Name _____

CS302 - Midterm 2 Review Questions

1. Probability Questions

   (a) If you have a coin that has 1/4 probability of heads and 3/4 probability of tails, what is the sample space if you flip it n times? What is the expected number of heads? (Use indicator random variables).

   (b) What is the expected runtime of the randomized search with replacement algorithm from the homework, if there are two elements with value x (the value we are looking for) in the array?

   (c) Explain why the probability of comparing z_i and z_j in Randomized QuickSort is $2/(|j-i|+1)$. (We discussed this in class - please explain in your own words.)

Solution

(a) The sample space is equivalent to all possible strings of length n made up of the letters H and T, corresponding to all possible sequences of heads and tails that might result. Let X_i be the indicator random variable that takes value 1 if the ith flip is heads, and zero otherwise. Then if X is the total number of heads $X = \sum_{i=1}^{n} X_i$. Using linearity of expectation, we have

$$\mathbb{E}[X] = \sum_{i=1}^{n} \mathbb{E}[X_i] = \sum_{i=1}^{n} Pr(i\text{th toss is heads}) = \sum_{i=1}^{n} \frac{1}{4} = \frac{n}{4}. \tag{1}$$

(b) Let X be the random variable that is the number of rounds of the search algorithm. Let X_k be the indicator random variable that takes value 1 if round k occurs. Then

$$\mathbb{E}[X] = \sum_{k=1}^{\infty} \mathbb{E}[X_k] = \sum_{k=1}^{\infty} Pr(k\text{th round occurs}) \tag{2}$$

Now the probability that the kth round occurs is the probability that we have not found the item we are searching for *before* the kth round, which is

$$\left(\frac{n-2}{n}\right)^{k-1}. \tag{3}$$

Thus

$$\mathbb{E}[X] = \sum_{k=1}^{\infty} \left(\frac{n-2}{n}\right)^{k-1} = \sum_{k=0}^{\infty} \left(\frac{n-2}{n}\right)^{k} = \frac{1}{1-\frac{n-2}{n}} = \frac{n}{2}. \tag{4}$$

(c) Items only get compared if one is the pivot, and the other is in the section of the array in that recursive call. As long as the pivot is chosen to be z_k where $k < i$ or $k > j$, then z_i and z_j move together in a recursive call. So something interesting only happens when the pivot is either chosen to be z_i or z_j, or is chosen to be z_k with $i < k < j$. In the later case, z_i and z_j are split into different recursive calls and so are never compared. In the former, they do get compared. Because there is equal probability of any of the points from z_i to z_j being chosen, there is a $2/(|j - i| + 1)$ probability that z_i or z_j is chosen from among this set, in which case, they are compared.

2. Suppose you have a graph T that is a binary tree, with weights on each vertex. Let T_v be the subtree with root v. Let $S(T_v)$ be the max-weight-independent set on T_v and let $W(T_v)$ be the weight of the max-weight independent set on T_v. Go through the steps of creating a dynamic programming algorithm.

   (a) What are the options for the optimal solution.

   (b) For each option, what is the form of the optimal solution in terms of the optimal solution of subproblems.

   (c) Use this analysis to create a recurrence relation for the maximum value of the objective function.

   (d) Write pseudocode to fill in an array with values of the objective function

Solution

(a) The root is part of the MWIS or the root is not part of the set.

(b) Let u and v be the children of r, the root. If the root is not part of the MWIS, then the MWIS is $S(T_u) \cup S(T_v)$. Let w, x, y, z be the grandchildren of r. Then the MWIS is $S(T_w) \cup S(T_x) \cup S(T_y) \cup S(T_z)$.

(c) $W(T) = \max\{W(T_u)+W(T_v), W(T_w)+W(T_x)+W(T_y)+W(T_z)\}$, $W(leaf)$ =weight of the leaf. $W(\emptyset) = 0$.

(d) (Sketch) Label the nodes of the tree in order of depth, starting with the leaves, and ending with the root. Create an array A with an index for each node that will containt $W(T_v)$ for each v. Initialize the values of the leaves, and then use a for loop to fill in the array using the recurrence relation from part c (with appropriate children and grandchildren nodes).