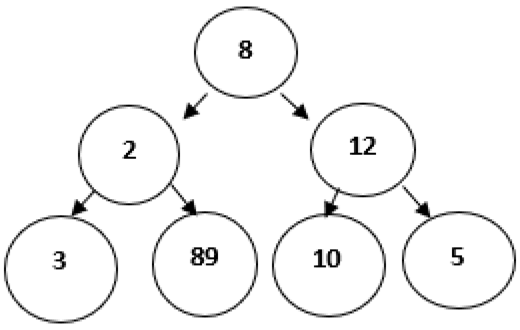
Greedy Approach or Technique

As the name implies, this is a simple approach which tries to find the **best** solution at every step. Thus, it aims to find the local optimal solution at every step so as to find the global optimal solution for the entire problem.

Consider that there is an **objective function** that has to be optimized (maximized/ minimized). This approach makes greedy choices at each step and makes sure that the objective function is optimized.

The greedy algorithm has only one chance to compute the optimal solution and thus, cannot go back and look at other alternate solutions. However, in many problems, this strategy fails to produce a global optimal solution. Let's consider the following binary tree to understand how a basic greedy algorithm works:



For the above problem the objective function is:

To find the path with largest sum.

Since we need to maximize the objective function, Greedy approach can be used. Following steps are followed to find the solution:

Step 1: Initialize **sum = 0**

Step 2: Select the root node, so its value will be added to `sum`, **sum = 0+8 = 8**

Step 3: The algorithm compares nodes at next level, selects the largest node which is **12**, making the **sum = 20**.

Step 4: The algorithm compares nodes at the next level, selects the largest node which is **10**, making the **sum = 30**.

Thus, using the greedy algorithm, we get **8-12-10** as the path. But this is not the optimal solution, since the path **8-2-89** has the largest sum ie **99**.

This happens because the algorithm makes decision based on the information available at each step without considering the overall problem.

When to use Greedy Algorithms?

For a problem with the following properties, we can use the greedy technique:

- **Greedy Choice Property**: This states that a globally optimal solution can be obtained by locally optimal choices.
- **Optimal Sub-Problem**: This property states that an optimal solution to a problem, contains within it, optimal solution to the sub-problems. Thus, a globally optimal solution can be constructed from locally optimal sub-solutions.

Generally, **optimization problem**, or the problem where we have to find maximum or minimum of something or we have to find some optimal solution, greedy technique is used.

An optimization problem has two types of solutions:

- **Feasible Solution**: This can be referred as approximate solution (subset of solution) satisfying the objective function and it may or may not build up to the optimal solution.
- **Optimal Solution**: This can be defined as a feasible solution that either maximizes or minimizes the objective function.

Key Terminologies used in Greedy Algorithms

- **Objective Function**: This can be defined as the function that needs to be either maximized or minimized.
- **Candidate Set**: The global optimal solution is created from this set.
- **Selection Function**: Determines the best candidate and includes it in the solution set.
- **Feasibility Function**: Determines whether a candidate is feasible and can contribute to the solution.

Standard Greedy Algorithm

This algorithm proceeds step-by-step, considering one input, say `x`, at each step.

- If `x` gives a local optimal solution (`x` is feasible), then it is included in the partial solution set, else it is discarded.
- The algorithm then goes to the next step and never considers `x` again.
- This continues until the input set is finished or the optimal solution is found.

The above algorithm can be translated into the following pseudocode:

```
Algorithm Greedy(a, n)   // n defines the input set

{

    solution= NULL;       // initialize solution set

     for i=1 to n do

     {

          x = Select(a); // Selection Function

          if Feasible(solution, x) then  // Feasibility solution

               solution = Union (solution, x);   // Include x in the
solution set

     }

     return solution;

}
```

Copy

Advantages of Greedy Approach/Technique

- This technique is easy to formulate and implement.
- It works efficiently in many scenarios.
- This approach minimizes the time required for generating the solution.

Now, let's see a few disadvantages too,

Disadvantages of Greedy Approach/Technique

- This approach does not guarantee a global optimal solution since it never looks back at the choices made for finding the local optimal solution.

Although we have already covered that which type of problem in general can be solved using greedy approach, here are a few popular problems which use greedy technique:

1. Knapsack Problem
2. Activity Selection Problem
3. Dijkstra’s Problem
4. Prim’s Algorithmfor finding Minimum Spanning Tree
5. Kruskal’s Algorithmfor finding Minimum Spanning Tree
6. Huffman Coding
7. Travelling Salesman Problem

Conclusion

Greedy Technique is best suited for applications where:

- Solution is required in real-time.
- Approximate solution is sufficient.