Show all posts by recent activity

? Lab 5: Exercise C- Wrong Answer 5
Hi, My Brute-force algorithm...

Lab 5 exercise C Distance for Bruteforce 3
Hello the distance for the br...

? Lab 6: Strange Results 6

? a headache question in lab5 exercise C 4
Hi, for the exercise C of the L...

Lab 5, C 3
Dear all, I cannot make exer...

How to I read the adjacency graph? 6
Hi, I am stuck at exercise C o...

Lab 5 (c) 3

? lab 5 Exercise C 21
Where the instructions say: ...

? lab 5 Exercise C 18
In the description of the gre...

Lab 5, C

discussion posted about a year ago by **annatum**

Dear all,

I cannot make exercise C in Lab 5 work. Here is my code. What can be wrong? Thanks!

```
def greedy_complete_graph(graph, initial_vertex):
    current_distance = 0

    # while we still have a vertex that is not explored
    #    get next target using choose_target function
    #    update player position
    #    update distance value
    #    remove next target from vertices list
    #
    # YOUR CODE HERE


    explored_vertices = []
    for row in graph:
        for vertex in row:
            if vertex not in explored_vertices:
                print(vertex)
                next_target = choose_target(distances,pieces_of_cheese)
                initial_vertex = next_target
                current_distance += distances[initial_vertex]
                explored_vertices.append(vertex)

    return current_distance



best_distance = infinity

def bruteforce_complete_graph_auxiliary_function(remaining_vertices, current_vertex,
current_distance, graph):
    global best_distance
    # if there are no remaining vertices:
    #   if the current distance is shorter than the best distance:
    #     update the best distance to match the current distance
    # else:
    #   for each vertex in the remaining vertices:
    #     perform a copy of the remaining vertices, let us call them "new_vertices"
    #     remove the vertex from new_vertices
    #     call the function bruteforce_complete_graph_auxiliary_function recursively
    #
    # YOUR CODE HERE
    if remaining_vertices == []:
        if current_distance < best_distance:
            best_distance = current_distance
    else:
        for vertex in remaining_vertices:
            new_vertices = remaining_vertices[:]
            new_vertices.remove(vertex)
            current_distance += graph[vertex][current_vertex]
            best_distance = bruteforce_complete_graph_auxiliary_function(new_vertices, vertex,
current_distance, graph)
    return best_distance
```

Anna

Related to: Week 5 / Greedy Algorithms
This post is visible to everyone.

Add a Response

2 responses

annatum
about a year ago

def bruteforce_complete_graph_auxiliary_function(remaining_vertices, current_vertex, current_distance, graph):

```
global best_distance
# if there are no remaining vertices:
#   if the current distance is shorter than the best distance:
#     update the best distance to match the current distance
# else:
#   for each vertex in the remaining vertices:
#     perform a copy of the remaining vertices, let us call them "new_vertices"
#     remove the vertex from new_vertices
#     call the function bruteforce_complete_graph_auxiliary_function recursively
#
# YOUR CODE HERE
print(remaining_vertices)
if remaining_vertices == []:
    if current_distance < best_distance:
        best_distance = current_distance
        #print("first {}".format(current_distance))
else:
    for vertex in remaining_vertices:
        new_vertices = remaining_vertices[:]
        new_vertices.remove(vertex)
        current_distance += graph[vertex][current_vertex]
        best_distance = bruteforce_complete_graph_auxiliary_function(new_vertices, vertex,
current_distance, graph)
        #print("second {}".format(current_distance))
return best_distance
```

Add a comment

vgripon (Staff)
about a year ago

I do not know if that is the source of the problem, but in your function **greedy_complete_graph**, you call **next_target = choose_target(distances,pieces_of_cheese)**. But I do not see where you define **distances** nor **pieces_of_cheese*.

Also, you should be cautious about the form of the graph that is sent to **greedy_complete_graph**. I recall here:

> You will need to define the greedy_complete_graph which uses the greedy algorithm to explore all vertices in a complete graph. In more details, the greedy_complete_graph function takes two arguments: the graph described by an adjacency matrix (we suppose vertices are indexed from 0 to n-1, where n is the order of the graph, and the initial_vertex, which is an integer between 0 and n-1. The output of this function is the total length of a walk going through all vertices of the graph and using the previously definied heuristic of systematically going to the closest unexplored vertex in the graph.

Add a comment

Showing all responses

Add a response:

Preview

Submit