CS 5633 Analysis of Algorithms – Spring 12

2/2/12

3. Homework

Due **2/9/12** before class

Always justify your answers.

1. **Dice game**

   (a) Consider playing the following game: You roll one loaded six-sided die, where the probability of rolling a six is 1/11 and the probability of rolling any other number is 2/11. When rolling a six you win $20 when rolling any other number you lose $3. Compute the expected win/loss of this game.

   (b) Now consider playing the same game but rolling two loaded dice of the same type. For each six included in your result you win $20 and for every other number you lose $3. Use linearity of expectation to compute the expected win/loss of this game.

   (c) Now consider playing the same game but rolling k loaded dice. Use linearity of expectation to compute the expected win/loss of this game.

   Clearly describe the sample space and the random variables you use. Half of the points will be given for correct notation. (The point of this exercise is to learn the notation, not just to get the intuition right.)

2. **Random permutation**

   (a) Give pseudocode to compute a random permutation of an array of n distinct numbers. What is the runtime of your algorithm? (Can you make the algorithm be in-place, i.e., using only additional constant space?)

   (b) Assume you are given an array $A[1..n]$ of n distinct numbers, and you compute a random permutation of it. What is the expected number of indices that contain the same number before and after? Clearly describe the sample space that you are using, and break your overall random variable into multiple (indicator) random variables. Use linearity of expectation.

3. **Quicksort**

   Consider the following types of input of n distinct numbers: (1) Sorted input, (2) reverse-orderd input, (3) random input.

   Determine the runtime of quicksort with the following pivot choices, for all three input types:

   (a) The pivot is chosen as the first element.

   (b) The pivot is chosen as the larger of the first two elements.

   (c) The pivot is chosen as a random element.

FLIP OVER TO BACK PAGE ⟹

4. **Quicksort with duplicate keys**
   This question is concerned with quicksort on arrays that contain duplicate keys.

   (a) How does deterministic quicksort behave on an array with n equal keys? What is its runtime? What is the behavior and the runtime of randomized quicksort in this case? Justify your answer.

   (b) If you change $A[j] \leq x$ to $A[j] < x$ in the pseudocode for partition, how does quicksort behave on an array with n equal keys? What is its runtime?

   (c) How does deterministic quicksort behave on an array with just two distinct keys (the total number of keys is still n)?

   (d) Give pseudo-code for a 3-way partition that partitions the array into three parts: keys less than the pivot, keys equal to the pivot, keys greater than the pivot. Your code should be in-place (so it should use at most constant extra storage) and it should run in linear time.

   (e) Consider an implementation of quicksort which uses 3-way partition and only recurses on the portions of the array with keys less than the pivot and with keys greater than the pivot. If the array of n keys contains only 2 different values, what is the worst-case runtime?

   (f) If the array contains n keys of d different values, show that the worst-case runtime of this variant of quicksort that uses 3-way partition is $O(dn)$.