CMPS 2200 Introduction to Algorithms – Fall 17

11/29/17

# 11. Homework

Due **12/6/17** at the beginning of class

**Remember, you are allowed to turn in homeworks in groups of two. One writeup, with two names.**

1. **Decision tree (5 points)**
   Below is the code for *Bubble Sort*:

```
void bubbleSort(int A[1..n]){
  for(int i=1; i <= n; i++)
    for(int j=n; j >= i+1; j--)
      if(A[j]<A[j-1])
        swap(A[j],A[j-1]);

}
```

   Draw the decision tree for Bubble Sort for an array $A[1..3]$ of $n = 3$ elements. Annotate the decision tree with comments indicating the part of the algorithm that a comparison belongs to.

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.

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

   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 back page $\Longrightarrow$

3. **Sorting runtimes (7 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.
   (g) Radix sort.

**Extra credit: Lower bound for comparison-based searching (5 extra credit points)**
Consider the problem of searching for a given key in a sorted array of $n$ numbers. Use a decision tree to show a lower bound of $\Omega(\log n)$ for any comparison-based search algorithm. *(Hint: What should be stored in the leaves as the output of the search algorithm?)*