Purdue University
CS 381

S. Hambrusch - C. Guerra
Fall 1999

Assignment 4

Due: Wednesday, Oct 6, 1999, in class

1) (15 pts.) Suppose that, instead of sorting, we wish to find the m smallest elements (in arbitrary order) in a given array of size n, where $1 \leq m \leq n$.
(i) Describe how *quicksort* can be adapted to this problem, doing less work than a complete sort.
(ii) Analyze the worst-case time performance when the pivot element is chosen at random.
(iii) Analyze the worst-case time performance when the pivot element is chosen in $O(n)$ time by a given median finding algorithm.

2) (15 pts.) The input is d sequences of elements, $S_1, S_2, \ldots, S_d$. Each sequence is already sorted and there is a total of n elements (in the d sequences). Design and analyze an $O(n \log d)$ algorithm based on Merge Sort to merge the d sequences into one sorted sequence of length n.

3) (10 pts.) Consider the $O(n)$ time solution for the celebrity problem presented in class. Assume that when the algorithm chooses two people j and k in step (1), it chooses the two people in set P having the smallest index. For example, in the first iteration, 1 and 2 are selected and compared.

For $n = 8$, give a "know relation" for which the algorithm eliminates person 2 in step (1), the recursive call on $\{1, 3, 4, 5, 6, 7, 8\}$ returns person 4 as a celebrity, and person 4 fails to be a celebrity for $\{1, 2, 3, 4, 5, 6, 7, 8\}$. Describe all possible ways this can happen.

4) (10 pts.) Consider the $O(n)$ time solution for the majority problem presented in class.
(i) Show that the majority algorithm does not find the mode of a set (show this by giving a counterexample). (The mode of a set is the element/are the elements occurring most frequently.)
(ii) Describe an $O(n)$ time algorithm to determine whether the mode in a set occurs at least $\frac{3n}{4}$ times.