DSC 190 - Homework 04

Due: Wednesday, February 2

Write your solutions to the following problems by either typing them up or handwriting them on another piece of paper. Unless otherwise noted by the problem's instructions, show your work or provide some justification for your answer. Homeworks are due via Gradescope at 11:59 PM.

Problem 1.

Suppose you initialize an LSH data for storing points in $\mathbb{R}^2$ by choosing $\ell = 1$, $w = 1$, and $k = 2$ random unit vectors (shown below):

$$\vec{u}^{(1)} = \left(\frac{1}{\sqrt{2}}, \frac{1}{\sqrt{2}}\right)^T$$
$$\vec{u}^{(2)} = \left(\frac{\sqrt{2}}{\sqrt{3}}, -\frac{1}{\sqrt{3}}\right)^T$$

You insert the following points into the LSH data structure one-by-one:

x	y
-1.5	1.5
2.5	-1.5
-0.5	0.5
1.5	-0.5
-1.5	-1.5
0.5	0.5

Upon querying the point (-2, -2), what other point(s) will be in the same cell? Do your calculations by hand and show your work (though you can use code to check your answer, if you wish).

Programming Problem 1.

In a file named `dsf.py`, create a class named `DisjointSetForest`. This class should have `.find_set`, `.make_set`, and `.union` methods as described in lecture, as well as the following methods and attributes:

- `.size_of_set(x)`: return the size of the set containing `x` as an integer. This should be done efficiently in $O(\alpha(n))$ amortized time.
- `.number_of_sets`: an integer attribute containing the current number of disjoint sets (not elements!) contained in the collection.

For methods that require elements to be specified, such as `.find_set`, `.union`, and `.size_of_set`, a `ValueError` should be raised if the element does not exist within the collection.

Starter code is available on the course webpage. Note that you may need to modify some of the existing methods from lecture, such as `.make_set`!

Problem 2.

Consider the following model for iteratively generating a random connected graph with n nodes. Start with a graph containing n nodes and no edges. Next, randomly select an edge from the uniform distribution on

the set of all possible edges and add it to the graph (if it isn't already in the graph). Repeat this process until the graph is connected[1].

How many edges will the resulting graph have? It is a random number, of course, but it must be at least $n - 1$. Let's visualize the distribution of this quantity through an empirical experiment, *a la* DSC 10.

Write a function to simulate the above procedure and return the number of edges in the resulting graph, assuming $n = 100$. Run your function 10,000 times and plot a histogram of the results. Include your figure, as well as your code. Your code should be fast enough so that all 10,000 trials take less than 1 minute or so in total.

Hint: you can use DFS/BFS to determine whether the graph is connected, but if you do your code will take forever to run... Is there a better way of keeping track of connected components?

[1]This random graph model is related to (but not the same as) the famous Erdös-Renyi random graph model. While simple, the E-R model has been the subject of a lot of theoretical analysis.