Johnson's algorithm time complexity

The Bellman-Ford algorithm is used to reweight the graph, and Dijkstra's algorithm is then applied for each vertex in Johnson's algorithm to identify all pairings with the shortest paths in the graph. The network's density, or whether it is a full graph, the distribution of edge weights, and the Bellman-Ford and Dijkstra algorithms' chosen implementations are some of the variables that affect Johnson's algorithm's time complexity.

Bellman-Ford Algorithm: -

* In the worst-case scenario, the time complexity of bellman ford algorithm is O( V . E), where V is the number of vertices and e is the number of edges in the graph.
* However, if the graph is a complete graph, the E will be order of V2 . Hence the time complexity becomes O(V3) for bellman ford algorithm in case of complete graph.

Time Complexity = O(V . E)

Dijkstra’s Algorithm: -

* The time complexity of Dijkstra’s algorithm depends on the data structure used to implement. Since we used priority queue for implementation, the time complexity is O((V + E) log V).
* If simple array is used, the time complexity increases to O( V2 ).
* In Johnson’s algorithm, dijkstra’s algorithm is applied for each vertices, resulting in the time complexity of O(V \* ((V + E)log V)), which can be simplified down to O( V2 log V).

Time complexity = O( V2 log V)

Total time complexity of Johnson’s algorithm: -

* In conclusion, total time complexity of Johnson’s algorithm is simply the sum of time complexities of running Bellman-ford algorithm for one time and Dijkstra’s algorithm for V times.
* Hence, the time complexity of Johnson’s algorithm can be said as

Johnson’s algorithm Time complexity = O( V2 log V + V . E)