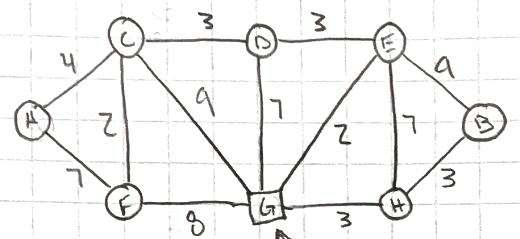
HW # 5

Problem 1.

A. * Best algorithm to find fastest route would be Dijkstra's algorithm.

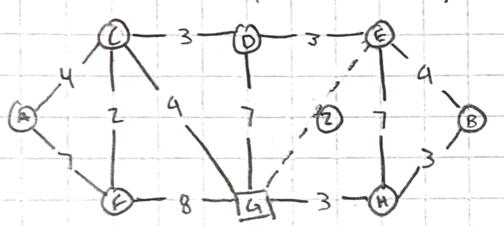
Graph Explanation:

1.

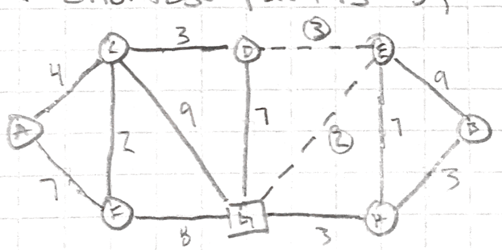


Distribution Center

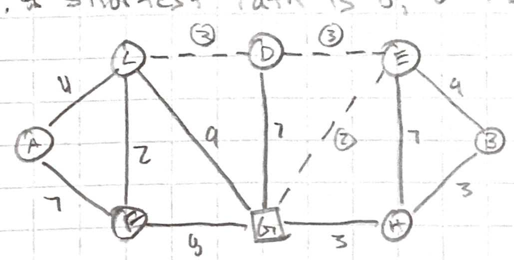
2. * Shortest path is 2, G => E



3. * Shortest path is 5, G => E => D

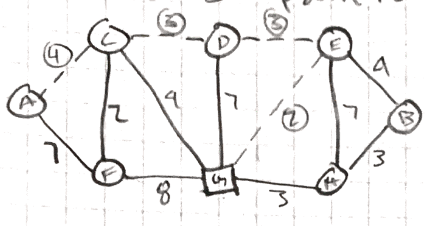


4. * Shortest Path is 8, G => E => D => C

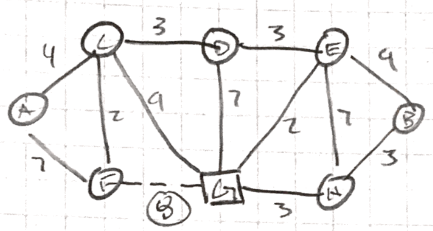


A. continued

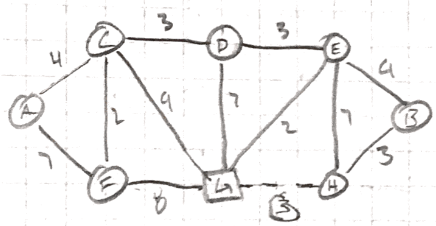
5. *Shortest path is 12, G=>E=>D=>C=>A



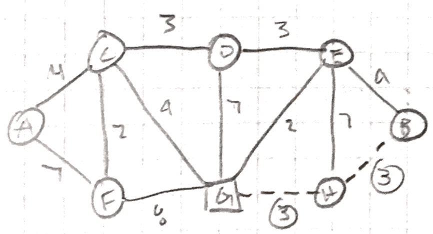
6. *Shortest path is 8, G=>F



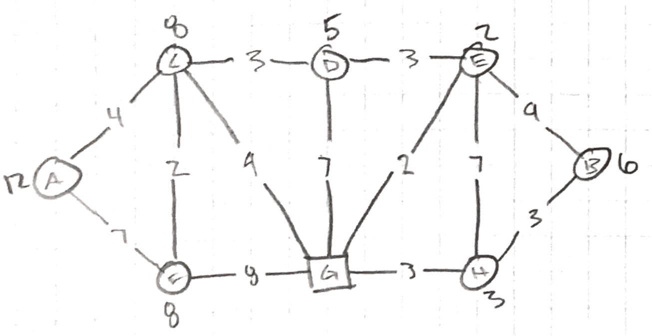
7. *Shortest path is 3, G=>H



8. *Shortest path is 6, G=>H=>B



9. *Shortest path to all from G

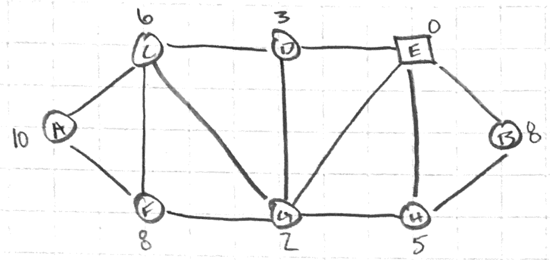


=>

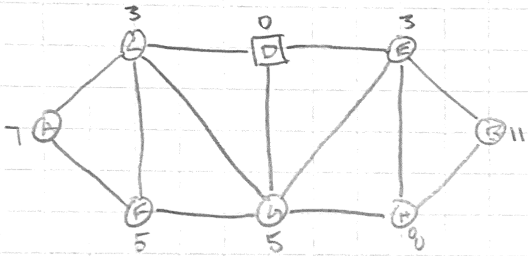
Vertex	Distance	Parent vertex	Shortest path
→G	0		G
A	12	C	G=>E=>D=>C=>A
B	6	H	G=>H=>B
C	8	D	G=>E=>D=>C
D	5	E	G=>E=>D
E	2	G	G=>E
F	8	G	G=>F
H	3	G	G=>H

B.

1. * If center is E, graph with shortest paths



2. * If center is D



* Algorithm for finding optimal solution:

```
Optimal_Location(City, t):

    OptimalLoc ← -1
    sum ← ∞
    //find optimal solution
    For i ← 0 to t-1:
        Localsum ← 0
        Distance[] ← Dijksta(City[][], t, i)
        Roads = Length(Distance)
        //find sum of all roads
        For j ← 0 to Roads - 1:
            Localsum ← Localsum + Distance[j]
        //End for Loop
        if sum > Localsum then:
            sum ← Localsum
            OptimalLoc ← i
        //End if
    // End for loop
    return optimalLoc
```

* time complexity:

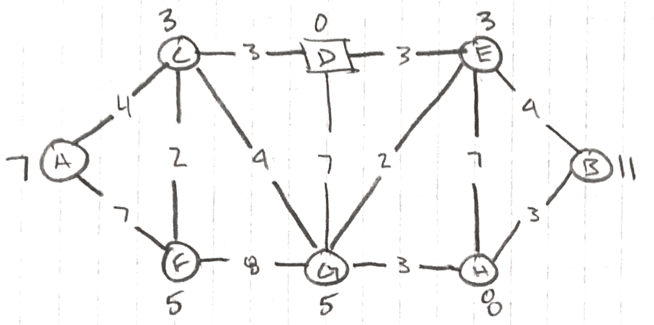
- Dijkstra's algo. runs t times

- time Comp. $= t\,O(r + t(\log t))$

- Converts to, $\boxed{O(tr + t^2(\log t))}$

C.

* Optimal town for distribution center is D



* Cost of spanning tree of citygraph is 20

Vertex	Distance	Parent vertex	Shortest Path
↳D	0		D
A	7	C	D => C => A
B	11	H	D => E => G => H => B
C	3	D	D => C
E	3	D	D => E
F	5	C	D => C => F
G	5	E	D => E => G
H	8	G	D => E => G => H

D. * My suggestion would be C & H

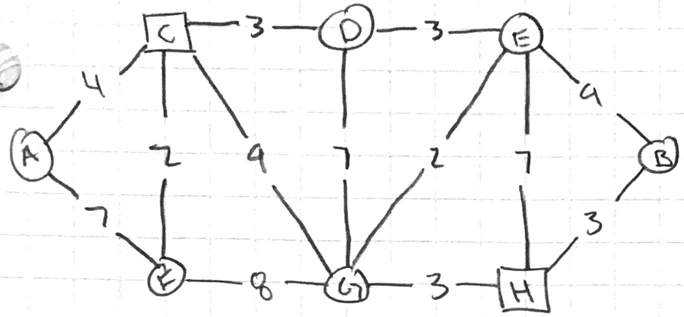
- For each pair (i, j) run dijkstra's algorithm and track the shortest distances D_i & D_j

$$D_i = \text{Dijkstra}(i)$$

$$D_j = \text{Dijkstra}(j)$$

- Compare the distances to the two vertices and track the smallest in the shortest path
- Record maximum path among the shortest paths
- Return a pair (x, y) with the smallest maximum Distance to a town as required optimal Locations to place distribution Centers.

E.



* Running time:

- Running time of dijkstra $= O(t^2)$ & there or $O(t^2)$ possible pairs so,
- Running time $= O(t^2) \times O(t^2) = O(t^4)$
- C & H have the smallest maximum distance to a location.

Problem 2:

* Algorithm:

Func(grap, Vertices)

Create list visited to check vertex is visted or not
Assign distance of every node

Assign 1st node distance to 0

Repeat below until all nodes are visited
Pick vertex thats unvisited & distance min. & add to list
For all adjacent vertizies to the current visited list

Update the distance

* time complexity $= O(n^2)$, n = # of vertices