DESIGN AND ANALYSIS OF ALGORITHMS

NATIONAL INSTITUTE OF TECHNOLOGY DELHI

Presented by Nikita Uikey

TRAVELLING SALESMAN PROBLEM

A BRIEF OUTLINE

The traveling salesman problem (TSP) is an algorithmic problem tasked with finding the shortest route between a set of points and locations that must be visited. In the problem statement, the points are the cities a salesperson might visit. The salesman‘s goal is to keep both the travel costs and the distance traveled as low as possible.

TRAVELLING SALESMAN PROBLEM

can be solved by using the following methods:

01.

BACKTRACKING

02.

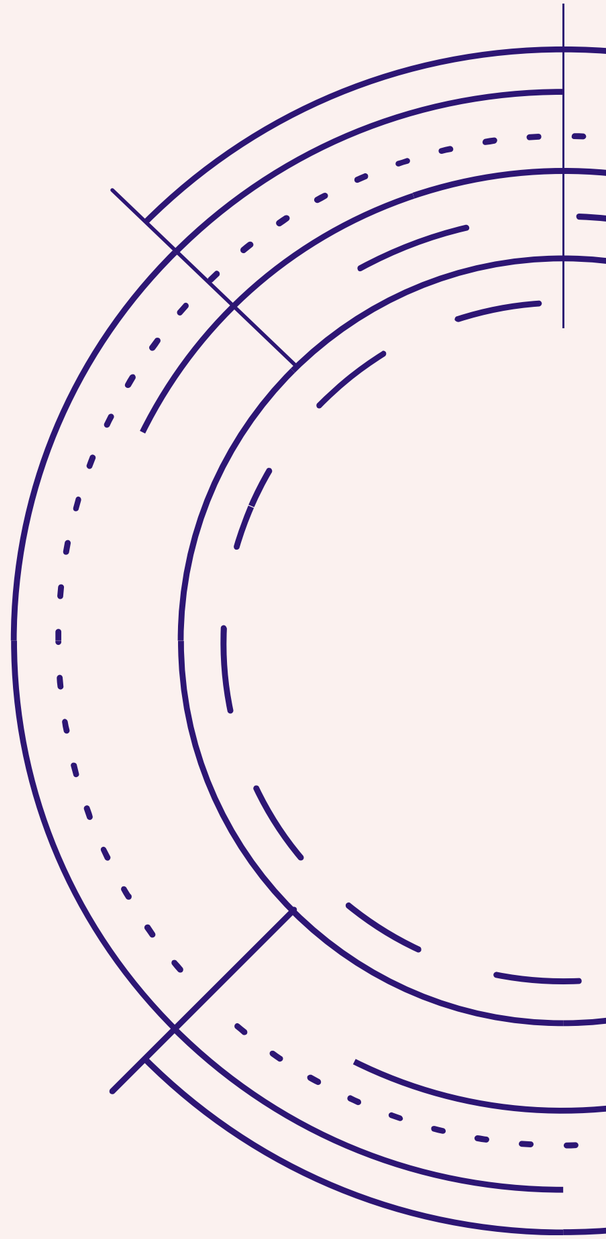
Dynamic programming

03.

BRANCH AND BOUND

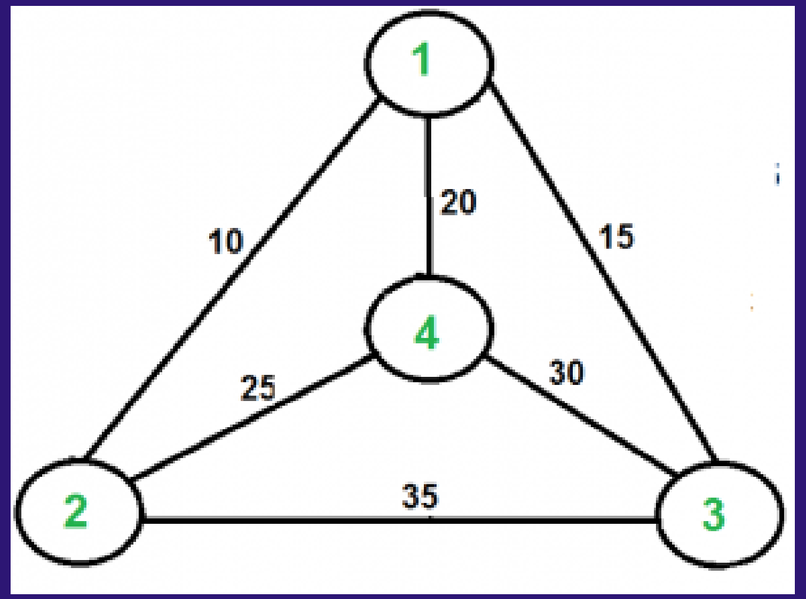
BACKTRACKING

The Backtracking is an algorithmic-method to solve a problem with an additional way. It uses a recursive approach to explain the problems. We can say that the backtracking is needed to find all possible combination to solve an optimization problem.



BACKTRACKING APPROACH:

- Consider city 1 (let say 0th node) as the starting and ending point. Since route is cyclic, we can consider any point as starting point.
- Start traversing from the source to its adjacent nodes in dfs manner.
- Calculate cost of every traversal and keep track of minimum cost and keep on updating the value of minimum cost stored value.
- Return the permutation with minimum cost.

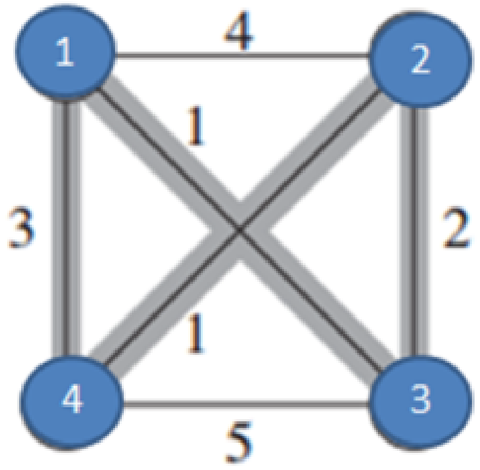


OUTPUT OF THE GIVEN GRAPH:

10 + 25 + 30 + 15 = 80

DYNAMIC PROGRAMMING

Dynamic Programming is mainly an optimization over plain recursion. Wherever we see a recursive solution that has repeated calls for same inputs, we can optimize it using Dynamic Programming. The idea is to simply store the results of subproblems, so that we do not have to re-compute them when needed later. This simple optimization reduces time complexities from exponential to polynomial



	1	2	3	4
1	0	4	1	3
2	4	0	2	1
3	1	2	0	5
4	3	1	5	0

ABOVE WE CAN SEE A COMPLETE DIRECTED GRAPH AND COST MATRIX WHICH INCLUDES DISTANCE BETWEEN EACH VILLAGE. WE CAN OBSERVE THAT COST MATRIX IS SYMMETRIC THAT MEANS DISTANCE BETWEEN VILLAGE 2 TO 3 IS SAME AS DISTANCE BETWEEN VILLAGE 3 TO 2.

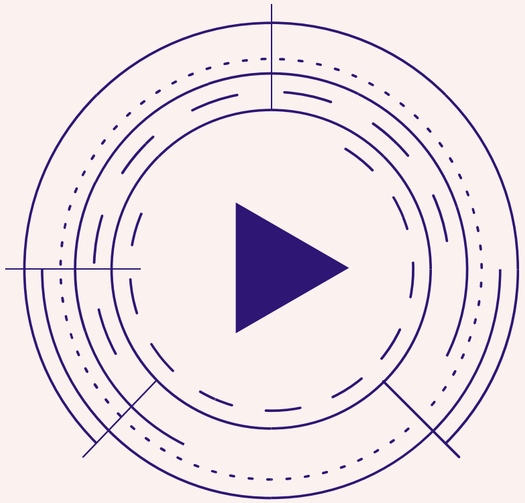
HERE PROBLEM IS TRAVELLING SALESMAN WANTS TO FIND OUT HIS TOUR WITH MINIMUM COST.

Say it is T (1,{2,3,4}), means, initially he is at village 1 and then he can go to any of {2,3,4}. From there to reach non-visited vertices (villages) becomes a new problem. Here we can observe that main problem spitted into sub-problem, this is property of dynamic programming.

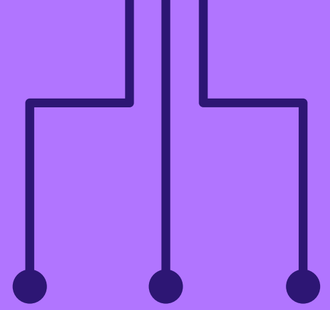
BRANCH AND BOUND

Branch and bound is an algorithm design paradigm which is generally used for solving combinatorial optimization problems. These problems are typically exponential in terms of time complexity and may require exploring all possible permutations in worst case. The Branch and Bound Algorithm technique solves these problems relatively quickly.

In Branch and Bound method, for current node in tree, we compute a bound on best possible solution that we can get if we down this node. If the bound on best possible solution itself is worse than current best (best computed so far), then we ignore the subtree rooted with the node.



BRANCH AND BOUND APPROACH:



the cost through a node includes two costs.
1) Cost of reaching the node from the root (When we reach a node, we have this cost computed)
2) Cost of reaching an answer from current node to a leaf (We compute a bound on this cost to decide whether to ignore subtree with this node or not).

- In cases of a maximization problem, an upper bound tells us the maximum possible solution if we follow the given node.

- In cases of a minimization problem, a lower bound tells us the minimum possible solution if we follow the given node.

BE INSPIRED

“

THANK YOU