Task 3 PageRank

In this project, I implemented the PageRank algorithm. Since a parallel implementation was not explicitly required in the subject, I started by implementing it using a normal approach. We also had the choice between the data-driven and topology-driven approaches. Initially, I tried the topology-driven approach, but I couldn't make it work, so I opted for the data-driven approach instead.

The main part of my program consists of a simple loop that repeatedly calls my PageRank calculation function to compute the PageRank values for all vertices. At the end, I printed the PageRank values of the first 10 vertices and obtained the following results:

PageRanks of the first 10 vertices:

- Vertex 1: 1.922100855280392e-05
- Vertex 254913: 0.0001422976237773444
- Vertex 9: 4.412559952198699e-06
- Vertex 438238: 0.007393709039819704
- Vertex 7: 5.7868160526803805e-06
- Vertex 17: 4.169728563147214e-06
- Vertex 11: 4.381134090219418e-06
- Vertex 5: 2.8968822956153745e-06
- Vertex 2: 4.296976697415173e-06
- Vertex 8: 2.608137110015108e-06

These values are very small, which is understandable since the initial PageRank value is set to 1/V, where V is the total number of vertices, and V is very large.

I also attempted a parallel implementation using Python's multiprocessing library by dividing the PageRank calculation across multiple processors. However, the reassembly of the new PageRank list proved to be quite time-consuming. The more I divided the task, the longer this reassembly step took, as the normal implementation is already quite fast. Nonetheless, it is reassuring to see that both the parallel and sequential versions produced the same PageRank results.