Journal of Algorithms **42,** 153–172 (2002)
doi:10.1006/jagm.2001.1197, available online at http://www.idealibrary.com on IDEAL®

General Partitioning on Random Graphs[1]

C. R. Subramanian

The Institute of Mathematical Sciences, C.I.T. Campus, Taramani, Chennai, 600 113, India
E-mail: crs@imsc.ernet.in

and

C. E. Veni Madhavan

Department of Computer Science and Automation, Indian Institute of Science, Bangalore, 560 012, India
E-mail: cevm@csa.iisc.ernet.in

Received November 29, 2000

Consider the general partitioning (GP) problem defined as follows: Partition the vertices of a graph into k parts $W_1, \dots, W_k$ satisfying a polynomial time verifiable property. In particular, consider properties (introduced by T. Feder, P. Hell, S. Klein, and R. Motwani, *in* "Proceedings of the Annual ACM Symposium on Theory of Computing (STOC '99), 1999" and) specified by a pattern of requirements as to which W_i forms a sparse or dense subgraph and which pairs W_i, W_j form a sparse or dense or an arbitrary (no restriction) bipartite subgraph. The sparsity or density is specified by upper or lower bounds on the edge density $d \in [0, 1]$, which is the fraction of actual edges present to the maximum number of edges allowed. This problem is NP-hard even for some fixed patterns and includes as special cases well-known NP-hard problems like k-coloring (each $d(W_i) = 0$; each $d(W_i, W_j)$ is arbitrary), bisection ($k = 2$; $|W_1| = |W_2|$; $d(W_1, W_2) \leq b$), and also other problems like finding a clique/independent set of specified size. We show that GP is solvable in polynomial time almost surely over random instances with a planted partition of desired type, for several types of pattern requirement. The algorithm is based on the approach of growing BFS trees outlined by C. R. Subramanian (*in* "Proceedings of the 8th Annual European Symposium on Algorithms (ESA '00), 2000," pp. 415–426). © 2002 Elsevier Science

[1]This research was partially supported by the EU ESPRIT LTR Project 20244 (ALCOM-IT), WP 3.3. A part of this research was done during the first author's visit to IISc, Bangalore.

0196-6774/02 $35.00

Key Words: partitioning problems; graph coloring; bisection; max-cut; min-k-section; random graphs; polynomial average time algorithms; probabilistic analysis; combinatorial probability; analysis of algorithms and problem complexity; complexity classes.

1. INTRODUCTION

Consider the general partitioning problem (GP) defined as follows: Partition the vertices of a graph into k parts $W_1, \dots, W_k$ satisfying a polynomial time verifiable property. In particular, consider properties (introduced in [6] and) specified by a pattern of requirements as to which W_i forms a sparse or dense subgraph and which pairs W_i, W_j form a sparse or dense or an arbitrary (no restriction) bipartite subgraph. The sparsity or density is specified by upper or lower bounds d_i on the edge density $d(W_i) \in [0, 1]$, which is the fraction of actual edges present to the maximum number of edges allowed. Similarly, we specify bounds on the density of edges joining different parts. Even the special cases (fixing the pattern of requirements) of GP are NP-hard. When each $d(W_i) = 0$ and each $d(W_i, W_j)$ is arbitrary, this specializes as the k-coloring problem. If $k = 2$, $|W_1| = |W_2|$, $d(W_1, W_2) \leq b$, this becomes the graph bisection problem. Hence, GP is a NP-hard problem even for some fixed pattern of requirements. We show that, with some assumptions on densities, one can almost surely find such a partition, provided the input is a random graph with a planted partition of this type. An algorithmic study of random instances of such hard problems could be useful in practical situations. In addition, studies on the average case analysis of NP-hard problems have certain implications to cryptography (see [11, 12]).

A random model for the GP problem can be obtained as follows: Consider a graph on kn vertices obtained by initially planting a partition $(V_1, \dots, V_k)$ of V into k parts of size n each. For i, choose each edge lying within V_i with probability $p_i < d_i$ (or $p_i > d_i$) if V_i is required to be sparse (or dense). Similarly, for each $i \neq j$, choose each edge joining V_i and V_j with probability $p_{i,j} < d_{i,j}$ or $p_{i,j} > d_{i,j}$, depending on the requirement. For a random graph drawn in this fashion, the planted partition (V_i) satisfies the requirements of the problem, almost surely, provided the bounds are not "too small" and the probabilities are *sufficiently* away from their respective bounds. For a concrete example, assume that edges of V_1 are chosen with probability $p \leq 0.8d'$ and $d' \geq (\log n)^2/\binom{n}{2}$. Then, by Chernoff bounds (see Theorem 4.3, Chapter 4 of [13]), it follows that

$$\Pr(d(V_1) > d') \leq e^{-(\log n)^2/48} = o(n^{-\omega(1)}).$$

We can further tighten the gap between p and d'. Similarly, we can show that other parts (or pairs of parts) also satisfy appropriate density requirements with high probability.

In this paper, we consider the following simpler model in which each edge lying within a part V_i is chosen with probability p and each crossing edge is chosen with probability q. We denote this model by $\mathcal{G}(n, k, p, q)$. This model can be used to obtain special cases of the general model by introducing appropriate upper or lower bounds on the edge densities of the parts or pairs of parts. With $p = 0$, this becomes a random model for k-colorable graphs. When $k = 2$ and $p > q$, this becomes a random model for the problem of bisection.

If p and q are *sufficiently* apart (this will be precisely quantified later), we show, given $G \in \mathcal{G}(n, k, p, q)$, how to find a partition $W_1, \ldots, W_k$ such that each $d(W_i)$ obeys the required bound on it in polynomial time. Thus, V has been partitioned into k sparse or dense sets. In fact, the algorithm will output a polynomial sized collection of partitions such that, with high probability, one of these is the actual partition $V_1, \ldots, V_k$. For each partition in the collection, we check if it is of required type and output if it is so. Since (V_i) is, almost surely, a partition of required type, the algorithm solves the random instance. The class of distributions supported by the algorithm is determined by the ranges of p and q. Our results also carry over to the model in which the vertices are uniformly randomly put into one of the k partite sets.

The algorithm is based on the BFS approach outlined by Subramanian [15] and uses the following intuition. In such a random graph, consider any vertex x belonging to V_1 and grow the BFS tree rooted at x. We prove that, with high probability, the following holds: The tree grows by a factor of "roughly" $n(p + (k-1)q)$ between successive levels till it crosses the "threshold" of $1/(p + (k-1)q)$. Also, in every level, the contribution of V_1 is either larger (or smaller) than that of any V_i, $i > 1$, by a significant amount. We make use of this phenomenon to separate V_1. Similarly, other V_i's are separated. This idea has been successfully applied to k-color in polynomial average time random k-colorable graphs [15]. This approach is conceptually simple and combinatorial in nature. We show that our algorithms have very small failure probabilities which are, for a large class of distributions, asymptotically much smaller than n^{-D} for any fixed D.

Intuition. We give an informal sketch of the growth of the BFS tree. To keep it brief, we only argue using expectations as actual values for various random variables. For simplicity, assume that (i) $k = 3$, (ii) $p < q$ and are sufficiently apart, (iii) $n(p + 2q) = \omega(1)$, and (iv) 2 is the smallest positive integer l such that $(n(p + 2q))^l \geq 1/(p + 2q)$. Let x be a vertex in V_1 and y is any other vertex. Grow the BFS tree in $G - y$ starting from x.

Let $n_i(l')$, $i = 1, 2$, $l' \leq 2$, denote the size (number of vertices) of the l'th level of the BFS tree contributed by V_i. We have $n_1(1) = np$ and $n_2(1) = n_3(1) = nq$ and $\sum_i n_i(1) \approx n(p + 2q)$. Given that this happens, using $n(p+2q) = o(1/(p+2q))$ and Fact 1.1, we have $n_1(2) \approx n^2(p^2 + 2q^2)$ and $n_2(2) \approx n_3(2) \approx n^2(2pq + q^2)$ and $\sum_i n_i(2) \approx n^2(p + 2q)^2$. As a result, y has approximately $n^2(p^3 + 6pq^2 + 2q^3)$ neighbors if $y \in V_1$ and has approximately $n^2(3p^2q + 3pq^2 + 3q^3)$ neighbors if $y \notin V_1$. Hence y has more neighbors if $y \notin V_1$ than otherwise. This helps us to correctly place y with high confidence. Several technical issues (like using expectations as actual values) have been ignored in this sketch and these are taken care of in the analysis.

Outline. In Sections 2 and 3, we assume that both p and q are above some value. With this assumption, we obtain tight estimates on the structure and growth of the BFS tree in Section 2 and in Section 3, we show how to retrieve the planted partition $(V_1, \ldots, V_k)$. In Sections 4 and 5, we remove this assumption and do the same (although we do not provide tight estimates). In Sections 6–11, we apply these results to show how to solve several NP-hard partitioning problems on random instances.

1.1. *Assumptions, Notions, and Tools*

The following assumptions and notations are valid for the rest of the paper.

a1 The probabilities p and q satisfy the following: $p + (k-1)q = n^{-1+\epsilon}$, $\epsilon \geq X/\sqrt{\log n}$, for some sufficiently large positive constant X. We use S (for sum) to denote $p + (k-1)q$ and D (for difference) to denote $p - q$.

a2 α denotes the value $\sqrt{\log n}/\sqrt{2}^{\sqrt{\log n}}$ $\cdot$ α is used as a common bound (independent of p, q) on the *relative* deviation of a random variable from its expectation. This choice of α is tentative. Ideally, we should make α to vary inversely with the value of S.

a3 As in [15], given $p + (k-1)q$, we can find a *unique positive* integer l so that we can write $p + (k-1)q = n^{-1+(1/l+1)+\delta}$ where δ is a positive *real* such that $\tau(l) \leq \delta < \frac{1}{l(l+1)} + \tau(l-1)$ and $\tau(l') \doteq \frac{1}{\sqrt{X}l'(l'+1)}$ for $l' \geq 1$ and $\tau(0) = 0$. To find the value of l exactly, partition $(0, 1)$ into $I_1 \cup I_2 \cup \cdots$ where, for each $l \geq 1$, $I_l \doteq [f(l), f(l-1))$ with $f(l) = \frac{1}{l+1}(1 + \frac{1}{l\sqrt{X}})$ for $l \geq 1$ and $f(0) = 1$. Given any $\epsilon > 0$, it falls into one of these subintervals and the index of this interval is used as the value of l. This meaning of l holds for the rest of the paper. l differs (by at most 1) from the smallest positive value of j such that $(n(p + (k-1)q))^j = \omega(1/(p + (k-1)q))$. We use l (with the above-stated meaning) instead of this smallest j so as to take care of certain technical difficulties arising in the analysis.

Since $p + (k-1)q \geq n^{-1+\epsilon}$ with $\epsilon \geq X/\sqrt{\log n}$, we have $l \leq 2\sqrt{\log n}/X$. Because of this, we have $(1-\alpha)^l = 1 - l\alpha[1-o(1)]$ and $(1+\alpha)^l = 1 + l\alpha[1+o(1)]$. Note that $\tau(l)$ and the upper bound on δ both become smaller as l becomes larger. It would be useful to note the following facts involving p, q, and l.

$$(nS)^{l-1} = S^{-1}n^{l\delta-1/(l+1)}$$
$$= o\left(\frac{\alpha}{S}\right) \quad \text{for} \quad \tau(l) \leq \delta \leq \frac{\sqrt{X}-1}{\sqrt{X}l(l+1)}. \tag{1}$$
$$(nS)^l = S^{-1}n^{(l+1)\delta}. \tag{2}$$
$$n^l S^{l+1} = n^{(l+1)\delta}. \tag{3}$$

Notation. Often, where it is convenient, we use $e_1 \in (f_l, f_u)e_2$ as a short notation for $f_l e_2 \leq e_1 \leq f_u e_2$. Here, f_l, f_u, e_1, and e_2 are real-valued expressions.

Fact 1.1. Let $S(n)$ be a set of vertices of size $M(n)$. For any $u \notin S(n)$, let each edge $\{u, s\}$ $(s \in S(n))$ be selected independently with probability $p(n)$. If $M = M(n)$ and $p = p(n)$ are such that $Mp = o(1)$ as $n \to \infty$, then

$$\Pr(u \text{ is adjacent to some vertex in } S(n)) = Mp - M^2p^2[1-o(1)]/2$$
$$= Mp[1-o(1)].$$

Fact 1.2. For any small $c > 0$, there exists a value X^* of X such that for any $\epsilon \geq X^*/\sqrt{\log n}$, the corresponding l satisfies $\alpha^{1/(l+1)} \leq c$.

We often use the following bounds (due to Chernoff) on tail probabilities of Poisson trials (see Chapter 4 of [13]).

THEOREM 1.1. *Let $X_1, \ldots, X_n$ be independent Poisson trials such that, for each i,* $\Pr[X_i = 1] = p_i$, *where* $0 < p_i < 1$. *Then, for* $X = \sum_{1\leq i\leq n} X_i$, $\mu = E[X] = \sum_{1\leq i\leq n} p_i$, *and* $0 < \delta < 1$,

$$\Pr[X < (1-\delta)\mu] < e^{-\mu\delta^2/2}, \qquad \Pr[X > (1+\delta)\mu] < e^{-\mu\delta^2/3}$$
$$\Pr[|X-\mu| > \delta\mu] < 2e^{-\mu\delta^2/3}.$$

2. STRUCTURE OF THE BFS TREE FOR $p, q \geq (\log n)2^{\sqrt{\log n}}/n$

We study the structure and growth of BFS trees in G with the stated lower bound on p and q. Recall the meaning of l given before. Let $x, w \in V$ with

$w \neq x$ be any two arbitrary but fixed vertices. Assume, w.l.o.g., that $x \in V_1$. Consider the BFS tree in $G - w$ grown from x. For $j \geq 1$ and $0 \leq l' \leq l$, define $n_j(x, w, l')$ to be the number of vertices in the l'th level of the tree which are also in V_j, that is, the number of vertices in $(V - w) \cap V_j$ which are at a distance of l' from x in $G - w$.

Consider the following expressions defined for each level $l' \geq 0$. These are defined by

$$F_1(0) = 1, F_2(0) = 0$$

$$F_1(l') = \frac{n^{l'}}{k}(S^{l'} + (k-1)D^{l'}) \qquad \text{for } l' \geq 1$$

$$F_2(l') = \frac{n^{l'}}{k}(S^{l'} - D^{l'}) \qquad \text{for } l' \geq 1.$$

One can verify that

$$F_1(l') - F_2(l') = (nD)^{l'} \quad \text{and} \quad F_1(l') + (k-1)F_2(l') = (nS)^{l'} \qquad \forall\, l'.$$

F_1 captures the expected contribution of V_1 to each level in the BFS tree. Similarly, F_2 captures the expected contribution of each V_j, $j > 1$, in the BFS tree. We have the following lemma.

LEMMA 2.1. *Assume that* (a1) *holds and that* $\tau(l) \leq \delta \leq \frac{\sqrt{X}-1}{\sqrt{X}l(l+1)}$. *Then, for x and w mentioned before, with probability at least* $1 - e^{-\Omega(\alpha^2 \min(np, nq))}$, *for all* $j > 1$,

$$n_1(x, w, l) \in ((1-\alpha)^l, (1+\alpha)^l)F_1(l) \tag{4}$$

$$n_j(x, w, l) \in ((1-\alpha)^l, (1+\alpha)^l)F_2(l). \tag{5}$$

Proof. We prove a more general statement. We will prove that (4) and (5) are true for each $0 \leq l' \leq l$. The proof is by induction on l'.

Recall that $n_j(x, w, l')$, shortly $n_j(l')$, denotes the contribution of V_j to l'th level of the BFS tree starting at x in $G - w$. We have $n_1(0) = 1$ and $n_j(0) = 0$ for $j > 1$. Thus, the quantities $F_1(0), F_2(0)$ satisfy (4) and (5) with probability 1.

Also, by Chernoff bounds, with probability at least $1 - 2e^{-\alpha^2 np/3}$,[2]

$$(1)\ n_1(1) \in ((1-\alpha), (1+\alpha))F_1(1).$$

Similarly, for each $j > 1$, with probability at least $1 - 2e^{-\alpha^2 nq/3}$,

$$(\mathrm{j})\ n_j(1) \in ((1-\alpha), (1+\alpha))F_2(1).$$

[2]The requirement that $p, q \geq (\log n)2^{\sqrt{\log n}}/n$ is to ensure that (j), $j \geq 1$, hold with high probability.

Hence, with probability at least $1 - 2ke^{-\alpha^2 \min(np, nq)/3}$, (j), $j = 1, \ldots, k$, simultaneously hold, considering only *potential* edges incident on x.

Now, assume that $l' \leq l - 1$ and that the lemma is satisfied for all values $\leq l'$ with required probability considering only potential edges, each of which is incident on a vertex in some level before level l'. Given that (4) and (5) are satisfied up to level l', we bound $n_j(l' + 1)$ for all $j \geq 1$. Consider any $y \in V_j - \{x, w\}$, $j \geq 1$, which is not yet visited by BFS. y is in level $l' + 1$ only if y is a neighbor of some vertex z in level l'. For a given z, this occurs with probability p if $z \in V_j$ and with probability q if $z \notin V_j$. Let m_1 denote $n_j(l')$ and let m_2 denote $\sum_{j' \neq j} n_{j'}(l')$. First, using the expression for $p + (k-1)q$, we get using (1)

$$\begin{aligned} m_1 + m_2 = \sum_{j \geq 1} n_j(l') &\leq ((1+\alpha)nS)^{l-1} \\ &= o\left(\frac{\alpha}{S}\right). \end{aligned} \tag{6}$$

As a result,

$$\begin{aligned} \Pr(y \text{ is in level } l' + 1) &= 1 - (1-p)^{m_1}(1-q)^{m_2} \\ &= 1 - (1 - m_1 p[1 - f_1])(1 - m_2 q[1 - f_2]) \\ &= m_1 p[1 - f_1] + m_2 q[1 - f_2] \\ &\quad - m_1 p m_2 q[1 - f_1][1 - f_2], \end{aligned}$$

where $f_1 = O(m_1 p)$, and $f_2 = O(m_2 q)$ by Fact 1.1. Simplifying, we get

$$\Pr(y \text{ is in level } l' + 1) = (m_1 p + m_2 q)[1 - f_3] \tag{7}$$

where $f_3 = O(m_1 p + m_2 q)$. From (6), it follows that $f_3 = o(\alpha)$.

Also, for each j, the number of eligible candidates in V_j for y can be seen to be $n[1 - o(\alpha)]$. Let m denote the quantity $\min\{n(m_1 p + m_2 q) : j \geq 1\}$. From (7), (6), (4), and (5) and considering only potential edges incident on level l', using Chernoff bounds, we can infer that the following holds: with probability at least $1 - 2ke^{-\alpha^2 m/4}$, for each $j \geq 1$:

$$\begin{aligned} n_j(x, w, l' + 1) &\in (1 - \alpha, 1 + \alpha)n(m_1 p + m_2 q)[1 - o(\alpha)] \\ &= (1 - \alpha, 1 + \alpha)n\Big(p \cdot n_j(l') + q \cdot \sum_{j' \neq j} n_{j'}(l')\Big). \end{aligned} \tag{8}$$

Using the inductive hypothesis and the definition of $F_1(l' + 1), F_2(l' + 1)$, we deduce that (4) and (5) hold for $l' + 1$ also with probability at least $1 - 2ke^{-\alpha^2 m/4}$. In addition, $m \geq m_1 + m_2 \geq [1 - o(1)](F_1(l') + (k-1)F_2(l')) \geq (nS)^{l'}/2 \geq \min(np, nq)$.

This proves Lemma 3.1 for all values of l'. In deriving (8), we have ignored several multiplicative factors of the form $[1 - f(n)]$, where $f(n) = o(\alpha)$, since all these can be absorbed by $(1 \pm \alpha)$. ∎

This lemma is true when $\delta \leq \frac{\sqrt{X}-1}{\sqrt{X}l(l+1)}$. The problem with the other case is the following: When $\delta \geq \frac{1}{l(l+1)}$, we have $(nS)^{l-1} = \Omega(1/S)$ and hence Fact 1.1 cannot be applied to derive (6) and (7) and to bound the sizes of the lth level (this happens in the proof of Lemma 2.1). For this case, we grow the BFS tree in a slightly different way at the last level.

After computing the $(l-1)$th level of the BFS tree, we choose a subset D' by randomly and independently including each vertex in level $l-1$ with probability $n^{-\kappa(l)}$, where $\kappa(l) \doteq l(\tau(l) + \tau(l-1))$. For our choice of $\kappa(l)$, with probability at least $1 - e^{-\Omega(\alpha^2 (nS)^{l-1} n^{-\kappa(l)})}$, for all $j \geq 1$,

$$\begin{aligned} &|D' \cap V_j| \in (1-\alpha, 1+\alpha) n_j(x, w, l-1) n^{-\kappa(l)} \\ &|D' \cap V_j| \leq 2S^{-1} n^{-\frac{1}{\sqrt{X}(l+1)}} = o\left(\frac{\alpha}{S}\right). \end{aligned} \tag{9}$$

Let D'' be the set of vertices in level l which are adjacent to some vertex of D'. Effectively, we consider the lth level to be grown by considering only edges incident on D'.

Applying the arguments used in the proof of Lemma 2.1, one gets

LEMMA 2.2. *Assume that* $\delta \geq \frac{\sqrt{X}-1}{\sqrt{X}l(l+1)}$. *Then, for* x *and* w *mentioned before, with probability at least* $1 - e^{-\Omega(\alpha^2 \min(np, nq, n^{0.25}))}$, *for all* $j > 1$,

$$|D'' \cap V_1| \in ((1-\alpha)^{l+1}, (1+\alpha)^{l+1}) F_1(l) n^{-\kappa(l)} \tag{10}$$

$$|D'' \cap V_j| \in ((1-\alpha)^{l+1}, (1+\alpha)^{l+1}) F_2(l) n^{-\kappa(l)}. \tag{11}$$

3. RETRIEVING THE HIDDEN PARTITION FOR $p, q \geq (\log n) 2^{\sqrt{\log n}}/n$

From (4) and (5), we see that at level l, the contribution $n_j(l)$ of each V_j is close to its expectation, which is either $F_1(l)$ or $F_2(l)$ depending on whether $j = 1$ or not. Moreover, for $p < q$, F_1 is larger than F_2 for even l and F_2 is larger than F_1 for odd l. Also, for $p > q$, F_1 is always larger than F_2 for any l'. In any case, $|n_1(l) - n_j(l)|$ is "roughly" $n^l |p-q|^l$ for each $j \neq 1$. We can exploit this difference to infer, with high confidence, whether $w \in V_1$ or not. The details are given below.

Consider a random graph $G \in \mathcal{G}(n, k, p, q)$ with p, q satisfying the stated lower bound and also the assumption (A1) stated below. We show how to retrieve the hidden partition $(V_1, \ldots, V_k)$ by growing BFS trees.

(A1) We assume that p, q, α satisfy

$$|q - p| \geq 2\alpha^{1/(l+1)} (p + (k-1)q).$$

(A2) Initially, we assume that $\tau(l) \leq \delta \leq \frac{\sqrt{X}-1}{\sqrt{X}l(l+1)}$.

BFSPart$(G = (V, E))$. Recall the definition of l given in Section 1. Compute a set $C(x)$ for each vertex $x \in V$. If $\{C(x) : x \in V\}$ forms a partition of V, output this partition. $C(x)$ is computed as follows. For each $y \neq x$, grow the BFS tree from x (in the subgraph induced by $V - \{y\}$) to the lth level and compute $n(x, y)$, the number of y's neighbors in the lth level.[3] If $p > q$ or l is odd, arrange the vertices in $V - x$ in decreasing order of their $n(x, y)$ values. Otherwise ($p < q$ and l is even), arrange these in increasing order. Set $C(x)$ to be the union of $\{x\}$ and the first $n - 1$ vertices in this order. (We can further check if the output partition satisfies the required density bounds. This depends on the problem requirements.)

The success of the algorithm is explained as follows: Assume, w.l.o.g., that the picked vertex x is from V_1 and consider any $y \in V_1 - x$ and any $z \in V - V_1$. As explained before, for $p < q$, vertices from V_1 will be present in a larger (or smaller) proportion in the last level l, compared to those from V_j for any $j > 1$ depending on whether l is even or odd. In the case of $p > q$, V_1 always has a larger contribution. Because of this, y will have more (or less) neighbors in the last level than z. Thus, the first $n - 1$ vertices in the order would, almost surely, be $V_1 - x$ and hence $C(x) = V_1$. In what follows, we give a formal proof of this argument. Assume, w.l.o.g., that $x \in V_1$. Consider the following expressions defined for all $l' \geq 0$.

$$E_1(l') = F_1(l' + 1)/n, \qquad E_2(l') = F_2(l' + 1)/n.$$

$E_1(l)$ captures the expectation of $n(x, y)$ for any $y \in V_1$ and $E_2(l)$ captures the expectation of $n(x, z)$ for any $z \notin V_1$. The correctness of our algorithm is based on the following theorem.

THEOREM 3.1. *The following hold with probability at least* $1-e^{-\Omega(\alpha^2 \min(np, nq, n^l S^{l+1}))}$: *for any* $y \in V_1$, $y \neq x$, *and any* $z \in V_j$, $j > 1$,

$$n(x, y) \in \left((1-\alpha)^{l+1}, (1+\alpha)^{l+1}\right) E_1(l) \tag{12}$$

$$n(x, z) \in \left((1-\alpha)^{l+1}, (1+\alpha)^{l+1}\right) E_2(l) \tag{13}$$

$$n(x, z) - n(x, y) = \Omega(n^l (q-p)^{l+1}) \qquad \text{for} p < q \text{ and even } l \tag{14}$$

$$n(x, y) - n(x, z) = \Omega(n^l (q-p)^{l+1}) \qquad \text{for } p < q \text{ and odd } l \tag{15}$$

$$n(x, y) - n(x, z) = \Omega(n^l (p-q)^{l+1}) \qquad \text{for } p > q. \tag{16}$$

[3]This seems to require that the algorithm know the value of $p + (k-1)q$. If this is known, then it outputs a single partition, which is almost surely the hidden one. Otherwise, since $l = O(\sqrt{\log n})$, we repeat the algorithm for $i = 1, \ldots, O(\sqrt{\log n})$ assuming that $l = i$ during the ith iteration. In this case, we output a $O(\sqrt{\log n})$-sized collection of partitions. Almost surely, one of these is the hidden partition.

Proof. From (3) and the lower bounds on p, q, it follows that it is sufficient to prove for any fixed x, y, and z. Fix these parameters. Let w denote any of y and z and fix w also. Consider the BFS tree in $G - w$ grown from x. Recall that $n_j(x, w, l')$ denotes the number of vertices in the l'th level of the tree which are also in V_j. Assume Lemma 2.1 and consider $w = y \in V_1 - x$. Consider the lth level of the BFS tree grown from x in $G - y$. There are $\sum_{j' \geq 1} n_{j'}(x, y, l)$ vertices in this level. Given that Lemma 2.1 holds for $w = y$, we have (by Chernoff's bounds) with probability at least $1 - e^{-\Omega(\alpha^2 n^l S^{l+1})}$,

$$n(x, y) \in ((1-\alpha), (1+\alpha))\left(n_1(x, y, l)p + q \cdot \sum_{j'>1} n_{j'}(x, y, l)\right). \tag{17}$$

Now consider $w = z \in V_j$ for some $j > 1$. As before, given that Lemma 2.1 holds for $w = z$, we have with similar probability,

$$n(x, z) \in ((1-\alpha), (1+\alpha))\left(n_j(x, z, l)p + q \cdot \sum_{j' \neq j} n_{j'}(x, z, l)\right). \tag{18}$$

For each of the cases $w = y$ and $w = z$, the events of Lemma 2.1 and the corresponding inequality ((17) or (18)) are independent since w is not included in the graph considered in Lemma 2.1. Hence, (17) and (18) both hold with probability at least $1 - e^{-\Omega(\alpha^2 \min(np, nq, n^l S^{l+1}))}$. Applying (4) and (5) to each of (17) and (18) and using the definition of $E_1(l), E_2(l)$, we get (12) and (13).

Using (12) and (13), we prove only (14) and leave the proofs of (15) and (16) since they are similar to the proof of (14). We have, using $(1+\alpha)^{l+1} - (1-\alpha)^{l+1} = 2(l+1)\alpha[1+o(1)]$,

$$\begin{aligned} n(x, z) - n(x, y) &\geq (1-\alpha)^{l+1} E_2(l) - (1+\alpha)^{l+1} E_1(l) \\ &\geq (1-\alpha)^{l+1} n^l (q-p)^{l+1} - 2(l+1)\alpha E_1(l)[1+o(1)] \\ &\geq (1-\alpha)^{l+1} n^l (q-p)^{l+1} \\ &\quad - 2(l+1)\alpha n^l (p + (k-1)q)^{l+1}/k. \end{aligned} \tag{19}$$

Using (A1), it follows that the second term of (19) is smaller by a multiplicative factor of k compared with the first term. Hence (14) follows. This completes the proof of Theorem 2.1. ∎

Thus, we see that BFSPart outputs the partition $(V_1, \ldots, V_k)$ in polynomial time with probability at least $1 - e^{-\Omega(\alpha^2 \min(np, nq, n^l S^{l+1}))}$.

When (A2) is not true (that is, $\delta \geq \frac{\sqrt{X}-1}{\sqrt{X} l(l+1)}$), we slightly modify BFSPart as follows. Instead of growing the BFS tree as usual, we grow it as explained at the end of Section 2 and compute the set D''. Define $n(x, y)$

to be the number of neighbors that y has in D''. One can derive an analogue of Theorem 3.1 in this case, just as the analogue of Lemma 2.1 was derived. This helps us in obtaining the hidden partition with required failure probability.

The failure probability of the random bits used by the algorithm not serving its purpose can be made as small as $e^{-\omega(n)}$ by repeating the experiment (with new and independent random bits for each execution) n times. Hence, we estimate failure probability with respect to randomness over the input; randomness of the algorithm is not taken into account. In this case, we output the collection of partitions obtained during each trial of the experiment.

Remark 3.1. 1. The lower bound on $|q - p|$ given in (A1) is only tentative and we have not tried to optimize it. But, this can be brought down even further. In other words, we can allow p and q to become even closer than is suggested in (A1). We have used a uniform value of α (for all p and q) to keep the proof simpler. This restricts our range of α since it is required that $\alpha^2 \min(np, nq) \geq (\log n)^2$ in the proof of Lemma 1. If p and q differ by a constant multiplicative factor and nS is "sufficiently large," then we can bring down α even further. Consequently, $|q - p|$ can also be reduced even further.

2. The algorithm only outputs (almost surely) either $(V_1, \ldots, V_k)$ or a collection of partitions containing $(V_1, \ldots, V_k)$, but it does not certify that any of these is the hidden one. In fact, no such certificate exists.

4. STRUCTURE OF THE BFS TREE, $\min(p, q) \geq 0$

If $p = \min(p, q)$ and p becomes very small, we cannot tightly bound $n_1(1)$ (using Chernoff bounds) with high confidence since the expectation itself becomes very small. Similarly, if $q = \min(p, q)$ and q becomes very small, $n_j(1)$ $(j > 1)$ cannot be tightly estimated. One exception is when $\min(p, q) = 0$. In that case, one knows that the respective value is exactly 0.

Hence, when there is no lower bound on $\min(p, q)$, we do not try to give tight estimates on each V_i's contribution. Rather, we only tightly estimate the total size of each level and also show that at *every* level, V_1 contributes significantly more (or less) than any V_j, $j > 1$. This is sufficient for our purposes. Fix three arbitrary distinct vertices x, w_1, w_2. For the sake of keeping the analysis simpler, we grow the BFS tree from x after removing w_1 and w_2 from G. Let $n_j(x, w_1, w_2, l')$, shortly $n_j(l')$, denote V_j's contribution to the l'th level in the BFS tree grown in $G - \{w_1, w_2\}$ starting from x. We have the following modified version of Lemma 2.1.

LEMMA 4.1. *Assume that* $\tau(l) \le \delta \le \frac{\sqrt{X}-1}{\sqrt{X}l(l+1)}$. *Then, for* x, w_1, w_2 *mentioned before, with probability at least* $1 - e^{-\Omega(\alpha^2 \max(np,nq))}$, *for all* $j > 1$,

$$\sum_{j' \ge 1} n_{j'}(l) \in \left((1-\alpha)^l, (1+\alpha)^l\right)(nS)^l \tag{20}$$

$$n_1(l) - n_j(l) \ge (1-\alpha)^l (nD)^l - 2l\alpha(1+\alpha)^l (nS)^l \qquad \textit{for } p > q \tag{21}$$

$$(-1)^l (n_1(l) - n_j(l)) \ge (1-\alpha)^l (n|D|)^l - 2l\alpha(1+\alpha)^l (nS)^l \qquad \textit{for } p < q. \tag{22}$$

Proof. As in the proof of Lemma 2.1, we use induction to prove (20), (21), and (22) for each $l' \le l$. When $l' = 0$, they are seen to hold with probability 1. Assume that the inequalities are true for all values $\le l'$ where $l' \le l-1$ with required probability considering only potential edges incident on vertices in levels before l'. For each $j \ge 1$, let $m_1(j) = n_j(l')$ and let $m_2(j) = \sum_{j' \ne j} n_{j'}(l')$. As shown before, for any $y \in V_j - \{x, w_1, w_2\}$, $j \ge 1$, which is not yet visited by BFS,

$$\Pr(y \text{ is in level } l'+1) = (pm_1(j) + qm_2(j))[1 - f_3],$$
$$\text{where } f_3 = O(m_1 p + m_2 q),$$

where $f_3 = o(\alpha)$. The analysis now splits into two cases.

Case $p > q$. Let m denote $n(pm_1(1) + qm_2(1))$. Then, with probability at least $1 - (k+2)e^{-\alpha^2 m/3}$, considering only potential edges incident on level l',

$$n_1(l'+1) \ge (1-\alpha)n(pm_1(1) + qm_2(1)) \tag{23}$$

$$\begin{aligned} n_1(l'+1) + n_j(l'+1) \le{}& (1+\alpha)n(pm_1(1) + qm_2(1) \\ &+ pm_1(j) + qm_2(j)), \qquad j > 1 \end{aligned} \tag{24}$$

$$\sum_{j \ge 1} n_j(l'+1) \in ((1-\alpha), (1+\alpha))n\left(\sum_{j \ge 1} pm_1(j) + qm_2(j)\right). \tag{25}$$

Subtracting (24) from twice (23), we get, for any $j > 1$,

$$\begin{aligned} n_1(l'+1) - n_j(l'+1) \ge{}& (1-\alpha)n(n_1(l') - n_j(l'))(p-q) \\ &- 2\alpha n\left(\sum_{j' \ge 1} pm_1(j') + qm_2(j')\right). \end{aligned}$$

Using the inductive hypothesis that (21) and (20) are true for l' and applying it to the previous inequality, we deduce that (21) and (20) are true for $l'+1$. Also, for $l' > 0$, $m \ge m_1(1) + m_2(1) \ge [1 - o(1)]nS \ge \max(np, nq)/2$. For $l' = 0$, $m \ge np = \max(np, nq)$. This establishes that (21) and (20) are true for all $l' \ge 0$.

Case $p < q$. Let m denote $\min\{n(pm_1(j) + qm_2(j)) : j > 1\}$. Then, with probability at least $1 - 2ke^{-\alpha^2 m/3}$, considering only potential edges incident on level l', for any $j > 1$,

$$n_j(l'+1) \in ((1-\alpha), (1+\alpha))n(pm_1(j) + qm_2(j)), \qquad j > 1 \tag{26}$$

$$\sum_{j\geq 1} n_j(l'+1) \in ((1-\alpha), (1+\alpha))n\left(\sum_{j\geq 1} pm_1(j) + qm_2(j)\right). \tag{27}$$

Each of the above represents two inequalities corresponding to usage of $\leq$ and $\geq$. Denote by $(j-)$ that inequality represented by (26) which is marked by j and $\geq$. Similarly, $(j+)$ is marked by j and $\leq$. (27+) and (27−) are similarly defined. Consider any $j > 1$. If l' is even, considering the inequality denoted by $(j-) + \sum_{j'>1}(j'-) - (27+)$, we get

$$\begin{aligned} n_j(l'+1) - n_1(l'+1) \geq{} & (1-\alpha)n(n_1(l') - n_j(l'))(q-p) \\ & - 2\alpha n\left(\sum_{j'\geq 1} pm_1(j') + qm_2(j')\right). \end{aligned}$$

Using the inductive hypothesis that (22) and (20) are true for l' (l' is even and $l'+1$ is odd) and applying it to the previous inequality, we deduce that (22) and (20) are also true for $l'+1$. In addition, for $l' > 0$, $m \geq m_1(1) + m_2(1) \geq [1 - o(1)]nS \geq \max(np, nq)/2$. For $l' = 0$, $m \geq nq = \max(np, nq)$.

Similarly, if l' is odd, by considering the inequality denoted by $(27-) - (j+) - \sum_{j'>1}(j'+)$ we can establish (22) and (20) for $l'+1$. This completes the proof of the lemma. ∎

When $\delta \geq \frac{\sqrt{X}-1}{\sqrt{X}l(l+1)}$, as explained in Section 2, we grow the lth level from a random subset D' of the $(l-1)$th level, with each element of level $l-1$ included in D' with probability $n^{-\kappa(l)}$, where $\kappa(l)$ is defined before. The remaining arguments are similar and are left.

5. RETRIEVING THE HIDDEN PARTITION, $\min(p, q) \geq 0$

Suppose that the assumptions (A1) and (A2) hold true. Then, as shown in Lemma 4.1, at level l, the contribution of V_1 is significantly more (or less) than that of any V_j, $j > 1$, depending on whether $p < q$ (or otherwise) and whether l is odd or even. We apply this phenomenon to find the hidden partition as follows. The algorithm is slightly different from BFSPart() described before. It is given below.

BFSPartMod$(G = (V, E))$. Compute a set $C(x)$ for each $x \in V$. If $\{C(x) : x \in V\}$ forms a partition of V, output this partition. $C(x)$ is computed as follows. For any two vertices y, z both distinct from x, grow the

BFS tree from x (in the subgraph induced by $V - \{y, z\}$) to the lth level for some suitable l which depends only on p, q, k and set $R(y, z) = +1, 0, -1$ according to whether the number of neighbors y has in the lth level is, respectively, greater than, equal to, or lesser than the number of neighbors z has in the lth level. If $p > q$ or l is odd, set $D(x) \doteq \{w \neq x \mid R(w, w') = 1$ for at least $(k-1)n$ vertices $w'\}$. Otherwise ($p < q$ and l is even), set $D(x) \doteq \{w \neq x \mid R(w, w') = -1$ for at least $(k-1)n$ vertices $w'\}$. If $D(x)$ has exactly $n - 1$ vertices, then set $C(x)$ to be $D(x) \bigcup \{x\}$.

The reason the algorithm succeeds is as follows: if $x \in V_1$, then, based only on p, q, and the parity of l, for any $y \in V_1 - x$ and any $z \in V - V_1$, y will have more (or less) neighbors than z in the lth level of the BFS tree grown from x in $G - \{y, z\}$. Hence, $C(x)$ will be exactly V_1. This is true for any x and thus we get the hidden partition. The following theorem is a formal statement of this.

Given G and three vertices x, y, and z, define $n_y(x, z)$ as the number of neighbors that y has in the lth level of the BFS tree grown from x in $G - \{y, z\}$. Similarly, $n_z(x, y)$ is defined. Assume, w.l.o.g., that $x \in V_1$. We have the following

THEOREM 5.1. *With probability at least* $1 - e^{-\Omega(\alpha^2 \min(\max(np, nq), n^l S^{l+1}))}$, *for any* $y \in V_1$, $y \neq x$, *and any* $z \in V_j$, $j > 1$,

$$n_y(x, z) - n_z(x, y) > 0 \qquad \textit{for } p > q \tag{28}$$

$$n_y(x, z) - n_z(x, y) > 0 \qquad \textit{for } p < q, \qquad \textit{odd } l \tag{29}$$

$$n_z(x, y) - n_y(x, z) > 0 \qquad \textit{for } p < q, \qquad \textit{even } l. \tag{30}$$

Proof. This is proved by applying arguments similar to those employed in the proof of Lemma 4.1. ∎

As before, when (A2) does not hold, we employ randomness to get the hidden partition.

Summary. 1. Either the hidden partition of G (if $p + (k-1)q$ is known) or a collection containing the hidden partition can be obtained in polynomial time almost surely for both $p < q$ and $p > q$. The failure probability (with or without randomization) is $o(n^{-\omega(1)})$. This can be verified from the statement of Theorem 5.1 by using (3).

2. If randomization is employed, the failure probability of this randomization not serving its purpose can be made as small as $e^{-\omega(n)}$ by repeating the algorithm n times.

Remarks. 3. The lower bound on $|p - q|$ given in (A1) can be made even smaller by making α to depend on p and q, instead of using a uniform value as has been done so far. In fact, $\alpha = (\log n)/\sqrt{\min\{\max(np, nq), n^l S^{l+1}\}}$ can be used in the proofs. Also, it can be seen that the given lower bound on $|p - q|$ is $o(p + (k-1)q)$ provided that $p + (k-1)q = n^{-1+\epsilon}$ for some $\epsilon = \omega(1/\sqrt{\log n})$. If $n^l S^{l+1}$ turns out to be "very small" compared to $\max(np, nq)$, one can lower α to values close to $\alpha = (\log n)(\max(np, nq))^{-1/3}$ by employing randomization. In this case, we randomly choose a $o(\alpha/S)$-sized subset E of the lth level and grow the $(l+1)$th level from E and compute the number of neighbors y or z has in this level. This is similar to what we do when (A2) does not hold. We skip these details. Note that the algorithm remains the same and various choices of α are only to ensure that the analysis works for a wide range of distributions.

4. The case of $p = 0, q > 0$ is the k-coloring problem studied in [15], which introduces the BFS approach and also shows how to achieve a trade-off in running time vs. failure probability.

5. The algorithmic results of Sections 4 and 5 imply those of Sections 2 and 3. Still, we treated the case $p, q \geq (\log n)2^{\sqrt{\log n}}/n$ separately to show that the sizes of various levels of the BFS tree can be tightly estimated.

In the following, we apply the results of Sections 2 through 5 to solve several NP-hard problems on random instances.

6. PARTITIONING INTO SPARSE PARTS

Suppose our problem $(P0)$ is, given H and d', to partition the vertices of H into k sparse parts with each part having edge density atmost d'. Fixing $d' = 0$, this specializes to the k-coloring problem. The latter problem is NP-hard (even if G is known to have a k-coloring in which all parts have equal sizes, see the arguments at the end of this section).

A random model for $(P0)$ is $\mathcal{G}(n, k, p, q)$ with the additional restriction that $p \leq \kappa d'$ for some positive constant κ *less than* 1. Assume that $d' \geq (\log n)^3/n$. For such a random graph, almost surely, the hidden partition satisfies $d(V_i) \leq d'$ for each i, as shown in Section 1.

We can assume that $q \geq d'(1 - 1/(\log n))$. Otherwise,

$$\Pr(\exists\, S \subseteq V, |S| = n, d(S) > d') \leq \binom{kn}{n} e^{-\Omega((\log n)^{-2} \cdot n(\log n)^3)}$$
$$= o(e^{-\Omega(n(\log n))}).$$

Thus, any partition into k subsets of size n each will suffice. Hence, we assume that $q \geq d'(1 - 1/(\log n))$. This implies that $q - p = \Omega(q)$ and hence, for sufficiently large X, by Fact 1.2, (A1) is satisfied.

Now apply BFSModPart and get a collection of partitions. If, for some partition $\{C(x) : x \in V\}$ in the collection, each $C(x)$ satisfies $d(C(x)) \leq d'$, then output this partition. From Section 5, we see that, almost surely, the collection contains the hidden partition and hence we output a partition of the required type.

The variation of $(P0)$ in which parts of the output partition are required to be of equal size, denoted by $(P0 =)$, is also NP-hard since its special case of finding a k-coloring of a given G with equal-sized color classes is NP-hard. By finding the hidden partition $(V_1, \ldots, V_k)$, we solve problem $(P0 =)$ also over random instances.

The hardness of finding a k-coloring with equal-sized color classes can be shown this way. Given an arbitrary G on n vertices, let G' be the graph obtained from G by adding $(k-1)n$ new vertices which have no edges incident on them. Then, G has a k-coloring if and only if G' has a k-coloring with equal-sized classes. In addition, this equivalence is constructive in the sense that one can obtain a solution of one problem from a solution of the other problem. This hardness persists even if the input is known to have such a k-coloring with equal sizes for all classes.

7. DENSE k-CUT

Let $(P1)$ denote the problem of partitioning the vertices of a graph into k parts with *each* inter-part cut (V_i, V_j) having edge density at least ρ. We refer to $(P1)$ as the *dense k-cut* problem. It is not known if $(P1)$ is NP-hard even though we suspect it to be so. A random model for $(P1)$ is $\mathcal{G}(n, k, p, q)$ with the additional restriction that $\rho \leq \kappa q$ for some positive constant κ *less than* 1. For $\rho \geq (\log n)^3/n$, a random graph drawn from this model almost surely satisfies $d(V_i, V_j) \geq \rho$ for each $i \neq j$.

We can assume that $p \leq \rho(1 + 1/(\log n))$. Otherwise, the problem becomes trivial since any partition of V into k parts of size n each will work. To see this, let $\mathcal{E}$ denote the event that $\exists\, S_1, S_2 \subseteq V : |S_1|, |S_2\| = n, S_1 \cap S_2 = \emptyset, d(S_1, S_2) < \rho$. Then,

$$\Pr(\mathcal{E}) \leq \binom{kn}{n}\binom{kn-n}{n} e^{-\Omega((\log n)^{-2} \cdot n(\log n)^3)} = o(e^{-\Omega(n(\log n))}).$$

This implies that $q - p = \Omega(q)$ and hence, for sufficiently large X, by Fact 1.2, (A1) is satisfied. As in Section 6, we apply BFSModPart and get a collection. If, for some partition $\{C(x) : x \in V\}$ in the collection, each interpart edge density is at least ρ, we output this partition. From Section 5, we see that, almost surely, the collection contains the hidden partition which is a desired partition.

8. MAX-k-CUT AND MAX-k-SECTION

Let $(P2)$ denote the problem of partitioning the vertices of a graph into k parts with maximum total cut size (number of edges joining vertices in different parts). This is the unweighted (unit edge weights) of the *max-k-cut* problem and is also known as the *maximum k-colorable subgraph* problem. When the parts of the output partition are also required to be of equal size, we denote it by $(P2 =)$. This is the unweighted *max-k-section* problem. The decision versions of $(P2)$ and $(P2 =)$ are both NP-complete since recoginizing k-colorable graphs is reducible to them.

Given an instance G (on n vertices) of $(P2)$, we construct an instance G' of $(P2 =)$ on kn vertices as explained in Section 6. Then, from a solution of G, we get a solution of G' of equal weight and vice versa. This shows that $(P2)$ is reducible to $(P2 =)$.

The mapping $G \mapsto G'$ shows that both $(P2)$ and $(P2 =)$ are NP-hard even if the input instance is known to have an optimal solution in which all part sizes are the same. In addition, $(P2)$ and $(P2 =)$ are both APX-complete [14]. Hence, for both problems, it is very likely that there is some $\delta > 0$ such that no polynomial algorithm can approximate the optimal solution within a multiplicative factor of $1 - \delta$. For random instances, however, an optimal solution can be found almost surely.

A random model for both problems is $\mathcal{G}(n, k, p, q)$ with p and q satisfying (A1). One can also show that, almost surely, $(V_1, \dots, V_k)$ is an optimal solution (for both problems). Also, for any $\delta > 0$, almost surely the total cut size of $(V_1, \dots, V_k)$ lies between $v \doteq \binom{k}{2} n^2 (1 - \delta) q$ and $v^* \doteq \binom{k}{2} n^2 (1 + \delta) q$. Given $G \in \mathcal{G}(n, k, p, q)$ and $\delta > 0$, we grow the BFS trees and obtain a collection as explained in Section 5 and from this output any partition having the maximum cut size. Almost surely, the collection obtained by the BFS approach contains the hidden partition because (A1) is satisfied. This means that we can approximate the optimal solution arbitrarily closely, almost surely, giving us the kind of approximate solution we are looking for.

9. PARTITIONING INTO DENSE PARTS

Suppose our problem $(Q0)$ is to partition the vertices of a given graph H into k dense parts with each part having edge density at least d''. When $d'' = 1$, this is the problem of partitioning V into k cliques which is equivalent to the k-coloring problem on the complement graph. Hence, $(Q0)$ is NP-hard even if G is known to have such a partition in which all parts have equal sizes. A random model for this is $\mathcal{G}(n, k, p, q)$ with the additional restriction that $d'' \le \kappa p$ for some positive constant κ *less than* 1. For $d'' \ge (\log n)^3/n$, for such a random graph, almost surely, the hidden partition satisfies $d(V_i) \geq d''$ for each i. To obtain partition into dense parts, we use arguments essentially similar to those given in Section 6. We briefly sketch them for the sake of completeness.

When $q \geq d''(1 + 1/(\log n))$, any partition into k subsets of size n each will suffice. Hence, we assume that $q \leq d''(1 + 1/(\log n))$. This implies that $p - q = \Omega(p)$ and hence, for sufficiently large X, (A1) is satisfied. As before, apply BFSModPart and obtain a collection. If, for some partition $\{C(x) : x \in V\}$ in the collection, each $C(x)$ satisfies $d(C(x)) \geq d''$, then output this partition. From Section 5, we see that, almost surely, this would output a partition into dense parts.

10. SPARSE k-CUT

Let $(Q1)$ denote the problem of partitioning the vertices of a graph into k parts with *each* interpart cut (V_i, V_j) having edge density at most ρ. We refer to this as the *sparse k-cut* problem. The NP-completeness status of this problem is not known yet. A random model for $(Q1)$ is $\mathcal{G}(n, k, p, q)$ with the additional restriction that $q \leq \kappa\rho$ for some positive constant κ *less than* 1. For $\rho \geq (\log n)^3/n$, a random graph drawn from this model almost surely satisfies $d(V_i, V_j) \leq \rho$ for each $i \neq j$.

By arguments similar to those given in Section 7, we can assume that $p \geq \rho(1 - 1/(\log n))$. Otherwise, any partition of V into k parts of size n each will work. This implies that $p - q = \Omega(p)$ and hence, for sufficiently large X, (A3) is satisfied. As in Section 7, we apply BFSModPart and obtain a partition with each interpart edge density at most ρ and output this partition.

11. MIN-k-SECTION

Let $(Q2)$ denote the problem of partitioning the vertices of a graph into k non-empty parts with minimum total cut size (number of edges joining vertices in different parts). This is a special case (unit edge weights) of the so-called *min-k-cut* problem. The latter problem (and hence $(Q2)$) is polynomial time solvable for fixed k [8] and for $k = 2$; this is the well-known min-cut problem.

When the parts are also required to be of equal size in $(Q2)$, we denote this problem by $(Q2 =)$. This is the unweighted version of the *min-k-section* (*min-k-cut* with equal sized parts) problem. Any partition into parts of equal size is referred to as a k-section. Even the special case of $(Q2 =)$ (when $k = 2$), known as the minimum bisection problem, is NP-hard.

These problems are difficult to approximate even within a reasonable additive error. Precisely, for any fixed $\delta > 0$, it is NP-hard to produce (in polynomial time) a bisection whose size is at most the optimal value plus $O(n^{2-\delta})$ [3].

However, interesting results have been obtained for this problem (mostly, for bisection) over random instances [1, 2, 4, 5, 7, 10] with a planted bisection. In a random graph, every bisection has the same expected width and hence there is not much information which helps us in separating optimal bisections from the rest. Thus, average case results are based on models in which a bisection of small width is planted (arbitrarily or randomly) in a random graph. The random graph is chosen so that the planted bisection is almost surely the unique optimal bisection.

A random model for this problem is $\mathcal{G}(n, k, p, q)$ with p and q with $p > q$. If p and q are *sufficiently* far apart, the hidden partition is the unique optimal k-section. In our model, we assume that p and q satisfy (A1).

References [1, 7] both proposed bisection algorithms which work for a quite large class of distributions, but these are quite complex, using the ellipsoid method or semidefinite programming tools. References [4, 10] proposed much simpler algorithms that work for a smaller class of distributions. The algorithm of [10] finds the unique optimal bisection almost surely if $p - q = \Theta(n^{-1/6+\epsilon})$ for any fixed $\epsilon > 0$. This also implies that we need to have $p \geq n^{-1/6}$. Recently, [4] presented a simple combinatorial algorithm for bisection and showed that this algorithm succeeds as long as $p - q \geq n^{-0.5+\delta}$ for some constant $\delta > 0$. This implies that we need to have $p \geq n^{-0.5+\delta}$. This also works for the *min-k-section* problem.

One can show that, for the assumed range of p, q, (V_i) is an optimal k-section. Also, for any $\delta > 0$, almost surely the width of $(V_1, \dots, V_k)$ lies between $\binom{k}{2}n^2(1-\delta)q$ and $\binom{k}{2}n^2(1+\delta)q$. Given that $G \in \mathcal{G}(n, k, p, q)$ and $\delta > 0$, we grow the BFS trees and obtain a collection of k-sections, and from this we output any k-section having minimum width. Almost surely, the collection obtained by the BFS approach contains the hidden partition because (A1) is satisfied. This means that we can approximate the optimal solution arbitrarily closely, almost surely.

In its simplicity, the BFS approach is comparable to those of [4, 10], but it works for a large class of distributions almost comparable to those of [1].

12. CONCLUSIONS

The BFS trees grown in a random graph (with a planted partition) possess several nice structural properties which can be exploited to retrieve the planted partition. These lead to algorithms for solving exactly several NP-hard partitioning problems over random instances. It would be of interest to see if there are some more hard partitioning problems which can be solved in this way over random instances. Throughout, we assumed that k is arbitrary but *fixed*. But, one can see that our results hold true even if k is allowed to grow slowly with n. However, the value of $p + (k - 1)q$ will have to be higher and hence its range becomes smaller.

REFERENCES

1. R. B. Boppana, Eigenvalues and graph bisection: An average-case analysis, *in* "Proceedings of the 28th Annual IEEE Symposium on Foundations of Computer Science (FOCS '87), 1987," pp. 280–285.
2. T. N. Bui, S. Chaudhuri, F. T. Leighton, and M. Sipser, Graph bisection algorithms with good average case behavior, *Combinatorica* **7**, No. 2 (1987), 171–191.
3. T. N. Bui and C. Jones, Finding good approximate vertex and edge partitions is NP-hard, *Inform. Proc. Lett.* **42** (1992), 153–159.
4. A. Condon and R. M. Karp, Algorithms for graph bisection on the planted bisection model, *in* "Proceedings of RANDOM '99 Workshop, 1999."
5. M. E. Dyer and A. M. Frieze, The solution of some random NP-hard problems in polynomial expected time, *J. Algorithms* **10** (1989), 451–489.
6. T. Feder, P. Hell, S. Klein, and R. Motwani, Complexity of graph partition problems, *in* "Proceedings of the Annual ACM Symposium on Theory of Computing (STOC '99), 1999."
7. U. Feige and J. Kilian, Heuristics for finding large independent sets, with applications to coloring semi-random graphs, *in* "Proceedings of the 39th Annual IEEE Symposium on Foundations of Computer Science (FOCS '98), 1998," pp. 674–683.
8. O. Goldschmidt and D. S. Hochbaum, Polynomial algorithm for the k-cut problem, *in* "Proceedings of the 29th Annual IEEE Symposium on Foundations of Computer Science (FOCS '88), 1988," pp. 444–451.
9. M. R. Garey and D. S. Johnson, "Computers and Intractability: A Guide to the Theory of NP-Completeness," Freeman, San Francisco, 1978.
10. M. Jerrum and G. B. Sorkin, Simulated annealing for graph bisection, *in* "Proceedings of the 34th Annual IEEE Symposium on Foundations of Computer Science (FOCS '93), 1993," pp. 94–103.
11. A. Juels and M. Peinado, Hiding cliques for cryptographic security, *in* "Proceedings of the 9th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA '98), 1998," pp. 678–684.
12. L. Kučera, A generalized encryption scheme based on random graphs, *in* "Graph Theoretic Concepts in Computer Science, WG '91," Lecture Notes in Computer Science, Vol. 570, pp. 180–186, Springer-Verlag, Berlin/New York, 1991.
13. R. Motwani and P. Raghavan, "Randomized Algorithms," Cambridge Univ. Press, Cambridge, UK, 1995.
14. C. H. Papadimitriou and M. Yannakakis, Optimization, approximation and complexity classes, *J. Comput. System Sci.* **43**, 425–440.
15. C. R. Subramanian, Coloring sparse random graphs in polynomial average time, *in* "Proceedings of the 8th Annual European Symposium on Algorithms (ESA '00), 2000," pp. 415–426.