Notes by Alistair Sinclair/Manuel Blum/Douglas Young.

Random graphs

- How many edges are needed to make a random graph on n nodes connected?

First, we need to define precisely what we mean by a "random graph."

Experiment 1:

> start with n vertices $V = \{1, 2, \ldots, n\}$ and $E = \emptyset$
> **do** m times
>   select an edge $e = \{i, j\} \notin E$ uniformly at random
>   add e to E
> output $G = (V, E)$

Sample space: all $\binom{\binom{n}{2}}{m}$ labeled graphs G with n vertices and m edges, each equally likely

Ex: Enumerate the entire sample space for $n = 4$, $m = 3$. $\square$

Let G be a random graph (sample point) as above; we want to find the smallest value of m s.t. $\Pr[G \text{ is connected}]$ is close to 1.

Actually, it is much easier to answer a similar question for a slightly different sample space: then we'll translate the answer to the above sample space.

Experiment 2:

> start with n vertices $V = \{1, 2, \ldots, n\}$ and $E = \emptyset$
> **do** until $G = (V, E)$ is connected
>   select an edge $e = \{i, j\} \notin E$ uniformly at random
>   add e to E
> output $G = (V, E)$

Sample space: all connected graphs G obtained from the above experiment, with probabilities ???

Ex: Enumerate the entire sample space (including the probabilities) for $n = 4$. $\square$

We are interested in the random variable $X = |E|$, i.e., the number of edges in the graph G at the end of the experiment. Let's compute $\mathrm{E}(X)$.

Idea: let X_k = number of edges added to reduce number of components from k to $k - 1$.
Then $X = \sum_{k=2}^{n} X_k$ and $\mathrm{E}(X) = \sum_{k=2}^{n} \mathrm{E}(X_k)$.

What does X_k look like?

Well, $X_n = 1$ always (with probability 1) — why?
And $X_{n-1} = 1$ with probability 1 — why?

The distribution of X_{n-2} depends on the first two edges (why?); but presumably its expectation is not much bigger than 1 (again, why?)
Similarly, for $k < n-2$, the distributions of the X_k become rather complicated, but maybe we can compute an upper bound on $\mathrm{E}(X_k)$.

Claim: For all k, we have $\mathrm{E}(X_k) \leq \frac{n-1}{k-1}$.

Proof: Suppose G has exactly $k > 1$ components. Consider *any* vertex i. Our experiment is equally likely to pick any of the edges $\{i, j\}$ that is not in E. There are at most $n-1$ such edges, of course. How many of them reduce the number of components? Well, *at least* $k-1$ (why?). Therefore, the probability that any such edge reduces the number of components is at least $\frac{k-1}{n-1}$. Since this holds for every vertex i, it holds in general. But now we see that $X_k \leq Y_k$, where $Y_k = \#$ coin flips up to and including first head for a coin with $\Pr[\text{heads}] = p = \frac{k-1}{n-1}$. And by qun. 2(a) of HW2, we know that $\mathrm{E}(Y_k) = \frac{1}{p} = \frac{n-1}{k-1}$. Hence $\mathrm{E}(X_k) \leq \frac{n-1}{k-1}$. $\square$

Now we are done:

$$\mathrm{E}(X) = \sum_{k=2}^{n} \mathrm{E}(X_k) \leq (n-1)\left\{1 + \tfrac{1}{2} + \tfrac{1}{3} + \cdots + \tfrac{1}{n-1}\right\} \sim (n-1)(\ln(n-1) + \gamma).$$

So the expected number of edges at the end of Experiment 2 is at most $(n-1)(\ln(n-1)+\gamma)$ as $n \to \infty$.

Note: This is only an upper bound on $\mathrm{E}(X)$. The exact answer (which requires more effort) is $\mathrm{E}(X) \sim \frac{n}{2} \ln n$. So we are off by only a factor of 2.

Ex: In the above proof, we said that $X_k \leq Y_k$. What does this mean, in view of the fact that X_k and Y_k are not numbers but random variables (over different sample spaces)? What it means, precisely, is that $\Pr[X_k \geq z] \leq \Pr[Y_k \geq z]$ for all z. Think about this statement, and convince yourself that it holds for the X_k and Y_k in the above proof. Also, show that $X_k \leq Y_k$ implies that $\mathrm{E}(X_k) \leq \mathrm{E}(Y_k)$, as we used in the proof. $\square$

We've seen that the *expected* number of edges required to make the graph connected is at most $M(n) = (n-1)(\ln(n-1)+\gamma)$. What's the probability that we need much more than this? We can get a bound on this probability using Markov's inequality:

Theorem [Markov's Inequality]: Let X be a r.v. taking non-negative values, and let $\mu = \mathrm{E}(X)$. Then

$$\Pr[X \geq c\mu] \leq \frac{1}{c} \qquad \text{for any } c \geq 1.$$

Proof:

$$\begin{aligned}
\mu = \mathrm{E}(X) &= \sum_k \Pr[X = k] \cdot k \\
&\geq \sum_{k \geq c\mu} \Pr[X = k] \cdot k \\
&\geq c\mu \sum_{k \geq c\mu} \Pr[X = k] \\
&= c\mu \Pr[X \geq c\mu].
\end{aligned}$$

Therefore, $\Pr[X \geq c\mu] \leq \frac{1}{c}$. $\square$

Ex: The above proof isn't formally valid when $\mu = 0$, since in the last step we cancel μ. Is the theorem still true when $\mu = 0$? $\square$

Ex: Give a simple counterexample which shows that Markov's inequality is definitely <u>false</u> if we drop the assumption that X is non-negative. $\square$

Applying Markov's inequality to our r.v. X, we get

$$\Pr[X \ge cM(n)] \le \tfrac{1}{c} \qquad \text{for any } c \ge 1. \tag{$*$}$$

So, for example, if we add $10M(n)$ edges, the probability that G is connected is at least $\frac{9}{10}$.

This will help us to analyze Experiment 1. First, suppose we modify Experiment 2 slightly so that, if G becomes connected before m edges have been added, we still continue to add random edges until G has exactly m edges. View each point of the sample space of this modified Experiment 2 as $G = G' + G''$, where G' is the graph consisting of the first m edges and G'' is the remainder. (Note that G'' will be empty if G' is connected.) Then it should be clear that

$$\Pr[X \le m] = \Pr[G' \text{ is connected}].$$

What is the relationship with Experiment 1? Well, if you think about it you should see that the sample space of graphs G' is *exactly* the same as the sample space of Experiment 1 (why?). So we get

$$\Pr_1[G \text{ is connected}] = \Pr_2[G' \text{ is connected}] = \Pr_2[X \le m],$$

where $\Pr_1$ and $\Pr_2$ denote probabilities in Experiments 1 and 2 respectively.
Now, putting $m = cM(n)$ in Experiment 1, we get from $(*)$ that

$$\Pr_1[G \text{ is connected}] = \Pr_2[X \le cM(n)] \ge 1 - \tfrac{1}{c},$$

which gives a good answer to our original question about Experiment 1; i.e., a random graph with n vertices and $m = c(n-1)(\ln(n-1)+\gamma)$ edges is connected with probability at least $1 - \frac{1}{c}$.

A randomized algorithm for a graph problem

Let $G = (V, E)$ be an undirected graph. A <u>cut</u> in G is a set of edges whose removal separates G into two (or more) components.
The problem MinCut involves finding a cut in G with the minimum number of edges.

Here is a very simple randomized algorithm (due to Karger) for MinCut:

> **while** G has more than two vertices **do**
>   pick an edge $e = (u, v)$ u.a.r.
>   contract e
> output the remaining edges

The operation "contract e" means that the endpoints, u and v, of e are merged into a single vertex, retaining all their connections to other vertices. More precisely, we retain all multiple edges that are created, but eliminate all self-loops.
Since the number of vertices decreases by 1 each time, there will be exactly $n-2$ iterations, where n is the number of vertices in G. So the algorithm runs in $O(n^2)$ time. (Check this.) But does it work???

First, note that the algorithm always outputs a valid cut in G. (Why?) We need to analyze the probability that it outputs a minimum cut.

Let's focus on a particular minimum cut, which we'll call $\mathcal{C}$. We'll look at the probability that $\mathcal{C}$ survives throughout the repeated contraction process of the algorithm.
Let E_i be the event that $\mathcal{C}$ survives iteration i. We want to compute $\Pr[E_1 \wedge E_2 \wedge \ldots \wedge E_{n-2}]$. Using the fact that $\Pr[E \wedge F] = \Pr[F]\Pr[E|F]$, we can write this as

$$\Pr[\textstyle\bigwedge_{i=1}^{n-2} E_i] = \Pr[E_1] \times \Pr[E_2|E_1] \times \Pr[E_3|E_1 \wedge E_2] \times \cdots \times \Pr[E_{n-2}|\textstyle\bigwedge_{j=1}^{n-3} E_j]. \quad (\ddagger)$$

What is $\Pr[E_1]$, the probability that $\mathcal{C}$ survives the first iteration?
Well, let the number of edges in $\mathcal{C}$ be k. Then every vertex in G must have degree[1] at least k (why?). So G must have at least $\frac{kn}{2}$ edges.
Therefore, $\Pr[E_1] \geq 1 - \frac{k}{(kn/2)} = 1 - \frac{2}{n} = \frac{n-2}{n}$. (Why?)

Now let's look at $\Pr[E_2|E_1]$, the probability that $\mathcal{C}$ survives the second iteration given that it survived the first.
By the same argument as above, G must now have at least $\frac{k(n-1)}{2}$ edges.
So $\Pr[E_2|E_1] \geq 1 - \frac{k}{k(n-1)/2} = 1 - \frac{2}{n-1} = \frac{n-3}{n-1}$.

In similar fashion, we can show for each i that

$$\Pr[E_i|\textstyle\bigwedge_{j=1}^{i-1} E_j] \geq 1 - \frac{2}{n-i+1} = \frac{n-i-1}{n-i+1}.$$

Plugging this into ($\ddagger$) gives

$$\Pr[\textstyle\bigwedge_{i=1}^{n-2} E_i] \geq \frac{n-2}{n} \times \frac{n-3}{n-1} \times \frac{n-4}{n-2} \times \ldots \times \frac{2}{4} \times \frac{1}{3} = \frac{2}{n(n-1)}.$$

So, our algorithm discovers the minimum cut $\mathcal{C}$ with probability at least $\frac{2}{n^2}$.

Ex: If there were many – say, m – minimum cuts, show that this probability would improve to $\frac{2m}{n^2}$. □

The observation in this exercise isn't much use, however, since in general we can't assume that G will have more than a single minimum cut. So the best lower bound we have on the success probability of the algorithm is about $\frac{2}{n^2}$.

Disappointing?
Not really: suppose we perform t independent trials of the algorithm, and choose the smallest cut we find. What is the probability that we fail to discover $\mathcal{C}$ on all t attempts?
Clearly, this prob. is at most $(1 - \frac{2}{n^2})^t$. (Why?)

[1] The degree of a vertex is the number of neighbors it has.

So if we take $t = cn^2$, with c a constant, the prob. is at most $(1 - \frac{2}{n^2})^{cn^2} \leq \mathrm{e}^{-2c}$.

So, to make the probability that the algorithm fails as small as (say) $\mathrm{e}^{-14} \approx 10^{-6}$*, it is enough to perform only* $7n^2$ *repetitions.*

Ex: The above proof shows that G can have at most $\frac{n(n-1)}{2}$ different minimum cuts. Why? $\square$

Ex: Suppose we modify the algorithm so that, instead of choosing an edge u.a.r. and merging its endpoints, it chooses two vertices u.a.r. and merges them. Find a family of graphs G_n (where G_n has n vertices for each n) such that, when the modified algorithm is applied to G_n, the probability that it finds a minimum cut is *exponentially* small in n. How many times would you have to repeat this algorithm to have a reasonable chance of finding a minimum cut? $\square$