Introduction

A greedy algorithm is a simple, intuitive algorithm that is used in optimization problems. Greedy algorithms work in stages. In each stage, a decision is made that is good at that point, without bothering about the future. This means that some local best is chosen. It assumes that a local good selection makes for a globally optimal solution.

Elements of the greedy algorithm

The basic properties of greedy algorithms are

- **Greedy choice property:** This property says that the globally optimal solution can be obtained by making a locally optimal solution. The choice made in a greedy algorithm may depend on earlier choices but never on future choices. It iteratively makes one greedy choice after another and reduces the given problem to a smaller one.
- **Optimal substructure:** A problem exhibits optimal substructure if the optimal solution to the problem contains optimal solutions to the subproblems. That means we can solve subproblems and build up the solution to solve the larger problems.

Example: **Fractional Knapsack Problem**: Given item t1, t2, ..., tn (items we might want to carry in a backpack) with associated weights s1, s2, ..., sn and benefit values v1, v2, ..., vn, how can we maximize the total benefit considering we are subject to an absolute weight limit C?

Algorithm:

1. Compute value per size density for each item di = vi/si.
2. Sort each item by its value density.
3. Take as much as possible of the density item not already in the bag.

Time Complexity: O(nlogn) for sorting and O(n) for greedy selection.

Advantages of Greedy Algorithm

The main advantage of the greedy method is that it is straightforward, easy to understand, and easy to code. In greedy algorithms, once we make a decision, we do not have to spend time re-examining the already computed values.

Disadvantages of Greedy Algorithm

The main disadvantage is that for many problems there is no greedy algorithm. This means that in many cases there is no guarantee that making locally optimal improvements gives the optimal global solution. We also need proof of local optimality as to why a particular greedy solution will work.\

Applications of Greedy Algorithm

- Sorting: Selection sort, Topological sort
- Priority Queues: Heap Sort
- Huffman coding Compression algorithm
- Prim's and Kruskal's algorithm
- Shortest path in the weighted graph(Dijkstra's)
- Coin change problem (for Indian currency)
- Fractional knapsack problem
- Disjoint sets: union by size or union by rank
- Job scheduling algorithm
- The greedy technique can be used as an approximation algorithm for complex problems.