CS2040S: Data Structures and Algorithms

Tutorial Problems for Week 7

For: Oct 4, 2019

Problem 1. Oizne Mak's party

Oizne Mak wants to host the greatest party in the world! He wants to invite as many people as possible, but he is lazy and wants to ask his friends to help invite their friends to the party. He needs to find out which of his friends know each other, so that he can reduce the number of invitations he has to send out himself. Oizne Mak thus needs to be able to collate information about all his friends. As Oizne Mak's best friend and an expert on graphs, he wants to ask you which graph representation will be the best to store the information. We can assume that there a total of V friends and E pairs of friends who know each other.

Problem 1.a. Oizne Mak needs to be able to quickly know some information about his friends when he checks the graph. Given any two friends of Oizne Mak, *Friend A* and *Friend B*, which representation is the best for answering the following questions quickly?

- Do *Friend A* and *Friend B* know each other?
- Who is a friend of *Friend A*? (i.e. find any one friend of *Friend A*)
- Who are all the friends of *Friend A*? (I want to know all the names of *Friend A*.)

Oizne Mak realises that he does not have enough space in his house to host a lot of people. Now he only wants to host friends who are close to each other. Each pair of two friends *Friend A* and *Friend B* will have a friendship value FV, which is a non-negative integer. If *Friend A* and *Friend B* are not friends, they will have a FV of 0. Again, Oizne Mak asks you for help on which graph representation will be the best to store the information now.

Problem 1.b. Now, which representation is the best for answering the following questions quickly?

- What is the FV of *Friend A* and *Friend B*?
- Who is the closest pair of friends among all the pairs? (i.e. pair with the highest FV)
- How many friends of *Friend A* have FV greater than some value M?

Problem 2. Oizne Mak's party

Oizne Mak wants to play a game with the people who came to his party. He wants them to stand in his garden in a particular layout called a planar graph. A planar graph is a special kind of graph where there exists a way to draw it out on a flat piece of paper without any edges crossing each other. A planar graph has these properties:

- The average vertex degree in a planar graph is strictly less than 6.
- After deleting a vertex (along with the incident edges) from a planar graph, the resulting graph is also planar.

To make sure nothing goes wrong during the party, Oizne Mak wants to simulate the game on his computer beforehand. He wants you to help him devise a data structure that supports $O(1)$ adjacency queries like an adjacency matrix for planar graphs. That is, given a graph as an adjacency list, build a data structure such that you can check whether two players *Player A* and *Player B* are neighbours in the planar graph in $O(1)$ time. Try building the data structure within the following time and space complexities:

- $O(N^2)$ time and $O(N^2)$ space
- $O(N \log N)$ time and $O(N)$ space
- $O(N)$ time and $O(N)$ space

where N is the number of vertices in the planar graph.

Hint: Think about orienting the edges.