Leading Question

Our project is to create a visualization of different Anime TV shows and movies and their similarities to each other by parsing data about users on MyAnimeList (an online anime rating service similar to IMDb) and their ratings on different shows. We want to calculate the difference in rating between different shows averaged between all different users and graph the “distance” between shows based on these average distances in rating. Using this graph, we can run certain graph algorithms on our result to find how similarly anime are connected to each other or recommend a show based on which shows a person has watched.

Dataset Acquisition and Processing

We plan on using this dataset: https://www.kaggle.com/azathoth42/myanimelist, which was obtained through data scraping every MyAnimeList.net url. We will download it as a CSV, process out the important values using C++, and store our needed data as a graph using an array of lists representing each node and its connections. Each Node will be a separate anime, and edges in the graph are intended to be made when the average difference between scores is less than 2 (though our declared sensitivity may change once we see the data in order to get better results). In order to maximize accuracy and efficiency, we will only use the 1001 most popular anime and cut out any errors manually as needed.

Graph Algorithms

Whenever the user inputs an anime, we will use BFS in order to find it when starting at the root anime, where the BFS searches every connection next to the root before searching the connected nodes. BFS runs in O(n) as our graph is not sorted so on average it will find what it’s looking for in n/2 checks. After finding two anime, we plan on running Djikstra’s Algorithm, which runs in O(nlogn + m) runtime, and takes in two nodes and returns the shortest path. Using this we will be able to find the “distance” between these two anime by counting the number of branches in the shortest path without having to store connections between every single one. Finally, we plan on making a force-directed graph drawing, which makes the graph look aesthetically pleasing by assigning forces to simulate position of the nodes. Nodes (anime) with more scores will be bigger and similarity in scores across users will lead to closer nodes. We believe that if we adjust our parameters enough, the fact that similar scores will lead to closer nodes will mean that there will be different groupings of “genre” that form, and possibly even different disjoint sets. Force-directed graphs typically have runtime equivalent to O(n^3).

Timeline

We want to finish processing our data into our data structure by November 21. This will be our data set that creates connections based on similarity. Simple print functions on smaller data sets will let us know if this works or not.
We want to implement BFS in order to get searching working by November 28. This part of the project goes hand in hand with the pathfinding Djikstra's algorithm, so we also may be working with this simultaneously.
We want to finish implementing the Djikstra's algorithm pathfinding by December 5. This is when we will be able to see the effects of our graph and make adjustments to the sensitivity in order to create useful data.
The final part of our project will be the Force-Directed Graph (planned to be finished December 13), which will be a visualization of our data set--projected end goal of this project. Ideally this will look as nice as possible.