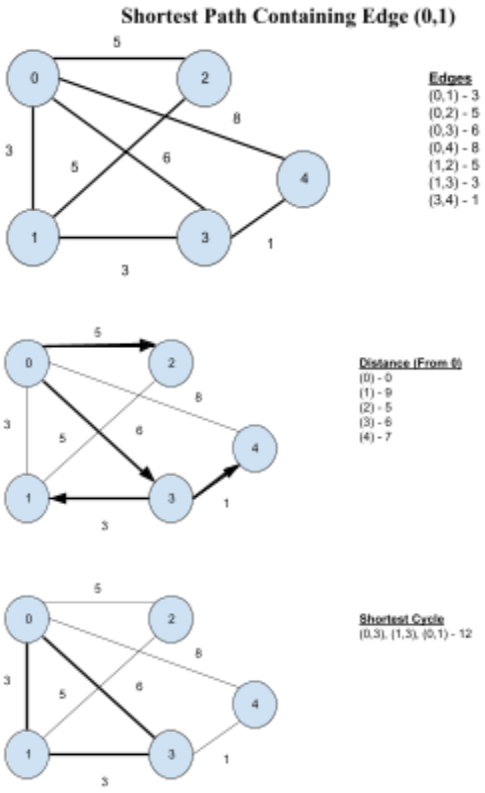
Samuel Sicklick

Solution



This implementation of *ShortestCycle* builds a graph in its constructor by calculating the number of vertices and then adding all of the edges into an *EdgeWeightedGraph*, provided by Sedgewick. The number of vertices is calculated by finding the highest vertice in any of the edges and then adding 1, to account for base 0 indexing. However, at this stage the graph is built without the *edge of interest*, which will be crucial to finding the cycle later. By instantiating and running *DijkstraUndirectedSP*, also from Sedgewick, on the aforementioned graph, the shortest paths are found from one vertex (*v*) of the *edge of interest* to all other vertices. Thus, the *pathTo* method can be called on the other vertex (*w*) of the *edge of interest* to attain the shortest path between them. At this point, the previously omitted *edge of interest* can be added to complete the cycle.

Proof of Correctivity

This implementation simply uses Dijkstra's shortest path algorithm, without internal alteration, to find the shortest path from vertex (*v*) of the *edge of interest* to its other vertex (*w*). This has already been proven, and the addition of the final *edge of interest* to that shortest path definitively completes the shortest cycle since that edge was the source of the two initial vertices used in the algorithm.

Proof of Performance

Since this implementation just uses Dijkstra's shortest path algorithm, which has already been proven to be O(*E log V*), and then adds a remaining edge at O(1), it can be concluded that the performance of this implementation of *ShortestCycle* is O(*E log V*).