William Yang
william_yang@berkeley.edu
3031801969

I was able to attend all of the meetings, except for the one on an SDP algorithm for recovering communities in a stochastic block model graph. I primarily studied the papers and notes relevant for my own presentations, but skimmed over some of the relevant material for some of the other presentations as well. I presented on the max clique in a random graph, specifically regarding the size of the max clique for a graph $G \sim \mathcal{G}_{n,\frac{1}{2}}$, and the derivations of the upper and lower bounds on the size of the max clique, using Markov's inequality and the second moment method, respectively. I also presented on the AKS spectral algorithm for recovering a planted clique, specifically on the correctness of the algorithm in the rounding procedures done to recover the clique.

I've attached following the notes for the presentations I presented on. We all collaborated equally on all the presentations. An implementation of the AKS spectral algorithm for recovering a planted clique from a $\mathcal{G}_{n,\frac{1}{2}}$ was also coded up here by Wilson.

Beyond Worst Case Analysis **Fall 2018**

Lecture 2: Largest Clique in a Random Graph

Scribe: William Yang, Jiazheng Zhao, James Hulett, Antares Chen 9/17/2018

2.1 Introduction

Worst-case analysis predicates on analyzing problems on classes of inputs that trigger "worst" case behavior. However, this analytical regime is sometimes too pessimistic and fails to capture what what usually happens when solving the problem on many inputs. Consider these examples:

- Quicksort and merge sort both require $O(n^2)$ time to sort a list of n elements in the worst-case, but in practice quicksort often out performs merge sort in arbitrary sorting tasks.
- Most clustering tasks that arise in topics such as unsupervised learning are formulated as NP-Hard optimization problems. However, there are a number of polynomial time algorithms that can still find a meaningful clustering if the pattern exists.
- The simplex algorithm for solving linear programs runs in worst-case exponential time, but in practice it rarely requires that long to solve an instance.

This reading group discusses analyzing algorithms beyond worst-case analysis, but to do so first requires us to define a model for inputs we wish to analyze the problem with. Over the past 30 years, there have been many models proposed and analyzed, varying widely in expressiveness and complexity. As a first step into this domain, we will look at the Erdős-Rényi model for generating random graphs. We will demonstrate that the size of the largest clique in G is with high probability $(2 \pm o(1))\log(n)$ and then show a simple greedy algorithm that will almost always recover $\log(n)$ sized clique.

2.2 The Problem with Max-Clique

Consider the max-clique problem. Given a graph G, we are interested in finding the largest subset of vertices in G such that they form a clique. This problem is well known to be NP-Hard, but it is also known to be extremely difficult to approximate. In fact, a theorem of Zuckerman [1] states

Theorem 2.1. *Let n denote the number of vertices in G, then for any constant $\epsilon > 0$, there does not exist an $O(n^{1-\epsilon})$-approximation algorithm for max-clique unless* P = NP.

To put this into context, a trivial way to get an n-approximation for max-clique is to output a single vertex. This theorem states that doing any better than this would imply P = NP! Things certainly look bleak in the land of worst-case analysis, but perhaps this is because worst-case analysis is too pessimistic. What happens when we sample random inputs for the max-clique problem? Enter the Erdős-Rényi graph.

2.3 Introducing the Erdős-Rényi Model

The Erdős-Rényi model, denoted as $\mathcal{G}_{n,p}$, defines a distribution on graphs constructed via the following procedure:

The Erdős-Rényi Model
Given n, p where $0 \leq p \leq 1$, a graph $G = (V, E)$ sampled from $\mathcal{G}_{n,p}$ is constructed via the following:

1. Fix n vertices $V = \{1, \ldots, n\}$
2. For any $i, j \in V$ where $i \neq j$, add the edge (i, j) to E with probability p independently at random.

One thing to note about this model is that it only generates simple graphs that do not contain self loops. Since the edges are sampled independently at random, the chance of sampling a specific $G \sim \mathcal{G}_{n,p}$ is given by

$$\Pr[G] = p^{|E(G)|}(1-p)^{\binom{n}{2} - |E(G)|}$$

where $E(G)$ denotes the edges of G. When $p = \frac{1}{2}$, the distribution $\mathcal{G}_{n,1/2}$ defines a uniform distribution over all simple graphs. There are many interesting properties that arise in graphs sampled from this model (see CS271 if interested), but our goal is to find the largest clique in G and so we will be interested in the size of G's largest clique. The calculations that we present in these notes can be generalized for any probability p, so for simplicity, we will restrict to using $\mathcal{G}_{n,1/2}$.

2.3.1 Largest Clique in $\mathcal{G}_{n,1/2}$

Let's first determine the size of the largest clique in $G \sim G_{n,1/2}$. The graph G is randomly constructed, so our statement of the largest clique size will have to be probabilistic in nature. One way to characterize this is as follows. Let X_k be a random variable denoting the number of k-cliques in G. If we can show that if

$$k_0(n) \leq k \leq k_1(n)$$

where $k_0(n)$ and $k_1(n)$ are two values dependent on n that are really close to each other (think an additive constant away), we have that

$$\Pr[X_{k_0(n)} > 0] \to 1 \text{ as } n \to \infty$$
$$\Pr[X_{k_1(n)} > 0] \to 0 \text{ as } n \to \infty$$

then we will know that G almost always has a clique of size about k as n becomes large. We will now show the following theorem

Theorem 2.2. *Given $G \sim \mathcal{G}_{n,1/2}$, then the largest clique of G has size $2(1 \pm o(1)) \log_2(n)$ with high probability. Specifically with $k_0(n) = 2(1 + o(1)) \log(n)$ and $k_1(n) = 2(1 - o(1)) \log(n)$, we have*

$$\Pr[X_{k_0(n)} > 0] \to 1 \text{ as } n \to \infty$$
$$\Pr[X_{k_1(n)} > 0] \to 0 \text{ as } n \to \infty$$

We will prove this result in three steps. First, we will calculate the expected number of k-cliques in $G \sim \mathcal{G}_{n,1/2}$. We will then prove the upper-bound statement for the largest clique of G having size at most $2(1+o(1))\log(n)$ w.h.p. using Markov's inequality. Finally, we demonstrate the lower-bound statement that the largest clique has size at least $2(1-o(1))\log n$ w.h.p. using what is sometimes called the second-moment method.

2.3.1.1 Number of k-Cliques in $\mathcal{G}_{n,1/2}$

Let's first quantify how many k-cliques G can have on expectation.

Claim 2.3. *Suppose* $G \sim \mathcal{G}_{n,1/2}$, *then* $\mathbb{E}[X_k] = \binom{n}{k} 2^{-\binom{k}{2}}$.

Proof. For $S \subseteq V$ the vertices of G, let Y_S be the indicator random variable for if G contains a clique on all vertices in S. The number of k-cliques is then

$$X_k = \sum_{S \subseteq V : |S| = k} Y_S$$

which by linearity of expectations is the following.

$$\mathbb{E}[X_k] = \sum_{S \subseteq V : |S| = k} \mathbb{E}[Y_S] = \sum_{S \subseteq V : |S| = k} \Pr[Y_S = 1]$$

For $|S| = k$, the chance that every edge is added to G between every pair of distinct vertices in S is given by

$$\Pr[Y_S = 1] = \frac{1}{2^{\binom{k}{2}}}$$

Finally, note that there are $\binom{n}{k}$ ways to choose k vertices from n. This means that

$$\mathbb{E}[X_k] = \sum_{S \subseteq V : |S| = k} \Pr[Y_S = 1] = \sum_{S \subseteq V : |S| = k} 2^{-\binom{k}{2}} = \binom{n}{k} 2^{-\binom{k}{2}} \qquad \square$$

2.3.1.2 The Upper-bound

Now that we know the expected number of k-cliques in G, we can calculate the probability that $\mathbb{E}[X_k] > 0$. First observe that

$$\mathbb{E}[X_k] = \binom{n}{k} 2^{-\binom{k}{2}} \leq n^k 2^{-\frac{k(k-1)}{2}} = 2^{k\log(n) - \frac{k(k-1)}{2}} = 2^{\frac{k}{2}(2\log n - k + 1)}$$

This means that $\mathbb{E}[X_{k_0(n)}]$ for $k_0(n) = 2\log(n) + 2$ is given by the following.

$$\mathbb{E}[X_{k_0(n)}] \leq 2^{-(\log(n)+1)} = n^{-\Omega(1)}$$

Because $X_{k_0(n)}$ is an integer, we have by Markov's inequality

$$\Pr[X_{k_0(n)} > 0] = 1 - \Pr[X_{k_0(n)} \leq 1] \geq 1 - n^{-\Omega(1)}$$

Indeed as $n \to \infty$, we have that $\Pr[X_{k_0(n)} > 0] \to 1$ satisfying the first part of theorem 2.2.

2.3.1.3 The Lower-bound

Markov's inequality is usually sufficient to demonstrate an upper-bound, but to prove the lower-bound requires us to control the variance of the random variable we are interested in. For example, one might be inclined to demonstrate that $\mathbb{E}[X_{k_1(n)}] \to 0$ in order to prove $\Pr[X_{k_1(n)} > 0] \to 0$ as $n \to \infty$. While limit is accurate, it is not sufficient to show $\Pr[X_{k_1(n)} > 0] \to 0$ because the variance of $X_{k_1(n)}$ could so large such that for any n, there is some amount of probability mass greater than 0.

Critically, the second moment method uses Chebyshev's inequality to show the probability tends toward 0:

$$\Pr[|X - \mathbb{E}[X]| \geq \mathbb{E}[X]] \leq \frac{\mathbf{Var}[X]}{\mathbb{E}[X]^2}$$

The form that will be most helpful for us is the following.

$$\Pr[X \leq 0] \leq \Pr[X \leq 0 \text{ or } X \geq 2\mathbb{E}[X]] = \Pr[|X - \mathbb{E}[X]| \geq \mathbb{E}[X]] \leq \frac{\mathbf{Var}[X]}{\mathbb{E}[X]^2}$$

Thus computing $\Pr[X_{k_1(n)} > 0]$ will require us to calculate the right most ratio. Finally, before we proceed with the proof of the lower-bound, observe if $Y = Y_1 + \ldots + Y_n$ is a sum of non-negative random variables

$$\mathbf{Var}[Y] = \sum_{i=1}^{n} \mathbf{Var}[Y_i] + \sum_{i \neq j} \mathbf{Cov}[Y_i, Y_j] \leq \sum_{i=1}^{n} \mathbb{E}[Y_i^2] + \sum_{i \neq j} \mathbb{E}[Y_i Y_j]$$

Let's now show that $\Pr[X_{k_1(n)} > 0] \to 0$ by first calculating the variance of $X_{k_1(n)}$. Again let Y_S be the indicator random variable for the case $S \subseteq V$ forms a clique and fix $|S| = k_1(n)$. Now observe that if $S \cap T = \emptyset$ then Y_S and Y_T are independent meaning $\mathbf{Cov}[Y_S, Y_T] = 0$. Let $S \sim T$ denote S and T are not independent (i.e. when $S \neq T$ and $|S \cap T| \geq 2$ as this is the case where S and T share an edge). The variance of $X_{k_1(n)}$ is

$$\begin{aligned}
\mathbf{Var}[X_{k_1(n)}] &\leq \sum_{S \subseteq V}^{n} \mathbb{E}[Y_S^2] + \sum_{S \sim T} \mathbb{E}[Y_S Y_T] \\
&= \sum_{S \subseteq V}^{n} \mathbb{E}[Y_S] + \sum_{S \sim T} \mathbb{E}[Y_S Y_T] \\
&= \mathbb{E}[X_{k_1(n)}] + \sum_{S \sim T} \mathbb{E}[Y_S Y_T]
\end{aligned}$$

with the second equality following as Y_S is a 0-1 random variable. But notice that $\mathbb{E}[Y_S Y_T] = 1$ if and only if $Y_S = Y_T = 1$. We have

$$\begin{aligned}
\mathbf{Var}[X_{k_1(n)}] &\leq \mathbb{E}[X_{k_1(n)}] + \sum_{S \sim T} \mathbb{E}[Y_S Y_T] \\
&= \mathbb{E}[X_{k_1(n)}] + \sum_{S \sim T} \Pr[Y_S = 1 \text{ and } Y_T = 1] \\
&= \mathbb{E}[X_{k_1(n)}] + \sum_{S \sim T} \Pr[Y_S = 1] \cdot \Pr[Y_T = 1 \mid Y_S = 1] \\
&= \mathbb{E}[X_{k_1(n)}] + \sum_{S \in V} \Pr[Y_S = 1] \cdot \sum_{T : S \sim T} \Pr[Y_T = 1 \mid Y_S = 1]
\end{aligned}$$

Now look at the right-most sum. It really does not matter where we choose S to condition T on because the edges are all sampled independently at random! By symmetry

$$\Pr[Y_T = 1 \mid Y_S = 1] = \Pr[Y_T = 1 \mid Y_{S_0} = 1]$$

for a fixed S_0 where $|S_0| = k_1(n)$, thus the variance becomes

$$\begin{aligned}\mathbf{Var}[X_{k_1(n)}] &\leq \mathbb{E}[X_{k_1(n)}] + \sum_{S \in V} \Pr[Y_S = 1] \cdot \sum_{T : S \sim T} \Pr[Y_T = 1 \mid Y_S = 1] \\ &= \mathbb{E}[X_{k_1(n)}] + \left(\sum_{S \in V} \Pr[Y_S = 1]\right) \cdot \left(\sum_{T : S_0 \sim T} \Pr[Y_T = 1 \mid Y_{S_0} = 1]\right) \\ &= \mathbb{E}[X_{k_1(n)}] + \mathbb{E}[X_{k_1(n)}] \sum_{T : S_0 \sim T} \Pr[Y_T = 1 \mid Y_{S_0} = 1]\end{aligned}$$

Now what is the conditional probability on the right? Given $i = |T \cap S_0|$, we have that

$$\Pr[Y_T = 1 \mid Y_{S_0} = 1] = \binom{k_1}{i}\binom{n-k_1}{k_1-i} 2^{-(\binom{k_1}{2} - \binom{i}{2})}$$

Here $\binom{k_1}{i}$ is the number of ways T can choose vertices to share with S_0, factor $\binom{n-k_1}{k_1-i}$ counts all the other ways that T can choose its vertices, and $2^{-(\binom{k_1}{2} - \binom{i}{2})}$ is the probability that edges in T but not in S_0 are added to G. This means that

$$\sum_{T : S_0 \sim T} \Pr[Y_T = 1 \mid Y_{S_0} = 1] = \sum_{i=2}^{k_1(n)} \binom{k_1}{i}\binom{n-k_1}{k_1-i} 2^{-(\binom{k_1}{2} - \binom{i}{2})}$$

Our variance is thus

$$\mathbf{Var}[X_{k_1(n)}] \leq \mathbb{E}[X_{k_1(n)}] + \mathbb{E}[X_{k_1(n)}] \sum_{i=2}^{k_1(n)} \binom{k_1}{i}\binom{n-k_1}{k_1-i} 2^{-(\binom{k_1}{2} - \binom{i}{2})}$$

Using the form of Chebyshev's inequality above

$$\Pr[X_{k_1(n)} \leq 0] \leq \frac{1}{\mathbb{E}[X_{k_1(n)}]} + \frac{\sum_{i=2}^{k_1(n)} \binom{k_1}{i}\binom{n-k_1}{k_1-i} 2^{-(\binom{k_1}{2} - \binom{i}{2})}}{\mathbb{E}[X_{k_1(n)}]}$$

Because $\mathbb{E}[X_{k_1(n)}] = \binom{n}{k_1} 2^{-\binom{k_1}{2}}$, the left summand tends toward 0 as $n \to \infty$. We need only show that the right summand does the same! Observe

$$\begin{aligned}\frac{\sum_{i=2}^{k_1(n)} \binom{k_1}{i}\binom{n-k_1}{k_1-i} 2^{-(\binom{k_1}{2} - \binom{i}{2})}}{\mathbb{E}[X_{k_1(n)}]} &= \frac{\sum_{i=2}^{k_1(n)} \binom{k_1}{i}\binom{n-k_1}{k_1-i} 2^{-(\binom{k_1}{2} - \binom{i}{2})}}{\binom{n}{k_1} 2^{-\binom{k_1}{2}}} \\ &= \frac{\sum_{i=2}^{k_1(n)} \binom{k_1}{i}\binom{n-k_1}{k_1-i} 2^{\binom{i}{2}}}{\binom{n}{k_1}} \\ &\leq \frac{k_1 \cdot \max_{2 \leq i \leq k_1} \binom{k_1}{i}\binom{n-k_1}{k_1-i} 2^{\binom{i}{2}}}{\binom{n}{k_1}}\end{aligned}$$

The numerator of the above is maximized at $i = 2$ (the proof is delegated the appendix) thus

$$\begin{aligned}
\frac{k_1 \cdot \max_{2\le i\le k_1} \binom{k_1}{i}\binom{n-k_1}{k_1-i}2^{\binom{i}{2}}}{\binom{n}{k_1}} &\le \frac{k_1 \cdot \binom{k_1}{2}\binom{n-k_1}{k_1-2}2^{\binom{2}{2}}}{\binom{n}{k_1}} \\
&\le k_1^3(k_1-1)^2\left(\frac{n-k_1}{n}\right)\cdots\left(\frac{n-2k_1+3}{n-k_1+3}\right)\left(\frac{1}{n-k_1+2}\right)\left(\frac{1}{n-k_1+1}\right) \\
&\le \frac{k_1^5}{n-k_1+1}
\end{aligned}$$

With $k_1 = 2(1+o(1))\log(n)$, we have that $\lim_{n\to\infty} \frac{\mathbf{Var}[X_{k_1(n)}]}{\mathbb{E}[X_{k_1(n)}]}$. This means that $\Pr[X_{k_1(n)} > 0] \to 0$ as required! This completes the proof that with high probability, the largest clique in $G \sim \mathcal{G}_{n,1/2}$ is $2(1 \pm o(1))\log(n)$. We will now use this fact to construct an algorithm that recovers a clique close to this size.

2.4 Finding Cliques in an Erdős-Rényi Graph

When G is constructed randomly, can we find a clique in polynomial time larger than that guaranteed by theorem 2.1? The fact that the largest clique in $G \sim \mathcal{G}_{n,1/2}$ is close to $2\log(n)$ allows us to say yes, and in fact, a simple greedy algorithm is all we need!

Greedy Algorithm
Given $G \sim \mathcal{G}_{n,1/2}$ do the following:

1. Initialize $S = \{v\}$ where v is an arbitrary vertex of G.
2. While there is still a vertex $i \in V$ connected to every $v \in S$ by an edge, add i to S.
3. Return S.

For the purposes of our analysis, assume that we do not sample an edge until one of its endpoints is added to S (or equivalently that when we add a vertex to S, all the edges incident to it are "revealed" to us); this will make it easier to talk about the probability of an event occurring over the random choices of the graph.

All this algorithm is doing is maintaining a clique S at every iteration of the algorithm thus the iteration that this algorithm terminates determines the size of the clique returned. When does this algorithm terminate? At the start of the algorithm, there are a total of n vertices that could eventually appear in the clique. However, each time we add a vertex v to S, we would expect that about half of the remaining vertices will end up not having an edge to v, so the pool of vertices we can choose from should get cut approximately in half each time we add a vertex to S. Intuitively, we can only add about $\log(n)$ vertices to S before our original pool of n vertices gets cut down to a size of 1. Thus, once the algorithm adds that last vertex to S, we are done.

Formalizing this intuition, we will show that, with high probability, this greedy algorithm returns a clique of size $\log(n) + \log\log(n)$. Notice that this is significantly better than what theorem 2.1 admits. Whereas in the worst-case, we may only find a clique that is roughly a $\frac{1}{n}$ factor the size of the max-clique's actual size, choosing G to be sampled randomly from $\mathcal{G}_{n,1/2}$ allows us to recover a clique that is roughly $\frac{1}{2}$ the size of the expected max size!

2.4.1 Upper-bounding Number of Iterations

We begin by demonstrating the algorithm will halt before step $\log(n) + \log\log(n)$ with high probability. We define R_k to be the number of vertices remaining after k additions to S; that is R_k is the number of vertices that have not been added to S but have edges to every vertex in S and so have the possibility of being added in the next round. We have that

$$\mathbb{E}[R_k|R_{k-1}] = \max\left(\frac{R_{k-1}-1}{2}, 0\right)$$

The first term in the maximum comes into play if $R_{k-1} > 1$, meaning that we lose one vertex from the previous step to being added to S, while in expectation half of the remainder still have an edge to everything in S; the second term comes into play when $R_{k-1} = 0$. We can simplify this to say that

$$\mathbb{E}[R_k|R_{k-1}] \leq \frac{R_{k-1}}{2}$$

Applying that $R_0 := n$ and repeatedly iterating expectations, we get that

$$\mathbb{E}[R_k] \leq \frac{n}{2^k}$$

We can now plug in $k = \log(n) + \log\log(n)$ to get that

$$\mathbb{E}[R_{\log(n)+\log\log(n)}] \leq \frac{n}{2^{\log(n)+\log\log(n)}} = \frac{n}{n\log(n)} = \frac{1}{\log(n)}$$

Applying a simple Markov bound, this gives us

$$\Pr(R_{\log(n)+\log\log(n)} \geq 1) \leq \frac{1}{\log(n)}$$

But saying that at least one vertex survived through $\log(n) + \log\log(n)$ rounds is exactly the same as saying that the algorithm has not yet terminated after that many additions to S. Hence, we get that with probability at least $1 - \frac{1}{\log(n)}$, the algorithm will have terminated by round $\log(n) + \log\log(n)$.

2.4.2 Lower-bounding Number of Iterations

We know that the algorithm will likely terminate in at most $\log(n) + \log\log(n)$ and now want to say the algorithm will likely terminate in at least $\log(n) - \log\log(n)$ so that with high probability it will output a clique of size roughly $\log(n)$. Say that a vertex "fails" if it either gets added to S or if it does not have an edge to some other vertex that got added to S. Notice that the algorithm terminates once all vertices have failed. Let F_k be the event that all vertices have failed by round k. If F_k happens, at most k of the vertices failed because they were added to S. This means at least $n-k$ of them failed because they did not have an edge to some vertex in S. The probability of a vertex failing in this latter way is just $(1-2^{-k})$, so

$$\Pr(F_k) \leq (1-2^{-k})^{n-k}$$

Since we will use $k = \log(n) - \log\log(n) \ll n$, we can safely replace the $n-k$ with a $\frac{n}{2}$ while still maintaining the bound. We now apply the "computer scientist's favorite inequality" ($1 - x \leq e^{-x}$) to rewrite this as

$$\Pr(F_k) \leq e^{-2^{-k}(n/2)}$$

Plugging in $k = \log(n) - \log\log(n)$ to the exponent, we get

$$\begin{aligned} -2^{-k}\binom{n}{2} &= -2^{-\log(n)+\log(\log(n)}\binom{n}{2} \\ &= -\left(\frac{\log(n)}{n}\right)\binom{n}{2} \\ &= -\frac{\log(n)}{2} \end{aligned}$$

Hence,

$$\Pr(F_{\log(n)-\log\log(n)}) \le e^{-\log(n)/2} = n^{-\Theta(1)}$$

The probability that our algorithm terminates in less than $\log(n) - \log\log(n)$ steps goes to zero as $n \to \infty$, so we can say that our algorithm finds a clique of size at least $\log(n) - \log\log(n)$ with high probability.

2.4.3 Some Refinements

Previously, we treated the algorithm as if it kept a list of all the vertices that could still be added to S, then chose an arbitrary one to add to S at each step. However, we can make the algorithm much simpler by simply fixing an arbitrary ordering of the vertices, then iterating through them and checking if each one can safely be added to S. In order to find the expected runtime of the algorithm given this refinement, we have the following claim.

Claim 2.4. *When considering whether or not we can add a vertex v to S, the expected number of edges we have to check for is at most 2.*

Proof. If v is the first vertex we consider, we know we can always add it to S (as there are no vertices there for it to have a problem with), so we don't have to look at any edges. Otherwise, suppose that there are already $k > 0$ vertices in S when we consider v. We always have to check to see if v is connected to the first vertex in S. However, if it is not, we don't have to check if v is connected to the second vertex; thus, we only need to check this second connection with probability $\frac{1}{2}$. Similarly, we only have to check the third connection with probability $\frac{1}{4}$, and so forth. Thus, the expected number of connections we have to check is

$$\sum_{i=1}^{k} \frac{1}{2^{i-1}} \le \sum_{i=0}^{\infty} \frac{1}{2^i} = 2 \qquad \square$$

This tells us that in expectation, we have to check at most $2n$ connections between vertices during the run of our algorithm. Hence, the expected runtime is just $O(n)$.

References

[1] Zuckerman, D. (2006, May). Linear degree extractors and the inapproximability of max clique and chromatic number. In *Proceedings of the thirty-eighth annual ACM symposium on Theory of computing* (pp. 681-690). ACM.

A Additional Proofs

A.1 Bounding the Binomial Coefficient

An inequality that we have used throughout these set of notes to bound the binomial coefficient is given by

Claim 2.5. $\left(\frac{n}{k}\right)^k \le \binom{n}{k} \le n^k$

Proof. First the lower-bound. Observe

$$\binom{n}{k} = \frac{n!}{k!(n-k)!} = \frac{n(n-1)\dots(n-k+1)}{k!} = \frac{n}{k} \cdot \frac{n-1}{k-1} \cdot \dots \cdot \frac{n-k+1}{1} \ge \left(\frac{n}{k}\right)^k$$

For the upper-bound observe

$$\binom{n}{k} = \frac{n!}{k!(n-k)!} \le n(n-1)\dots(n-k+1) \le n^k$$

as required. $\square$

A.2 Maximizing the Variance Term

We use the fact that $\max_{2\le i\le k_1} \binom{k_1}{i}\binom{n-k_1}{k_1-i}2^{\binom{i}{2}}$ is maximized when $i = 2$. The following demonstrates this fact.

Claim 2.6. $\operatorname{argmax}_{2\le i\le k_1} \binom{k_1}{i}\binom{n-k_1}{k_1-i}2^{\binom{i}{2}} = 2$

Proof. Observe that $\binom{k}{i}\binom{n-k}{k-i}2^{\binom{i}{2}}$ is a decreasing sequence for $2 \le i \le t$. This can be seen as follows:

$$\frac{a_i}{a_{i-1}} = \frac{\binom{k}{i}\binom{n-k}{k-i}2^{\binom{i}{2}}}{\binom{k}{i-1}\binom{n-k}{k-(i-1)}2^{\binom{i-1}{2}}} = \frac{(k-i+1)^2}{i(n-2k+i)}2^{i-1} < 1$$

$\square$

Beyond Worst Case Analysis **Fall 2018**

Lecture 3: Planted Clique in Random Graph

Scribe: Jiazheng Zhao, William Yang Sep 24 2018

3.1 Introduction

In this note, we introduce the planted clique problem, where instead of finding a large clique in a random graph, we are tasked with finding a large clique that is "planted" within a random graph. We discuss the AKS spectral algorithm for finding planted cliques of size $k \geq \Omega(\sqrt{n})$ and demonstrate a flavor of algorithmic analysis that apply to a broad class of spectral algorithms.

3.2 The Planted Clique Problem

The planted clique problem can be formulated as a search problem over random graphs selected from the *planted clique distribution* which is defined as follows. Given n, k as parameters, the planted clique distribution is a distribution over simple graphs constructed by the process

1. Sample $G' = (V, E)$ from the Erdős-Rényi distribution with $p = \frac{1}{2}$ (i.e. $G' \sim \mathcal{G}_{n,1/2}$).
2. Choose $S \subseteq V$ a subset of k vertices uniformly at random.
3. Construct G by planting a clique on S (i.e. connect all edges among vertices in S)

With G sampled from this process, the *planted clique problem* is to then recover S from G without knowledge of where the clique is planted. We should expect this problem to be easy to solve if the cliques in G' are much smaller than k. Last week, we demonstrated that w.h.p. the size of the maximum clique in $G' \sim \mathcal{G}_{n,1/2}$ is at most $(2 \pm o(1)) \log n$. It is information theoretically impossible to recover S if $k \leq 2 \log n$. However, the max-clique size of G' also implies a straight-forward quasi-polynomial time algorithm: simply brute force search for a clique over subsets of $3 \log n$ vertices.

This $n^{O(\log(n))}$-time algorithm will likely succeed as w.h.p. if G' contains a $3 \log n$ sized clique, it will be the only clique of that size and thus must be a subset of the planted k-clique. A natural question is to ask for what k is it possible to recover S in polynomial time? The AKS (Alon, Krevich and Sudakov) spectral algorithm we present recovers S if $k \geq \Omega(\sqrt{n})$. Many people believe this to be the best achievable in polynomial time. Indeed, the planted clique conjecture states for any $\epsilon \geq \Omega(1)$, it is NP-Hard to recover S from G where $k \leq n^{1/2-\epsilon}$. There have additionally been a number of hardness results using this conjecture in computing nash equilibria, property testing, and solving online learning problems.

3.2.1 Algorithm

The AKS spectral algorithm utilizes the spectrum of A, the adjacency matrix of sampled graph G, to recover S. Since A is a random matrix, a natural starting point for building an algorithm is to analyze the spectrum of $\mathbb{E}[A]$. Without loss of generality, let's relabel the vertices in G such that the planted clique comes first (i.e. vertices $1, \dots, k$ contain the planted clique). The matrix $\mathbb{E}[A]$ looks like

$$\mathbb{E}[A] = \left(\begin{array}{c:c} 1 & 1/2 \\ \hdashline 1/2 & 1/2 \end{array}\right) - D$$

where $D_{ii} = 1$ if $i \in S$ otherwise $D_{ii} = \frac{1}{2}$. Written another way, $\mathbb{E}[A]$ is

$$\mathbb{E}[A] = \frac{1}{2}J + \frac{1}{2}\mathbf{1}_S\mathbf{1}_S^T - D$$

where J is the all-ones matrix and $\mathbf{1}_S$ is the 0-1 indicator vector on S. Why is this the expectation?

- First ignore the clique we planted. Since A is the adjacency matrix of $G \sim \mathcal{G}_{n,\frac{1}{2}}$, each edge exists with probability $\frac{1}{2}$, so each entry of A is 1 with probability 0.5 and 0 with probability 0.5. Therefore $\mathbb{E}[A_{ij}] = \frac{1}{2}$. We use a matrix where all entries are $\frac{1}{2}$ to represent that.
- S is the set of vertices that forms the clique. Because we deterministically plant an edge between all vertices in S, $A_{ij} = 1$ where $i, j \in S$. Note that the (i, j)-th entry of $\mathbf{1}_S\mathbf{1}_S^T$ is exactly one if $i, j \in S$ and 0 elsewhere. Therefore adding $\frac{1}{2}\mathbf{1}_S\mathbf{1}_S^T$ to the all $\frac{1}{2}$ will adjust the expected value of entries that represent edges in clique.

This says something rather useful! The diagonal matrix $-D$ can be split into a sum of rank-one matrices weighted by the elements of $-D$. By the spectral theorem, the top eigenvector of $\mathbb{E}[A] - \frac{1}{2}J$ is $\mathbf{1}_S$ since $-D$ only contains negative entries. The vector $\mathbf{1}_S$ is the indicator vector for the clique we wish to recover. For G sampled from the planted clique distribution, if the top eigenvector of $A - \frac{1}{2}J$ is close to that of $\mathbb{E}[A] - \frac{1}{2}J$, then we can hope to recover the planted clique S via some rounding procedure with high probability. This is precisely the intuition behind the AKS algorithm.

AKS Spectral Algorithm for Planted Clique
Given G sampled from the planted clique distribution, do the following:

1. Let $M = A - \frac{1}{2}J$ where A is the adjacency matrix of G.
2. Compute $\mathbf{x}$ the eigenvector corresponding to the largest eigenvalue of M.
3. Let I be the set of k vertices v corresponding to the largest values of $|\mathbf{x}_v|$.
4. Return $C = \{v \in V : v \text{ has at least } \frac{3}{4}k \text{ neighbors in } I\}$

The algorithm can be divided into two parts: steps (1) and (2) obtain an eigenvector from sampled G; steps (3) and (4) round the eigenvector to extract the planted clique. We note that spectral algorithms for "planted" problems like planted clique have a similar flavor to AKS. Supposing one can show on expectation, that a witness for the planted structure manifests itself as (perhaps) a large eigenvector and if the behavior of sampled instances concentrates around the expectation, then with an appropriate (but sometimes non-trivial) rounding procedure, one can (at least) approximately recover the planted solution.

3.3 Analysis

Since the algorithm splits into two parts, the analysis of this algorithm is divided accordingly: first, we focus on the first two lines of the algorithm and show that the eigenvector $\mathbf{x}$ is very close to $\mathbf{1}_S$, the top eigenvector of $\mathbb{E}[A] - \frac{1}{2}J$ and planted solution, with high probability. Second, we argue that the way we recover the solution from the eigenvector selects the planted clique with high probability.

3.3.1 Analysis of Retrieved Eigenvector

If we can get that $\mathbf{x}$ is close to $\mathbf{1}_S$, then we have good reason to believe that rounding $\mathbf{x}$ will yield the planted solution. To show this, we will begin by demonstrating that M and $\frac{1}{2}\mathbf{1}_S\mathbf{1}_S^\top$ are "close" under the appropriate notion of distance. Once we show that these two matrices are "close", we can then show that their top eigenvectors (i.e. $\mathbf{x}$ and $\mathbf{1}_S$) are also "close".

How do we quantify "close" for matrices? Since we have only worked with real, symmetric matrices so far, we can use the spectral norm to measure distance between the two matrices. Let $\lambda_1, \ldots, \lambda_n$ denote the eigenvalues of any real, symmetric matrix M'. The spectral norm of M' is defined as

$$\|M'\| = \max_i |\lambda_i|$$

It will be helpful to know that A and $\mathbb{E}[A]$ are "close" in the spectral norm – their largest eigenvalues are not too far away from each other. This is what the following lemma states.

Lemma 3.1. *Fix a set S of size k. With high probability the following is true:*

$$\|A - \mathbb{E}[A]\| \leq (1 + o(1)) \cdot \sqrt{n}$$

where $\|\cdot\|$ is the spectral norm.

There are a number of ways to show this, so we omit the proof for a later set of notes to keep this exposition simple. We can apply this lemma and bound the "closeness" of M and $\frac{1}{2}\mathbf{1}_S\mathbf{1}_S^T$:

$$\left\| M - \frac{1}{2}\mathbf{1}_S\mathbf{1}_S^T \right\| = \left\| A - \frac{1}{2}J - \frac{1}{2}\mathbf{1}_S\mathbf{1}_S^T \right\| \leq \|A - \mathbb{E}[A] + D\| \leq \|A - \mathbb{E}[A]\| + \|D\| \leq (1 + o(1)) \cdot \sqrt{n}$$

The last two inequalities follow by triangle inequality and lemma 3.1 respectively. We have that the top eigenvalues of M and $\frac{1}{2}\mathbf{1}_S\mathbf{1}_S^T$ are close, but does that translate to their top eigenvectors? By the Davis-Kahan theorem, yes!

Theorem 3.2 (Davis and Kahan)**.** *If M is a symmetric matrix, $\mathbf{y}$ is a vector, $\mathbf{x}$ is an eigenvector of the largest eigenvalue of M, and $\hat{\mathbf{xy}}$ is the angle between $\mathbf{x}, \mathbf{y}$*

$$|\sin(\hat{\mathbf{xy}})| \leq \frac{\|M - \mathbf{y}\mathbf{y}^T\|}{\|\mathbf{y}\mathbf{y}^T\| - \|M - \mathbf{y}\mathbf{y}^T\|}$$

Again we leave the proof to a later set of notes. We quantify vector distance via $\|\cdot\|_2$, the ℓ_2-norm.

Corollary 3.3. *If $\|M - \frac{1}{2}\mathbf{1}_S\mathbf{1}_S^T\| \leq (1+o(1))\sqrt{n}$, $k > 100\sqrt{n}$, and $\mathbf{x}$ is an eigenvector corresponding to the largest eigenvalue of M scaled so that $\|\mathbf{x}\|_2^2 = k$, then*

$$\min\left\{\|\mathbf{x} - \mathbf{1}_S\|_2^2\,,\ \|-\mathbf{x} - \mathbf{1}_S\|_2^2\right\} \leq 0.002k$$

for sufficiently large n.

Proof. First we find $\|\frac{1}{2}\mathbf{1}_S\mathbf{1}_S^T\|$. Notice that the spectral norm of this rank one matrix is just the maximized Rayleigh quotient of the matrix. Since the matrix is rank one and the only eigenvector is $\mathbf{1}_S$, we can get

$$\left\|\frac{1}{2}\mathbf{1}_S\mathbf{1}_S^T\right\| = \frac{1}{2}\|\mathbf{1}_S\mathbf{1}_S^T\| = \frac{1}{2}\frac{\mathbf{1}_S^T\mathbf{1}_S\mathbf{1}_S^T\mathbf{1}_S}{\mathbf{1}_S^T\mathbf{1}_S} = \frac{1}{2}\mathbf{1}_S^T\mathbf{1}_S = \frac{k}{2}$$

Then we apply the Davis-Kahan theorem to derive the following.

$$|\sin(\mathbf{x}\hat{\mathbf{1}}_S)| \leq \frac{(1+o(1))\sqrt{n}}{\frac{k}{2} - (1+o(1))\sqrt{n}} = \frac{1}{49} + o(1)$$

Regarding ℓ_2-distance between $\mathbf{x}$ and $\mathbf{1}_S$, we have the following.

$$\|\mathbf{x} - \mathbf{1}_S\|_2^2 = \|\mathbf{x}\|_2^2 + \|\mathbf{1}_S\|_2^2 - 2\langle\mathbf{x},\mathbf{1}_S\rangle = 2k - 2\langle\mathbf{x},\mathbf{1}_S\rangle$$

Similarly

$$\|-\mathbf{x} - \mathbf{1}_S\|_2^2 = 2k + 2\langle\mathbf{x},\mathbf{1}_S\rangle$$

Therefore we have

$$\min\{\|\mathbf{x} - \mathbf{1}_S\|_2^2, \|-\mathbf{x} - \mathbf{1}_S\|_2^2\} = 2k - 2\cdot|\langle\mathbf{x},\mathbf{1}_S\rangle|$$

Combining the previous results, we have

$$|\langle\mathbf{x},\mathbf{1}_S\rangle| = \|\mathbf{x}\|_2\cdot\|\mathbf{1}_S\|_2\cdot\cos(\mathbf{x}\hat{\mathbf{1}}_S) = k\cos(\mathbf{x}\hat{\mathbf{1}}_S) = k\cdot\sqrt{1-\sin^2(\mathbf{x}\hat{\mathbf{1}}_S)} \geq k\sqrt{\frac{49^2-1}{49^2} - o(1)} > 0.999\cdot k$$

Therefore $\min\{\|\mathbf{x} - \mathbf{1}_S\|^2, \|-\mathbf{x} - \mathbf{1}_S\|^2\} \leq 2k - 2\cdot 0.999\cdot k = 0.002k$. □

We have finally shown that the eigenvector $\mathbf{x}$ is close to the solution vector $\mathbf{1}_S$. But since $\mathbf{x}$ is a real valued vector, the solution to this problem is still not clear yet. The next step is to perform some deterministic rounding to retrieve the solution from the vector $\mathbf{x}$.

3.3.2 Analysis of the Rounding Step

To analyze the second half of the algorithm, we want to show that our rounding procedures recovers S exactly by demonstrating that, with high probability, C contains all the elements of S ($S \subseteq C$) and no elements outside of S ($C \subseteq S$).

3.3.2.1 $S \subseteq C$ with high probability

The first step towards showing this is that if I is defined to be the set of k vertices v for which $|\mathbf{x}_v|$ is largest, then the sets I and S are almost the same. This results from the following lemma:

Lemma 3.4. *If* $\mathbf{x}$ *is a vector such that* $\|\mathbf{x}\|_2^2 = |S| = k$ *where* $\min\{\|\mathbf{x} - \mathbf{1}_S\|_2^2, \| - \mathbf{x} - \mathbf{1}_S\|_2^2\} \le \epsilon k$*, then*

$$|I \cap S| \ge k(1 - 4\epsilon)$$

Proof. Let's begin by defining bad vertices. Since we would like I and S to be as similar as possible, bad vertices relative to I and S should be v such that $v \in I$ but $v \notin S$ or $v \notin I$ but $v \in S$ (more concisely, $v \in I \oplus S$). Call $v \in V$ is *bad* if $v \in S$ and $|\mathbf{x}_v| \le \frac{1}{2}$ or $v \notin S$ and $|\mathbf{x}_v| > \frac{1}{2}$. Let B be the set of bad vertices.

This definition matches our intuitive notion of bad vertices, since vertices v with higher values of $|\mathbf{x}_v|$ (say, $|\mathbf{x}_v| > \frac{1}{2}$) are more likely to be contained in I (as I is the set of k vertices for which $|\mathbf{x}_v|$ is the largest) and thus are bad if $v \notin S$, while vertices with lower values of $|\mathbf{x}_v|$ (say, $|\mathbf{x}_v| \le \frac{1}{2}$) are less likely to be contained in I and are thus bad vertices if $v \in S$.

Observe that $|B| \le 4\epsilon k$ since each vertex of B contributes at least $\frac{1}{4}$ to both $\|\mathbf{x} - \mathbf{1}_S\|_2^2$ and $\| - \mathbf{x} - \mathbf{1}_S\|_2^2$. Consider a bad vertex v and its contribution to $\|\mathbf{x} - \mathbf{1}_S\|_2^2$ and $\| - \mathbf{x} - \mathbf{1}_S\|_2^2$. There are two cases:

1. Suppose $v \in S$ and $|\mathbf{x}_v| \le \frac{1}{2}$. The corresponding entry in the clique indicator vector is $\mathbf{1}_{S_v} = 1$ since $v \in S$. Vertex v would then contribute at least $\frac{1}{4}$ to both $\|\mathbf{x} - \mathbf{1}_S\|_2^2$ and $\| - \mathbf{x} - \mathbf{1}_S\|_2^2$ as both $(\mathbf{x}_v - 1)^2 \ge \frac{1}{4}$ and $(-\mathbf{x}_v - 1)^2 \ge \frac{1}{4}$ (terms in the expansion of the ℓ_2-norm).
2. Suppose $v \notin S$ and $|\mathbf{x}_v| > \frac{1}{2}$. The corresponding entry in the indicator vector of the clique is $\mathbf{1}_{S_v} = 0$. Thus v would also contribute at least $\frac{1}{4}$ to both $\|\mathbf{x} - \mathbf{1}_S\|_2^2$ and $\| - \mathbf{x} - \mathbf{1}_S\|_2^2$.

Our inequality for $|B|$ thus follows since

$$\frac{|B|}{4} \le \min\{\|\mathbf{x} - \mathbf{1}_S\|_2^2, \| - \mathbf{x} - \mathbf{1}_S\|_2^2\} \le \epsilon k$$

Concluding $|B| \le 4\epsilon k$. The next observation we make is that $|I \cap S| \ge k - |B|$. Let $t = \min_{v \in I} |\mathbf{x}_v|$ and consider the two cases:

1. If $t > \frac{1}{2}$, then any bad vertices $v \in I$ but $v \notin S$ must have $|\mathbf{x}_v| > \frac{1}{2}$ by definition of t. However, I can contain at most $|B|$ of these bad vertices, and thus must contain at least $k - |B|$ vertices from S (since $|I| = |S| = k$).
2. If $t \le \frac{1}{2}$ then any bad vertex v such that $v \notin I$ but $v \in S$ must have $|\mathbf{x}_v| \le \frac{1}{2}$ otherwise v would have been added to I. However, this must mean that at least $k - |B|$ of the remaining vertices in S are also included in I since they have $|\mathbf{x}_v| \ge t$.

In both cases we have $|I \cap S| \ge k - |B|$. By the inequality above $|I \cap S| \ge k(1 - 4\epsilon)$ as required. □

From corollary 3.3, we can apply lemma 3.4 with $\epsilon = .002$ to derive with high probability, $I \cap S$ will contain at least $k - |B| \ge k - 4\epsilon k = .992k$ vertices. Immediately, we have that $S \subseteq C$ since all vertices in S will have at least $.992k > \frac{3}{4}k$ neighbors in I as S contains a clique.

3.3.2.2 $C \subseteq S$ with high probability

The final portion of our analysis is to show $C \subseteq S$ with high probability which will require Chernoff bounds.

Theorem 3.5 (Chernoff bound for a Binomial Random Variable). *If $X_1, \ldots, X_k$ are i.i.d* Bernoulli(p) *random variables, and $X := \sum_{i=1}^k X_i$, so that $X \sim \text{Binomial}(k,p)$, then, for every $0 < \epsilon < 1$*

$$\Pr[X - \mathbb{E}[X] \geq \epsilon k] \leq e^{-2\epsilon^2 k}$$
$$\Pr[X - \mathbb{E}[X] \leq -\epsilon k] \leq e^{-2\epsilon^2 k}$$

To relate this to the situation we have at hand, note that for a vertex $v \notin S$, the number of neighbors of v that are in S is a binomial random variable $Y = \sum_{i=1}^k Y_i$ where Y_i is an indicator random variable for the corresponding edge between a vertex $i \in S$ and v (i.e. $Y_i \sim \text{Bernoulli}\left(\frac{1}{2}\right)$). With $\mathbb{E}[Y] = \frac{|S|}{2} = \frac{k}{2}$ in mind, we can apply a Chernoff bound as follows.

Corollary 3.6. *Fix a vertex $v \notin S$. With probability at least $1 - e^{-.02k}$ over the choice of G, vertex v has at most $.6k$ neighbors in S.*

Proof. Apply a Chernoff bound with $\epsilon = 0.1$. Letting $Y \sim \text{Binomial}\left(k, \frac{1}{2}\right)$ be a random variable denoting the number of neighbors of v that are in S as before, we get that

$$\Pr[Y \leq .6k] = 1 - \Pr[Y > .6k] = 1 - \Pr\left[Y - \frac{k}{2} > .1k\right] = 1 - \Pr[Y - \mathbb{E}[Y] > .1k]$$

The last line is greater than or equal to $1 - e^{-2(.1)^2 k} = 1 - e^{-.02k}$ by Chernoff's bound. □

This is for one $v \notin S$ but we want to make sure this holds for *all* $v \notin S$. Let's apply a union bound to complete our analysis.

Corollary 3.7. *With probability at least $1 - (n-k)e^{-.02k}$ over the choice of G, every vertex $v \notin S$ has at most $.6k$ neighbors in S.*

Proof. Let A denote the event that every vertex $v \notin S$ has at most $.6k$ neighbors in S. Additionally define B_i as events where $i \in V$ has more than $.6k$ neighbors in S, we have that. The negation of A is the event where there is at least one vertex $v \notin S$ with more than $.6k$ neighbors in S. This is the union of $n-k$ events B_i thus by the union bound

$$\Pr[A] = 1 - \Pr\left[\bigcup_{i \notin S} B_i\right] \geq 1 - \sum_{i \notin S} \Pr[B_i]$$

However, $\Pr[B_i] = \Pr[Y > .6k]$ from the proof of Corollary 3.6, which is at most $e^{-.02k}$ by Chernoff. Since there are $n-k$ vertices not in S, we have $\Pr[A] \geq 1 - (n-k)e^{-.02k}$ as required. □

With the results of corollary 3.6, we can now verify that $C \subseteq S$ with high probability, and thus $C = S$. Note with probability $1 - e^{-\Omega(\sqrt{n})}$ (since $k \geq \Omega(\sqrt{n})$), each vertex $v \notin S$ has at most $.6k$ neighbors in S. The results from section 3.3.2.1 implies I contains at least $.992k$ of the k vertices of S, thus I contains at most $.008k$ other vertices not in S. Thus, in the worst case, each vertex $u \notin S$ has at most $.608k$ neighbors in I. Since the algorithm sets C to be the set of vertices $v \in V$ that have at least $\frac{3}{4}k$ neighbors in I, the vertices $u \notin S$ will thus not be included in C, completing our analysis.