Greedy Method

- The Greedy method is a most straight forward design technique which can be applied to a wide variety of problems.
- This algorithm works in steps. In each step it selects the best available options until all options are finished.
- Most of the problems have n inputs and require us to obtain a subset that satisfies some constraints.
- Any subset that satisfies these constraints is called as a feasible solution.
- A feasible solution that either minimizes or maximizes a given objective function is called as Optimal Solution.

Optimization Problem: An optimization problem is the problem of finding the best solution (optimal solution) from all the feasible solutions (practicable of possible solutions). In an optimization problem we are given a set of constraints and an optimization functions. Solutions that satisfy the constraints are called feasible solutions. A feasible solution for which the optimization function has the best possible value is called optimal solution.

Greedy Algorithms

- **Greedy algorithms** are a class of algorithms that make **locally optimal** choices at each step with the hope of finding a **global optimum** solution.
- In these algorithms, decisions are made based on the information available at the current moment without considering the consequences of these decisions in the future.
- The key idea is to select the best possible choice at each step, leading to a solution that may not always be the most optimal but is often good enough for many problems.

(control abstraction – is an abstract algorithm that describes
the essential steps of a greedy algorithm)

Control Abstraction for Greedy Method:

```
Algorithm GreedyMethod (a, n)
{
   // a is an array of n inputs
   Solution: =0;
   for i: =0 to n do
    {
       s: = select (a);
       if (feasible (Solution, s)) then
       {
            Solution: = union (Solution, s);
        }
        else
             reject (); // if solution is not feasible reject it.
    }
    return solution;
}
```

In greedy method there are three important activities.

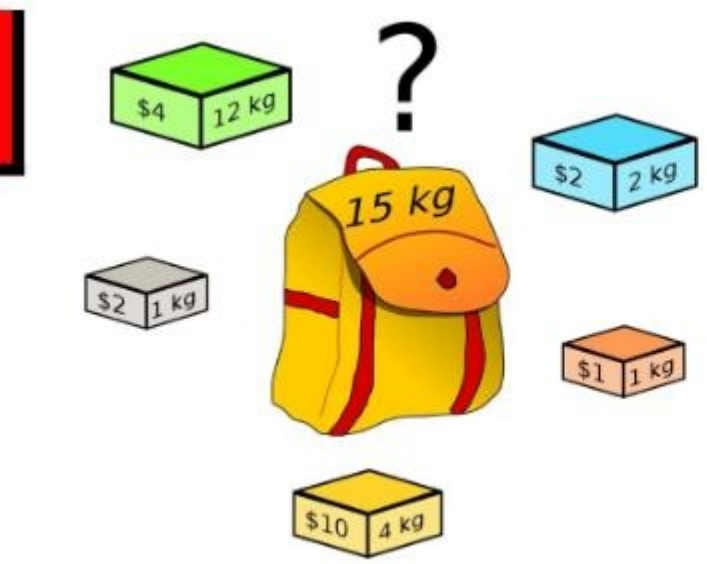
1. A selection of solution from the given input domain is performed, i.e. s:= select(a).
2. The feasibility of the solution is performed, by using feasible ‘(solution, s)and then all feasible solutions are obtained.
3. From the set of feasible solutions, the particular solution that minimizes or maximizes the given objection function is obtained. Such a solution is called optimal solution.

Types of Greedy Algorithms

- **Selection Sort**
- **Knapsack Problem**
- **Minimum Spanning Tree**
- **Single-Source Shortest Path Problem**
- **Job Scheduling Problem**
- **Prim’s Minimal Spanning Tree Algorithm**
- **Kruskal’s Minimal Spanning Tree Algorithm**
- **Dijkstra's Minimal Spanning Tree Algorithm**
- **Huffman Coding**
- **Ford-Fulkerson Algorithm**

Topics Covered In Below Pages

1) Knapsack Problem
2) Job Sequencing With Deadlines
3) Minimum Cost Spanning Trees
4) Single Source Shortest Path



WHAT IS KNAPSACK PROBLEM

The Knapsack Problem is an Optimization Problem in which we have to find an optimal answer among all possible combinations.There are three types of knapsack problems :

1) 0-1 Knapsack,
2) Fractional Knapsack
3) Unbounded Knapsack.

Fractional Knapsack Problem

Follow the below link:

https://www.javatpoint.com/fractional-knapsack-problem

(or)

https://www.tutorialspoint.com/data_structures_algorithms/fractional_knapsack_problem.htm

Programmatic way of Fractional Knapsack link :
https://www.slideshare.net/slideshow/fractional-knapsack-class-13/30522394