EASWARI ENGINEERING COLLEGE
Department of Information Technology

Question Bank

Subject code : CS6402 Degree / Branch : B.Tech. (IT)

Subject Name : Design and Analysis Of Algorithm Year/ Sem / Sec: II / IV /A & B sec

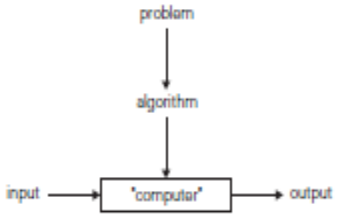
Faculty : Mijula Navis.J

UNIT I
INTRODUCTION

1. What is an algorithm?

   An *algorithm* is a sequence of unambiguous instructions for solving a problem, i.e., for obtaining a required output for any legitimate input in a finite amount of time.

2. What is the notion of algorithm?



3. Write the Euclids algorithm to compute GCD of two numbers
   Euclid's algorithm for computing gcd*(m, n)*
   Step 1 If $n = 0$, return the value of m as the answer and stop; otherwise, proceed to Step 2.
   Step 2 Divide m by n and assign the value of the remainder to r.
   Step 3 Assign the value of n to m and the value of r to n. Go to Step 1.

4. Write the psedocode to compute GCD of two numbers
   ALGORITHM *Euclid(m, n)*
   //Computes gcd*(m, n)* by Euclid's algorithm
   //Input: Two nonnegative, not-both-zero integers m and n
   //Output: Greatest common divisor of m and n
   while n _= 0 do
   $r \leftarrow m \bmod n$
   $m \leftarrow n$
   $n \leftarrow r$
   return m

5. List the steps to solve a given problem
   1. Understand the problem
   2. Decide on:computational means,exact vs. approximate solving,algorithm design technique
   3. Design an algorithm
   4. Prove correctness
   5. Analyze the algorithm
   6. Code the algorithm

6. List the important problem types
   Sorting
   Searching
   String processing
   Graph problems
   Combinatorial problems
   Geometric problems
   Numerical problems

7. Explain about the enhanced version of sequential search.

   Sequential search simply compares successive elements of a list with a given search key until either a match is encountered or the list is exhausted without finding a match. The enhancement in this version is to append the search key to the end of the list , then the search for the key will have to be successful & so we can eliminate a check for the list's end on each iteration.

8. What is the improvement that can be applied to sequential search if the list is sorted?

   The straightforward improvement that can be incorporated in sequential search if a given list is known to be sorted is that searching in the list can be stopped as soon as an element greater than or equal to the search key is encountered.

9. Define brute force string matching.

   The brute force string matching has a given string of n characters called the text and a string of m characters called the pattern, find a substring of the text that matches the pattern. And find the index I of the leftmost character of the first matching substring in the text.

10. What are the types of efficiency

    Time efficiency is defined as how fast the algorithm runs and space efficiency is defined as the amount of extra memory space the algorithm occupies.

11. List the asymptotic notations with respect to efficiency

    1. Best case efficiency denoted by Big omega
    2. Worst case efficiency denoted by Big oh
    3. Average case efficiency denoted by Big theta

12. Define Big oh notation May/June 2006, April/May 2008

    A function $t(n)$ is said to be in $O(g(n))$, denoted $t(n) \in O(g(n))$,
    if $t(n)$ is bounded above by some constant multiple of $g(n)$ for all large n, i.e., if there exist some positive constant c and some nonnegative integer $n0$ such that
    $t(n) \leq cg(n)$ for all $n \geq n0$.

13. Define Big Omega notation

    A function $t(n)$ is said to be in $\Omega(g(n))$, denoted $t(n) \in \Omega(g(n))$, if
    $t(n)$ is bounded below by some positive constant multiple of $g(n)$ for all large n,
    i.e., if there exist some positive constant c and some nonnegative integer $n0$ such that
    $t(n) \geq cg(n)$ for all $n \geq n0$.

14. Define Big Theta notation

A function $t(n)$ is said to be in $\Theta(g(n))$, denoted $t(n) \in \Theta(g(n))$, if $t(n)$ is bounded both above and below by some positive constant multiples of $g(n)$ for all large n, i.e., if there exist some positive constants $c1$ and $c2$ and some nonnegative integer $n0$ such that
$c2g(n) \leq t(n) \leq c1g(n)$ for all $n \geq n0$.

15. What is the general plan for analysis of non recursive algorithms
1.Decide on a parameter (or parameters) indicating an input's size.
2. Identify the algorithm's basic operation. (As a rule, it is located in the innermost loop.)
3. Check whether the number of times the basic operation is executed depends only on the size of an input. If it also depends on some additional property, the worst-case, average-case, and, if necessary, best-case efficiencies have to be investigated separately.
4. Set up a sum expressing the number of times the algorithm's basic operation is executed.4
5. Using standard formulas and rules of sum manipulation, either find a closed form formula for the count or, at the very least, establish its order of growth.

16. What are Sequential Algorithms?

The central assumption of the RAM model is that instructions are executed one after another, one operation at a time. Accordingly, algorithms designed to be executed on such machines are called Sequential algorithms.

17. What are Parallel Algorithms?

The central assumption of the RAM model does not hold for some newer computers that can execute operations concurrently, i.e., in parallel algorithms that take advantage of this capability are called Parallel algorithms.

18. What is Exact and Approximation algorithm?

The principal decision to choose solving the problem exactly is called exact algorithm.

The principal decision to choose solving the problem approximately is called Approximation algorithm.

19. What is Algorithm Design Technique? Nov/Dec 2005

An algorithm design technique is a general approach to solving problems algorithmically that is applicable to a variety of problems from different areas of computing.

20. Define Pseudo code.

A Pseudo code is a mixture of a natural language and programming language like constructs. A pseudo code is usually more precise than a natural language, and its usage often yields more succinct algorithm descriptions.

21. Define Flowchart.

A method of expressing an algorithm by a collection of connected geometric shapes containing descriptions of the algorithm's steps.

22. Explain Algorithm's Correctness

To prove that the algorithm yields a required result for every legitimate input in a finite amount of time.

Example: Correctness of Euclid's algorithm for computing the greatest common divisor stems from correctness of the equality gcd (m, n) = gcd (n, m mod n).

23. What do you mean by "Amortized efficiency"?

The "Amortized efficiency" applies not only a single run of an algorithm but rather to a sequence of operations performed on the same data structure. It turns out that in some situations a single operation can be expensive ,but the total time for an entire sequence of n such operations is always significantly better than the worst case efficiency of that single operation multiplied by n. This is known as "Amortized efficiency".

24. Define order of growth.

The efficiency analysis framework concentrates on the order of growth of an algorithm's basic operation count as the principal indicator of the algorithm's efficiency. To compare and rank such orders of growth we use three notations

i. O (Big oh) notation
ii. Ω (Big Omega) notation &
iii. Θ (Big Theta) notation

PART - B

1. (a) Describe the steps in analyzing & coding an algorithm. (10)
   (b) Explain some of the problem types used in the design of algorithm. (6)

2. (a) Discuss the fundamentals of analysis framework . (10)
   (b) Explain the various asymptotic notations used in algorithm design. (6)

3. (a) Explain the general framework for analyzing the efficiency of algorithm. (8)
   (b) Explain the various Asymptotic efficiencies of an algorithm. (8)

4. (a) Explain the basic efficiency classes. (10)
   (b) Explain briefly the concept of algorithmic strategies. (6)

5. Describe briefly the notions of complexity of an algorithm. (16)

6. (a) What is Pseudo-code?Explain with an example. (8)
   (b) Find the complexity C(n) of the algorithm for the worst case, best case and average case.(Evaluate average case complexity for n=3,Where n is the number of inputs) (8)

7. Set up & solve a recurrence relation for the number of key comparisons made by above pseudo code. (10)

UNIT –II
BRUTE FORCE AND DIVIDE-AND-CONQUER
Part - A

1) Explain divide and conquer algorithms

Divide and conquer is probably the best known general algorithm design technique. It work according to the following general plan

i)A problem's instance is divided into several smaller instances of the same problem, ideally of about the same size.

ii) The smaller instances are solved

iii) If necessary, the solutions obtained for the smaller instances are combined to get a solution to the original problem

2) Define Merge Sort

Merge sort is a perfect example of a successful application of the divide and conquer technique. It sorts a given array A[0...n-l] by dividing it into two halves A[0...[n/2] - 1] and A[[n/2]....n-l], sorting each of them recursively, and then merging the two smaller sorted arrays into a single sorted one.

3) Define Binary Search

Binary Search is remarkably efficient algorithm for searching in a sorted array. It works by comparing a search key K with the array's middle element A[m]. If they match, the algorithm stops; Otherwise, the same operation is repeated recursively for the first half of the array if K< A[m] and for the second half if K > A[m]

A[0].....A[m-l] A[m] A[m+l]..... A[n-l]

4) What can we say about the average case efficiency of binary search?

Asophisticated analysis shows that the average number of key comparisons made by binary search is only slightly smaller than that in the worst case

$C_{avg}(n) = \log_2 n$

5) Define Binary Tree

Abinary tree T is defined as a finite set of nodes that is either empty or consists of s root and two disjoint binary trees TL, and T_R called, respectively the left and right subtree of the root.

6) How divide and conquer technique can be applied to binary trees?

Since the binary tree definition itself divides a binary tree into two smaller structures of the same type, the left subtree and the right subtree, many problems about binary trees can be solved by applying the divide-conquer technique.

7) Explain Internal and External Nodes

To draw the tree's extension by replacing the empty subtrees by special nodes.The extra nodes shown by little squares are called external. The original nodes shown by littile circles are called internal.

8) Define Preorder, inorder and postorder Traversal

PREORDER -The root is visited before the left and right subtrees are visited (in that order)

INORDER -The root is visited after visiting its left subtree but before visiting the right Subtree

POSTORDER - The root is visited after visiting the left and right subtrees(in that order)

9) Define the Internal Path Length

The Internal Path Length I of an extended binary tree is defined as the sum of the lengths of the paths - taken over all internal nodes- from the root to each internal node.

10) Define the External Path Length

The External Path Length E of an extended binary tree is defined as the sum of the lengths of the paths - taken over all external nodes- from the root to each external node.

11) Define closest pair problem

The closest pair problem finds the two elements or points in a set that are closest to each other in a plane. The simplest of a variety of problems in computational geometry that deals with proximity of points in a plane or higher dimensional spaces. Points represents physical objects such as airplane, post offices as well as data base records.

12) Define convex hull problem

The smallest polygon obtained from the set of planes which covers all the points in the plane is the convex hull probem. The simplest of a variety of problems in computational geometry that deals with proximity of points in a plane or higher dimensional spaces. Points represents physical objects such as airplane, post offices as well as data base records.

Part - B

1) Write a pseudo code for divide & conquer algorithm for merging two sorted arrays in to a single sorted one.Explain with example. (16)
2) Explain travelling Salesman Problem with an example(16)
3) Explain the convex hull and closest pair problem with an example(16)
4) Explain the algorithm of merge sort and quick sort with its efficiency(16)
5) Describe Binary search with an example(8)
6) Describe strassen's matrix multiplication in detail (10)

UNIT-III
DYNAMIC PROGRAMMING AND GREEDY TECHNIQUE

Part - A

1)Define Dynamic Programming

Dynamic programming is a technique for solving problems with overlapping problems. Typically, these subproblems arise from a recurrence relating a solution to a given problem with solutions to its smaller subproblems of the same type. Rather than solving overlapping subproblems again and again, dynamic programming suggests solving each of the smaller sub problems only once and recording the results in a table from which we can then obtain a solution to the original problem.

2) Define Binomial Coefficient

The Binomial Coefficient, denoted C(n,k) or $\binom{n}{k}$ is the number of Combinations(subsets) of k

elements from an n-element set(0<k<n). The name "binomial coefficients" comes from the participation of these numbers in the so called binomial formula

$(a+b)^n$ =(n,0)a^n+.....+C(n,i)$a^{n-i}b^i$+.....+C(n,n)b^n 3) Define Transitive closure

The transitive closure of a directed graph with n vertices can be defined as the n by n Boolean matrix T = {ti,}, in which the element in the i^{th} row (1<i≤n) and the j^{th} column (l<j≤n) is 1 if there exists a non trivial directed path from the i^{th} vertex to j^{th} vertex ; otherwise , t_{ij} is 0.

4) Explain Warshalls algorithm

Warshall's algorithm constructs the transitive closure of a given digraph with n vertices through a series of n by n Boolean matrices $R^{(0)}$, ……, $R^{(k-l)}R^{(k)}$,……., $R^{(n)}$

Each of these matrices provides certain information about directed paths in the digraph.

4) Explain All-pair shortest-paths problem

Given a weighted connected graph (undirected or directed), the all pairs shortest paths problem asks to find the distances(the lengths of the shortest path) from each vertex to all other vertices.

5) Explain Floyd's algorithm

It is convenient to record the lengths of shortest paths in an n by n matrix D called the distance matrix: the element d_{ij} in the i^{th} row and the jth column of this matrix indicates the length of the shortest path from the i^{th} vertex to the j^{th} vertex . We can generate the distance matrix with an algorithm that is very similar to warshall's algorithm. It is called Floyd's algorithm.

6) What does Floyd's algorithm do?

It is used to find the distances (the lengths of the shortest paths) from each vertex to all other vertices of a weighted connected graph.

7) Explain principle of Optimality

It says that an optimal solution to any instance of an optimization problem is composed of optimal solutions to its subinstances.

8) Explain Optimal Binary Search Trees

One of the principal application of Binary Search Tree is to implement the operation of searching. If probabilities of searching for elements of a set are known, it is natural to pose a question about an optimal binary search tree for which the average number of comparisons in a search is the smallest possible.

9) Explain Knapsack problem

Given n items of known weights $w_1,w_2.....w_n$ and values $v_1,v_2.....v_n$ and a knapsack of capacity W, find the most valuable subset of the items that fit into the knapsack.(Assuming all the weights and the knapsack's capacity are positive integers the item values do not have to be integers.)

10) Explain the Memory Function technique

The Memory Function technique seeks to combine strengths of the top down and bottom-up approaches to solving problems with overlapping subproblems. It does this by solving, in the top-down fashion but only once, just necessary sub problems of a given problem and recording their solutions in a table.

11) Explain Traveling salesman problem"?

A salesman has to travel n cities starting from any one of the cities and visit the remaining cities exactly once and come back to the city where he started his journey in such a manner that either the distance is minimum or cost is minimum. This is known as traveling salesman problem.

12)Explain about greedy technique

The greedy technique suggests constructing a solution to an optimization problem through a sequence of steps, each expanding a partially constructed solution obtained so far, until a complete solution to the problem is reached. On each step, the choice made must be feasible, locally optimal and irrevocable.

13) Define Spanning Tree

ASpanning Tree of a connected graph is its connected acyclic subgraph(i.e., a tree) that contains all the vertices of the graph.

14)Define Minimum Spanning Tree

Aminimum spanning tree of a weighted connected graph is its spanning tree of the smallest weight where the weight of a tree is defined as the sum of the weights on all its edges.

15)Define min-heap

Amin-heap is a complete binary tree in which every element is less than or equal to its children. All the principal properties of heaps remain valid for min-heaps, with some obvious modifications.

16)Define Kruskal's Algorithm

Kruskal's algorithm looks at a minimum spanning tree for a weighted connected graph G =(V,E) as an acyclic subgraph with | V| - 1 edges for which the sum of the edge weights is the smallest.

17) Define Prim's Algorithm

Prim's algorithm is a greedy algorithm for constructing a minimum spanning tree of a weighted connected graph.It works by attaching to a previously constructed subtree a vertex to the vertices already in the tree.

18) Explain Dijkstra's Algorithm

Dijkstra's algorithm solves the single - source shortest - path problem of finding shortest paths from a given vertex (the source) to all the other vertices of a weighted graph or digraph. It works as prim's algorithm but compares path lengths rather than edge lengths. Dijkstra's algorithm always yields a correct solution for a graph with nonnegative weights.

Part - B

1) Solve the all pair shortest path problem for the diagraph with the weighted matrix given below:-a b c d

a
0 ∞ 3 ∞
B
2 0 ∞1
C
∞ 7 0 1
d
6 ∞ ∞ 0 (16)

2) Explain Warshall's & Floyd's Algorithm. (16)

3) Define optimal binary search trees with example. (16)

4) Explain knapsack problem with example.

5).Discuss the solution for travelling salesman problem using branch & bound technique. (16)

6) Construct a minimum spanning tree using Kruskal's algorithm with your own example. (10)

7) Explain Dijikstra algorithm (16)

8) Define Spanning tree. Discuss design steps in Prim's algorithm to construct minimum spanning tree with an example. (16)

9) Explain Kruskal's algorithm. (16)

UNIT – IV ITERATIVE IMPROVEMENT
PART –A

1. Define Optimal solution

The best solution out of n number of feasible solutions with the best value of the objective function

Eg: The shortest Hamiltonian Circuit

The most valuable subset of items that fit the knapsack.

2. Define Feasible Solution

A feasible solution is a point in the problem's search space that satisfies all the problem's constraints.

Ex: A Hamiltonian Circuit in the traveling salesman problem. A subset of items whose total weight does not exceed the knapsack's Capacity.

3. What are the requirements of standard form?

- It must be a maximization problem
- All the constraints must be in the form of linear equations
- All the variables must be required to be nonnegative.

4. What are the steps involved in simplex method?

- Initialization
- Optimality test
- Finding the entering variable
- Finding the departing variable
- Forming the next tableau.

5. Define matching.

A matching in a graph is a subset of its edges with the property that no two edges share a vertex. A maximum matching, more precisely, a maximum cardinality matching, is a matching with the largest number of edges.

6. Define bipartite graph.

All the vertices can be partitioned into two disjoint sets V and U, not necessarily of same size, so that every edge connects a vertex in one of these sets to a vertex in the other set.

7. What is stable marriage problem?

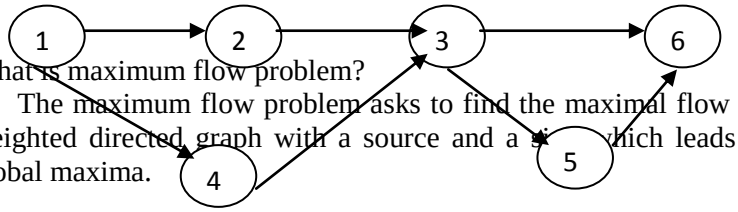
The stable marriage problem is to find a stable matching for elements of two n-element sets based on given matching preferences.

8. Define iterative improvement technique.

This technique involves finding a solution to an optimization problem by generating a sequence of feasible solutions with improving values of the problem's objective function.

9. Define flow network.

A digraph satisfying source, sink and capacity properties is called a flow network or simply a network.



10. What is maximum flow problem?

The maximum flow problem asks to find the maximal flow possible in a network, a weighted directed graph with a source and a sink, which leads to local maxima and a global maxima.

11. Define the Extreme point Theorem

Any linear programming problem with a nonempty bounded feasible region has an optimal solution; moreover, an optimal solution can always be found at an extreme point of the problem's feasible region.

12. Define Flow conservation requirement

The total amount of the material entering an intermediate vertex must be equal to the total amount of the material leaving the vertex. This condition is called the flow-conservation requirement.

13. How Max-flow and Min-cut are related to each other

The value of a maximum flow in a network is equal to the capacity of its minimum cut.

14. What is iterative improvement technique

The *iterative-improvement technique* involves finding a solution to an optimization problem by generating a sequence of feasible solutions with improving values of the problem's objective function. Each subsequent solution in such a sequence typically involves a small, localized change in the previous feasible solution. When no such change improves the value of the objective function, the algorithm returns the last feasible solution as optimal
and stops.

15. Define Simplex method

The *simplex method* is the classic method for solving the general linear programming problem. It works by generating a sequence of adjacent extreme points of the problem's feasible region with improving values of the objective function.

PART –B

1. a) Consider the following linear programming problem in two variables:

maximize $3x + 5y$
subject to $x + y \leq 4$
$x + 3y \leq 6$
$x \geq 0, y \geq 0.$ (16)

2. Explain in detail about an outline of an simplex method with example. (16)

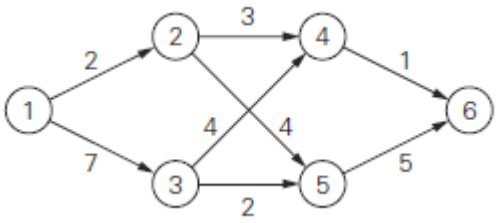
3. Prove the Max-flow Min-cut theorem with example. (16)

4. Write short notes on the following:

i) Flow conservation requirement
ii) Augmenting path method
iii) Shortest augmenting path algorithm
iv) Forward and Backward edges (16)

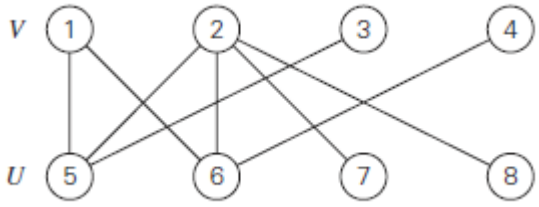
5. Explain how the maximum flow problem for a network with several sources and sinks can be transformed into the same problem for a network with a single source and a single link. (16)

6. Find the local maxima & global maxima using network flow for the given flow graph (16)



7. i) Define the following: source, sink, capacity, flow network and preflow. (10)
ii) Proof a matching M is maximal if and only if there exists no augmenting path with respect to M. (6)

8. Write an algorithm for Maximum Bipartite matching with example.
(16)



9. Write an algorithm for stable marriage algorithm with example. (16)
10. Explain the following:
i) Blocking pair
ii) Stable marriage problem
iii) Man-optimal
iv) Woman-optimal (16)

UNIT – V COPING WITH THE LIMITATIONS OF ALGORITHM POWER
PART –A

1)Explain Backtracking in detail

The principal idea is to construct solutions one component at a time and evaluate such partially constructed candidates as follows.

> If a partially constructed solution can be developed further without violating the problem's constraints, it is done by taking the first remaininig legitimate option for the next component.

> If there is no legitimate option for the next component, no alternatives for any remaining component need to be considered.

In this case, the algorithm backtracks to replace the last component of the partially constructed solution with its next option

2) Explain State Space Tree

If it is convenient to implement backtracking by constructing a tree of choices being made, the tree is called a state space tree. Its root represents an initial state before the search for a solution begins.

3**) Explain promising and nonpromising node**

A node in a state space tree is said to be promising if it corresponds to a partially constructed solution that may still lead to a complete solution;otherwise it is called nonpromising

4**) Explain n-Queens problem**

The problem is to place n queens on an n by n chessboard so that no two queens attack each other by being in the same row or same column or on the same diagonal.

5) **Define tractable and intractable problems**

Problems that can be solved in polynomial time are called tractable problems, problems that cannot be solved in polynomial time are called intractable problems.

A problem's intractability remains the same for all principal models of computations and all reasonable input encoding schemes for the problem under consideration

6) **Explain class P problems**

Class P is a class of decision problems that can be solved in polynomial time by(deterministic)
algorithms. This class of problems is called polynomial.

7) **Explain undecidable problems**

If the decision problem cannot be solved in polynomial time, and if the decision problems cannot be
solved at all by any algorithm. Such problems are called Undecidable.

8) **Explain the halting problem**

Given a computer program and an input to it,determine whether the program will halt on that input or continue working indefinitely on it.

9) **Explain class NP problems**

Class NP is the class of decision problems that can be solved by nondeterministic polynomial algorithms.Most decision problems are in NP. First of all, this class includes all the problems in P. This class of problems is called Nondeterministic polynomial.

1. Write the differences between backtracking & branch and bound techniques.

2. What is NP-hard and NP-completeness?

NP hard: The NP hard problem is a class of problems in computational complexity that is as hard as the hardest problem in NP. If an NP hard problem can be solved in polynomial time then all the NP complete problems can also be solved in polynomial time.

S.No	Backtracking	Branch and Bound
1	Typically decision problems can be solved using backtracking	Typically optimization problems can be solved using branch and bound.
2	While finding the solution to the problem bad choices can be made.	Only better solutions can be made.
3	The state space search tree is searched until the solution is obtained.	The state space search tree needs to be searched completely as there may be chances of being an optimum solution ant where in state space tree.
4	Applications of backtracking are: m-coloring problem, knapsack problem.	Applications of branch and bound are job sequencing and TSP.

Ex: Sum of subset problem, Traveling salesman problem.

NP Completeness: A problem D is called NP-complete if

(i) It belongs to class NP.

(ii) Every problem in NP can also be solved in polynomial time.

Ex: Finding Hamiltonian path.

3. Give the formula for finding the upper bound for the knapsack problem.

The upper bound can be computed using following formula.

$$ub = v + (W-w)(v_{i+1}/w_{i+1})$$

where , v_i is the profit of i^{th} item

w_i is the weight of i^{th} item

W is the capacity of knapsack.

initially v and w are 0.

4. When a decision problem is said to be polynomially reducible

A decision problem Dl is said to be polynomially reducible to a decision problem D2 if there exists a function t that transforms instances of Dl to instances ofD2 such that

i) it maps all yes instances of d1 to yes instances odf d2 and all no instances of dl to no instances ofd2

ii) it is computable by a polynomial time algorithm.

5. What is meant by decision tree?

Many important algorithms, especially those for sorting and searching, work by comparing items of their inputs. We can study the performance of such algorithms with a device called the decision tree.

6. What is subset-sum problem? When can a node be terminate in it?

Let $S = \{S_1, S_2, \ldots S_n\}$ be a set of n positive integers, then we have to find a subset whose sum is equal to given positive integer d. It is always convenient to sort the set's elements in ascending order. That is $S_1 \leq S_2 \leq \ldots.. \leq S_n$

A node can be terminated when it arrives the solution or it prunes.

7. Define Branch and Bound Technique

Compared to backtracking, branch and bound requires. The idea to be strengthened further if we deal with an optimization problem, one that seeks to minimize or maximize an objective function, usually subject to some constraints.

8. Define n-Queens problem

The problem is to place n queens on an n by n chessboard so that no two queens attack each other by being in the same row or same column or on the same diagonal.

9. What is "Traveling salesman problem"?

A salesman has to travel n cities starting from any one of the cities and visit the remaining cities exactly once and come back to the city where he started his journey in such a manner that either the distance is minimum or cost is minimum. This is known as traveling salesman problem.

10. Define Backtracking

The principal idea is to construct solutions one component at a time and evaluate such partially constructed are as follows.

- If a partially constructed solution can be developed further without violating the problem's constraints, it is done by taking the first remaining legitimate option for the next component.
- If there is no legitimate option for the next component, no alternatives for any remaining component need to be considered.

In this case, the algorithm backtracks to replace the last component of the partially constructed solution with its next option.

11. Define tractable and intractable problems

Problems that can be solved in polynomial time are called tractable problems, problems that cannot be solved in polynomial time are called intractable problems.

12. Define the theory of computational complexity

A problem's intractability remains the same for all principal models of computations and all reasonable input encoding schemes for the problem under consideration.

13. Define undecidable problems

If the decision problem cannot be solved in polynomial time, and if the decision problems cannot be solved at all by any algorithm. Such problems are called Undecidable.

14. Define the halting problem

Given a computer program and an input to it,determine whether the program will halt on that input or continue working indefinitely on it.

15. Define "graph coloring" problem.

The graph coloring problem asks us to assign the minimum number of colors to vertices of a graph so that no two adjacent vertices are the same color.

Define Knapsack Problem

Find the most valuable subset of n items of given positive integer weights and values that fit into a knapsack of a given positive integer capacity.

16. What is articulation point.

A vertex v in a connected graph G is an articulation point if and only if the deletion of vertex v
together with all edged incident to v disconnects the graph in to two or more nonempty

components.

PART –B

1. Explain the 8-Queen's problem & discuss the possible solutions. (16)
2. Solve the following instance of the knapsack problem by the branch & bound algorithm .(16)
3. Apply backtracking technique to solve the following instance of subset sum problem: S={1,3,4,5} and d=11

(16)

4. Solve the following 6 city traveling salesperson problem using the branch and bound algorithm. (16)

$$\begin{pmatrix} \alpha & 21 & 42 & 31 & 6 & 24 \\ 11 & \alpha & 17 & 7 & 35 & 18 \\ 25 & 5 & \alpha & 27 & 14 & 9 \\ 12 & 9 & 24 & \alpha & 30 & 12 \\ 14 & 7 & 21 & 15 & \alpha & 48 \\ 39 & 15 & 16 & 5 & 20 & \alpha \end{pmatrix}$$

5. Explain how branch and bound technique is used to solve 0/1 knapsack problem. for n=4 , W=10, (p_1,p_2,p_3,p_4) = (40,42,25,12) and (w_1,w_2,w_3,w_4) = (4,7,5,3). (16)

6. Briefly explain NP-Hard and NP-Completeness with examples. (16)
7. Explain about assignment problem using branch and bound with example. (16)
8. Discuss the solution for travelling salesman problem using branch and bound technique (16)
9. Discuss the decision trees for sorting algorithms. (16)
10. Write a pseudocode for the following:
    i) Approximation algorithms of the knapsack problem. (8)
    ii) Approximation algorithms of the traveling salesman problem. (8)

Prepared by

Mijula Navis J

Approved by

HoD/IT