

Finding a Maximum Independent Set in a Sparse Random Graph

Uriel Feige Eran Ofek

Department of Computer Science and Applied Mathematics
Weizmann Institute, Rehovot 76100, Israel
{uriel.feige, eran.ofek}@weizmann.ac.il

April 18, 2005

Abstract

We consider the problem of finding a maximum independent set in a random graph. The random graph G is modelled as follows. Every edge is included independently with probability $\frac{d}{n}$, where d is some sufficiently large constant. Thereafter, for some constant α , a subset I of αn vertices is chosen at random, and all edges within this subset are removed. In this model, the planted independent set I is a good approximation for the maximum independent set I_{max} , but both $I \setminus I_{max}$ and $I_{max} \setminus I$ are likely to be nonempty. We present a polynomial time algorithm that with high probability (over the random choice of random graph G , and without being given the planted independent set I) finds the maximum independent set in G when $\alpha \geq \sqrt{\frac{c_0}{d}}$, where c_0 is some sufficiently large constant independent of d .

1 Introduction

Let $G = (V, E)$ be a graph. An independent set I is a subset of vertices which contains no edges. The problem of finding a maximum size independent set in a graph is a fundamental problem in Computer Science and it was among the first problems shown to be NP-hard [15]. Moreover, Hastad shows [13] that for any $\epsilon > 0$ there is no $n^{1-\epsilon}$ approximation algorithm for the maximum independent set problem unless NP=ZPP. The best approximation ratio currently known for maximum independent set [6] is $O(n(\log \log n)^2/(\log n)^3)$.

In light of the above mentioned negative results, we may try to design a heuristic which performs well on typical instances. Karp [16] proposed trying to find a maximum independent set in a random graph. However, even this problem appears to be beyond the capabilities of current algorithms. For example let $G_{n,1/2}$ denote the random graph on n vertices obtained by choosing randomly and independently each possible edge with probability $1/2$. A random $G_{n,1/2}$ graph has almost surely maximum independent set of size $2(1 + o(1)) \log_2 n$. A simple greedy algorithm

almost surely finds an independent set of size $\log_2 n$ [12]. However, there is no known polynomial time algorithm which almost surely finds an independent set of size $(1 + \epsilon) \log_2 n$ (for any $\epsilon > 0$).

To further simplify the problem, Jerrum [14] and Kucera [17] proposed a *planted model* $G_{n,1/2,k}$ in which a random graph $G_{n,1/2}$ is chosen and then a clique of size k is randomly placed in the graph. (A clique in a graph G is an independent set in the edge complement of G , and hence all algorithmic results that apply to one of the problems apply to the other.) Alon Krivelevich and Sudakov [2] gave an algorithm based on spectral techniques that almost surely finds the planted clique for $k = \Omega(\sqrt{n})$. More generally, one may extend the range of parameters of the above model by planting an independent set in $G_{n,p}$, where p need not be equal to $1/2$, and may also depend on n . The $G_{n,p,\alpha}$ model is as follows: n vertices are partitioned at random into two sets of vertices, I of size αn and C of size $(1 - \alpha)n$. No edges are placed within the set I , thus making it an independent set. Every other possible edge (with at least one endpoint not in I) is added independently at random with probability p . The goal of the algorithm, given the input G (but without being given the partition into I and C) is to find a maximum independent set. Intuitively, as α becomes smaller the size of the planted independent is closer to the probable size of the maximum independent set in $G_{n,p}$ and the problem becomes harder.

We consider values of p as small as d/n where d is a large enough constant. A difficulty which arises in this sparse regime (e.g. when d is constant) is that the planted independent set I is not likely to be a maximum independent set. Moreover, with high probability I is not contained in a maximum independent set of G . For example, there are expected to be $e^{-d}n$ vertices in C of degree one. It is very likely that two (or more) such vertices $v, w \in C$ will have the same neighbor, and that it will be some vertex $u \in I$. This implies that every maximum independent set will contain v, w and not u , and thus I contains vertices that are not contained in the maximum independent set.

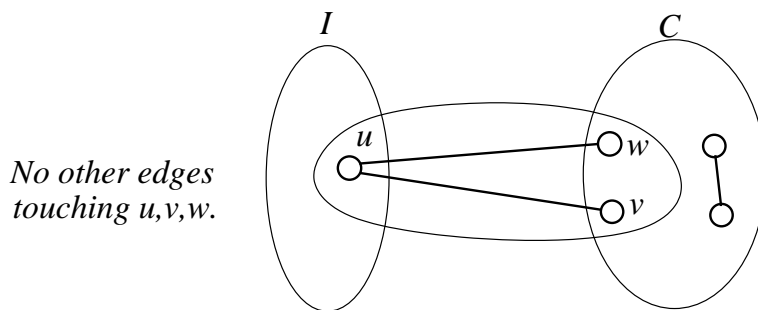


Figure 1: I is not contained in a maximum independent set.

A similar argument shows that there are expected to be $e^{-\Omega(d)}n$ isolated edges. This implies that there will be an exponential number of maximum independent sets.

1.1 Our result

We give a polynomial time algorithm that searches for a maximum independent set of G . Given a random instance of $G_{n, \frac{d}{n}, \alpha}$, the algorithm almost surely succeeds, when $d > d_0$ and $\alpha \geq \sqrt{c_0/d}$

(d_0, c_0 are some universal constants). The parameter d can be also an arbitrary increasing function of n .

1.2 Related work

For $p = 1/2$, $\alpha = \Omega(1/\sqrt{n})$, Alon Krivelevich and Sudakov [2] gave an efficient spectral algorithm which almost surely finds the planted independent set. For the above mentioned parameters the planted independent set is likely to be the unique maximum independent set.

A few papers deal with *semi-random* models which extend the planted model by enabling a mixture of random and adversarial decisions. Feige and Kilian [7] considered the following model: a random $G_{n,p,\alpha}$ graph is chosen, then an adversary may add arbitrarily many edges between I and C , and make arbitrary changes (adding or removing edges) inside C . For any constant $\alpha > 0$ they give a heuristic that almost surely outputs a list of independent sets containing the planted independent set, whenever $p > (1 + \epsilon) \ln n / \alpha n$ (for any $\epsilon > 0$). The planted independent set may not be the only independent set of size αn since the adversary has full control on the edges inside C . Possibly, this makes the task of finding the planted independent set harder. In [8] Feige and Krauthgamer considered a less adversarial semi-random model in which an adversary is allowed to add edges to a random $G_{n, \frac{1}{2}, \frac{1}{\sqrt{n}}}$ graph. Their algorithm almost surely extracts the planted independent set and certifies its optimality. Heuristics for optimization problems different than max independent set will be discussed in the following section.

1.2.1 Technique and outline of the algorithm

Our algorithm builds on ideas from the algorithm of Alon and Kahale [1], which was used for recovering a planted 3-coloring in a random graph. The algorithm we propose has the following 4 phases:

1. Get a coarse approximation I', C' of I, C with $|C \Delta C'| + |I \Delta I'| < 0.041|I|$.
2. Improve the approximation by performing some "error correction". The error term $|C \Delta C'| + |I \Delta I'|$ is reduced to at most n/d^c (c is some big enough constant).
3. Remove vertices of I', C' which have non typical degrees to a new set OUT .

Stop when I', C' become *promising*: I' is an independent set, every vertex of C' has at least 4 edges to I' and no vertex of I' has edges to OUT . Using the fact that sparse random graphs (almost surely) have no small dense sets, it can be shown that $I' \subseteq I_{max}$.

4. Extend the independent set I' optimally using the vertices of OUT . This is done by finding a maximum independent set among the vertices of OUT and adding it to I' . The structure of OUT will be easy enough so that a maximum independent set can be efficiently found (OUT is a random graph of size $n/poly(d)$ with each edge chosen with probability d/n). Notice however, that the set OUT depends on the graph itself thus we can not argue that it is a random $G_{\frac{n}{poly(d)}, \frac{d}{n}}$ graph.

The technique of [1] was implemented successfully on various problems in the planted model: planted hypergraph coloring, planted 3-SAT, planted 4-NAE, min-bisection (by Chen and Frieze [4], Flaxman [10], Goerdt and Lanka [11], Coja-Oghlan [5] respectively).

Perhaps the work closest in nature to the work in the current paper is that of Amin Coja-Oghlan [5] on finding a bisection in a sparse random graph. Both in our work and in that of [5], one is dealing with an optimization problem, and the density of the input graph is such that the planted solution is not an optimal solution. The algorithm for bisection in [5] is based on spectral techniques, and has the advantage that it provides a certificate showing that the solution that it finds is indeed optimal. We do not address the issue of certification in this paper. In [5] the random instance is generated as follows. The vertices of the graph are partitioned into two classes of equal size randomly. Then the edges are inserted: edges inside the two classes with probability p' and edges crossing the partition with probability p independently. Intuitively, as $p' - p$ becomes smaller, the problem becomes harder. Denote by $d_1 = np'/2, d_2 = np/2$ the expected degree of a vertex into its own class and into the other class respectively. The algorithm in [5] is proven to succeed (almost surely) whenever $d_1 - d_2 \geq \sqrt{c_0 d_1 \log d_1}$. In our independent set model the problem becomes harder as αd becomes smaller. If we denote by $\tilde{d}_1 = d, \tilde{d}_2 = (1 - \alpha)d$ the expected degrees of a vertex in C, I respectively then our algorithm (almost surely) succeeds whenever $\tilde{d}_1 - \tilde{d}_2 = \alpha \tilde{d}_1 \geq \sqrt{c_0 \tilde{d}_1}$.

An important difference between planted models for independent set and those for other problems such as 3-coloring and min-bisection is that in our case the planted classes I, C are not symmetric. The lack of symmetry between I and C makes some of the ideas used for the more symmetric problems insufficient. In the approach of [1], a vertex is removed from its current color class and placed in *OUT* if its degree into some other current color class is very different than what one would typically expect to see between the two color classes. This procedure is shown to "clean" every color class C from all vertices that should have been from a different color class, but were wrongly assigned to class C in phase 2 of the algorithm. (The argument proving this goes as follows. Every vertex remaining in the wrong color class by the end of phase 3 must have many neighbors that are wrongly assigned themselves. Thus the set of wrongly assigned vertices induces a small subgraph with large edge density. But G itself does not have any such subgraphs, and hence by the end of phase 3 it must be the case that all wrongly assigned vertices were moved into *OUT*.) It turns out that this approach works well when classes are of similar nature (such as color classes, or two sides of a bisection), but does not seem to suffice in our case where I' is supposed to be an independent set whereas C' is not. Specifically, the set I' might still contain wrongly assigned vertices, and might not be a subset of a maximum independent set in the graph. Under these circumstances, phase 4 will not result in a maximum independent set. Our solution to this problem involves the following aspects, not present in previous work. In phase 3 we remove from I' every vertex that has even one edge connecting it to *OUT*. This adds more vertices to *OUT* and may possibly create large connected components in *OUT*. Indeed, we do not show that *OUT* has no large connected components, which is a key ingredient in previous approaches. Instead, we analyze the 2-core of *OUT* and show that the 2-core has no large components. Then, in phase 4, we use dynamic programming to find a maximum independent set in *OUT*, and use the special structure of *OUT* to show that the algorithm runs in polynomial time.

1.3 Notation

Let $G = (V, E)$ and let $U \subset V$. The subgraph of G induced by the vertices of U is denoted by $G[U]$. When the set of edges used is clear from the context, we will use $\deg(v)_U$ to denote the degree of a vertex v into a set $U \subset V$. To specify exactly the set of edges used, we use $\deg^E(v)_U$ which is the degree of a vertex v into a set U induced by the set of edges E . We use $\Gamma(U)$ to denote the vertex neighborhood of $U \subset V$ (excluding U). We use the graph vertices to index the eigenvectors v_1, \dots, v_n of the adjacency matrix of G . For example we will use $v_1(i)$ to denote the coordinate i that corresponds to vertex i . The parameter d (specifying the expected degree in the random graph G) is assumed to be sufficiently large, and some of the inequalities that we shall derive implicitly use this assumption, without stating it explicitly.

2 The Algorithm

Algorithm *FindIS*(V, E)

1. Let A' be the adjacency matrix of the graph induced by removing from G all vertices of degree $> 5d$. Compute the eigenvector of the most negative eigenvalue of A' denoted by $v_{n'}$. Set I_1 to contain the αn vertices with largest absolute value in $v_{n'}$. Set $C_1 = V \setminus I_1$.
2. Set $C_2^0 = C_1, I_2^0 = I_1$. Iterate $j = 1, 2, \dots, \log n$:
for every vertex v : if $\deg(v)_{I_2^{j-1}} < \alpha d/2$ then $v \in I_2^j$, otherwise $v \in C_2^j$.
3. (a) Set $I_3 = I_2^{\log n}, C_3 = C_2^{\log n}, OUT_3 = \emptyset$.
(b) For every edge (u, v) such that both u, v are in I_3 , move u, v to OUT_3 .
(c) A vertex $v \in C_3$ is *removable* if $\deg(v)_{I_3} < 4$.
Iteratively: find a removable vertex v and move it from C_3 to OUT_3 . If v has neighbors in I_3 , move these neighbors from I_3 to OUT_3 .
4. Find a maximum independent set in $G[OUT_3]$ (this will be shown to be doable in polynomial time, see Corollary 3.4). Output the union of this independent set and I_3 .

Figure 2 depicts the situation after step 3 of the algorithm. At that point, I_3 is an independent set, there are no edges between I_3 and OUT_3 , and every vertex $v \in C_3$ has at least four neighbors in I_3 .

3 Correctness

Let I_{max} be a maximum independent set of G . We establish two claims. Claim 3.1 guarantees the correctness of the algorithm and Claim 3.3 guarantees its efficient running time. Here we present these two claims, and their proofs are deferred to later sections.

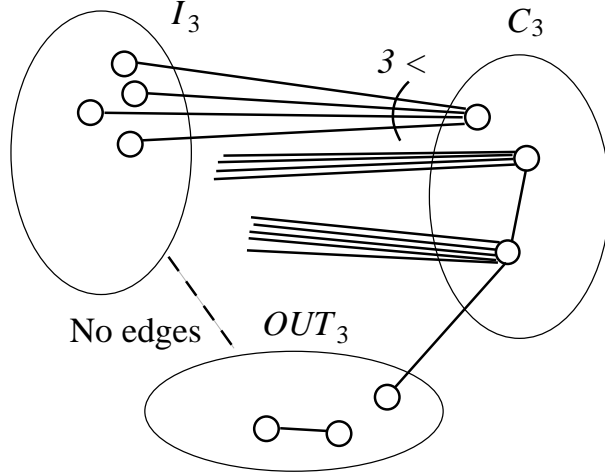


Figure 2: *FindIS* Step 3 outcome

Claim 3.1. *With high probability there exists I_{max} such that $I_3 \subseteq I_{max}, C_3 \cap I_{max} = \emptyset$.*

Definition 3.2. The 2-core of a graph G is the maximal subgraph in which the minimal degree is 2.

It is easy to see that the 2-core is unique and can be found by iteratively removing vertices whose degree is smaller than 2.

Claim 3.3. *With high probability the largest connected component in the 2-core of $G[OUT_3]$ has cardinality of at most $3 \log n$.*

Let G be any graph. The subgraph induced on those vertices of G that do not belong to the 2-core is a forest. Each tree of this forest is either disconnected from the 2-core or it is connected by exactly one edge to the 2-core. To find a maximum independent set of $G[OUT_3]$ we need to find a maximum independent set in each connected component of $G[OUT_3]$ separately. For each connected component D_i of $G[OUT_3]$ we find the maximum independent set as follows: let C_i be the intersection of D_i with the 2-core of G . We enumerate all possible independent sets in C_i (there are at most $2^{|C_i|}$ possibilities), each one of them can be optimally extended to an independent set of D_i by solving (separately) a maximum independent set problem on each of the trees connected to C_i . For some trees we may have to exclude the tree vertex which is connected to D_i if it is connected to a vertex of the independent set that we try to extend. On each tree the problem can be solved by dynamic programming.

Corollary 3.4. *A maximum independent set in OUT_3 can be found in polynomial time.*

3.1 Dense Sets and Degree Deviations

In proving the correctness of the algorithm, we will use structural properties of the random graph G . In particular, such a random graph most likely has no small dense sets (small sets of vertices that

induce many edges). This fact will be used on several occasions to derive a proof by contradiction. Namely, certain undesirable outcomes of the algorithm cannot occur, as otherwise they will lead to a discovery of a small dense set in G . The following lemmas are standard. We state them here and give their proof in Section A for completeness.

Lemma 3.5. *Let G be a random $G_{n,p}$ ($p = \frac{d}{n}$, $d < n^{1/40}$) random graph.*

1. *With probability at least $1 - d^2/n$ for every set $U \subset V$ of cardinality smaller than n/d^5 the number of edges inside U is bounded by $\frac{4}{3}|U|$.*
2. *Let $c \geq 3$. With probability $> 1 - n^{-0.9(c-1)}$ for every set of vertices U of size smaller than n/d^5 the number of edges inside U is less than $c|U|$.*
3. *With probability $> 1 - n^{-\sqrt{d}}$ there is no $U \subset V$ of size $< 0.041\alpha n$ containing $\alpha d/12$ edges.*
4. *With probability $> 1 - 1/n$ there is no set U of size $< \frac{10n \log d}{d}$ containing $50 \log d |U|$ edges.*

Given two small enough disjoint sets of vertices A, B , if every vertex of B has at least 2 edges going to A then $|A|$ cannot be too small relative to $|B|$. This is true as otherwise $|A \cup B|$ would contain too many internal edges with contradiction to part 1 of Lemma 3.5.

Corollary 3.6. *Let G be a graph which has the property from Lemma 3.5 part 1. Let A, B be any two disjoint sets of vertices each of size smaller than n/d^5 . If every vertex of B has at least 2 edges going into A , then $|A| \geq |B|/2$.*

Corollary 3.7. *With probability of at least $1 - 1/n$ there is no $C' \subseteq C$ such that $n/2d^5 \leq |C'| \leq \frac{n \log d}{d}$ and $|\Gamma(C') \cap I| \leq |C'|$.*

The following lemma bounds the number of vertices whose degree largely deviates from its expectation.

Lemma 3.8. *Let $d < n^{1/40}$.*

1. *With probability of at least $1 - e^{-n^{0.4}}$, the number of vertices from C whose degree into I is $< 0.9\alpha d$ is at most n/d^{21} .*
2. *With probability of $1 - e^{-n^{0.2+o(1)}}$, the number of edges that contain a vertex with degree at least $3d$ is at most $3e^{-d}dn$.*

3.2 Proof of Claim 3.1

We will use the assumption that $d < n^{1/40}$ which is needed for some concentration results. When $d > n^{1/40}$ the planted independent set is almost surely the maximum unique independent set and it can be found using only (a variant of) the first step of the algorithm (details are omitted). For convenience, when analyzing the first step we will further assume that $\alpha < \sqrt{100 \log d/d}$ (when $\alpha > \sqrt{100 \log d/d}$ the eigenvector computation performed in step 1 can be avoided: a coarse approximation of I, C can be derived using only the vertices degrees, by putting in I_1 all the vertices of degrees $< (1 - \alpha/2)d$ and the rest in C_1).

We first show that steps 1, 2, 3 give a very good approximation of I, C .

Lemma 3.9. Let $\sqrt{\frac{c_0}{d}} \leq \alpha \leq \sqrt{\frac{100 \log d}{d}}$. Almost surely $|I_1 \Delta I| < 0.041|I|$.

The proof of Lemma 3.9 is deferred to Section B. The approximation I_1, C_1 serves as a bootstrap for the "error correction" done in step 2. We will show that step 2 significantly reduces the error term i.e. $|I_2 \Delta I| < n/d^{20}$ ($< |I|/d^{19}$).

Lemma 3.10. With probability $> 1 - 2n^{-\sqrt{d}}$, it holds that $|I_2 \Delta I| < n/d^{20}$.

Proof. The idea of the proof is as follows. There exists a set V_2 of size $> (1 - 1/d^{20})n$ with the following property: every iteration of step 3 of *FindIS* reduces the number of errors (with respect to I, C) in V_2 by a factor of 2. It then follows that after step 3 is done, all the vertices of V_2 are assigned correctly. Define the set V_2 as follows.

Initialization: $V_2 = I \cup \{v \in C \mid \deg(v)_I \geq 0.9\alpha d\}$.

Iteratively: (i) if there is $v \in C \cap V_2$ with $\deg(v)_{I \cap V_2} < 0.8\alpha d$, remove v from V_2 .
(ii) if there is $v \in I \cap V_2$ with $\deg(v)_{V \setminus V_2} > \alpha d/4$ remove v from V_2 .

When the process ends, every vertex of $C \cap V_2$ has at least $0.8\alpha d$ edges to $I \cap V_2$ and every vertex of $I \cap V_2$ has at most $\alpha d/4$ edges to $V \setminus V_2$.

We will first show that that each iteration of step 2 of *FindIS* reduces the number of errors in V_2 by a factor of at least 2. Let E_i be the set of wrongly assigned vertices of V_2 in iteration i of step 2 ($E_i = V_2 \cap (I_2^i \Delta I)$). We will show that every vertex of E_i has at least $\alpha d/4$ edges to E_{i-1} , thus if $|E_i| > |E_{i-1}|/2$ the set $|E_i \cup E_{i-1}|$ is dense (its average degree above $\alpha d/6$). This event has probability of at most $1 - n^{-\sqrt{d}}$ by Lemma 3.5 part 3 (we can use the lemma since $|E_i|$ is decreasing and $|E_0| < 0.041|I|$).

case 1: $v \in E_{i-1} \cap E_i$ (either $v \in I \cap C_2^{i-1}$ or $v \in C \cap I_2^{i-1}$):

If $v \in I \cap C_2^{i-1}$ (and $v \in E_i$) then in round $i - 1$ it has at least $\alpha d/2$ neighbors in I_2^{i-1} , these neighbors are in $C \cap I_2^{i-1}$ since $v \in I$. At least $\alpha d/4$ of these neighbors are in V_2 since v has at most $\alpha d/4$ edges to $V \setminus V_2$ (because $v \in I \cap V_2$). Thus v has $> \alpha d/4$ neighbors in E_{i-1} . If $v \in C \cap I_2^{i-1}$ (and $v \in E_i$) then in round $i - 1$ it has at most $\alpha d/2$ neighbors in I_2^{i-1} . Since $v \in V_2 \cap C$ it has $0.8\alpha d$ neighbors in $I \cap V_2$, thus at least $0.3\alpha d$ of them are in $I \cap C_2^{i-1} \subseteq E_{i-1}$.

case 2: $v \in E_i \setminus E_{i-1}$:

If $v \in E_i \setminus E_{i-1} \cap I$ then v was moved from I_2^{i-1} to C_2^i , therefore it has at least $\alpha d/2$ neighbors in $I_2^{i-1} \cap C$. Among them at least $\alpha d/4$ belong to V_2 because v has at most $\alpha d/4$ edges to $V \setminus V_2$. If $v \in E_i \setminus E_{i-1} \cap C$ then v was moved from C_2^{i-1} to I_2^i , therefore it has at most $\alpha d/2$ neighbors in I_2^{i-1} . Since $v \in V_2 \cap C$ it has at least $0.8\alpha d$ neighbors in $I \cap V_2$, among which at least $0.3\alpha d$ are in $I \cap C_2^{i-1} \subseteq E_{i-1}$.

We will now prove that $|V \setminus V_2| < n/d^{20}$. After setting $V_2 = I \cup \{v \in C \mid \deg(v)_I \geq 0.9\alpha d\}$ (and before the iteration process) it holds that $|V \setminus V_2| < n/d^{21}$ with probability $1 - e^{-n^{0.4}}$ (see Lemma 3.8 part 1). In the iteration process, every vertex that we remove from V_2 contributes at least $0.1\alpha d$ edges to $V \setminus V_2$. If the iteration steps are repeated too many times, $V \setminus V_2$ will become dense. Assume by contradiction that at some point the set $V \setminus V_2$ doubled its size (when we compare it to the size before the first iteration). At this point it contains at least $\frac{1}{2}|V \setminus V_2|0.1\alpha d$ edges and its size is at most $2n/d^{21}$. By Lemma 3 part 2 this happens with probability $< n^{-0.9(0.1\alpha d/2-1)} < n^{-\sqrt{d}}$. \square

So far we have shown that at most n/d^{20} vertices of I_2, C_2 are wrongly assigned (with respect to I, C). The goal of step 3 is to "clean" I_2, C_2 yielding I_3, C_3 that can be extended into an optimal solution. Before showing that $I_3 \subseteq I_{max}$ (for some maximum independent set I_{max}) we show that the process of "cleaning" in step 3 does not move too many vertices are moved to OUT_3 . This will be used later for proving that $I_3 \subseteq I_{max}$.

Lemma 3.11. *With probability $> 1 - 3n^{-\sqrt{d}}$, it holds that $|OUT_3| \leq n/d^{18}$.*

Proof. The idea of the proof is the following: there exists a set $V_3 \subseteq V_2$ (the set V_2 is defined in Lemma 3.10) of size $> (1 - 1/d^{18})n$ such that $V_3 \cap OUT_3 = \emptyset$. Define V_3 as follows.

Initialization: $V_3 = V_2$,

remove from V_3 all the vertices of $V_3 \cap I$ that have edges to $V \setminus V_3$.

Iteratively: find a vertex $v \in V_3 \cap C$ with $\deg(v)_{V_3 \cap I} < 4$, remove v and its neighbors in I from V_3 .

We now prove that $V_3 \subseteq I_3 \cup C_3$ (after step 3a this is certainly true, we will show it is kept during steps 3b, 3c). Initially $V_3 = V_2$. Removing from V_3 all the vertices of $V_3 \cap I$ that have edges to $V \setminus V_3$ ensures that there are no edges between vertices of $V_3 \cap I$ and vertices which were assigned incorrectly. Thus, step 3b of the algorithm does not touch any vertex of V_3 as it removes only edges that contain at least one wrongly assigned endpoint. Finally, the iteration process in the definition of V_3 ensures that every vertex of $V_3 \cap C$ has at least 4 edges to vertices in I . Since V_3 is a subset of $I_3 \cup C_3$ at the beginning of step 3c and there are no wrongly assigned vertices in V_3 , during step 3c there will never be a vertex of $V_3 \cap C$ that has fewer than 4 edges to vertices of $V_3 \cap I$. We conclude that $V_3 \subseteq I_3 \cup C_3$ at the end of step 3.

It remains to bound from above the size of $V \setminus V_3$. Initially $V_3 = V_2$, at this point $|V \setminus V_3| \leq n/d^{20}$ with probability $1 - 2^{-\sqrt{n}}$ (see Lemma 3.10). We then remove from V_3 all the vertices of I that have edges to $V \setminus V_3$. Doing so, we loose at most $3dn/d^{20} + 3de^{-d}n$ vertices (Lemma 3.8 part 2) with probability $1 - e^{n^{0.2+o(1)}}$. At this point (just before the iteration steps) $|V \setminus V_3| \leq 4n/d^{19}$. We now begin the iteration process. In every iteration we move at most 4 vertices to $V \setminus V_3$; these vertices contribute at least $0.8\alpha d$ edges to the set $V \setminus V_3$. If the iteration step is repeated too many times the set $V \setminus V_3$ will become too dense. Assume by contradiction that at some point (for the first time during the iterations) $|V \setminus V_3|$ doubled its size when compared to the size of $V_3 \setminus V$ before the first iteration. At this point the number of edges inside $V \setminus V_3$ is at least $\frac{1}{2}|V \setminus V_3| \cdot \frac{1}{4} \cdot 0.8\alpha d$. The size of $V \setminus V_3$ is at most $8n/d^{19} + 3 < n/d^5$. By Lemma 3 part 2 the probability for this event is at most $n^{-0.9(0.8\alpha d/8-1)} < n^{-\sqrt{c_0 d}/12} < n^{-\sqrt{d}}$. We conclude that $|V \setminus V_3| < n/d^{18}$. \square

As $|I_3 \Delta I| < |I_2 \Delta I| + |OUT_3|$, using Lemmas 3.10, 3.11 we deduce:

Corollary 3.12. $|I_3 \Delta I| < 2n/d^{18}$.

It turns out that I_3, C_3 is also a good approximation of I_{max}, C_{max} (some fixed maximum independent set and its corresponding cover). This is stated in the following lemma whose proof is deferred to Section 3.1.

Lemma 3.13. *Almost surely $|I_3 \Delta I_{max}| < n/d^5$.*

At this point we know that I_3, C_3 have the following two properties:

(i) the error term $|(I_3 \cap C_{max}) \cup (I_{max} \cap C_3)| < |I_{max} \Delta I_3| < n/d^5$.

(ii) I_3 is an independent set and every vertex of C_3 has at least 4 neighbors in I_3 .

These two properties and the fact that G does not contain dense sets imply that $I_3 \subseteq I_{max}$. This is proven in the following Lemma.

Lemma 3.14 (Extention Lemma). *Let I be any independent set of G and let $C \triangleq V \setminus I$. Let I', C', OUT' be an arbitrary partition of V for which I' is an independent set. If the following hold:*

1. $|(I' \cap C) \cup (I \cap C')| < n/d^5$.
2. Every vertex of C' has 4 neighbors in I' . None of the vertices of I' have edges to OUT' .
3. The graph G has no small dense subsets as described in Lemma 3.5 part 1.

then there exists an independent set I_{new} (and $C_{new} \triangleq V \setminus I_{new}$) such that $I' \subseteq I_{new}, C' \subseteq C_{new}$ and $|I_{new}| \geq |I|$.

Proof. If we could show that on average a vertex of $U = (I' \cap C) \cup (I \cap C')$ contributes at least $4/3$ internal edges to U , then U would form a small dense set that contradicts Lemma 3.5. This would imply that $U = (I' \cap C) \cup (I \cap C')$ is the empty set, and we could take $I_{new} = I$ in the proof of Lemma 3.14. The proof below extends this approach to cases where we cannot take $I_{new} = I$.

Every vertex $v \in C'$ has at least 4 edges into vertices of I' . Since I is an independent set it follows that every vertex of $I \cap C'$ has at least 4 edges into $I' \cap C$. To complete the argument we would like to show that every vertex of $I' \cap C$ has at least 2 edges into $I \cap C'$. However, some vertices $v \in I' \cap C$ might have less than two neighbors in $I \cap C'$. In this case, we will modify I to get an independent set I_{new} (and $C_{new} \triangleq V \setminus I_{new}$) at least as large as I , for which every vertex of $I' \cap C_{new}$ has 2 neighbors in $I_{new} \cap C'$. This is done iteratively; after each iteration we set $I = I_{new}, C = C_{new}$. Consider a vertex $v \in (I' \cap C)$ with $\deg(v)_{I \cap C'} < 2$:

- If v has no neighbors in $I \cap C'$, then define $I_{new} = I \cup \{v\}$. I_{new} is an independent set because v has no neighbors in $I \cap I'$ nor in $I \setminus I' \subseteq OUT'$ (there are no edges inside I' nor between I' and OUT).
- If v has only one edge into $w \in (I \cap C')$ then define $I_{new} = (I \setminus \{w\}) \cup \{v\}$. I_{new} is an independent set: v has no neighbors in $I \cap I'$ nor in $I \setminus (I' \cup C') \subseteq OUT$. The only neighbor of v in $I \cap C'$ is w .

The three properties are maintained also with respect to I_{new}, C_{new} (replacing I, C): properties 2, 3 are independent on the sets I, C and property 3 is maintained since after each iteration it holds that $|(I' \cap C_{new}) \cup (I_{new} \cap C')| < |(I' \cap C) \cup (I \cap C')|$.

When the process ends, we have the following situation: each vertex of $I \cap C'$ has 4 edges into $I' \cap C$. Each vertex of $I' \cap C$ has at least 2 edges into $I \cap C'$ and thus using Corollary 3.6 we get $|I \cap C'| \geq \frac{1}{2}|I' \cap C|$. The number of edges in $(I \cap C') \cup (I' \cap C)$ is at least $2|(I \cap C') \cup (I' \cap C)|$ and also $|(I \cap C') \cup (I' \cap C)| < n/d^5$, which implies that this set is empty (see Lemma 3.5 part 1). \square

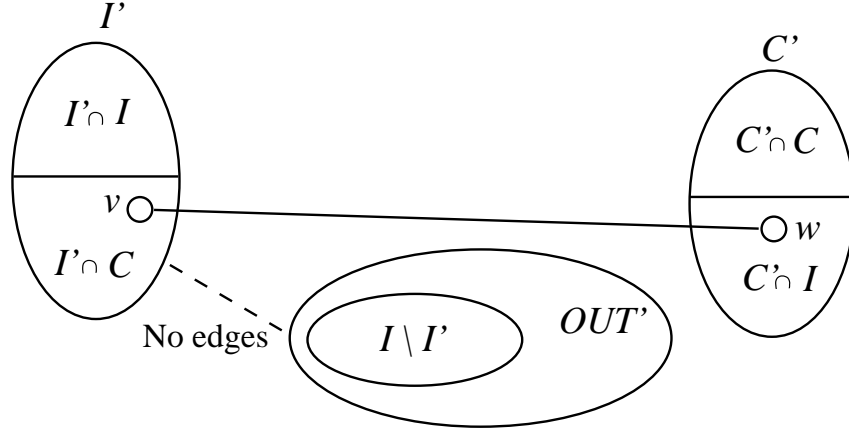


Figure 3: A vertex $v \in (I' \cap C)$ which has strictly less than 2 edges into $I \cap C'$

So far we have proved that I_3 is an independent set which is contained in some maximum independent set I_{max} . It remains to show that the 2-core of OUT_3 has no large connected components.

3.3 Proof of Claim 3.3

Having established that OUT_3 is small (Lemma 3.11), we would now like to establish that its structure is simple enough to allow one to find a maximum independent set of $G[OUT_3]$ in polynomial time. Establishing such a structure would have been easy if the vertices of OUT_3 were chosen independently at random, because a small random subgraph of a random graph G is likely to decompose into connected components no larger than $O(\log n)$. However, OUT_3 is extracted from G using some deterministic algorithm, and hence might have more complicated structure. For this reason, we shall now consider the 2-core of $G[OUT_3]$, and bound the size of its connected components.

Let A denote the 2-core of $G[OUT_3]$. In order to show that A has no large component, it is enough to show that A has no large tree. We were unable to show such a result for a general tree. Instead, we prove that A has no large *balanced* trees, that is trees in which at least $1/3$ fraction of the vertices belong to C . Fortunately, this turns out to be enough. Any set of vertices $U \subset V$ is called *balanced* if it contains at least $|U|/3$ vertices from C . We use the following reasoning: any maximal connected component of A is balanced - see Proposition 3.15 below. Furthermore, any balanced connected component of size at least $2 \log n$ (in vertices) must contain a balanced tree of size is in $[\log n, 2 \log n - 1]$ - see Lemma 3.16. We then complete the argument by showing that OUT_3 does not contain a balanced tree with size in $[\log n, 2 \log n]$.

Proposition 3.15. *Every maximal connected component of the 2-core of OUT_3 is balanced.*

Proof. Let A_i be such a maximal connected component. Every vertex of A_i has degree of at least 2 in A_i because A_i is a maximal connected component of a 2-core. $|A_i| \leq |OUT_3| < \frac{n}{d^5}$. If $\frac{|A_i \cap I|}{|A_i|}$ is more than $\frac{2}{3}$, then the number of internal edges in A_i is $> 2 \cdot \frac{2}{3}|A_i| > \frac{4}{3}|A_i|$ which contradicts Lemma 3.5. \square

Lemma 3.16. *Let G be a connected graph whose vertices are partitioned into two sets: C and I . Let $\frac{1}{k}$ be a lower bound on the fraction of C vertices, where k is an integer. For any $1 \leq t \leq |V(G)|/2$ there exists a tree whose size is in $[t, 2t - 1]$ and at least $\frac{1}{k}$ fraction of its vertices belong to C .*

Proof. We use the following well know fact: any tree T contains a *center* vertex v such that each subtree hanged on v contains strictly less than half of the vertices of T .

Let T be an arbitrary spanning tree of G , with center v . We proceed by induction on the size of T . Consider the subtrees T_1, \dots, T_k hanged on v . If there exists a subtree T_j with at least t vertices then also $T \setminus T_j$ has at least t vertices. In at least one of $T_j, T \setminus T_j$ the fraction of C vertices is at least $\frac{1}{k}$ and the lemma follows by induction on it. Consider now the case in which all the trees have less than t vertices. If in some subtree T_j the fraction of C vertices is at most $\frac{1}{k}$, then we remove it and apply induction to $T \setminus T_j$. The remaining case is that in all the subtrees the fraction of C vertices is strictly more than $\frac{1}{k}$. In this case we start adding subtrees to the root v until for the first time the number of vertices is at least t . At this point we have a tree with at most $2t - 1$ vertices and the fraction of C vertices is at least $\frac{1}{k}$. To see that the fraction of C vertices is at least $\frac{1}{k}$, we only need to prove that the tree formed by v and the first subtree has $\frac{1}{k}$ fraction of C vertices. Let r be the number of C vertices in the first subtree and let b be the number of vertices in it. Since k is integer we have: $\frac{r}{b} > \frac{1}{k} \implies \frac{r}{b+1} \geq \frac{1}{k}$. \square

We shall now prove that OUT_3 contains no balanced tree of size in $[\log n, 2 \log n]$. Fix t to be some value in $[\log n, 2 \log n]$. It is enough to show that for this fixed t there are no balanced trees of size t with probability of $o(1/\log n)$ (so we can use union bound on all possible values of $t \in [\log n, 2 \log n]$). The probability that OUT_3 contains a balanced tree of size t is at most:

$$\sum_{\substack{T \text{ is balanced,} \\ |T|=t}} \Pr[T \subseteq E] \cdot \Pr[V(T) \subseteq OUT_3 \mid T \subseteq E] \leq \quad (1)$$

$$t \cdot \max_{\substack{t_1+t_2=t, \\ t_2 \geq t/3}} \binom{\alpha n}{t_1} \binom{(1-\alpha)n}{t_2} t^{t-2} \left(\frac{d}{n}\right)^{t-1} \cdot \max_{\substack{T \text{ is balanced,} \\ |T|=t}} \{\Pr[V(T) \subseteq OUT_3 \mid T \subseteq E]\} \leq \quad (2)$$

$$t \max_{\substack{t_1+t_2=t, \\ t_2 \geq t/3}} \left(\frac{e\alpha n}{t_1}\right)^{t_1} \left(\frac{e(1-\alpha)n}{t_2}\right)^{t_2} t^{t-2} \left(\frac{d}{n}\right)^{t-1} \cdot \max_{\substack{T \text{ is balanced,} \\ |T|=t}} \{\Pr[V(T) \subseteq OUT_3 \mid T \subseteq E]\} \quad (3)$$

$$nt \left((3ed(1-\alpha)^{1/3})\right)^t \max_{\substack{T \text{ is balanced,} \\ |T|=t}} \{\Pr[V(T) \subseteq OUT_3 \mid T \subseteq E]\} \quad (4)$$

$$(1-\alpha)^{t/3} e^{\log n + t(\log d + 3)} \max_{\substack{T \text{ is balanced,} \\ |T|=t}} \{\Pr[V(T) \subseteq OUT_3 \mid T \subseteq E]\} \quad (5)$$

To upper bound the above expression by $o(1/\log n)$, it is enough to prove that for any fixed balanced tree of size t :

$$\Pr_E[V(T) \subseteq OUT_3 \mid T \subseteq E] \leq e^{-3 \log n \log d} / (1-\alpha)^{t/3}$$

(using $\log n + t(\log d + 3) \leq \log n(1 + 2 \log d + 3) < 3 \log n \log d$, which is true for large enough d). For any set of edges E we use $OUT_3(E)$ to denote the outcome of OUT_3 when $FindIS$ is invoked on E . We will use the following equality:

$$\Pr_E[V(T) \subseteq OUT_3(E) \mid T \subseteq E] = \Pr_E[V(T) \subseteq OUT_3(E \cup T)] \quad (6)$$

which is true because the distribution of E given that $T \subseteq E$ is exactly the distribution of $E \cup T$. We have to show that for any balanced tree T of size t , $\Pr[V(T) \subseteq OUT_3(E \cup T)] = e^{-3 \log n \log d} / (1 - \alpha)^{t/3}$. The difficulty in bounding the above probability is that $OUT_3(E \cup T)$ is not a uniformly chosen random set.

We will use a technique introduced at [1]. We give a review of this technique and its implementation in our setting. Using this technique in our setting involves some complications which do not exist in [1]. The new complications in our case are due to the fact that I, C are not symmetric as opposed to the coloring classes in [1]. The main idea is as follows: given a fixed tree T of size t , we define a set $R(T) \subset V(T)$ and a new algorithm $Step3$. When $Step3$ is invoked on $E, I, C, R(T)$ it outputs a set $\tilde{O}UT_3$ containing $OUT_3(E \cup T)$. Given $E, I, C, R(T)$, the set $L(T) = V(T) \setminus R(T)$ (which is chosen to be contained in C) looks as any other subset of $C \setminus R(T)$ of the same size and thus it is equally likely to be in $\tilde{O}UT_3$. We will use the following properties:

1. For every fixed configuration of edges E it holds that $OUT_3(E \cup T) \subseteq \tilde{O}UT_3(E, I, C, R(T))$.
2. The set $L(T) = V(T) \setminus R(T)$ is contained in C and its size is $\geq |V(T)|/6$.
3. The size of $\tilde{O}UT_3(E, I, C, R(T))$ is bounded by $e^{-18 \log d n}$, with probability $> 1 - 0.5e^{-3 \log d \log n}$.

To ease the notation we will use $\tilde{O}UT_3(E)$ instead of $\tilde{O}UT_3(E, I, C, R(T))$. These three properties and $t \in [\log n, 2 \log n]$ imply that (to be explained):

$$\Pr_E[L(T) \subseteq OUT_3(E \cup T)] \leq \Pr_E[L(T) \subseteq \tilde{O}UT_3(E)] < e^{-3 \log n \log d} / (1 - \alpha)^{t/3}.$$

The first inequality follows from property 1. The second inequality follows from properties 2, 3 using the following reasoning:

$$\Pr[L(T) \subseteq \tilde{O}UT_3(E)] \leq \Pr[L(T) \subseteq \tilde{O}UT_3(E) \mid \#(\tilde{O}UT_3(E) \cap C) < e^{-9 \log d n}] + \Pr[\#\tilde{O}UT_3(E) \geq e^{-9 \log d n}].$$

The second term is at most $0.5e^{-3 \log d \log n}$ by property 3. It remains to upper bound the first term. Given that the intersection of $\tilde{O}UT_3(E)$ with $C \setminus R(T)$ is of size m , its distribution is uniform over all subsets of $C \setminus R(T)$ of size m . It then follows that $\Pr[L(T) \subseteq \tilde{O}UT_3(E)]$ is bounded by the probability that a binary random variable $X \sim Bin(m, p = \frac{|L(T)|}{|C \setminus R(T)| - m})$ has $|L(T)|$ successes. Since $m \leq e^{-18 \log d n}$, $|L(T)| \geq t/6$ this probability is bounded by:

$$\binom{m}{t/6} p^{t/6} \leq \left(\frac{mep}{t/6} \right)^{t/6} = \left(\frac{me}{t/6} \cdot \frac{t/6}{|C \setminus R(T)| - m} \right)^{t/6} \leq \left(\frac{e^{-18 \log d n}}{(1 - \alpha)n/2} \right)^{t/6} \leq 0.5e^{-3t \log d} / (1 - \alpha)^{t/3}$$

In the second inequality we used $|C \setminus R(T)| - m \geq (1 - \alpha)n - 2 \log n - n/d^{18} \geq (1 - \alpha)/2$. To justify it we assume $1 - \alpha \gg 1/d^{18}$ as otherwise almost surely a random $G_{n, \frac{d}{n}, \alpha}$ graph has no connected components of size more than $\log n$ (exploring the graph from a fixed vertex is similar to a subcritical branching process, details omitted).

We will now describe $R(T)$ and the procedure $\tilde{Step}3$. We then show that the above mentioned three properties hold. Let T be a balanced tree. We partition the vertices of T into:

$$\begin{aligned} R(T) &= V(T) \cap (I \cup \{v \in C : \deg^T(v) > 11\}), \\ L(T) &= V(T) \setminus R(T). \end{aligned}$$

Algorithm $\tilde{Step}3(E, I, C, R(T))$

3. (a) Define the set \tilde{V}_2 using the following process.

Initialization: $\tilde{V}_2 = V \setminus (R(T))$,
 set $\tilde{V}_2 = \tilde{V}_2 \setminus \{v \in C \cap \tilde{V}_2 \mid \deg(v)_{I \cap \tilde{V}_2} < 0.9\alpha d\}$.

Iteratively: (i) if there is $v \in C \cap \tilde{V}_2$ with $\deg(v)_{I \cap \tilde{V}_2} < 0.8\alpha d$, remove v from \tilde{V}_2 .
 (ii) if there is $v \in I \cap \tilde{V}_2$ with $\deg(v)_{V \setminus \tilde{V}_2} > \alpha d/4 - 11$ remove v from \tilde{V}_2 .

(b) Initialization: $\tilde{V}_3 = \tilde{V}_2$,

Remove from \tilde{V}_3 all the vertices of $\tilde{V}_3 \cap I$ that have edges to $V \setminus \tilde{V}_3$.

Iteratively: if there is a vertex $v \in \tilde{V}_3 \cap C$ such that $\deg(v)_{\tilde{V}_3 \cap I} < 4$,
 remove v and its neighbors in I from \tilde{V}_3 .

(c) Set $O\tilde{U}T_3 = V \setminus \tilde{V}_3$.

A property that we will use in the proof of Lemma 3.17 is that after step 3a (of $\tilde{Step}3$) there are no T edges which touch $I \cap \tilde{V}_2$ (nor $\tilde{V}_3 \cap I$). There are no such edges because the set $I(T)$ is excluded from \tilde{V}_2 . The following Lemma shows property 1.

Lemma 3.17. $OUT_3(E \cup T) \subseteq O\tilde{U}T_3(E, I, C, R(T))$.

Proof. The set V_2 defined at Lemma 3.10 is a function of I, C and a set of edges. Denote by $V_2(E \cup T)$ the set V_2 derived using $E \cup T$ as the set of edges. Similarly, the set V_3 defined on Lemma 3.11 is a function of I, C and a set of edges. Denote by $V_3(E \cup T)$ the set V_3 derived when using $E \cup T$ as the set of edges. The idea is to show that $\tilde{V}_2 \subseteq V_2(E \cup T)$ and $\tilde{V}_3 \subseteq V_3(E \cup T)$. This is enough as $OUT_3(E \cup T) \subseteq V \setminus V_3(E \cup T)$ (see Lemma 3.11).

To ease the notation we will use V_2 instead of $V_2(E \cup T)$ and V_3 instead of $V_3(E \cup T)$. We first show that $\tilde{V}_2 \subseteq V_2$. Both \tilde{V}_2, V_2 are built using an initialization step followed by an iterative step. It is easy to see that after the initialization it holds that $I \cap \tilde{V}_2 \subseteq I \cap V_2$ and $C \cap \tilde{V}_2 \subseteq C \cap V_2$ (and thus $\tilde{V}_2 \subseteq V_2$). The initialization step is then followed by an (identical) iterative step. Consider the execution of the iterative process on V_2 . We show a parallel execution of the iterative process on \tilde{V}_2 , for which the invariant $I \cap \tilde{V}_2 \subseteq I \cap V_2$ and $C \cap \tilde{V}_2 \subseteq C \cap V_2$ is kept. Assume the vertex v is removed in the process of V_2 :

case 1: $v \in C \cap V_2$. If v is removed then $\deg^{E \cup T}(v)_{I \cap V_2} < 0.9\alpha d$. Since $I \cap \tilde{V}_2 \subseteq I \cap V_2$ it follows that $\deg^E(v)_{I \cap \tilde{V}_2} < 0.9\alpha d$ and we can remove v from $C \cap \tilde{V}_2$.

case 2: $v \in I \cap V_2$. If v is removed then, $\deg^{E \cup T}(v)_{V \setminus V_2} > \alpha d/2$. If $v \in T$ then v is already in $V \setminus \tilde{V}_2$ (by the initialization), otherwise no edges of T touch v and thus $\deg^E(v)_{V \setminus \tilde{V}_2} > \alpha d/2$ (as $V \setminus V_2 \subseteq V \setminus \tilde{V}_2$). This concludes the proof that $I \cap \tilde{V}_2 \subseteq I \cap V_2$ and $C \cap \tilde{V}_2 \subseteq C \cap V_2$.

We will now show that $\tilde{V}_3 \subseteq V_3$ by showing that $I \cap \tilde{V}_3 \subseteq I \cap V_3$ and $C \cap \tilde{V}_3 \subseteq C \cap V_3$. Both \tilde{V}_3, V_3 are built using an initialization step followed by an iterative step. The initialization first step sets: $V_3 = V_2, \tilde{V}_3 = \tilde{V}_2$. In the second initialization step the vertices of $I \cap \tilde{V}_3$ that have edges to $V \setminus V_3$ are removed. Similarly, the vertices of $I \cap \tilde{V}_2$ that have edges to $V \setminus \tilde{V}_3$ are removed. Since before the second initialization step it holds:

- (i) there are no T edges touching $I \cap \tilde{V}_3$ (since $\tilde{V}_3 = \tilde{V}_2$),
- (ii) $V \setminus V_3 \subseteq V \setminus \tilde{V}_3$,

we get that $I \cap \tilde{V}_3 \subseteq I \cap V_3$ also after the second initialization step. Consider the execution of the iterative process on V_3 . We show a parallel execution of the iterative process on \tilde{V}_3 , for which the invariant $I \cap \tilde{V}_3 \subseteq I \cap V_3$ and $C \cap \tilde{V}_3 \subseteq C \cap V_3$ is kept. Assume the vertex $v \in C \cap V_3$ is removed, thus $\deg^{E \cup T}(v)_{I \cap V_3} < 4$. Since $I \cap \tilde{V}_3 \subseteq I \cap V_3$ we deduce that $\deg^E(v)_{I \cap \tilde{V}_3} < 4$ and we may remove v from \tilde{V}_3 as well. This concludes the proof. \square

As we consider only balanced trees, the set $L(T)$ contains at list $1/6$ of the vertices in T .

Lemma 3.18. *For a balanced tree T , the size of $L(T)$ is at least $|V(T)|/6$.*

Proof. The tree T contains at least $|V(T)|/3$ vertices from C . At least $1/2$ of them are of degree at most 11 in T , as otherwise the sum of degrees in T will be at least $|T|(\frac{11}{6} + \frac{1}{6}) > 2|T| - 2$. \square

Lemma 3.19. *The size of $O\tilde{U}T_3$ is at most n/d^{18} with probability of at least $1 - n^{-\sqrt{d}}$.*

Proof. The proof is similar to the proofs of Lemmas 3.10, 3.11. The only difference is that in the initialization at step 3a (of *Step3*) we remove more vertices than in the initialization of V_2 (Lemma 3.10). Still, the number of vertices removed to $O\tilde{U}T_3$ in the initialization at step 3a is of order $< n/d^{20}$ and this is enough for bounding also the number of vertices removed in 3b. \square

Acknowledgements

This work was supported in part by a grant from the G.I.F., the German-Israeli Foundation for Scientific Research and Development. Part of this work was done while the authors were visiting Microsoft Research in Redmond, Washington.

References

- [1] N. Alon and N. Kahale. A spectral technique for coloring random 3-colorable graphs. *SIAM Journal on Computing*, 26(6):1733–1748, 1997.
- [2] N. Alon, M. Krivelevich, and B. Sudakov. Finding a large hidden clique in a random graph. *Random Structures and Algorithms*, 13(3-4):457–466, 1988.

- [3] N. Alon and J. Spencer. *The Probabilistic Method*. John Wiley and Sons, 2002.
- [4] H. Chen and A. Frieze. Coloring bipartite hypergraphs. In *Proceedings of the 5th International Conference on Integer Programming and Combinatorial Optimization*, pages 345–358, 1996.
- [5] A. Coja-Oghlan. A spectral heuristic for bisecting random graphs. In *To appear in Proceedings of the 16th ACM-SIAM Symposium on Discrete Algorithms (SODA)*, 2005.
- [6] U. Feige. Approximating maximum clique by removing subgraphs. *Siam J. on Discrete Math.*, 18(2):219–225, 2004.
- [7] U. Feige and J. Kilian. Heuristics for semirandom graph problems. *Journal of Computing and System Sciences*, 63(4):639–671, 2001.
- [8] U. Feige and R. Krauthgamer. Finding and certifying a large hidden clique in a semirandom graph. *Random Structures and Algorithms*, 16(2):195–208, 2000.
- [9] U. Feige and E. Ofek. Spectral techniques applied to sparse random graphs. Technical report, Weizmann Institute of Science, 2003.
- [10] A. Flaxman. A spectral technique for random satisfiable 3cnf formulas. In *Proceedings of the fourteenth annual ACM-SIAM symposium on Discrete algorithms*, pages 357–363, 2003.
- [11] A. Goerdt and A. Lanka. On the hardness and easiness of random 4-sat formulas. In *Proceedings of the 15th International Symposium on Algorithms and Computation (ISAAC)*, pages 470–483, 2004.
- [12] G. Grimmet and C. McDiarmid. On colouring random graphs. *Math. Proc. Cam. Phil. Soc.*, 77:313–324, 1975.
- [13] J. Håstad. Clique is hard to approximate within $n^{1-\epsilon}$. In *Proceedings of the 37th Annual Symposium on Foundations of Computer Science*, pages 627–636, 1996.
- [14] M. Jerrum. Large clique elude the metropolis process. *Random Structures and Algorithms*, 3(4):347–359, 1992.
- [15] R. M. Karp. Reducibility among combinatorial problems. In *Proceedings of a Symposium on the Complexity of Computer Computations*, pages 85–103. 1972.
- [16] R. M. Karp. *The Probabilistic Analysis of Some Combinatorial Search Algorithms*, pages 1–19. Academic Press, NY, 1976.
- [17] L. Kučera. Expected complexity of graph partitioning problems. *Discrete Appl. Math.*, 57(2-3):193–212, 1995.

A Proof of technical lemmas

Proof of lemma 3.5 part 1. Denote $c = 4/3, k = n/d^5$. The statement of the lemma trivially holds for sets with at most 2 vertices. The probability that the statement in the lemma is false is bounded by:

$$\sum_{i=3}^k \binom{n}{i} \underbrace{\left(\sum_{j=\lceil ic \rceil}^{\binom{i}{2}} \binom{i}{j} \left(\frac{d}{n} \right)^j \right)}_{\geq \text{prob. for at least } ic \text{ successes}} \leq \sum_{i=3}^k \binom{n}{i} 2^{\binom{i}{2}} \left(\frac{d}{n} \right)^{\lceil ic \rceil} \leq 2 \sum_{i=3}^k \left(\frac{n}{ei} \right)^i \left(\frac{i^2}{2eic} \right)^{\lceil ic \rceil} \left(\frac{d}{n} \right)^{\lceil ic \rceil} \quad (7)$$

the first inequality holds because the inside summation is geometric with a factor of $\frac{\binom{i}{j+1}}{\binom{i}{j}} \frac{d}{n} \leq$

$\frac{i^2}{2(j+1)} \frac{d}{n} \leq \frac{i^2}{ic} \frac{d}{n} \leq \frac{id}{cn} \leq \frac{kd}{cn} < 1/2$ (for $k \leq n/2d$). The last term in (7) is bounded by:

$$\begin{aligned} &\leq 2 \sum_{i=3}^k \frac{n^i}{i^i} \left(\frac{dn}{2eic} \right)^{\lceil ic \rceil} \left(\frac{i}{n} \right)^{2\lceil ic \rceil} \leq \frac{2}{(4e)^2} \sum_{i=3}^k \left(\frac{d}{c} \right)^{\lceil ic \rceil} \left(\frac{i}{n} \right)^{\lceil ic \rceil - i} \leq \frac{1}{32} \sum_{i=3}^k \left(\frac{d}{c} \right)^{ic+1} \left(\frac{i}{n} \right)^{ic-i} \\ &\leq \frac{d}{32c} \sum_{i=3}^k \left[\left(\frac{d}{c} \right)^c \left(\frac{i}{n} \right)^{c-1} \right]^i \leq \frac{d}{6c} \left(\left(\frac{d}{c} \right)^c \left(\frac{3}{n} \right)^{c-1} \right)^3 \leq \frac{1}{2} \left(\frac{d^{3c+1}}{n^{3c-3}} \right) \leq \frac{d^5}{n} \end{aligned}$$

(the inequality before the last one holds because the sum is geometrically decreasing with a factor of $\left(\frac{d}{c} \right)^c \left(\frac{i+1}{n} \right)^{c-1}$ which is smaller than $4/5$ for $i \leq n/d^5, c \geq 4/3, d \geq 2$). \square

Proof of lemma 3.5 part 2. The proof is essentially the same as the proof of part 1. The only difference is in the last inequality where we use:

$$\frac{d}{6c} \left(\left(\frac{d}{c} \right)^c \left(\frac{3}{n} \right)^{c-1} \right)^3 \leq e^{-(c-1) \log n + (c+1) \log d} \leq e^{-0.9(c-1) \log n}$$

\square

Proof of lemma 3.5 part 3. The proof is essentially the same as the proof of part 2. Here $k = 0.041\alpha n, c = \alpha d/12$, it holds that $\frac{kd}{cn} < \frac{1}{2}$. We also use the inequality $\left(\frac{d}{c} \right)^c \left(\frac{k+1}{n} \right)^{c-1} \leq \frac{d}{c} \left(\frac{\alpha n}{cn} \right)^{c-1} \leq \frac{12}{\alpha} (1/2)^{\alpha d/12-1} < \frac{1}{2}$ (for sufficiently large d). It then follows that the sum is bounded by:

$$\begin{aligned} \frac{2d}{6c} \left(\left(\frac{d}{c} \right)^c \left(\frac{1}{n} \right)^{c-1} \right) &\leq \left(\frac{d}{c} \right)^2 e^{-(c-1)(\log n - \log(d/c))} \leq e^{-3\sqrt{d}(\log n - 0.5 \log d) + 2 \log(12/\alpha)} \\ &\leq e^{-1.5\sqrt{d} \log n + \log d} \leq n^{-\sqrt{d}} \end{aligned}$$

In inequality 2 we used the fact that $\alpha d = c_0 \sqrt{d} \gg \sqrt{d}$. \square

Proof of lemma 3.5 part 4. Modify the proof of Lemma 3.5 by setting the parameters: $c = 50 \log d$ and $k = \frac{10n \log d}{d}$ (the set size bound). In the proof of Lemma 3.5 we used the following inequalities:

$$\left(\frac{d}{2c}\right)^c \left(\frac{2}{n}\right)^{c-1} < \frac{4}{5}; \quad \frac{kd}{cn} < \frac{1}{2}; \quad c < d$$

for showing that certain sums are geometric. These inequalities hold also for the current values of k, c . \square

Proof of Corollary 3.6. By contradiction, assume that $|A| = \delta|B|$ for some $0 < \delta < 1/2$. The number of internal edges of $A \cup B$ is at least $\frac{2|B|}{(1+\delta)|B|} = \frac{2}{1+\delta} > 4/3$. The last inequality contradicts Lemma 3.5. \square

Proof of Corollary 3.7. Let C' be such a bad set. It holds that $|\Gamma(C') \cap I| \leq |C'|$. By Lemma 3.8 part 1 at least $|C'| - n/d^{20} > \frac{9}{10}|C'|$ vertices of C' have at least $\frac{\alpha d}{2}$ edges to I . It follows that $C' \cup (\Gamma(C') \cap I)$ has at least $\frac{\alpha d}{5}|C' \cup (\Gamma(C') \cap I)| > \sqrt{d}|C' \cup (\Gamma(C') \cap I)|$ internal edges with contradiction to Lemma 3.5 part 4. \square

Proof of Lemma 3.8

Proof of part 1. The degrees into I are independent random variables. Set $\delta = 1/d^{21}$. For a fixed set of size δn the expected sum of degrees is $\mu = \delta n \alpha d$. A bad set has only $0.9\delta n \alpha d$ edges to I . The probability for a bad set of size δn is bounded by:

$$\binom{n}{\delta n} e^{-\frac{1}{2}(0.1)^2 \mu} \leq e^{-\delta n (\alpha d / 200 - \log(e/\delta))} \leq e^{-\delta n (\sqrt{c_0 d} / 200 - 10 \log d - 1)} \leq e^{-\delta n} < e^{-n/d^{21}} < e^{-n^{0.4}}.$$

In the last inequality we used $d < n^{1/40}$. \square

Proof of part 2. The proof of this lemma is very similar to the proof of Lemma B.4, detailed are omitted. \square

Proof of Lemma 3.13.

$$|I_{max} \Delta I_3| \leq |I_{max} \Delta I| + |I \Delta I_3|$$

By Corollary 3.12 $|I_3 \Delta I| < n/d^{18}$. It remains to bound $|I_{max} \Delta I|$:

$$|I_{max} \Delta I| = |I_{max} \setminus I| + |I \setminus I_{max}| \leq 2|I_{max} \setminus I| = 2|I_{max} \cap C|$$

$I_{max} = (I_{max} \cap I) \cup (I_{max} \cap C)$. One can always replace $I_{max} \cap C$ with $\Gamma(I_{max} \cap C) \cap I$ to get an independent set $(I_{max} \cap I) \cup (\Gamma(I_{max} \cap C) \cap I)$. The size of a maximum independent set of C is at most $\frac{n \log d}{d}$, this upper bounds $|I_{max} \cap C|$. From Corollary 3.7 if $|I_{max} \cap C| > n/(2d^5)$ then $|\Gamma(I_{max} \cap C) \cap I| > |I_{max} \cap C|$ which contradicts the maximality of I_{max} . \square

B Spectral Approximation

Let V' be the set of vertices of G with degree $< 5d$. We will use n' to denote $|V'|$. Notice that n' is also the dimension of A' – the adjacency matrix of $G[V']$. Let $I' = V' \cap I$, $C' = V' \cap C$. We will use α' to denote $|I'|/|V'|$. With high probability it holds that $\alpha' = \alpha(1 + O(e^{-\Omega(d)}))$; similarly $n' \geq n(1 - e^{-\Omega(d)})$. Denote by \bar{A}' the $n' \times n'$ matrix such that $\bar{A}'_{i,j} = 0$ for any $\{i, j\} \subset I'$ and $\bar{A}'_{i,j} = p = d/n$ for the other entries. We will use the fact that \bar{A}' (which is the "expectation" of A' if we ignore the diagonal) is almost surely a good spectral approximation A' (i.e. the spectral norm of $A' - \bar{A}'$ is small). The rank of \bar{A}' is 2 and it has two non zero eigenvalues. Each of the two non-zero eigenvectors which we denote by $\bar{v}_1, \bar{v}_{n'}$ is constant on I' and constant on C' (this follows from symmetry). Given that each one of $\bar{v}_1, \bar{v}_{n'}$ has only two values, we need to find β, λ which satisfy:

$$\begin{array}{c} \overbrace{\hspace{1.5cm}} \\ \begin{array}{|cccc|cccc} \hline 0 & \cdot & \cdot & 0 & p & \cdot & p & p \\ \cdot & \cdot & & & & & & \\ \cdot & & \cdot & & & & & \\ 0 & \cdot & \cdot & 0 & & & & \\ \hline p & & & & p & & & \\ \cdot & & & & & & & \\ p & & & & & & p & \\ p & & & & & & & p \\ \hline \end{array} & \begin{bmatrix} 1 \\ \cdot \\ \cdot \\ 1 \\ \beta \\ \cdot \\ \cdot \\ \beta \end{bmatrix} = \begin{bmatrix} (1 - \alpha')n'p\beta \\ \cdot \\ \cdot \\ \alpha'n'p + \beta(1 - \alpha')n'p \\ \cdot \\ \cdot \\ \cdot \\ \cdot \end{bmatrix} = \lambda \begin{bmatrix} 1 \\ \cdot \\ \cdot \\ 1 \\ \beta \\ \cdot \\ \cdot \\ \beta \end{bmatrix} \end{array} \quad \text{equivalently:}$$

$$(1 - \alpha')n'\beta p = \lambda$$

$$\alpha'n'p + \beta(1 - \alpha)n'p = \lambda\beta$$

a simple calculation gives a quadratic equation in β whose solutions are:

$$\beta_{1,2} = \frac{1}{2} \left(1 \pm \sqrt{1 + \frac{4\alpha'}{1 - \alpha'}} \right) = \frac{1}{2} \pm \left(\frac{1}{2} + \frac{\alpha'}{1 - \alpha'} + O\left(\left(\frac{\alpha'}{1 - \alpha'}\right)^2\right) \right).$$

Set $\gamma = \frac{\alpha}{1 - \alpha}$. Using $\alpha' = \alpha(1 + O(e^{-\Omega(d)}))$, $n' = n(1 + O(e^{-\Omega(d)}))$, $\sqrt{c_0/d} \leq \alpha \leq 1/2$ we get:

$$\begin{aligned} \beta_{1,2} &= \frac{1}{2} \pm \frac{1}{2} \pm \gamma + O(\alpha^2), \\ \bar{\lambda}_1 &= (1 - \alpha')n'\beta_1 p = (1 + O(\alpha^2))d && \approx d \text{ for small } \alpha, \\ \bar{\lambda}_{n'} &= (1 - \alpha')n'\beta_2 p = (-\alpha + O(\alpha^2))d && \approx -\sqrt{c_0 d} \text{ for small } \alpha, \\ \bar{v}_1 &= \underbrace{(1, 1, \dots, 1)}_{\alpha'n'} + \underbrace{(1 + \gamma, 1 + \gamma, \dots, 1 + \gamma)}_{(1 - \alpha')n'} + O(\alpha^2)\vec{1}, \\ \bar{v}_{n'} &= \underbrace{(1, 1, \dots, 1)}_{\alpha'n'} + \underbrace{(-\gamma, -\gamma, -\gamma, \dots, -\gamma)}_{(1 - \alpha')n'} + O(\alpha^2)\vec{1}. \end{aligned}$$

Remark: for every vector x which is perpendicular to $\bar{v}_1, \bar{v}_{n'}$ it holds that $\bar{A}'x = 0$ and thus $\sum_{i \in C'} x_i = 0$, $\sum_{i \in I'} x_i = 0$.

Additional notation: we will use $v_1, v_2, \dots, v_{n'}$ to denote the eigenvectors of A' corresponding to the eigenvalues $\lambda_1 \geq \lambda_2, \dots, \lambda_{n'}$. The vector of all ones is denoted by $\vec{1}$.

Proof of Lemma 3.9. The vector $\bar{v}_{n'}$ equals: $\underbrace{(1, 1, \dots, 1)}_{\alpha'n'} + \underbrace{(-\gamma, -\gamma, \dots, -\gamma)}_{(1-\alpha')n'} + O(\alpha^2)\vec{1}$ (where $\gamma = \frac{\alpha}{1-\alpha}$).

We claim that the vector $\bar{v}_{n'}$ is a fairly good approximation of $v_{n'}$: after scaling $v_{n'}$ with a proper constant it holds that $\|\bar{v}_{n'} - v_{n'}\| \leq 0.01\|\bar{v}_{n'}\|$ (see Lemma B.1 for the proof details). Without loss of generality we assume that $v_{n'}$ is already scaled (the scaling does not affect step 1 of the algorithm, thus the algorithm need not do it. We use this scaling merely for this proof).

The $\alpha'n'$ indexes with the largest absolute value belong to vertices of I' . Let \tilde{I} be the set of vertices corresponding to the $\alpha'n'$ largest absolute values in $v_{n'}$. Let $k = \tilde{I} \setminus I$ (the error term). There are exactly k vertices in $I \setminus \tilde{I}$. Match the k vertices of $I \setminus \tilde{I}$ with the k vertices of $\tilde{I} \setminus I$ in an arbitrary way so as to get k pairs. Let $i \in I, j \in C$ be such a pair. Since $j \in \tilde{I}, i \notin \tilde{I}$ it holds that $|v_{n'}(j)| \geq |v_{n'}(i)|$. Using $\bar{v}_{n'}(i) = 1, \bar{v}_{n'}(j) = -\gamma$ we conclude that $|\bar{v}_{n'}(i) - v_{n'}(i)| + |\bar{v}_{n'}(j) - v_{n'}(j)| \geq 1 - \gamma$. It follows that each such pair contributes at least $1/4$ to $\|\bar{v}_{n'} - v_{n'}\|^2$ (using $\alpha \ll 1 - \alpha$). Since $\|\bar{v}_{n'} - v_{n'}\| \leq 0.01\|\bar{v}_{n'}\|$ we conclude that $k/4 \leq 0.01\|\bar{v}_{n'}\|^2 \leq 0.01(\alpha'n' + (\frac{\alpha}{1-\alpha})^2(1-\alpha')n') \leq 0.01\alpha n$, i.e. $k < 0.04|I|$. The algorithm takes the αn (rather than $\alpha'n'$) vertices with the largest absolute value, but since $\alpha' \geq (1 - e^{-\Omega(d)})\alpha$, the additional error term can be made arbitrarily small. \square

Lemma B.1. *Let $v_{n'}$ be the eigenvector corresponding to the most negative eigenvalue of A' and let $\bar{v}_{n'}$ be the last eigenvector of \bar{A}' . Almost surely there exists a vector δ such that:*

- (i) $\bar{v}_{n'} - \delta$ is a multiple of $v_{n'}$,
- (ii) $\|\delta\| \leq 0.01\|\bar{v}_{n'}\|$.

Proof of Lemma B.1. The vector $\bar{v}_{n'}$ can be written in the basis of $v_1, \dots, v_{n'}$ as $\bar{v}_{n'} = \sum_{i=1}^{n'} c_i v_i$. It is enough to show that $c_{n'}^2$ can be made arbitrary close to $\|\bar{v}_{n'}\|^2$. We will use the following two properties: $\|(A' - \bar{\lambda}_{n'}I)\bar{v}_{n'}\| \leq \sqrt{2d}\|\bar{v}_{n'}\|$, all the eigenvalues of A' except $\lambda_1, \lambda_{n'}$ are bounded by $c\sqrt{d}$ in absolute value.

$$2d\|\bar{v}_{n'}\|^2 \geq \|(A' - \bar{\lambda}_{n'}I)\bar{v}_{n'}\|^2 = \|(A' - \bar{\lambda}_{n'}I)(\sum_{i \in I' \cup C'} c_i v_i)\|^2 =$$

$$\sum_{i=1}^{n'} (c_i)^2 (\lambda_i - \bar{\lambda}_{n'})^2 > \sum_{i=1}^{n'-1} (c_i)^2 (\lambda_i - \bar{\lambda}_{n'})^2 \geq (-c\sqrt{d} + \sqrt{c_0 d})^2 \sum_{i=1}^{n'-1} (c_i)^2.$$

The first inequality is due to Lemma B.2 part (ii). The last inequality holds because for $i \neq n' : \lambda_i \geq -c\sqrt{d}$ (see Lemma B.2 part (iii)) and $\bar{\lambda}_{n'} = (-\alpha + O(\alpha^2))d$. We will use $\sum_{i=1}^{n'-1} c_i v_i$ as δ . For sufficiently large c_0 it holds that $\|\delta\|^2 = \sum_{i=1}^{n'-1} (c_i)^2 \leq \frac{3}{c_0} \|\bar{v}_{n'}\|^2$. \square

Lemma B.2. *Let $\bar{v}_1, \bar{v}_{n'}$ be the first and last eigenvectors of the matrix \bar{A}' with corresponding eigenvalues $\bar{\lambda}_1, \bar{\lambda}_{n'}$. The following holds with high probability:*

- (i) $\|(A' - \bar{\lambda}_1)\bar{v}_1\| \leq \sqrt{3d}\|\bar{v}_1\|$,

(ii) $\|(A' - \bar{\lambda}_{n'})\bar{v}_{n'}\| \leq \sqrt{3d}\|\bar{v}_{n'}\|$,

(iii) $\forall x \perp \bar{v}_1, \bar{v}_{n'} \quad \|A'x\| \leq c\sqrt{d}\|x\|$ (c is a universal constant independent of d, α).

Proof of lemma B.2 parts (i),(ii). We will prove that $\|(A' - \bar{\lambda}_{n'}I)\bar{v}_{n'}\|^2 < 3d\|\bar{v}_{n'}\|^2$. We use the estimations: $\bar{\lambda}_{n'} = (-\alpha + O(\alpha^2))d$ and $\bar{v}_{n'} = \underbrace{(1, 1, \dots, 1)}_{\alpha'n'} + \underbrace{(-\gamma, -\gamma, \dots)}_{(1-\alpha')n'} + O(\alpha^2)\bar{1}$ where $\gamma = \frac{\alpha}{1-\alpha}$.

$$(A' - \bar{\lambda}_{n'}I)\bar{v}_{n'} \approx (A' + \alpha dI) \begin{bmatrix} 1 \\ \cdot \\ \cdot \\ -\frac{\alpha}{1-\alpha} \\ \cdot \\ \cdot \end{bmatrix} = \begin{bmatrix} -\frac{\alpha}{1-\alpha}\text{deg}_C(v) + \alpha d \\ \cdot \\ \cdot \\ \text{deg}_I(v) - \frac{\alpha}{1-\alpha}\text{deg}_C(v) - \frac{\alpha}{1-\alpha}\alpha d \\ \cdot \\ \cdot \end{bmatrix} \approx \begin{bmatrix} -\frac{\alpha}{1-\alpha}\text{deg}_C(v) + \alpha d \\ \cdot \\ \cdot \\ \text{deg}_I(v) - \frac{\alpha}{1-\alpha}\text{deg}_C(v) \\ \cdot \\ \cdot \end{bmatrix}$$

The symbol \approx is used when we neglect vectors whose squared norm is bounded by $5c_0^2n$. We can neglect these terms as their contribution to the squared norm is $\ll d\|\bar{v}_n\|^2$ (d is sufficiently large so that $c_0^2 \ll d$). The terms $O(\alpha^2)$ appearing in $\bar{\lambda}_{n'}$ and in $\bar{v}_{n'}$ contribute to the squared norm at most $5nc_0^2$. The same is true for the factor $-\frac{\alpha}{1-\alpha}\alpha d$ that is omitted in the last \approx .

To bound the squared norm of the above vector we need to estimate the random variables:

$$\sum_{v \in I} \left(\frac{\alpha}{1-\alpha} (\text{deg}_C(v) - (1-\alpha)d) \right)^2, \quad \sum_{v \in C} \left(\frac{1}{1-\alpha} ((1-\alpha)\text{deg}_I(v) - \alpha\text{deg}_C(v)) \right)^2.$$

Since $d\|\bar{v}_n\|^2 \geq \sqrt{c_0dn}(1 - e^{-\Omega(d)})$ it is enough to show that with high probability the above sums are bounded by $1.2\sqrt{c_0dn}$. First let us compute $\mathbb{E}[(\text{deg}_C(v) - (1-\alpha)d)^2]$ for a vertex $v \in I$. This is just the variance of $\text{deg}_C(v)$ which is $(1-\alpha)d + O(d/n)$. Thus the expected value of the first sum is $\alpha n \frac{\alpha^2}{1-\alpha} d + O(d) \ll \sqrt{c_0dn}$. Bounding $\mathbb{E}[(1-\alpha)\text{deg}_I(v) - \alpha\text{deg}_C(v)]^2$ for $v \in C$ is more tedious. Setting $\beta = 1 - \alpha$, the above expectation is:

$$\beta^2\mathbb{E}[\text{deg}_I(v)^2] + \alpha^2\mathbb{E}[\text{deg}_C(v)^2] - 2\alpha\beta\mathbb{E}[\text{deg}_I(v)]\mathbb{E}[\text{deg}_C(v)],$$

using $\mathbb{E}[Y^2] = \text{VAR}(Y) + \mathbb{E}[Y]^2$, which is true for any r.v. Y , the above becomes (up to an error of $O(d/n)$):

$$\beta^2(\alpha d + (\alpha d)^2) + \alpha^2(\beta d + (\beta d)^2) - 2(\alpha\beta d)^2 = \alpha\beta d = \alpha(1-\alpha)d.$$

The expectation of the second sum is $(1-\alpha)n \left(\frac{\alpha d}{1-\alpha} + O(d/n) \right) = \sqrt{c_0dn} + O(d)$. To complete the argument we need to show a concentration result for the above two expectations. This is done at Lemma B.4. The proof of part (i) is similar, details omitted. \square

Proof of lemma B.2 part (iii). If $x \perp \bar{v}_1, x \perp \bar{v}_{n'}$, then $\sum_{i \in I'} x_i = 0, \sum_{i \in C'} x_i = 0$. Thus the following holds:

The first term $\Pr[\Delta \geq D]$ is bounded by $ne^{-D/d}$ using a combination of the union bound and a proper version of the Chernoff bound ($D > 100d$). To bound the other term, we will use the vertex exposure martingale. The probability space that we use is $G_{n,d/n,\alpha}$ conditioned on $\Delta < D$. Define

$$f(G) = \sum_{v \in C} \left(\frac{1}{1-\alpha} ((1-\alpha)\deg_I(v) - \alpha\deg_C(v)) \right)^2.$$

Let X_0, X_1, \dots, X_n be the martingale sequence, where X_i is the expectation of f after exposing the edges of the graph induced by the first i vertices. Notice that X_0 is just $\mathbb{E}[f(G) \mid \Delta < D]$. The value of X_n is the value of $f(G)$, where G is a random graph (from $G_{n,d/n,\alpha}$) given that all vertices have degrees bounded by D . To use the Azuma inequality we need to upper bound the martingale difference $|X_{i+1} - X_i|$ (for $i = 0, \dots, n-1$). It is known that if f satisfies the vertex Lipschitz condition with some constant Λ , then also the martingale difference is bounded by Λ (see [3]). We will show that f satisfies a Lipschitz property with respect to the constant $3D^2$. Fix a vertex v and move from one configuration of v 's edges into a different configuration of v 's edges by changing (add/remove) one edge at a time. After all the changes are done the difference in the value of the summand of v is never more than D^2 . The difference induced by the other summands is bounded in the following way: there are at most D changes, each change has an influence bounded by $2D$ on some other summand (since all degrees are bounded by D). It follows that the total difference is bounded by $D^2 + 2D^2$. By Azuma's inequality the following holds:

$$\Pr[X_n > X_0 + \lambda] \leq e^{-\lambda^2/(2n3D^2)}.$$

Setting $\lambda = 0.1\mu$ we derive:

$$\Pr[X_n > 1.1\mu] \leq e^{-0.01\mu^2/(2n3D^2)} < e^{-c_0dn^2/1000nD^2} = e^{-c_0dn/1000D^2}.$$

Using $D = 100d \log n$: the last term is at most $e^{-\sqrt{n}}$, the first term (from equation (8)) is at most $e^{-D/d+\log n} < e^{-99 \log n}$.

□