CS 512

Quiz 3

Total Marks: 20

Time: 9:15 – 9:45 AM, 17-10-2020

1. An array A containing n distinct integers A[1], … ,A[n] satisfies the following condition: there exists an index k such that A[1],…,A[k] is an increasing sequence and A[k+1], …, A[n] is a decreasing sequence. Answer true or false for the following claim: There is an algorithm that finds the index k with running time O(log n). (2 marks)
   i. True
   ii. False

   Answer: True. Use divide and conquer. Find the midpoint mid and determine whether A[mid-1] < A[mid] and A[mid] < A[mid+1]. Depending on the results one half of the array can be thrown away.

2. We have seen an algorithm for solving the closest pair of points problem in 2-D in O(n log n) time. Suppose after solving the problem with n points, one more point is added to the set of points. Answer true or false for the following claim: The closest pair of the modified set of points can be obtained in O(n) additional time using the result of the original problem. (2 marks)

   Answer: True. Just compute the distance of the new point with each of the other points, find the minimum and compare with the closest pair distance in the old set.

3. A Boolean array B contains n bits B[1], … ,B[n]. The array is not sorted but we are given that B[1] and B[n] are not equal. Answer true or false for the following claim: There is an algorithm that

finds an index k such that B[k] and B[k+1] are not equal in time O (log n). (2 marks)

i. True
ii. False

Answer: True. Use binary search to maintain the invariant "B[lo] is not equal to B[hi]". Initially lo = 1 and hi = n. At every step compute the mid-point mid of lo and hi and adjust either lo or hi to be equal to mid, depending on whether B[mid] = B[lo].

4. Given a sorted array A containing n distinct integers A[1], … ,A[n] you have to find whether there is an index k for which A[k]=k. Note that the integers could be positive, negative or zero. Answer true or false for the following claim: There is an algorithm that finds whether such an index k exists with running time O(log n). (2 marks)

   Answer: True. Use binary search based on the following fact. For any i, if A[i] > i we can ignore all the indices to the right of I, including i. If A[i] < i we can ignore all the indices to the left of i, including i. If A[i] = i we answer yes.

5. We are given an array A containing n elements from some domain D. We have a divide-and-conquer algorithm as follows:

   i. Obtain the pairs (A[1],A[2]), (A[3],A[4]), …
   ii. Do the following for each pair: if the two elements are different discard both of them; if they are same just keep one of them. We have a new array A[1], A[2], .. of remaining elements.
   iii. Repeat steps (i) and (ii) till either one or no elements are left. If one element is left then answer yes; otherwise answer no.

Assuming that n is a power of 2, answer true or false for the following claim: The running time of the algorithm is O(n). (2 marks)

Answer: True. Assume $n=2^k$. Each time step (ii) is executed, the size of the array is cut by at least half. Therefore steps (i) and (2) are executed at most $\log(n) = k$ times. The number of comparisons is $1 + 2 + 3 + \ldots 2^{k-1} = 2^k - 1 = O(n)$.

6. An independent set in any graph is a subset S of nodes such that no two vertices in S are joined by an edge. A line graph is a graph G = (V,E) where the nodes in V can be ordered as $v_1, v_2, \ldots, v_n$ such that there is an edge between v_i and v_j iff $|i-j|=1$. Further, suppose we associate a positive integer weight w_i with each node v_i. We want a dynamic programming solution to the following problem: Find an independent set in the line graph G whose total weight is maximum. Suppose OPT(j) is the optimal value for the solution involving the nodes $v_1, v_2, \ldots, v_j$. Then write the complete Bellman equation for OPT(j), including the base case for the recursion. You don't have to justify your answer, just write the equation. (5 marks)

   Answer: $OPT(j) = 0$ for $j=0$

   $OPT(j) = w_j$ for $j=1$

   $OPT(j) = \max(OPT(j-1), w_j + OPT(j-2))$ for $j > 1$

7. A pizza business owner wants to open a series of outlets along a straight highway. There are n possible locations for these outlets and the distances of these locations from the start of the highway in increasing order are $d_1, d_2, \ldots, d_n$. The outlets must satisfy the following constraints. Each location may have at most one outlet. The expected profit from opening an outlet at location i is p_i

where $p_i > 0$ and $1 \le i \le n$. Any two outlets must be separated by a distance of at least d. Using dynamic programming, suppose OPT(j) is the maximum expected total profit from outlets in locations 1,2,…,j. Then write the complete Bellman equation for OPT(j), including the base case for the recursion. You don't have to justify your answer, just write the equation. (5 marks)

Answer: Let r(j) be the rightmost $i < j$ such that $d_j - d_i \ge d$. Then

$$OPT(0) = 0$$

$$OPT(j) = \max(OPT(j-1), p_j + OPT(r(j)))$$