# Google Page Rank Commentary

By Jordan Smith

## Data Structures:

The objects I created to help complete this project get more complex as the program executes. I used a simple Node object to hold a string value, an outdegree variable (made a double for easy computation later), and a vector<string> neighbors that holds the names of all the Nodes that the Node object points to. I then created a Graph object to store a couple of maps that allow one to search for a Node when given its name. I also stored the complete list of nodes inputted into the program in this object, in order to more easily operate on and construct the Graph's main object, which is a map that relates a node's string value to a vector<string> of its neighbor's names, which is just the Node's vector<string> neighbors. Finally, I created a Adjacency Matrix object that stores the Graph object and copies of its important data structures, as well as 2D vector object for the iteration matrix and the matrix itself.

I made heavy use of vectors and maps in this project for three reasons: I didn't want to risk being slowed down by the intricacies of dynamic array allocations, I liked the various amounts of functions you are able to use from the containers, and I wanted many different ways to access data, such as a Node's string value and index in the matrix. Although, these data structures are not the most efficient method of completing this project, they made conceptualizing the project's core problem must easier for me in my opinion. If I were to do this project over again I would employ the use of data structures such as heaps or queues because of their superior computation time as compared to maps.

## Graph Implementation:

For the matrix I choose to use a 2D vector, as to do away with any possible problems of allocating a dynamic 2D array. It is also easier to work with a 2D vector than a 2D array, an example of this being how it takes one line of code to fill a 2D vector with 0's and two for-loops worth of code to fill a 2D array with 0's. I don't think I would change this part of the project if given a second chance as it made doing computations on the matrix very easy.

## Computational Complexity:

For all the methods in the Node object, the time complexity is O(1). This is due to how the Node is mostly used for just placing data, so no complex calculation is done in any of the member methods. All of the methods in the Graph object are O(number of Nodes), due to all of the functions needing to search the vector of Nodes for a particular node for a specific piece of information. The most complex method is in the Adjacency Matrix object, being the setMatrix() having a time complexity of O((number of nodes)^3

* log(number of nodes))! All other methods are either O(number of nodes) or O(number of nodes * number of nodes).

## **Reflection:**

The hardest part about this project was figuring out how to translate matrix multiplication into code, with most of my time being spent scratching my head at the weird values I was getting for the iteration matrix after a number of iterations. I also had some problems early on figuring out how to best go about organizing the data into separate containers, as well as implementing safety and space-saving measures such as storing my data privately and passing Nodes by reference.

I did learn a bit about 2D containers, such as how they're not as bad to work with if one writes out what they want to do ahead of time in pseudocode or with pictures. I employed the use of drawing out the graphs heavily for this assignment, and I learned the hard way that some problems in computing can't be easily thought through without the help of a visual aid of some sort. Learning about the page rank algorithm was interesting, although I'm still not quite completely certain of how the power iterations of the matrix help compute the page rank. Nevertheless, the page rank algorithm at least demonstrated to me the power of complex mathematical structures such as matrices and their usefulness in a specific problem such as this.

In summary, I learned about the value of data structures in relation to solving as complex a problem as trying to map out the Internet, and I will use the knowledge I gained to build even more complex projects in the future.