Nov - Dec
2011

[4063] – 359

T.E. (Information Technology) (Semester – II) Examination, 2011
DESIGN AND ANALYSIS OF ALGORITHMS
(New) (2008 Course)

Time : 3 Hours Max. Marks : 100

N.B. : 1) *Answer* ***three*** *questions from* ***each*** *Section.*
2) *Answer to the* ***two*** *Sections should be written in* ***separate*** *answer-books.*
3) ***Neat*** *diagrams must be drawn* ***whenever*** *necessary.*
4) *Figures to the* ***right*** *indicate* ***full*** *marks.*
5) *Assume* ***suitable*** *data,* ***if necessary.***

SECTION – I

1. a) Consider the following algorithm : **8**

int sq(n)

If n = 0 then return 0

Else return 2n + sq(n – 1) – 1

Prove by mathematical induction that the above code always return a square of numbers.

1. b) How do we analyze and measure the time complexity of algorithm ? What are the basic components, which contribute to space complexity ? In what way the asymmetry between Big-Oh notation and Big-Omega notation helpful ? **10**

OR

2. a) Prove by mathematical induction that "$n^3 < 2^n$". **8**

2. b) Reorder the following complexity from smallest to largest **10**

i) $n\log_2(n)$, $n + n^2 + n^3$, 2^4, sqrt(n)

ii) n^2, 2^n, $n\log_2(n)$, $\log_2(n)$, n^3

iii) $n\log(n)$, n^8, $n^2/\log n$, $(n^2 - n + 1)$

iv) $n!$, 2^n, $(n + 1)!$, 2^{2n}, n^n, $n^{\log n}$

P.T.O.

3. a) Consider the binary search algorithm so that it splits the input not into two sets of almost equal sizes, but into three sets of sizes approximately one third, write down the recurrence for this ternary search algorithm and find the asymptotic complexity of this algorithm. **10**

3. b) Obtain a set of optimal huffman codes for the messages (M_1 ... M_7) with relative frequencies (q_1, ..., q_7) = (4, 5, 7, 8, 10, 12, 20) draw the decode tree for this set of codes. **6**

OR

4. a) Design and analyze a divide and conquer algorithm for finding minimum and maximum number in the array of n-numbers that uses (3n/2) – 2 comparison for any n. **10**

4. b) Write algorithm to compute shortest distance between source and destination vertices of a connected graph. Will your algorithm for weights may be negative ? **6**

5. a) Consider 0/1 knapsack problem N = 3, w = (4, 6, 8), p = (10, 12, 15) using dynamic programming devise the recurrence relations for the problem and solve the same. Determine the optimal profit for the knapsack of capacity 10. **10**

5. b) What is principle of optimality in dynamic programming ? Give suitable example for this property of optimality in dynamic programming. **6**

OR

6. a) N = 3 and {a_1, a_2, a_3} = {do, if, while} let p(1:3) = (0.5, 0.1, 0.05), q(0 : 3) = (0.15, 0.1, 0.05, 0.05). Compute and construct OBST for the above values using dynamic programming. **10**

6. b) What are the common steps in the dynamic programming to solve any problem ? Compare dynamic programming with greedy approach. **6**

SECTION – II

7. a) What is backtracking method for algorithmic design ?

Solve the sum of subset problem using backtracking algorithmic strategy for the following data

$N = 4$ $(w_1, w_2, w_3, w_4) = (11, 13, 24, 7)$ and $M = 31$. **12**

7. b) Discuss and analyze the problem of finding Hamiltonian cycle using backtracking. **6**

OR

8. a) What is backtracking method for algorithmic design ?

Solve the sum of subset problem using backtracking algorithmic strategy for the following data

$N = 4$ $(w_1, w_2, w_3, w_4) = (11, 13, 24, 7)$ and $M = 31$. **12**

8. b) Consider the following instance for knapsack problem using backtracking $n = 8$

$P = (11, 21, 31, 33, 43, 53, 55, 65)$ $W = (1, 11, 21, 23, 33, 43, 45, 55)$ $M = 110$. **6**

9. a) Explain the branch and bound algorithmic strategy for solving the problem, take an example of traveling salesman problem using branch and bound. **10**

9. b) Differentiate between backtracking and branch and bound. Illustrate with example of 4-Queens problem. **6**

OR

10. Explain the 8-Queen problem and explain the following with respect to 8-queen problem : **16**

1) State space tree
2) Solution state
3) State space
4) Answer state
5) Static tree
6) Dynamic tree
7) Live node
8) Bounding function.

11. a) What do you mean by NP hard and NP complete ? Show that Knapsack's problem is NP-Complete. 10

11. b) What are the advantages of proving the relation P = NP ? 6

OR

12. a) Explain the following : 10

i) Computational complexity

ii) Decision problems

iii) Deterministic and non-deterministic algorithms

iv) Complexity classes

v) Intractability

12. b) State the Cook's theorem and prove it. 6
