\mathcal{G} \simeq \mathcal{G}'$. For all $X \subseteq V$, we define $\mathcal{G}[X] = (G[X], c|_X)$, and say $\mathcal{G}[X]$ is an *induced subgraph* of $\mathcal{G}$. Given coloured graphs $\mathcal{H}$ and $\mathcal{G}$, we denote the number of sets $X \subseteq V(\mathcal{G})$ with $\mathcal{G}[X] \simeq \mathcal{H}$ by $\#\,\mathrm{Ind}(\mathcal{H} \to \mathcal{G})$. Finally, we define $V(\mathcal{G}) = V$ and $E(\mathcal{G}) = E$ and we define $\Delta(\mathcal{G})$ to be the maximum degree of G.

For each positive integer Δ, we consider a counting version of the induced subgraph isomorphism problem for coloured graphs of degree at most Δ.

Name: #Induced-Coloured-Subgraph[Δ].
Input: Two coloured graphs $\mathcal{H}$ and $\mathcal{G}$, each with maximum degree bounded by Δ.
Output: $\#\,\mathrm{Ind}(\mathcal{H} \to \mathcal{G})$.
Parameter: $|V(\mathcal{H})|$.

We will later reduce our bipartite independent set counting problems to the coloured induced subgraph problem. Note that #Induced-Coloured-Subgraph[Δ] can be expressed as a first-order model-counting problem in bounded-degree structures. A well-known result of Frick [11, Theorem 6] would yield an algorithm for #Induced-Coloured-Subgraph[Δ] with running time $g(k) \cdot n$, where $k = |V(\mathcal{H})|$ and $n = |V(\mathcal{G})|$. (To our knowledge this fact has not appeared in the literature, but the proof is not hard.) However, the function g of Frick's algorithm may grow faster than any constant-height tower of exponentials. In the following, we provide an algorithm for #Induced-Coloured-Subgraph[Δ] that is substantially faster: It runs in time $O(nk^{(2\Delta+3)k})$.

The algorithm follows the strategy of [4] to count small subgraphs: Instead of counting (coloured) induced subgraphs, we can count (coloured) homomorphisms and recover the number of induced subgraphs via a simple basis transformation. Transforming to homomorphisms is useful because homomorphisms from small patterns to bounded-degree host graphs can be counted by a simple branching procedure–this

is however not true for small induced subgraphs. The following lemma encapsulates counting homomorphisms in graphs of bounded degree. Given coloured graphs $\mathcal{H}$ and $\mathcal{G}$, we denote the number of homomorphisms from $\mathcal{H}$ to $\mathcal{G}$ by $\#\mathrm{Hom}(\mathcal{H} \to \mathcal{G})$.

Lemma 12 *There is an algorithm to compute* $\#\mathrm{Hom}(\mathcal{H} \to \mathcal{G})$ *in time* $O(nk^k(\Delta+1)^k)$, *where $\mathcal{G}$ is a coloured graph with n vertices, $\mathcal{H}$ is a coloured graph with k vertices, and both graphs have maximum degree at most Δ.*

Proof The algorithm works as follows: If $\mathcal{H}$ is not connected, let $\mathcal{H}_1, \ldots, \mathcal{H}_\ell$ be its connected components. Then it is straightforward to verify that

$$\#\mathrm{Hom}(\mathcal{H} \to G) = \prod_{i=1}^{\ell} \#\mathrm{Hom}(\mathcal{H}_i \to \mathcal{G}).$$

Thus it remains to describe the algorithm for connected pattern graphs $\mathcal{H}$.

Let $\mathcal{H}$ be connected. A sequence of vertices $v_1, \ldots, v_k$ in a graph F is a *traversal* if, for all $i \in \{1, \ldots, k-1\}$, the vertex v_{i+1} is contained in $\{v_1, \ldots, v_i\} \cup \Gamma(\{v_1, \ldots, v_i\})$. Let $u_1, \ldots, u_k$ be an arbitrary traversal of $\mathcal{H}$ with $\{u_1, \ldots, u_k\} = V(\mathcal{H})$; the latter property can be satisfied since $\mathcal{H}$ is a connected graph with k vertices. Note that if $f : V(\mathcal{H}) \to V(\mathcal{G})$ is a homomorphism from $\mathcal{H}$ to $\mathcal{G}$, then $f(u_1), \ldots, f(u_k)$ is a traversal in $\mathcal{G}$, and this correspondence is injective. Thus the algorithm computes the number of traversals $v_1, \ldots, v_k$ in $\mathcal{G}$ for which the mapping f with $f(u_i) = v_i$ for all i is a homomorphism from $\mathcal{H}$ to $\mathcal{G}$. This number is equal to $\#\mathrm{Hom}(\mathcal{H} \to \mathcal{G})$, which the algorithm seeks to compute.

Since the maximum degree of G is Δ, any set $S \subseteq V(\mathcal{G})$ satisfies $|\Gamma(S)| \leq \Delta|S|$. Thus there are at most $n \cdot (\Delta k + k)^{k-1}$ traversal sequences in $\mathcal{G}$, which can be generated in linear time in the number of such sequences. For each traversal sequence, verifying whether the sequence corresponds to a homomorphism takes time $O(k\Delta)$ (in the word-RAM model with incidence lists for $\mathcal{H}$ already prepared). Overall, we obtain a running time of $O(n \cdot k^k \cdot (\Delta + 1)^k)$.   □

Using the above algorithm, we now construct an algorithm that performs a kind of basis transformation to obtain the number of induced coloured subgraphs.

Theorem 13 *For all positive integers Δ, there is a fixed-parameter tractable algorithm for* #Induced-Coloured-Subgraph[Δ] *with time complexity* $O(n \cdot k^{(2\Delta+3)\cdot k})$ *for n-vertex coloured graphs $\mathcal{G}$ and k-vertex coloured graphs $\mathcal{H}$.*

Proof Let $(\mathcal{H}, \mathcal{G})$ be an instance of #Induced-Coloured-Subgraph[Δ], write $\mathcal{G} = (G, c)$ and $\mathcal{H} = (H, c')$, and let k be the number of vertices of $\mathcal{H}$. Without loss of generality, suppose that the ranges of c and c' are $[q]$ for some positive integer $q \leq k$. Namely, if any vertices of G receive colours not in the range of c', then our algorithm may remove them without affecting $\#\mathrm{Ind}(\mathcal{H} \to \mathcal{G})$; if any vertices of H receive colours not in the range of c, then $\#\mathrm{Ind}(\mathcal{H} \to \mathcal{G}) = 0$.

For coloured graphs $\mathcal{K}$ and $\mathcal{B}$, let $\#\mathrm{Surj}(\mathcal{K} \to \mathcal{B})$ be the number of vertex-surjective homomorphisms from $\mathcal{K}$ to $\mathcal{B}$, i.e., the number of those homomorphisms from $\mathcal{K}$ to $\mathcal{B}$ that contain all vertices of $\mathcal{B}$ in their image.

Let S be the set of all q-coloured graphs $\mathcal{K}$ such that $\Delta(\mathcal{K}) \leq \Delta$ and, for some $t \in [k]$, $V(\mathcal{K}) = [t]$. Let S' be a set of representatives of (coloured) isomorphism classes of S.

Let $\boldsymbol{x}$ be the vector indexed by S' such that $\boldsymbol{x}_\mathcal{K} = \#\operatorname{Ind}(\mathcal{K} \to \mathcal{G})$ for all $\mathcal{K} \in S'$. This vector contains the number of induced subgraph copies of $\mathcal{H}$ in $\mathcal{G}$, but it also contains the number of subgraph copies of all other graphs in S' in $\mathcal{G}$. Let $\boldsymbol{b}$ be the vector indexed by S' such that $\boldsymbol{b}_\mathcal{K} = \#\operatorname{Hom}(\mathcal{K} \to \mathcal{G})$ for all $\mathcal{K} \in S'$; each entry of this vector can be computed via the algorithm of Lemma 12. Then we will show that $\boldsymbol{x}$ and $\boldsymbol{b}$ can be related to each other via an invertible matrix A such that $A\boldsymbol{x} = \boldsymbol{b}$. By calculating A and $\boldsymbol{b}$, we can then output $\#\operatorname{Ind}(\mathcal{H} \to \mathcal{G}) = (A^{-1}\boldsymbol{b})_\mathcal{H}$.

To elaborate on this linear relationship between induced subgraph and homomorphism numbers, let us first consider some arbitrary graph $\mathcal{K} \in S'$. By partitioning the homomorphisms from $\mathcal{K}$ to $\mathcal{G}$ according to their image, we have

$$\#\operatorname{Hom}(\mathcal{K} \to \mathcal{G}) = \sum_{\substack{X \subseteq V(\mathcal{G}) \\ |X| \leq k}} \#\operatorname{Surj}(\mathcal{K} \to \mathcal{G}[X]).$$

In the right-hand side sum, we can collect terms with isomorphic induced subgraphs $\mathcal{G}[X]$, since we clearly have $\#\operatorname{Surj}(\mathcal{K} \to \mathcal{B}) = \#\operatorname{Surj}(\mathcal{K} \to \mathcal{B}')$ if $\mathcal{B} \simeq \mathcal{B}'$. Hence, we obtain

$$\#\operatorname{Hom}(\mathcal{K} \to \mathcal{G}) = \sum_{\mathcal{K}' \in S'} \#\operatorname{Surj}(\mathcal{K} \to \mathcal{K}') \cdot \#\operatorname{Ind}(\mathcal{K}' \to \mathcal{G}). \tag{8}$$

Let A be the matrix indexed by S' with $A_{\mathcal{K},\mathcal{K}'} = \#\operatorname{Surj}(\mathcal{K} \to \mathcal{K}')$ for all $\mathcal{K}, \mathcal{K}' \in S'$. Then (8) implies that $A\boldsymbol{x} = \boldsymbol{b}$. (An uncoloured version of this linear system is originally due to Lovász [18].)

We next prove that A is invertible. Indeed, given $\mathcal{K}, \mathcal{K}' \in S'$, write $\mathcal{K} \precsim \mathcal{K}'$ if $\mathcal{K}$ admits a vertex-surjective homomorphism to $\mathcal{K}'$. Since $\precsim$ is a partial order, as is readily verified, it admits a topological ordering π. Permuting the rows and columns of A to agree with π does not affect the rank of A, and it yields an upper triangular matrix with non-zero diagonal entries, so it follows that A is invertible.

The algorithm is now immediate. It first determines S by listing all q-coloured graphs on at most k vertices with at most $\lfloor \Delta k/2 \rfloor$ edges, then checking each one to see whether it satisfies the degree condition. It then determines S' from S by testing every pair of coloured graphs in S for isomorphism (by brute force). It then determines each entry $A_{\mathcal{K},\mathcal{K}'}$ of A (by brute force) by listing the vertex-surjective maps $\mathcal{K} \to \mathcal{K}'$. It then determines $\boldsymbol{b}$ by invoking Lemma 12 to compute each entry $\boldsymbol{b}_\mathcal{K} = \#\operatorname{Hom}(\mathcal{K} \to \mathcal{G})$ for $\mathcal{K} \in S'$. Finally, it outputs $\#\operatorname{Ind}(\mathcal{H} \to \mathcal{G}) = (A^{-1}\boldsymbol{b})_\mathcal{H}$.

Running time. All arithmetic operations are applied to integers bounded by n^k, so they each fit into $O(k)$ words, and we bound the complexity of each operation crudely by $O(k^2)$. The number of q-coloured graphs on t vertices with at most $\lfloor \Delta k/2 \rfloor$ edges is at most

$$k \cdot q^k \cdot \sum_{m=0}^{\lfloor \Delta k/2 \rfloor} \binom{\binom{k}{2}}{m} \leq k \cdot k^k \cdot \frac{\Delta k}{2} \cdot k^{2\lfloor \Delta k/2 \rfloor} = O(k^{2+(\Delta+1)k}) \text{ as a function of } k,$$

so our algorithm determines S in time $O(k^{(\Delta+2)k})$ and $|S| = O(k^{2+(\Delta+1)k})$. Moreover, checking whether two graphs in S are isomorphic by brute force requires $O(k^2 \cdot k!)$ time, so our algorithm determines S' in time $O(|S|^2 k^2 \cdot k!) = O(k^{(2\Delta+3)k})$ time. In determining A, the algorithm checks at most $k!$ possible homomorphisms for each of $|S'|^2$ pairs of graphs, so it again takes time $O(k^{(2\Delta+3)k})$. In determining $\boldsymbol{b}$, the algorithm computes $|S'| = O(k^{2+(\Delta+1)k})$ entries in total, each of which takes time $O(nk^k(\Delta+1)^k)$, so in total it takes time $O(nk^{(\Delta+3)k})$. Finally, it takes $O(k^2|S'|^2) = O(k^{(2\Delta+3)k})$ time to invert A and determine $\boldsymbol{x}$ (since A can be put into upper triangular form by permuting rows and columns). Overall, the running time of the algorithm is $O(nk^{(2\Delta+3)k})$, as claimed. □

We note that the above algorithm is not limited to host graphs of bounded degree. That is, the same approach can be taken for any host graph class for which counting homomorphisms from (vertex-coloured) patterns with k vertices has an $f(k) \cdot n^{O(1)}$ time algorithm. To this end, simply use this algorithm as a sub-routine instead of Lemma 12 in the algorithm constructed in the proof of Theorem 13. Examples for such classes of host graphs are planar graphs or, more generally, any graph class of bounded local treewidth [11].

Theorem 14 *For all positive integers Δ:*

  (i) *#Size-BIS [Δ] has an algorithm with time complexity $O(|V(G)| \cdot k^{(2\Delta+3)k})$;*

  (ii) *#Size-Left-BIS[Δ] has an algorithm with time complexity $O(|V(G)| \cdot \ell^{\ell(2\Delta^2+8\Delta+4)})$;*

  (iii) *#Size-Left-Max-BIS[Δ] has an algorithm with time complexity $O(|V(G)| \cdot \ell^{\ell(2\Delta^2+8\Delta+4)})$.*

Recent independent work by Patel and Regts [20] implicitly contains an algorithm for counting independent sets of size k in graphs of maximum degree Δ in time $O(c^k n)$, where c is a constant depending on Δ. This implies Theorem 14(i). Since our own proof is very short, we provide it for the benefit of the reader. Subsequent work [21], published after our original paper [3], includes a version of Theorem 13 with running time $\tilde{O}((4\Delta)^{2k} n)$ (which is essentially best-possible under ETH). Note that using this result in the proof of Theorem 14(ii) and (iii) in place of Theorem 13 would not yield algorithms with running times $n \cdot \ell^{o(\ell)}$, as the quantity $|\mathcal{S}'_{\ell,r}|$ defined in the proof is $\ell^{\Omega(\ell)}$ when $\Delta = 3$ (for suitable values of r).

Proof **Proof of part (i):** This is immediate from Theorem 13, since #Size-BIS[Δ] is a special case of #Induced-Coloured-Subgraph[Δ] (taking $\mathcal{G}$ to be monochromatic and $\mathcal{H}$ to be a monochromatic independent set of size k).

Proof of part (ii): For any bipartite graph $G = (U, V, E)$ with degree at most Δ and any non-negative integers ℓ and r, let $N_{\ell,r}(G)$ be the number of sets $X \subseteq U$ with $|X| = \ell$ and $|\Gamma(X)| = r$. Let $N'_{\ell,r}(G)$ be the number of pairs of sets $X \subseteq U, Y \subseteq V$ such that $|X| = \ell$, $|Y| = r$ and $Y \subseteq \Gamma(X)$. Then we have

$$N_{\ell,r}(G) = N'_{\ell,r}(G) - \sum_{i=r+1}^{\Delta\ell} \binom{i}{r} N_{\ell,i}(G). \tag{9}$$

For any bipartite graph $J = (U_J, V_J, E_J)$, we define the corresponding 2-colouring by $c_J(v) = 1$ for all $v \in U_J$ and $c_J(v) = 2$ for all $v \in V_J$. We define the corresponding coloured graph by $\phi(J) = ((U_J \cup V_J, \{\{u, v\} \mid (u, v) \in E_J\}), c_J)$. Let $S_{\ell,r}$ be the set of all bipartite graphs $J = (U_J, V_J, E_J)$ with $U_J = [\ell]$, $V_J = \{\ell + 1, \ldots, \ell + r\}$, degree at most Δ and no isolated vertices in V_J. Let $\mathcal{S}_{\ell,r}$ be the corresponding set of coloured graphs, and let $\mathcal{S}'_{\ell,r}$ be a set of representatives of (coloured) isomorphism classes in $\mathcal{S}_{\ell,r}$. Then $N'_{\ell,r}(G) = \sum_{\mathcal{K} \in \mathcal{S}'_{\ell,r}} \#\mathrm{Ind}(\mathcal{K} \to \phi(G))$, and hence by (9) we have

$$N_{\ell,r}(G) = \sum_{\mathcal{K} \in \mathcal{S}'_{\ell,r}} \#\mathrm{Ind}(\mathcal{K} \to \phi(G)) - \sum_{i=r+1}^{\Delta\ell} \binom{i}{r} N_{\ell,i}(G). \tag{10}$$

Now suppose that (G, ℓ) is an instance of #Size-Left-BIS[Δ]. Then we have

$$\mathrm{IS}_{\ell\text{-left}}(G) = \sum_{\substack{X \subseteq U \\ |X| = \ell}} 2^{|V| - |\Gamma(X)|} = \sum_{r=0}^{\Delta\ell} N_{\ell,r}(G) 2^{|V| - r}. \tag{11}$$

To compute $N_{\ell,\Delta\ell}(G), \ldots, N_{\ell,0}(G)$, our algorithm applies (10). For each $r \in \{\Delta\ell, \ldots, 0\}$, it determines the $\#\mathrm{Ind}(\mathcal{K} \to \phi(G))$ terms of (10) using the #Induced-Coloured-Subgraph[Δ] algorithm of Theorem 13, and the remaining terms of (10) recursively with dynamic programming. Finally, it computes $\mathrm{IS}_{\ell\text{-left}}(G)$ using (11).

To determine the time complexity, first note that $|S_{\ell,r}| \leq \binom{\Delta\ell^2}{\Delta\ell} = O(\ell^{3\Delta\ell})$ holds for all $r \in \{\Delta\ell, \ldots, 0\}$. The algorithm therefore determines $\mathcal{S}'_{\ell,r}$ by brute force in time $O(|S_{\ell,r}|^2(\ell + \Delta\ell)^{\ell+\Delta\ell}) = O(\ell^{\ell(8\Delta+2)})$. The algorithm then calculates each $N_{\ell,r}(G)$ in time

$$O(|S'_{\ell,r}| \cdot |V(G)| \cdot (\ell + \Delta\ell)^{(2\Delta+3)(\ell+\Delta\ell)}) = O(|V(G)| \cdot \ell^{\ell(2\Delta^2+8\Delta+4)}).$$

The overall running time is therefore $O(|V(G)| \cdot \ell^{\ell(2\Delta^2+8\Delta+4)})$, so part (ii) of the result follows.

Proof of part (iii): Finally, suppose that (G, ℓ) is an instance of #Size-Left-Max-BIS [Δ]. Let $\mu = \min\{r \mid N_{\ell,r}(G) \neq 0\}$, and note that $\mathrm{IS}_{\ell\text{-left-max}}(G) = N_{\ell,\mu}(G)$. As above, our algorithm determines $N_{\ell,\Delta\ell}(G), \ldots, N_{\ell,0}(G)$ using (10), and thereby determines and outputs $N_{\ell,\mu}(G)$. The overall running time is again $O(|V(G)| \cdot \ell^{\ell(2\Delta^2+8\Delta+4)})$, so part (iii) of the result follows.   □

References

1. Arvind, V., Raman, V.: Approximation Algorithms for Some Parameterized Counting Problems, pp. 453–464. Springer, Berlin (2002)
2. Cai, J.-Y., Galanis, A., Goldberg, L.A., Guo, H., Jerrum, M., Stefankovic, D., Vigoda, E.: #BIS-hardness for 2-spin systems on bipartite bounded degree graphs in the tree non-uniqueness region. J. Comput. Syst. Sci. **82**(5), 690–711 (2016)
3. Curticapean, R., Dell, H., Fomin, F., Goldberg, L.A., Lapinskas, J.: A fixed-parameter perspective on #BIS. In: 12th International Symposium on Parameterized and Exact Computation (IPEC), pp. 13:1–13:13 (2017)
4. Curticapean, R., Dell, H., Marx, D.: Homomorphisms are a good basis for counting small subgraphs. In: Proceedings of the 49th Annual ACM Symposium on Theory of Computing, pp. 210–213 (2017)
5. Curticapean, R., Marx, D.: 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, 18–21 Oct 2014, pp. 130–139 (2014)
6. Cygan, M., Fomin, F., Kowalik, Ł., Lokshtanov, D., Marx, D., Pilipczuk, M., Pilipczuk, M., Saurabh, S.: Parameterized Algorithms. Springer, Berlin (2015)
7. Downey, R.G., Fellows, M.R.: Fundamentals of Parameterized Complexity. Springer, Berlin (2013)
8. Dyer, M.E., Goldberg, L.A., Greenhill, C.S., Jerrum, M.: The relative complexity of approximate counting problems. Algorithmica **38**(3), 471–500 (2003)
9. Flum, J., Grohe, M.: The parameterized complexity of counting problems. SIAM J. Comput. **33**(4), 892–922 (2004)
10. Flum, J., Grohe, M.: Parameterized Complexity Theory. Springer, Berlin (2006)
11. Frick, M.: Generalized model-checking over locally tree-decomposable classes. Theory Comput. Syst. **37**(1), 157–191 (2004)
12. Galanis, A., Stefankovic, D., Vigoda, E., Yang, L.: Ferromagnetic Potts model: refined #BIS-hardness and related results. SIAM J. Comput. **45**(6), 2004–2065 (2016)
13. Gessel, I., Viennot, G.: Binomial determinants, paths, and hook length formulae. Adv. Math. **58**(3), 300–321 (1985)
14. Goldberg, L.A., Jerrum, M.: A complexity classification of spin systems with an external field. Proc. Natl. Acad. Sci. **112**(43), 13161–13166 (2015)
15. Impagliazzo, R., Paturi, R.: On the complexity of k-SAT. J. Comput. Syst. Sci. **62**(2), 367–375 (2001)
16. Janson, S., Łuczak, T., Rucinski, A.: Random Graphs. Wiley, New York (2000)
17. Liu, J., Lu, P.: FPTAS for #BIS with degree bounds on one side. In: Proceedings of the Forty-Seventh Annual ACM on Symposium on Theory of Computing, STOC 2015, Portland, OR, USA, 14–17 June 2015, pp. 549–556 (2015)
18. Lovász, L.: Large Networks and Graph Limits. Colloquium Publications, vol. 60. American Mathematical Society, Providence (2012)
19. Müller, M.: Randomized approximations of parameterized counting problems. In: Proceedings of the Second International Conference on Parameterized and Exact Computation, IWPEC'06, pp. 50–59. Springer, Berlin (2006)
20. Patel, V., Regts, G.: Deterministic polynomial-time approximation algorithms for partition functions and graph polynomials. CoRR, abs/1607.01167 (2016)
21. Patel, V., Regts, G.: Computing the number of induced copies of a fixed graph in a bounded degree graph. CoRR, abs/1707.05186 (2017)
22. Sipser, M.: Introduction to the Theory of Computation, 1st edn. International Thomson Publishing, Stamford (1996)
23. Vadhan, S.P.: The complexity of counting in sparse, regular, and planar graphs. SIAM J. Comput. **31**(2), 398–427 (2001)
24. Xia, M., Zhang, P., Zhao, W.: Computational complexity of counting problems on 3-regular planar graphs. Theor. Comput. Sci. **384**(1), 111–125 (2007)

Affiliations

Radu Curticapean[1] · Holger Dell[2] · Fedor Fomin[3] · Leslie Ann Goldberg[4] · John Lapinskas[4]

✉ John Lapinskas
    john.lapinskas@cs.ox.ac.uk

    Radu Curticapean
    radu.curticapean@gmail.com

    Holger Dell
    hold@itu.dk

    Fedor Fomin
    fomin@ii.uib.no

    Leslie Ann Goldberg
    leslie.goldberg@cs.ox.ac.uk

[1] Basic Algorithms Research Copenhagen, IT University of Copenhagen, Copenhagen, Denmark

[2] Department of Computer Science, IT University of Copenhagen, Copenhagen, Denmark

[3] Department of Informatics, University of Bergen, Bergen, Norway

[4] Department of Computer Science, University of Oxford, Oxford, UK