CS265, Fall 2023

Class 2: Agenda and Questions

1 Warm-up

There are n pigeons and n pigeon-holes; each pigeon has its own pigeon-hole. However, after a wild night the n pigeons return to a uniformly random pigeon-hole (so it could be that some holes are empty, and some have more than one pigeon). What's the expected number of empty pigeon-holes?

Group Work: Solutions

The expected number of empty holes, by linearity of expectation, is

$$\mathbb{E}\sum_{i=1}^{n}\mathbf{1}[\text{hole } i \text{ is empty}] = \sum_i \Pr[\text{hole } i \text{ is empty}].$$

The probability that any one hole is empty is $(1-1/n)^n$. (that is, with probability $1-1/n$ independently for each of n pigeons, a pigeon does not pick that hole). Thus, the expected number of empty holes is

$$n(1-1/n)^n.$$

As n gets large, this tends to n/e. (This is because $1-1/n \approx e^{-1/n}$ when n is large; you can see this by writing out the Taylor series for $e^{-1/n} = 1-1/n+1/(2n^2)-1/(6n^3)+\cdots$).

2 Welcome Back and Announcements!

- First homework is up now!

3 Questions?

Any questions from the lecture notes or short lecture videos?

4 Coupon Collecting

This will be a good chance to practice the power of linearity of expectation.

Let W be a collection of n words. For example, $W = \{$`hello`, `randomized`, `algorithms`, `penguin`, `spaceship`, …, `mushroom`$\}$. There's a button in front of you. Each time you push the button, it will say a word uniformly at random from W. If you push the button multiple times, it answers independently each time. (In particular, it's possible to get the same word twice). The question is:

> What is the expected number of times you push the button before you see all n words?

4.1 Group work

You'll answer this question by answering the following questions in your groups.

Let X_i be the time at which you see the i'th new word. For example, if you push the button six times and see

`penguin, penguin, randomized, hello, randomized, spaceship`

then $X_1 = 1$, since you saw your first new word ("`penguin`") on push 1. $X_2 = 3$, since you saw the second new word ("`randomized`") on push 3. And $X_3 = 4, X_4 = 6$.

Group Work

1. What is $\mathbb{E}X_1$? (This is not a trick question).
2. What is $\mathbb{E}(X_2 - X_1)$? That is, in expectation, how many times do you press the button, after you have seen the first word, before you see a new, second word?
3. What is $\mathbb{E}(X_3 - X_2)$?
4. For any $i = 2, 3, \ldots, n$, what is $\mathbb{E}(X_i - X_{i-1})$?
5. Use your answers to the above, plus linearity of expectation, to answer our question: what is the expected number of times you push the button before you see all n words? It's okay if your answer is a summation, but if you have time try to simplify it to get a big-Theta expression.

Group Work: Solutions

1. $\mathbb{E}X_1 = 1$.
2. $\mathbb{E}(X_2 - X_1) = \frac{1}{1-1/n}$. This is because we have a $1 - 1/n$ chance of getting a word other than the first one. We saw in the lecture video on linearity of expectation (or you know anyway) that the expected number of times you have to flip a p-biased coin before seeing heads is $1/p$, so the answer is $1/(1 - 1/n)$.
3. $\mathbb{E}(X_3 - X_2) = \frac{1}{1-2/n}$.
4. $\mathbb{E}(X_i - X_{i-1}) = \frac{1}{1-(i-1)/n}$

5. We have

$$\begin{aligned}\mathbb{E}X_n = \mathbb{E}X_1 + (X_2 - X_1) + (X_3 - X_2) + \cdots + (X_n - X_{n-1}) \quad &= \mathbb{E}X_1 + \sum_{i=1}^{n-1} \mathbb{E}(X_{i+1} - X_i) \\ &= 1 + \sum_{i=1}^{n-1} \frac{1}{1 - i/n} \\ &= 1 + \sum_{i=1}^{n-1} \frac{n}{i} \\ &= 1 + n \sum_{i=1}^{n-1} \frac{1}{i} \\ &= \Theta(n \log n).\end{aligned}$$

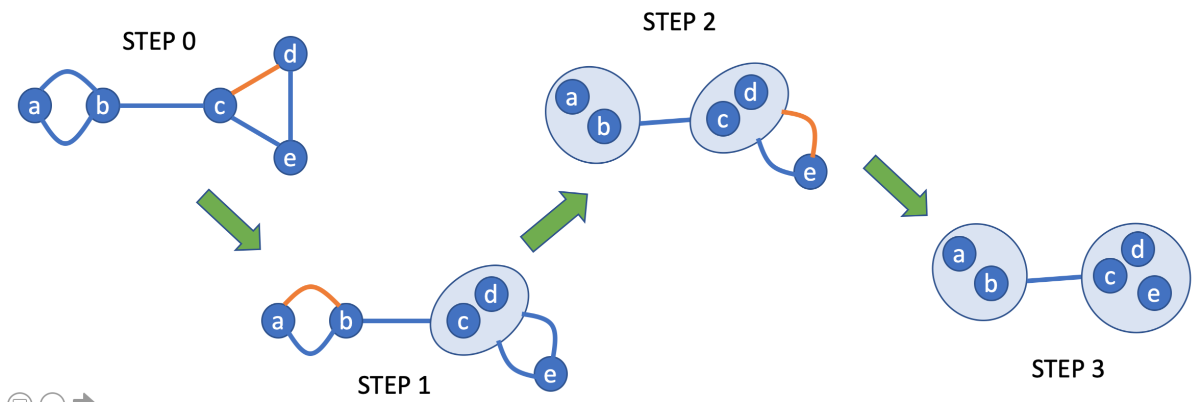
To see this last thing, you could approximate $\sum_{i=1}^{n} \frac{1}{i} \approx \int_{x=1}^{n} \frac{1}{x}\, dx = \log(n)$. (Note: if we wanted a tighter bound than a $\Theta(\cdot)$ expression, we could get that too by being more careful about the $\approx$...you'll do this on your homework!)

5 Karger-Stein Algorithm

Next, we'll take a look at a way to speed up Karger's algorithm.

5.1 Motivation for Karger-Stein

Here is one small run of Karger's algorithm:



Group Work

What is the probability of failure at each point in this run? That is, what is the probability that we choose an edge crossing the minimum cut?

Based on your answers, think of ways to improve Karger's algorithm. (In particular, might there be a smarter way to boost the success probability than just to repeat it a bunch of times?)

Group Work: Solutions

The answer is $1/6, 1/5, 1/3$. We see that repeating Karger's algorithm is wasteful since earlier steps are more likely to be successful than later steps; we should repeat later steps more!

Your answer perhaps motivates the following algorithm:

MODIFIED-KARGER:

1. Start with a graph G on n vertices.
2. Run Karger's algorithm (once) until there are m vertices remaining. Call the graph you end up with G'. (Note that G' will have some "mega-vertices" comprised of merged vertices).
3. Repeat Karger's algorithm k times independently on G' until we end up with only two mega-vertices. Return the smallest cut we find.

5.2 Group Work

Consider the Modified-Karger algorithm above. In this group work, you will analyze this algorithm.

Group Work

1. Give a bound on the probability, in terms of n and m and k, that MODIFIED-KARGER is successful. (You may not be able to find the probability exactly, but give a decent lower bound, like we did for the original Karger's algorithm; it's okay if your expression is a bit complicated).

   You may want to consult the lecture notes on Karger's algorithm. They are available on the course website: `cs265.stanford.edu`

   Hint*: You can break up the failure probability into two parts: the probability that we choose an edge crossing the mincut when reducing G to G', and the probability that we choose an edge crossing the cut in* all *of the k runs of Karger of G'.*

2. Choose $m = \sqrt{n}$ and $k = n \log n$. Use part 1 to show that the success probability of MODIFIED-KARGER is $\Omega(1/n)$.

   Hint*: The fact that we're looking for a $\Omega(\cdot)$ answer means that it's okay to ignore pesky constants in your analysis.*

Hint: *You might want to use the useful fact that* $1 - x \leq e^{-x}$ *for all* x.

3. Show that if you repeat Modified-Karger, with the parameters above, $\Theta(n)$ times, you can obtain a success probability of 0.99.
4. How does this compare to the original Karger's algorithm? More precisely:

   We saw in the mini-lecture that repeating Karger's algorithm $O(n^2)$ times leads to a success probability of 0.99. This would involve $O(n^3)$ edge contractions (n edge contractions per run of Karger's algorithm).

   You've just shown that "repeating Modified-Karger $\Theta(n)$ times" also has a success probability of 0.99. How many edge contractions does this involve?
5. If you're done with the above and we haven't finished group work yet, you can think about the following two challenge problems:

   Above, we saw how to improve Karger's algorithm by breaking one step down into two steps. What if we were to iterate on this idea? (That is, instead of just running Karger on G', recursively run Modified-Karger on G'). How would you pick the parameters here? How few edge contractions can you get, if you want success probability 0.999?

   Can you derandomize Karger's algorithm?

Group Work: Solutions

1. The probability that the Modified Karger's algorithm fails is the probability that the first step fails or the second step fails. By the union bound, this is bounded above by

   $$\Pr[G \to G' \text{fails}] + \left(\Pr[\text{ fail on } G' \text{ }]\right)^k .$$

   The probability that Karger's algorithm chooses an edge crossing the min cut when reducing G to G' is at most

   $$1 - \left(\frac{n-2}{n}\right)\left(\frac{n-3}{n-1}\right)\cdots\left(\frac{m}{m+2}\right)\left(\frac{m-1}{m+1}\right) = 1 - \frac{m(m-1)}{n(n-1)},$$

   using exactly the same reasoning we used when we analyzed Karger's algorithm in the mini-lecture.

   The probability that Karger's algorithm fails on G' is $1 - \frac{2}{m(m-1)}$, since this is just plugging in the result from the mini-lecture into a graph with m vertices instead of n vertices.

Adding them together, we get a bound of:

$$1 - \frac{m(m-1)}{n(n-1)} + \left(1 - \frac{2}{m(m-1)}\right)^k$$

2. Plugging in $m = \sqrt{n}$ and $k = n \log n$, and (following the hint), ignoring pesky constants, the failure probability from part 1 is at most

$$\begin{aligned}
1 - \frac{m(m+1)}{n(n-1)} + \left(1 - \frac{2}{m(m-1)}\right)^k &\approx 1 - \frac{m^2}{n^2} + (1 - 2/m^2)^k \\
&\leq 1 - \frac{n}{n^2} + \left(e^{-2/n}\right)^{n \log n} \\
&= 1 - \frac{1}{n} + e^{-2 \log n} \\
&= 1 - \frac{1}{n} + \frac{1}{n^2} \\
&\leq 1 - 1/2n.
\end{aligned}$$

3. If we repeat $100n$ times, the failure probability is

$$(1 - 1/(2n))^{100n} \leq e^{-50},$$

which is tiny.

4. The total number of edge contractions is $O(n^{5/2} \log n)$. This is because there are $O(n)$ contractions to reduce G to G', and $O(km) = O(n^{3/2} \log n)$ to repeat Karger k times on G'. The second part dominates, and then we repeat all of this $\Theta(n)$ times, to get $O(n^{5/2} \log n)$. This is better than $O(n^3)$!

5. Check out [Karger, Stein 1996] (linked from the course website) to see one solution! Check out [Karger 1998] (also linked from website) for another algorithm, also based on random contractions, that gets near-linear time!

   The state-of-the-art deterministic algorithm is (I believe) [Kawarabayashi, Thorup, 2015]. They get a near-linear time deterministic algorithm!