9-1 Largest i numbers in sorted order

Given a set of n numbers, we wish to find the i largest in sorted order using a comparison-based algorithm. Find the algorithm that implements each of the following methods with the best asymptotic worst-case running time, and analyze the running times of the algorithms in terms of n and i .

a. Sort the numbers, and list the i largest.

b. Build a max-priority queue from the numbers, and call EXTRACT-MAX i times.

c. Use an order-statistic algorithm to find the i th largest number, partition around that number, and sort the i largest numbers.

Solutions:

We assume that the numbers start out in an array.

a. Sort the numbers using merge sort or heapsort, which take θ(n lg n) worst-case time. (Don't use quicksort or insertion sort, which can take $\theta(n^2)$time.) Put the i largest elements (directly accessible in the sorted array) into the output array, taking $\theta(i)$time.

Total worst-case running time: θ(n lg n + i) = θ(n lg n) (because i <= n).

b. Implement the priority queue as a heap. Build the heap using BUILD-HEAP, which takes θ(n) time, then call HEAP-EXTRACT-MAX i times to get the i largest elements, in θ(i lg n) worst-case time, and store them in reverse order of extraction in the output array. The worst-case extraction time is θ(i lg n)

because i extractions from a heap with O(n) elements takes i • O(lg n) = O(i lg n)

time, and half of the i extractions are from a heap with >= n/2 elements, so those i/2

extractions take (i/2)Ω(lg(n/2)) = Ω(i lg n) time in the worst case.

Total worst-case running time: θ(n + i lg n).

c. Use the SELECT algorithm of Section 9.3 to find the i th largest number in θ(n) time. Partition around that number in θ(n) time. Sort the i largest numbers in θ(i lg i) worst-case time (with merge sort or heapsort).

Total worst-case running time: θ(n + i lg i).

Note that method (c) is always asymptotically at least as good as the other two methods, and that method (b) is asymptotically at least as good as (a). (Comparing (c) to (b) is easy, but it is less obvious how to compare (c) and (b) to (a).

(c) and (b) are asymptotically at least as good as (a) because n, i lg i , and i lg n are all O(n lg n). The sum of two things that are O(n lg n) is also O(n lg n).)