CS 3343 Analysis of Algorithms – Fall 07

10/9/07

5. Homework

Due: Tuesday 10/16/07 before class

1. **Quicksort**
   In the following, if not said otherwise, assume Quicksort always takes the first element as a pivot (so, no randomized pivot), and it uses the Partition routine we had in class (you can find them on the slides or online on the class webpage).

   (a) In the best case the pivot always splits the array in half, for all recursive calls of quicksort. Give a sequence of 3 distinct numbers, a sequence of 7 distinct numbers, and a sequence of 15 distinct numbers that cause this best-case behavior. (For the sequence of 15 numbers the first two recursive calls should be on sequences of 7 numbers each, and the next recursive calls on sequences of 3 numbers).

   (b) The runtime analyses in class were for Quicksort on an array with distinct elements. Assume now there is an array given that contains the same number n times (so for example: $2, 2, 2, \ldots, 2$). What is the runtime of Quicksort?

   (c) Assume again that an array is given that contains the same number n times. If the pivot is chosen as the i-th element in the array (instead of the first), what is the runtime of Quicksort now?

2. **d-Heaps**
   A *d-ary max-heap*, d-heap for short, is the generalization of a binary heap to a d-ary tree. The tree still has to be almost complete, and for every child of a parent the child's value is less or equal than the parent's value.

   (a) Suppose a d-heap is stored in an array (that begins with index 0). For an entry located at index i in which location is its parent and in which locations are its children? (You do not have to formally prove your answer, but please give at least an example)

   (b) What is the height of a d-heap that contains n elements? The height should be a function of n and d.

   (c) What is the runtime of inserting an element into a d-heap of n elements? The runtime should be a function of n and d (so, do not consider d as a constant).

   (d) What is the runtime of extracting the maximum from a d-heap of n elements? The runtime should be a function of n and d (so, do not consider d as a constant).

FLIP OVER TO BACK PAGE ⟹

3. **Decision Tree**
Consider the selection sort algorithm which sorts an array $a[0 \ldots n-1]$ of n numbers in place:

```
void selectionSort(int a[], int n){

  for (int i=0; i<n-1; i++){

    int min = i;
    for (int j = i+1; j<n; j++){
       if (a[j] < a[min])
          min = j;
    }
    swap(a[i], a[min]);
  }
}
```

Please draw the decision tree for selection sort for $n = 3$. Note that the indices start at 0 and not at 1. (*Hint: It is very helpful to make notes on the decision tree, for example showing the current state of the array.*)