Simplified Page Rank

I chose to use adjacency matrix mostly because I found more resources with this than I did with the adjacency list as well as it being easier to implement. Additional thoughts when choosing between using a list or a matrix is I liked efficiency of using a matrix when removing, or querying a matrix as compared with the list. One con of using a matrix over a list is the matrix takes up more space and takes longer to add a vertex on average, with both being O(n2). Space for an adjacency list, on the other hand, is only O(n). Overall, the list seems to be slightly better in terms of total efficiency, but harder to implement. If I had more time to work on this project, I would probably consider building an adjacency list instead.

I initially tried using an adjacency list, but I had trouble with the implementation. The matrix seemed to be a little easier, so I decided to switch to a matrix instead. Some challenges I had with the matrix was properly implementing and calculating the power iterations. This was probably due to my misunderstanding of power iterations and how they’re used. If I could do this again, I would spend more time trying to understand this concept and looking more closely at both the matrix and list so that I can make a more informed decision on which to use, instead of switching later.

Overall, I enjoyed the experience. I especially found it interesting reading all the real world applications that graphs can be used in, such as travel directions, social media, maps, operating systems, and more.

|  |  |  |
| --- | --- | --- |
| Complexity of Code | | |
| Operation | **Complexity** |
| Adding Vertex | O(n2) |
| Removing Vertex | O(1) |
| Edge Query | O(1) |