CMPUT204
Department of Computing Science
University of Alberta

Homework Assignment #2
Due: Oct 13, 2004

Question 1. (28 marks) Prove asymptotic upper and lower bounds for each of the following recurrences. Assume that in each case, the function is some positive constant for sufficiently small n. To make it more convenient, you may assume that $n = c^k$ for some constant c that you choose. Make your bounds as tight as you can.

a. (4 marks) $T(n) = 7T(\frac{n}{3}) + n\sqrt{n}$.

b. (4 marks) $T(n) = T(n-1) + \frac{1}{n}$

c. (5 marks) $T(n) = 2T(\frac{n}{2}) + \frac{n}{\lg n}$.

d. (5 marks) $T(n) = T(n-1) + n \lg n$.

e. (5 marks) $T(n) = 3T(\frac{n}{3} + 1) + \frac{n}{2}$.

f. (6 marks) $T(n) = T(\frac{n}{2}) + 2T(\frac{n}{4}) + 2n$. (Hint: use induction directly to prove $T(n) \in O(n \log n)$ and $T(n) \in \Omega(n \log n)$).

Question 2. (6 marks) Suppose that we want to sort a list A consisting of a sorted list with many elements followed by a few "random" elements. Which of the sorting methods we have seen so far would be especially suitable for such a task? Explain why and justify your answer.

Question 3. (25 marks) A d-ary heap is like a binary heap, except that non-leaf nodes have d children instead of 2 children (with one possible exception for the last leaf).

a. (2 marks) Explain how to implement a d-ary heap in an array and implement the corresponding Parent and Child operation as we had for the binary heap.

b. (2 marks) What is the height of a d-ary heap of n elements in terms of n and d?

c. (4 marks) Write a version of Max-Heapify for a d-ary max-heap and analyze its running time.

d. (5 marks) Write a version of Build-Heap for a d-ary max-heap and analyze its running time.

e. (4 marks) Give an efficient implementation of Extract-Max in a d-ary heap and analyze its running time.

f. (4 marks) Give an efficient implementation of Increase-Key(A, i, k), which first sets $A[i] \longleftarrow \max(A[i], k)$ and then updates the d-ary heap appropriately. Analyze the running time.

g. (4 marks) Write a version of Insert in a d-ary heap and analyze its running time.

Question 4. (12 marks)

a. (10 marks) Give an algorithm (in pseudo-code) for the following problem: given an array A of size n of integers, find the $\sqrt{n}$ largest elements of A and return them. Your algorithm must run in $O(n)$. Explain your algorithm and analyze its running time.

b. (2 marks) Suppose that you modify your algorithm of part (a) to find the p largest elements of A (instead of $\sqrt{n}$). What is the largest value of p for which your algorithm still runs in $O(n)$ time?

Question 5. (14 marks) Consider the following very simple and elegant(?) sorting algorithm:

```
SomeSort (A, b, e)
if e = b + 1 then
    if A[b] > A[e] then
        exchange A[b] and A[e]
    end if
else if e > b + 1 then
    p ⟵ ⌊(e − b + 1)/3⌋
    SomeSort (A, b, e − p)
    SomeSort (A, b + p, e)
    SomeSort (A, b, e − p)
end if
```

a. (4 marks) Explain why SomeSort correctly sorts its input array A, assuming that $n = e - b + 1$ is the length of the array.

b. (4 marks) Find a recurrence for the worst-case running time of SomeSort. Give a tight (i.e. Θ) asymptotic bound for the worse-case running time of SomeSort (Hint: for simplicity assume that $n = 3^k$ for some constant k).

c. (2 marks) By comparing SomeSort with insertion sort, merge sort, heap-sort, and quicksort argue if this simple algorithm is efficient.