Pranilash Reddy

LCB2023040

INDIAN INSTITUTE OF INFORMATION TECHNOLOGY, LUCKNOW

END TERM EXAMINATION

DESIGN AND ANALYSIS OF ALGORITHMS

B.Tech (IT/CS/CSAI/CSB)-3rd Semester

Course Instructor- Dr. Deepshikha Agarwal

Time duration: 3 hours **Max. Marks: 70**

- ***Attempt any 7 questions. All questions carry equal marks. Use Assumptions wherever required. Read the Question paper carefully***

Q1) **(i)** Solve the recurrence equation $T(n)= 4T(n-3) + n^2$ using Iteration Tree method
(ii) State the regularity condition in Master's theorem and support with a suitable example.
(iii) Write the Recurrence for the tower of Hanoi problem and compute its time complexity in Best case and Worst case. **[3+3+4]**

Q2) Compare between (i) Greedy Approach & Backtracking technique (ii) Brute force Technique and Branch and bound technique (ii) TSP Optimization problem & Decision problem **[3+3+4]**

Q3) Sort the given array using Mergesort with 3 subproblems created in every iteration (3-way Mergesort). Show each step clearly. What will happen if half of the numbers are duplicate. Compute the time complexity in best case and worst case. **[4+2+4]**

A[n] =

228	314	900	14	24	1000	433	78	51	7

Q4) **(i)** ***Job sequencing with deadlines problem:*** You are given a set of jobs. Every job has a defined deadline and some profit associated with it. The profit of a job is given only when that job is completed within its deadline. Only one processor is available for processing all the jobs. Processor takes one unit of time to complete a job. Schedule and complete the Jobs in such a way so that total profit can be maximized. Suggest a greedy approach for the same:-

Jobs	J1	J2	J3	J4	J5	J6
Deadline (units of time)	5	3	3	2	4	2
Profit (Rs.)	228	180	190	310	120	100

(ii) Answer the questions based on the procedure given below:-

```
FIND-MAX-CROSSING-SUBARRAY(A, low, mid, high)
// Find a maximum subarray of the form A[i .. mid].
left-sum = −∞
sum = 0
for i = mid downto low
    sum = sum + A[i]
    if sum > left-sum
        left-sum = sum
        max-left = i
// Find a maximum subarray of the form A[mid + 1 .. j].
right-sum = −∞
sum = 0
for j = mid + 1 to high
    sum = sum + A[j]
    if sum > right-sum
        right-sum = sum
        max-right = j
// Return the indices and the sum of the two subarrays.
return (max-left, max-right, left-sum + right-sum)
```

(a) Explain what is happening in this procedure? (b) Is it Iterative or Recursive? Elaborate
(c) Calculate the Best case complexity (d) Calculate the Worst Case time complexity
(e) Draw the order of growth graph

[5+5]

Q5) Define the n-Queen's problem. Can you suggest one solution to the 6-Queen's problem using Backtracking approach? Draw the complete State-Space tree for the solution. [2+5+3]

Q6) For the given Matrix, find a solution to the Travelling Salesman problem using Branch and bound technique. [10]

	1	2	3	4	5
1	∞	20	30	10	11
2	15	∞	30	10	11
3	3	5	∞	2	4
4	19	6	18	∞	3
5	16	4	7	16	∞

Q7) ***Longest Common Subsequence Problem-*** **(i)** Write the recursive algorithm for finding the LCS of two strings. Convert this logic into an iterative algorithm. Is there any difference in the time complexities or space complexities of both the versions?
(ii) Given two sequences X={ RGBGARGA} and Y= {BGRARG}, Compute the LCS(X,Y) using Dynamic programming.

[5+5]

Q8) Solve the following instance of the 0/1 Knapsack problem by branch-and-bound algorithm given W=16 , n=4, Total weights ={10,7,8,4} and Total profits={100,63,56,12} Solve the same instance for F-KSP using greedy approach. Compare the solutions. [4+4+2]

Q9) **(i)** Can you prove that if a problem 'X' is NP-Hard then it is also NP-Complete? Why or why not? Support your answer with a suitable example.
(ii) Elaborate why Searching problem is considered P and NP both?
(ii) Label the complexity classes-1,2,3,4 in the following diagram:- [4+2+4]