CS141 – Intermediate Algorithms and Data Structures
Assignment 2 – All Pairs Shortest Path

[Carlos Gomez, 860925367]

1 Introduction

- The problem we are solving is trying to find the shortest path between two vertices in a given directed graph with weights.
- We are going to use two different methods to solve the problem we have. The first one will be the Bellman-Ford algorithm that we will use and the second one will be the Floyd-Warshall algorithm.
- Both methods are good to use in their specific application, as both will give you the shortest path of the vertex you are looking for. Bellman-Ford has the capability to detect a negative cycle in a graph, but will take longer to relax all the edges. Floyd-Warshall will relax all the edges faster than the Bellman-Ford and will finish it in a single call of the function.
- Both of them will be used as either one can be used in different situations. The Bellman-Ford algorithm will be used multiple times in order to relax all of the edges of a pair of vertices. The Floyd-Warshall algorithm will be used in one pass of our graph G and relax all the edges, while giving the shortest path of pairs.

2 Bellman-Ford

- The Bellman-Ford algorithm is an algorithm that finds the shortest paths from all the pairs in a graph. It is also an algorithm that detects negative cycles in a given graph and will return a False boolean value if it finds one.
- We are using the Bellman-Ford algorithm to find the shortest path from the source vertex to the sink vertex and doing the same for every vertex until we have relaxed all the edges in the graph.
- I adapted the Bellman-Ford algorithm to work for all pairs by implementing the same single vertex Bellman-Ford algorithm for all vertices. This you can pass a single graph into the function and have it run for all of the vertices in the graph.
- The run time of the Bellman-Ford algorithm as a worst case scenario is $O(|V||E|)$. In my implementation of the Bellman-Ford algorithm, because I apply the algorithm for every vertex in my list of vertices, the worst-case runtime is $O(|V|^2|E|)$

3 Floyd-Warshall

- The Floyd-Warshall algorithm is an algorithm for finding the shortest paths in a weighted graph that can have positive or negative edge weights, but no negative cycles.
- I am using this algorithm in order to find the shortest paths in a weight graph for every source vertex to every sink vertex.
- This algorithm is better than Bellman-Ford as it allows you to find all of the shortest paths of a weighted graph with both positive and negative edge weights, but no negative cycles, in a shorter amount of time and in a single execution.
- The run time of the Floyd-Warshall algorithm is $O(|V|^3)$

Benchmarks	Bellman-Ford		Floyd-Warshall	
	$O(.)$	Actual	$O(.)$	Actual
input1.txt	$O(\|V\|^4)$	7.199e-05	$O(\|V\|^3)$	1.00e-06
input2.txt	$O(\|V\|^4)$	8.599e-05	$O(\|V\|^3)$	9.99e-07

4 Results

- The Bellman-Ford algorithm is a more versatile algorithm than Floyd-Warshall as it is capable of detecting negative cycles in a weighted graph. So if a graph has negative cycles, then using the Bellman-Ford algorithm is a better idea in finding the shortest path. If a graph does not have negative cycles, then using the Floyd-Warshall algorithm is better to use, as it finds all shortest paths in a single execution and faster than Bellman-Ford.
- The theoretical run-time of the Bellman-Ford algorithm is $O(|V|^2|E|) = O(|V|^4)$ and the run-time of the Floyd-Warshall algorithm is $O(|V|^3)$. The Floyd-Warshall will find all the shortest paths of a weighted graph with no negative cycles faster than the Bellman-Ford algorithm.
- The actual run-times are located in the table above, and when it comes to speed the Floyd-Warshall algorithm is better. It is much faster than the Bellman-Ford algorithm when it comes to computing the shortest paths of a graph without negative cycles.

5 Conclusions

- Understanding the algorithms themselves was pretty simple, but actually implementing the algorithms themselves in Python, proved to be difficult. I spent more time trying to understand the language and sytnax of Python, than understanding and using the actual algorithms.
- I learned a solution to find negative cycles in a graph that will prove to be really helpful in the future that I will be able to apply to problems I encounter.
- One real world problem problem for a Bellman-Ford algorithm would be to have the ability to mark and track new routes. When tracking routes you would have the ability to verify if a route will have a negative cycle and leads back to itself so you would be able to verify it. Tracking the fastest route to a destination would require the Floyd-Warshall algorithm as finding the fastest path to two vertices would be faster using Floyd-Warshall.