Design and Analysis of Algorithms
M. Tech Computer Science, IIT Bhilai

(Tutorial Problem sheet)

Greedy Algorithm

1. Not just any greedy approach to the activity-selection problem produces a maximum-size set of mutually compatible activities. Give an example to show that the approach of selecting the activity of least duration from among those that are compatible with previously selected activities does not work.

2. Give an example to show that the approach of always selecting the compatible activity that overlaps the fewest other remaining activities does not work.

3. Give an example to show that the approach of always selecting the activity with the earliest start time does not work.

4. Suppose that instead of always selecting the first activity to finish, we instead select the last activity to start that is compatible with all previously selected activities. Describe how this approach is a greedy algorithm, and prove that it yields an optimal solution.

5. Prove that approach of selecting the most valuable item does not work for 0-1 knapsack problem.

6. Prove that a binary tree that is not full cannot correspond to an optimal prefix code.

7. What is an optimal Huffman code for the following set of frequencies, based on the first 8 Fibonacci numbers?

   $a : 1, b : 2, c : 3, d : 5, e : 8, f : 13, g : 21, h : 34$

   Can you generalize your answer to find the optimal code when the frequencies are the first n Fibonacci numbers?

8. Not just any greedy approach to the activity-selection problem produces a maximum-size set of mutually compatible activities. Give one example for each of the following cases to show that the respective strategies does not work.
   (a) select the activity of least duration from among those that are compatible with previously selected activities.
   (b) select the compatible activity that overlaps the fewest other remaining activities.
   (c) select the activity with the earliest start time.

9. Let S be a text of length $|S|$. Prove that in the huffman coding algorithm, if all characters in S occur with frequency less than $\frac{|S|}{3}$, then there can not be any codeword of length 1.

Dynamic Programming

1. Suppose we want to make change for n cents, using the least number of coins of denominations 1, 10, and 25 cents. Give an $O(n)$ dynamic programming algorithm to find the optimal set of change.

2. You are driving from Ahmedabad to Chennai, using one or more rental cars for the trip. Along the way, you will visit cities $c_1 \cdots c_n$ in order (here, c_1 is Ahmedabad and c_n is Chennai). You have a fee schedule $f(i, j)$ that gives the cost of a rental that is picked up in city c_i and dropped off in city c_j , $j > i$. Note that these costs are arbitrary and possibly non-monotonic; for example, it may cost Rs 200 for a rental from c_1 to c_2 but only Rs 100 for a rental from c_1 to c_3. Give an algorithm to choose a set of rentals that minimize the total cost of your trip. Note that you can rent only one car at a time, and that you can never be without a car. Hence, a solution is a set of non-overlapping rentals that span all cities.

3. A palindrome is a string that reads the same from front and back. Any string can be viewed as a sequence of palindromes if we allow a palindrome to consist of one letter. **For Example**: "bobseesanna" can e.g be viewed as being made up of palindromes in the following ways:

   (a) bobseesanna="bob"+"sees"+"anna"

   (b) bobseesanna="bob"+"s"+"ee"+"s"+"anna"

   (c) bobseesanna= "b"+"o"+"b"+"sees" "a"+"nn"+"a"

   We are interested in computing the minimum number of palindromes from which one can construct a given string "s". Describe an $O(n^3)$ algorithm for solving the problem.

4. Give an $O(n^2)$-time algorithm to find the longest monotonically increasing subsequence of a sequence of n numbers X without using LCS for X and $SORT(X)$.

DFS, BFS, MST

1. A connected graph is vertex-biconnected if there is no vertex whose removal disconnects the graph. A connected graph is edge-biconnected if there is no edge whose removal disconnects the graph.

   Give a proof or counterexample for each for the following statements:

   (a) A vertex-biconnected graph is edge-biconnected.

   (b) An edge-biconnected graph is vertex-biconnected.

2. Prove that if G is an undirected connected graph, then each of its edges is either in the depth-first search tree or is a back edge.

3. Suppose G is a connected undirected graph. An edge e whose removal disconnects the graph is called a bridge. Must every bridge e be an edge in a depth-first search tree of G, or can e be a back edge? Give a proof or a counterexample.

4. Let G be a graph and the BFS tree and the DFS tree of G are same. Prove that G is a tree.

5. A bipartite graph is a graph whose vertices can be partitioned into two subsets such that there is no edge between any two vertices in the same subset. Give a linear-time algorithm to determine if a graph G is bipartite.

6. A mother vertex in a directed graph $G = (V, E)$ is a vertex v such that all other vertices G can be reached by a directed path from v.

(a) Give an $O(n+m)$ algorithm to test whether a given vertex v is a mother of G, where $n = |V|$ and $m = |E|$.

(b) Give an O(n + m) algorithm to test whether graph G contains a mother vertex.

7. Describe and analyze an algorithm to compute the maximum-weight spanning tree of a given edge-weighted graph.

8. An articulation vertex of a graph G is a vertex whose deletion disconnects G. Let G be a graph with n vertices and m edges. Give a simple $O(n+m)$ algorithm for finding a vertex of G that is not an articulation vertex, ie whose deletion does not disconnect G.

9. Prove that if the weights of the edges of a connected graph G are distinct, then G has a unique minimum spanning tree.

10. Does either Prim's and Kruskal's algorithm work if there are negative edge weights? Explain why or why not.

11. Is the path between a pair of vertices in a minimum spanning tree necessarily a shortest path between the two vertices in the full graph? Give a proof or a counterexample.

12. Assume that all edges in the graph have distinct edge weights (i.e., no pair of edges have the same weight). Is the path between a pair of vertices in a minimum spanning tree necessarily a shortest path between the two vertices in the full graph? Give a proof or a counterexample.

13. In the minimum product spanning tree problem, the cost of a tree is the product of all the edge weights in the tree, instead of the sum of the weights. You may assume that all edges have positive weight.

    (a) Give a graph whose minimum product spanning tree is different than the minimum weight spanning tree.

    (b) Give an efficient algorithm to compute the minimum product spanning tree. (Hint: think logarithms).

14. Modify Prim's algorithm so that it runs in time O(n log k) on a graph that has only k different edges costs.

NP Completeness

15. Suppose an oracle has given you a magic computer, C, that when given any Boolean formula ϕ in CNF will tell you in one step whether ϕ is satisfiable. Show how to use C to construct an actual assignment of satisfying Boolean values to the variables in any satisfiable formula B. How many calls do you need to make to C in the worst case in order to do this?

16. A Hamiltonian cycle of a graph G is a simple cycle that visits every vertex exactly once. Suppose someone gives you a function $HamCycle(G)$, which, given an undirected graph G, returns true if G has a Hamiltonian cycle and false if G does not. Use this function to create an algorithm that outputs the sequence of vertices that defines a Hamiltonian cycle, or correctly states that G does not have a Hamiltonian cycle.

17. Draw the graph that results from the reduction of vertex cover to Hamiltonian cycle for a cycle with tree nodes a, b, and c. For vertex cover of the cycle, show the corresponding Hamiltonian cycle in the newly constructed graph.

18. The baseball card collector problem is as follows. Given packets $P_1, \cdots P_m$, each of which contains a subset of that year's baseball cards, is it possible to collect all the year's cards by buying at most k packets?
    For example, if the players are {Aaron,Mays, Ruth, Skiena} and the packets are {Aaron, Mays}, {Mays, Ruth}, {Skiena}, {Mays,Skiena}; there does not exist a solution for $k = 2$ but there does for $k = 3$, such as {{Aaron, Mays}, {Mays, Ruth}, {Skiena}} Prove that the baseball card collector problem is NP-hard using a reduction from vertex cover.

19. The Set Partition Problem takes as input a set S of numbers. The question is whether the numbers can be partitioned into two sets A and $\bar{A} = S - A$ such that

$$\sum_{x \in A} x = \sum_{x \in \bar{A}} x$$

    .

    Show that SET-PARTITION is NP-Complete.

20. Prove that the clique, no-clique problem is NP-hard:
    Input: An undirected graph $G = (V, E)$ and an integer k.
    Output: Does G contain a clique of size k and an independent set of size k.

21. Fair 3-coloring: Given a graph $G = (V, E)$, where $|V|$ is a multiple of 3, the Fair 3-coloring problem asks if G can be 3-colored so that there are the same number of nodes of each color ($|V|/3$). Prove that Fair 3-coloring is NP-complete. You can prove this with a reduction from any NP-complete problem, including the standard 3-coloring problem.

1. Suppose we want to make change for n cents, using the least number of coins of denominations 1, 10, and 25 cents. Give an $O(n)$ dynamic programming algorithm to find the optimal set of change.

Soln

Let $M(n)$ denotes the optimal number of coins in the change of n cents.

Then $M(n) = \min \{ M(n-1)+1, M(n-10)+1, M(n-25)+1 \}$

if $n \geq 25$

$= \min \{ M(n-1)+1, M(n-10)+1 \}$

if $n < 25$ & $n \geq 10$

Base Case

$M(n) = n$ for $n \leq 9$

Use the above recursive relation in a for loop to obtain the value of $M(n)$.

2. You are driving from Ahmedabad to Chennai, using one or more rental cars for the trip. Along the way, you will visit cities $c_1 \cdots c_n$ in order (here, c_1 is Ahmedabad and c_n is Chennai). You have a fee schedule $f(i, j)$ that gives the cost of a rental that is picked up in city c_i and dropped off in city c_j , $j > i$. Note that these costs are arbitrary and possibly non-monotonic; for example, it may cost Rs 200 for a rental from c_1 to c_2 but only Rs 100 for a rental from c_1 to c_3. Give an algorithm to choose a set of rentals that minimize the total cost of your trip. Note that you can rent only one car at a time, and that you can never be without a car. Hence, a solution is a set of non-overlapping rentals that span all cities.

soln

Let $G(i,j)$ be the optimal car rental cost from city i to city j. $(j > i)$

Then $G(i,j) = \min_{i<k<j} \{(G(i,k) + G(k,j)), f(i,j)\}$

Base case

$G(i,i) = 0$

$G(i,i+1) = f(i,i+1)$

use the above recursive relation in a for loop to obtain the value of $G(1,n)$

3. A palindrome is a string that reads the same from front and back. Any string can be viewed as a sequence of palindromes if we allow a palindrome to consist of one letter. **For Example**: "bobseesanna" can e.g be viewed as being made up of palindromes in the following ways:

   (a) bobseesanna="bob"+"sees"+"anna"

   (b) bobseesanna="bob"+"s"+"ee"+"s"+"anna"

   (c) bobseesanna= "b"+"o"+"b"+"sees" "a"+"nn"+"a"

   We are interested in computing the minimum number of palindromes from which one can construct a given string "s". Describe an $O(n^3)$ algorithm for solving the problem.

Soln Let $S(i,j)$ denotes the the substring from i-th character to the j-th character of S.

let $T(i,j)$ denotes minimum number of palindrome for $S(i,j)$.

Then $T(i,j) = 1$ if $S(i,j)$ is a palindrome

$$= \min_{i \le k < j} \left(S(i,k) + S(k,j) \right), \text{ otherwise}$$

Base case $T(i,i) = 1 \quad \forall i = 1, 2 \ldots n$

4. Give an $O(n^2)$-time algorithm to find the longest monotonically increasing subsequence of a sequence of n numbers X without using LCS for X and $SORT(X)$.

Soln Let $X = x_1, x_2, \ldots x_n$

Let $P(t)$ denotes the longest monotonically increasing subsequence for $x_1, x_2, \ldots x_t$

then $P(t) = \max_{j<t : x_t > x_j} \{P(j) + 1\}$

Base case $P(1) = 1$.