Problem Solving with Applications
Algorithms: First Steps
Video: Brute Force Search
11 min
Quiz: Brute Force Algorithm
1 question
Quiz: Average Weight
1 question
Video: Nearest Neighbor
8 min
Quiz: Nearest Neighbors
1 question
Visualizing Cycles

Congratulations! You passed!

Grade received 100% Latest Submission Grade 100% To pass 80% or higher

Go to next item

Brute Force Algorithm

Quiz • 30 min

Review Learning Objectives

1. Implement the brute force algorithm for the Traveling Salesman Problem. The algorithm should check all the permutations of the vertices and return the minimum weight of a cycle visiting each vertex exactly once.

1 / 1 point

Submit your assignment

Due Jan 1, 11:59 PM IST

Receive grade

To Pass 80% or higher

Try again

Your grade

100%

View Feedback

We keep your highest score

```
import networkx as nx
from itertools import permutations

# This function takes as input a graph g.
# The graph is complete (i.e., each pair of distinct vertices is connected by an edge),
# undirected (i.e., the edge from u to v has the same weight as the edge from v to u),
# and has no self-loops (i.e., there are no edges from i to i).
#
# The function should return the weight of a shortest Hamiltonian cycle.
# (Don't forget to add up the last edge connecting the last vertex of the cycle with the first one.)
#
# You can iterate through all permutations of the set {0, ..., n-1} and find a cycle of the minimum weight.


def all_permutations(g):
    # n is the number of vertices.
    n = g.number_of_nodes()
    best_cycle_weight = float('inf')
    # Iterate through all permutations of n vertices
    for p in permutations(range(n)):
        # Write your code here.
        cur_cycle_weight = 0
        for i in range(n):
            cur_cycle_weight += g[p[i]][p[(i+1)%n]]['weight']
        if cur_cycle_weight < best_cycle_weight:
            best_cycle_weight = cur_cycle_weight

    return best_cycle_weight

```

Like Dislike Report an issue

Run

Reset

```
No Output
```

Correct

Good job!