Problem Set 1 CS265/CME309, Autumn 2022

Due: Friday 10/7, 11:59pm on Gradescope

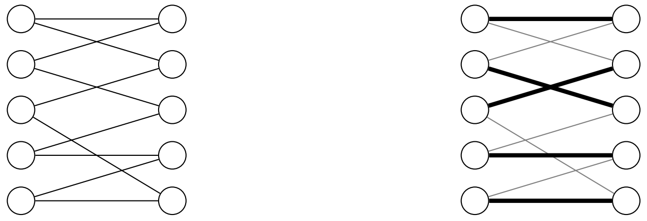
Please follow the homework policies on the course website.

---

1. **(5 pt.) [Perfect Matchings.]**

   Let $G = (V, E)$ be a *bipartite graph* with n vertices on each side. A *perfect matching* in G is a list of edges $M \subset E$ so that every vertex in V is incident to exactly one edge.

   For example, here is a bipartite graph G (on the left), and a perfect matching in G (shown in **bold** on the right):



   Your goal is to determine if the graph G has a perfect matching.

   There are efficient deterministic algorithms for this problem, but in this problem you'll work out a simple randomized one.[1]

   (a) **(2 pt.)** Recall that the *determinant* of an $n \times n$ matrix A is given by

$$\det(A) = \sum_{\sigma \in S_n} \mathrm{sgn}(\sigma) \prod_{i=1}^{n} A_{i,\sigma(i)},$$

   where the sum is over all permutations $\sigma : \{1, \ldots, n\} \to \{1, \ldots, n\}$, and where $\mathrm{sgn}(\sigma)$ denotes the signature[2] of the permutation σ. (For example, if $n = 3$, then the function $\sigma : \{1, 2, 3\} \to \{1, 2, 3\}$ that maps $1 \mapsto 2$, $2 \mapsto 1$, $3 \mapsto 3$ is a permutation in S_3. The signature of σ happens to be -1, although as noted in the footnote, if you haven't seen this definition before, don't worry about it).

   Let A be the $n \times n$ matrix so that

$$A_{ij} = \begin{cases} x_{ij} & (i,j) \in E \\ 0 & \text{otherwise} \end{cases}$$

   where the x_{ij} are variables, and $(i, j) \in E$ if and only if the i-th vertex on the left and the j-th vertex on the right are connected by an edge in G. Notice that $\det(A)$ is a multivariate polynomial in the variables x_{ij}.

   Explain why $\det(A)$ is not identically zero if and only if G has a perfect matching.

[1]This randomized algorithm has the advantage that (a) it generalizes to all graphs (not necessarily bipartite), and (b) it can be parallelized easily. Moreover, it's possible to generalize it to actually recover the perfect matching (and not just decide if there is one or not).

[2]The *signature* of a permutation is defined as -1 if the permutation can be written as an odd number of transpositions, and $+1$ otherwise. **The exact definition isn't important** to this problem, all you need to know is that it's either ± 1 in a way that depends on σ.

(b) **(3 pt.)** Use the part above to develop a randomized algorithm for deciding whether or not there is a perfect matching. Your algorithm should run in $O(n^3)$ operations. If G has no perfect matching, your algorithm should return "There is no perfect matching" with probability 1. If G has a perfect matching, your algorithm should return "There is a perfect matching" with probability at least 0.9.

You should clearly state your algorithm and explain why it has the desired properties.

[**HINT:** *You may use the fact that one can compute the determinant of a matrix $A \in \mathbb{R}^{n\times n}$ in $O(n^3)$ operations.*]

(c) **(0 pt.)** **[Optional: this won't be graded.]** Extend your algorithm to actually *return* a perfect matching. And/or, extend your algorithm to non-bipartite graphs. As a hint, consider the matrix

$$A = \begin{cases} x_{ij} & \{i,j\} \in E \text{ and } i < j \\ -x_{ji} & \{i,j\} \in E \text{ and } i \geq j \\ 0 & \text{else} \end{cases}$$

2. **(8 pt.)** Suppose you are rolling a fair, 6-sided die repeatedly.

(a) **(4 pt.)** What is the expected number of rolls until you get two 3's in a row (counting both 3's)? Justify your answer.

[**HINT:** *The answer is not* 36*...*]

[**HINT:** *If you find yourself doing a tedious computation, try to think of a simpler way. Perhaps look to the mini-lecture on linearity of expectation for some inspiration...*]

(b) **(4 pt.)** What is the expected number of rolls until you get a 3 followed by *either* a 3 or a 4 (counting both rolls)? Justify your answer.

(c) **(0 pt.)** **[Optional: this won't be graded]** What is the expected number of rolls until you get a 3 followed by a 4 (counting both rolls)?

3. **(10 pt.)** Suppose you are given a fair coin (that is, it lands heads/tails with probability 1/2 each) and want to use it to "simulate" a coin that lands heads with probability exactly 1/3. Specifically, you will design an algorithm whose only access to randomness is by flipping the fair coin (repeatedly, if desired), and your algorithm should return "heads" with probability exactly 1/3 and "tails" with probability exactly 2/3.

(a) **(4 pt.)** Prove that it is *impossible* to do this if the algorithm is only allowed to flip the fair coin at most $1,000,000,000$ times.

[**HINT:** *Read the next two parts of the problem first...*]

(b) **(4 pt.)** Design an algorithm for the above task that flips the fair coin a finite number of times *in expectation.*

(c) **(2 pt.)** Show that for *any* value v in the interval $[0,1]$, there is an algorithm that flips a fair coin at most 2 times in expectation, and outputs "heads" with probability v and "tails" with probability $1-v$.

Note: if you do this part correctly, you can write "follows from (c)" in part (b) get full credit for both parts.

[**HINT:** *Think about representing the desired probability in its binary representation.*]

4. **(8 pt.)** You are walking down a street full of restaurants, trying to decide where to eat for lunch. There are n restaurants on the street, but you haven't been to this street before, so you're not sure what's coming up, and you don't want to walk all the way to the end of the street and then potentially double back to pick the best-looking restaurant.

   Instead, your plan is to walk for a little while to get a sense of the restaurant distribution, then eat at the first place that looks good after that. Formally, you will walk past the first m restaurants, and assign them all a rating based on how good they look. Then, starting with the $m+1$'st restaurant you pass, you will eat at the first one you find that you rate higher than *all* of the first m restaurants. (Suppose for simplicity that there are no ties in the ratings). If you get to the end of the street and don't find such a restaurant, you don't get lunch :(.

   Suppose that the restaurants are in a uniformly random order on the street.

   (a) **(4 pt.)** Let E be the event that you end up eating at the best-looking (i.e., your highest-rated) restaurant on the whole street. Show that

   $$\mathbb{P}[E] = \frac{m}{n} \sum_{j=m+1}^{n} \frac{1}{j-1}.$$

   [**HINT:** *Let E_i be the event that the i'th restaurant you pass is the best-looking one, and that you eat there. What is the probability of E_i?*]

   (b) **(2 pt.)** Bound $\sum_{j=m+1}^{n} \frac{1}{j-1}$ to show that

   $$\frac{m}{n}(\ln n - \ln m) \leq \mathbb{P}[E] \leq \frac{m}{n}(\ln(n-1) - \ln(m-1))$$

   [**HINT:** *Compare the sum to an integral.*]

   (c) **(2 pt.)** Explain how to pick m so that

   $$\mathbb{P}[E] \geq 1/e - o(1).$$

   (Here, the $o(1)$ term means something that tends to 0 as $m, n \to \infty$.)

   [Note: the $o(1)$ term was added 10/5/22.]

   [**HINT:** *What m should you pick to maximize $\frac{m}{n}(\ln n - \ln m)$?*]