Algorithm (Edge Bfs):-

Assuming you have a road-network and to have a proximity metric between 2 zipprs this algorithm can be used, initially both source and destination will be on some road say AMAN, AMCU now at each iteration we will expand our sample space to all the roads directly connected to the roads in our sample space. Now we will check if there is a common road which belongs to both the sample spaces.

Process:-

Iteration=1

Sample space 1:-{AMAN}

Sample space 2:- {AMCU}

Iteration=2

Sample space 1 :- {AMAN,AMO3,AM05,AMAG} #Roads directly attached to AMAN

Sample space 2 :- {AMCU,AMAP,AMAR,AMCV} # Roads directly attached to AMCU

If there is common road in both of the sample spaces we have a solution.

Each road will have a struct

|  |  |
| --- | --- |
| Cost to reach this road from Source Road | Cost to reach this road from Destination Road |
| Predecessor to this road while travelling from source | Predecessor to this road while travelling from destination. |

At each iteration newly added roads predecessor and cost from respective source, destination are entered, If these details are already present and if the new cost is less than cost in the struct we update the details accordingly. If a struct has predecessor for both source, destination then we found a solution. All the structs having a cost greater than this will be deleted.