Heuristics for Semirandom Graph Problems

Uriel Feige[*] Joe Kilian[†]

January 23, 2000

Abstract

We consider semirandom graph models for finding large independent sets, colorings and bisections in graphs. These models generate problem instances by blending random and adversarial decisions. To generate semirandom independent set problems, an independent set S of αn vertices is randomly chosen. Each edge connecting S with $\bar{S}$ is chosen with probability p, and an adversary is then allowed to add new edges arbitrarily, provided that S remains an independent set. The smaller p is, the greater the control the adversary has over the semirandom graph. We give a heuristic that with high probability recovers an independent set of size αn whenever $p > (1+\epsilon)\ln n/\alpha n$, for any constant $\epsilon > 0$. We show that when $p < (1-\epsilon)\ln n/\alpha n$, an independent set of size $|S|$ cannot be recovered, unless $NP \subseteq BPP$.

We use our result for maximum independent sets to obtain greatly improved heuristics for the model of k-colorable semirandom graphs introduced by Blum and Spencer. For constant k, our results are optimal up to constant factors in the edge probabilities..

In the semirandom model for graph bisection, a random bisection $(S, \bar{S})$ of the vertices is chosen. Each edge $(u, v) \in S \times \bar{S}$ is independently chosen with probability q and each edge $(u, v) \notin S \times \bar{S}$ is independently chosen with probability $p > q$. The adversary may then arbitrarily remove edges in $S \times \bar{S}$ and add edges not in $S \times \bar{S}$. Extending the work of Boppana, we give a heuristic that recovers this bisection with high probability when $p - q \geq c\sqrt{p \log n/n}$, for c a sufficiently large constant.

[*]Department of Applied Math and Computer Science, the Weizmann Institute, Rehovot 76100, Israel. feige@wisdom.weizmann.ac.il. Incumbent of the Joseph and Celia Reskin Career Development Chair. Part of this work done while the author was visiting the NEC Research Institute.

[†]NEC Research Institute, 4 Independence Way, Princeton, New Jersey, 08540. joe@research.nj.nec.com

1 Introduction

1.1 Heuristics and their evaluation

The plethora of worst-case NP-hardness results for problems in graph theory motivates the study of *heuristics* that do not solve every problem exactly, but give "useful" answers to a "typical" subset of the problems, where "useful" and "typical" are usually not well defined. In practice, a heuristic may be evaluated by running it on a collection of input graphs ("benchmarks"), and recording the quality of the answers obtained. If one heuristic consistently gives better answers than another, then we have empirical evidence of it being a better heuristic.

Though running heuristics on benchmarks is sometimes informative, we seek more rigorous measures for evaluating heuristics. In this paper, we consider the problems of finding maximum independent sets, graph colorings and graph bisections. Our goal is to develop heuristics for these problems with well defined and provable properties, and to identify useful general techniques for developing such heuristics.

A rigorous analog to a "useful" answer is the notion of approximation, where the goal of the heuristic is to guarantee a solution that is within a small factor of the optimal one. For example, a heuristic is said to approximate the maximum independent set (MIS) within a ratio $\rho > 1$ if for every input graph, the size of the maximum independent set is at most a multiplicative factor of ρ larger than the size of the independent set returned by the heuristic.

However, although approximation algorithms are known for a number of NP-hard problems, at least two of the problems we consider seem very hard to approximate in the worst case. It is known (through work culminating in [20]) that for any constant $\epsilon > 0$, it is NP-hard (under randomized reductions) to approximate MIS to within $n^{1-\epsilon}$ (throughout this paper, n denotes the number of vertices in the input graph); the best approximation ratio known to be achievable for MIS is $O(n/(\log n)^2)$ [9]. Similarly, it is known (through work culminating in [13]) that one cannot approximate the minimum number of colors needed to color a graph to within $n^{1-\epsilon}$ for any constant $\epsilon > 0$; the best known approximation ratio is $O(N(\log\log N)^2/(\log N)^3)$ [19]. Even when the graph is 3-colorable, the best known coloring algorithm requires $\tilde{O}(n^{3/14})$ colors [7]. Hence, if we were to evaluate heuristics for MIS and coloring based on their worst-case approximation ratios, they would all perform badly. For graph bisection, the third problem that we study, the situation with respect to approximation algorithms is ambiguous. At the time the work reported in this paper was done, no polynomial time algorithm was known to approximate the number of edges in the optimum bisection within factors significantly smaller than n. However, recently an algorithm approximating bisection within a ratio of $O(\sqrt{n}\log n)$ was announced [15], and this ratio is likely to be further improved in the near future. No NP-hardness result for approximating bisection size is known.

1.2 Random models for graph problems

When very little can be done in the worst case, one would like to compare the performance of heuristics on "average" instances, or those that typically occur in practice. But how does one model such instances? One possible model is that of a random graph (see the survey in [16]). The question then arises of how well random graphs model inputs that interest us in "real life" applications. But even regardless of this question, the random graph model does not seem to provide a good way of distinguishing between good and bad heuristics in the case of MIS. For example, if each edge is chosen independently at random with probability $1/2$, then the size of the maximum independent set is almost surely roughly $2\log n$. An elementary greedy heuristic almost surely finds an independent set of size $\log n$. No heuristic, not even the most sophisticated one, is known to find independent sets significantly larger than $\log n$. Hence, most heuristics have roughly the same performance in this random graph model, making it an inadequate framework for comparing between them.

Another random model that has been suggested is similar to the random graph model, but has a solution "planted" in the graph. That is, the graph is created by a two-step process. First, a solution is randomly determined, even before the graph itself has been determined. Then, edges are included in the graph essentially at random, but with simple restrictions to ensure that the planted solution remains a solution.

In the random "planted" MIS model, an independent set S of size K is chosen at random. Then, each edge $(u, v) \notin S \times S$ is chosen with probability p. The restriction on (u, v) ensures that S remains an independent set; for reasonably large K and p, S is the unique largest independent set. The larger K is, the easier it is to find the independent set. For a constant p, the lowest value of K that can be provably handled by known heuristics (spectral techniques, in this case) is $\Omega(\sqrt{n})$ [4].

In the random k-colorable graph model, a graph is partitioned into k color classes, and edges are placed at random with probability p between color classes. Alon and Kahale [2], also using spectral methods, show that with high probability a k-coloring can be recovered when $p \geq c/n$ and $c > 0$ is a sufficiently large constant.

In the random bisection model, a set S containing $n/2$ vertices of the graph are randomly selected. Edges crossing this bisection are included with probability q and edges not crossing are included with probability $p > q$. Boppana [8] shows that with high probability one can recover an optimal bisection when $p - q \geq c\sqrt{p \log n/n}$, for c a sufficiently large constant.

The random planted solution models allow one to have greater control over the distribution and nature of the optimal solution, but still run into problems. On the technical side, this framework still doesn't differentiate between simple and sophisticated heuristics for the MIS problem. When $K > c\sqrt{n \log n}$ for sufficiently large c, then with high probability the vertices of the independent set are easily recognized to be the K vertices of lowest degree in G. Hence a trivial heuristic will solve MIS in this case; the sophisticated heuristics guarantee only a marginal improvement in the size of K.

A more serious problem with these random models is that they may simply not capture the space of "useful" problems. The problems produced are extremely unstructured, which probably does not reflect real-world examples. Furthermore, there is a danger that one might overly exploit the statistical properties of these graphs (such as their eigenvalue structure) and produce heuristics that will perform well on these specific distributions, but fail to perform on more realistic distributions.

1.3 Semirandom models for graph problems

In this paper we study *semirandom* models for finding maximum independent sets, colorings and bisections. A number of similar semirandom models for the coloring problem were first defined and studied by Blum and Spencer [6]; we extend their notion in a natural way to the other problems. Recall that in one of the randomized frameworks discussed above, a solution to the problem was first generated and then the edges of the graph were generated via some very simple random process. For semirandom models, we have a third step in which an adversary is allowed to further modify the graph. These modifications cannot be arbitrary, or the adversary can remake the graph into a worst-case problem. We therefore place some restrictions on the manner in which the adversary is allowed to modify the graph. This semirandom framework is the strongest of those studied in [6], and in particular is more adversarial than a model in which an adversary first chooses a worst case input instance, and then its decisions are corrupted by random noise.

The motivation for this framework is that it mediates between the unstructured, "uninteresting" graphs produced by the purely random models and the worst-case graphs that are seemingly beyond any heuristic's ability to solve. Each adversary induces a different distribution on the problem instances, so heuristics can't fall into the trap of overtraining to a particular probability distribution. Informally, we replace *consists of random edges* with *contains some random edges*, a much weaker hypothesis to make about a graph. It is thus hopeful that heuristics that work well for this model will be more robust.

We note that the semirandom adversary can foil many popular tools for heuristics. The adversary can alter the degrees of vertices, and foil heuristics based on vertex degrees, create "local maxima" (such as large, but not large enough independent sets) to foil heuristics based on local search, and modify the spectrum of the graph. As a result, comparatively much less is known about these models.

A semirandom model for finding large independent sets

Our semirandom model for MIS has a size parameter $0 \leq \alpha \leq 1$ and a random density parameter $0 \leq p \leq 1$. Given α and p and a problem size n, a problem graph G is generated by the following three steps:

1. A set S of αn vertices (we ignore all rounding issues in this paper) is chosen uniformly at random from the n vertices of G.

2. Each edge $(u, v) \in S \times \bar{S}$ is included in G with independent probability p. We call this random bipartite graph G_{min}.

3. The adversary A may add to G_{min} any edges $(u, v) \notin S \times S$, giving G, the final problem instance.

We denote by $\mathsf{MIS}_A(\alpha, p)$ the resulting distribution on G. The goal of the heuristic is to identify an independent set in G that is at least as large as S.

We can equivalently view the adversary as choosing a graph "sandwiched" between G_{min} and the graph G_{max} whose edges include all of $(S \times \bar{S}) \cup (\bar{S} \times \bar{S})$.

The smaller p is, the more freedom the adversary has to choose the graph in a worst-case fashion. When $p = 0$, the adversary can choose G to be an arbitrary graph with independent set S, giving the worst case model as a limiting instance.

Note that the random process does *not* generate edges in $\bar{S} \times \bar{S}$, unlike the planted MIS model mentioned earlier. As these edges only constrain the adversary, we have a more general model without them. We need this generality when we apply our heuristics for this model to the coloring model described below. Note also that when $\bar{S}$ contains more vertices than S, then $\bar{S}$ and not S is the maximal independent set in G_{min}. In this case, G may also contain independent sets larger than S. We only require our heuristic to find an independent set of size $|S|$. It would be unrealistic to require the heuristic to find the largest independent set in G, as there are no restrictions on the subgraph of G induced on $\bar{S}$.

A semirandom model for k-coloring

We use one of the semirandom models for k-coloring put forth by Blum and Spencer [6]. This model has a colorability parameter k and a density parameter p. Given k, p and n, G is generated as follows:

1. The n vertices of G are randomly partitioned into k disjoint sets, $S_1, \ldots, S_k$, each of size n/k.

2. Each edge (u, v) such that $u \in S_i$ and $v \in S_j$, where $i \neq j$, is included in G_{min} with probability p.

3. The adversary A may add to G_{min} any edges (u, v) such that $u \in S_i, v \in S_j$ and $i \neq j$, giving G, the final problem instance.

We denote by $\mathsf{Color}_A(n, k, p)$ the resulting distribution on G.

We can equivalently view the adversary as choosing an arbitrary graph sandwiched between G_{min} and the graph G_{max}, containing all edges (u, v) such that $u \in S_i, v \in S_j$ and $i \neq j$. As with the model for MIS, setting $p = 0$ gives an essentially worst case model for k-coloring, though with the stipulation that all the color classes must be the same size. More generally, one can specify the sizes of the color classes as an additional parameter. We call this the *imbalanced* case and the case where the classes are of equal size the *balanced* case.

A semirandom model for graph bisection

Our semirandom model for graph bisection has two density parameters $0 \leq q < p \leq 1$, that may depend on the size of the input graph, n. Given p and q and a problem size n, a problem graph G is generated by the following three steps:

1. A set S of $n/2$ vertices is chosen uniformly at random from the n vertices of G.
2. Each edge $(u, v) \in (S \times S) \cup (\bar{S} \times \bar{S})$ is included in G_{rand} with independent probability p and each edge $(u, v) \in S \times \bar{S}$ is included in G_{rand} with independent probability q.
3. The adversary may add to G_{rand} any edges $(u, v) \in (S \times S) \cup (\bar{S} \times \bar{S})$ and remove any edges $(u, v) \in S \times \bar{S}$, yielding G, the final problem instance.

We denote by $\mathsf{Bisect}_A(n, p, q)$ the resulting distribution on G.

Notation: Throughout, α, p and q will be used only as above. For a set T of vertices, $N(T)$ denotes the set of their neighbors in graph G. We denote set subtraction by $A \setminus X$ (that is, those elements in A that are not in X).

1.4 Our results

We give algorithms for the three semirandom models described above, and for the case of independent sets and coloring, we give lower bounds showing that our algorithms are close to optimal in the allowed settings of the parameters.

For a given set of parameters, we say that an algorithm succeeds in a semirandom model with "high probability" if it succeeds with probability $1 - o(1)$, as n grows sufficiently large, for all adversaries. Here, the probability is taken over the choice of the algorithm's coin flips and the choice of the graph (before modification by the adversary). By a standard argument we can assume that an optimal deterministic adversary exists; this subtlety doesn't arise in our arguments.

For simplicity, we speak of "semirandom graphs"; which of the three models being used is obvious by context.

Theorem 1 *For α any positive constant and $p = (1+\epsilon)\ln n/\alpha n$, where ϵ is any positive constant, there is a random polynomial time algorithm that with high probability recovers an independent set of size αn from a semirandom graph.*

Using the adversarial component of the graph, it can be shown that the value of p in Theorem 1 is best possible, up to low order terms.

Theorem 2 *In the semirandom model, if $p = (1-\epsilon)\ln n/\alpha n$ for some $\epsilon > 0$, then unless $NP \subseteq BPP$, every random polynomial time algorithm will with high probability fail to find an independent set of size αn in G (against an optimal adversary).*

Our algorithm for finding independent sets can be used for solving semirandom k-coloring problems, in the balanced case, for any constant k and $p = (1+\epsilon)k\ln n/n$. In contrast, the algorithm of Blum and Spencer [6] requires that $p \geq n^{\delta}/n$, where $\delta > (k^2-k-2)/(k^2+k-2)$.

Theorem 3 *For any constant k, there is a polynomial time algorithm that k-colors k-colorable semirandom graphs with high probability (over the choice of the graph) whenever $p > (1+\epsilon)k\ln n/n$ and the color classes are balanced. When the color classes are arbitrary, then the algorithm recovers a largest color class with high probability.*

For the balanced case, our result is close to optimal, by the following hardness result:

Theorem 4 *Let $\epsilon > 0$, $k \geq 3$ be a constant and $p < (1-\epsilon)\ln n/n$. Then unless $NP \subseteq BPP$, every random polynomial time algorithm will with high probability fail to k-color a semirandom k-colorable graph G with balanced color classes (against an optimal adversary).*

For the unbalanced case, Blum and Spencer [6] show that it is NP-hard to 4-color semirandom graphs whenever $p < n^{-\epsilon}$ for any constant $\epsilon > 0$. In the construction behind their proof, the largest color class (comprising nearly all the graph) is easily found, but it is NP-hard to recover any one of the other three color classes. For $k < 3$, the coloring problem is trivial in the worst case, and for $k = 3$ removing the vertices of a single color class yields a 2-colorable graph, which may then be colored. Hence, our result in Theorem 3 for the unbalanced case is essentially the best that can be hoped for.

We remark that our algorithm for finding independent sets works also in models that have less randomness (and hence, are more adversarial) than our semirandom graph model. One such model is the d-neighbors model, where in G_{min}, each vertex of $\bar{S}$ has d random neighbors in S. The graph G is then an arbitrary graph sandwiched between G_{min} and G_{max}. It can be shown that a simple modification of our algorithm recovers in this model independent sets of size αn, when d is a large enough constant that depends only on α. Observe that in this model $G_{\min}$ has only $O(n)$ edges, whereas in our original semirandom model $G_{\min}$ has $\Omega(n \log n)$ edges.

Finally, we show that one can find an optimal bisection for semirandom bisection problems.

Theorem 5 *Let c be a sufficiently large constant and let $p - q \geq c\sqrt{p \log n / n}$. Then there is a polynomial time algorithm that recovers an optimal bisection from a semirandom graph with high probability.*

1.5 Techniques and related work

Lovasz introduced the *theta* function as an upper bound on the size of the maximum independent set [22]. The theta function can be approximated within arbitrary precision in polynomial time, using semidefinite programming. Goemans and Williamson [18] showed how semidefinite programming can be used in order to approximate problems such as max-cut. Inspired by their work, Karger, Motwani and Sudan [21] used semidefinite programming to obtain improved coloring algorithms. Alon and Kahale [3] used the work of [21] to show that the theta function can be used to find medium size (n^δ vertex) independent sets in graphs that have linear size independent sets (improving the values of δ previously obtained in [9]).

In terms of approximation ratio, the theta function (and similar semidefinite programs) appear to have little to offer in the worst case. In [12] it is shown that for every $\epsilon > 0$ there are graphs with multiplicative gaps of $n^{1-\epsilon}$ between the size of the maximum independent set and the value of the theta function. Indeed, Håstad's result [20] implies that, unless NP is easy, no easily computable function will give better than a $n^{1-\epsilon}$ approximation in the worst case.

However, our current work singles out semidefinite programming as an approach that can cope with the semirandom graph model, unlike other heuristics for MIS. In more detail, our algorithm for finding large independent sets has two phases. In the first phase (Sections 2.2 and 2.3) G is partitioned into a small number of parts, such that some of these parts are composed mostly of vertices of S. This first phase uses semidefinite programming. Its analysis is based only on $G_{\min}$, and goes through regardless of what the adversary does. This illustrates the robustness of (some) algorithms based on semidefinite programming.

In the second phase (Sections 2.4, 2.5 and 2.6) we "clean up" the output of the first phase, and extract S (or a different independent set of the same size). Many of the difficulties introduced by the adversary manifest themselves in the second phase. In particular, there is the problem of getting out of local maxima. To illustrate this problem, assume that the algorithm already found a maximal independent set I composed mostly of vertices of S (though not containing all of S). One may then hope that local heuristics such as k-opt (exchanging a constant k number of vertices of I with $V \setminus I$ so as to hopefully get a new independent set that is not maximal and hence can be expanded) would allow one to eventually extract S. However, in our semirandom model, the adversary is strong enough so as to make no k-exchange possible, even when I is almost as large as αn (details omitted). Our method of improving over local maxima is based on global computations (finding maximum matchings) rather than local ones, and may be of independent interest.

To find good bisections for semirandom graphs, we propose an algorithm similar to that of Boppana [8] (which is based on a lower bound of Donath and Hoffman [11] on the bisection size). Given a graph we construct a semidefinite program whose optimal value is (hopefully) the number of edges crossing the the minimum bisection of the graph; given such an oracle, it is straightforward to recover the actual bisection. We use methods due to Boppana to show that the semidefinite program gives the correct value with high probability in the purely random case, when the adversary doesn't add or delete any edges. To deal with the adversary, we show that our semidefinite program is robust in that if it gives the correct answer for a given graph it will still give the correct answer after the adversary has modified it.

2 Finding large independent sets

We present a randomized heuristic for finding large independent sets in semirandom graphs. In analyzing the success probability of our heuristic, we distinguish between two sources of randomness. One is the random process generating $G_{\min}$, the other is the random coin tosses of our heuristic. To deal with the first type of randomness, we define properties that a *typical* $G_{\min}$ has (details are given in Section 2.1). We show that with high probability $G_{\min}$ is typical. Thereafter, we assume that $G_{\min}$ is typical, and analyze our heuristic under this assumption. It is shown that when $G_{\min}$ is typical, then regardless of the graph G constructed by the adversary, our heuristic outputs a list of at most n independent sets (rather than just one, for reasons explained in Section 2.4), and with probability at least $1/2$, S is on this list. Here probability is taken only over the coin tosses of the heuristic. Repeating the heuristic several times with independent coin tosses recovers S almost surely. It may happen that the heuristic recovers several different large independent sets in G (as we remarked earlier, S need not be the largest independent set in G). In such a case we may not be able to tell which of the large independent sets is S, but this is not required in Theorem 1.

Our heuristic has five phases, described and analyzed in the following subsections. Phase 1 performs a coarse partitioning on the vertices. Phase 2 further refines this partition and extracts an independent set from each partition. Phase 3 combines some of these independent sets to form an independent set of linear size. Phase 4 trims this independent set down so that it fits entirely within the planted independent set. Phase 5 expands this independent set until it becomes equal to the planted independent set.

Many of the constants involved are arbitrary and are specified only for concreteness. Throughout our analysis, we ignore divisibility issues, eschewing careful roundoff analyses. Such considerations do not materially affect our algorithms or proofs.

In Section 2.1 we give some properties of random and semirandom graphs. We describe and analyze our heuristic in Sections 2.2– 2.6, proving Theorem 1. In Section 2.7 we prove Theorem 2, showing the near optimality of our results.

2.1 Preliminaries

Given a graph $G = (V, E)$ and a set of vertices $X \subseteq V$, we define $G[X]$ to be the graph G induced on X. We denote the complement of S by $\bar{S}$ and we denote $X \bigcap \bar{Y}$ by $X \setminus Y$ (i.e., set subtraction).

We review some well known properties of random graphs.

We first consider the expansion properties of the random graph G_{min} generated by $\mathsf{MIS}_A(\alpha, p)$ (i.e., the graph generated before modification by the adversary). Recall that G_{min} is a bipartite graph on $S \bigcup \bar{S}$.

Definition 1 *We say that G_{min} has the* k-collision property *if for every $T \subseteq \bar{S}$ and $U \subseteq S$ such that $|T|, |U| \geq k$, there is an edge (t, u) in G_{min}, where $t \in T$ and $u \in U$.*

Definition 2 *We say that G_{min} has the* d-expansion property *if for every $T \subset \bar{S}$ of cardinality at most $31\alpha n/32d$, $|N(T) \bigcap S| \geq d|T|$.*

Lemmata 6 and 7 show that with high probability, G_{min} will have the k-collision and d-expansion for suitably chosen k and d. We assume that α is some positive constant.

Lemma 6 *Let $p = c \ln n/n$ for some constant $c > 0$ and let $k = \frac{2n \ln \ln n}{c \ln n}$. Then when n is sufficiently large, with high probability G_{min} will have the k-collision property.*

Proof: There are at most $\binom{n}{k}^2$ ways of choosing T and U. For each such choice, the probability of the bad event that there is no edge joining T and U is $(1-p)^{k^2}$. By the union bound on the probabilities of the bad events, it suffices to show that

$$\binom{n}{k}^2 (1-p)^{k^2} = o(1),$$

for n sufficiently large. By standard inequalities, $\binom{n}{k} \leq (en/k)^k$ and $(1-p) < e^{-p}$. Hence, the above expression is at most

$$\left(\left(\frac{en}{k}\right)^2 e^{-pk}\right)^k = \left(\left(\frac{ec \ln n}{2 \ln \ln n}\right)^2 e^{-2 \ln \ln n}\right)^k = \left(\frac{ec}{2 \ln \ln n}\right)^{2k}.$$

For n sufficiently large the lemma follows. $\square$

Similarly, we can show

Lemma 7 *Let $p = (1+\epsilon)\ln n/\alpha n$ and let $d > 0$ be an arbitrary constant. For n sufficiently large, G_{min} has the d-expansion property.*

The proof of Lemma 7 appears in Section A.1 in the appendix.

Lemma 8 follows immediately from standard bounds on the tails of the binomial distribution (cf. [6], A.4).

Lemma 8 *Let $p = (1+\epsilon)\ln n/\alpha n$, with $0 < \epsilon < 1$. Then with high probability over the choice of G_{min}, the number of edges in G_{min} is at most $2n\ln n$.*

We call this the *sparseness property.*

Lemmata 6, 7 and 8 imply that for every $\delta < 1$ and every constant $d > 0$, when n is sufficiently large there is high probability that an n-node graph G_{min} has the $n/(\ln n)^{\delta}$-collision and d-expansion properties and has at most $2n\ln n$ edges. We say that such a G_{min} is *typical.* We will assume that G_{min} is typical in our analyses.

Hall's theorem gives a necessary and sufficient condition for finding a matching in a bipartite graph; we note a simple (undoubtably known) generalization, whose proof we sketch for completeness.

Definition 3 *Given a graph G and disjoint vertex sets L and R, a* perfect d-matching *from L to R is a matching from each vertex $l \in L$ to d adjacent vertices in R, such that each vertex $r \in R$ is matched at most once. A maximum d-matching is a d-matching of maximum cardinality.*

Maximum d-matchings may be found in a straightforward way given an algorithm that finds perfect d-matchings when they exist.

Lemma 9 *(Hall) G has a perfect d-matching from L to R iff for every subset $S \subseteq L$, $|N(S) \bigcap R| \geq d|S|$.*

Proof: Clearly, if G has a perfect d-matching, then $|N(S)| \geq d|S|$ for all $S \subseteq L$. We now show the converse. Consider the graph G' whose vertices contain R and d copies of each vertex $l \in L$, denoted $l_1, \ldots, l_d$, and for which (l_i, r) is an edge of G' iff (l, r) is an edge in G. If $|N(S)| \geq d|S|$ for all $S \subseteq L$ then $|N(S)| \geq |S|$ for all $S \subseteq L'$. This is because the vertices in S correspond to at least $|S|/d$ "representatives" from L. By Hall's theorem, there is a complete matching from L' to R, which trivially translates into a d-matching from L to R. $\square$

Note also that this proof is constructive in that it shows how to efficiently find a perfect d-matching, if it exists, using as a subroutine a standard bipartite matching algorithm.

Phase 1

1. Let $G = (V, E)$, $G_0 = G$, $\ell = 0$ and $n = |V|$.
2. While $(V_{\ell+1}, Z_{\ell+1}) = \mathsf{SetVec}(G_\ell, \alpha n/4) \neq \mathsf{fail}$
   $\ell = \ell + 1$
   $G_\ell = G[V \setminus \bigcup_{j \leq \ell} V_j]$

Figure 1: Phase 1 computes a constant number of "almost" independent sets $(V_1, \ldots, V_\ell)$ along with additional information used to further partition these sets.

2.2 A coarse partition using semidefinite programming

The following lemma is implicit in [3, 21].

Lemma 10 *Let $G(V, E)$ be a graph on n vertices that contains an independent set of size K. There exists a polynomial time procedure* $\mathsf{SetVec}(G, K)$ *that returns (Q, Z), where $Q \subseteq V$ consists of $K/2$ vertices, and Z consists of $K/2$ corresponding unit vectors in R^n; we denote by $z_i \in Z$ the unit vector corresponding to $v_i \in Q$. If $v_i, v_j \in Q$ and $(v_i, v_j) \in E$, then $\langle z_i, z_j \rangle < -K/(2n - K)$ (i.e., the angle between v_i and v_j is large). If G doesn't have an independent set of size K,* $\mathsf{SetVec}(G, K)$ *may return either a valid (Q, Z) with the above properties, or* fail.

For completeness, the proof of Lemma 10 is presented in Section A.2 in the appendix.

Phase 1, given in Figure 1, uses Lemma 10 to partially decompose V into disjoint subsets $V_1, V_2, \ldots$. Phase 1 stops producing new V_i when SetVec fails to find such a set. By Lemma 10, each set V_i contains $\alpha n/8$ vertices. For $v_j \in V_i$, let z_j denote the corresponding vector in Z_i. Again by Lemma 10, whenever $(v_j, v_k) \in E$ and $v_j, v_k \in V_i$,

$$\langle z_j, z_k \rangle < -(\alpha n/4)/(2n - \alpha n/4) < -\alpha/8.$$

We define $Q_i \triangleq G[V_i]$ (the subgraph of G induced on V_i), and define $S_i \triangleq V_i \bigcap S$, where S is the independent set planted in G.

Definition 4 *A set V_i is* good *if $|S_i| \geq \alpha^2 n/32$.*

Proposition 11 *At the end of Phase 1,* $\displaystyle\sum_{\{i | V_i \text{ is good}\}} |S_i| \geq \alpha n/2.$

partition(V_i, Z_i, h)

1. Choose h hyperplanes passing through the origin, at random, by choosing their unit normal vectors uniformly on the unit sphere.
2. Vertices v_j, v_k are equivalent if they are on the same side of all h hyperplanes. Denote the resulting equivalence classes by $V_{i1}, \ldots, V_{im}$ ($m \leq 2^h$).
3. Return$(V_{i1}, \ldots, V_{im})$

Figure 2: Partitioning vertices based on their vector values [18, 21].

Proof: By construction, $V_1, \ldots, V_\ell$ are disjoint. Since $|V_i| = \alpha n/8$ for all $1 \leq i \leq \ell$, $\ell \leq 8/\alpha$. Less than $\alpha n/4$ vertices of S are not contained in any V_i; if $\alpha n/4$ such vertices remained, SetVec$(G_\ell, \alpha n/4)$ would not have failed. The number of vertices of S contained in sets V_i that are not good is at most $\frac{\alpha^2 n}{32} \cdot \frac{8}{\alpha} \leq \frac{\alpha n}{4}$. As $|S| = \alpha n$, the proof follows. $\square$

2.3 Refining the partition using random hyperplanes

In Phase 2, described in Figure 3, we use a technique developed by [18, 21] to further partition the sets V_i, to obtain a collection $\{I_{ij}\}$ of independent sets; the desired outcome of this phase is summarized in Lemma 13.

Given the set Z_i of unit vectors corresponding to V_i, we (following [18, 21]) partition V_i by choosing random hyperplanes through the origin. Two vertices of V_i are in the same partition if they are on the same side of all of the hyperplanes. We denote this procedure partition(V_i, Z_i, h), described in Figure 2. Better partitioning techniques are also suggested in [21], but are not needed for our results.

Let $\bar{S}_i = V_i \setminus S_i$, and let E_i be the set of edges connecting S_i and $\bar{S}_i$ in $G_{\min}$. Note that E_i does not contain the edges added by the adversary. We first define the edges of E_i that are contained in a partition.

Definition 5 *Let $E_i(V_{i1}, \ldots, V_{im})$ denote all pairs $(v_j, v_k) \in E_i$ such that $v_j, v_k \in V_{iq}$ for some q.*

Lemma 12 shows that the partition procedure produces partitions containing few edges of E_i.

Lemma 12 *([21]) For a set of vertices V_i computed in Phase 1 and for E_i as defined above, let $V_{i1}, \ldots, V_{im}$ denote the (randomized) output of* partition(Q_i, Z_i, h). *The expectation of $|E_i(V_{i1}, \ldots, V_{im})|$ is at most $|E_i|(1-\rho)^h$, where $\rho > 1/2$ is a constant depending only on α and the expectation is taken over the coin tosses of* partition(Q_i, Z_i, h).

Phase 2

1. Let $\rho = \cos^{-1}(-\alpha/8)/\pi$, /* as in Lemma 12 */
   h be the least integer satisfying $(1-\rho)^h \le \alpha/16 \ln n$.
2. For $1 \le i \le \ell$,
   Let $Q_i = G[V_i]$,
   $(V_{i1}, \ldots, V_{im}) = \mathsf{partition}(Q_i, h)$
   For $1 \le j \le m$
   Let M_{ij} be the vertices in a maximal matching of $G[V_{ij}]$,
   $I_{ij} = V_{ij} - M_{ij}$

Figure 3: Phase 2 further subdivides $V_1, \ldots, V_j$, and extracts from them a set of independent sets, $\{I_{ij}\}$.

Proof: Suppose that $v_j, v_k \in V_i$ and that $(v_j, v_k) \in E_i$ (and hence, $(v_j, v_k) \in E$). By the definition of SetVec, and the construction of Phase 1, the associated vectors z_j, z_k satisfy $\langle z_j, z_k \rangle < -\alpha/8$. This inequality implies that the angle θ_{jk} between the vectors z_j and z_k is at least some constant $\theta > \pi/2$. (One can take $\theta = \cos^{-1}(-\alpha/8)$; the exact value is irrelevant to our analysis.) As shown in [18], if $(v_j, v_k) \in E$, a random hyperplane will separate z_j and z_k with probability $\rho = \theta_{jk}/\pi \ge \theta/\pi > 1/2$. Hence, v_j and v_k will fall into the same equivalence class with probability at most $(1-\rho)^h$. The lemma then follows by the linearity of expectation. $\square$

Using the approach of [21], Phase 2 makes a careful choice of the parameter h, partitions each V_i using partition, and extracts an independent set from each subpartition by removing a maximal matching. Our choice for h is the least integer satisfying $(1-\rho)^h \le \alpha/16 \ln n$. Then the number of equivalence classes $m = 2^h \le O((\ln n)^\delta)$ for some $\delta < 1$, where $\delta < 1$ follows from $\rho > 1/2$.

We consider the independent sets $\{I_{ij}\}$ produced by Phase 2. We first note that the I_{ij} are indeed independent sets, since if I_{ij} had an edge it could be added to the "maximal" matching M_{ij}. The number of independent sets produced is at most $lm \le O((\ln n)^\delta)$ for some $\delta < 1$ and n sufficiently large. (The inequality follows from the fact that $l \le 8/\alpha$ and $m \le O((\ln n)^\delta)$.) A priori, these independent sets may have no relation to S, or any other large independent set in the graph. However, we can show that some of the $\{I_{ij}\}$ are mostly made up of vertices in S and that a constant fraction of the vertices of S are contained in these useful independent sets.

Definition 6 *An independent set I_{ij} is* useful *if $|I_{ij} \bigcap S_i| \ge 3|I_{ij}|/4$.*

The following lemma is of central importance to our analysis.

Lemma 13 *For typical $G_{\min}$, with probability at least $1/2$ (over the random choices of* partition*), at least $\alpha n/8$ vertices of S are in useful independent sets I_{ij}.*

Proof: Consider an arbitrary set V_{ij} that is the outcome of $\mathsf{partition}(Q_i, h)$. We define the *surplus* of V_{ij} as $\mathsf{sur}(V_{ij}) \triangleq |S \bigcap V_{ij}| - |\bar{S} \bigcap V_{ij}|$. From V_{ij}, an arbitrary maximal matching is removed so as to obtain an independent set I_{ij} (that may possibly be empty). As each edge can have at most one endpoint in S, it follows that each time an edge is removed, the surplus of the remainer either stays the same or increases. Therefore $\mathsf{sur}(I_{ij}) \geq \mathsf{sur}(V_{ij})$, implying also that $|I_{ij} \bigcap S| \geq \mathsf{sur}(V_{ij})$. Recall that I_{ij} is useful if $|I_{ij} \bigcap S| \geq 3|I_{ij}|/4$. From the collision property of $G_{\min}$, it can easily be deduced that if $|I_{ij} \bigcap S| > n/(\log n)^{\delta}$ for some $0 < \delta < 1$, then $|I_{ij} \bigcap \bar{S}| = o(|I_{ij}|)$. Hence I_{ij} is useful. From the above it follows that if $\mathsf{sur}(V_{ij}) > n/(\log n)^{\delta}$ then necessarily I_{ij} will be useful. We now show that many sets V_{ij} have a large surplus, implying Lemma 13.

By Proposition 11, at least $\alpha n/2$ vertices from S belong to good V_i. A good V_i is partitioned by h random hyperplanes into m parts $\{V_{ij}\}$. From Lemma 12, the expectation of $|E(V_{i1}, \ldots, V_{im})|$ is at most $|E_i|(1-\rho)^h$. Hence the expectation of $\sum_i |E(V_{i1}, \ldots, V_{im})|$ is at most $\sum_i |E_i|(1-\rho)^h$. By the sparseness of $G_{\min}$ (Lemma 8), it follows that $\sum_i |E_i| \leq 2n \ln n$. Hence with probability at least $1/2$ we have by Markov's inequality that

$$\sum_i |E(V_{i1}, \ldots, V_{im})| \leq 4n \ln n (1-\rho)^h/\alpha.$$

By our choice of h, $(1-\rho)^h \leq \alpha/16 \ln n$, implying

$$\sum_i |E(V_{i1}, \ldots, V_{im})| \leq \alpha n/4.$$

Recall that our algorithm removes a maximal matching from each V_{ij}. For the sake of analysis, we as a thought experiment remove a maximal matching containing only edges from E_i, giving a set U_{ij} (the algorithm itself does not have the luxury of knowing which edges of Q_i belong to E_i). Then over all good V_i, the total number of edges removed is at most $\alpha n/4$, meaning that at least $\alpha n/2 - \alpha n/4 = \alpha n/4$ vertices from S remain in the sets $\{U_{ij}\}$. Call a set U_{ij} *large* if $|U_{ij} \bigcap S| \geq \alpha^2 n/128m$, where $m = 2^h$. As there are at most $8/\alpha$ sets V_i, there are at most $8m/\alpha$ sets U_{ij}. Hence, those U_{ij} that are not large can contain in total at most $\alpha n/16$ vertices from S. Therefore, at least $\alpha n/4 - \alpha n/16 = 3\alpha n/16$ vertices from S are contained in large U_{ij}.

Recall that by our choice of h, we have $m \leq O((\ln n)^{\delta})$ for some $\delta < 1$. Hence large U_{ij} contain $\Omega(n/(\log n)^{\delta})$ vertices of S for some $\delta < 1$. As they contain no edges from $G_{\min}$, the property of Lemma 6 implies that for large U_{ij}, $|U_{ij} \bigcap \bar{S}| \leq o(|U_{ij}|)$, and hence the surplus of U_{ij} is $(1 - o(1))|U_{ij}|$. By our construction, U_{ij} and V_{ij} have the same surplus. So if we consider only

Phase 3

1. Nondeterministically guess the set $Q \subseteq \{I_{ij}\}$ of useful independent sets, I_{ij}.
2. Let $V = \bigcup_{I_{ij} \in Q} I_{ij}$
3. Let M be the vertices of a maximal matching of $G[V]$.
4. $I = V - M$

Figure 4: Phase 3 nondeterministically (with only polynomially many choices) recovers a large independent set, I.

those V_{ij} that had large U_{ij} in the above experiment, they give rise to useful I_{ij} containing a total of at least

$$(1 - o(1)) \sum_{\text{large } U_{ij}} |U_{ij}| \geq (1 - o(1)) 3\alpha n/16 > \alpha n/8$$

vertices from S. $\square$

2.4 Creating a linear size independent set I

Phase 3 finds in G an independent set of size $\Omega(n)$. Recall that the number of independent sets constructed in Section 2.3 is at most $8m/\alpha \leq \log n$ (for large enough n). For typical G_{min}, Lemma 13 shows that with probability at least 1/2 (over the coin tosses of partition), $\alpha n/8$ vertices of S are in useful independent sets. We continue under the assumption that this indeed happens. (The whole heuristic can be repeated with independent coin tosses to boost up the success probability.)

The algorithm for Phase 3 is described in Figure 4. For simplicity, we allow this algorithm to nondeterministically guess the set Q of useful independent sets. However, given the bound on $|\{I_{ij}\}|$, there are less than n possible values for Q; one can try them all. One might not know which guess was correct, but this is not important since one can check the solution (if any) produced by each computation path. We remark that for greater efficiency, one may guess just one useful (sufficiently large) independent set, and deduce the rest via matching techniques, but for brevity we eliminate this optimization.

By the same argument as for I_{ij}, the outcome I of Phase 3 is an independent set. Lemma 14 shows that it contains many vertices in S.

Lemma 14 $|I \bigcap S| \geq |S|/16$

Phase 4

1. Let $l = \lfloor 1 + 32/\alpha \rfloor$.
2. Find a maximum l-matching from I to $\bar{I}$; let M be the vertices in this matching.
3. Let $I' = I - M$.

Figure 5: Phase 4 constructs a large subset of I containing only vertices in S.

Proof: Since V is the union of good independent sets, that are disjoint by construction, it follows that at most $|V|/4$ of its vertices are in $\bar{S}$. Since no edge of G has both its vertices in S,

$$|M \bigcap S| \leq |M \bigcap \bar{S}| \leq |V|/4.$$

It follows that

$$|I \bigcap S| \geq \frac{3}{4}|V| - \frac{1}{4}|V| \geq |S|/16.$$

□

2.5 Purifying I.

Let I be an independent set in G with $|I \bigcap S| \geq |S|/16$. Observe that by the collision property, almost all the vertices of I belong to S; $|I \bigcap \bar{S}| = o(|I|)$. Phase 4, described in Figure 5, extracts from I a subset I', all of which is contained in S.

Clearly, I' will be an independent set, since I is. Lemma 15 shows that for typical G_{min}, Phase 4 indeed finds a large independent set contained in S. Note (again) that the algorithm is being run on G; G_{min} is unknown to the algorithm and appears only in the analysis.

Lemma 15 *Let G_{min} be typical, and assume that n is large enough so that the expansion parameter $d = d(n)$ in Lemma 7 can be chosen to be larger than $l = \lfloor 1 + 32/\alpha \rfloor$. Then $I' \subseteq S$ and $I' > |S|/32$.*

Proof: To see that $|I'| > |S|/32$, observe that at most $n/l < \alpha n/32$ vertices of I can be in any (partial) l-matching. Hence, $|I'| > |I| - \alpha n/32 \geq \alpha n/32$, since $|I| > \alpha n/16$.

It remains to show that $I' \subseteq S$. Let $A \triangleq I \bigcap S$ and $B \triangleq I \bigcap \bar{S}$ and $C \triangleq \bar{I} \bigcap S$. By assumption, $|A| > |S|/16$. We show that any maximum l-matching from I to $\bar{I}$ must match all the vertices in B, completing the proof of the lemma.

Phase5

1. Let $V' = V \setminus N(I')$ and $G' = G[V']$.
2. Let M be the vertices in a maximum (nonbipartite) matching on G'.
3. Let $I'' = V' \setminus M$.
4. If $|I''| > |I'|$, Let $I' = I''$ and goto 1.

Figure 6: Phase 5 expands I' until $I' = S$.

As $N(B)\bigcap I = \emptyset$ (since I is an independent set), it follows that $|N(B)\bigcap S| \leq 15|S|/16$. By the expansion property of G_{min} it follows that $|B| \leq (15|S|/16)/d$; it then follows that for every $B' \subseteq B$, $|N(B')\bigcap C| \geq l|B'|$ (note that $N(B')\bigcap C = N(B')\bigcap S$). Observe that $N(B)\bigcap C$ is disjoint from $N(A)\bigcap \bar{I}$, since $A \subseteq S$; hence none of the vertices of A are matched to vertices in $N(B)\bigcap C$. By Lemma 9, it follows that there exists an l-matching from B to $N(B)\bigcap C$. It follows that any l-matching from $A'\bigcup B'$ to $\bar{I}$, for any $A' \subseteq A$, may be transformed into an l-matching from $A'\bigcup B$ by keeping l-matching from A' and adding the (nonconflicting) matching from B. Hence, any maximum l-matching must contain all of B $\square$

2.6 Expanding I'.

Phase 5, described in Figure 6, expands I', ultimately recovering S.

Lemma 16 shows that each iteration of Phase 5 indeed expands I', eventually making it equal to S.

Lemma 16 *Let $I' \subseteq S$ and $|S|/32 < |I'| < |S|$, and let G_{min} be typical, and large enough so that the parameter d in Lemma 7 can be chosen to be larger than* 32. *Then at the end of Step* 4 *of Phase* 5*, $I'' \subseteq S$ and $|I''| > |I'|$.*

Proof: Since $I' \subseteq S$, and S is an independent set, V' will contain S. Define $Q = V' \setminus S$. Since S is an independent set, the maximum size of a matching on G' is at most $|Q|$, and this size can be achieved only if every vertex in Q is contained in the matching. By construction, $S \cap N(Q)$ (the neighborhood of Q, restricted to S) is contained in $S - I'$. Noting that $|S - I'| < 31|S|/32$, it follows from the expansion properties of G (inherited from G_{min}) that $|Q| < |S|/32$, or $S \cap N(Q)$ would have at least $31|S|/32$ vertices. It then follows from the expansion properties that $|S \cap N(Q)| > |Q|$, unless Q is empty (in which case $I'' = S$), and that $|S \cap N(Q')| > |Q'|$ for every nonempty $Q' \subseteq Q$. Hence there exists a complete matching from Q to S. This implies that any maximum matching found will include every vertex from Q (it will not conflict with any partial submatching from S) and that at least one vertex from $S - I'$ will be left over, since $|S - I'| \geq |S \cap N(Q)| > |Q|$. This implies that I'' will be contained in S and properly contain I'. $\square$

Lemma 16 implies that repeated iterations of the main loop of Phase 5 expands I' until $I' = S$.

2.7 NP-Hardness results

Proof of Theorem 2: Let $p = (1-\epsilon)\ln n/\alpha n$ for some $\epsilon > 0$, and assume for the sake of contradiction that there is a random polynomial time algorithm that finds independent sets of size αn in the semirandom model. We shall show how such an algorithm can be used in order to find the maximum independent sets in arbitrary graphs, implying the unlikely outcome that NP has random polynomial time algorithms.

We first observe that for any constant $\epsilon > 0$, $G_{\min}$ contains polynomially many isolated vertices. For n sufficiently large, a vertex $v \in \bar{S}$ is isolated with (independent) probability

$$\left(1 - \frac{(1-\epsilon)\ln n}{\alpha n}\right)^{\alpha n} = n^{-1+\epsilon}(1+o(1)),$$

by standard inequalities. Hence there are $(1-\alpha)n^{\epsilon}$ expected isolated vertices in $\bar{S}$; by a standard bound on the tails of the binomial distribution ([5], A.13), with high probability at least $m = (1-\alpha)n^{\epsilon}/2$ vertices in $\bar{S}$ will be isolated.

Let G' be a graph on $3m/2$ vertices, V', in which we seek to find an independent set of size $m/2$ (this problem is NP-hard). Let V_1 consist of $\alpha n - m/2$ vertices, V_2 consist of $n - \alpha n - m$ vertices. Construct the graph G with vertices $V' \cup V_1 \cup V_2$. The edges of G consist of those of G' (connecting vertices in V'), and all edges between every vertex in V_2 and any other vertex (hence vertices in V_2 have degree $n-1$, and vertices in V_1 have degree $|V_2|$). It follows from the construction that G has an independent set of size αn iff G' has an independent set of size $m/2$.

It is thus NP-hard to find an independent set of size αn in G, and hence NP-hard to find an independent set of size αn with constant probability when the distribution on problem graphs consists of randomly permuted copies of G. We conclude the proof by exhibiting an adversary such that the semirandom graphs have this distribution. Note that we do not need for the adversary to run in polynomial time.

Given G' and $G_{\min}$, our adversary finds (not in polynomial time) an independent set I of G', where $|I| = m/2$ and a set T of m isolated vertices in $\bar{S}$ of $G_{\min}$ (failing if T doesn't exist; this happens with insignificant probability). The adversary then maps the $m/2$ vertices of I randomly to distinct vertices of S and maps the m vertices of $V' \setminus I$ randomly (but distinctly) to T. Given this mapping, it then adds the edges corresponding to those in G'. By the construction, it is allowed to add these edges, and furthermore, every nonedge in G'

corresponds to a nonedge in G_{min}. Finally, the adversary connects every vertex of $\bar{S} \setminus T$ to every other vertex. It can be verified that this graph is isomorphic to G. V_1 corresponds to those vertices in S that do not correspond to I and V_2 corresponds to $\bar{S} \setminus T$.

The generation of G_{min} and the operations of the adversary are invariant with respect to permutations on the vertex names, implying that the graphs are produced according to the desired distribution. □

3 Finding k-colorings

3.1 Applying our heuristic to find k-colorings

We achieve nearly optimal results for coloring semirandom graphs simply by using our algorithm for finding independent sets.

Proof of Theorem 3: We show that the independent set heuristic works well in the coloring framework. Consider an arbitrary color class S of size at least n/k (such a color class must exist), corresponding to $\alpha = 1/k$. Since k is a constant, α is bounded above 0. We can view the semirandom model for coloring as first inserting random edges between S and $\bar{S}$, then inserting random edges between the other color classes, and finally allowing the adversary to add edges. The first set of random edges are placed just as in the semirandom model for finding independent sets. The second set of random edges and the edges the coloring adversary is allowed to add form a proper subset of the edges the independent set adversary is allowed to add. Hence, for any coloring adversary there exists an independent set adversary yielding the same distribution on problem graphs. It follows that with high probability the color class S will be among the independent sets output by the heuristic.

When the color classes are balanced, the argument above applies to all color classes simultaneously. Hence running the heuristic several times, we get polynomially many independent sets as output, and among them the k color classes. A k-coloring is obtained by picking k disjoint independent sets that together cover the whole graph. This may be done by exhaustive search over all possible choices, as there are only polynomially many choices to consider. More efficient methods also exist, but are omitted. □

3.2 An NP-hardness result

The proof of Theorem 4 is nearly the same as that of Theorem 2; we sketch it below.

Proof of Theorem 4: By the same analysis as the proof of Theorem 2, with high probability after the random edges have been placed each color class will have $m = n^{\Omega(1)}$ isolated vertices. The adversary can, as before, imbed an arbitrary (balanced) k colorable graph G into these vertices. The adversary connects any (u, v) corresponding to different-colored vertices not being used to imbed G'. The resulting graph is equal to a complete balanced k-partite graph on $k(n - m)$ vertices unioned with a disjoint copy of G', with the vertices randomly permuted. Any k coloring of this graph yields a k-coloring of G'. The theorem follows. $\square$

4 Graph bisection

Let $G(V, E)$ be a graph with n vertices, where n is even. A *bisection* of G is a partition of V into a set S and and its complement set $\bar{S} = V \setminus S$, each of size $|S| = |\bar{S}| = n/2$. Let $E(S, \bar{S})$ denote the set of edges connecting S and $\bar{S}$ in G. Then the *bisection size* of a graph G, denoted by $b(G)$, is the minimum number of edges in a bisection of G. That is,

$$b(G) = \min_S [|E(S, \bar{S})|]$$

Computing $b(G)$ is NP-hard. Boppana [8] develops a heuristic for this problem (based on [11]), and analyzes its performance on random graphs with planted bisections. Specifically, he considers the following model for random graphs:

- The vertex set is partitioned into two equal size sets S and $\bar{S}$.
- For every pair of vertices independently:
  - If both vertices belong to the same side of the bisection (either both belong to S or both belong to $\bar{S}$) then they are connected by an edge with probability p.
  - If the two vertices belong to different sides of the bisection (one belongs to S, the other to $\bar{S}$) then they are connected by an edge with probability q.

If q is sufficiently smaller than p, then w.h.p. $(S, \bar{S})$ is a unique minimum bisection. Boppana shows that when

$$p - q \geq c\sqrt{p\frac{\log n}{n}}$$

for a sufficiently large constant c, then with high probability (over the choice of input graph) his heuristic recovers the minimum bisection. Moreover, the heuristic also provides a certificate that the bisection found is the minimum possible.

We consider a semirandom model for graph bisection. In this model, a graph G_{rand} is chosen at random as above. Then a *monotone adversary* is allowed to modify G_{rand} by applying an arbitrary sequence of the following *monotone transformations*:

- The monotone adversary may remove from the graph any edge (u, v) crossing a minimal bisection ($u \in S$ and $v \in \bar{S}$).
- The monotone adversary may add to the graph any edge (u, v) not crossing the bisection ($u, v \in S$ or $u, v \in \bar{S}$).

The resulting graph is denoted by G. We say that these transformations are monotone because if $(S, \bar{S})$ is the {unique} minimum bisection in G_{rand}, then it is also the {unique} minimum bisection of G.

4.1 Computing the size of the bisection

Let $h(G)$ be an arbitrary function on graphs. It is our intention that h provides a heuristic for graph bisection in the sense that for many graphs, $h(G) = b(G)$.

Definition 7 *A function h is* robust *with respect to monotone adversaries if for every graph G_1 and every graph G_2 obtainable as a sequence of monotone transformations of G_1, $h(G_1) = b(G_1)$ implies $h(G_2) = b(G_2)$.*

Remark: By a simple argument, we may equivalently require that G_2 is obtained through a single monotone transformation on G_1.

A monotone adversary is powerless against a robust function h: whenever G_{rand} is such that h computes the exact value of $b(G_{\text{rand}})$, the monotone adversary cannot prevent h from giving the true value of $b(G)$.

Proposition 17 *Let h be a function satisfying the following properties:*

Lower bound. For every graph G, $h(G) \leq b(G)$.

Bounded monotonicity. Let G^+ be the graph G with one additional edge. Then $h(G) \leq h(G^+) \leq h(G) + 1$.

Then h is robust in the sense of Definition 7.

Proof: Let G_1 be an arbitrary graph for which $h(G_1) = b(G_1)$. Let G_2 be a graph obtained by a single monotone transformation (exploiting our above remark), using a (minimal) bisection $(S, \bar{S})$ of G_1 with $b(G_1)$ edges. We need to show that $h(G_2) = b(G_2)$.

The inequality $h(G_2) \leq b(G_2)$ follows directly from the lower bound property for h. It remains to show that $b(G_2) \leq h(G_2)$. Suppose G_2 is obtained from G_1 by adding edge (u, v), where $u, v \in S$ or $u, v \in \bar{S}$. Then $b(G_2) = b(G_1)$. By (the "monotonicity" part of) bounded monotonicity,

$$b(G_2) = b(G_1) = h(G_1) \leq h(G_2).$$

Suppose that G_2 is obtained from G_1 by subtracting an edge (u, v), where $u \in S$ and $v \in \bar{S}$. Then $b(G_2) = b(G_1) - 1 = h(G_1) - 1$. By (the "bounded" part of) bounded monotonicity, $h(G_2) \geq h(G_1) - 1 \geq b(G_2)$. $\square$

To compute the bisection in our semirandom model, we would like to find a polynomial time computable function h that is robust, and in addition has the following property:

Probably good. With high probability over the choice of G_{rand},

$$h(G_{\text{rand}}) = b(G_{\text{rand}}).$$

Boppana [8] describes a polynomial time computable function h and shows that it satisfies the "lower bound" property and the "probably good" property. It is not easy (for us) to verify that Boppana's function has the "bounded monotonicity" property. Instead of analyzing Boppana's function, we propose the following semidefinite relaxation of bisection, for which proving robustness is straightforward, and "probable goodness" is proved using the techniques of [8].

For a graph $G(V, E)$ find an order n matrix $X = \{x_{ij}\}$ satisfying:

1. $\forall i$, $x_{ii} = 1$,
2. $\sum_{ij} x_{ij} = 0$,
3. The matrix X is symmetric and positive semidefinite,

and let $h(G)$ be given by

$$h(G) = \min_X h_X(G)$$

where X ranges over all matrices satisfying the above constraints, and

$$h_X(G) \triangleq \sum_{\substack{(i,j) \in E \\ i<j}} \frac{1 - x_{ij}}{2}.$$

Using semidefinite programming, $h(G)$ can be computed in polynomial time within arbitrary precision limits.

Proposition 18 *The function h described above is robust.*

Proof: From Proposition 17, it suffices to show that h satisfies two properties: "lower bound", and "bounded monotonicity".

To see that $h(G) \leq b(G)$, consider the indicator vector $s \in \{+1, -1\}^n$ having entry $+1$ for vertices in S and -1 for vertices in $\bar{S}$, and let $X = ss^T$. Note that $x_{ij} = -1$ when i and j are on opposite sides of the cut and $x_{ij} = 1$ otherwise. Clearly, X satisfies the constraints of the semidefinite program. Moreover, $h_X(G) = b(G)$, because $(1 - x_{ij})/2 = 1$ when $x_{ij} = -1$, and $(1 - x_{ij})/2 = 0$ when $x_{ij} = 1$. Note that $h(G) \leq h_X(G)$.

To see that h is bounded monotone, use the fact that $|x_{ij}| \leq 1$, shown below, implying that $0 \leq \frac{1-x_{ij}}{2} \leq 1$. It then follows that for any X and any G, G^+ as in Proposition 17,

$$h_X(G) \leq h_X(G^+) \leq h_X(G) + 1.$$

Observe that feasibility of a matrix X depends only on the number of vertices in a graph but not on the actual graph itself. Taking X to be an optimal solution for $h(G^+)$ we obtain $h(G) \leq h_X(G) \leq h_X(G^+) = h(G^+)$, and taking X to be the optimal solution for $h(G)$ we obtain $h(G^+) \leq h_X(G^+) \leq h_X(G)+1 = h(G)+1$.

To see that $|x_{ij}| \leq 1$, consider the standard decomposition of a semidefinite matrix X into MM^T, for some matrix M. The condition that $x_{ii} = 1$ implies that the sum of squares of each row of M is equal to 1. It then follows that the dot product of any two rows is between -1 and 1. $\square$

The function h is the solution to a semidefinite minimization problem. Similar to linear programs, semidefinite programs have duals, and the optimal value of a dual maximization problem is never larger than the optimal value of the primal minimization problem (though unlike linear programming, it may happen that both problems have bounded feasible solutions that are not equal to each other). For information on how to obtain the dual of a semidefinite program, see for example [1]. For completeness, we summarize the method below. Using the notation $C \cdot X = \sum_{ij} c_{ij} x_{ij}$, then for the primal SDP
minimize $C \cdot X$ subject to

- $A_i \cdot X = b_i$ for $1 \leq i \leq m$,
- X is positive semidefinite,

there corresponds a dual
maximize $b^t y$ subject to

- $C - \sum_{i=1}^{m} y_i A_i$ is positive semidefinite.

Simple manipulations show that the dual of our semidefinite program is:

Maximize $m/2 + (\sum_{i=1}^{n} y_i)/4$
subject to the constraint that the matrix $M = -A - y_0 J - Y$ is positive semidefinite.
Here m is the number of edges in the graph, A is its adjacency matrix, J is the all 1 matrix of order n, y_0 is an auxiliary variable which affects feasibility but not the objective function, and Y is a diagonal matrix with $y_1, \ldots, y_n$ along its diagonal.

Using the dual formulation we prove:

Lemma 19 *For a sufficiently large constant c, the function h defined above is "probably good" when*

$$p - q \geq c\sqrt{p\frac{\log n}{n}}.$$

That is, with probability at least $1 - 2n^{-5}$*,* $h(G_{rand}) = b(G_{rand})$*, where the probability is taken over the choice of* G_{rand}*.*

Proof: We shall show that if $p - q$ is sufficiently large, than with high probability over the choice of G_{rand}, the dual semidefinite maximization program has a feasible solution with value $b(G_{\text{rand}})$. This shows that $h(G_{\text{rand}}) \geq b(G_{\text{rand}})$. Together with the "lower bound" property, we obtain that $h(G_{\text{rand}}) = b(G_{\text{rand}})$.

We now "guess" a feasible solution for the semidefinite maximization problem. For every $1 \leq i \leq n$, let y_i be the *contribution* of vertex i to the bisection $(S, \bar{S})$, namely, y_i is the difference between the number of neighbors that vertex i has on the other side of the bisection and the number of neighbors that vertex i has on its own side of the bisection.

The value of the objective function of the maximization semidefinite program is then

$$\begin{aligned} \frac{m}{2} + \frac{\sum_{i=1}^{n} y_i}{4} &= \frac{m}{2} + \frac{2|E(S,\bar{S})| - 2(m - |E(S,\bar{S})|)}{4} \\ &= |E(S,\bar{S})| \end{aligned}$$

which is $b(G_{\text{rand}})$, as desired.

It remains to show that w.h.p. (over the choice of G_{rand}), it is possible to choose y_0 such the matrix $M = -A - y_0 J - Y$ is positive semidefinite.

Observe that regardless of the value of y_0, the indicator vector s for S described in the proof of Proposition 18 is an eigenvector of the matrix M with eigenvalue 0. We shall choose $y_0 = -1$ and show that then the matrix M has no negative eigenvalues, implying that it is positive semidefinite. As the proof of this part is technical and long, it is presented in Section A.3 in the appendix. $\square$

Remark: In [23] there is a variety of semidefinite formulations for graph bisection. Possibly, for every graph G, our function h and Boppana's function give the same value, making Lemma 19 a restatement of the main theorem of [8].

4.2 Finding the minimum bisection

We have seen that the function h almost surely computes the size of the bisection in our semirandom graph model. Not surprisingly, it can also be used so as to actually find the original bisection $(S, \bar{S})$. For this discussion, we assume that $q > 1/n^{2+o(1)}$ (otherwise, G_{rand} is likely to break into two equal size connected components, making the bisection problem trivial) and $p < 1 - 1/n^{2+o(1)}$ (otherwise, the complement of G_{rand} is likely to be bipartite and connected, making the bisection problem trivial).

Let $G \oplus e$ denote the graph that contains edge e iff G doesn't contain e, and is otherwise identical to G. Lemma 20 shows that with high probability,

$h(G \oplus e) = b(G \oplus e)$ for any e. This allows for the following search procedure: check for every edge $e = (1, i)$ whether $h(G \oplus e)$ differs from $h(G)$, determining whether vertices 1 and i are on different sides of the bisection. We use the fact that the bisection $(S, \bar{S})$ is with high probability the unique bisection of G.

Lemma 20 *Let G_{rand} be chosen as above, and let G be obtained from G_{rand} by a monotone adversary. Then with probability at least $1 - 1/n^{1-o(1)}$ over the choice of G_{rand}, $h(G) = b(G)$ and for every e, $h(G \oplus e) = b(G \oplus e)$.*

Proof: Fix e. It is easy to verify that $G \oplus e$ is derivable through a series of monotone transformations from either G_{rand} or $G_{\text{rand}} \oplus e$. In the former case, $h(G \oplus e) = b(G \oplus e)$ immediately; in the latter case it suffices to prove that $h(G_{\text{rand}} \oplus e) = b(G_{\text{rand}} \oplus e)$.

Assume that $q > 1/n^{2+o(1)}$ and $p < 1 - 1/n^{2+o(1)}$. Then the a priori probabilities that G_{rand} and $G_{\text{rand}} \oplus e$ are generated differ by a multiplicative factor of at most $n^{2+o(1)}$. It follows that $h(G_{\text{rand}} \oplus e) \neq b(G_{\text{rand}} \oplus e)$ with probability at most $n^{2+o(1)}(2n^{-5}) = 2n^{-3+o(1)}$. Taking the union bound over all $\binom{n}{2}$ possible choices for e, $h(G_{\text{rand}} \oplus e) = b(G_{\text{rand}} \oplus e)$ for all e with probability at least $1 - 1/n^{1-o(1)}$. $\square$

The algorithm implied by Lemma 20 requires n calls to h. A more efficient algorithm, based on a single computation of h, is implicit in the following theorem:

Theorem 21 *Let the matrix X be an optimal solution for the minimization SDP for h. Partition the vertices of G to those whose column has a positive entry in the first row of X, and those whose column has a negative entry in the first row of X. Then with high probability over the choice of G_{rand}, this gives the original partition $(S, \bar{S})$.*

Proof: Observe that X is also a feasible, though perhaps not optimal, solution for $G \oplus e$. Let $e = (i, j)$, where vertices i and j are on different sides of the bisection, and there is no edge in G between i and j. Lemma 20 then implies that $x_{ij} = -1$, as otherwise, $h(G \oplus e) < h(G) + 1$. Let $e = (i, j)$, where vertices i and j are on the same side of the bisection, and there is an edge in G between i and j. Lemma 20 then implies that $x_{ij} = 1$, as otherwise $h(G \oplus e) < h(G)$. The fact that $h(G) = b(G)$ then implies that $x_{ij} = -1$ whenever (i, j) is an edge connecting different sides of the bisection. The fact that the entries of X sum up to 0 and that $|x_{ij}| \leq 1$ then implies that $x_{ij} = 1$ whenever i and j are on the same side of the bisection. $\square$

5 Conclusions

We studied heuristics for three problems: independent set, k-coloring, and bisection. In all three cases, the driving force behind our heuristics was semidefinite programming. The heuristics were evaluated on semirandom models in which an input graph is chosen at random, and then undergoes "monotone" changes (that supposedly make the problem easier).

Semidefinite programs have been used for approximation algorithms for the above and similar problems. The current work provides observations that may be useful also in the context of approximation algorithms.

The random hyperplane technique was suggested by Goemans and Williamson [18] as a method of "rounding" the semidefinite solution so as to obtain a valid solution of the original problem. We have used this technique in our heuristic for independent set. For bisection we have not used this technique. Instead, we have used a deterministic rounding technique (in Theorem 21). It would be interesting to see if similar deterministic rounding techniques can be used in approximation algorithms for problems such as max-cut. We remark that the random hyperplane technique does recover the optimal bisection in our semirandom model (details omitted).

For the independent set, we note that our semirandom model does not exclude the possibility that the graph has independent sets larger than $|S|$. Nevertheless, our heuristic does recover S. This is somewhat surprising. Had we applied a semidefinite program only once, its solution may well have been completely determined by the subgraph induced on $\bar{S}$, giving us no clue as to where S is. However, in the first phase of our algorithm (the coarse partitioning), we keep using semidefinite programming to cut out parts of the graph until no independent set of size $|S|/4$ remains. This repeated use of semidefinite programming allows us to recover S. It would be interesting to see approximation algorithms based on repeated applications of semidefinite programming.

We remark that if $G_{\min}$ is defined so as to also include edges within $\bar{S}$ with probability p, then S is likely to be the largest independent set, and one application of a semidefinite program may suffice in order to recover it. This model was recently studied in [14].

Acknowledgements

Alan Frieze gave useful comments on an earlier version of this manuscript.

References

[1] F. Alizadeh. "Interior point methods in semidefinite programming with applications to combinatorial optimization", *SIAM J. Optimization*, 5(1), 13–51, 1995.

[2] N. Alon and N. Kahale. "A spectral technique for coloring random 3-colorable graphs". *SIAM J. Comput.*, 26(6), 1733–1748, 1997.

[3] N. Alon and N. Kahale. "Approximating the independence number via the ϑ-function". *Math. Programming* 80 (1998), 253–264.

[4] N. Alon, M. Krivelevich and B. Sudakov. "Finding a large hidden clique in a random graph". *Random Structures and Algorithms* 13 (1998), 457–466.

[5] N. Alon, J. Spencer, P. Erdos. The Probabilistic Method. Wiley, 1992.

[6] A. Blum and J. Spencer. "Coloring random and semirandom k-colorable graphs". *Journal of Algorithms*, 19(2):204–234, September 1995.

[7] Avrim Blum and David Karger. An $\tilde{O}(n^{3/14})$-coloring algorithm for 3-colorable graphs. *Information Processing Letters*, 61(1):49–53, 14 January 1997.

[8] R. Boppana. "Eigenvalues and graph bisection: An average-case analysis". In *Proceedings of the 28th Annual Symposium on Foundations of Computer Science*, pages 280–285, Los Angeles, CA, October 1987. IEEE Computer Society Press.

[9] R. Boppana and M. Halldorsson. "Approximating maximum independent sets by excluding subgraphs". *BIT, 32 (1992), 180–196.*

[10] David S. Johnson and Michael A. Trick (editors). "Cliques, Coloring, and Satisfiability: Second DIMACS Implementation Challenge, 1993", *DIMACS Series in Discrete Mathematics and Theoretical Computer Science, 26, American Mathematical Society, 1996.*

[11] W. Donath and A. Hoffman. "Lower bounds for the partitioning of graphs". *IBM Journal of Research and Development* 17, 420–425, 1973.

[12] U. Feige. Randomized graph products, chromatic numbers, and the Lovasz ϑ-function. *Combinatorica 17 (1) (1997) 79–90.*

[13] U. Feige and J. Kilian. Zero knowledge and the chromatic number. *Journal of Computer and System Sciences*, 57(2):187–199, October 1998.

[14] U. Feige and R. Krauthgamer. "Finding and certifying a large hidden clique in a semi-random graph". *Technical report MCS99-04 of the Weizmann Institute.* To appear in *Random Structures and Algorithms.*

[15] U. Feige, R. Krauthgamer and K. Nissim. "Approximating the minimum bisection size". *Manuscript*, October 1999.

[16] A. Frieze and C. McDiarmid. "Algorithmic theory of random graphs". *Random Structures and Algorithms 10 (1997), 5–42.*

[17] Z. Furedi and J. Komlos. "The eigenvalues of random symmetric matrices". *Combinatorica*, 1(3), 233–241, 1981.

[18] Michel X. Goemans and David P. Williamson. "Improved approximation algorithms for maximum cut and satisfiability problems using semidefinite programming", *Journal of the ACM*, 42 (6) (1995), 1115–1145.

[19] M. Halldórsson. A still better performance guarantee for approximate graph coloring. *Information Processing Letters*, 45(1):19–23, January 1993.

[20] J. Håstad. Clique is hard to approximate within $n^{1-\epsilon}$. *Proc. 37th Annual Symp. on Foundations of Computer Science*, pages 627–636, 1996.

[21] D. Karger, R. Motwani, and M. Sudan. "Approximate graph coloring by semidefinite programming." *Journal of the ACM*, 45(2) , 246–265, 1998.

[22] Lovász. On the Shannon capacity of a graph. *IEEE Transactions on Information Theory IT-25*, pp. 1-7, 1979.

[23] S. Poljak, F. Rendl. "Nonpolyhedral relaxations of graph-bisection problems". *SIAM J. Optimization*, 5(3), 467–487, 1995.

[24] C. Subramanian. "Minimum coloring random and semirandom graphs in polynomial expected time". *Proc. 36th Annual Symposium on Foundations of Computer Science, 1995, 463–472.*

A Technical lemmata

A.1 Proof of expansion property

Proof of Lemma 7: First, we observe that with high probability, each vertex $t \in \bar{S}$ has superconstant degree. The following bound is not the best possible but suffices for our purposes.

Proposition 22 *For some constant $\gamma > 0$ (depending on ϵ and α), $|N(t) \bigcap S| \geq \gamma \ln n$ for all $t \in \bar{S}$ with probability $1 - o(1)$ as n grows sufficiently large.*

The proof of Proposition 22 appears after the proof of Lemma 7.

For the rest of the proof of Lemma 7 we assume that the graph G is chosen as before, but conditioned on each vertex $t \in \bar{S}$ having degree $\gamma \ln n$ in S; by Proposition 22, this conditioning will not affect what happens with probability $1 - o(1)$.

We consider the bad event that $N(T) \subseteq U$ for some $T \subset \bar{S}$ of size $k \leq 31\alpha n/32d$ and $U \subset S$ of size dk. There are at most

$$\binom{n}{k} \cdot \binom{\alpha n}{dk} \leq \left(\frac{en}{k}\right)^k \left(\frac{e\alpha n}{dk}\right)^{dk}$$

possible bad events. By symmetry and assuming our lower bound on the degree of each vertex, we have that for any fixed U and $t \in \bar{S}$, $N(t) \subseteq U$ with probability at most $(dk/\alpha n)^{\gamma \ln n}$. Hence, $N(T) \subseteq U$ with probability at most $(dk/\alpha n)^{k\gamma \ln n}$. Setting $R = dk/\alpha n$, we have the product of the number of bad cases times the probability that any bad case occurs is at most

$$\left(\frac{en}{k}\right)^k e^{dk} R^{(\gamma \ln n - d)k}.$$

For every value of $1 \leq k \leq 31\alpha n/32d$, the above value is very small (e.g., asymptotically smaller than $1/n^2$). (For k that is close to $31\alpha n/32d$, use $R \leq 31/32$. For k that is substantially smaller, use $R = O(k/n)$.) This holds even when one sums over all $1 \leq k \leq 31\alpha n/32d$. The lemma follows. □

It remains to prove Proposition 22.

Proof of Proposition 22: Recall that the probability that a vertex $t \in \bar{S}$ is connected to a vertex $s \in S$ is $p = (1+\epsilon)\ln n/\alpha n$. Using the binomial formula and routine inequalities and simplifications we have that a vertex $t \in \bar{S}$ has degree Δ in S with probability

$$\binom{\alpha n}{\Delta} p^{\Delta}(1-p)^{\alpha n - \Delta} \leq \left(\frac{e}{\Delta}\right)^{\Delta} (1+\epsilon)^{\Delta}(\ln^{\Delta} n) n^{-(1+\epsilon)(1-\Delta/\alpha n)}.$$

Let $\Delta = \gamma \ln n$, for some constant $\gamma > 0$. We have that for any constant $\delta > 0$ we can set γ small enough so that

$$\begin{aligned} (e/\gamma)^{\gamma \ln n}, (1+\epsilon)^{\Delta} &\leq n^{\delta} \text{ and} \\ n^{-(1+\epsilon)(1-\Delta/\alpha n)} &\leq n^{-1-\epsilon+\delta}. \end{aligned}$$

Setting $\delta = \epsilon/4$, we have that a vertex has degree $\gamma \ln n$ with probability at most $n^{-1-\epsilon/4}$; by a simple monotonicity argument, the same bound holds for any $\Delta < \gamma \ln n$. The proposition then follows by invoking the union bound over bad events (t being of low degree in S). □

A.2 The Alon-Kahale partitioning algorithm

Proof of Lemma 10: Recall that a real symmetric matrix X is *positive semidefinite* iff there is some matrix Y such that $X = YY^T$, where Y^T denotes the transpose of Y. Consider the following semidefinite program (which can be solved up to arbitrary precision in polynomial time, using the Ellipsoid algorithm). Given the graph G, we find an order n matrix $X = \{x_{ij}\}$ satisfying the following constraints:

1. $\forall i, j,\ 0 \leq x_{ij} \leq 1$,
2. $\forall (i,j) \in E,\ x_{ij} = 0$,
3. $\sum_i x_{ii} = K$,

4. $\forall i$, $\sum_j x_{ij} = Kx_{ii}$,

5. The matrix X is positive semidefinite.

As G has an independent set S of size K, the above semidefinite program is feasible. Setting $x_{ij} = 1$ whenever $v_i, v_j \in S$, and $x_{ij} = 0$ otherwise, gives a matrix X satisfying the above constraints. To see that X is semidefinite, let q_i be 1 if $v_i \in S$, and 0 otherwise, and observe that $X = YY^T$, where $Y = [q_1\ q_2 \cdots q_n]^T$.

A positive semidefinite matrix X can be decomposed in polynomial time into $X = YY^T$, where Y is a matrix with n rows. We denote the row vectors of Y by $y_1, \ldots, y_n$. The entry x_{ij} is the inner product $\langle y_i, y_j \rangle$. Let $y_0 = \sum_i y_i/K$. From constraints 3 and 4 it follows that $\langle y_0, y_0 \rangle = K^2/K^2 = 1$. Hence y_0 is a unit vector. Moreover, constraint 4 implies that for every i, $\langle y_i, y_i \rangle = \langle y_0, y_i \rangle$. (Geometrically, this means that the points y_i, $1 \le i \le n$, all lie on an n-dimensional sphere of radius $1/2$, and the points 0 and y_0 are antipodal.)

Assume w.l.o.g. that the vectors y_i are sorted by their lengths, and consider now only the vectors $y_1, \ldots, y_{K/2}$. For every such vector, $K/2n < \langle y_i, y_i \rangle \le 1$ (from constraints 1 and 3). Associate now with the vertices $v_1, \ldots, v_{K/2}$ unit vectors $z_1, \ldots, z_{K/2}$, where vector z_i is in direction $y_i - \langle y_0, y_i \rangle y_0$ (i.e., we project out direction y_0).

Lemma 10 then follows from Proposition 23. □

Proposition 23 *For $1 \le i < j \le K/2$, if $(v_i, v_j) \in E$, then*

$$\langle z_i, z_j \rangle < -K/(2n - K).$$

Proof: Let $w_i = y_i - \langle y_0, y_i \rangle y_0$. If $(v_i, v_j) \in E$ then $\langle y_i, y_j \rangle = 0$. This gives

$$\begin{aligned} \langle w_i, w_j \rangle &= 0 - 2\langle y_0, y_i \rangle \langle y_0, y_j \rangle + \langle y_0, y_0 \rangle \langle y_0, y_i \rangle \langle y_0, y_j \rangle \\ &= -\langle y_0, y_i \rangle \langle y_0, y_j \rangle. \end{aligned}$$

This implies that $\langle z_i, z_j \rangle < 0$, but does not bound its magnitude, since the w_i are not unit vectors. By using the interpretation of the y_i as lying on a sphere of radius $1/2$, and considering their projections in the direction of y_0 and on the hyperplane orthogonal to y_0, we obtain that their lengths satisfy $(|w_i|)^2 + (1/2 - \langle y_0, y_i \rangle)^2 = 1/4$, implying that $|w_i| = \sqrt{\langle y_0, y_i \rangle - (\langle y_0, y_i \rangle)^2}$. Hence $z_i = w_i / \sqrt{\langle y_0, y_i \rangle - (\langle y_0, y_i \rangle)^2}$. It follows that

$$\langle z_i, z_j \rangle = \\ -\frac{\langle y_0, y_i \rangle}{\sqrt{\langle y_0, y_i \rangle - (\langle y_0, y_i \rangle)^2}} \cdot \frac{\langle y_0, y_j \rangle}{\sqrt{\langle y_0, y_j \rangle - (\langle y_0, y_j \rangle)^2}}.$$

The function $f(x) = x/\sqrt{x - x^2} = 1/\sqrt{x^{-1} - 1}$ is positive and monotone increasing over $(0, 1)$; hence the absolute value of $\langle z_i, z_j \rangle$ is minimized when $\langle y_0, y_i \rangle$

and $\langle y_0, y_j\rangle$ are minimized. As these minimums are in both cases more than $K/2n$, it follows that $\langle z_i, z_j\rangle < -(K/2n)^2/(K/2n - (K/2n)^2) = -K/(2n-K)$. $\square$

A.3 A bound on the eigenvalues of random graphs

In this section we complete the proof of Lemma 19. This requires evaluating the eigenvalues of a random matrix M. We first review and develop some tools that will be used in our analysis.

A symmetric matrix is positive semidefinite if and only if all its eigenvalues are nonnegative. We denote the eigenvalues of a matrix by $\lambda_1 \geq \lambda_2 \geq \ldots \geq \lambda_n$ and a corresponding eigenvector basis by $e_1, \ldots, e_n$ (which may not be unique). We shall use the following known theorem, whose proof we sketch for completeness.

Theorem 24 *Let A and B be two symmetric matrices of order n. Then $\lambda_n(A) + \lambda_i(B) \leq \lambda_i(A+B) \leq \lambda_1(A) + \lambda_i(B)$.*

Proof: We start with some basic facts about eigenvalues. For a symmetric matrix C of order n, its eigenvalues satisfy

$$\lambda_i(C) \geq \min_{x\in F} \frac{x^t C x}{x^t x}, \tag{1}$$

where F is any subspace of dimension i and $x \neq 0$. Furthermore, equality is achieved if F spans $e_1(C), \ldots, e_i(C)$ (with the minimum achieved by $x = e_i$). Similarly,

$$\lambda_i(C) \leq \max_{x\in F} \frac{x^t C x}{x^t x}, \tag{2}$$

where F is any subspace of dimension $n+1-i$ and $x \neq 0$. Equality is achieved if F spans $e_i(C), \ldots, e_n(C)$ (with the maximum achieved by $x = e_i$).

Now, for any x,

$$\frac{x^t(A+B)x}{x^t x} = \frac{x^t A x}{x^t x} + \frac{x^t B x}{x^t x}.$$

If we let F be the subspace generated by x, then by Equations 1 and 2 we have that

$$\lambda_n(A) \leq \frac{x^t A x}{x^t x} \leq \lambda_1(A)$$

for any x. Taking F to be spanned by $e_1(B), \ldots, e_i(B)$, we have $\lambda_i(B) \leq \frac{x^t B x}{x^t x}$ for all $x \in F$ (by the remark following Equation 1). Hence, choosing $x \in F$ so as to minimize $\frac{x^t(A+B)x}{x^t x}$, we have that

$$\begin{aligned} \lambda_i(A+B) &\geq \frac{x^t(A+B)x}{x^t x} \\ &\geq \lambda_n(A) + \lambda_i(B). \end{aligned}$$

Similarly, taking F to be spanned by $e_i(B), \ldots, e_n(B)$, we have $\lambda_i(B) \geq \frac{x^t B x}{x^t x}$ for all $x \in F$. Choosing $x \in F$ so as to maximize $\frac{x^t(A+B)x}{x^t x}$, we have

$$\begin{aligned} \lambda_i(A+B) &\leq \frac{x^t(A+B)x}{x^t x} \\ &\leq \lambda_1(A) + \lambda_i(B). \end{aligned}$$

□

We now analyze the eigenvalues of adjacency matrices of random graphs. Let A be the adjacency matrix of an undirected graph. Then A is symmetric and all its eigenvalues are real. λ_1 is positive and has largest absolute value among all eigenvalues. If the graph is bipartite, then for every i, $\lambda_i(A) = -\lambda_{n-i+1}(A)$. Let $Tr(A)$ denote the *trace* of A, i.e., the sum of entries along its main diagonal. Then $\sum_{i=1}^n \lambda_i(A) = Tr(A)$. The eigenvalues of A^k are $(\lambda_i(A))^k$.

Proposition 25 *Let A be the adjacency matrix of a graph with m edges. Then $\lambda_1(A) \leq \sqrt{2m}$.*

Proof: Observe that $Tr(A^2) = 2m$. Hence $2m = \sum_{i=1}^n (\lambda_i(A))^2 \geq (\lambda_1(A))^2$, and the proof follows. □

A *closed walk* of length k in graph G is a sequence of $(k+1)$ vertices, starting and ending at the same vertex, such that every two consecutive vertices in the sequence are adjacent in G.

Proposition 26 *Let G be a graph with n vertices and m edges. Then the number of closed walks of length k in G is at most $n(2m)^{k/2}$.*

Proof: Let A be the adjacency matrix of G. Observe that $Tr(A^k)$ is the number of closed walks of length k in G. Also, $Tr(A^k) = \sum_{i=1}^n (\lambda_i(A))^k \leq n(\lambda_1(A))^k$. The proof now follows from proposition 25. □

Theorem 27 *Let c be a sufficiently large constant, let $a > 0$ be an arbitrary constant, and let n be sufficiently large. Let B be the adjacency matrix of a random graph on n vertices in which each edge has probability p, where $1/n \leq p \leq (n-1)/n$. Then with probability at least $1 - n^{-a}$ over the choice of B, $\max[|\lambda_2|, |\lambda_n|] \leq \max[c\sqrt{pn \log n}, c\sqrt{(1-p)n \log n}]$. Let C be the adjacency matrix of a random bipartite graph on $n/2 + n/2$ vertices in which each edge connecting the two parts has probability p. Then with probability at least $1 - n^{-a}$ over the choice of C, $\max[|\lambda_2|, |\lambda_{n-1}| \leq \max[c\sqrt{pn \log n}, c\sqrt{(1-p)n \log n}]$.*

Proof: Consider a matrix B as in the theorem and assume $p \leq 1/2$. Let J be the all 1 matrix and consider the matrix $A = B - pJ + pI$. Then $B = A + pJ - pI$. As the only nonzero eigenvalue of J is n, we obtain from Theorem 24 that $\lambda_n(B) \geq \lambda_n(A) - p$ and $\lambda_2(B) \leq \lambda_1(A) - p$.

The matrix A is a random matrix. Each entry has expectation 0, and variance $p(1-p)$ (except for entries on the diagonal which are all 0). Let

$\lambda(A) = \max[|\lambda_1(A)|, |\lambda_n(A)|]$. As A is random, we shall compute the expectation $E[\lambda]$ over the choice of random A. Let k be an even integer (we will later choose $k = \Theta(\log n)$). Then $(E[\lambda])^k \leq E[\lambda^k] \leq E[Tr(A^k)]$.

The trace of A^k is computed by summing over all closed walks of length k on the complete graph, and for each such walk associating a weight equal to the product of the entries for the corresponding edges in A (taking into account the number of times each edge appears in the walk). For $E[Tr(A^k)]$, this computation is simplified by the fact that if a walk crosses an edge exactly once, then the contribution of the edge to the expectation of the weight is 0 (as this is the expected value of each entry of A), making the expected weight of the walk 0. If an edge is crossed twice by a walk, its expected contribution to the product is its variance $p(1-p) \leq p$. The upper bound of p on the contribution applies also when an edge participates more than twice in a walk.

As we may assume that edges are crossed by the walk either at least twice or not at all, then the total number of distinct edges crossed by the walk, which we denote by l, satisfies $l \leq k/2$. As the edges of the walk induce a connected graph, the number of vertices in this graph, denoted by t, satisfies $t \leq l+1$.

We now parameterize the walks by t and l. (In the following, we omit some multiplicative terms that are $(O(1))^k$, as they influence the final result only by a multiplicative constant factor.)

The number of possible induced subgraphs. A walk visits a subgraph of G. There are $\binom{n}{t}$ ways of choosing the vertices. As the walk induces a connected graph, it has a spanning tree. $t-1$ of the edges belong to the spanning tree, and there are t^{t-2} possible spanning trees on t vertices. There are at most $(t^2)^{l-t+1}$ ways of choosing the remaining edges. Altogether, the number of possible subgraphs is bounded by $n^t t^{2(l-t)} e^t$.

The number of walks. As seen from Proposition 26, the number of walks of length k can be bounded by $t(2l)^{k/2}$.

Expected weight of a walk. The expected weight of a walk that crosses l edges is at most p^l.

Let $E(k,t,l)$ denote the sum of expected weight of all closed walks of length k that visit t vertices and l edges. Then we have:

$$E(k,t,l) \leq n^t t^{2(l-t)} e^t t (2l)^{k/2} p^l \simeq e^t (\frac{n}{t^2})^t (t^2 p)^l (2l)^{k/2} t$$

Let $E(k,l)$ denote the sum of expected weight of all closed walks of length k that visit l edges. When $t = O(\log n)$ and n is large enough, then the expression $(n/t^2)^t$ is increasing with t. Hence for fixed k and l, $E(k,t,l)$ increases (more than) geometrically with t, implying that

$$\begin{aligned} E(k,l) &\leq 2E(k, t=l+1, l) \\ &\leq 2e^{l+1} (\frac{n}{(l+1)^2})^{l+1} ((l+1)^2 p)^l (2l)^{k/2} \frac{1}{l+1} \end{aligned}$$

$$\leq \quad e^{l+1}n^{l+1}p^l(2l)^{k/2}$$

When $p \geq 1/n$, $E(k,l)$ is maximized when l is maximized. We obtain:

$$E[Tr(A^k)] \leq \frac{k}{2}E(k,l=k/2) \leq \frac{k}{2}e^{k/2+1}n^{k/2+1}p^{k/2}k^{k/2}$$

Taking the kth root and noting that $k = \Theta(\log n)$, we obtain that $E[\lambda] \leq O(\sqrt{pn\log n})$. The probability of λ exceeding this upper bound on the expectation by a factor of 2 can be made an arbitrarily small inverse polynomial in n when n is large enough, by using Markov's inequality on $E[\lambda^k]$ (with $k = \Theta(\log n)$).

This completes the proof for the case that B is the adjacency matrix of a random graph and $p \leq 1/2$. When $p > 1/2$ we bound the variance $p(1-p)$ by $(1-p)$ rather than by p, and obtain the claimed bound. For the case that C is the adjacency matrix of a random bipartite graph, let D be a matrix with entries $d_{ij} = 0$ whenever $i,j \in S$ or $i,j \in \bar{S}$, and 1 otherwise. The two nonzero eigenvalues of D are $+n/2$, $-n/2$. Let $A = C - pD$. Then $\lambda_2(C) \leq \lambda_1(A)$ and $\lambda_{n-1}(C) \geq \lambda_n(A)$. The upper bound on $E[Tr(A^k)]$ applies as before, and we get the results claimed. $\square$

Remark: Our proof of Theorem 27 follows that of [17] who showed that for arbitrary (fixed) p, when n is large enough, all but the largest eigenvalue can be bounded by $O(\sqrt{pn})$. Our analysis looses an extra factor of $\sqrt{\log n}$ because we also deal with values of p as small as $1/n$. It is not clear to us for what range of values of p the extra $\sqrt{\log n}$ factor becomes unnecessary. Theorem 4.2 in [8] implies that $p \geq \Omega(\frac{\log n}{n})$ suffices, but the proof of that theorem has not been published.

We are now ready to continue the proof of Lemma 19.

Proof of Lemma 19 (continued): When $y_0 = -1$, we can rewrite $-A + J$ as $A^c + I$, where A^c is the adjacency matrix of the complement of the graph G_{rand}, and I is the identity matrix. Hence $M = A^c + I - Y$. The eigenvalues of I are all 1. In Propositions 28 and 29 we show that given our bound on $p - q$ the following eigenvalue bounds each hold with probability at least $1 - n^{-5}$ for c and n sufficiently large:

$$\begin{aligned} \lambda_n(-Y) &\geq (c/4)\sqrt{pn\log n} \\ \lambda_{n-1}(A^c) &\geq -(c/4)\sqrt{pn\log n}. \end{aligned}$$

Hence, both inequalities simultaneously hold with probability at least $1 - 2n^{-5}$. As $M = A^c + I - Y$ we have $\lambda_{n-1}(M) \geq \lambda_{n-1}(A^c) + \lambda_n(I) + \lambda_n(-Y)$ by repeated applications of Theorem 24. We thus obtain $\lambda_{n-1}(M) \geq -(c/4)\sqrt{pn\log n} + 1 + (c/4)\sqrt{pn\log n} > 0$ for sufficiently large n. As 0 is also an eigenvalue of M, it follows that $\lambda_n(M) = 0$, and the matrix is positive semidefinite. $\square$

It remains to prove Propositions 28 and 29. For both propositions, we assume that

$$p - q \geq c\sqrt{p\frac{\log n}{n}}$$

for some large enough constant c.

Proposition 28 *For n sufficiently large, $\lambda_n(-Y) \geq (c/4)\sqrt{pn\log n}$ with probability $1 - n^{-5}$ over the choice of G_{rand}.*

Proof: The matrix $-Y$ is a diagonal matrix. Hence its diagonal entries, $-y_i$, are its eigenvalues. To bound $-y_i$ w.h.p., we compute the number of neighbors that vertex i has on each side of the bisection.

The expected number of neighbors of i on the other side of the bisection is $pn/2$. The probability of having less than $pn/2 - a$ such neighbors is at most $e^{-a^2/pn}$ (see for example Theorem A.13 in [5]). The expected number of neighbors of i on the same side of the bisection is $qn/2$ (for simplicity of notation, we also assume here that i has a self loop with probability q). The probability of having more than $qn/2 + a$ such neighbors is at most $e^{-a^2/qn + 2a^3/(qn)^2}$ (see for example Theorem A.11 in [5]). For simplicity in the manipulations that follows, we replace this last upper bound by the upper bound $e^{-a^2/pn+2a^3/(pn)^2}$ that assumes (wrongly) that q is as large as p. This upper bound is still correct, because the larger q is, the higher the probability of exceeding the expectation by a. Summing up, $-y_i > (p-q)n/2 - 2a$ with probability at least $1 - (e^{-a^2/pn} + e^{-a^2/pn+2a^3/(pn)^2})$. Recall that $p - q \geq c\sqrt{p\frac{\log n}{n}}$. Taking $a = (c/8)\sqrt{pn\log n}$ we get that $y_i > (c/4)\sqrt{pn\log n}$ with probability at least $1 - n^{-6}$, when c is a large enough constant (the 6 can be replaced by any constant by making c sufficiently large). Taking the union bound over all i, the proposition is proved. $\square$

Proposition 29 *For n sufficiently large, $\lambda_{n-1}(A^c) \geq -(c/4)\sqrt{pn\log n}$ with probability $1 - n^{-5}$ over the choice of G_{rand}.*

Proof: The matrix A^c is the adjacency matrix of G^c, the edge-complement of the graph G_{rand}. Choosing at random a graph G^c with the right distribution can be done using the following process. For convenience, we identify graphs with their adjacency matrices. Choose a random graph B in which every pair of vertices of S are connected with probability $(1-p)$. Choose a random bipartite graph C (with bipartition $(S, \bar{S})$) in which each cut edge has probability $(1-q)$. Choose a random graph D in which every pair of vertices of $\bar{S}$ are connected with probability $(1-p)$. Then $A^c = B + C + D$.

Theorem 24 implies that $\lambda_{n-1}(A^c) \geq \lambda_n(B) + \lambda_{n-1}(C) + \lambda_n(D)$. Taking the constant c in Lemma 19 to be 12 times as large as the constant c in Theorem 27, and substituting in the bounds from this latter theorem, the proposition is proved. $\square$