Anna Vasconcelos
DS210: Programming for Data Science
Kontothanassis
3 May 2023

Final Project DS210: Analysis of Attributes in the Facebook Degrees of Separation and Connections

The dataset I had chosen is based on SNAP data from Facebook and contains over 4,000 nodes and 88,000 edges. The data was taken from a survey of users on their friendships/social circles. In this dataset, the nodes represent different profiles of people, and the edges represent "friend" connections. The data is anonymized to prevent personal data from being shown to the public. The project I really wanted to focus on is the concept of "Six Degrees of Separation". This idea is that any two people on Facebook can be connected through a chain with no more than 6 people in between.

In the main.rs part of this project, the goal is to study the degrees of separation idea. First, the code reads the Facebook dataset file and creates an adjacency list that represents a graph of connections. Then it uses the Breadth First Search (BFS) algorithm to find the shortest path between two randomly selected vertices (users) in the graph. This process is repeated 3000 times to see a wide sample of degrees of separation to be compared later. The last part of this code is that the average distance between pairs of vertices is calculated and that is the value that is returned in the output.

The output for this part of the code is the average number of degrees of separation in the repeated process. To speed up computation times when running this code, use cargo run --release. After running this code between 10-20 times, the average distance between vertices was anywhere between 5.1 and 5.5 degrees of separation, thus showing that in this dataset the notion of 6 degrees of separation holds true because in random choosing of pairs, it is shown there is only about 5 degrees of separation between two profiles/people.

This code also uses a test that utilizes a test, to see if it returns a value that uses 5.0 degrees of separation with 1.0 degrees of separation of error so it tests if it could be 4.0 to 6.0 degrees of separation and uses asset eq.

Then, my code is split into a module called pt2.rs to look into a different attribute of this graph which is on average how many different connections does one user of Facebook have? Specifically, in this dataset how many "friends" through edges does one user have? This code functions similarly, in that it reads the file into a graph through an adjacency list.

The first part of pt2.rs calculates the number of nodes and edges in a graph, this is done by iterating through the adjacency list created by reading the file into an adjacency list, it counts the number of elements and this is returned as a tuple. In this is a struct called Graph with two methods new() that reads the actual file and puts it into an adjacency list, calculate_degree() which selects a random node and calculates its degree (number of connections). The second part of this code finds the actual amount of connections, this does it by selecting 500 random nodes and calculating the number of connections it has, these values are stored in a vector which is printed as a result. Then, using this vector the calculated average is from the 500 nodes, sums the degrees, and divides by the length of the vector which is 500. Then this code prints a print statement that shows how many connections on average each person/profile has. After running this code around 10 times the average amount of connections each profile has is between 20 and 23 profiles.

This code for pt2.rs has a test case as well that ensures the implementation is correct, that the adjacency matrix is read correctly, the degree is calculated correctly and there are no other errors. This test case also reads from a quick test.txt I created with some trial vertices and edges to test the usage of the functions.

Therefore, the output of this program when using cargo run --release will print out a print statement showing the print statement of what is average degrees of separation, a vector of 500 nodes each element having the number of connections that each node has, then a print statement of the average number of connections each node has when taken from that sample of 500. Both tests for the main.rs and pt2.rs use assert_approx_eq, which checks if calculated averages are equal to expected values within the error possibilities. These can be tested using cargo test and it will show if it is passed as well.

Overall, I learned a lot about how social networks work and how to implement algorithms we had learned in class such as BFS, and then also just exploring two different yet connected facets in social network graphs as well (such as both the amount of connections users have and the degrees of separation between users).

Printed Outputs:

Entire Program:

```
Average degrees of separation in Facebook dataset: 5.4418607
Random 500 nodes and the degree value they each have
[21, 5, 0, 12, 22, 8, 21, 20, 2, 23, 8, 18, 0, 22, 9, 8, 64, 8, 13, 0, 89, 6, 37, 5, 3, 7, 3, 0, 1, 0, 3, 6, 87, 1, 3, 1, 0, 48, 2, 6, 83, 2, 47,
 36, 1, 13, 5, 75, 7, 0, 7, 0, 1, 31, 10, 39, 37, 17, 78, 4, 10, 33, 100, 1, 27, 41, 20, 2, 1, 57, 133, 122, 7, 34, 16, 5, 4, 32, 14, 92, 6, 87,
22, 20, 10, 15, 18, 2, 0, 7, 53, 12, 6, 94, 23, 2, 61, 19, 86, 3, 82, 28, 19, 78, 9, 38, 87, 76, 11, 42, 118, 5, 7, 3, 1, 3, 31, 12, 14, 31, 0, 1
6, 0, 55, 4, 112, 24, 20, 68, 4, 3, 6, 17, 15, 8, 37, 1, 106, 53, 133, 12, 9, 31, 9, 1, 10, 174, 10, 11, 44, 131, 86, 23, 0, 28, 8, 18, 6, 0, 14,
 15, 11, 25, 16, 36, 3, 14, 54, 33, 6, 3, 9, 93, 9, 16, 11, 5, 48, 5, 4, 31, 1, 0, 7, 2, 179, 11, 22, 4, 89, 1, 8, 0, 24, 0, 69, 11, 1, 12, 0, 6,
 0, 65, 14, 11, 1, 3, 2, 2, 0, 33, 0, 9, 76, 17, 7, 26, 1, 66, 0, 14, 18, 3, 8, 3, 18, 9, 3, 0, 0, 31, 19, 104, 23, 9, 45, 2, 28, 2, 5, 6, 0, 0,
49, 1, 9, 3, 1, 1, 61, 16, 9, 5, 1, 104, 20, 39, 7, 15, 154, 137, 0, 90, 26, 129, 5, 1, 29, 23, 16, 30, 0, 1, 23, 9, 4, 5, 1, 5, 130, 2, 18, 77,
29, 7, 64, 12, 0, 0, 0, 6, 29, 160, 9, 1, 23, 164, 52, 24, 14, 7, 8, 3, 1, 10, 8, 1, 8, 6, 11, 32, 170, 0, 12, 10, 19, 9, 72, 0, 15, 50, 1, 18, 4
, 5, 11, 8, 18, 33, 23, 2, 2, 2, 1, 15, 1, 16, 8, 11, 0, 4, 1, 37, 34, 42, 51, 31, 29, 3, 4, 6, 5, 7, 7, 36, 0, 16, 6, 69, 3, 2, 2, 1, 1, 17, 11,
 0, 13, 37, 111, 147, 7, 7, 8, 40, 1, 23, 121, 0, 9, 10, 19, 56, 91, 12, 11, 1, 22, 84, 59, 6, 69, 15, 2, 0, 4, 2, 6, 18, 2, 122, 21, 20, 7, 6, 3
6, 55, 16, 19, 0, 1, 0, 20, 11, 6, 12, 93, 0, 0, 25, 9, 14, 0, 3, 11, 33, 70, 18, 13, 20, 13, 23, 63, 1, 12, 58, 26, 1, 4, 94, 3, 1, 133, 31, 8,
0, 9, 73, 4, 0, 1, 2, 3, 14, 77, 1, 7, 11, 0, 8, 0, 13, 14, 2, 33, 7, 25, 1, 63, 5, 4, 11, 41, 9, 24, 11, 9, 39, 7, 0, 45, 13, 27, 8, 13, 12, 19,
 0, 0, 7, 1, 0, 7, 119, 5, 64, 2, 13, 1, 6]
Average degree of all the nodes (average amount of connections): 23.804
(base) annavasconcelos@crc-dot1x-nat-10-239-105-216 finalproj %
```

Test for Part 1:

```
running 1 test
test test_average_distance ... ok

test result: ok. 1 passed; 0 failed; 0 ignored; 0 measured; 3 filtered out; finished in 4.24s

 *  Terminal will be reused by tasks, press any key to close it.
```

Test for Part 2:

```
test pt2::tests::test_main ... ok

test result: ok. 3 passed; 0 failed; 0 ignored; 0 measured; 1 filtered out; finished in 0.33s
```

Resources Used:

Facebook Dataset: https://snap.stanford.edu/data/ego-Facebook.html

BFS: https://www.sotr.blog/articles/breadth-first-search

Assert_approx_eq: https://crates.io/crates/assert_approx_eq

Test cases: https://doc.rust-lang.org/book/ch11-02-running-tests.html

Using Github: https://www.tutorialspoint.com/how-to-upload-a-project-to-github-from-vs-code

Pathfinding: https://docs.rs/pathfinding/latest/pathfinding/

BFS: https://www.geeksforgeeks.org/breadth-first-search-or-bfs-for-a-graph/

BFS:

https://medium.com/nuclei-technology-blog/social-networking-with-bfs-and-neo4j-ea52dc4ce198