Test4-Canvas-Graphs Results for Caden Roberts

ⓘ Correct answers are hidden.

Score for this quiz: **10** out of 10
Submitted Aug 30 at 9:02pm
This attempt took 28 minutes.

Question 1

4 / 4 pts

A common recommendation feature used in social networking websites is the number of common friends. In an undirected graph G, find a pair (u, v) with the largest number of common neighbors.

Your Answer:

Main Idea: A common neighbor of two nodes u and v in a graph G is a node that has edges from it to both u and v. We have to iterate through all possible node pairs and count their common neighbors to determine the pair with the largest count. For each pair of nodes (u, v), count the # of common neighbors by checking the intersection of their adjancency lists. The pair with the maximum # of common neighbors is the result.

Psuedocode:

1. Initialize maximumCommon = 0 and resultPair = (Null, Null)

2. For each pair of vertices u and v in the graph:

   2a. If u is not v, find the intersection of the adjacency lists of u and v.

   2b. Let commonCount = size of this intersection.

   2c. If commonCount > maximumCommon:

      - set maxCommon = commonCount.

      - set pair = (u, v).

return pair.

Running time analysis:

Initializing variables takes O(1). The loop iterates all vertices, giving O(V^2) iterations, with V as the number of vertices.

To compute the intersection of their adjacency lists takes O(minimum of the degrees of u or v). So, the worst case time would be O(V) time per pair. Returning the pair is O(1). Overall running time is O(V^3).

Question 2

6 / 6 pts

Biologists often construct a food network of species in an ecosystem. The vertices represent species, and a directed edge (u, v) means species u eats species v. An apex species is one that is not eaten by another species. Suppose you are give a list of all possible “eating” relationships (so a list of “u eats v”). Determine all apex species.

Your Answer:

Main Idea:

We have to identify apex species in a directed graph where vertices represent species and directed edges represent 'eats' relationships (u eats v for (u.v)) An apex species is a species that is not eaten by any other species. We have to find all vertices with no incoming edges. We will construct a directed graph from the list of 'u eats v' relationships. For each vertex in the graph we will count the number of incoming edges. We collect all vertices with an incoming edge count of 0 and they are the apex species.

Psuedocode:

1. Initialize an empty dictionary graph to represent the adjacency list of the graph.

2. Initialize a dictionary degree to store the # of in-degrees of each vertex

3. Initialize a set apexSpecies to store the apex species.

4. For each pair (u. v) in the list of relationships:

   4a. If v is not in degree, set degree[v] to 0.

   4b. Increment degree[v] by 1.

4c. If u is not in degree, set degree[u] to 0.

5. For each vertex u in the graph:

5a. If degree[u] equals 0:

Add u to apexSpecies.

return apexSpecies.

Running time analysis:

Constructing the graph using the adjacency list graph takes O(E) because each edge is processed once. Initializing/updating degree dictionary takes O(E) as well. Finding the apex species takes O(V) to check each vertex. Overall running time then is O(V+E).

Quiz Score: **10** out of 10