Combinatorics, Probability and Computing (2014) 23, 29–49. © Cambridge University Press 2013
doi:10.1017/S096354831300045X

Finding Hidden Cliques in Linear Time with High Probability

YAEL DEKEL[1], ORI GUREL-GUREVICH[2] and YUVAL PERES[3]

[1]The Selim and Rachel Benin School of Computer Science and Engineering,
The Hebrew University of Jerusalem, Edmond J. Safra Campus, Jerusalem 91904, Israel
(e-mail: yaelvin@cs.huji.ac.il)
[2]Department of Mathematics, University of British Columbia,
121-1984 Mathematics Road, Vancouver, BC, V6T 1Z2 Canada
(e-mail: origurel@math.ucb.ca)
[3]Microsoft Research, One Microsoft Way, Redmond, WA 98052, USA
(e-mail: peres@microsoft.com)

Received 1 April 2011; revised 8 September 2013; first published online 14 November 2013

We are given a graph G with n vertices, where a random subset of k vertices has been made into a clique, and the remaining edges are chosen independently with probability $\frac{1}{2}$. This random graph model is denoted $G(n, \frac{1}{2}, k)$. The hidden clique problem is to design an algorithm that finds the k-clique in polynomial time with high probability. An algorithm due to Alon, Krivelevich and Sudakov [3] uses spectral techniques to find the hidden clique with high probability when $k = c\sqrt{n}$ for a sufficiently large constant $c > 0$. Recently, an algorithm that solves the same problem was proposed by Feige and Ron [12]. It has the advantages of being simpler and more intuitive, and of an improved running time of $O(n^2)$. However, the analysis in [12] gives a success probability of only 2/3. In this paper we present a new algorithm for finding hidden cliques that both runs in time $O(n^2)$ (that is, linear in the size of the input) and has a failure probability that tends to 0 as n tends to ∞. We develop this algorithm in the more general setting where the clique is replaced by a dense random graph.

2010 *Mathematics subject classification*: Primary 05C80
Secondary 05C85, 68W20

1. Introduction

A clique in a graph G is a subset of its vertices any two of which are connected by an edge. The problem of determining the size of the maximum clique in a graph is known to be NP-complete [22]. It has also been proved [5, 4, 13] that assuming P $\neq$ NP, there exists a constant $b > 0$ for which it is hard to approximate the size of the maximum clique within a factor of n^b. Therefore, it is natural to investigate the hardness of this problem in the average case.

The Erdős–Rényi random graph model, also denoted $G(n, \frac{1}{2})$, is a probability measure on graphs with n vertices. In this model, a random graph is generated by including each pair of vertices as an edge in the graph independently with probability $\frac{1}{2}$. It is known that with probability tending to 1 as n tends to infinity, the size of the largest clique in $G(n, \frac{1}{2})$ is $(2 + o(1)) \log_2 n$. There exists a polynomial-time algorithm (see, *e.g.*, [17]) that finds a clique of size $(1 + o(1)) \log_2 n$ in $G(n, \frac{1}{2})$ with high probability, but even though in expectation $G(n, \frac{1}{2})$ contains many cliques of size $(1 + \varepsilon) \log_2 n$ for any fixed $0 < \varepsilon < 1$, there is no known polynomial-time algorithm that finds one. It is plausible to conjecture that this problem is computationally hard, and this hardness has been used in several cryptographic applications [21, 24].

Finding a large clique may be easier in models where the graphs contain larger cliques. Define, therefore, the hidden clique model, denoted $G(n, \frac{1}{2}, k)$. Let $G(n, \frac{1}{2}, k)$ be the probability space whose members are pairs (G, K), where G is an n vertex graph with vertex set V, and K is a subset of V of size k. The edges between vertices in $V \setminus K$ are present independently with probability p, as are the edges between a vertex in $V \setminus K$ and a vertex in K. The edges between vertices in K are all present with probability 1. The construction of an instance of $G(n, \frac{1}{2}, k)$ can be reformulated as follows: First construct a random graph $G(n, \frac{1}{2})$, then randomly choose k vertices to form a clique. Jerrum [20] and Kučera [25] suggested this model independently and posed the problem of finding the hidden clique. When $k \geqslant c_0 \sqrt{n \log n}$ for some sufficiently large constant c_0, Kučera observed [25, Theorem 6.1] that the hidden clique can be found with high probability by taking the k highest degree vertices in the graph. For $k = c\sqrt{n}$, there is an algorithm due to Alon, Krivelevich and Sudakov [3] that uses spectral techniques to find the hidden clique with high probability when c is sufficiently large. In a more recent paper [12], Feige and Ron propose a simple algorithm that runs in time $O(n^2)$ and finds the hidden clique for $k = c\sqrt{n}$ with probability at least $2/3$. Both of these algorithms can work also for smaller values of c, at the expense of increasing the running time, using a technique introduced in [3]. In this paper we present a new algorithm that has the advantages of both algorithms, as it runs in time $O(n^2)$, and fails with probability at most $\exp(-n^{\varepsilon})$ for some $0 < \varepsilon < 1$. We study the *hidden dense graph model*, a different, more general model, denoted $G(n, p, k, q)$, where $k \leqslant n$ and $0 < p < q \leqslant 1$. Here too, $G(n, p, k, q)$ is a probability distribution over pairs (G, K), where G is an n-vertex graph on vertex set V, and K is a subset of V of size k. The edges between vertices in $V \setminus K$ are present independently with probability p, as are the edges between a vertex in $V \setminus K$ and a vertex in K. The edges between vertices in K are present independently with probability q. In this model too, a random n-vertex graph can equivalently be generated by randomly choosing k vertices from the vertex set, and including each pair of vertices among them as an edge in the graph independently with probability q. Every other pair of vertices is included as an edge in the graph independently with probability p. The hidden clique model $G(n, \frac{1}{2}, k)$ is equivalent to $G(n, \frac{1}{2}, k, 1)$. The algorithm we present in this paper is an algorithm that finds hidden dense graphs in $G(n, p, k, q)$ for any $p < q \leqslant 1$ and $k = c\sqrt{n}$, where c is a large enough constant. In particular, it can find hidden cliques of size $k = c\sqrt{n}$ in $G(n, \frac{1}{2}, k)$. In Section 4 we give a generalization of the technique used in [3] to reduce the constant c that works in the hidden dense graph model.

1.1. Main result

We now describe the algorithm presented in this paper. The algorithm is assumed to know the value of k, and it uses the following parameters: $0 < \alpha < 1$, $\beta, \eta > 0$ and an integer $t > 0$. There are three phases of the algorithm. In the first phase, we iteratively find subgraphs of the input graph G, denoted $G_1, G_2, \ldots, G_t$. We do so in such a way that the relative size of the subset of the hidden dense graph contained in the subgraph grows with each iteration. In the second phase, we find a subset of the hidden dense graph that is contained in G_t. Finally, in the third phase we find the whole hidden dense graph using its subset found in the second phase. In order to describe the algorithm more precisely, we use the following notations and definitions.

Notation 1.1. Given a graph $G = (V, E)$, for every $v \in V$ and $S \subseteq V$ we denote by $d_S(v)$ the number of neighbours v has in S. Formally,

$$d_S(v) = |\{u \in S : \{u, v\} \in E\}|.$$

We abbreviate $d_V(v)$ by $d(v)$.

Notation 1.2. Let $\varphi(x)$ denote the Gaussian probability density function

$$\varphi(x) = \frac{1}{\sqrt{2\pi}} \exp(-x^2/2).$$

We denote by $\Phi(x)$ the Gaussian cumulative distribution function $\Phi(x) = \int_{-\infty}^{x} \varphi(t)dt$, and we denote $\overline{\Phi}(x) = 1 - \Phi(x)$.

Definition 1.3. Throughout the paper, the subgraph of the graph where the edge probability is q, will be referred to as the 'dense graph', and its k vertices will be referred to as 'dense graph vertices'.

Definition 1.4. Given $0 < \alpha < 1$, $0 < p \leqslant 1$ and $\beta, \eta, c > 0$, we define

$$\gamma = \alpha\overline{\Phi}(\eta),$$

$$\delta = \alpha\overline{\Phi}\left(\eta - c\sqrt{\alpha}\sqrt{\frac{1-p}{p}}\right),$$

$$\tau = (1-\alpha)\overline{\Phi}(\beta),$$

and

$$\rho = (1-\alpha)\overline{\Phi}\left(\beta - \frac{c\delta}{\sqrt{\gamma}}\sqrt{\frac{1-p}{p}}\right).$$

Definition 1.5. For every $i \geqslant 0$, define $n_i = \tau^i n$ and $k_i = \rho^i k$. Also, define $\tilde{n}_0, \tilde{n}_1, \ldots$ and $\tilde{k}_0, \tilde{k}_1, \ldots$ to be the actual sizes of G_i and the hidden dense graph in G_i, respectively, when running the algorithm. Note that $\tilde{n}_0, \tilde{n}_1, \ldots$ and $\tilde{k}_0, \tilde{k}_1, \ldots$ are random variables. Note also that the user of the algorithm has no way of determining the values of $\tilde{k}_1, \tilde{k}_2$ and so on.

Definition 1.6. For every i, we call iteration i successful with respect to constants M, ε_1 and ε_2, if for some constants M and $\varepsilon_1, \varepsilon_2 > 0$, $|\tilde{n}_i - n_i| \leqslant M n_i^{1-\varepsilon_1}$ and $|\tilde{k}_i - k_i| \leqslant M k_i^{1-\varepsilon_2}$. The definition of a successful iteration is given only for the purpose of proving the correctness of the algorithm and computing its success probability. As mentioned in Definition 1.5, the user of the algorithm has no way of determining $\tilde{k}_i$, and thus has no way of determining whether a particular iteration has been successful or not.

Definition 1.7. For every $0 < \alpha < 1$, $0 < p \leqslant 1$ and $\beta, \eta > 0$, denote the minimal c for which $\rho \geqslant \sqrt{\tau}$ by $\tilde{c}(\alpha, \beta, \eta, p)$. Define c^* as the infimum of $\tilde{c}(\alpha, \beta, \eta, p)$ over α, β and η (note that c^* depends on p).

The algorithm proceeds as follows.

- (First phase.) Iteratively find a decreasing sequence of subgraphs of G of length t, denoted $G = G_0 \supset G_1 \supset \cdots \supset G_t$, with vertex sets $V = V_0 \supset V_1 \supset \cdots \supset V_t$. For $i = 1, \ldots, t$, in the ith iteration we pick the graph G_i. To do so, we first pick a random subset of vertices $S_i \subseteq V_{i-1}$ by including each vertex in S_i independently with probability α. Then define
$$\tilde{S}_i = \left\{ v \in S_i : d_{S_i}(v) \geqslant p|S_i| + \eta\sqrt{p(1-p)|S_i|} \right\}.$$
V_i is defined as
$$V_i = \left\{ v \in V_{i-1} \setminus S_i : d_{\tilde{S}_i}(v) \geqslant p|\tilde{S}_i| + \beta\sqrt{p(1-p)|\tilde{S}_i|} \right\},$$
and G_i is defined as the induced subgraph of G_{i-1} containing the vertices in V_i.
- (Second phase.) Let $\tilde{K}$ be the set of vertices in G_t whose degree is at least
$$p|V_t| + \frac{1}{2}(p+q)k_t.$$
- (Third phase.) Let K' be the set of vertices containing $\tilde{K}$ and all the vertices in G that have at least $\frac{1}{2}(p+q)|\tilde{K}|$ neighbours in $\tilde{K}$. Let K^* be the set of all vertices in G that have at least $\frac{1}{2}(p+q)k$ neighbours in K'.
- Return K^* as the candidate for the hidden dense graph.

Note that in the algorithm just described, there is no control imposed on the number of vertices in the set K^*. Despite this striking feature of the algorithm, when choosing the parameters correctly, with high probability K^* is equal to the hidden dense graph. We choose α, β and η in such a way that the relative size of the subset of the hidden dense graph contained in $\tilde{V}_i$ grows with each iteration. We choose t to be sufficiently large that with high probability $\tilde{K}$ is a subset of the hidden dense graph contained in G_t. A logarithmic number of iterations is enough. For the exact way of choosing α, β, η and t, see the proof of Lemma 3.5. We now state the main theorem of the paper.

Theorem 1.8. *For every $c > c^*$ and $0 < p < q \leqslant 1$, there exist α, β, η that depend only on c, p and q, and an integer $t = O(\log n)$ such that, given $G \sim G(n, p, c\sqrt{n}, q)$, the probability that $K^* = K^*(\alpha, \beta, \eta, t)$ is the hidden dense graph is at least $1 - \exp(-\Theta(n^{\varepsilon_0}))$ for some $\varepsilon_0 = \varepsilon_0(c)$.*

1.2. Related work

Since the work of Alon, Krivelevich and Sudakov [3], there have been many papers describing algorithms that solve different variants of the hidden clique problem. Feige and Krauthgamer [14] give an algorithm for finding hidden cliques of size $\Omega(\sqrt{n})$ based on the Lovász theta function, that has two advantages over previously known algorithms. The first is its ability to find the clique in a semi-random hidden clique model, in which an adversary can remove edges that are not in the clique, and the second is its ability to certify the optimality of its solution by providing an upper bound on the size of the maximum clique in the graph.

McSherry [26] gives an algorithm that solves the more general problem of finding a planted partition. In the random graph model described there, we are given a graph where the vertices are randomly partitioned into m classes, and between every pair of vertices where one is in class i and the other in class j there is an edge with probability p_{ij}. With the appropriate parameters, this model can be reduced to the hidden dense graph model, and thus also to the hidden clique model. For both these cases, the result is a polynomial-time algorithm that finds the hidden clique (dense graph) with high probability for $k = c\sqrt{n}$.

Several attempts have been made to develop polynomial-time algorithms for finding hidden cliques of size $k = o(\sqrt{n})$, so far with no success. For example, Jerrum [20] described the Metropolis process and proved that it cannot find the clique when $k = o(\sqrt{n})$. Feige and Krauthgamer [15] explain why the algorithm described in [14] fails when $k = o(\sqrt{n})$. Frieze and Kannan [16] give an algorithm to find a hidden clique of size $k = \Omega(n^{1/3} \log^4 n)$. However, the algorithm maximizes a certain cubic form, and there are no known polynomial-time algorithms for maximizing cubic forms.

There are many problems in different fields of computer science that are related to the hidden clique problem. Among others, there are connections to cryptography, testing and game theory. For connections to cryptography, see for example Kučera [24], where an encryption scheme is described that is based on hiding an independent set in a graph, or Juels and Peinado [21], where the function whose input is a graph G and a set K of k vertices and whose output is G with a clique on K is proposed as a one-way function for certain values of k. For connections to testing see [2], where Alon and co-workers prove that if there is no polynomial-time algorithm to find hidden cliques of size $k > \log^3 n$ then there is no polynomial-time algorithm that can test t-wise independence of a distribution even when given a polynomial number of samples from it, for $t = \Theta(\log n)$. For connections to game theory, see [18], where Hazan and Krauthgamer prove that if there is a polynomial-time algorithm that finds a Nash equilibrium of a two player game whose social welfare is close to the maximum, then there is a randomized polynomial-time algorithm that finds the hidden clique for $k = O(\log n)$. The hidden clique model is also related to the planted-SAT model (see, *e.g.*, Ben Sasson, Bilu and Gutfreund [8, 23]) and some models in computational biology (see, *e.g.*, Ben-Dor, Shamir and Yakhini [7]).

2. Probability of success in the first phase

In this section we calculate the probability of success in a single iteration of the first phase, as defined in Definition 1.6. In order to do so, we first prove that in every iteration, conditioned on V_{i-1}, S_i and the values of $\tilde{n}_i$ and $\tilde{k}_i$, the graph G_i has the same distribution as $G(\tilde{n}_i, p, \tilde{k}_i, q)$, and therefore it is enough to calculate the success probability of the first iteration.

The calculations of the success probability rely on the following two well-known theorems. The first is the central limit theorem for binomial random variables and its rate of convergence, which was independently discovered by Berry in 1941 [9] and by Esseen in 1942 [11]. For details, see, *e.g.*, [10, Section 3.4.4].

Theorem 2.1 (Berry–Esseen). *Let $B(n,p)$ be a binomial random variable with parameters n,p. Then, for every $x \in \mathbb{R}$,*

$$\left|\mathbb{P}\left(\frac{B(n,p)-pn}{\sqrt{p(1-p)n}} \leqslant x\right) - \Phi(x)\right| = O\left(\frac{1}{\sqrt{n}}\right).$$

The second is the well-known Azuma–Hoeffding inequality (see [19, 6]).

Theorem 2.2 (Azuma–Hoeffding inequality). *Let $S = X_1 + \cdots + X_n$, where $X_1, X_2, \ldots$ is a sequence of martingale differences such that $a_i \leqslant X_i \leqslant b_i$. Then, for every $t > 0$,*

$$\mathbb{P}(|S - \mathbb{E}S| \geqslant t) \leqslant 2\exp\left(-2t^2 \Big/ \sum_{i=1}^{n}(b_i - a_i)^2\right).$$

The first step in calculating the success probability of a single iteration is the following lemma.

Lemma 2.3. *For every $i \geqslant 0$, conditioned on V_{i-1}, S_i and the values of $\tilde{n}_i$ and $\tilde{k}_i$, the graph G_i defined in the ith iteration of the algorithm has the same distribution as $G(\tilde{n}_i, p, \tilde{k}_i, q)$.*

Proof. Assume, by induction, that G_{i-1} is distributed as $G(\tilde{n}_{i-1}, p, \tilde{k}_{i-1}, q)$, and conditioned on the set S_i. Consider the following equivalent way of generating $G(\tilde{n}_{i-1}, p, \tilde{k}_{i-1}, q)$ (given S_i). First, pick the $\tilde{k}_{i-1}$ hidden dense graph vertices. Then, pick all the edges between $V_{i-1} \setminus S_i$ and S_i, and all the edges inside S_i. At this point, we still need to pick the edges in $V_{i-1} \setminus S_i$, but V_i, and thus $\tilde{n}_i$ and $\tilde{k}_i$, are already determined. Since we can find the vertices of G_i before exposing any of the edges in it, and since the dense graph vertices of G_i are picked randomly, G_i has the same distribution as $G(\tilde{n}_i, p, \tilde{k}_i, q)$. □

As a result of Lemma 2.3, it is enough to calculate the success probability of the first iteration. We therefore begin with a concentration result for the size of S_1 and $S_1 \cap K$.

Lemma 2.4. *For every $0 < \varepsilon_1 < \frac{1}{2}$ and $0 < \varepsilon_2 < \frac{1}{2}$, and for every $M > 0$, the set S_1 satisfies*

$$\mathbb{P}\big(||S_1| - \alpha n| \geqslant Mn^{1-\varepsilon_1}\big) \leqslant 2\exp\big(-2M^2 n^{1-2\varepsilon_1}\big)$$

and

$$\mathbb{P}\big(||S_1 \cap K| - \alpha k| \geqslant Mk^{1-\varepsilon_2}\big) \leqslant 2\exp\big(-2M^2 k^{1-2\varepsilon_2}\big).$$

Proof. The proof follows directly from Theorem 2.2, by setting $t = Mn^{1-\varepsilon_1}$ for the bound on $|S_1|$ and $t = Mk^{1-\varepsilon_2}$ for the bound on $|S_1 \cap K|$. □

To prove concentration results for the sizes of $\tilde{S}_1$ and V_1 and their intersections with K we use the following two concentration results. The first is a concentration result for the number of vertices above a certain degree in bipartite graphs (this result is used for the concentration of the size of V_1, of its intersection with K, and of $\tilde{S}_1 \cap K$), and the second is a concentration result for the number of vertices above a certain degree in a non-bipartite graph (this result is used for the concentration of the size of $\tilde{S}_1$).

The first result is a simple corollary of Theorem 2.2.

Corollary 2.5. *Let A, B be two disjoint sets of vertices in $G \sim G(n,p)$ with $|A| = n_1$ and $|B| = n_2$, where $n_1 = n_1(n)$ and $n_2 = n_2(n)$ are such that $n_1 \leqslant O(n_2)$. Given $a \in \mathbb{R}$, define the random variable*

$$X = |\{v \in A : d_B(v) \geqslant pn_2 + a\sqrt{p(1-p)n_2}\}|.$$

Then, for every $M > 0$ and $0 < \varepsilon < \frac{1}{2}$ it holds that

$$\mathbb{P}\big(|X - \overline{\Phi}(a)n_1| \geqslant Mn_1^{1-\varepsilon}\big) \leqslant 2\exp\big(-M^2 n_1^{1-2\varepsilon}\big).$$

Proof. From Theorem 2.1 we know that

$$|\overline{\Phi}(a)n_1 - \mathbb{E}X| \leqslant c\frac{n_1}{\sqrt{n_2}}$$

for some constant $c > 0$. Therefore, by Theorem 2.2, for any constant $M > 0$,

$$\begin{aligned}\mathbb{P}\big(|X - \overline{\Phi}(a)n_1| \geqslant Mn_1^{1-\varepsilon}\big) &\leqslant \mathbb{P}\left(|X - \mathbb{E}X| \geqslant Mn_1^{1-\varepsilon} - c\frac{n_1}{\sqrt{n_2}}\right)\\ &\leqslant 2\exp\left(-2\frac{\big(Mn_1^{1-\varepsilon} - cn_1/\sqrt{n_2}\big)^2}{n_1}\right) \leqslant 2\exp\big(-M^2 n_1^{1-2\varepsilon}\big),\end{aligned}$$

where the last inequality holds because

$$\frac{n_1}{\sqrt{n_2}} \leqslant O\big(\sqrt{n_1}\big) = o\left(n_1^{1-\varepsilon}\right).$$ □

Next, we prove a concentration result for the number of vertices above a certain degree in a non-bipartite graph.

Lemma 2.6. *Let $G \sim G(n,p)$ and $a, c' > 0$. Define a random variable*

$$X = |\{v \in V(G) : d(v) \geqslant pn + a\sqrt{p(1-p)n}\}|.$$

Then, for every $0 < \varepsilon' < \frac{1}{4}$,

$$\mathbb{P}\big(|X - \overline{\Phi}(a)n| \geqslant c'n^{1-\varepsilon'}\big) \leqslant 2\exp\left(-\frac{\pi}{32}c'^4 p(1-p)n^{1-4\varepsilon'}\right).$$

Proof. For every $v \in V(G)$ define a random variable

$$X_v = \begin{cases} 1 & d(v) \geqslant pn + a\sqrt{p(1-p)n}, \\ 0 & \text{otherwise.} \end{cases}$$

Then $X = \sum X_v$. By Theorem 2.1 we have

$$|\overline{\Phi}(a)n - \mathbb{E}X| \leqslant c''\sqrt{n}$$

for some constant c''. To prove that X is concentrated around its mean we define additional random variables. Let $\varepsilon > 0$, to be defined later, and define three thresholds:

$$t_1 = pn + (a-\varepsilon)\sqrt{p(1-p)n},$$
$$t_2 = pn + a\sqrt{p(1-p)n},$$

and

$$t_3 = pn + (a+\varepsilon)\sqrt{p(1-p)n}.$$

For every $v \in V(G)$ define

$$F_v = \begin{cases} 0 & d(v) < t_1, \\ \frac{d(v)-t_1}{\varepsilon\sqrt{p(1-p)n}} & t_1 \leqslant d(v) \leqslant t_2, \\ 1 & d(v) > t_2, \end{cases}$$

and

$$G_v = \begin{cases} 0 & d(v) < t_2, \\ \frac{d(v)-t_2}{\varepsilon\sqrt{p(1-p)n}} & t_2 \leqslant d(v) \leqslant t_3, \\ 1 & d(v) > t_3. \end{cases}$$

Define $F = \sum_v F_v$ and $G = \sum_v G_v$. For every $v \in V$, we bound $\mathbb{E}F_v - \mathbb{E}X_v$ and $\mathbb{E}X_v - \mathbb{E}G_v$:

$$\mathbb{E}F_v - \mathbb{E}X_v = \sum_{i=t_1}^{t_2} p^i(1-p)^{n-1-i}\frac{i-t_1}{\varepsilon\sqrt{p(1-p)n}}\binom{n-1}{i} \tag{2.1}$$
$$\leqslant \sum_{i=t_1}^{t_2} p^i(1-p)^{n-1-i}\binom{n-1}{i}$$

$$\leqslant \varepsilon\sqrt{p(1-p)n}p^{p(n-1)}(1-p)^{(1-p)(n-1)}\binom{n-1}{\lfloor p(n-1)\rfloor} \tag{2.2}$$

$$\leqslant \frac{\varepsilon}{\sqrt{2\pi}}\left(1+O\left(\frac{1}{n}\right)\right) \leqslant \frac{2\varepsilon}{\sqrt{2\pi}}, \tag{2.3}$$

where the inequality in (2.2) follows from the fact that $p^i(1-p)^{n-1-i}\binom{n-1}{i}$ is maximized when $i = \lfloor p(n-1)\rfloor$, and the inequality in (2.3) follows from Stirling's approximation (see, *e.g.*, [1]):

$$n! = \sqrt{2\pi n}\left(\frac{n}{\mathrm{e}}\right)^n\left(1+O\left(\frac{1}{n}\right)\right).$$

Repeating this calculation for $\mathbb{E}X_v - \mathbb{E}G_v$ gives

$$\mathbb{E}X_v - \mathbb{E}G_v = \sum_{i=t_2}^{t_3} p^i(1-p)^{n-1-i}\left(1-\frac{i-t_2}{\varepsilon\sqrt{p(1-p)n}}\right)\binom{n-1}{i}$$

$$\leqslant \frac{\varepsilon}{\sqrt{2\pi}}\left(1+O\left(\frac{1}{n}\right)\right) \leqslant \frac{2\varepsilon}{\sqrt{2\pi}}. \tag{2.4}$$

From (2.1) and (2.4) we have that for $\lambda > 0$

$$\mathbb{P}\big(|X-\mathbb{E}X| \geqslant \lambda n\big) \leqslant \mathbb{P}\left(F-\mathbb{E}F \geqslant \left(\lambda - \frac{2\varepsilon}{\sqrt{2\pi}}\right)n\right)$$
$$+\,\mathbb{P}\left(G-\mathbb{E}G \leqslant -\left(\lambda - \frac{2\varepsilon}{\sqrt{2\pi}}\right)n\right).$$

Thus, we need to calculate the concentration of F and G. Both are edge exposure martingales with Lipschitz constant at most

$$\frac{2}{\varepsilon\sqrt{p(1-p)n}}.$$

Therefore, by Theorem 2.2 we get

$$\mathbb{P}\big(|X-\mathbb{E}X|\geqslant \lambda n\big) \leqslant 2\exp\left(-\frac{\left(\lambda-\frac{2\varepsilon}{\sqrt{2\pi}}\right)^2 n^2}{2\binom{n}{2}\left(\frac{2}{\varepsilon\sqrt{p(1-p)n}}\right)^2}\right)$$
$$\leqslant 2\exp\left(-\frac{p(1-p)}{4}\varepsilon^2\left(\lambda-\frac{2\varepsilon}{\sqrt{2\pi}}\right)^2 n\right).$$

Choosing $\lambda = 2c'n^{-\varepsilon'}$ and $\varepsilon = \frac{1}{2}\sqrt{2\pi}c'n^{-\varepsilon'}$ concludes the proof. □

In the next two lemmas we prove concentration results for the sizes of $\tilde{S}_1, V_1$ and their intersections with K.

Lemma 2.7. *Let*

$$\tilde{S}_1 = \{v \in S_1 : d_{S_1}(v) \geqslant p|S_1| + \eta\sqrt{p(1-p)|S_1|}\}.$$

Then, for every $0 < \varepsilon_1 < \frac{1}{4}$, $0 < \varepsilon_2 < \frac{1}{2}$, *and* $M > 0$,

$$\mathbb{P}\big(||\tilde{S}_1| - \gamma n| \geqslant Mn^{1-\varepsilon_1}\big) \leqslant \exp(-\Theta(n^{1-4\varepsilon_1})) \tag{2.5}$$

and

$$\mathbb{P}\big(||\tilde{S}_1 \cap K| - \delta k| \geqslant Mk^{1-\varepsilon_2}\big) \leqslant \exp(-\Theta(k^{1-2\varepsilon_2})). \tag{2.6}$$

The Θs in the exponents depend on p and on η.

Proof. We begin by proving inequality (2.5). From Lemma 2.4, we have that

$$\mathbb{P}\big(||S_1| - \alpha n| \geqslant Mn^{1-\varepsilon_1}\big) \leqslant \exp\big(-\Theta\big(n^{1-2\varepsilon_1}\big)\big).$$

Therefore,

$$\begin{aligned}\mathbb{P}\big(||\tilde{S}_1| - &\overline{\Phi}(\eta)\alpha n| \geqslant Mn^{1-\varepsilon_1}\big)\\ &\leqslant \exp\big(-\Theta\big(n^{1-2\varepsilon_1}\big)\big) + \mathbb{P}\big(||S_1| - \alpha n| \leqslant M'n^{1-\varepsilon_1} \wedge ||\tilde{S}_1| - \overline{\Phi}(\eta)\alpha n| \geqslant Mn^{1-\varepsilon_1}\big)\\ &\leqslant \exp\big(-\Theta\big(n^{1-2\varepsilon_1}\big)\big) + \mathbb{P}\big(||\tilde{S}_1| - \overline{\Phi}(\eta)|S_1|| \geqslant \big(M - \overline{\Phi}(\eta)M'\big)n^{1-\varepsilon_1}\big).\end{aligned}$$

Now, by Lemma 2.6, we have

$$\mathbb{P}\big(||\tilde{S}_1| - \overline{\Phi}(\eta)|S_1|| \geqslant \big(M - \overline{\Phi}(\eta)M'\big)n^{1-\varepsilon_1}\big) \leqslant \exp\big(-\Theta\big(n^{1-4\varepsilon_1}\big)\big).$$

This concludes the proof of inequality (2.5).

To prove inequality (2.6), consider a dense graph vertex $v \in S_1$. Notice that if $d_{S_1}(v) \geqslant p|S_1| + \eta\sqrt{p(1-p)|S_1|}$, then

$$d_{S_1 \setminus K}(v) \geqslant p|S_1 \setminus K| + \left(\eta - \sqrt{\frac{1-p}{p}}\frac{|S_1 \cap K|}{\sqrt{|S_1|}}\right)\sqrt{\frac{|S_1|}{|S_1 \setminus K|}}\sqrt{p(1-p)|S_1 \setminus K|}.$$

On the other hand, if $d_{S_1}(v) < p|S_1| + \eta\sqrt{p(1-p)|S_1|}$, then

$$d_{S_1 \setminus K}(v) < p|S_1 \setminus K| + \left(\eta + \sqrt{\frac{p}{1-p}}\frac{|S_1 \cap K|}{\sqrt{|S_1|}}\right)\sqrt{\frac{|S_1|}{|S_1 \setminus K|}}\sqrt{p(1-p)|S_1 \setminus K|}.$$

Therefore,

$$\begin{aligned}\mathbb{P}\big(||\tilde{S}_1 \cap K| - \delta k| \geqslant Mk^{1-\varepsilon_2}\big) &\leqslant \mathbb{P}\big(|\tilde{S}_1 \cap K| - \delta'\alpha k \geqslant Mk^{1-\varepsilon_2}\big)\\ &\quad + \mathbb{P}\big(|\tilde{S}_1 \cap K| - \delta''\alpha k \leqslant -Mk^{1-\varepsilon_2}\big),\end{aligned}$$

where

$$\delta' = \overline{\Phi}\left(\left(\eta - \sqrt{\frac{1-p}{p}}\frac{|S_1 \cap K|}{\sqrt{|S_1|}}\right)\sqrt{\frac{|S_1|}{|S_1 \setminus K|}}\right)$$

and

$$\delta'' = \overline{\Phi}\left(\left(\eta + \sqrt{\frac{p}{1-p}}\frac{|S_1 \cap K|}{\sqrt{|S_1|}}\right)\sqrt{\frac{|S_1|}{|S_1 \setminus K|}}\right).$$

By Lemma 2.4, we can condition on the events

$$||S_1| - \alpha n| \leqslant Mn^{1-\varepsilon_1}$$

and

$$||S_1 \cap K| - \alpha k| \leqslant M'k^{1-\varepsilon_2},$$

and replace δ' and δ'' by constants v' and v'' such that

$$v' \leqslant \overline{\Phi}\left(\left(\eta - \sqrt{\frac{1-p}{p}}\left(c\sqrt{\alpha} - \Theta\left(\frac{1}{n^{\varepsilon_1}}\right) - \Theta\left(\frac{1}{k^{\varepsilon_2}}\right)\right)\right)\left(1 + \Theta\left(\frac{1}{k}\right) + \Theta\left(\frac{1}{n^{\varepsilon_1}}\right)\right)\right)$$

and

$$v'' \geqslant \overline{\Phi}\left(\left(\eta + \sqrt{\frac{p}{1-p}}\left(c\sqrt{\alpha} - \Theta\left(\frac{1}{n^{\varepsilon_1}}\right) - \Theta\left(\frac{1}{k^{\varepsilon_2}}\right)\right)\right)\left(1 + \Theta\left(\frac{1}{n^{\varepsilon_1}}\right)\right)\right).$$

Now, we start by bounding $\mathbb{P}\big(|\tilde{S}_1 \cap K| - v'\alpha k \geqslant Mk^{1-\varepsilon_2}\big)$. Recall that we have conditioned on the event $||S_1 \cap K| - \alpha k| \leqslant M'k^{1-\varepsilon_2}$, and therefore, by replacing αk with its lower bound $|S_1 \cap K| - M'k^{1-\varepsilon_2}$, Corollary 2.5 gives us that

$$\begin{aligned}
&\mathbb{P}\big(|\tilde{S}_1 \cap K| - v'\alpha k \geqslant Mk^{1-\varepsilon_2}\big)\\
&\quad \leqslant \mathbb{P}\big(||\tilde{S}_1 \cap K| - v'|S_1 \cap K|| \geqslant (M - v'M')k^{1-\varepsilon_2}\big)\\
&\quad \leqslant \exp\big(-\Theta\big(k^{1-2\varepsilon_2}\big)\big).
\end{aligned}$$

Using Corollary 2.5 and replacing αk with its upper bound $|S_1 \cap K| + M'k^{1-\varepsilon_2}$ gives us the second bound

$$\begin{aligned}
&\mathbb{P}\big(|\tilde{S}_1 \cap K| - v''\alpha k \leqslant -Mk^{1-\varepsilon_2}\big)\\
&\quad \leqslant \mathbb{P}\big(||\tilde{S}_1 \cap K| - v''|S_1 \cap K|| \geqslant (M - v''M')k^{1-\varepsilon_2}\big)\\
&\quad \leqslant \exp\big(-\Theta\big(k^{1-2\varepsilon_2}\big)\big),
\end{aligned}$$

which concludes the proof. □

Lemma 2.8. *Let*

$$V_1 = \left\{v \in V \setminus S_1 : d_{\tilde{S}_1}(v) \geqslant p|\tilde{S}_1| + \beta\sqrt{p(1-p)|\tilde{S}_1|}\right\}.$$

For every $0 < \varepsilon_1 < \frac{1}{4}$ *and* $0 < \varepsilon_2 < \frac{1}{2}$, *the set* V_1 *satisfies*

$$\mathbb{P}\big(||V_1| - \tau n| \geqslant Mn^{1-\varepsilon_1}\big) \leqslant \exp\big(-\Theta(n^{1-2\varepsilon_1})\big) \tag{2.7}$$

and

$$\mathbb{P}\big(||V_1 \cap K| - \rho k| \geqslant Mk^{1-\varepsilon_2}\big) \leqslant \exp\big(-\Theta(k^{1-2\varepsilon_2})\big). \tag{2.8}$$

The Θ*s in the exponents depend on* p *and on* β.

Proof. To prove (2.7), we consider the vertices in $(V \setminus S_1) \setminus K$. From Lemma 2.4, we have that

$$\mathbb{P}\big(||V \setminus S_1| - (1-\alpha)n| \geqslant Mn^{1-\varepsilon_1}\big) \leqslant \exp\big(-\Theta(n^{1-2\varepsilon_1})\big).$$

Therefore, we can condition on the event $||V \setminus S_1| - (1-\alpha)n| \leqslant M'n^{1-\varepsilon_1}$. Replacing $(1-\alpha)n$ with its upper and lower bounds gives that

$$\begin{aligned}
&\mathbb{P}\big(||V_1| - \overline{\Phi}(\beta)(1-\alpha)n| \geqslant Mn^{1-\varepsilon_1}\big)\\
&\quad \leqslant \exp\big(-\Theta(n^{1-2\varepsilon_1})\big) + \mathbb{P}\big(||V_1| - \overline{\Phi}(\beta)|V \setminus S_1|| \geqslant \big(M - \overline{\Phi}(\beta)M'\big)n^{1-\varepsilon_1}\big).
\end{aligned}$$

Now, by Corollary 2.5, we have

$$\mathbb{P}\big(||V_1| - \overline{\Phi}(\beta)|V \setminus S_1|| \geqslant (M - \overline{\Phi}(\beta)M')n^{1-\varepsilon_1}\big) \leqslant \exp\big(-\Theta(n^{1-2\varepsilon_1})\big).$$

This concludes the proof of inequality (2.7).

To prove (2.8), we consider a dense graph vertex $v \in V \setminus S_1$. If

$$d_{\tilde{S}_1}(v) \geqslant p|\tilde{S}_1| + \beta\sqrt{p(1-p)|\tilde{S}_1|},$$

then

$$d_{\tilde{S}_1 \setminus K}(v) \geqslant p|\tilde{S}_1 \setminus K| + \left(\beta - \sqrt{\frac{1-p}{p}}\frac{|\tilde{S}_1 \cap K|}{\sqrt{|\tilde{S}_1|}}\right)\sqrt{\frac{|\tilde{S}_1|}{|\tilde{S}_1 \setminus K|}}\sqrt{p(1-p)|\tilde{S}_1 \setminus K|}.$$

On the other hand, if

$$d_{\tilde{S}_1}(v) < p|\tilde{S}_1| + \beta\sqrt{p(1-p)|\tilde{S}_1|},$$

then

$$d_{\tilde{S}_1 \setminus K}(v) < p|\tilde{S}_1 \setminus K| + \left(\beta + \sqrt{\frac{p}{1-p}}\frac{|\tilde{S}_1 \cap K|}{\sqrt{|\tilde{S}_1|}}\right)\sqrt{\frac{|\tilde{S}_1|}{|\tilde{S}_1 \setminus K|}}\sqrt{p(1-p)|\tilde{S}_1 \setminus K|}.$$

Therefore,

$$\begin{aligned}\mathbb{P}\big(||V_1 \cap K| - \rho k| \geqslant Mk^{1-\varepsilon_2}\big) &\leqslant \mathbb{P}\big(|V_1 \cap K| - \rho'(1-\alpha)k \geqslant Mk^{1-\varepsilon_2}\big)\\ &\quad + \mathbb{P}\big(|V_1 \cap K| - \rho''(1-\alpha)k \leqslant -Mk^{1-\varepsilon_2}\big),\end{aligned}$$

where

$$\rho' = \overline{\Phi}\left(\left(\beta - \sqrt{\frac{1-p}{p}}\frac{|\tilde{S}_1 \cap K|}{\sqrt{|\tilde{S}_1|}}\right)\sqrt{\frac{|\tilde{S}_1|}{|\tilde{S}_1 \setminus K|}}\right)$$

and

$$\rho'' = \overline{\Phi}\left(\left(\beta + \sqrt{\frac{p}{1-p}}\frac{|\tilde{S}_1 \cap K|}{\sqrt{|\tilde{S}_1|}}\right)\sqrt{\frac{|\tilde{S}_1|}{|\tilde{S}_1 \setminus K|}}\right).$$

Again, by Lemma 2.4 we can condition on the events

$$||V \setminus S_1| - (1-\alpha)n| \leqslant Mn^{1-\varepsilon_1}$$

and

$$||(V \setminus S_1) \cap K| - (1-\alpha)k| \leqslant M'k^{1-\varepsilon_2},$$

and replace ρ' and ρ'' by constants v' and v'' such that

$$v' \leqslant \overline{\Phi}\left(\left(\beta - \sqrt{\frac{1-p}{p}}\left(\frac{c\delta}{\sqrt{\gamma}} - \Theta\left(\frac{1}{n^{\varepsilon_1}}\right) - \Theta\left(\frac{1}{k^{\varepsilon_2}}\right)\right)\right)\left(1 + \Theta\left(\frac{1}{k}\right) + \Theta\left(\frac{1}{n^{\varepsilon_1}}\right)\right)\right)$$

and

$$v'' \geqslant \overline{\Phi}\left(\left(\beta + \sqrt{\frac{p}{1-p}}\left(\frac{c\delta}{\gamma} - \Theta\left(\frac{1}{n^{\varepsilon_1}}\right) - \Theta\left(\frac{1}{k^{\varepsilon_2}}\right)\right)\right)\left(1 + \Theta\left(\frac{1}{n^{\varepsilon_1}}\right)\right)\right).$$

Now, we start by bounding

$$\mathbb{P}\big(|V_1 \cap K| - v'(1-\alpha)k \geqslant Mk^{1-\varepsilon_2}\big).$$

Recall that we have conditioned on the event

$$||(V \setminus S_1) \cap K| - (1-\alpha)k| \leqslant M'k^{1-\varepsilon_2}.$$

Therefore, by replacing $(1-\alpha)k$ with its lower bound $|(V \setminus S_1) \cap K| - M'k^{1-\varepsilon_2}$, Corollary 2.5 gives us that

$$\begin{aligned}&\mathbb{P}\big(||V_1 \cap K| - v'(1-\alpha)k| \geqslant Mk^{1-\varepsilon_2}\big)\\ &\qquad \leqslant \mathbb{P}\big(||V_1 \cap K| - v'|(V \setminus S_1) \cap K| \geqslant \big(M - v'M'\big)k^{1-\varepsilon_2}\big) \leqslant \exp\big(-\Theta(k^{1-2\varepsilon_2})\big).\end{aligned}$$

Using Corollary 2.5 and replacing $(1-\alpha)k$ with its upper bound $|(V \setminus S_1) \cap K| + M'k^{1-\varepsilon_2}$ gives us the second bound

$$\begin{aligned}&\mathbb{P}\big(||V_1 \cap K| - v''(1-\alpha)k| \leqslant -Mk^{1-\varepsilon_2}\big)\\ &\qquad \leqslant \mathbb{P}\big(||V_1 \cap K| - v''|(V \setminus S_1) \cap K| \geqslant \big(M - v''M'\big)k^{1-\varepsilon_2}\big) \leqslant \exp\big(-\Theta(k^{1-2\varepsilon_2})\big),\end{aligned}$$

which concludes the proof. □

3. Proof of Theorem 1.8

In order to prove the correctness of the algorithm, we first calculate the success probability in each of the three phases, assuming that t is still an unknown parameter with $t > C \log\log n$ for some sufficiently large constant C. In Lemma 3.1 we calculate the success probability of the first phase, in Lemma 3.2 we calculate the success probability of the second phase conditioned on the success of the first phase, and in Lemma 3.4 we calculate the success probability of the third phase, conditioned on the success of the first and second phases. We then optimize the values of α, β, η, t to minimize the failure probability of the algorithm in Lemma 3.5. The logarithms in the definitions and lemmas in this section are natural logarithms.

Lemma 3.1. *Let $G \sim G(n,p,k,q)$, where $k = c\sqrt{n}$ and $c > c^*$. Then if we run the algorithm with parameters α, β, η and $t > C \log\log n$, the failure probability of the first phase is at most*

$$t\big(\exp(-\Theta(\tau^{t(1-4\varepsilon_1)}n^{1-4\varepsilon_1})) + \exp(-\Theta(\rho^{t(1-2\varepsilon_2)}k^{1-2\varepsilon_2}))\big)$$

Proof. This is a simple consequence of Lemma 2.8 and the union bound. □

Next we prove a lemma about graphs distributed as $G(m,p,r,q)$. This lemma can be applied to all values of m and r that are relevant to our algorithm, *i.e.*, to $n_0, n_1, \dots$ and $k_0, k_1, \dots$.

Lemma 3.2. *Let $G \sim G(m,p,r,q)$. Define the thresholds*

$$D_1 = pm + \frac{1}{3}(q-p)r$$

and

$$D_2 = pm + \frac{2}{3}(q-p)r - q.$$

Then

$$\mathbb{P}\big(\exists v \in V \setminus K : d(v) \geqslant D_1 \text{ or } \exists u \in K : d(u) < D_2\big) \leqslant (m+r)\exp(-(2(q-p)^2 r^2/9m)).$$

Proof. By Theorem 2.2

$$\begin{aligned}\mathbb{P}\big(\exists v \notin K : d(v) \geqslant D_1\big) &\leqslant m\mathbb{P}\big(B(m,p) \geqslant D_1\big)\\ &\leqslant m\mathbb{P}\left(|B(m,p) - pm| \geqslant \frac{1}{3}(q-p)r\right)\\ &\leqslant 2m\exp(-2(q-p)^2 r^2/9m).\end{aligned}$$

On the other hand,

$$\begin{aligned}\mathbb{P}\big(\exists u \in K : d(u) < D_2\big) &\leqslant r\mathbb{P}\big(B(m-r,p) + B(r-1,q) \leqslant D_2\big)\\ &\leqslant r\mathbb{P}\left(|B(m-r,p) + B(r-1,q) - p(m-r) - q(r-1)| \geqslant \frac{1}{3}(q-p)r\right)\\ &\leqslant 2r\exp(-2(q-p)^2 r^2/9m).\end{aligned}$$

Therefore, the probability that there exists a vertex $v \in V \setminus K$ such that $d(v) \geqslant D_1$, or a vertex $u \in K$ such that $d(u) < D_2$, is at most $2(m+r)\exp(-2(q-p)^2 r^2/9m)$. □

Corollary 3.3. *There exists a constant C such that, for $t > C\log\log n$, if $\tilde{K}$ is the set produced by the second phase of the algorithm after t iterations of the first phase, then, conditioned on the success of all t iterations, with probability at least*

$$1 - \exp\left(-\Theta\left(\left(\frac{\rho^2}{\tau}\right)^t\right)\right),$$

we have $\tilde{K} = V_t \cap K$.

Proof. The algorithm estimates $\tilde{k}_t$, the number of hidden dense graph vertices in G_t, by $k_t = \rho^t k$. If the input graph has n vertices and a hidden dense graph K of size $k = c\sqrt{n}$, and all the iterations are successful, then $|\tilde{k}_t - k_t| \leqslant O(k_t^{1-\varepsilon_2})$. In this case, by Lemma 3.2, with probability at least

$$1 - \exp\left(-\Theta\left(\frac{\rho^{2t}k^2}{\tau^t n}\right)\right)$$

every vertex $v \in V_t \setminus K$ has

$$d(v) < p\tilde{n}_t + \frac{1}{3}(q-p)\tilde{k}_t \leqslant p\tilde{n}_t + \frac{1}{3}(q-p)\big(k_t + O(k_t^{1-\varepsilon_2})\big) < p\tilde{n}_t + \frac{1}{2}(q-p)k_t,$$

and every vertex $u \in V_t \cap K$ has

$$d(u) \geqslant p\tilde{n}_t + \frac{2}{3}(q-p)\tilde{k}_t - q \geqslant p\tilde{n}_t + \frac{2}{3}(q-p)\big(k_t - O(k_t^{1-\varepsilon_2})\big) > p\tilde{n}_t + \frac{1}{2}(q-p)k_t.$$

Recall that $\tilde{K}$ is defined as the subset of vertices of G_t with degree above $p\tilde{n}_t + \frac{1}{2}(q-p)k_t$, and therefore, with probability at least

$$1 - \exp\left(-\Theta\left(\frac{\rho^{2t}k^2}{\tau^t n}\right)\right),$$

$\tilde{K}$ is exactly equal to $V_t \cap K$. □

Using the bound on t and the fact that $\rho > \sqrt{\tau}$, the expression

$$\exp\left(-\Theta\left(\frac{\rho^{2t}k^2}{\tau^t n}\right)\right)$$

becomes $o(1)$. The last step in proving the correctness of the algorithm is proving the correctness of the third phase.

3.1. Finding hidden dense graphs from partial information

In this subsection we consider a problem of independent interest: reconstructing a hidden dense graph given a small part of it (possibly chosen by an adversary). This is used in Lemma 3.5 to bound the failure probability of the third phase.

Lemma 3.4. *Let $G \sim G(n,p,r,q)$ be a random graph with a hidden dense graph K of size r, where $r \geqslant \Omega(\log n)$. Let A be an arbitrary subset of K of size $s \leqslant r$. Suppose that either*

(1) $r = O(\log n \log\log n)$ *and* $s \geqslant \left(\frac{2}{(q-p)^2} + \varepsilon\right)\log n$ *for some constant* $\varepsilon > 0$, *or*
(2) $r \geqslant \omega(\log n \log\log n)$ *and* $s \geqslant \frac{2}{(q-p)^2}\log n + 1$.

Let

$$K' = A \cup \left\{v \in G : d_A(v) \geqslant \frac{1}{2}(p+q)s\right\}.$$

Define K^* to be the set of vertices of G that have at least $\frac{1}{2}(p+q)r$ neighbours in K'. Then, for every $0 < \varepsilon_3 < \frac{1}{2}$,

$$\mathbb{P}\big(K^* \neq K\big) \leqslant \exp(-\Theta(s\log r + \log n)) + \exp(-\Theta(r^{1-2\varepsilon_3})).$$

Proof. Consider an arbitrary subset $B \subseteq K$ of size s. For every such set, we define B_{bad} as the set of vertices in $V \setminus B$ that are either in K and have less than $\frac{1}{2}(p+q)s$ neighbours in B, or not in K and have more than $\frac{1}{2}(p+q)s$ neighbours in B. By Theorem 2.2, the probability that a specific vertex $v \notin K$ has more than $\frac{1}{2}(p+q)s$ neighbours in B is at most $\exp(-(q-p)^2 s/2)$. The probability that a specific vertex $v \in K$ has less than

$\frac{1}{2}(p+q)s$ neighbours in B is also at most $\exp(-(q-p)^2 s/2)$. Therefore, the probability that $|B_{\text{bad}}| \geqslant l_0$ is at most $\sum_{l=l_0}^{n} n^l \exp(-(q-p)^2 sl/2)$. Taking the union bound over all subsets of size s of K gives that the probability that there exists a subset B with $|B_{\text{bad}}| \geqslant l_0$ is at most

$$r^s \sum_{l=l_0}^{n} \exp\left(l\left(\log n - \frac{(q-p)^2 s}{2}\right)\right) \leqslant n \exp\left(s \log r - l_0\left(\frac{(q-p)^2 s}{2} - \log n\right)\right)$$
$$= \exp\left(\log n + s \log r - l_0\left(\frac{(q-p)^2 s}{2} - \log n\right)\right).$$

Therefore, this is also a bound on the probability that $|A_{\text{bad}}| \geqslant l_0$. By our assumptions on s, we know that $\frac{(q-p)^2 s}{2} - \log n$ is positive. Therefore, we can take

$$l_0 = \frac{2(\log n + s \log r)}{(q-p)^2 s/2 - \log n},$$

and get that the probability that $|A_{\text{bad}}| \geqslant l_0$ is at most $\exp(-\log n - s \log r)$. Therefore, with probability at least $1 - \exp(-\log n - s \log r)$ there are at most l_0 bad vertices in K'. Specifically, this implies that K' contains at least $r - l_0$ vertices from K and at most l_0 vertices not from K, and that $|K'| \leqslant r + l_0$. By Theorem 2.2 and the union bound, the probability that there exists a vertex $v \in K$ with less than $qr - r^{1-\varepsilon_3}$ neighbours in K is at most $\exp(-\Theta(r^{1-2\varepsilon_3}))$, and so is the probability that there exists a vertex $v \notin K$ with more than $pr + r^{1-\varepsilon_3}$ neighbours in K. Therefore, with probability at least $1 - \exp(-\Theta(r^{1-2\varepsilon_3}))$ the number of neighbours every $v \in K$ has in K' is at least $qr - r^{1-\varepsilon_3} - l_0$, and the number of neighbours every $v \notin K$ has in K' is at most $pr + r^{1-\varepsilon_3} + l_0$. Thus, if s and r are such that $l_0 = o(r)$ then

$$\mathbb{P}\big(K^* \neq K\big) \leqslant \exp(-\log n - s \log r) + \exp(-\Theta(r^{1-2\varepsilon_3})).$$

If $r = \omega(\log n \log\log n)$, then letting $s = \frac{2}{(q-p)^2} \log n + 1$ gives

$$l_0 = \frac{4}{(q-p)^2}\left(\log n + \frac{2}{(q-p)^2} \log n \log r + \log r\right).$$

Clearly, $\log n + \log r = o(r)$. To see that $\log n \log r = o(r)$, denote

$$r = \log n f(n),$$

where

$$f(n) = \omega(\log\log n).$$

Then

$$\log n \log r = \log n\big(\log\log n + \log\big(f(n)\big)\big).$$

Clearly,

$$\log n \log(f(n)) = o(\log n f(n)),$$

and from the definition of $f(n)$ we also have

$$\log n \log\log n = o(\log n f(n)).$$

If $r \leqslant O(\log n \log\log n)$, then letting

$$s = \left(\frac{2}{(q-p)^2} + \varepsilon\right)\log n$$

for some small constant $\varepsilon > 0$ is enough, since then

$$l_0 = \frac{4}{(q-p)^2\varepsilon} + \frac{4\left(\frac{2}{(q-p)^2} + \varepsilon\right)}{(q-p)^2\varepsilon}\log r = o(r). \qquad \square$$

3.2. Bounding the failure probability

Lemma 3.5. *For every $c > c^*$, there exist $0 < \alpha < 1$ and $\beta, \eta > 0$ such that if we define τ, ρ as in Definition 1.4, and*

$$a = -\frac{\log \tau}{\log \frac{\rho^2}{\tau}},$$

then for every $\varepsilon_0 < \frac{1}{a}$, the failure probability of the algorithm, when the first phase is iterated

$$t = \frac{\varepsilon_4 \log n}{\log \frac{\rho^2}{\tau}}$$

times for some $0 < \varepsilon_4 < \frac{1}{a}$, is at most $\exp(-\Theta(n^{\varepsilon_0}))$.

Proof. In order for the probability proved in Corollary 3.3 to tend to 0, we need τ and ρ to satisfy $\frac{\rho}{\sqrt{\tau}} > 1$. From Definition 1.7 we know that there exists a constant $c^* > 0$ such that for every $c > c^*$ there exist α, β, η that satisfy this inequality.

Denote

$$b = -\frac{\log \rho^2}{\log \frac{\rho^2}{\tau}}.$$

By Lemma 3.1, the failure probability during the iteration phase of the algorithm can be bounded above by

$$\exp(-\Theta(n^{(1-4\varepsilon_1)(1-\varepsilon_4 a)})) + \exp(-\Theta(n^{\frac{1}{2}(1-2\varepsilon_2)(1-\varepsilon_4 b)})).$$

By Corollary 3.3, the failure probability in the second phase of the algorithm is at most

$$\exp(-\Theta(n^{\varepsilon_4})).$$

Finally, if t is as defined above, then assuming that the first two phases succeed, we know that

$$|\tilde{K}| \geqslant \rho^t k - o(\rho^t k) = k^{1-b\varepsilon_4}(1 - o(1))$$

(notice that $b = a - 1$ so $\varepsilon_4 < \frac{1}{a}$ implies that $1 - b\varepsilon_4 > 0$). The set $\tilde{K}$ is sufficiently large that we can use Lemma 3.4 to conclude that the probability of failing in the third phase is at most

$$\exp(-\Theta(n^{\frac{1}{2}(1-\varepsilon_4 b)}\log n)) + \exp(-\Theta(k^{1-2\varepsilon_3})).$$

For any choice of $0 < \varepsilon_1 < \frac{1}{4}$, $0 < \varepsilon_2 < \frac{1}{2}$ and $0 < \varepsilon_4 < \frac{1}{a}$, denote

$$\varepsilon_0 = \min\left\{\varepsilon_4, (1-4\varepsilon_1)(1-\varepsilon_4 a), \frac{1}{2}(1-2\varepsilon_2)(1-\varepsilon_4 b)\right\},$$

and take $\varepsilon_3 = \frac{1-2\varepsilon_0}{2}$ (notice that $\varepsilon_3 > 0$ because $\varepsilon_0 < \frac{1}{2}$). With these parameters, the failure probability of the whole algorithm is at most $\exp(-\Theta(n^{\varepsilon_0}))$. $\square$

3.3. Analysis of the running time

To find the total running time of the algorithm, we calculate the running time of each of the three phases. The running time of the first phase is $O(n^2)$, since every edge that is examined in an iteration i does not participate in iteration $i+1$: it is either an edge within S_i or an edge between $V_i \setminus S_i$ and $\tilde{S}_i$. Since none of the vertices of S_i participate in subsequent iterations, none of these edges participate either.

In the second phase, we calculate the degrees of the vertices in V_t. In this calculation, each edge in G_t is examined at most twice, so the running time of this phase is $O(n^2)$.

In the third phase, we examine each edge in the graph at most twice: once when counting the number of neighbours of each vertex in $\tilde{K}$, and once when counting the number of neighbours in K'. Thus, the running time of the third phase is also $O(n^2)$.

More precise analysis of the second and third phases would yield asymptotically smaller running times, since $|V_t|, |\tilde{K}|, |K'| = o(n)$. Because of the first phase, the total running time of the algorithm cannot be asymptotically smaller than n^2, so we leave the analysis as is.

4. Discussion

As mentioned in the Introduction, the algorithm presented in this paper is also an algorithm for finding hidden cliques, since $G(n, \frac{1}{2}, k)$ is equivalent to $G(n, \frac{1}{2}, k, 1)$. Numerical calculations show that for $p = \frac{1}{2}$ and $q = 1$, the value of c^* is close to 1.261, which means that our algorithm can solve the hidden clique problem for any $k \geqslant 1.261\sqrt{n}$.

In the case of the hidden clique model, we can use a slightly simpler version of the algorithm. In the third phase, after finding K' we can simply let K^* be the set of k highest degree vertices in the graph induced by the vertices in K', and we do not need to go over the rest of the vertices in the graph to calculate their degree in K'. This simplification is possible due to the fact that when the dense graph is a clique, *all* the hidden clique vertices have at least $\frac{1}{2}(p+q)|\tilde{K}|$ (which is $\frac{3}{4}|\tilde{K}|$ in the hidden clique case) neighbours in $\tilde{K}$. In fact, conditioned on the success of the second phase, all the hidden clique vertices have $|\tilde{K}|$ neighbours in $\tilde{K}$. Furthermore, it can be easily proved that with high probability there are very few non-clique vertices in K', and that their degree is smaller than that of the clique vertices. On the other hand, when the dense graph is not a clique, we can only prove (as we do in Lemma 3.4) that with high probability the set K' contains *most* of the dense graph vertices and a few non-dense-graph vertices, which means that in order to find *all* of the hidden dense graph vertices we need to examine the vertices that are not in K'.

Our results bring up some interesting questions for future research. For example, one of the advantages of the algorithm presented here is a failure probability of at most $\exp(-n^{\varepsilon})$

for some $\varepsilon > 0$. Experimental results shown by Feige and Ron [12] suggest that the failure probability of the algorithm described there may also be $o(1)$. Whether the analysis can be improved to prove this rigorously is an interesting open question. One can also ask whether Alon, Krivelevich and Sudakov's analysis [3] can be improved to show failure probability of at most $\exp(-n^{\varepsilon})$ for some $0 < \varepsilon < 1$.

Aside from the most interesting open question of whether there exists an algorithm that finds hidden cliques for $k = o(\sqrt{n})$, one can ask about ways to find hidden cliques of size $k = c\sqrt{n}$ for small values of c. Alon, Krivelevich and Sudakov [3] show how to improve the constant for which their algorithm works, at the expense of increasing the running time. This technique can be used for any algorithm that finds hidden cliques, so we describe it here. Pick a random vertex $v \in V$, and run the algorithm only on the subgraph containing v and its neighbourhood. If v is a clique vertex, then the parameters of the algorithm have improved, since instead of having a graph with n vertices and a hidden clique of size $c\sqrt{n}$ we now have a graph with $\frac{n}{2}$ vertices and a hidden clique of size $c\sqrt{n}$. The expected number of trials we need to do until we pick a clique vertex is $O(\sqrt{n})$. This means that if we have an algorithm that finds a hidden clique of size $c\sqrt{n}$, where $c \geqslant c_0$, we can also find a hidden clique for $c \geqslant \frac{c_0}{\sqrt{2}}$, while increasing the running time by a factor of $\sqrt{n}$. If we wish to improve the constant even further, we can pick r random vertices and run the algorithm on the subgraph containing them and their common neighbourhood. This gives an algorithm that works for constants smaller by up to a factor of $2^{r/2}$ than the original constant, at the expense of increasing the running time of the algorithm by a factor of $n^{r/2}$.

The technique described here can be generalized for $G(n, p, k, q)$ in the following way. Instead of picking r random vertices and running the algorithm on them and their common neighbourhood, we pick a set of r random vertices (denoted R) and run the algorithm on the graph containing R and all the vertices with at least $\frac{1}{2}(p+q)r$ neighbours in R (denoted $G(R)$). We call the set R 'good' if $G(R)$ contains at least $\frac{99k}{100}$ vertices of the hidden dense graph. The probability that the set R is good can be bounded above by the probability that R is contained in the hidden dense graph (this probability is $\Theta(n^{-r/2})$), multiplied by the probability that $G(R)$ contains at least $\frac{99k}{100}$ hidden dense graph vertices, conditioned on R being contained in the hidden dense graph. The latter probability can be estimated in the following way. By Theorem 2.2, the probability that a hidden dense graph vertex has less than $\frac{1}{2}(p+q)r$ neighbours in R is at most

$$\exp\left(-\frac{1}{2}(q-p)^2 r\right).$$

Therefore, by the union bound, the probability that more than $\frac{k}{100}$ hidden dense graph vertices have less than $\frac{1}{2}(p+q)r$ neighbours in R is at most

$$\binom{k}{0.01k} \exp\left(-\frac{1}{2}(q-p)^2 r \frac{k}{100}\right).$$

If we choose r to be a large enough constant, then this expression is $\exp(-O(k))$. Therefore, if R is contained in the hidden dense graph, then with high probability R is good, so the expected number of trials we have to do until we pick a good set R is $O(n^{r/2})$. The expected number of non-dense graph vertices in $G(R)$ is $\exp(-\frac{1}{2}(q-p)^2 r)n$, so if we pick a good set R, the parameters of the algorithm change from n vertices and $k = c\sqrt{n}$ hidden dense graph vertices to $\exp(-\frac{1}{2}(q-p)^2 r)n$ vertices and $\frac{99k}{100} = \frac{99}{100}c\sqrt{n}$ hidden dense graph vertices. Thus, an algorithm that finds hidden dense graphs for $c \geqslant c^*$ can be improved to an algorithm that finds hidden dense graphs for

$$c \geqslant \frac{100}{99}\exp\left(-\frac{1}{4}(q-p)^2 r\right)c^*$$

at the expense of increasing the expected running time by a factor of $n^{r/2}$.

In this section we have described a sequence of algorithms whose running times increase by factors of $\sqrt{n}$. It is not known whether the constant can be decreased if we can only increase the running time by a factor smaller than $\sqrt{n}$.

Question. Given an algorithm that runs in time $O(n^2)$ and finds hidden dense graphs of size $c\sqrt{n}$ for any $c \geqslant c^*$, is there an algorithm that runs in time $O(n^{2+\varepsilon})$, where $\varepsilon < \frac{1}{2}$, and finds hidden dense graphs of size $c\sqrt{n}$ where $c < c^*$? How small can c be as a function of ε, p and q?

References

[1] Abramowitz, M. and Stegun, I. A., eds (1964) *Handbook of Mathematical Functions with Formulas, Graphs, and Mathematical Tables*, Dover.

[2] Alon, N., Andoni, A., Kaufman, T., Matulef, K., Rubinfeld, R. and Xie, N. (2007) Testing k-wise and almost k-wise independence. In *STOC*, ACM, pp. 496–505.

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

[4] Arora, S. and Safra, S. (1998) Probabilistic checking of proofs: A new characterization of NP. *J. Assoc. Comput. Mach.* **45** 70–122.

[5] Arora, S., Lund, C., Motwani, R., Sudan, M. and Szegedy, M. (1992) Proof verification and hardness of approximation problems. In *FOCS*, ACM, pp. 14–23.

[6] Azuma, K. (1967) Weighted sums of certain dependent random variables. *Tohoku Math. J.* **19** 357–367.

[7] Ben-Dor, A., Shamir, R. and Yakhini, Z. (1999) Clustering gene expression patterns. *J. Comput. Biol.* **6** 281–297.

[8] Ben Sasson, E., Bilu, Y. and Gutfreund, D. (2002) Finding a randomly planted assignment in a random 3CNF. Manuscript.

[9] Berry, A. C. (1941) The accuracy of the Gaussian approximation to the sum of independent variates. *Trans. Amer. Math. Soc.* **49** 122–136.

[10] Durrett, R. (2010) *Probability: Theory and Examples*, fourth edition, Cambridge University Press.

[11] Esseen, C. G. (1942) On the Liapunoff limit of error in the theory of probability. *Arkiv för Matematik, Astronomi och Fysik* **A28** 1–19.

[12] Feige, U. and Ron, D. (2010) Finding hidden cliques in linear time. In *AOFA*, DMTCS.

[13] Feige, U., Goldwasser, S., Lovász, L., Safra, S. and Szegedy, M. (1991) Approximating clique is almost NP-complete (preliminary version). In *FOCS*, IEEE Computer Society, pp. 2–12.

[14] Feige, U. and Krauthgamer, R. (2000) Finding and certifying a large hidden clique in a semirandom graph. *Random Struct. Alg.* **16** 195–208.

[15] Feige, U. and Krauthgamer, R. (2003) The probable value of the Lovász–Schrijver relaxations for maximum independent set. *SIAM J. Comput.* **32** 345–370.

[16] Frieze, A. M. and Kannan, R. (2008) A new approach to the planted clique problem. In *FSTTCS*, Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik, pp. 187–198.

[17] Grimmett, G. and McDiarmid, C. (1975) On colouring random graphs. *Math. Proc. Cam. Phil. Soc.* **77** 313–324.

[18] Hazan, E. and Krauthgamer, R. (2009) How hard is it to approximate the best Nash equilibrium? In *SODA*, Society for Industrial and Applied Mathematics, pp. 720–727.

[19] Hoeffding, W. (1963) Probability inequalities for sums of bounded random variables. *J. Amer. Statist. Assoc.* **58** 13–30.

[20] Jerrum, M. (1992) Large cliques elude the metropolis process. *Random Struct. Alg.* **3** 347–359.

[21] Juels, A. and Peinado, M. (2000) Hiding cliques for cryptographic security. *Des. Codes Cryptography* **20** 269–280.

[22] Karp, R. M. (1972) Reducibility among combinatorial problems. In *Complexity of Computer Computations* (R. E. Miller and J. W. Thatcher, eds), Plenum, pp. 85–103.

[23] Krivelevich, M. and Vilenchik, D. (2006) Solving random satisfiable 3CNF formulas in expected polynomial time. In *SODA*, ACM, pp. 454–463.

[24] Kučera, L. (1991) A generalized encryption scheme based on random graphs. In *Graph-Theoretic Concepts in Computer Science* (G. Schmidt and R. Berghammer, eds), Vol. 570 of *Lecture Notes in Computer Science*, Springer, pp. 180–186.

[25] Kučera, L. (1995) Expected complexity of graph partitioning problems. *Discrete Applied Math.* **57** 193–212.

[26] McSherry, F. (2001) Spectral partitioning of random graphs. In *FOCS*, IEEE Computer Society, pp. 529–537.