Greedy Algorithm

The greedy method is one of the strategies like Divide and conquer used to solve the problems. This method is used for solving optimization problems. An optimization problem is a problem that demands either maximum or minimum results

The Greedy method is the simplest and straightforward approach. it is a technique. The main function of this approach is that the decision is taken on the basis of the currently available information. Whatever the current information is present, the decision is made without worrying about the effect of the current decision in future.

This technique is basically used to determine the feasible solution that may or may not be optimal. The feasible solution is a subset that satisfies the given criteria. The optimal solution is the solution which is the best and the most favourable solution in the subset. In the case of feasible, if more than one solution satisfies the given criteria then those solutions will be considered as the feasible, whereas the optimal solution is the best solution among all the solutions.

Characteristics of Greedy method

The following are the characteristics of a greedy method:

- To construct the solution in an optimal way, this algorithm creates two sets where one set contains all the chosen items, and another set contains the rejected items.
- A Greedy algorithm makes good local choices in the hope that the solution should be either feasible or optimal.

Components of Greedy Algorithm

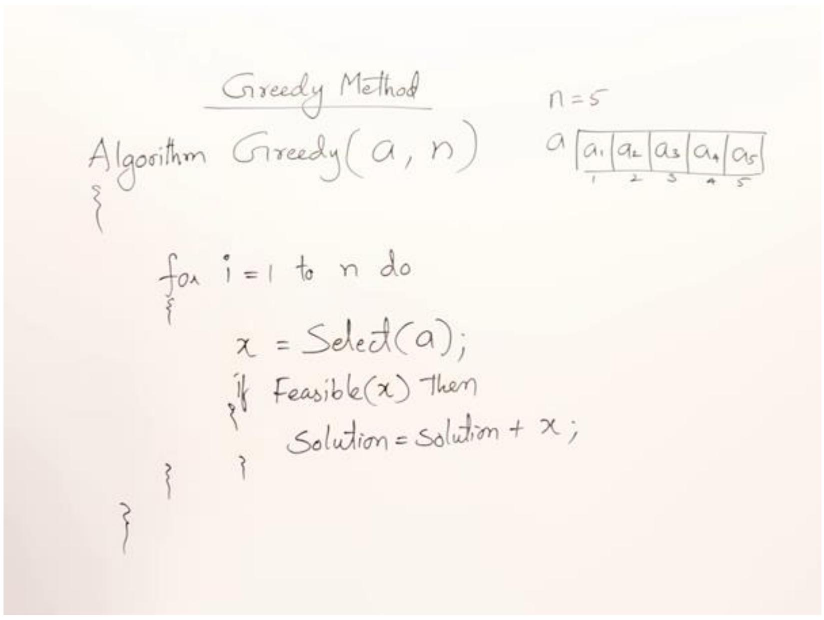
The components that can be used in the greedy algorithm are:

- **Candidate set:** A solution that is created from the set is known as a candidate set.
- **Selection function:** This function is used to choose the candidate or subset which can be added in the solution.
- **Feasibility function:** A function that is used to determine whether the candidate or subset can be used to contribute to the solution or not.
- **Objective function:** A function is used to assign the value to the solution or the partial solution.
- **Solution function:** This function is used to intimate whether the complete function has been reached or not.

Applications of Greedy Algorithm

- It is used in finding the shortest path.
- It is used to find the minimum spanning tree using the prim's algorithm or the Kruskal's algorithm.
- It is used in a job sequencing with a deadline.
- This algorithm is also used to solve the fractional knapsack problem.

Pseudo code of Greedy Algorithm



```
Greedy Method

Algorithm Greedy(a, n)
{
    for i = 1 to n do
    {
        x = Select(a);
        if Feasible(x) Then
        {
            Solution = Solution + x;
        }
    }
}
```

n = 5
a
a1
a2
a3
a4
a5
1
2
3
4
5

```
Algorithm Greedy (a, n)
{
  Solution : = 0;
 for i = 0 to n do
 {
     x: = select(a);
    if feasible(solution, x)
   {
       Solution: = union(solution , x)
   }
     return solution;
} }
```

The above is the greedy algorithm. Initially, the solution is assigned with zero value. We pass the array and number of elements in the greedy algorithm. Inside the for loop, we select the element one by one and checks whether the solution is feasible or not. If the solution is feasible, then we perform the union.

Let's understand through an example.

Suppose there is a problem 'P'. I want to travel from A to B shown as below:

P : A → B

The problem is that we have to travel this journey from A to B. There are various solutions to go from A to B. We can go from A to B by **walk, car, bike, train, aeroplane**, etc. There is a constraint in the journey that we have to travel this journey within 12 hrs. If I go by train or aeroplane then only, I can cover this distance within 12 hrs. There are many solutions to this problem but there are only two solutions that satisfy the constraint.

If we say that we have to cover the journey at the minimum cost. This means that we have to travel this distance as minimum as possible, so this problem is known as a minimization problem. Till now, we have two feasible solutions, i.e., one by train and another one by air. Since travelling by train will lead to the minimum cost so it is an optimal solution. An optimal solution is also the feasible solution, but providing the best result so that solution is the optimal solution with the minimum cost. There would be only one optimal solution.

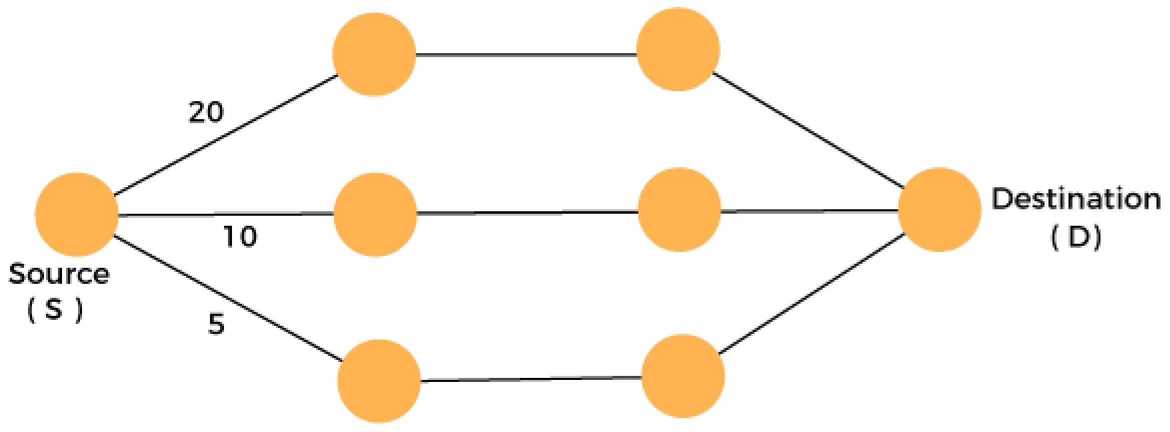
The problem that requires either minimum or maximum result then that problem is known as an optimization problem. Greedy method is one of the strategies used for solving the optimization problems.

Disadvantages of using Greedy algorithm

Greedy algorithm makes decisions based on the information available at each phase without considering the broader problem. So, there might be a possibility that the greedy solution does not give the best solution for every problem.

It follows the local optimum choice at each stage with a intend of finding the global optimum. Let's understand through an example.

Consider the graph which is given below:



We have to travel from the source to the destination at the minimum cost. Since we have three feasible solutions having cost paths as 10, 20, and 5. 5 is the minimum cost path so it is the optimal solution. This is the local optimum, and in this way, we find the local optimum at each stage in order to calculate the global optimal solution.

Is an optimal solution also a feasible solution?
An optimal solution is a feasible solution **where the objective function reaches its maximum (or minimum) value** – for example, the most profit or the least cost. A globally optimal solution is one where there are no other feasible solutions with better objective function values

Fractional Knapsack Problem

The fractional knapsack problem is also one of the techniques which are used to solve the knapsack problem. In fractional knapsack, the items are broken in order to maximize the profit. The problem in which we break the item is known as a Fractional knapsack problem.

This problem can be solved with the help of using two techniques:

- Brute-force approach: The brute-force approach tries all the possible solutions with all the different fractions but it is a time-consuming approach.
- Greedy approach: In Greedy approach, we calculate the ratio of profit/weight, and accordingly, we will select the item. The item with the highest ratio would be selected first.

There are basically three approaches to solve the problem:

- The first approach is to select the item based on the maximum profit.
- The second approach is to select the item based on the minimum weight.
- The third approach is to calculate the ratio of profit/weight.

Consider the below example:

Objects:	1	2	3	4	5	6	7
Profit (P):	5	10	15	7	8	9	4
Weight(w):	1	3	5	4	1	3	2

W (Weight of the knapsack): 15

n (no of items): 7

First + approach:

First approach:

Object	Profit	Weight	Remaining weight
3	15	5	15 - 5 = 10
2	10	3	10 - 3 = 7
6	9	3	7 - 3 = 4
5	8	1	4 - 1 = 3
7	7 * ¾ = 5.25	3	3 - 3 = 0

The total profit would be equal to (15 + 10 + 9 + 8 + 5.25) = 47.25

Second approach:

The second approach is to select the item based on the minimum weight.

Object	Profit	Weight	Remaining weight
1	5	1	15 - 1 = 14
5	8	1	14 - 1 = 13
7	4	2	13 - 2 = 11
2	10	3	11 - 3 = 8
6	9	3	8 - 3 = 5
4	7	4	5 - 4 = 1
3	15 * 1/5 = 3	1	1 - 1 = 0

In this case, the total profit would be equal to (5 + 7 + 4 + 10 + 9 + 7 + 3) = 46

Third approach:

In the third approach, we will calculate the ratio of profit/weight.

Objects:	1	2	3	4	5	6	7
Profit (P):	5	10	15	7	8	9	4
Weight(w):	1	3	5	4	1	3	2
	5	3.33	3	1.7	8	3	2

In this case, we first calculate the profit/weight ratio.

Object 1: 5/1 = 5

Object 2: 10/3 = 3. 33

Object 3: 15/5 = 3

Object 4: 7/4 = 1.7

Object 5: 8/1 = 8

Object 6: 9/3 = 3

Object 7: 4/2 = 2

P:w: 5 3.3 3 1.7 8 3 2

In this approach, we will select the objects based on the maximum profit/weight ratio. Since the P/W of object 5 is maximum so we select object 5.

Object	Profit	Weight	Remaining weight
5	8	1	15 - 8 = 7

After object 5, object 1 has the maximum profit/weight ratio, i.e., 5. So, we select object 1 shown in the below table:

Object	Profit	Weight	Remaining weight
5	8	1	15 - 1 = 14
1	5	1	14 - 1 = 13

After object 1, object 2 has the maximum profit/weight ratio, i.e., 3.3. So, we select object 2 having profit/weight ratio as 3.3.

Object	Profit	Weight	Remaining weight
5	8	1	15 - 1 = 14
1	5	1	14 - 1 = 13
2	10	3	13 - 3 = 10

After object 2, object 3 has the maximum profit/weight ratio, i.e., 3. So, we select object 3 having profit/weight ratio as 3.

Object	Profit	Weight	Remaining weight
5	8	1	15 - 1 = 14
1	5	1	14 - 1 = 13
2	10	3	13 - 3 = 10
3	15	5	10 - 5 = 5

After object 3, object 6 has the maximum profit/weight ratio, i.e., 3. So we select object 6 having profit/weight ratio as 3.

Object	Profit	Weight	Remaining weight
5	8	1	15 - 1 = 14
1	5	1	14 - 1 = 13
2	10	3	13 - 3 = 10
3	15	5	10 - 5 = 5
6	9	3	5 - 3 = 2

After object 6, object 7 has the maximum profit/weight ratio, i.e., 2. So we select object 7 having profit/weight ratio as 2.

Object	Profit	Weight	Remaining weight
5	8	1	15 - 1 = 14
1	5	1	14 - 1 = 13
2	10	3	13 - 3 = 10
3	15	5	10 - 5 = 5
6	9	3	5 - 3 = 2
7	4	2	2 - 2 = 0

As we can observe in the above table that the remaining weight is zero which means that the knapsack is full. We cannot add more objects in the knapsack. Therefore, the total profit would be equal to (8 + 5 + 10 + 15 + 9 + 4), i.e., 51.

In the first approach, the maximum profit is 47.25. The maximum profit in the second approach is 46. The maximum profit in the third approach is 51. Therefore, we can say that the third approach, i.e., maximum profit/weight ratio is the best approach among all the approaches.

Example: **EXAMPLE**

Given data

weights = [1,2,4,5,1,3,7]
profit = [6,10,18,15,3,5,7]
object = [4,0,5,2,6,1,3]
c = 15

n=7