MARIA COLLEGE OF ENGINEERING AND TECHNOLOGY

Design and Analysis of Algorithm

2 – Marks Question & Answer

1. **What is performance measurement?**

   Performance measurement is concerned with obtaining the space and the time requirements of a particular algorithm.

2. **What is an algorithm?**

   An algorithm is a finite set of instructions that, if followed, accomplishes a particular task. In addition, all algorithms must satisfy the following criteria:

   1) input
   2) Output
   3) Definiteness
   4) Finiteness
   5) Effectiveness.

3. **Define Program.**

   A program is the expression of an algorithm in a programming language. Sometimes works such as procedure, function and subroutine are used synonymously program.

4. **Write the For LOOP general format.**

   The general form of a for Loop is

   For variable : = value 1 to value 2 step
   Step do
   {
   <statement 1>
   <statement n >
   }

5. **What is recursive algorithm?**

   An algorithm is said to be recursive if the same algorithm is invoked in the body. An algorithm that calls itself is Direct recursive. Algorithm A is said to be indeed recursive if it calls another algorithm, which in turn calls A.

6. **What is space complexity?**

   The space complexity of an algorithm is the amount of memory it needs to run to completion.

7. **What is time complexity?**

   The time complexity of an algorithm is the amount of computer time it needs to run to completion.

8. Give the two major phases of performance evaluation

Performance evaluation can be loosely divided into two major phases:

(i) a prior estimates (performance analysis)

(ii) a Posterior testing(performance measurement)

9. Define input size.

The input size of any instance of a problem is defined to be the number of words(or the number of elements) needed to describe that instance.

10. Define best-case step count.

The best-case step count is the minimum number of steps that can be executed for the given parameters.

11. Define worst-case step count.

The worst-case step count is the maximum number of steps that can be executed for the given parameters.

12. Define average step count.

The average step count is the average number of steps executed an instances with the given parameters.

13. Define the asymptotic notation "Big on" (0)

The function f(n) = O(g(n)) iff there exist positive constants C and no such that $f(n) \leq C * g(n)$ for all n, $n \geq n_0$.

14. Define the asymptotic notation "Omega" (Ω).

The function $f(n) = \Omega (g(n))$ iff there exist positive constant C and no such that $f(n)$ C * g(n) for all n, $n \geq n_0$.

15. Define the asymptotic t\notation "theta" (θ)

The function $f(n) = \theta (g(n))$ iff there exist positive constant C_1, C_2, and no such that $C_1 g(n) \leq f(n) \leq C_2 g(n)$ for all n, $n \geq n_0$.

16. Define Little "oh".

The function f(n) = 0(g(n))

iff

$$\lim_{n \to \infty} \frac{f(n)}{g(n)} = 0$$

17. Define Little Omega.

The function $f(n) = \omega (g(n))$

iff

$$\lim_{n \to \infty} \frac{f(n)}{g(n)} = 0$$

18. Write algorithm using iterative function to fine sum of n numbers.

```
Algorithm sum(a,n)
{       S : = 0.0
        For i=1 to n do
                S : - S + a[i];
Return S;
}
```

19. Write an algorithm using Recursive function to fine sum of n numbers,

```
Algorithm Rsum (a,n)
{
If(n≤ 0) then
        Return 0.0;
Else    Return Rsum(a, n- 1) + a(n);
```

20. Define the divide an conquer method.

Given a function to compute on 'n' inputs the divide-and-comquer strategy suggests splitting the inputs in to'k' distinct susbsets, 1<k <n, yielding 'k' subproblems. The subproblems must be solved, and then a method must be found to combine subsolutions into a solution of the whole. If the subproblems are still relatively large, then the divide-and conquer strategy can possibly be reapplied.

21. Define control abstraction.

A control abstraction we mean a procedure whose flow of control is clear but whose primary operations are by other procedures whose precise meanings are left undefined.

22. Write the Control abstraction for Divide-and conquer.

```
Algorithm DAnd(ρ )
{
if small(p) then return S(ρ );
else
{
     divide P into smaller instance ρ 1, ρ 2, ρ k, k ≥ 1;
    Apply D and C to each of these subproblems
    Return combine (DAnd C(ρ1) DAnd C(ρ2),----, DAnd ((ρk));
    }
}
```

23. What is the substitution method?

One of the methods for solving any such recurrence relation is called the substitution method.

24. What is the binary search?

If 'q' is always chosen such that 'aq' is the middle element(that is, q=[(n+1)/2), then the resulting search algorithm is known as binary search.

25. Give computing time for Bianry search?

In conclusion we are now able completely describe the computing time of binary search by giving formulas that describe the best, average and worst cases.

Successful searches

θ (1) θ (logn) θ (Logn)

best average worst

unsuccessful searches

θ (logn)

best, average, worst

26. Define external path length?

The external path length E, is defines analogously as sum of the distance of all external nodes from the root.

27. Define internal path length.

The internal path length 'I' is the sum of the distances of all internal nodes from the root.

28. What is the maximum and minimum problem?

The problem is to find the maximum and minimum items in a set of 'n' elements. Though this problem may look so simple as to be contrived, it allows us to demonstrate divide-and-comquer in simple setting.

29. What is the Quick sort?

n quicksort, the division into subarrays is made so that the sorted subarrays do not need to be merged later.

30. Write the Anlysis for the Quick sot.

In analyzing QUICKSORT, we can only make the number of element comparisions c(n). It is easy to see that the frequency count of other operations is of the same order as C(n).

31. Is insertion sort better than the merge sort?

Insertion sort works exceedingly fast on arrays of less then 16 elements, though for large 'n' its computing time is $O(n^2)$.

32. Write a algorithm for straightforward maximum and minimum>

```
        algorithm straight MaxMin(a,n,max,min)
//set max to the maximum and min to the minimum of a[1:n]
{
        max := min: = a[i];
        for i = 2 to n do
{
        if(a[i] >max) then max: = a[i];
        if(a[i] >min) then min: = a[i];


}
}
```

33. Give the recurrence relation of divide-and-conquer?

The recurrence relation is

$$T(n) = \begin{cases} g(n) \\ T(n_1) + T(n_2) + \text{----+ } T(n_k) + f(n) \end{cases}$$

34. Write the algorithm for Iterative binary search?

```
        Algorithm BinSearch(a,n,x)
    //Given an array a[1:n] of elements in nondecreasing
    // order, n>0, determine whether x is present
{
    low : = 1;
    high : = n;
    while (low <  high) do
{
    mid : = [(low+high)/2];
    if(x < a[mid]) then high:= mid-1;
    else if (x >a[mid]) then low:=mid + 1;
        else return mid;
    }
    return 0;
}
```

35. What are internal nodes?

The circular node is called the internal nodes.

36. Describe the recurrence relation ofr merge sort?

If the time for the merging operation is proportional to n, then the computing time of merge sort is described by the recurrence relation

$$T(n) = \begin{cases} a & n = 1, \text{ a a constant} \\ 2T(n/2) + n & n > 1, \text{ c a constant} \end{cases}$$

37.What is meant by feasible solution?

Given n inputs and we are required to form a subset such that it satisfies some given constraints then such a subset is called feasible solution.

38. Write any two characteristics of Greedy Algorithm?

* To solve a problem in an optimal way construct the solution from given set of candidates.

* As the algorithm proceeds, two other sets get accumulated among this one set contains the candidates that have been already considered and chosen while the other set contains the candidates that have been considered but rejected.

39. Define optimal solution?

A feasible solution either maximizes or minimizes the given objective function is called as optimal solution

40. What is Knapsack problem?

A bag or sack is given capacity n and n objects are given. Each object has weight w_i and profit p_i .Fraction of object is considered as x_i (i.e) $0<=x_i<=1$ .If fraction is 1 then entire object is put into sack. When we place this fraction into the sack we get w_ix_i and p_ix_i.

41. Define weighted tree.

A directed binary tree for which each edge is labeled with a real number (weight) is called as weighted tree.

42. What is the use of TVSP?

In places where the loss exceeds the tolerance level boosters have to the placed. Given a network and loss tolerance level the tree vertex splitting problems is to determine an optimal placement of boosters.

43. What is the Greedy choice property?

* The first component is greedy choice property (i.e.) a globally optimal solution can arrive at by making a locally optimal choice.
* The choice made by greedy algorithm depends on choices made so far but it cannot depend on any future choices or on solution to the sub problem.
* It progresses in top down fashion.

44. What is greedy method?

Greedy method is the most important design technique, which makes a choice that looks best at that moment. A given 'n' inputs are required us to obtain a subset that satisfies some constraints that is the feasible solution. A greedy method suggests that one can device an algorithm that works in stages considering one input at a time.

45. What are the steps required to develop a greedy algorithm?

* Determine the optimal substructure of the problem.
* Develop a recursive solution.
* Prove that at any stage of recursion one of the optimal choices is greedy choice. Thus it is always safe to make greedy choice.
* Show that all but one of the sub problems induced by having made the greedy choice are empty.
* Develop a recursive algorithm and convert into iterative algorithm.

46. What is activity selection problem?

The 'n' task will be given with starting time s_i and finishing time f_i. Feasible Solution is that the task should not overlap and optimal solution is that the task should be completed in minimum number of machine set.

47. Write the specification of TVSP

Let T= (V, E, W) be a weighted directed binary tree where

V vertex set
E edge set
W weight function for the edge.

W is more commonly denoted as w (i,j) which is the weight of the edge $<i,j> \varepsilon$ E.

48. Define forest.

Collection of sub trees that are obtained when root node is eliminated is known as forest

49. .What does Job sequencing with deadlines mean?

* Given a set of 'n' jobs each job 'i' has a deadline d_i such that $d_i>=0$ and a profit p_i such that $p_i>=0$.
* For job 'i' profit p_i is earned iff it is completed within deadline.
* Processing the job on the machine is for 1unit of time. Only one machine is available.

50. Define post order traversal.

The order in which the TVSP visits the nodes of the tree is called the post order traversal.

51. Write the formula to calculate delay and also write the condition in which the node gets splitted?

To calculate delay

$$d(u)=\max\{d(v)+w(u, v)\}$$
$$v \varepsilon c(u)$$

The condition in which the node gets splitted are:

$d(u)+w(u ,v)>\delta$ and also d(u) is set to zero.

52. What is meant by tolerance level?

The network can tolerate the losses only up to a certain limit tolerance limit.

53. When is a task said to be late in a schedule?

A task is said to be late in any schedule if it finishes after its deadline else a task is early in a schedule.

54. Define feasible solution for TVSP.

Given a weighted tree T(V,E,W) and a tolerance limit δ any subset X of V is a feasible solution if $d(T/X)<= \delta$.

55. Define optimal solution for TVSP.

An optimal solution is one in which the number of nodes in X is minimized.

56. Write the general algorithm for Greedy method control abstraction.

```
Algorithm Greedy (a, n)
{
  solution=0;
  for i=1 to n do
  {
    x= select(a);
    if feasible(solution ,x) then
       solution=Union(solution ,x);
   }
    return solution;
 }
```

57. . Define optimal solution for Job sequencing with deadlines.

Feasible solution with maximum profit is optimal solution for Job sequencing with deadlines.

58.Write the difference between the Greedy method and Dynamic programming.

Greedy method	Dynamic programming
1.Only one sequence of decision is generated.	1.Many number of decisions are generated.
2.It does not guarantee to give an optimal solution always.	2.It definitely gives an optimal solution always.

59.Define dynamic programming.

Dynamic programming is an algorithm design method that can be used when a solution to the problem is viewed as the result of sequence of decisions.

60.What are the features of dynamic programming?

Optimal solutions to sub problems are retained so as to avoid recomputing their values.
Decision sequences containing subsequences that are sub optimal are not considered.
It definitely gives the optimal solution always.

61.What are the drawbacks of dynamic programming?

Time and space requirements are high, since storage is needed for all level.
Optimality should be checked at all levels.

62.Write the general procedure of dynamic programming.

The development of dynamic programming algorithm can be broken into a sequence of 4 steps.

1. Characterize the structure of an optimal solution.
2. Recursively define the value of the optimal solution.
3. Compute the value of an optimal solution in the bottom-up fashion.
4. Construct an optimal solution from the computed information.

63.Define principle of optimality.

It states that an optimal sequence of decisions has the property that whenever the initial stage or decisions must constitute an optimal sequence with regard to stage resulting from the first decision.

64.Give an example of dynamic programming and explain.

An example of dynamic programming is knapsack problem. The solution to the knapsack problem can be viewed as a result of sequence of decisions. We have to decide the value of x_i for $1 \leq i \leq n$. First we make a decision on x_1 and then on x_2 and so on. An optimal sequence of decisions maximizes the object function $\sum p_i x_i$.

65.Write about optimal merge pattern problem.

Two files x_1 and x_2 containing m & n records could be merged together to obtain one merged file. When more than 2 files are to be merged together. The merge can be accomplished by repeatedly merging the files in pairs. An optimal merge pattern tells which pair of files should be merged at each step. An optimal sequence is a least cost sequence.

66.Explain any one method of finding the shortest path.

One way of finding a shortest path from vertex i to j in a directed graph G is to decide which vertex should be the second, which is the third, which is the fourth, and so on, until vertex j is reached. An optimal sequence of decisions is one that results in a path of least length.

67.Define 0/1 knapsack problem.

The solution to the knapsack problem can be viewed as a result of sequence of decisions. We have to decide the value of x_i. x_i is restricted to have the value 0 or 1 and by using the function knap(l, j, y) we can represent the problem as maximum $\Sigma\ p_i\ x_i$ subject to $\Sigma\ w_i\ x_i \leq y$ where l - iteration, j - number of objects, y – capacity.

68.What is the formula to calculate optimal solution in 0/1 knapsack problem?

The formula to calculate optimal solution is

$$g_0(m)=\max\{g_1,\ g_1(m-w_1)+p_1\}.$$

69.Write about traveling salesperson problem.

Let g = (V, E) be a directed. The tour of G is a directed simple cycle that includes every vertex in V. The cost of a tour is the sum of the cost of the edges on the tour. The traveling salesperson problem to find a tour of minimum cost.

70.Write some applications of traveling salesperson problem.

Routing a postal van to pick up mail from boxes located at n different sites.

Using a robot arm to tighten the nuts on some piece of machinery on an assembly line.

Production environment in which several commodities are manufactured on the same set of machines.

71.Give the time complexity and space complexity of traveling salesperson problem.

Time complexity is $O(n^2\ 2^n)$.

Space complexity is $O(n\ 2^n)$.

72.Define flow shop scheduling.

The processing of jobs requires the performance of several distinct job. In flow shop we have n jobs each requiring n tasks i.e. T_{1i}, T2i,…...T_{mi}, $1 \leq i \leq n$.

73.What are the conditions of flow shop scheduling?

Let T_{ji} denote jth task of the ith job. Task T_{ij} is to be performed on P_j number of processors where $1 \leq j \leq m$ i.e. number of processors will be equal to number of task

Any task T_{ji} must be assigned to the processor P_j.

No processor can have more than one task assigned to it at any time. For any job I processing the task for j>1 cannot be started until T(j-i),i has been completed.

74.Define nonpreemptive schedule.

A nonpreemptive schedule is a schedule in which the processing of a task on any processor is not terminated until the task is complete.

75. Define preemptive schedule.

A preemptive schedule is a schedule in which the processing of a task on any processor can be terminated before the task is completed.

76.Define finish time

The finish time f_i (S) of job i is the time at which all tasks of job i have been completed in schedule S.The finish time F(S) of schedule S is given by

$$F(S)=\max\{ f_i (S)\} \quad {}^{1\le i\le n}$$

77.Define mean flow time

The mean flow time MFT (S) is defined to be]

$$MFT (S) = \frac{1}{n} \sum_{1\le i\le n} fi(S)$$

78.Define optimal finish time.

Optimal finish time scheduling for a given set of tasks is a nonpreemptive schedule S for which F (S) is minimum over all nonpreemptive schedules S.

79.Define preemptive optimal finish time.

Preemptive optimal finish time scheduling for a given set of tasks is a preemptive schedule S for which F (S) is minimum over all preemptive schedules S.

80. What are the requirements that are needed for performing Backtracking?

To solve any problem using backtracking, it requires that all the solutions satisfy a complex set of constraints. They are:

i. Explicit constraints.
ii. Implicit constraints.

81.Define explicit constraint.

They are rules that restrict each x_i to take on values only from a give set. They depend on the particular instance I of the problem being solved. All tuples that satisfy the explicit constraints define a possible solution space.

82. Define implicit constraint.

They are rules that determine which of the tuples in the solution space of I satisfy the criteria function. It describes the way in which the x_i must relate to each other.

83.Define state space tree.

The tree organization of the solution space is referred to as state space tree.

84.Define state space of the problem.

All the paths from the root of the organization tree to all the nodes is called as state space of the problem

85.Define answer states.

Answer states are those solution states s for which the path from the root to s defines a tuple that is a member of the set of solutions of the problem.

86.What are static trees?

The tree organizations that are independent of the problem instance being solved are called as static tree.

87.What are dynamic trees?

The tree organizations those are independent of the problem instance being solved are called as static tree.

88.Define a live node.

A node which has been generated and all of whose children have not yet been generated is called as a live node.

89. Define a E – node.

E – node (or) node being expanded. Any live node whose children are currently being generated is called as a E – node.

90.Define a dead node.

Dead node is defined as a generated node, which is to be expanded further all of whose children have been generated.

91.,What are the factors that influence the efficiency of the backtracking algorithm?

The efficiency of the backtracking algorithm depends on the following four factors. They are:

i. The time needed to generate the next x_k
ii. The number of x_k satisfying the explicit constraints.
iii. The time for the bounding functions B_k
iv. The number of x_k satisfying the B_k.

92.Define Branch-and-Bound method.

The term Branch-and-Bound refers to all the state space methods in which all children of the E-node are generated before any other live node can become the E- node.

93.What are the searching techniques that are commonly used in Branch-and-Bound method.

The searching techniques that are commonly used in Branch-and-Bound method are:

i. FIFO
ii. LIFO
iii. LC
iv. Heuristic search

94.State 8 – Queens problem.

The problem is to place eight queens on a 8 x 8 chessboard so that no two queen "attack" that is, so that no two of them are on the same row, column or on the diagonal.

95.State Sum of Subsets problem.

Given n distinct positive numbers usually called as weights , the problem calls for finding all the combinations of these numbers whose sums are m.

96. State m – colorability decision problem.

Let G be a graph and m be a given positive integer. We want to discover whether the nodes of G can be colored in such a way that no two adjacent nodes have the same color yet only m colors are used.

97.Define chromatic number of the graph.

The m – colorability optimization problem asks for the smallest integer m for which the graph G can be colored. This integer is referred to as the chromatic number of the graph.

98. Define a planar graph.

A graph is said to be planar iff it can be drawn in such a way that no two edges cross each other.

99. What are NP- hard and Np-complete problems?

The problems whose solutions have computing times are bounded by polynomials of small degree.

100. What is a decision problem?

Any problem for which the answer is either zero or one is called decision problem.

101. What is maxclique problem?

A maxclique problem is the optimization problrm that has to determine the size of a largest clique in Grapg G where clique is the maximal subgraph of a graph.

102. what is approximate solution?

A feasible solution with value close to the value of an optimal solution is called approximate solution.