Expected

July 19, 2020

```
In [ ]: import networkx as nx
        from itertools import permutations
        import math

        # This function computes the distance between two points.
        def dist(x1, y1, x2, y2):
            return math.sqrt((x1 - x2) ** 2 + (y1 - y2) ** 2)
```

```
In [ ]: # This function receives a list of 2-tuples representing the points' coordinates,
        # and returns the corresponding graph.
        def get_graph(coordinates):
            g = nx.Graph()
            n = len(coordinates)
            for i in range(n):
                for j in range(i + 1):
                    g.add_edge(i, j, weight=dist(coordinates[i][0], coordinates[i][1], coordinat
            return g
```

```
In [ ]: # This function computes the weight of the given cycle.
        def cycle_length(g, cycle):
            # Checking that the number of vertices in the graph equals the number of vertices in
            assert len(cycle) == g.number_of_nodes()
            # Write your code here.
            return sum(g[cycle[i]][cycle[i + 1]]['weight'] for i in range(len(cycle) - 1)) + g[c

        # This function iterates through all permutations and returns the length of an optimal c
        # You can implement any other algorithm here and visualize it.
        #
        def all_permutations(g):
            # n is the number of vertices.
            n = g.number_of_nodes()

            # Iterate through all permutations of n vertices
            opt = float('inf')
            for p in permutations(range(n)):
                # Write your code here.
                opt = min(opt, cycle_length(g, p))
            return opt
```

```
In [ ]: # Insert your function computing the average length of Hamiltonian cycles here:
        def average(g):
```

```
In [ ]: # Check how close an average solution is to an optimal one for the two following example
        coordinates1 = [
            (0,0),
            (60, 0),
            (30, 51.9615)
            ]

        coordinates2 = [
            (0, 0),
            (300, 0),
            (0, 10),
            (300, 10),
            (0, 20),
            (300, 20),
            (0, 30),
            (300, 30)
            ]

        g1 = get_graph(coordinates1)
        print("Example 1. The length of an optimal cycle is", all_permutations(g1))
        print("Example 1. The average cycle length is", average(g1))

        g2 = get_graph(coordinates2)
        print("Example 2. The length of an optimal cycle is", all_permutations(g2))
        print("Example 2. The average cycle length is", average(g2))

        # You might want to copy these coordinates into the previous Jupiter Notebook
        # to visualize the datasets and see other examples.
```