DESIGN AND ANALYSIS OF ALGORITHMS

ASSIGNMENT-1

Submitted By - Suraj Rai
- 18102
BSc(H) computer science

Submitted To - Mr. Manvendra sir
DEPARTMENT OF COMPUTER SCIENCE

1(a) :-

Let T(n) represent the running time of sorting an array of size n using the recursive insertion sort:-

① - sort the array A[1:n-1] recursively, which take T(n-1) time

② - insert the element A[n] into the sorted array A[1:n-1]

This step required finding the correct position for the element A[n] in the sorted subarray and shifting the element if necessary.

In worth case

to compare A[n] with half of the element in the sorted subarray which is n/2 comparision

recurrance Relation

$$T(n) = T(n-1) + n/2$$

This recurrance relation state that the running time of sorting an array of size (n) is equal to running time of of sorting an array of size (n-1) plus the time required to insert the element A[n] into the sorted array A[1:n-1], which is n/2 on average.

(b) - What are the minimum and maximum ---- height h:

→ minimum number of element (N_{min}):

In Binary heap, the minimum number of element occurs when the tree is perfectly balanced.

no. of node at height h given by = 2^h

minimum no. of element (N_{min}) = 2^h

→ maximum number of element (N_{max}):

The maximum no. element occured when the heap is complete binary tree up to height h-1.

max. no. of node = $2^{(h+1)} - 1$

maximum no. of element = $2^{(h+1)} - 1$

(c) Among the given option, insertion sort is stable sorting algorithm, while Quick sort is not.

A stable sorting algorithm maintains the relative order of element with equal keys. In other word, if element have same key and appear in particular order in the input, a stable algorithm will ensure that they remain in the same order in the sorted output.

Example:.
we have following array element with key-value pairs-

[(5,A), (2,B), (5,C), (1,D), (2,E)]

using Insertion sort:.

[(1,D), (2,B), (2,E), (5,A), (5,C)]

If we use Quick sort on the same input:-

[(1,D), (2,E), (2,B), (5,A), (5,C)]

→ the element with equal key such as (2,B) and (2,E) have switched positions compared to their original order.

(d) when determining whether dynamic programing is applicable for an optimization problem, there are two factor considered-

(1) overlapping subproblem:
solving a problem by breaking it down into smaller overlapping subproblem. If the problem can be devided into subproblem that are solved independently and the same problem are encountered multiple times.

(ii) Optimal substructure:

if the optimal solution to the problem can be constructed from the optimal solution of its subproblem. if we can recursively breakdown the problem into smaller subproblem and combine their solution to obtain the optional solution for the original problem.

Shot on OnePlus
Powered by Triple Camera