CMPS 2200 Introduction to Algorithms – Fall 14

11/18/14

9. Homework

Due **12/2/14** at the beginning of class

Remember, you are allowed to turn in homeworks in groups of two.

1. **Random permutation (8 points)**
   (a) (3 points) Give pseudocode for an efficient algorithm that computes a random permutation of an array of n distinct numbers. What is the runtime of your algorithm?
   (b) (5 points) Assume you are given an array $A[0..n-1]$ of n distinct numbers, and you compute a random permutation of it. Use linearity of expectation to compute the expected number of fixpoints (indices that contain the same number before and after).

   Clearly describe the sample space and the random variable that you are using, and break your overall random variable into multiple (indicator) random variables.

2. **Minimum in an array (8 points)**
   An array $A[0..n-1]$ contains n distinct numbers that are randomly ordered, with each permutation of the n numbers being equally likely. The task is to compute the expected value of the index of the minimum element in the array.
   (a) (1 point) Describe the sample space.
   (b) (1 point) Describe the random variable of interest *(Hint: We want to compute the expected value of the random variable; look at the problem statement.)*
   (c) (2 points) Consider an example array of $n = 5$ numbers. Consider four different orderings of the numbers in this array, and for each of these orderings provide the value of the random variable.
   (d) (2 points) Now consider an arbitrary n again. What is the probability that the minimum of the array is contained in the first slot? And what is the probability that it is contained in the second slot?
   (e) (2 points) Use the following definition of an expected value to compute the expected value of your random variable.

$$E(X) = \sum_{x \in \{0,1,\ldots,n-1\}} P(X = x) * x$$

   Note that $P(X = x)$ is short for $p(\{s \in S \mid X(s) = x\})$, or in other words this is the probability that the random variable X takes one specific value x.

Flip over to next page $\Longrightarrow$

3. **Best case for quicksort (4 points)**
Let "Deterministic Quicksort" be the non-randomized Quicksort which takes the first element as a pivot, using the partition routine that we covered in class on slide 12 of the randomized algorithms slides.

   In the best case the pivot always splits the array in half, for all recursive calls of Deterministic Quicksort. Give a sequence of 3 distinct numbers and a sequence of 7 distinct numbers that cause this best-case behavior. *(Hint: For the sequence of 7 numbers the first two recursive calls should be on sequences of 3 numbers each.)*

4. **Sorting runtimes (6 points**
Consider the input array $[2^{n-1}, 2^{n-2}, \ldots, 2^3, 2^2, 2, 1]$ of n numbers. For this particular input array, what are the runtimes of the following algorithms? Justify your answers shortly.

   (a) Deterministic insertion sort.

   (b) Randomized insertion sort.

   (c) Deterministic quicksort where the pivot is always chosen as the first element.

   (d) Randomized quicksort.

   (e) Mergesort.

   (f) Counting sort.