Project Proposal and Outline

Teammates:

Alice Getmanchuk - aliceg3

Hassan Farooq - hfaroo9

Jerry Balan - agbalan2

Timothy Vitkin - tvitkin2

Project Goals:

Create a visual representation for Computer Engineering/Computer Science students' course requirements and prerequisites. Our project aims to visualize the different classes/pathways required prior to taking the class inputted into the program. It will look similar to what Wade did with his website as seen here: https://waf.cs.illinois.edu/discovery/class_hierarchy_at_illinois/ . We will adjust the scope as needed.

Datasets:

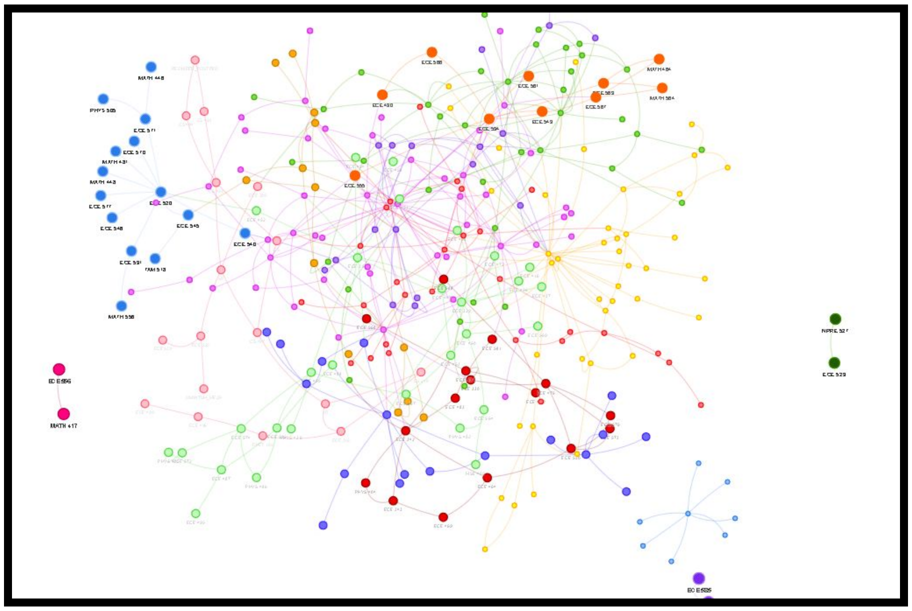
We will be using the course prerequisite csv file found here: https://github.com/illinois/prerequisites-dataset , possibly combining it with the course catalog and gened datasets found here: https://github.com/wadefagen/datasets . Both repos specify the format of the csv files within their README's. We will be using information publicly accessible from the University of Illinois to extract information regarding ECE/CS curriculum and prerequisites in order to construct our graph(s).

Algorithms:

We will implement a BFS traversal of a force-directed graph. (We chose to do BFS because ECE/CS classes are not usually convoluted in the way they are connected, so BFS will be more efficient.) We will use the cs225 namespace classes (PNG, HSLAPixel, etc.) to display the graph of classes. We are thinking of having a predetermined location for each class on the canvas and we recolor/highlight the relevant parts when the user inputs the class of interest. We will input the class of interest as a command line argument when we run the code. Based on that class, if it finds a match to a class on the graph, it will highlight that class's prerequisites and what other classes it leads into, and co-requisites if applicable. (We are also thinking of adding junior eligibility for ECE into it.) We will need to figure out how to utilize multiple connection types so this web of different classes can actually be utilized well. From the specified options on the project goals doc, we'll likely end up using some sort of shortest path algorithm (Djikstra's algorithm), and we'll be providing a graphic output of the graph.

*****We will use adjacency matrices as well as other implementations for directed graphs to construct accurate and quick (run-time) accessibility of desired information. We will have a matrix for vertices (classes) as well as an adjacency matrix to store the directed edges in order to more easily access edges when looking for a specific path. It may also be beneficial to have a matrix of incoming edges and also outcoming edges in order to speed up our class searching process knowing which classes come before.*****

It will look kind of like this, but more tailored to a specific input course:



Potential Features:

- Highlighting classes that are prerequisites for the selected class
- Different pathways for different majors
- Highlight prerequisites / corequisites