Bollina. Madhuri
CSE-1

Regdno: 19131A0532

Algorithms and Analysis
Assignment - 2

26/July/2021

Time Complexities :

1. GREEDY METHOD :

a) Knapsack problem:

Sorting is the keystep to arrange profit/weight ratio in decreasing order.

→ Accept the input, find Piwi, Select highest sort (quicksort - O(nlogn)), Based on weights (O(1)) and then sum

Total Time complexity = O(nlogn)

b) Job sequencing with deadlines :

Each job has a deadline and profit associated to it.

→ Since there exists two nested for loops.

Time Complexity = $O(n^2)$

c) Single source shortest path problem :

Time complexity of Dijkstra's Algorithm is $O(V^2)$. Optimising it by using a min-priority queue reduces to $O(V+E\log V)$. Here V refers to no. of vertices and E is no. of edges.

d) Prim's Algorithm:

Time complexity of prim's Algorithm using adjacency matrix is $O(n^2)$ where 'n' indicates no. of vertices. It has complexity of $O(E\log V)$ using binary heap and can be improved using fibonacci Heap to $O(E+\log V)$. Here E → no. of Edges, V → no. of vertices.

e) Kruskal's Algorithm:

For deletion and reheapify, Kruskal's Algorithm takes $O(E\log E)$ time complexity. The value of E can be almost $O(V^2)$. So the Overall time complexity is $O(E\log E)$ or $O(E\log V)$ where E indicates no. of edges and V is no. of vertices.

Regdno: 19131A0532

2. DYNAMIC PROGRAMMING:

a) 0/1 Knapsack problem:

The time complexity of 0/1 knapsack problem is $o(nW)$ where n is the number of items and W is the capacity of the knapsack.

b) Matrix Chain Multiplication:

The time complexity of matrix chain multiplication is $o(n^3)$ deue to three nested for loops.

c) All pairs shortest path problem:

The time complexity of all pairs shortest path problem is $o(V^3)$ where V is the number of vertices in the graph.

d) Optimal Binary Search tree:

The time complexity of OBST is $o(n^4)$ which can be optimised to $o(n^3)$ by pre calculating sum of frequencies instead of calling sum() again and again.

e) Travelling sales person problem:

The time complexity of travelling salesperson algorithm is $o(n^2 \times 2^n)$ where n is the number of nodes. There are atmost $n \times 1^n$ subsets. Each subset requires $o(n)$ time complexity.

f) Reliability Design:

The time complexity of reliability design is $o(2^n)$ using multiplication optimisation function)

g) Transitive Closure of graph:

The time complexity of Transitive closure of a graph is $o(n^3)$ since there exists three nested for loops.

Regdno: 19131A0532

3. BACK TRACKING:

a) N-Queens Problem:

Time complexity of N-Queens problem is $O(n^n)$ since we check every position on mxN board where 'n' is no. of queens.

b) Graph Coloring:

Time complexity of Graph coloring problem is $O(n \cdot m^n)$ as we have maximum of m^n combinations with 'm' colours to be placed in 'n' places. Thus it takes maximum of 'mn' time to check each combination.

c) Hamiltonian Cycle:

Time complexity of Hamiltonian cycle is $O(n^n)$ since it tries for n! ways

d) Sum of subsets:

Time complexity of sum of subsets problem is $O(2^n)$.

4. BRANCH AND BOUND:

a) Travelling Sales Persm Problem:

Time complexity of travelling sales problem is $O(n^2)$ because algorithm has two iterations embedded one inside the other for calculation of reduced cost matrix.

b) 0/1 Knapsack problem:

Time complexity of 0/1 knapsack is $O(2^n)$. Here 2^n solutions would be generated