CS320 Lab: Breadth-First Search

Prof. Craig Partridge

Lab #7

This lab is focused on teaching you the mechanics of traversing a graph. We will provide a graph module, which implements a graph class and a function to load graphs. Your job is to implement a Breadth-First Search (BFS) function that operates on a graph as defined by this class.

1 What is Breadth-First Search?

Breadth-First search is explained in detail in section 14.4 of the course textbook and there is a pseudo-code implementation in Algorithm 14.4.1.

2 The Graph Module

We have created an edge list graph package, *edgegraph.py*, which you should download from canvas and then import into your BFS module.

3 The Assignment

You are to write a function `bfs(graph, start)` which accepts a graph (GraphEL) and a start vertex (VertexEL) and returns a list of tuples in a valid BFS order. Specifically, if `lot` is the returned list of tuples, `lot[0]` contains `start`, `lot[1]` contains the vertices that are adjacent (1 hop) from `start`, and `lot[2]` contains the vertices that are 2 hops from `start`. A vertex must only appear once in `lot`, in the tuple corresponding the shortest distance from `start`.

You may assume that `graph` is not a forest.

3.1 Requirements

Reminder that every assignment has a code portion, which is worth 60 points (40 for correctness, 20 for literate programming), and a reflection (worth 40 points).

3.1.1 Code Requirements

Your code must run in time $O(n + m)$ where n is the number of vertices and m is the number of edges.

If `graph` or `start` is `None`, your code should raise the `ValueError` exception with the message "Invalid graph or vertex".

If `start` is not in `graph`, you should return an empty list (`[]`).

You may **not** alter the graph (e.g. place markings in the graph).

3.1.2 The Reflective Essay

We start with a reminder that this essay is not about impressing us with how smart you are. Rather we are looking for evidence that you thought (reflected) on the problem-solving process you used for this lab.

In this week's reflection, rather than describing your thought process, we're asking you to explain the concept of BFS to two audiences:

- A seven year old (e.g. a first grader who is just being introduced to reading and addition and subtraction); AND
- A first year Computer Science student who has taken CS150B (so knows Python) but not much more.

Structure your essay in two parts: one for the seven year old, one for the new CS student.

The total length of your essay is limited to 300 words (across both explanations).

The goal of this exercise is for us to see how well you understand BFS. In principle (though we recognize, not always in practice), the better you understand BFS, the simpler it should be to explain it to different audiences.

3.2 Tips

This assignment is really just implementing Algorithm 14.4.1 from the textbook with two small wrinkles:

- You need a structure for marking which vertices are explored (see injunction not to mark up the graph above);
- Rather than creating multiple lists L_i you are to structure them as tuples in a list of tuples and return that list.