



Ask a Question



Introduction to Greedy Techniques

Let's go over the Greedy Techniques pattern, its real-world applications, and some problems we can solve with it.

We'll cover the following... 

About the pattern

An **algorithm** is a series of steps used to solve a problem and reach a solution. In the world of problem-solving, there are various types of problem-solving algorithms designed for specific types of challenges. Among these, greedy algorithms are an approach for tackling optimization problems where we aim to find the best solution under given constraints.

Imagine being at a buffet, and we want to fill the plate with the most satisfying combination of dishes available, but there's a catch: we can only make our choice one dish at a time, and once we move past a dish, we can't go back to pick it up. In this scenario, a greedy approach would be to always choose the dish that looks most appealing to us at each step, hoping that we end up with the best possible meal.

Greedy is an algorithmic paradigm that builds up a solution piece by piece. It makes a series of choices, each time picking the option that seems best at the moment, the most greedy choice, with the goal of finding an overall optimal solution. They don't worry about the future implications of these choices and focus only on maximizing immediate benefits. This means it chooses the next piece that offers the most obvious and immediate benefit. A greedy algorithm, as the name implies,



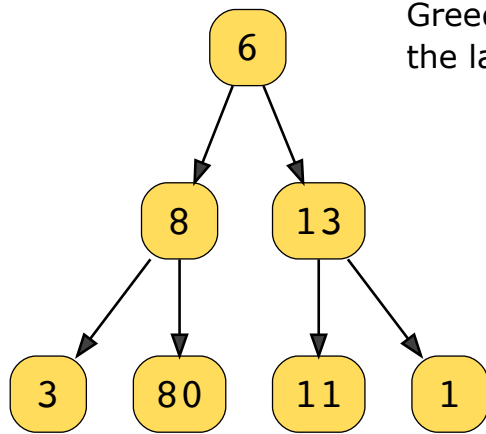
always makes the choice that seems to be the best at the time. It makes a locally-optimal choice in the hope that it will lead to a globally optimal solution. In other words, greedy algorithms are used to solve optimization problems.

Greedy algorithms work by constructing a solution from the smallest possible constituent parts. However, it's important to understand that these algorithms might not always lead us to the best solution for every problem. This is because, by always opting for the immediate benefit, we might miss out on better options available down the line. Imagine if, after picking the most appealing dishes, we realize we've filled our plate too soon and missed out on our favorite dish at the end of the buffet. That's a bit like a greedy algorithm getting stuck in what's called a local optimal solution without finding the global optimal solution or the best possible overall solution.

However, let's keep in mind that for many problems, especially those with a specific structure, greedy algorithms work wonderfully. One classic example where greedy algorithms shine is in organizing networks, like connecting computers with the least amount of cable. Prim's algorithm, for instance, is a greedy method that efficiently finds the minimum amount of cable needed to connect all computers in a network.

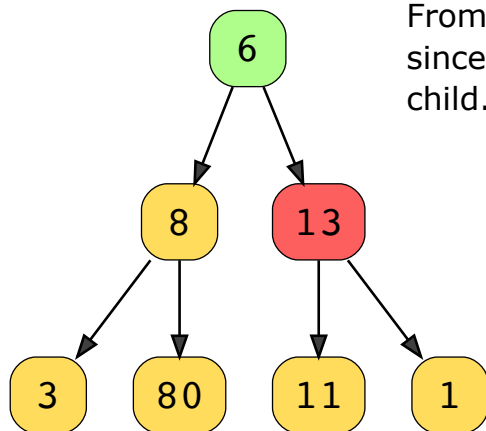
Note: To sum up, greedy algorithms offer a straightforward and often effective way to solve optimization problems by making the most advantageous choice at each step.

The illustration below shows a simple example of finding the path with the maximum sum of node values that demonstrates the working of a greedy algorithm and also shows how a greedy algorithm doesn't guarantee an optimal solution.



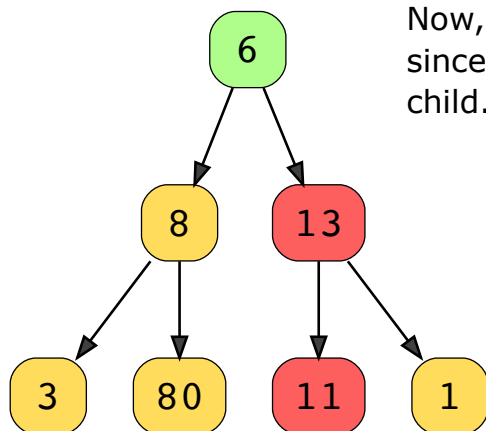
Greedy algorithm will look for nodes with the largest values.

1 of 5



From root node, it will go the right child since its value is greater than the left child.

2 of 5



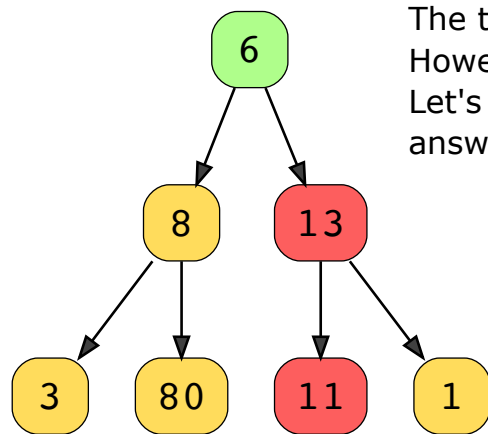
Now, it will go the left child since its value is greater than the right child.

?

Tt

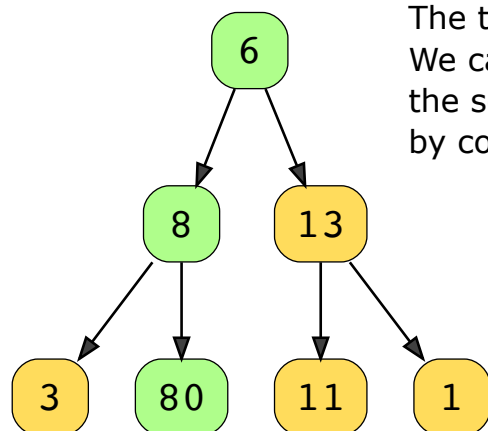
☾

3 of 5



The total sum from greedy approach is 30. However, this is not the optimal answer. Let's look at the next slide to see the optimal answer.

4 of 5



The total sum from optimal approach is 94. We can see that the greedy algorithm just selects the solution by looking at the small subsets and not by considering the entire picture.

5 of 5

—

[]

Examples

The following examples illustrate some problems that can be solved with this approach: ?

1. **Loading maximum containers in a cargo:** For the given container T and the cargo having a capacity of 90kg, fill the cargo with maximum containers without exceeding the capacity. ☾

20 kg 40 kg

60 kg 25 kg

Cargo Capacity = 90kg

To fill cargo with maximum containers, we will start loading cargos with the lowest weight. Therefore, container with 20kg is loaded first.

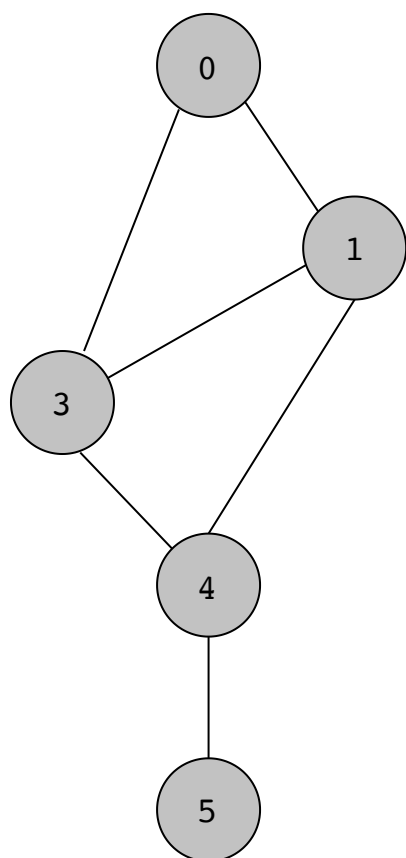
1 of 3

2. **Graph coloring:** Color a graph so that no two adjacent vertices are colored using the same color and minimum colors are used.

?

Tt






Color first node with a color.

1 of 6



Does your problem match this pattern?

- Yes, if both of these conditions are fulfilled:
 - **Optimization problem:** The problem is an optimization problem where we are looking to find the best solution under a given set of constraints. This could involve minimizing or maximizing some value, such as cost, distance, time, or profit.

- 
- **Making local choices leads to a global solution:** The problem can be solved by making simple decisions based on the current option or state without needing to look ahead or consider many future possibilities.
 - No, if any of these conditions is fulfilled:
 - **Local choices lead to sub-optimal solutions:** Our analysis shows that making local greedy choices leads us to a sub-optimal solution.
 - **Problem lacks clear local optima:** If the problem doesn't naturally break down into a series of choices where we can identify the best option at each step, a greedy algorithm might not be applicable.

Real-world problems


Many problems in the real world use the greedy techniques pattern. Let's look at some examples.

- **CPU scheduling algorithms:** In operating systems, managing how processes are assigned to the CPU for execution is critical for efficiency and performance. Greedy algorithms are used in CPU scheduling to decide the order of tasks based on specific criteria, such as minimizing waiting time, maximizing utilization, or ensuring fairness among users. By making greedy choices, such as selecting the process with the shortest expected completion time next (Shortest Job Next), systems aim to optimize overall throughput and resource utilization.

?


Tt



- 
- **Network Design in LANs:** For large Local Area Networks (LANs) connecting numerous devices, efficient data transmission is vital. Greedy algorithms help in designing the network's infrastructure to ensure data packets travel in the most efficient way possible. By finding a Minimum Spanning Tree (MST) for the network, a greedy algorithm like Prim's or Kruskal's ensures that all switches are connected with the least total cable length, reducing costs and improving network performance. This approach ensures minimal data transmission times and reduced chances of bottlenecks.
 - **Friend recommendations on social networking websites:** Social networking platforms like Facebook, LinkedIn, and X (formerly Twitter) enhance user engagement by suggesting new friends or connections. A key algorithm behind this feature is Dijkstra's algorithm, a greedy technique used to find the shortest paths between nodes in a graph. In this context, nodes represent users, and edges represent connections between them (like friendships). By finding the shortest path of connections between users, the algorithm can suggest people we may know or should connect with, based on your existing network of friends. This makes the platform more engaging by encouraging the growth of our social network through relevant connections.

Strategy time!

Match the problems that can be solved using the greedy techniques pattern.



Note: Select a problem in the left-hand column by clicking it, and then click one of the two options in the right-hand column.



Match The Answer

① Select an option from the left-hand side



Find the shortest period of time to schedule a set of jobs on a set of machines.

Find the number of palindromic substrings contained in a given string.

Find the shortest path between two intersections on a road map.

Find the most cost-

Greedy Techniques

Some other pattern

