15295 Spring 2019 #10 -- Counting and Probability -- Problem Discussion
April 3, 2019

This is where we collectively describe algorithms for these problems. To see the problem statements follow this link. To see the scoreboard, go to this page and select this contest.

A. Archer

Simple Math:
$\frac{a}{b} + (1-\frac{a}{b})(1-\frac{c}{d})\frac{a}{b} + (1-\frac{a}{b})^2(1-\frac{c}{d})^2\frac{a}{b} + \cdots = \frac{a}{b}\frac{1}{1-\frac{(b-a)(d-c)}{bd}} = \frac{ad}{ad+bc-ac}$
---Yucheng

B. Wet Shark and Flowers

Notice that the contribute of each adjacent pair are independent of each other. Therefore, if we need to calculate the contribute of $s_i s_{i+1} (i \in [1, n],\ s_{n+1} = s_n)$, we can first calculate the probability that $p \mid s_i s_{i+1}$. Since p is a prime, we have either $p \mid s_i\ or\ p \mid s_{i+1}$. Thus, the probability is $p_i + p_{i+1} - p_i \cdot p_{i+1}$ by inclusion-exclusion principle where p_i is the probability that $p \mid s_i$.
---Yucheng

C. Game on Tree

Conclusion: Ans = $\sum_{i=1}^{n} \frac{1}{dep(i)}$.
Proof: TO BE ADDED…
---Yucheng

(Inspired by Yucheng) For each node v, consider the path to it from the root. This path has dep(v) nodes. We ask the question: which of these nodes is the first to be chosen in the path? If it's anything except v, then v is useless. If v is the first chosen, then v saves those old guys and makes a difference (by exactly 1). This happens with a probability of 1/dep(n), so the contribution of v is 1/dep(v). (Could take v as a leaf for a less generalized, more convincing proof)
--Fei

D. Game with String

If we reveal ' the first time, we're going to look at the i-th letter after every 'a' in the word. Suppose we open the i-th window after the one we chose; then we'll know k iff this revealed letter appears exactly once in the list (:= letters i positions after some 'a'). This probability for i is #(unique letters in that list) divided by the size of the list (=#('a')). Then choose the maximum among all i's. That's the probability of getting it right once the first letter is 'a'. Then weight these probabilities with the chance of actually seeing 'a' in the first place

(=#('a')/len(s)), something only wished for. Sum everything up (note: #(a) gets cancelled). We can skip later occurrences of 'a' after one calculation, so the whole thing is O(n^2).
--Fei

E. LRU

F. Devu and Flowers

Inclusion and exclusion; complexity = O(2^n) * the complexity to compute {~10^14 choose ~n-1} % p. (p=10^9+7) One can hardcode the inverse of 1! through 20! in Z/pZ. Then the latter is O(19)。。。
--Fei