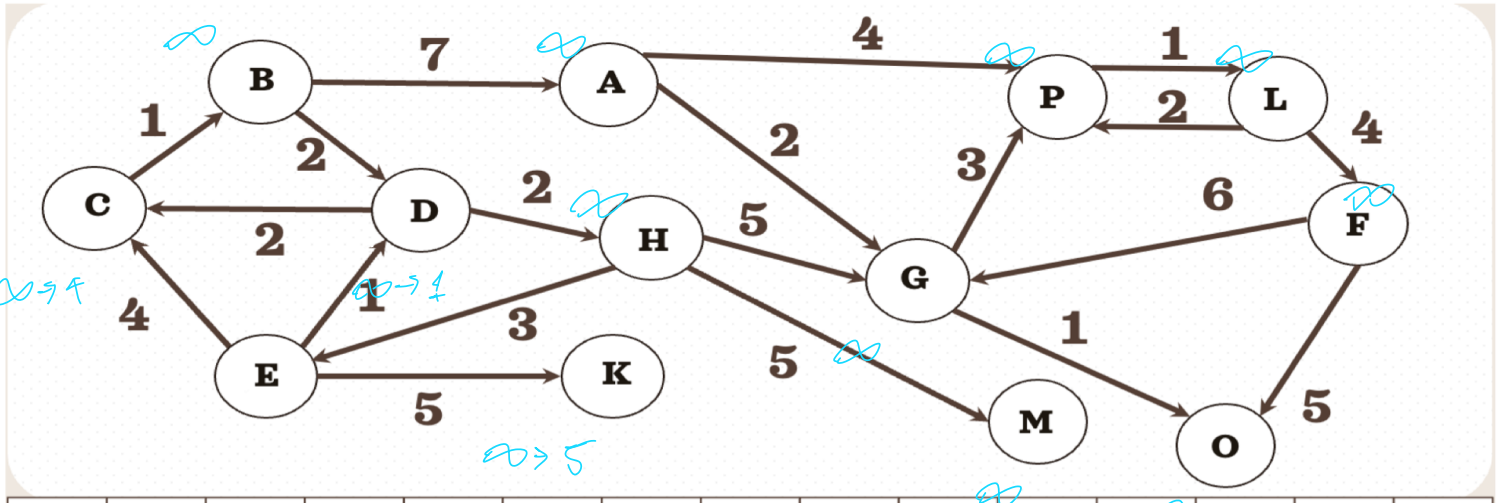


1 (text) Dijkstra [15 points] Find the shortest path from E to all other nodes using the Dijkstra Algorithm (Show your work and intermediate steps)

From to Cost

Visited: [ ] E → E : 0 → Set all to ∞,
E → C : 4 go to the cheapest cost (D)
E → K : 5
E → D : 1

then [E, D]
ED → H : 3 } choose C
ED → C : 3 }

⇒ [E, D, C]
EDC → B : 4 → [E, D, C, B]
EDCB → A : 11 ⇒ [E, D, C, B, A]

⇒ EDCBA → P : 15
EDCBA → G : 13

⇒ Choose H → [E, D, H]
EDH → G : 8 } choose G
EDH → M : 8 }

EDHG → P : 11 } ⇒ [E, D, H, G, O]
EDHG → O : 9 }

EDHGO → Done

From all this, the shortest path is:
E → D : 1
E → C : 3 (E, D, C)
E → K : 5 (E K)

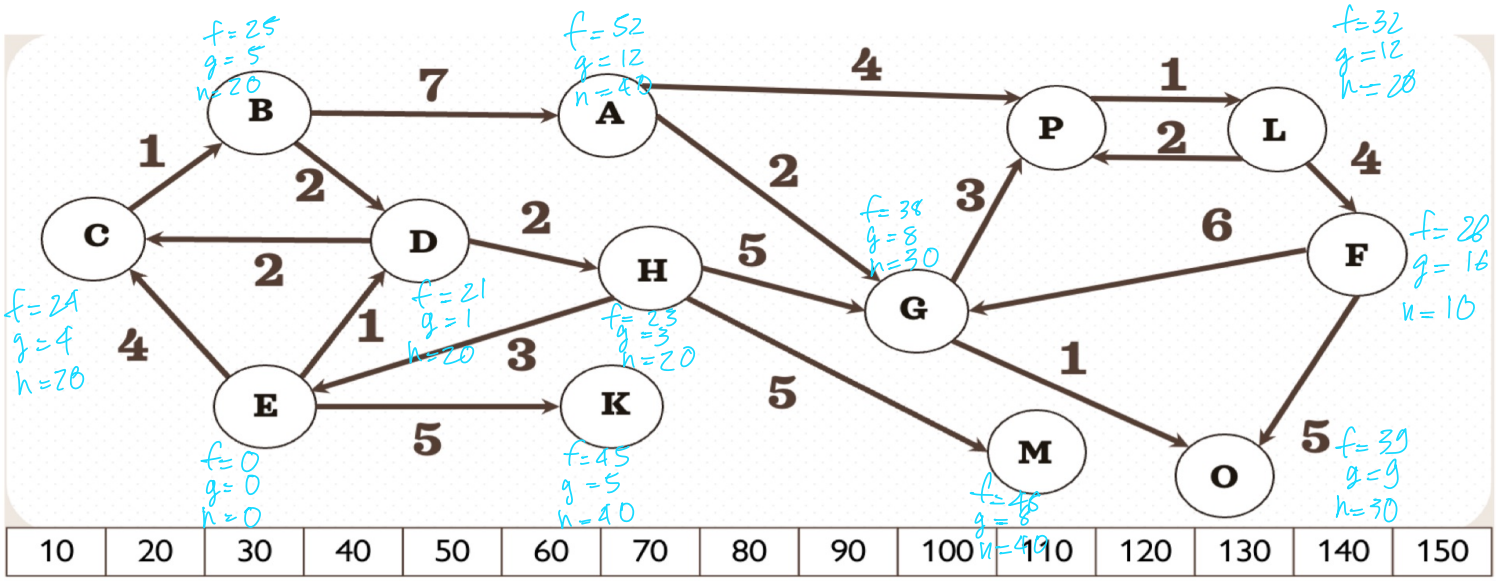
E → H : 3 EDH
E → B : 4 EDCB
E → G : 8 EDHG
E → M : 8 EDHM
E → P : 11 EDHEP
E → O : 9 EDHGO
E → L : 12 EDHEP
E → F : 16 EDHEPLF
E → A : 11 EDCBA

[EDHEP]
EDHEP → L : 12 ⇒ [EDHEPL]

EDHGPL → F : 16

⇒ [EDHGPLF]

2 (text) A* [20 points] Find the shortest path from E to all other nodes using the A* Algorithm (Show your work and intermediate steps) **Hint:** Use the geographical location values to derive the heuristic to use.



1 (text) Dijkstra [15 points] Find the shortest path from E to all other nodes using the Dijkstra Algorithm (Show your work and intermediate steps)

3(text) Comparison

We can see that Dijkstra and A* both locate node "H" from E. They both the same # of iterations, even though A* is known to be faster. This isn't the case since they run the same amount of time

7. Text

Time Complexity for Graph Direct check is $O(V^2)$ because it takes n x n matrix, which we can look at how it look in picture

$$n \begin{bmatrix} \cdots \\ \vdots \\ \cdots \end{bmatrix}_{n} \Rightarrow n \times n \Rightarrow O(V^2)$$

Space is also $O(V^2)$ because it needs to save it down as a matrix n x n

For Path Finder:

Time Complex: $O(V+E)$ where n is # of vertices. Because that we use DFS to explore and worst case of DFS is $O(V+E)$

Space also $O(V+E)$ since pathlist take up V space and visitset would store all the V and they all visited E in total.