

CS6550 Homework 2 Solutions

October 13, 2005

Solution 1

a) Let X_i be the random variable which is 1 if bin i is empty, otherwise 0. Note that the probability that $X_i = 1$ after m balls are thrown is $(1 - \frac{1}{n})^m$. Since X_i 's are 0,1 variables, $\mathbb{E}X_i = (1 - \frac{1}{n})^m$. Thus the expected number of empty bins is $\sum_i \mathbb{E}X_i = n(1 - \frac{1}{n})^m$

b) The probability that no bin gets more than a single ball is given by $\frac{m! \binom{n}{m}}{n^m}$, where the numerator is the number of ways one can arrange m balls in n bins with no bin getting more than a single ball, while the denominator is the total number of possible configurations. For this probability to be less than $1/2$, we need

$$\begin{aligned} \frac{m! \binom{n}{m}}{n^m} &= \prod_{i=0}^m (1 - \frac{i}{n}) \\ &\leq \prod_{i=0}^m e^{-\frac{i}{n}} \\ &= e^{-\sum_{i=0}^m \frac{i}{n}} \\ &= e^{-\frac{m(m+1)}{2n}} \end{aligned}$$

to be less than $1/2$. Thus we need $m = O(\sqrt{n})$

Solution 2

The algorithm is this: Toss the coin N times, let n be the number of heads. Let $p' := \frac{n}{N}$. If p is the real bias, then by Chernoff Bounds we have $\mathbb{P}[|n - Np| \geq \epsilon Np] \leq (\frac{e^\epsilon}{(1+\epsilon)(1+\epsilon)})^{Np} \leq e^{-Npc_\epsilon}$ where c_ϵ is a constant depending on ϵ alone. Thus dividing by N gives us $\mathbb{P}[|p' - p| > \epsilon p] \leq e^{-Npc_\epsilon} \leq e^{-aNc_\epsilon}$. Thus we need $e^{-aNc_\epsilon} \leq \delta$ which gives us $N \geq \frac{1}{ac_\epsilon} \log \frac{1}{\delta}$.

Solution 3

a) The algorithm is as follows: One keeps tracks of the numbers in the n intervals $(\frac{i}{n}, \frac{i+1}{n}]$. Let the set of numbers in the i th interval be N_i . Sort each N_i using any sorting algorithm, and output the answer in increasing order of N_i 's. To analyse the algorithm note that first for each a_i we have to ascertain which interval it goes to. Moreover we have to sort the intervals. The first part takes linear time. Thus we have to show that the second operation also takes linear time in expectation. In fact, if the size of N_i were a constant, then since sorting a constant number of elements takes constant time, we would be done. To be more precise, let X_i be the random variable corresponding to the number of elements in the i th interval. X_{ij} be the random variable which is 1 if the j th element goes into the i th bin, and 0 otherwise. $X_i = \sum_j X_{ij}$. Time taken by our algorithm in the sorting step is $\sum_i O(X_i^2) = \sum_i O(\sum_j X_{ij})^2 = \sum_i \sum_{(j,k)} X_{ij} X_{ik}$. Thus the expected time taken is $\sum_i \sum_{(j,k)} O(\mathbb{E}X_{ij} X_{ik})$. But X_{ij} and X_{ik} are independent, moreover the expected value of both is $\frac{1}{n}$. Putting these values, we get the expected running time of the algorithm is $O(n)$.

b) In class we saw that the probability any bin gets more than k balls is less than $e^{-f(k)}$ where f was a superlinear function. Thus with high probability ($o(1)$), we see that any bin contains a constant number of balls. Thus, by previous arguments, the algorithm described above runs in linear time with high probability.

Its important to note what with high probability means. In this case, we sufficed ourselves with $o(1)$,

some function tending to zero as n tends to infinity. Most of the times, however, this $o(1)$ function is required to be $\frac{1}{poly(n)}$.

Solution 4

a) We might assume that there are only three possible configurations we need to consider, namely, (HH,TT), (HT,TH), (HH,TH), where we have written first the top row and the bottom row. Any other configuration can be obtained by either changing head for tails, or by rotating the board. Since the adversary can rotate the board any how he likes, these are the only structures of interest. The point to note is if we are in the (HT,TH) configuration, then swapping any diagonals would make us win, irrespective of what the adversary does. If this were not the configuration, and (HH,TT) were the config, then flipping two adjacent coins, would lead us to the diagonal configuration, no matter what the adversary does, and then we might repeat as before. Thirdly, if there were three coins of one type on the board, then flipping any one either makes us win, or sends us to one of the earlier configurations. Thus the following seven steps suffice: Flip diagonals, Flip Adjacent, Flip diagonal, Flip any one, Flip diagonal, Flip adjacent and Flip Diagonal.

(b) If there were only three coins, then we may assume there are two of one type and one of the other. Now no matter what the player does, the adversary can always rotate the board and stay in the same state, showing that the player would never win.

(c) Pick any random subset of coins and flip them. There are 2^n subsets, and 2 of them make us win. Thus with probability $2^{-(n-1)}$ we shall win (note that the randomness renders the adversary powerless). Thus in an expected number of 2^{n-1} steps we would win.

(d) The strategy here is as follows: pick any coin at random, if its heads keep it as it is, if its tails, flip it. Thus we would win if we have flipped all the tail coins unto heads. This is exactly the coupon collector problem where the tail coins are the coupons. Thus in expected $n \log n$ steps we shall get all our tail coins, that is, we shall win.