Computing the Number of Induced Copies of a Fixed Graph in a Bounded Degree Graph

Patel, V.; Regts, G.

DOI
10.1007/s00453-018-0511-9

Publication date
2019

Document Version
Final published version

Published in
Algorithmica

Algorithmica (2019) 81:1844–1858
https://doi.org/10.1007/s00453-018-0511-9

Computing the Number of Induced Copies of a Fixed Graph in a Bounded Degree Graph

Viresh Patel[1] · **Guus Regts**[1]

Received: 21 July 2017 / Accepted: 27 August 2018 / Published online: 5 September 2018

Abstract
In this paper we show that for any graph H of order m and any graph G of order n and maximum degree Δ one can compute the number of subsets S of $V(G)$ that induces a graph isomorphic to H in time $O(c^m \cdot n)$ for some constant $c = c(\Delta) > 0$. This is essentially best possible (in the sense that there is no $c^{o(m)}\, poly(n)$-time algorithm under the exponential time hypothesis).

Keywords Induced graph · Computational counting · Fixed parameter tractability

1 Introduction

For two graphs H and G we denote by $\mathrm{ind}(H, G)$ the number of subsets of the vertex set of G that induce a graph that is isomorphic to H. (We recall that two graphs $H = (V_H, E_H)$ and $G = (V_G, E_G)$ are said to be *isomorphic* if there exists a bijection $f : V_H \to V_G$ such that for any $u, v \in V_H$, we have that $f(u)f(v) \in E_G$ if and only if $uv \in E_H$.) Throughout we take G and H to have n and m vertices respectively unless otherwise stated.

Understanding the numbers $\mathrm{ind}(H, G)$ for different choices of H gives us much important information about G. For example, if H is the disjoint union of m isolated vertices, $\mathrm{ind}(H, G)$ equals the number of independent sets of size m in G. Determining these induced subgraph counts is closely related to determining subgraph counts and homomorphism counts; these parameters play a central role in the theory of graph

Viresh Patel: Supported by the Netherlands Organisation for Scientific Research (NWO) through the Gravitation Programme Networks (024.002.003). Guus Regts: Supported by a personal NWO Veni grant.

✉ Viresh Patel
vpatel@uva.nl

Guus Regts
guusregts@gmail.com

[1] Korteweg de Vries Institute for Mathematics, University of Amsterdam, Amsterdam, The Netherlands

limits [15], and frequently appear in statistical physics, see e.g. Sect. 2.2 in [19] and the references therein.

When H and G are both part of the input, computing $\mathrm{ind}(H, G)$ is clearly an NP-hard problem because it includes the problem of determining the size of a maximum clique in G. When the graph H is fixed (with m vertices), the brute-force algorithm takes time $O(n^m)$ and a linear improvement has been made to the exponent [17].

This problem and some its variants (e.g. where we count not necessarily induced subgraphs) have been studied from a fixed parameter tractability (FTP) perspective; see [2,6,10,12]). In general computing $\mathrm{ind}(H, G)$ when parameterizing by $m = |V(H)|$ is $\#W[1]$-hard because even deciding whether G contains an independent set of size m is $\#W[1]$-hard [11]. Curticapean, Dell and Marx [9] prove a number of interesting dichotomy results for $\#W[1]$-hardness using the treewidth of H (and of a certain class of graphs obtained from H) as an additional parameter.

However, when the graph G is of bounded degree, which is often of interest in statistical physics, the problem is no longer $\#W[1]$-hard. Indeed, Curticapean, Dell, Fomin, Goldberg, and Lapinskas [8, Theorem 13] showed that for a graph H on m vertices and a bounded degree graph G on n vertices, $\mathrm{ind}(H, G)$ can be computed in time $O(m^{O(m)}n)$, thus giving an FPT algorithm in terms of m. In the present paper we go further and give an algorithm with essentially optimal running time. We assume the standard word-RAM machine model with logarithmic-sized words.

Theorem 1.1 *There is an algorithm which, given an n-vertex graph G of maximum degree at most Δ, and an m-vertex graph H, computes $\mathrm{ind}(H, G)$ in time $\tilde{O}((7\Delta)^{2m}n + 2^{10m})$. (Here the $\tilde{O}$-notation means that we suppress polynomial factors in m.)*

Remark 1.1 Theorem 13 in [8] in fact concerns vertex-coloured graphs H and G. Our proof of Theorem 1.1 also easily extends to the coloured setting. We discuss this in Sect. 4.

The running time here is essentially optimal under the exponential time hypothesis. Indeed, if we could find an algorithm with an improved running time $c^{o(m)}\,poly(n)$ (for some constant c possibly dependent on Δ), we could use it to determine the size of a maximum independent set in time $nc^{o(n)}\,poly(n) = c^{o(n)}\,poly(n)$, which is not possible (even in graphs of maximum degree 3) under the exponential time hypothesis; see [13, Lemma 2.1]. (In fact, under the exponential time hypothesis, we may draw the stronger conclusion that there is no $c'^{m}\,poly(n)$-time algorithm for some $c' > 0$.)

Note that our algorithm allows us to compute $\mathrm{ind}(H, G)$ in polynomial time in $|G|$, even when $|H|$ is logarithmic in $|G|$ and G has bounded degree. The special case of this when H is an independent set was a crucial ingredient in our recent paper [18], which uses the Taylor approximation method of Barvinok [3] to give (amongst others) a fully polynomial time approximation scheme for evaluating the independence polynomial for bounded degree graphs. Our present paper completes the running time complexity picture for computing $\mathrm{ind}(H, G)$ on bounded degree graphs G.

We add a few remarks to give further perspective on the problem. Note that computing $\mathrm{ind}(H, G)$ in time $\tilde{O}(\Delta^m n)$ is relatively straightforward for G of bounded degree Δ when H is *connected* (see Lemma 2.2). Thus the difficulty lies in graphs H that have

many components. Note also that Curticapean et al. [8] use the fact that induced graph counts can be expressed in terms of homomorphism counts (see e.g. [15]) and that homomorphism counts from H to G can be computed in time $\tilde{O}(\Delta^m n)$ in their FPT algorithm. However the limiting factor is the time dependence (on m) of expressing induced graph counts in terms of homomorphism graph counts, which could be as large as the Bell number. This is significantly larger than the time dependence in our algorithm.

Both our approach and the approach in [8] crucially use the bounded degree assumption. It would be very interesting to know if Theorem 1.1 could be extended to graphs of average bounded degree such as for example planar graphs.

Question 1 For which class of graphs $\mathcal{C}$ does there exist a constant $c = c(\mathcal{C})$ and an algorithm such that given an n-vertex graph $G \in \mathcal{C}$ and an m-vertex graph H, the algorithm computes $\text{ind}(H, G)$ in time $O(c^m poly(n))$?

We note that Nederlof [16] recently showed that there exists a constant c and an algorithm that computes the number of independent sets of size m in an n-vertex planar graph in time $c^{O(m)}n$.

Organization The remainder of the paper is devoted to proving Theorem 1.1. The main idea in our proof is to define a multivariate graph polynomial where the coefficients are certain induced graph counts; in particular $\text{ind}(H, G)$ will be the coefficient of a monomial. We cannot compute these coefficients directly, but use machinery from [18] to compute the coefficients of univariate evaluations of this polynomial. In Sect. 3, we use algebraic techniques to efficiently extract $\text{ind}(H, G)$ (i.e. the coefficient of interest) from the coefficients of the univariate evaluations. The resulting algorithm to compute $\text{ind}(H, G)$ is summarized in Sect. 3.

We however need to slightly modify the result from [18]. This will be done in the next section.

2 Computing Coefficients of Graph Polynomials

An efficient way to compute the coefficients of a large class of (univariate) graph polynomials for bounded degree graphs was given in [18]. We will need a small modification of this result, for which we will provide the details here. In particular, the running time has been improved compared to [18] and we have clarified the dependence on the parameters. We start with some definitions after which we state the main result of this section.

By $\mathcal{G}$ we denote the collection of all graphs and by $\mathcal{G}_k$ for $k \in \mathbb{N}$ we denote the collection of graphs with at most k vertices. A graph invariant is a function $f : \mathcal{G} \to S$ for some set S that takes the same value on isomorphic graphs. A (univariate) *graph polynomial* is a graph invariant $p : \mathcal{G} \to \mathbb{C}[z]$, where $\mathbb{C}[z]$ denotes the ring of polynomials in the variable z over the field of complex numbers. Call a graph invariant f *multiplicative* if $f(\emptyset) = 1$ and $f(G_1 \cup G_2) = f(G_1)f(G_2)$ for all graphs G_1, G_2 (here $G_1 \cup G_2$ denotes the disjoint union of the graphs G_1 and G_2). We can now give the key definition and tool we need from [18].

Definition 2.1 Let p be a multiplicative graph polynomial defined by

$$p(G)(z) := \sum_{i=0}^{d(G)} e_i(G) z^i \tag{1}$$

for each $G \in \mathcal{G}$ with $e_0(G) = 1$, where $d(G)$ is the degree of the polynomial $p(G)$. We call p a *bounded induced graph counting polynomial (BIGCP)* if there exists $\alpha \in \mathbb{N}$, an algorithm A and a non-decreasing sequence $\beta \in \mathbb{N}^{\mathbb{N}}$ such that the following two conditions are satisfied:

(i) for every graph G, the coefficients e_i satisfy

$$e_i(G) := \sum_{H \in \mathcal{G}_{\alpha i}} \lambda_{H,i} \mathrm{ind}(H, G) \tag{2}$$

for certain $\lambda_{H,i} \in \mathbb{C}$;

(ii) for each i and $H \in \mathcal{G}_{\alpha i}$, the algorithm A computes the coefficient $\lambda_{H,i}$ in time β_i.

We note that the coefficients of BIGCPs can be seen as *graph motif parameters* as introduced in [9].

We have the following result for computing coefficients of BIGCPs.

Theorem 2.1 *Let $n, m, \Delta \in \mathbb{N}$ and let $p(\cdot)$ be a bounded induced graph counting polynomial with parameters α and β. Then there is a deterministic $\tilde{O}(n(e\Delta)^{\alpha m}\beta_m 4^{\alpha m})$-time algorithm, which, given any n-vertex graph G of maximum degree at most Δ, computes the first m coefficients $e_1(G), \ldots, e_m(G)$ of $p(G)$. (Here the $\tilde{O}$-notation means that we suppress polynomial factors in m.)*

Remark 2.1 The algorithm in the theorem above only has access to the polynomial p via the algorithm A in the definition of BIGCP that computes the complex numbers $\lambda_{H,i}$.

Before we prove Theorem 2.1 we will first gather some facts from [18] about induced subgraph counts and the number of connected induced subgraphs of fixed size that occur in a graph. Compared to [18] we actually need to slightly sharpen the statements.

2.1 Induced Subgraph Counts

Define $\mathrm{ind}(H, \cdot) : \mathcal{G} \to \mathbb{C}$ by $G \mapsto \mathrm{ind}(H, G)$. So we view $\mathrm{ind}(H, \cdot)$ as a graph invariant. We can take linear combinations and products of these invariants. In particular, for two graphs H_1, H_2 we have

$$\mathrm{ind}(H_1, \cdot) \cdot \mathrm{ind}(H_2, \cdot) = \sum_{H \in \mathcal{G}} c^H_{H_1,H_2} \mathrm{ind}(H, \cdot), \tag{3}$$

where for a graph H, $c^H_{H_1,H_2}$ is the number of pairs of subsets of $V(H)$, (U,T), such that $U \cup T = V(H)$ and $H[U] = H_1$ and $H[T] = H_2$. In particular, given H_1 and H_2, $c^H_{H_1,H_2}$ is nonzero for only a finite number of graphs H.

In what follows we will often have to maintain a list L of subsets S of $[n]$ with $|S| \leq k$ (for some k) as well as some (complex) number c_S associated to S. We will use the standard word-RAM machine model with logarithmic-sized words. This means that given a set S of size k, we have access to c_S in $O(k)$ time. In particular, this also means we can determine whether S is contained in our list in $O(k)$ time.

The next lemma says that computing $\mathrm{ind}(H,G)$ is fixed parameter tractable when G has bounded degree and H is connected. The following lemma is a variation on [18, Lemma 3.5].

Lemma 2.2 *Let H be a connected graph on k vertices and let $\Delta \in \mathbb{N}$. Then*

(i) *there is an $O(n\Delta^{k-1})$-time algorithm, which, given any n-vertex graph G with maximum degree at most Δ, checks whether $\mathrm{ind}(H,G) \neq 0$;*
(ii) *there is an $O(n\Delta^{k-1}k)$-time algorithm, which, given any n-vertex graph G with maximum degree at most Δ, computes the number $\mathrm{ind}(H,G)$.*

Note that Lemma 2.2 (i) enables us to test for graph isomorphism between connected bounded degree graphs when $|V(G)| = |V(H)|$.

Proof We follow the proof from [18]. We assume that $V(G) = [n]$. Let us list the vertices of $V(H)$, $v_1, \ldots, v_k$ in such a way that for $i \geq 1$ vertex v_i has a neighbour among $v_1, \ldots, v_{i-1}$. Then to embed H into G we first select a target vertex for v_1 and then given that we have embedded $v_1, \ldots, v_{i-1}$ with $i \geq 2$ there are at most Δ choices for where to embed v_i. After k iterations, we have a total of at most $n\Delta^{k-1}$ potential ways to embed H and each possibility is checked in the procedure above. Hence we determine if $\mathrm{ind}(H,G)$ is zero or not in $O(n\Delta^{k-1})$ time.

Throughout the procedure above we maintain a list L that contains all sets S such that $G[S] = H$ found thus far. Each time we find a set $S \subset [n]$ such that $G[S] = H$ we check if it is contained in L. If this is not the case we add S to L and we discard S otherwise. The length of the resulting list, which we update at each iteration, gives the value of $\mathrm{ind}(H,G)$. □

Next we consider how to enumerate all possible connected induced subgraphs of fixed size in a bounded degree graph. We will need the following result of Borgs, Chayes, Kahn, and Lovász [5, Lemma 2.1]:

Lemma 2.3 *Let G be a graph of maximum degree Δ. Fix a vertex v_0 of G. Then the number of connected induced subgraphs of G with k vertices containing the vertex v_0 is at most $\frac{(e\Delta)^{k-1}}{2}$.*

As a consequence we can efficiently enumerate all connected induced subgraphs of logarithmic size that occur in a bounded degree graph G.

Lemma 2.4 *There is a $O(nk^3(e\Delta)^k)$-time algorithm which, given $k \in \mathbb{N}$ and an n-vertex graph G on $[n]$ of maximum degree Δ, outputs $\mathcal{T}_k$, the list of all $S \subseteq [n]$ satisfying $|S| \leq k$ and $G[S]$ connected.*

Proof We assume that $V(G) = [n]$. By the previous result, we know that $|\mathcal{T}_k| \leq n(e\Delta)^{k-1}$ for all k.

We inductively construct $\mathcal{T}_k$. For $k = 1$, $\mathcal{T}_k$ is clearly the set of singleton vertices and takes time $O(n)$ to output.

Given that we have found $\mathcal{T}_{k-1}$ we compute $\mathcal{T}_k$ as follows. We iteratively compute $\mathcal{T}_k$ by going over all $S \in \mathcal{T}_{k-1}$ going over all $v \in N_G(S)$ (the collection of vertices that are connected to an element of S) and checking whether $S \cup \{v\}$ is already contained in $\mathcal{T}_k$ or not. We add it to $\mathcal{T}_k$ if it is not already contained in $\mathcal{T}_k$.

The set $N_G(S)$ has size at most $|S|\Delta \leq k\Delta$ and takes time $O(k\Delta)$ to find (assuming G is given in adjacency list form). Therefore computing $\mathcal{T}_k$ takes time bounded by $O(|\mathcal{T}_{k-1}|k^2\Delta) = O(nk^2(e\Delta)^k)$.

Starting from $\mathcal{T}_1$, we perform the above iteration k times, requiring a total running time of $O(nk^3(e\Delta)^k)$. The proof that $\mathcal{T}_k$ contains all the sets we desire is straightforward and can be found in [18]. □

We call a graph invariant $f : \mathcal{G} \to \mathbb{C}$ *additive* if for each $G_1, G_2 \in \mathcal{G}$ we have $f(G_1 \cup G_2) = f(G_1) + f(G_2)$. The following lemma is a variation of a lemma due to Csikvári and Frenkel [7]; it is fundamental to our approach. See [18] for a proof.

Lemma 2.5 *Let $f : \mathcal{G} \to \mathbb{C}$ be a graph invariant given by $f(\cdot) := \sum_{H \in \mathcal{G}} a_H \mathrm{ind}(H, \cdot)$. Then f is additive if and only if $a_H = 0$ for all graphs H that are disconnected.*

Let $p(z) = a_0 + \cdots + a_d z^d$ be a polynomial of degree d with nonzero constant term a_0 and with complex roots $\zeta_1, \ldots, \zeta_d$. Define for $j \in \mathbb{N}$, *the inverse power sum* p_j by

$$p_j := \zeta_1^{-j} + \cdots + \zeta_d^{-j}.$$

The next proposition is a variant of the Newton identities that relate the inverse power sums and the coefficients of a polynomial. We refer to [18] for a proof.

Proposition 2.6 *Let $p(z) = a_0 + \cdots + a_d z^d$ be a polynomial of degree d with $a_0 \neq 0$ and inverse power sums p_j, $j \in \mathbb{N}$. Then for each $k = 1, 2, \ldots$, we have*

$$k a_k = -\sum_{i=0}^{k-1} a_i \, p_{k-i}.$$

(Here we take $a_i = 0$ if $i > d$.)

2.2 Proof of Theorem 2.1

We follow the proof as given in [18], which we modify slightly at certain points.

Recall that $p(\cdot)$ is a bounded induced graph counting polynomial (BIGCP). Given an n-vertex graph G with maximum degree at most Δ, we must show how to compute the first m coefficients of p. We will use $\tilde{O}$-notation throughout to mean that we suppress polynomial factors in m (and k). To reduce notation, let us write $p = p(G)$,

$d = d(G)$ for the degree of p, and $e_i = e_i(G)$ for $i = 0, \ldots, d$ for the coefficients of p (from (1)). We also write $p_k := \zeta_1^{-k} + \cdots + \zeta_d^{-k}$, where $\zeta_1, \ldots, \zeta_d \in \mathbb{C}$ are the roots of the polynomial $p(G)$.

Noting $e_0 = 1$, Proposition 2.6 gives

$$p_k = -ke_k - \sum_{i=1}^{k-1} e_i p_{k-i}, \tag{4}$$

for each $k = 1, \ldots, d$.

By (2), for $i \geq 1$, the e_i can be expressed as linear combinations of induced subgraph counts of graphs with at most αi vertices. Since $p_1 = -e_1$, this implies that the same holds for p_1. By induction, (3), and (4) we have that for each k

$$p_k = \sum_{H \in \mathcal{G}_{\alpha k}} a_{H,k} \mathrm{ind}(H, G), \tag{5}$$

for certain, yet unknown, coefficients $a_{H,k}$.

Since p is multiplicative, the inverse power sums are additive (since the multiset of roots of $p(G_1 \cup G_2)$ is exactly the union of the multisets of the roots of $p(G_1)$ and $p(G_2)$). Thus Lemma 2.5 implies that $a_{H,k} = 0$ if H is not connected. Denote by $\mathcal{C}_i(G)$ the set of connected graphs of order at most i that occur as induced subgraphs in G. Let us assume that G has vertex set $[n]$. Denote by $\mathcal{T}_{\leq \alpha k}(G)$ the list consisting of those sets $S \subseteq [n]$ of size at most αk that induce a connected graph in G. This way we can rewrite (5) as follows:

$$p_k = \sum_{H \in \mathcal{C}_{\alpha k}(G)} a_{H,k} \mathrm{ind}(H, G) = \sum_{S \in \mathcal{T}_{\leq \alpha k}(G)} a_{G[S],k}. \tag{6}$$

The next lemma says that we can compute the coefficients $a_{S,k} := a_{G[S],k}$ efficiently for $k = 1, \ldots, m$.

Lemma 2.7 *There is an $\tilde{O}(n(e\Delta)^{\alpha m}\beta_m 4^{\alpha m})$-time algorithm, which given a BIGCP p (with parameters α and β) and an n-vertex graph G of maximum degree Δ, computes and lists the coefficients $a_{S,k}$ in (6) for all $S \in \mathcal{T}_{\leq \alpha k}(G)$ and all $k = 1, \ldots, m$.*

Proof We assume that the vertex set of G is equal to $[n]$. Using the algorithm of Lemma 2.4, we first compute the list $\mathcal{T}_{\leq \alpha k}$ consisting of all subsets S of $V(G)$ such that $|S| \leq \alpha k$ and $G[S]$ is connected. This takes time bounded by

$$O\left(n(\alpha m)^3 (e\Delta)^{\alpha m}\right) = \tilde{O}(n(e\Delta)^{\alpha m}). \tag{7}$$

(Note that the algorithm in Lemma 2.4 actually computes $\mathcal{T}_{\leq \alpha k}$ when it computes $\mathcal{T}_{\alpha m}$.)

To prove the lemma, let us fix $k \leq m$ and show how to compute the coefficients $a_{S,k}$, assuming that we have already computed and listed the coefficients $a_{S',k'}$ for all

$k' < k$ and $S' \in \mathcal{T}_{\leq \alpha k'}$. Let us fix $S \in \mathcal{T}_{\leq \alpha k}$. Let $H = G[S]$. By (4), it suffices to compute the coefficient of $\mathrm{ind}(H, \cdot)$ in $p_{k-i}e_i$ for $i = 1, \ldots, k$ (where we set $p_0 = 1$). By (2) and (5) we know that

$$p_{k-i}e_i = \sum_{H_1 \in \mathcal{G}_{\alpha(k-i)}} \sum_{H_2 \in \mathcal{G}_{\alpha i}} a_{H_1,(k-i)} \lambda_{H_2,i} \mathrm{ind}(H_1, G) \cdot \mathrm{ind}(H_2, G).$$

So by (3) we know that the coefficient of $\mathrm{ind}(H, \cdot)$ in $p_{k-i}e_i$ is given by

$$\sum_{H_1, H_2} c^H_{H_1,H_2} a_{H_1,(k-i)} \lambda_{H_2,i} = \sum_{(T,U): T \cup U = V(H)} a_{H[T],(k-i)} \lambda_{H[U],i}. \tag{8}$$

As $|V(H)| \leq \alpha k$, the second sum in (8) is over at most $4^{\alpha k} = O(4^{\alpha m})$ pairs (T, U). For each such pair, we need to compute $\lambda_{H[U],i}$ and $a_{H[T],(k-i)}$. We can compute $\lambda_{H[U],i}$ in time bounded by $O(\beta_i) = O(\beta_m)$ since p is a BIGCP. As $H[T] = G[T]$, to compute $a_{H[T],(k-i)}$ we just need to look up the coefficient $a_{T,k-i}$, which takes time $O(k - i)$.

Together, all this implies that the coefficient of $\mathrm{ind}(H, \cdot)$ in $p_{k-i}e_i$ can be computed in time bounded by

$$O(4^{\alpha m}(\beta_m + m)) = \tilde{O}(\beta_m \cdot 4^{\alpha m}). \tag{9}$$

So the coefficient $a_{H,k}$ can be computed in the same time (since we suppress polynomial factors in m). Thus all coefficients $a_{S,k}$ for $S \in \mathcal{T}_{\leq \alpha k}$ can be computed and listed in time bounded by $|\mathcal{T}_{\leq \alpha k}|$ multiplied by the expression (9), which is bounded by

$$\tilde{O}(n(e\Delta)^{\alpha m} \beta_m 4^{\alpha m}) \tag{10}$$

by Lemma 2.3.

So the total running time is bounded by the time to compute the list $\mathcal{T}_{\leq \alpha m}$ (which is given by (7)) plus the time to compute the $a_{S,k}$ for $S \in \mathcal{T}_{\leq \alpha k}$ (which is given by (10)) for $k = 1, \ldots, m$. This proves the lemma. □

To finish the proof of the theorem, we compute p_k for each $k = 1, \ldots, m$ by adding all the numbers $a_{S,k}$ over all $S \in \mathcal{T}_{\leq \alpha k}(G)$ using (6) (these numbers were computed in the previous lemma in time $\tilde{O}((e\Delta)^{\alpha m} \beta_m 4^{\alpha m} n)$). Doing this addition takes time

$$O(m|\mathcal{T}_{\leq \alpha m}(G)|) = \tilde{O}(n(e\Delta)^{\alpha m}).$$

Finally, knowing the p_i, we can inductively compute the e_i for $i = 1, \ldots, m$ using the relations (4), in quadratic time in m. So we see that the total running time for computing $e_1, \ldots, e_m$ is dominated by the computation of the $\alpha_{S,k}$ and is $\tilde{O}(n(e\Delta)^{\alpha m} \beta_m 4^{\alpha m})$. This proves the theorem.

Since our algorithm is spread over several lemmas, we give an overview below.

Algorithm 1 Input: a BIGCP p (with parameters α and β_i), a natural number m and a graph G. Recall that $p(G)(z) := \sum_{i=0}^{d(G)} e_i(G) z^i$ where the coefficients e_i satisfy

$e_i(G) := \sum_{H \in \mathcal{G}_{\alpha i}} \lambda_{H,i} \mathrm{ind}(H, G)$. One inputs p via an algorithm A that can compute $\lambda_{H,i}$ in time β_i.

- Step 1: Use the algorithm of Lemma 2.4 to compute, for $k = 1, \ldots, m$, the collection $\mathcal{T}_k = \{S \subseteq V(G) \mid G[S] \text{ connected } |S| \leq \alpha k\}$.
- Step 2: For each $k = 1 \ldots, m$, and $S \subseteq V(G)$ with $|S| \leq \alpha k$, we iteratively compute the coefficients $a_{S,k}$ using the following recursion

$$a_{S,k} = \begin{cases} -k\lambda_{G[S],k} - \sum_{i=1}^{k-1} \sum_{(U,T):U \cup T = S} a_{T,(k-i)} \lambda_{G[U],i} & \text{if } S \in \mathcal{T}_k; \\ 0 & \text{otherwise,} \end{cases}$$

  as detailed in the proof of Lemma 2.7 using (6) and (8). (Formally, we only compute $a_{S,k}$ for $S \in \mathcal{T}_k$ and implicitly assume $a_{S,k} = 0$ for any $S \subseteq V(G)$ with $|S| \leq \alpha k$ and $S \notin \mathcal{T}_k$.)
- Step 3: For each $k = 1, \ldots, m$ compute $p_k = \sum_{S \in \mathcal{T}_k} a_{S,k}$.
- Step 4: Use the following recursion (i.e. the Newton identities (4) together with the values p_k computed in Step 3 to iteratively compute $e_1(G), \ldots, e_m(G)$:

$$e_i = \frac{1}{k} \left(-p_k - \sum_{i-1}^{k-1} e_i p_{k-1} \right)$$

Output: the m coefficients $e_1(G), \ldots, e_m(G)$.

3 Proof of Theorem 1.1

We first set up some notation before we state our key definition. For a graph H we write $H = i_1 H_1 \cup \cdots \cup i_r H_r$ to mean that H is the disjoint union of i_1 copies of H_1, i_2 copies of H_2 all the way to i_r copies of H_r for connected and pairwise non-isomorphic graphs $H_1, \ldots, H_r$. We write $\underline{H} = (H_1, \ldots, H_r)$. For vectors $\boldsymbol{\mu} = (\mu_1, \ldots, \mu_r)$, $\boldsymbol{\nu} = (\nu_1, \ldots, \nu_r) \in \mathbb{Z}^r_{\geq 0}$ we write $\boldsymbol{\mu} \circ \boldsymbol{\nu}$ for the vector in $\mathbb{Z}^r_{\geq 0}$ that is the pointwise or Hadamard product of $\boldsymbol{\mu}$ and $\boldsymbol{\nu}$. We denote by $\langle \boldsymbol{\mu}, \boldsymbol{\nu} \rangle$ the usual scalar product of $\boldsymbol{\mu}$ and $\boldsymbol{\nu}$. For a vector of variables $\boldsymbol{x} = (x_1, \ldots, x_r)$ and $\boldsymbol{\mu} \in \mathbb{Z}^r_{\geq 0}$ we sometimes write $\boldsymbol{x}^{\boldsymbol{\mu}} := x_1^{\mu_1} \cdots x_r^{\mu_r}$.

For pairwise non-isomorphic and connected graphs $H_1, \ldots, H_r$ we define the multivariate graph polynomial $Z_{\underline{H}}(G) \in \mathbb{Z}[x_1, \ldots, x_r]$ as follows. For a graph G we let

$$Z_{\underline{H}}(G; \boldsymbol{x}) = \sum_{\boldsymbol{\gamma} \in \mathbb{Z}^r_{\geq 0}} \mathrm{ind}(\boldsymbol{\gamma} \underline{H}, G) \boldsymbol{x}^{\boldsymbol{\gamma} \circ \boldsymbol{h}},$$

where $\boldsymbol{x} = (x_1, \ldots, x_r)$, $\boldsymbol{\gamma} = (\gamma_1, \ldots, \gamma_r)$, $\boldsymbol{h} := (|V(H_1)|, \ldots, |V(H_r)|)$ and where $\boldsymbol{\gamma} \underline{H}$ denotes the graph $\gamma_1 H_1 \cup \cdots \cup \gamma_r H_r$. Note that this is a finite polynomial because G is finite and so there are only a finite number of non-zero terms $\mathrm{ind}(\boldsymbol{\gamma} \underline{H}, G)$.

Computing $\mathrm{ind}(H, G)$ for any two graphs $H = i_1 H_1 \cup \cdots \cup i_r H_r$ and G can now be modelled as computing the coefficient of the monomial $x_1^{i_1 h_1} x_2^{i_2 h_2} \cdots x_r^{i_r h_r}$ in $Z_{\underline{H}}(G; \boldsymbol{x})$. Let us start by gathering some facts about the polynomial $Z_{\underline{H}}$.

Proposition 3.1 *The polynomial $Z_{\underline{H}}$ is multiplicative, i.e., for any two graphs G_1 and G_2, $Z_{\underline{H}}(G_1 \cup G_2; \boldsymbol{x}) = Z_{\underline{H}}(G_1; \boldsymbol{x}) \cdot Z_{\underline{H}}(G_2; \boldsymbol{x})$. In particular, any evaluation of $Z_{\underline{H}}$ is also multiplicative.*

Proof Note first that every monomial in $Z_{\underline{H}}(G; \boldsymbol{x})$ is of the form $\boldsymbol{x}^{\boldsymbol{\gamma} \circ \boldsymbol{h}}$ for some unique choice of $\boldsymbol{\gamma}$. For notational convenience we write $s_{\boldsymbol{\gamma}}(G) := \mathrm{ind}(\boldsymbol{\gamma}\underline{H}, G)$. Consider the coefficient of $\boldsymbol{x}^{\boldsymbol{\gamma} \circ \boldsymbol{h}}$ in the polynomial $Z_{\underline{H}}(G_1; \boldsymbol{x}) \cdot Z_{\underline{H}}(G_2; \boldsymbol{x})$. The coefficient is given by

$$\sum_{\boldsymbol{\mu}+\boldsymbol{\nu}=\boldsymbol{\gamma}} s_{\boldsymbol{\mu}}(G_1) s_{\boldsymbol{\nu}}(G_2),$$

(where $\boldsymbol{\mu} + \boldsymbol{\nu}$ denotes the usual vector addition of $\boldsymbol{\mu}$ and $\boldsymbol{\nu}$). This counts precisely the number of copies of $\gamma_1 H_1 \cup \cdots \cup \gamma_r H_r$ in $G_1 \cup G_2$, that is, $s_{\boldsymbol{\gamma}}(G_1 \cup G_2)$, which is the coefficient of $\boldsymbol{x}^{\boldsymbol{\gamma} \circ \boldsymbol{h}}$ in the polynomial $Z_{\underline{H}}(G_1 \cup G_2; \boldsymbol{x})$. □

Suppose $\boldsymbol{\mu} \in \mathbb{Z}^r_{\geq 0}$ and let z be a variable. Define the graph polynomial $Z_{\boldsymbol{\mu}} = Z_{\boldsymbol{\mu},\underline{H}}(G) \in \mathbb{Z}[z]$ by

$$Z_{\boldsymbol{\mu}}(z) = Z_{\underline{H}}(G; (\mu_1 z, \ldots, \mu_r z)) =: \sum_{i \geq 0} s_i(\boldsymbol{\mu}) z^i;$$

here the second equality defines the numbers $s_i(\boldsymbol{\mu}) = s_i(\boldsymbol{\mu})(G)$. Evaluating $Z_{\underline{H}}(G; \boldsymbol{x})$ at $\boldsymbol{x} = (\mu_1 z, \ldots, \mu_r z)$, each monomial $\boldsymbol{x}^{\boldsymbol{\gamma} \circ \boldsymbol{h}}$ will evaluate to $(\mu_1 z)^{\gamma_1 h_1} \cdots (\mu_r z)^{\gamma_r h_r} = \boldsymbol{\mu}^{\boldsymbol{\gamma} \circ \boldsymbol{h}} z^{\langle \boldsymbol{\gamma}, \boldsymbol{h} \rangle}$. Therefore the coefficient of z^i in $Z_{\boldsymbol{\mu}}(z)$ is

$$s_i(\boldsymbol{\mu}) = \sum_{\substack{\boldsymbol{\gamma} \in \mathbb{Z}^r_{\geq 0} \\ \langle \boldsymbol{\gamma}, \boldsymbol{h} \rangle = i}} \boldsymbol{\mu}^{\boldsymbol{\gamma} \circ \boldsymbol{h}} \mathrm{ind}(\boldsymbol{\gamma}\underline{H}, G) = \sum_{\substack{\boldsymbol{\gamma} \in \mathbb{Z}^r_{\geq 0} \\ \langle \boldsymbol{\gamma}, \boldsymbol{h} \rangle = i}} \left(\prod_{j=1}^{r} \mu_j^{\gamma_j h_j} \right) \mathrm{ind}(\boldsymbol{\gamma}\underline{H}, G). \tag{11}$$

Proposition 3.2 *Fix $\underline{H} = (H_1, \ldots, H_r)$ where the H_i are pairwise non-isomorphic connected graphs each of maximum degree at most Δ and fix $\boldsymbol{\mu} \in \mathbb{Z}^r_{\geq 0}$. Then $Z_{\boldsymbol{\mu},\underline{H}}(G; z)$ is a BIGCP with parameters $\alpha = 1$ and $\beta_i = i^2 r \Delta^{i-1}$.*

Proof Since $Z_{\boldsymbol{\mu},\underline{H}}(G)$ is a particular evaluation of $Z_{\underline{H}}(G)$, we know by Proposition 3.1 that it is multiplicative.

The coefficient of z^i in $Z_{\boldsymbol{\mu},\underline{H}}(G; z)$ is given by (11). Since $\boldsymbol{\gamma}\underline{H}$ is a graph with exactly $\langle \boldsymbol{\gamma}, \boldsymbol{h} \rangle = i$ vertices, we can take α to be 1 in the definition of BIGCP.

For a given graph F, we must determine $\lambda_{F,i}$ in the definition of BIGCP and the time β_i required to do this. Note that we may assume $|V(F)| = i$; otherwise $\lambda_{F,i} = 0$. If $|V(F)| = i$, we must test if F is isomorphic to a graph of the form $\boldsymbol{\gamma}\underline{H}$ with $\langle \boldsymbol{\gamma}, \boldsymbol{h} \rangle = i$ and if so we must output the value of $\lambda_{F,i}$ as $\boldsymbol{\mu}^{\boldsymbol{\gamma} \circ \boldsymbol{h}}$ (this last step taking i

arithmetic operations). To test if F is isomorphic to a graph of the form $\boldsymbol{\gamma}\underline{H}$, we test isomorphism of each component of F against each of the graphs $H_1, \ldots, H_r$, which takes time at most $O(i^2 r \Delta^{i-1})$ using Lemma 2.2 at most ir times. Thus the total time to compute $\lambda_{F,i}$ is at most $O(i^2 r \Delta^{i-1})$. □

Now since $Z_{\boldsymbol{\mu},\underline{H}}(G; z)$ is a BIGCP, Theorem 2.1 allows us to compute the coefficients $s_i(\boldsymbol{\mu})$ in (11) with the desired running time. However the $s_i(\boldsymbol{\mu})$ are linear combinations of the numbers $\mathrm{ind}(\boldsymbol{\gamma}\underline{H}, G)$, while we wish to compute one of these numbers in particular, say $\mathrm{ind}(\rho\underline{H}, G)$. By making careful choices of different $\boldsymbol{\mu}$, we will obtain an invertible linear system whose solution will include the number $\mathrm{ind}(\rho\underline{H}, G)$. We will require Alon's Combinatorial Nullstellensatz [1], which we state here for the reader's convenience.

Theorem 3.3 ([1]) *Let $f(x_1, \ldots, x_n)$ be a polynomial of degree d over a field $\mathbb{F}$. Suppose the coefficient of the monomial $x_1^{\mu_1} \cdots x_n^{\mu_n}$ in f is nonzero and $\mu_1 + \cdots + \mu_n = d$. If $S_1, \ldots, S_n$ are finite subsets of $\mathbb{F}$ with $|S_i| \geq \mu_i + 1$ then there exists a point $x \in S_1 \times \cdots \times S_n$ for which $f(x) \neq 0$.*

Given a vector $h \in \mathbb{N}^r$, let us write $\mathcal{P}_{m,r,\boldsymbol{h}}$ for the set of vectors $\boldsymbol{\gamma} \in \mathbb{Z}^r_{\geq 0}$ such that $\langle \boldsymbol{\gamma}, \boldsymbol{h} \rangle = m$. We note that, as the the number of elements in $\mathcal{P}_{m,r,\boldsymbol{h}}$ is at most the number of monomials in r variables of degree m, we have

$$|\mathcal{P}_{m,r,\boldsymbol{h}}| \leq \binom{m+r-1}{r-1}. \tag{12}$$

Enumerate the vectors in $\mathcal{P}_{m,r,\boldsymbol{h}}$ as $\boldsymbol{\gamma}_1, \ldots, \boldsymbol{\gamma}_k$ and write $\gamma_{i,j}$ for the jth component of $\boldsymbol{\gamma}_i$. Given a vector $\boldsymbol{v} \in \mathbb{N}^r$, we write $\boldsymbol{v}^* \in \mathbb{N}^k$ for the vector whose ith component v_i^* is $\boldsymbol{v}^{\boldsymbol{\gamma}_i \circ \boldsymbol{h}}$, i.e.

$$v_i^* = v_1^{\gamma_{i,1} h_1} \cdots v_r^{\gamma_{i,r} h_r}.$$

Lemma 3.4 *Fix $m, r \in \mathbb{N}$ and $\boldsymbol{h} \in \mathbb{N}^r$, and let $\boldsymbol{\gamma}_1, \ldots, \boldsymbol{\gamma}_k$ be an enumeration of the elements in $\mathcal{P}_{m,r,\boldsymbol{h}}$ as before. In time $O(k^5 + k^2 m e^m)$, we can find vectors $\boldsymbol{v}_1, \ldots, \boldsymbol{v}_k \in \mathbb{N}^r$ such that $\boldsymbol{v}_1^*, \ldots, \boldsymbol{v}_k^* \in \mathbb{N}^k$ (as defined above) are linearly independent.*

Proof For any vector $\boldsymbol{v}$, let us write $\boldsymbol{v}|_j$ to denote the vector consisting of the first j components. Suppose we have found vectors $\boldsymbol{v}_1, \ldots, \boldsymbol{v}_{\ell-1} \in \mathbb{N}^r$ such that the $(\ell-1) \times (\ell-1)$ matrix

$$M_{\ell-1} := \left(\boldsymbol{v}_1^*|_{\ell-1}, \ldots, \boldsymbol{v}_{\ell-1}^*|_{\ell-1}\right)$$

has non-zero determinant. We will show how to find $\boldsymbol{v}_\ell \in \mathbb{N}^r$ such that the corresponding matrix M_ℓ has non-zero determinant. First consider the components of $\boldsymbol{v}_\ell$ to be unknown variables $x_1, \ldots, x_r$ so that $\det(M_\ell)$ becomes a polynomial $P = P(x_1, \ldots, x_r)$ in the variables $x_1, \ldots, x_r$. In fact it is a homogeneous polynomial of degree m. Writing $\boldsymbol{x} = (x_1, \ldots, x_r)$, we know that the coefficient of $\boldsymbol{x}^{\boldsymbol{\gamma}_\ell \circ \boldsymbol{h}}$ is $\pm \det(M_{\ell-1}) \neq 0$ (consider the determinant expansion of the matrix M_ℓ along the

ℓth column). We must now find $\boldsymbol{v}_\ell \in \mathbb{N}^r$ such that $\det(M_\ell) = P(v^*_{\ell,1}, \ldots, v^*_{\ell,\ell}) \neq 0$, where $v^*_{\ell,i}$ is the ith component of $\boldsymbol{v}^*_\ell$.

Assume the components of $\boldsymbol{\gamma}_\ell \in \mathbb{N}^r$ are $a_1, \ldots, a_r$. Applying Theorem 3.3 to the monomial $\boldsymbol{x}^{\boldsymbol{\gamma}_\ell \circ \boldsymbol{h}}$ and taking the sets $S_i = \{1, \ldots, a_i h_i + 1\}$ for $i = 1, \ldots, r$, we know there exists a vector $\boldsymbol{v}_\ell \in S := S_1 \times \cdots \times S_r$ such that $P(v^*_{\ell,1}, \ldots, v^*_{\ell,\ell}) \neq 0$. Computing the polynomial P requires time at most $O(k \cdot k^3)$ (using that computing the determinant of an $n \times n$ matrix takes $O(n^3)$ time) and evaluating it at every point in S requires at most $O(m \cdot k \cdot |S|)$ operations. We can bound $|S|$ as follows:

$$|S| = (a_1 h_1 + 1) \cdots (a_r h_r + 1) \leq \left(\frac{1}{r} \sum_{i=1}^{r} (a_i h_i + 1) \right)^r = \left(\frac{m + r}{r} \right)^r$$
$$= \left(1 + \frac{m}{r}\right)^r \leq e^m.$$

The first inequality follows from the arithmetic-geometric mean inequality. Iterating the procedure, we can determine $\boldsymbol{v}_1, \ldots, \boldsymbol{v}_k$ in time $O(k \cdot (k^4 + mk|S|)) \leq O(k^5 + k^2 m e^m)$. □

Remark 3.1 We suspect there should be a simpler argument than the one we have just given (perhaps one where the vectors $\boldsymbol{v}_1, \ldots, \boldsymbol{v}_k$ can be explicitly written down rather than having an algorithm to determine them). Note that one can also use a faster randomised algorithm by applying the Schwarz-Zippel Lemma.

We can now state our algorithm to compute $\text{ind}(H, G)$ for Theorem 1.1.

Algorithm 2 Input: two graphs H and G.

- Step 1: Determine the components of H using e.g. breadth-first search. Compute, using Lemma 2.2 (i), the pairwise nonisomorphic components $H_1, \ldots, H_r$ of H and their multiplicities $i_1, \ldots, i_r$.
- Step 2: Write $H = i_1 H_1 \cup \ldots \cup i_r H_r$ and $\underline{H} = (H_1, \ldots, H_r)$, let $\boldsymbol{h}$ be the vector in $\mathbb{N}^r$ defined by $h_j = |H_j|$ for each j, and compute $m = \sum_{j=1}^r i_j h_j$. Consider the multivariate polynomial $Z_{\underline{H}}(G; \boldsymbol{x})$.
- Step 3: Recall

$$\mathcal{P}_{m,r,\boldsymbol{h}} = \{\boldsymbol{\gamma} \in \mathbb{N}^r : \langle \boldsymbol{\gamma}, \boldsymbol{h} \rangle = m\}.$$

  Enumerate the set $\mathcal{P}_{m,r,\boldsymbol{h}}$ as $\{\boldsymbol{\gamma}_1, \ldots, \boldsymbol{\gamma}_k\}$ with $\boldsymbol{\gamma}_1 = (i_1, \ldots, i_r)$.
- Step 4: Recall that given a vector $\boldsymbol{v} \in \mathbb{N}^r$, we write $\boldsymbol{v}^* \in \mathbb{N}^k$ for the vector $(\boldsymbol{v}^{\boldsymbol{\gamma}_i \circ \boldsymbol{h}})_{i=1}^k$. Use Lemma 3.4 to determine vectors $\boldsymbol{v}_1, \ldots, \boldsymbol{v}_k$ such that the vectors $\boldsymbol{v}_1^*, \ldots, \boldsymbol{v}_k^*$ are linearly independent.
- Step 5: For each $i = 1, \ldots, k$ compute the mth coefficient $s_m(\boldsymbol{v}_i)$ of the univariate polynomial $Z_{\underline{H},\boldsymbol{v}_i}$ using Algorithm 1. (Here $Z_{\underline{H},\boldsymbol{v}_i}(z)$ is the evaluation of $Z_{\underline{H}}(\boldsymbol{x})$ at the vector $\boldsymbol{x} = z\boldsymbol{v}_i$, where z is a scalar variable and this univariate satisfies the properties of a BIGCP by Proposition 3.2.)

- Step 6: Invert the system of linear equations

$$\langle \boldsymbol{v}_i^*, \boldsymbol{s} \rangle = s_m(\boldsymbol{v}_i) \quad \text{for } i = 1, \ldots, k,$$

to find $\boldsymbol{s} \in \mathbb{N}^k$ (with components $s_1, \ldots, s_k$).

Output: $s_1 = \mathrm{ind}(H, G)$.

Proof of Theorem 1.1 Step 1 can be executed in time $O(m^3 \Delta^m)$ by Lemma 2.2 (i) to test for isomorphism. Step 3, the enumeration of the elements of $\mathcal{P}_{m,r,\boldsymbol{h}}$, can be executed in time $O(2^{2m})$, as the size of $\mathcal{P}_{m,r,\boldsymbol{h}}$ is bounded by $\binom{m+r-1}{r-1} = O(2^{m+r}) = O(2^{2m})$.

By Lemma 3.4, we can find the vectors $\boldsymbol{v}_1, \ldots, \boldsymbol{v}_k \in \mathbb{N}^r$ such that the vectors $\boldsymbol{v}_1^*, \ldots, \boldsymbol{v}_k^* \in \mathbb{N}^k$ are linearly independent in time $O(k^5 + k^2 m e^m) = \tilde{O}(2^{10m})$, noting that by (12) k is at most $\binom{m+r-1}{r-1} = O(2^{m+r}) = O(2^{2m})$. So step 4 can be executed in $\tilde{O}(2^{10m})$ time.

By Proposition 3.2 and Theorem 2.1 we can compute the coefficient $s_m(\boldsymbol{v}_i)$ of z^m in $Z_{\underline{H},\boldsymbol{v}_i}(G)(z)$ in time $\tilde{O}((e\Delta)^{\alpha m} \beta_m 4^{\alpha m} n)$ with $\alpha = 1$ and $\beta_j = j^2 r \Delta^{j-1}$. So Step 5, i.e. computing all these coefficients for $i = 1, \ldots, k$ takes time

$$\tilde{O}(k \cdot ((4\Delta)^m)(e\Delta)^m n) = \tilde{O}((7\Delta)^{2m} n).$$

Recall that this coefficient is given by

$$s_m(\boldsymbol{v}_i)(G) = \sum_{j=1}^{k} \boldsymbol{v}_i^{\boldsymbol{\gamma}_j \circ \boldsymbol{h}} \mathrm{ind}(\boldsymbol{\gamma}_j \underline{H}, G) = \sum_{j=1}^{k} \boldsymbol{v}_i^* \mathrm{ind}(\boldsymbol{\gamma}_j \underline{H}, G).$$

More conveniently, writing $\boldsymbol{s} \in \mathbb{Z}_{\geq 0}^k$ for the vector whose jth component is $s_j = \mathrm{ind}(\boldsymbol{\gamma}_j \underline{H}, G)$, we have the invertible system of linear equations given by

$$\langle \boldsymbol{v}_i^*, \boldsymbol{s} \rangle = s_m(\boldsymbol{v}_i) \quad \text{for } i = 1, \ldots, k,$$

where we have computed the values of $s_m(\boldsymbol{v}_i)$ and $\boldsymbol{v}_i^*$, while the vector $\boldsymbol{s}$ is unknown (the system is invertible because we chose the $\boldsymbol{v}_i^*$ to be linearly independent). We can then invert the system in time $O(k^3) = \tilde{O}(2^{6m})$ (Step 6). In particular finding the value of $s_1 = \mathrm{ind}(H, G)$ takes time $\tilde{O}(2^{6m})$. This proves correctness of the algorithm.

The total running time is dominated by the time to execute Step 4 and 5, which is bounded by $\tilde{O}(n(7\Delta)^{2m} + 2^{10m}))$. □

4 Concluding Remarks

As we remarked in the introduction our approach also works in the setting of vertex- and edge-coloured graphs. We will not elaborate on the details here, but just refer the interested reader to Section 3.3 of [18] where we have briefly explained how to extend the results for computing coefficients of BIGCPs to the setting of coloured graphs.

In addition we note that the part of the proof given in Sect. 3 also carries over to the coloured graphs setting replacing graph by coloured graph everywhere.

The approach used to prove Theorem 1.1 is very robust. Besides extending to the coloured setting, it also easily extends to other graph like structures. For example, in [18] it has been extended to *fragments*, i.e., vertex-coloured graphs in which some edges may be unfinished, Liu et al. [14] extended it to *insects*, i.e., vertex-coloured hypergraphs in which some edges may be unfinished and very recently, Barvinok and the second author [4] applied this approach to enumerate integer points in certain polytopes. We expect our approach to be applicable to the problem of counting (induced) substructures in other structures as well, as long as there is a notion of connectedness and maximum degree.

Acknowledgements We thank John Lapinskas for raising a question about the complexity of our main algorithm in a previous version of this paper, which led to an improved running time. We also thank Radu Curticapean for informing us of some historical context to our result. We are grateful for the excellent comments by the anonymous referees leading to an improved presentation of our result.

References

1. Alon, N., Tarsi, M.: Combinatorics probability and computing. Combinatorial nullstellensatz **8**(1), 7–30 (1999)
2. Arvind, V., Raman, V.: Approximation algorithms for some parameterized counting problems. In: Algorithms and Computation, 13th International Symposium, ISAAC 2002 Vancouver, BC, Canada, 21–23 Nov 2002, Proceedings, pp. 453–464 (2002)
3. Barvinok, A.: Combinatorics and Complexity of Partition Functions, volume 30 of Algorithms and Combinatorics. Springer, Berlin (2017)
4. Barvinok, A., Regts, G.: Weighted counting of non-negative integer points in a subspace (2017). arXiv preprint arXiv:1706.05423
5. Borgs, C., Chayes, J., Kahn, J., Lovász, L.: Left and right convergence of graphs with bounded degree. Random Struct. Algorithms **42**(1), 1–28 (2013)
6. Chen, Y., Thurley, M., Weyer, M.: Understanding the complexity of induced subgraph isomorphisms. In: Automata, Languages and Programming, 35th International Colloquium, ICALP 2008, Reykjavik, Iceland, 7–11 July 2008, Proceedings, Part I: Tack A: Algorithms, Automata, Complexity, and Games, pp. 587–596 (2008)
7. Csikvári, P., Frenkel, P.E.: Benjamini–Schramm continuity of root moments of graph polynomials. Eur. J. Comb. **52**, 302–320 (2016)
8. Curticapean, R., Dell, H., Fomin, F.V., Goldberg, L.A., Lapinskas, J.: A fixed-parameter perspective on #bis. In: 12th International Symposium on Parameterized and Exact Computation, IPEC 2017, 6–8 Sept 2017, Vienna, Austria, pp. 13:1–13:13 (2017)
9. Curticapean, R., Dell, H., Marx, D.: Homomorphisms are a good basis for counting small subgraphs. In: Proceedings of the 49th Annual ACM SIGACT Symposium on Theory of Computing, STOC 2017, Montreal, QC, Canada, 19–23 June 2017, pp. 210–223 (2017)
10. 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)
11. Downey, R.G., Fellows, M.R.: Fixed-parameter tractability and completeness II: on completeness for W[1]. Theor. Comput. Sci. **141**(1&2), 109–131 (1995)

12. Flum, J., Grohe, M.: The parameterized complexity of counting problems. SIAM J. Comput. **33**(4), 892–922 (2004)
13. Johnson, D.S., Szegedy, M.: What are the least tractable instances of max independent set? In: Proceedings of the Tenth Annual ACM-SIAM Symposium on Discrete Algorithms, 17–19 Jan 1999, Baltimore, Maryland, pp. 927–928 (1999)
14. Liu, J., Sinclair, A., Srivastava, P.: The Ising partition function: Zeros and deterministic approximation. In: 58th IEEE Annual Symposium on Foundations of Computer Science, FOCS 2017, Berkeley, CA, USA, 15–17 Oct 2017, pp. 986–997 (2017)
15. Lovász, L.: Large Networks and Graph Limits, volume 60 of American Mathematical Society Colloquium Publications. American Mathematical Society, Providence (2012)
16. Nederlof, J.: Personal communication
17. Nešetřil, J., Poljak, S.: On the complexity of the subgraph problem. Comment. Math. Univ. Carolin. **26**(2), 415–419 (1985)
18. Patel, V., Regts, G.: Deterministic polynomial-time approximation algorithms for partition functions and graph polynomials. SIAM J. Comput. **46**(6), 1893–1919 (2017)
19. Scott, A.D., Sokal, A.D.: The repulsive lattice gas, the independent-set polynomial, and the Lovász local lemma. J. Stat. Phys. **118**(5), 1151–1261 (2005)