**Adjacent location Information Gathering and building a Network Graph:**

**Sayandev Roy**

**Introduction:**

The purpose for the title to gather adjacent node information and cost from its location, and using the node details building network graph.

**Technical Details:**

Below is the diagram to conceptualize the flow:

N2

N1 N3

N4

N5 N6

There are six nodes, each node have encapsulated information some are public, private and protected. Now some nodes have information that will like to share with particular/adjacent nodes. From above diagram N2 shares information with N1 & N3 and N1 shares some information with N5 of N2.

Now after a moment say N1 die the the graph will get reconfigure and N2 will have adjacent N5, below the diagram will be:

N2

N1 dead N3

N4

N5 N6

Similarly N3 dead hence N2 connect to N4:

N2

N1 dead N3 dead

N4

N5 N6

Now to implement the above concept I will be proposing below flow chart:

Insert identifier and distance for all nodes to a priority queue and assign distance Uint for all nodes but for root nodes use zero

Retrieve a node with minimum distance from queue

Queue Empty or distance == max

Yes

No

Get Neighbour for current node

No

New Distance < Old one?

Calculate New Distance of Neighbour; Distance=neighbour metric+ current node distance

Neighbour Found?

Yes

No

Yes

Update Neighbour distance in Queue

**Choice of Data Structure, Simplicity and Efficiency for implementation**:

Planning to use Queue data structure as the shortest path in an un-weighted graph means the smallest number of edges that must be traversed in order to reach the destination in the graph. This is the same problem as solving the weighted version where all the weights happen to be 1. If we use Queue (FIFO) instead of Priority Queue (Min Heap), we get the shortest path in linear time O(|V| + |E|). Basically we do BFS traversal of the graph to get the shortest paths.