CS 5633 Analysis of Algorithms – Spring 08

2/7/08

4. Homework

Due **2/14/08** before class

1. **Heapsort decision tree (5 points)**
   The decision tree on the slides in class was for insertion sort of three elements. Draw the decision tree for heapsort. Use the heapsort pseudo code covered in class, and annotate the decision tree with comments about the state of the heap and location in the pseudocode.

2. **Radix sort and counting sort (3 points)**
   Given n numbers between 0 and $f(n) - 1$. Please give the runtimes for counting sort, and for radix sort for each of the functions $f(n)$ below, and state which of the two sorting algorithms is faster in each case.

   a) $f(n) = n^3$ b) $f(n) = 2^n$

3. **Radix Sort vs. Insertion Sort (2 points)**

   Given n numbers between 0 and $n!$. Which sorting algorithm is faster: Insertion Sort or Radix Sort? Justify your answer.

4. **Radix sort with most significant digit first (5 points)**
   Try to sort the numbers

   $$175, 126, 347, 777, 711, 339, 727, 348, 376, 176, 335$$

   using radix sort but starting with the **most** significant digit (so, from left to right, not from right to left).

   Why would a program that implements this strategy be much more complicated than the radix sort that starts with the least significant digit? (Hint: What kind of variables or data structures would you have to maintain?)

5. **Sorting (4 points)**
   Please give the runtimes of the sorting algorithms listed below, each for an input array of n elements sorted i) in increasing order and ii) in decreasing order.

   a) Runtime of merge sort

   b) Runtime of deterministic quicksort (pivot = 1st element)

   c) Expected runtime of randomized quicksort (pivot = random element)

   d) Runtime of heapsort (using a max-heap)

FLIP OVER TO BACK PAGE $\Longrightarrow$

6. **Randomized code snippet (4 points)**
Consider the following code snippet:

```
for(i=1; i<=n; i++){
  if(RandomInteger(i)==1){
    for(j=1; j<=n; j++){
      print(‘‘42 ’’);
    }
  }
}
```

Suppose `RandomInteger(i)` takes $O(1)$ time and returns an integer between 1 and i, each with probability $1/i$.

a) What is the worst case runtime, in terms of n, of this code snippet? Describe what triggers a worst-case scenario.

b) Now analyze the **expected** runtime. Clearly define your random variable. *Hint: Define a separate random variable for each iteration of the loop. And remember, random variables are functions.*

Related questions from previous PhD Exams

Just for your information. You **do not** need to solve them for homework credit.

P1 (a) A comparison sort is an algorithm that sorts based only on comparisons between pairs of input elements. Which of the following sorting algorithms are comparison sorts: bucket sort, counting sort, heapsort, insertion sort, merge sort, quicksort, radix sort?

(b) A binary decision tree is a binary tree in which the internal nodes are boolean expressions and the leaves are the outcomes. The execution of the decision tree traces a path from the root to a leaf. At each internal node, its expression is evaluated, and the left branch is taken if the expression is true; otherwise the right branch is taken.

i. Describe the decision tree model of comparison sorts.

ii. What are the expressions in the internal nodes?

iii. What are the outcomes at the leaves?

iv. Explain why there are at least $n!$ leaves.

(c) Show an example of the decision tree model for $n = 3$.

(d) The height of the decision tree model represents the worst-case number of comparisons. Show that the height of the decision tree model must be at least $\lg(n!)$.

(e) Show that $\lg(n!)$ is $\Theta(n \lg n)$. Hint: Use the fact that $\lg(mn) = \lg(m) + \lg(n)$.