Prevention of Sexual Harassment through a Route Search Algorithm using Near Search

UNIVERSIDAD EAFIT

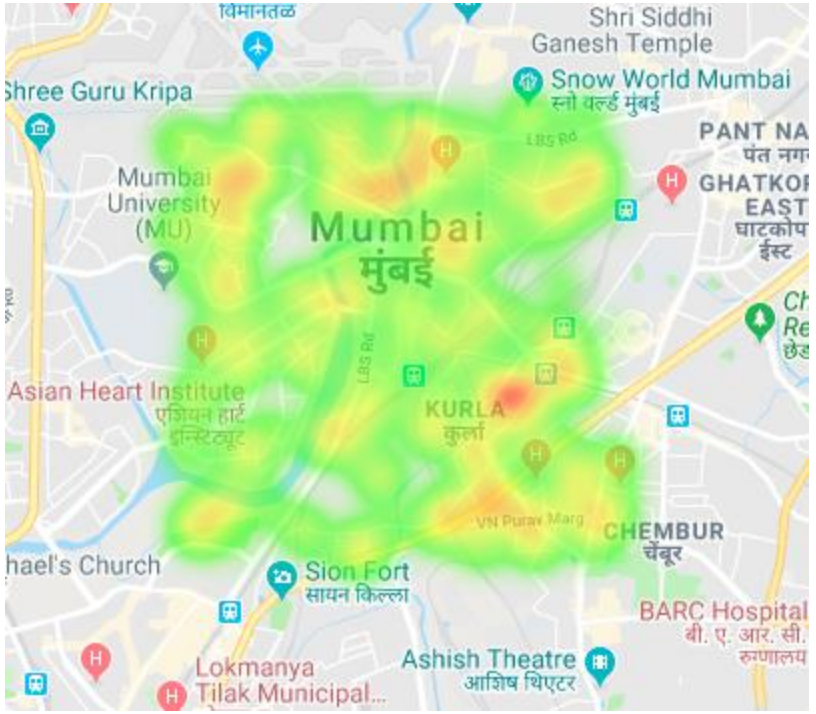
Presentación del equipo

Juan Pablo Rueda Vera
Recopilador de Information

Sebastian Vasquez Saldarriaga
Investigación de la información

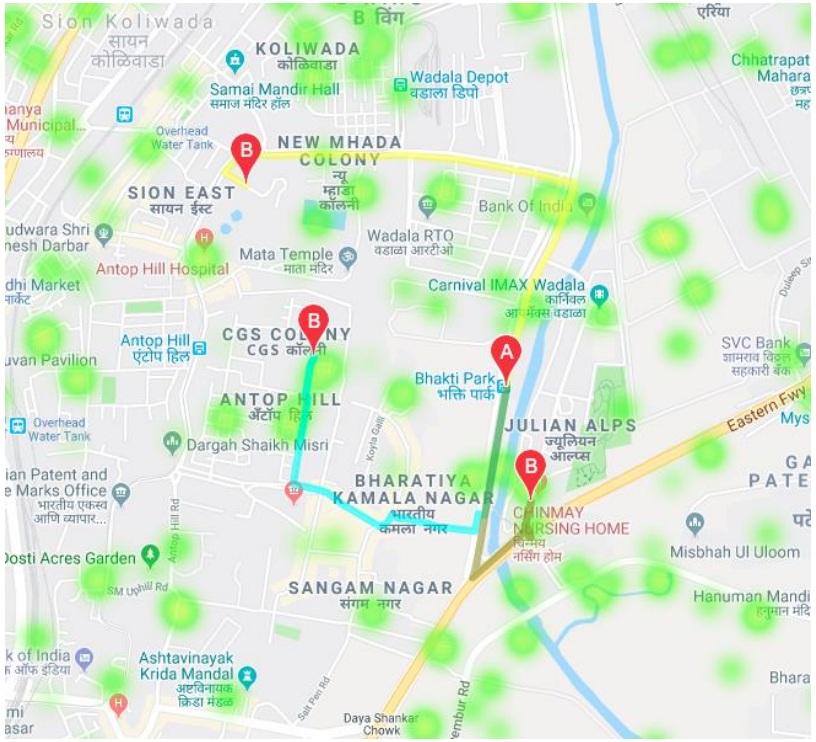
Algoritmo de búsqueda de rutas para prevenir el acoso sexual mediante la búsqueda cercana

Los resultados de este estudio de caso dependen de trabajos previos sobre mapas de calor, que predicen lugares con alto riesgo de incidentes de acoso sexual



Algoritmo de búsqueda de rutas para prevenir el acoso sexual mediante la búsqueda cercana

A continuación necesitamos encontrar "lugares seguros" cercanos y direcciones para caminar a esos lugares utilizando un algoritmo de búsqueda de caminos para prevenir casos de acoso sexual



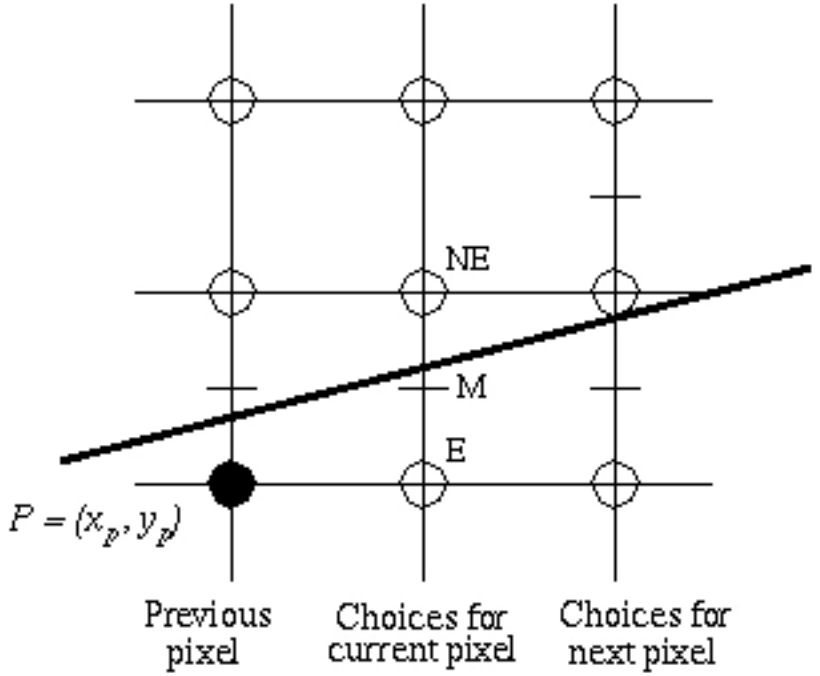
¿Cómo podemos determinar la seguridad de las rutas?

- ✓ Determinar la seguridad general de la ruta únicamente por la puntuación de riesgo asociada con el destino
- ✓ Calcular el promedio de las puntuaciones de riesgo en función dela cobertura de la cuadrícula de la ruta en línea recta desde el origen hasta el destino
- ✓ Calcule el promedio de los puntajes de riesgo en función de la cobertura de la red década paso de la ruta para llegar al destino

$$d(a,b) = \sqrt{(a_1 - b_1)^2 + (a_2 - b_2)^2 + \ldots + (a_n - b_n)^2}$$

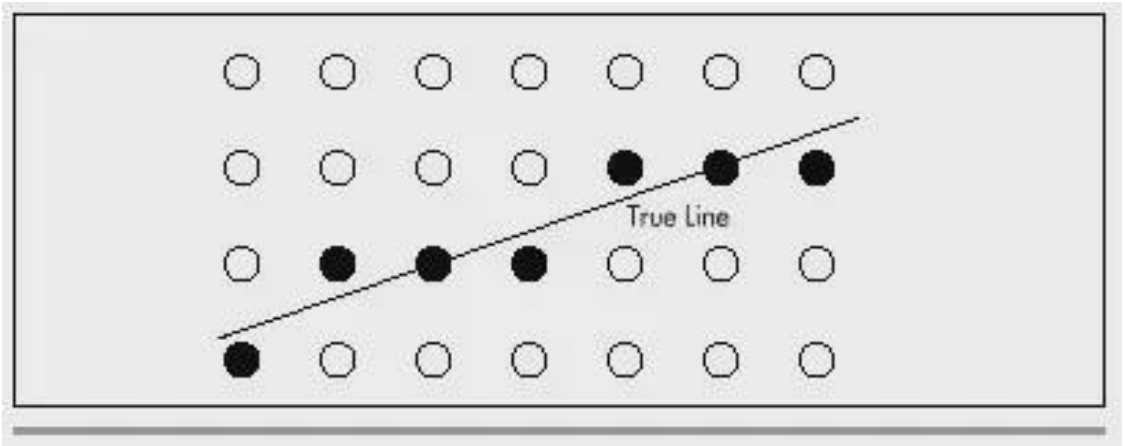
Demostración del algoritmo de dibujo lineal de Bresenham

La idea general detrás de este algoritmo es: dado un punto final inicial de un segmento de línea, el siguiente punto de cuadrícula que atraviesa para llegar al otro punto final se determina evaluando dónde cruza el segmento de línea en relación con el punto medio (por encima o por debajo) de las dos posibles opciones de puntos de cuadrícula.



Puntos de cuadrícula sombreados usando el algoritmo de Bresenham

Es un algoritmo de dibujo lineal digital y fue inventado por Bresenham en el año 1962 y es por eso que tiene el mismo nombre. Este algoritmo es más preciso y utiliza la resta y la suma para calcular el valor del píxel al dibujar la línea. La precisión del algoritmo de Bresenham es confiable al dibujar curvas y círculos también. Veamos cómo funciona este algoritmo



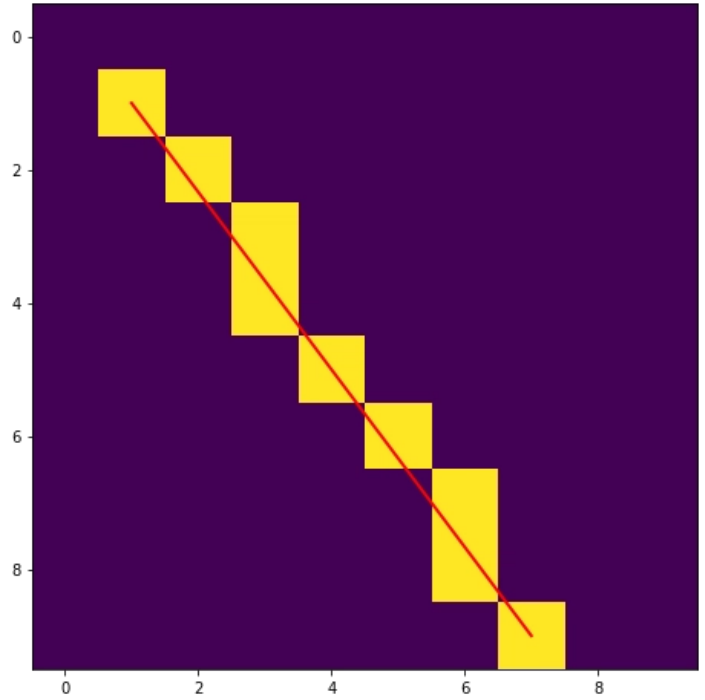
Planteamiento del problema

```
from skimage.draw import line
import matplotlib.pyplot as plt
# create array of zeros
example_arr = np.zeros((10, 10))
# set endpoints for line segment
x0, y0 = 1, 1
x1, y1 = 7, 9
# draw the line
rr, cc = line(y0, x0, y1, x1)
example_arr[rr, cc] = 1
# plot it out
plt.figure(figsize=(8, 8))
plt.imshow(example_arr, interpolation='nearest', cmap='viridis')
plt.plot([x0, x1], [y0, y1], linewidth=2, color='r')
```

Aquí hay un pequeño fragmento de cómo se ve el código de skimage ya tiene esta implementación

Ilustración del método de dibujo lineal de Bresenham usando skimage y matplotlib

Como podemos ver, las aproximaciones del sombreado del punto de cuadrícula son imperfectas, ya que definitivamente hay puntos de cuadrícula que la línea ha cruzado que no están sombreados.

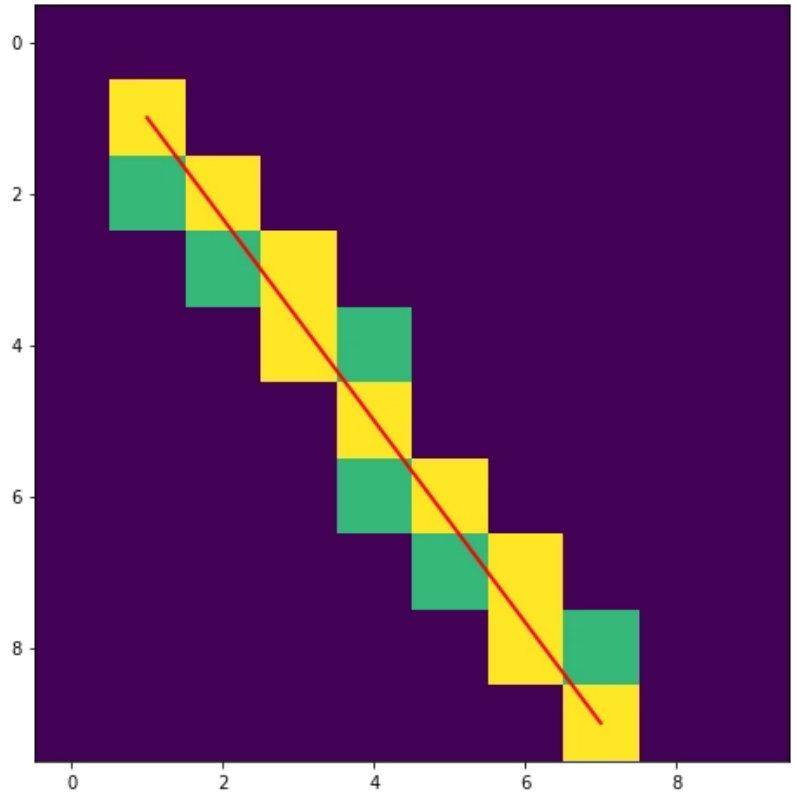


Método para encontrar la «supercubierta» de un segmento de línea

Pero aquí es donde entra el otro algoritmo de búsqueda de caminos para mejorar esto donde todos los puntos que se cruzan están sombreados

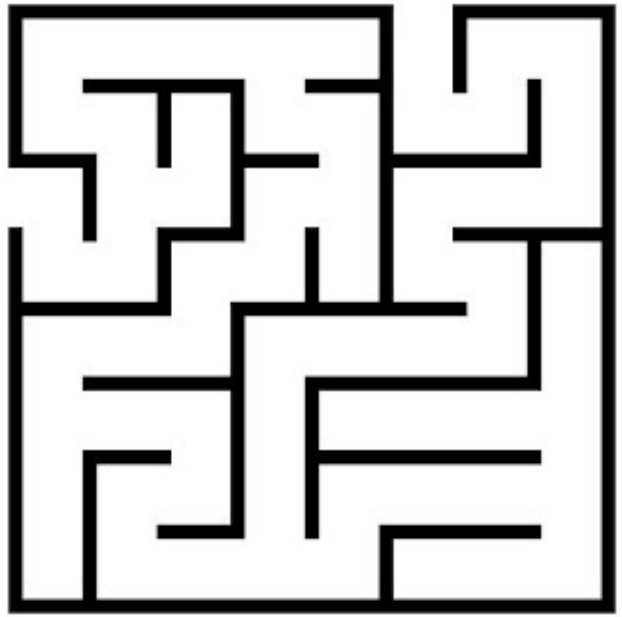
Aplicamos estos métodos de cobertura de cuadrícula para calcular el promedio de puntajes de riesgo para cada ruta y asi determinar la mejor.

Priorizamos la seguridad de la ruta primero, antes de tener en cuenta la distancia (es decir, la ruta más segura se sugiere primero si dos rutas están vinculadas en seguridad, luego sugerimos el destino más cercano), utilizando el análisis de mapas de calor para prevenir casos de acoso sexual.



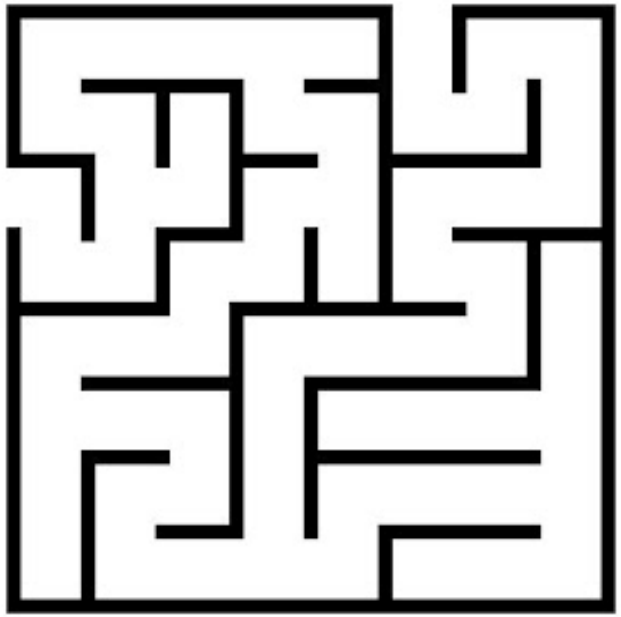
Encuentra el camino más corto en un laberinto

Dado un laberinto en forma de matriz rectangular binaria, encuentre la longitud del camino más corto en el laberinto desde una fuente dada hasta un destino dado. El camino solo se puede construir a partir de celdas que tienen valor 1, y en cualquier momento, solo podemos mover un paso en una de las cuatro direcciones



Los movimientos válidos son:

Go Top: (x, y) ——> (x – 1, y)
Go Left: (x, y) ——> (x, y – 1)
Go Down: (x, y) ——> (x + 1, y)
Go Right: (x, y) ——> (x, y + 1)

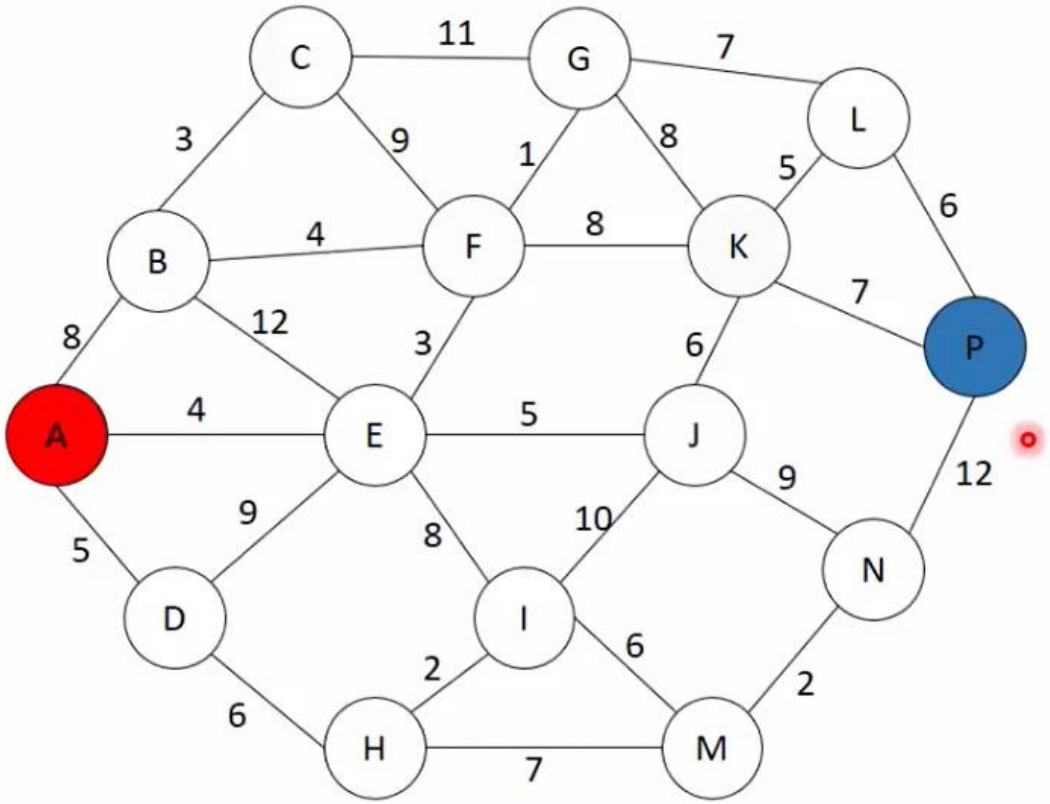


Planteamiento del problema

Por ejemplo, considere la siguiente matriz binaria. Si origen = y destino =, la ruta más corta desde el origen hasta el destino
(0, 0)(7, 5)

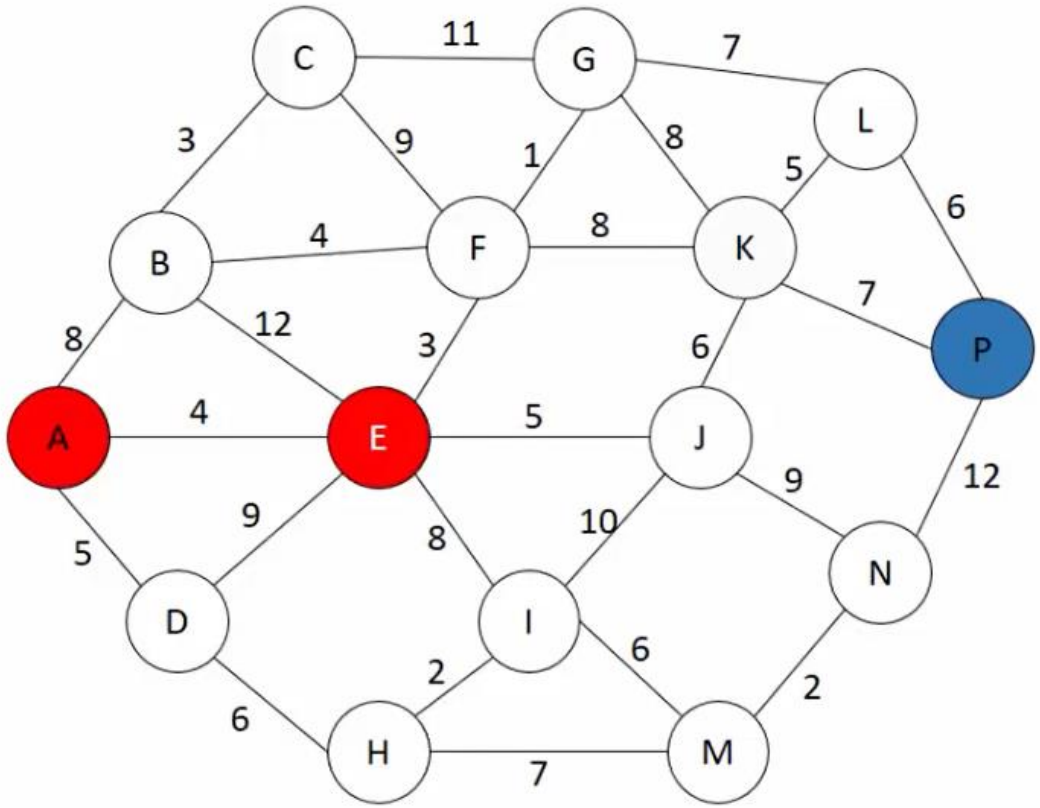
```
[ 1  1  1  1  1  0  0  1  1  1 ]
[ 0  1  1  1  1  1  0  1  0  1 ]
[ 0  0  1  0  1  1  1  0  0  1 ]
[ 1  0  1  1  1  0  1  1  0  1 ]
[ 0  0  0  1  0  0  0  1  0  1 ]
[ 1  0  1  1  1  0  0  1  1  0 ]
[ 0  0  0  0  1  0  0  1  0  1 ]
[ 0  1  1  1  1  1  1  1  0  0 ]
[ 1  1  1  1  1  0  0  1  1  1 ]
[ 0  0  1  0  0  1  1  0  0  1 ]
```

Encontrar el camino mas corto de A->P



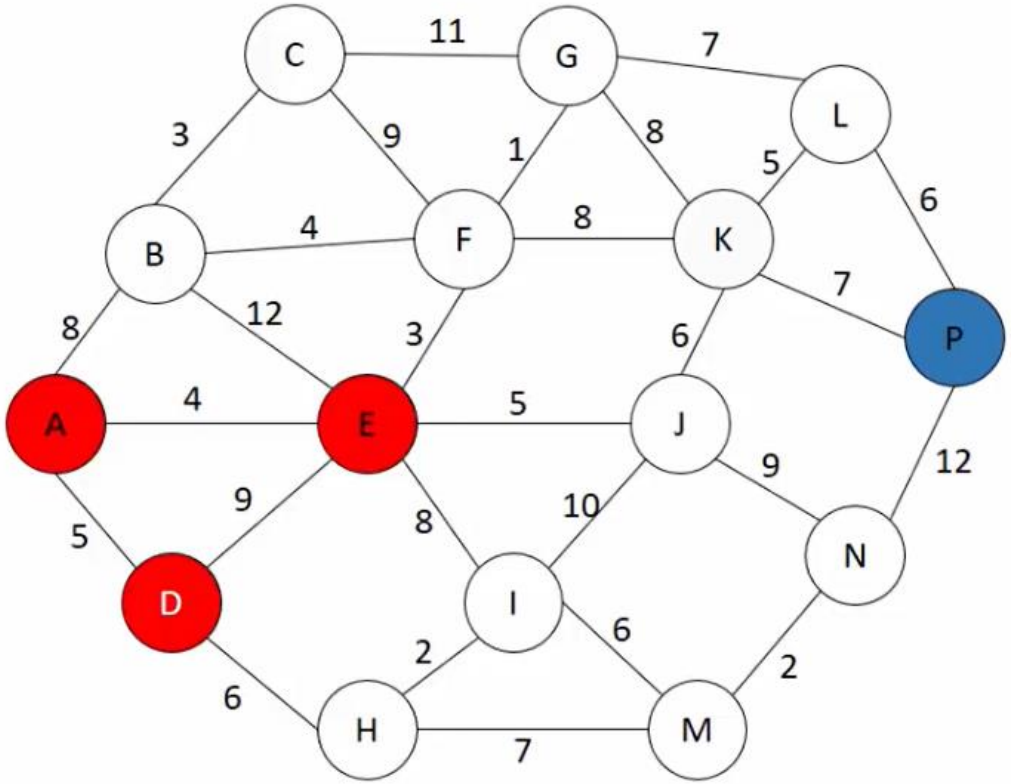
Vertice	Final	Temporal
A		
B		
C		
D		
E		
F		
G		
H		
I		
J		
K		
L		
M		
N		
P		

Encontrar el camino mas corto de A->P



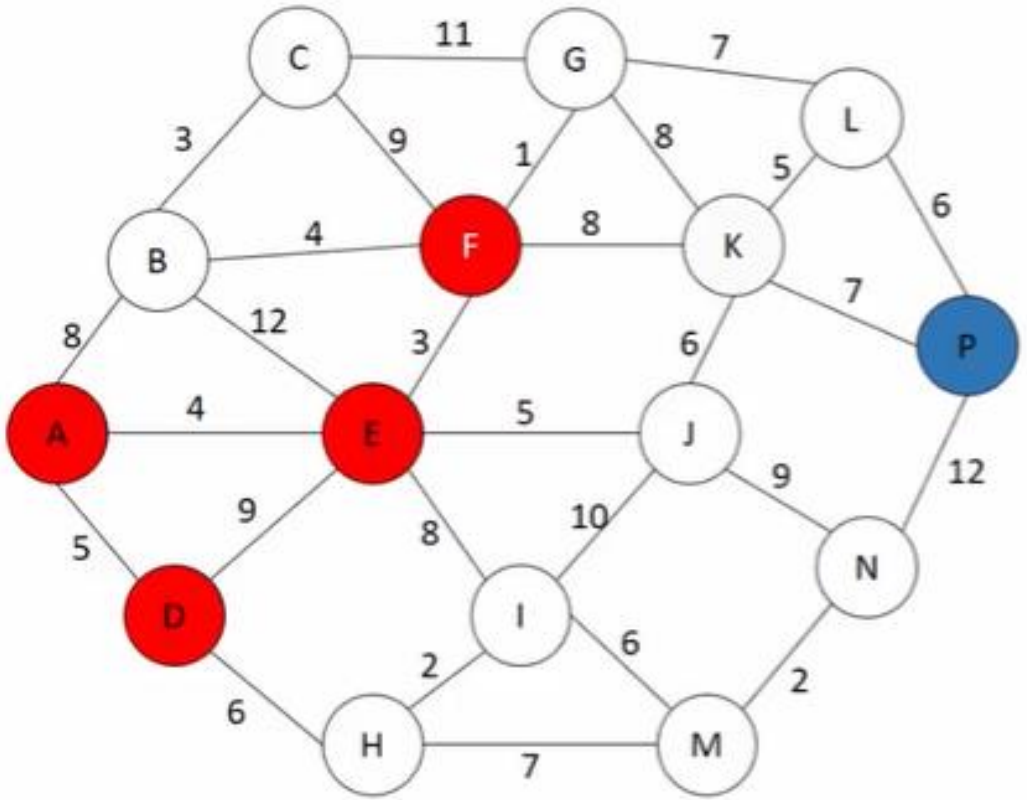
Vertice	Final	Temporal
A	0	0
B		8
C		
D		5
E	4	4
F		7
G		
H		
I		12
J		9
K		
L		
M		
N		
P		

Encontrar el camino mas corto de A->P



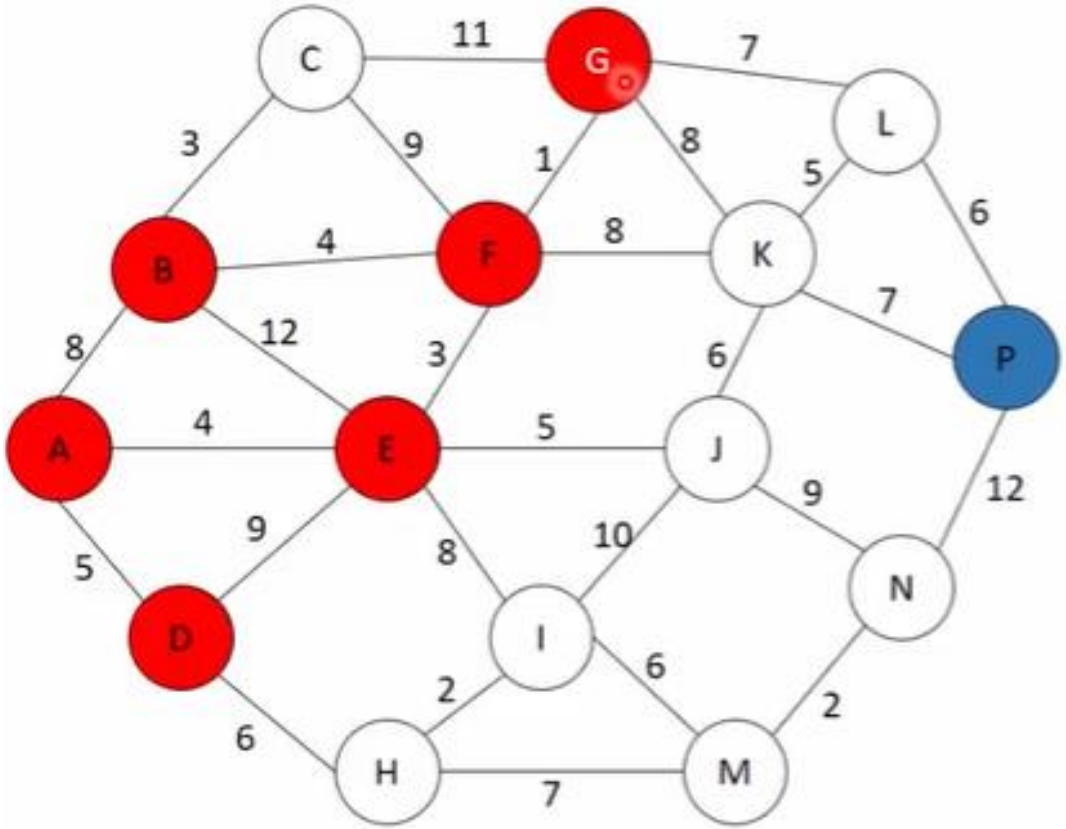
Vertice	Final	Temporal
A	0	0
B		8
C		
D	5	5
E	4	4
F		7
G		
H		
I		12
J		9
K		
L		
M		
N		
P		

Encontrar el camino mas corto de A->P



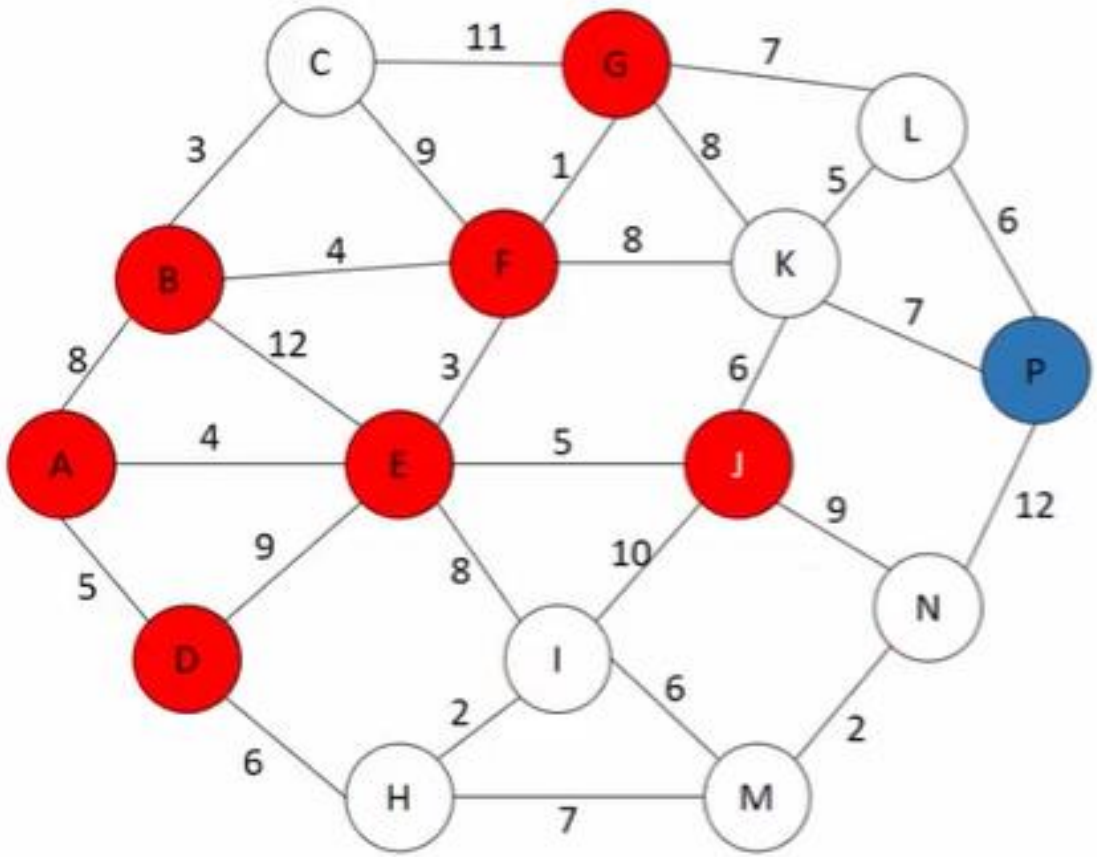
Vertice	Final	Temporal
A	0	0
B		8
C		
D	5	5
E	4	4
F	7	7
G		
H		11
I		12
J		9
K		
L		
M		
N		
P		

Encontrar el camino mas corto de A->P



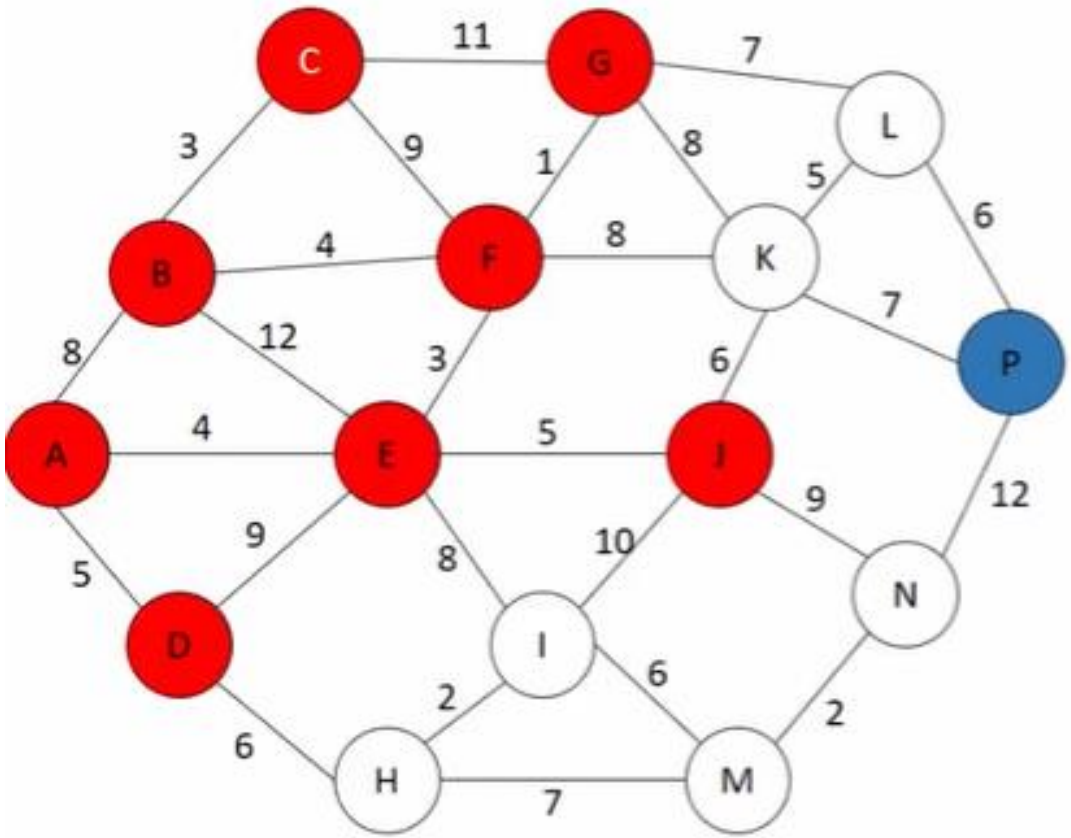
Vertice	Final	Temporal
A	0	0
B	8	8
C		11
D	5	5
E	4	4
F	7	7
G	8	8
H		11
I		12
J		9
K		15
L		
M		
N		
P		

Encontrar el camino mas corto de A->P



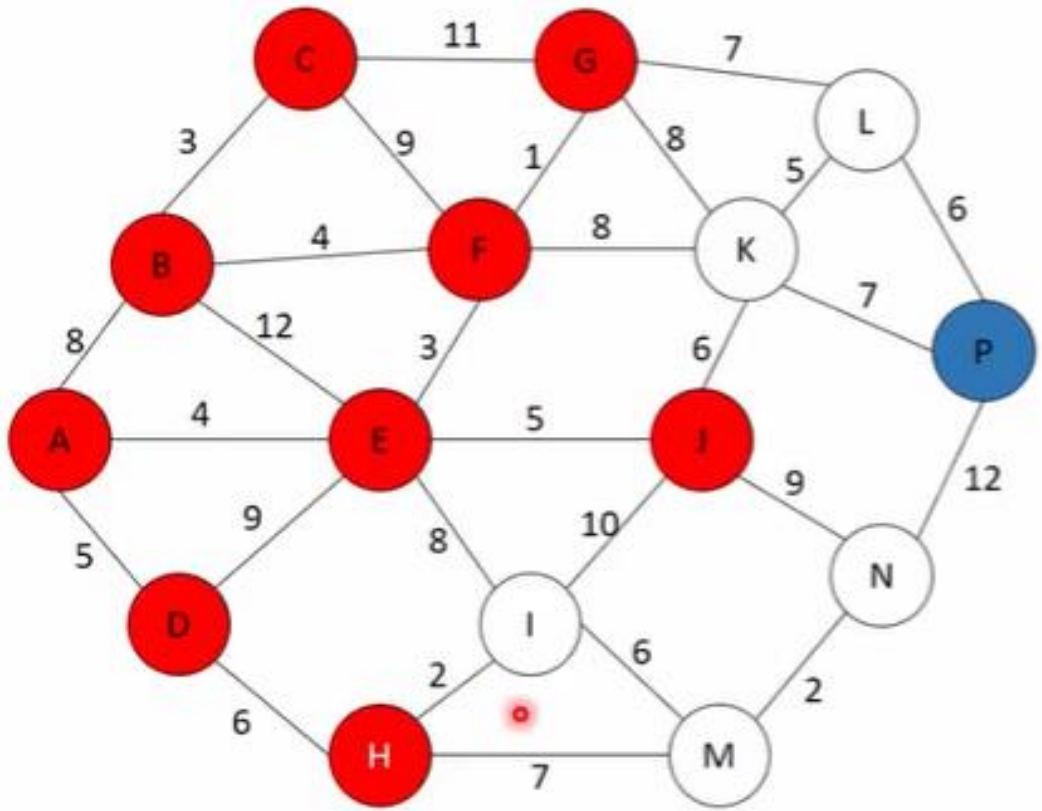
Vertice	Final	Temporal
A	0	0
B	8	8
C		11
D	5	5
E	4	4
F	7	7
G	8	8
H		11
I		12
J	9	9
K		15
L		15
M		
N		
P		

Encontrar el camino mas corto de A->P



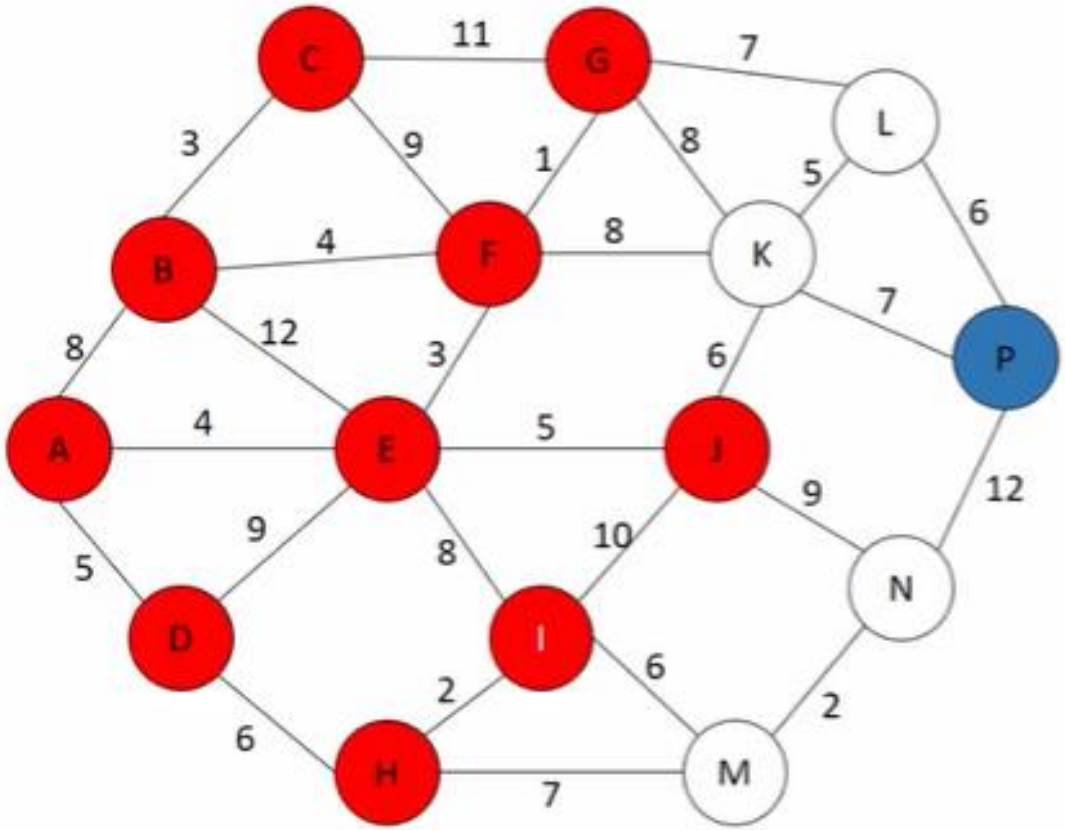
Vertice	Final	Temporal
A	0	0
B	8	8
C	11	11
D	5	5
E	4	4
F	7	7
G	8	8
H		11
I		12
J	9	9
K		15
L		15
M		
N		18
P		

Encontrar el camino mas corto de A->P



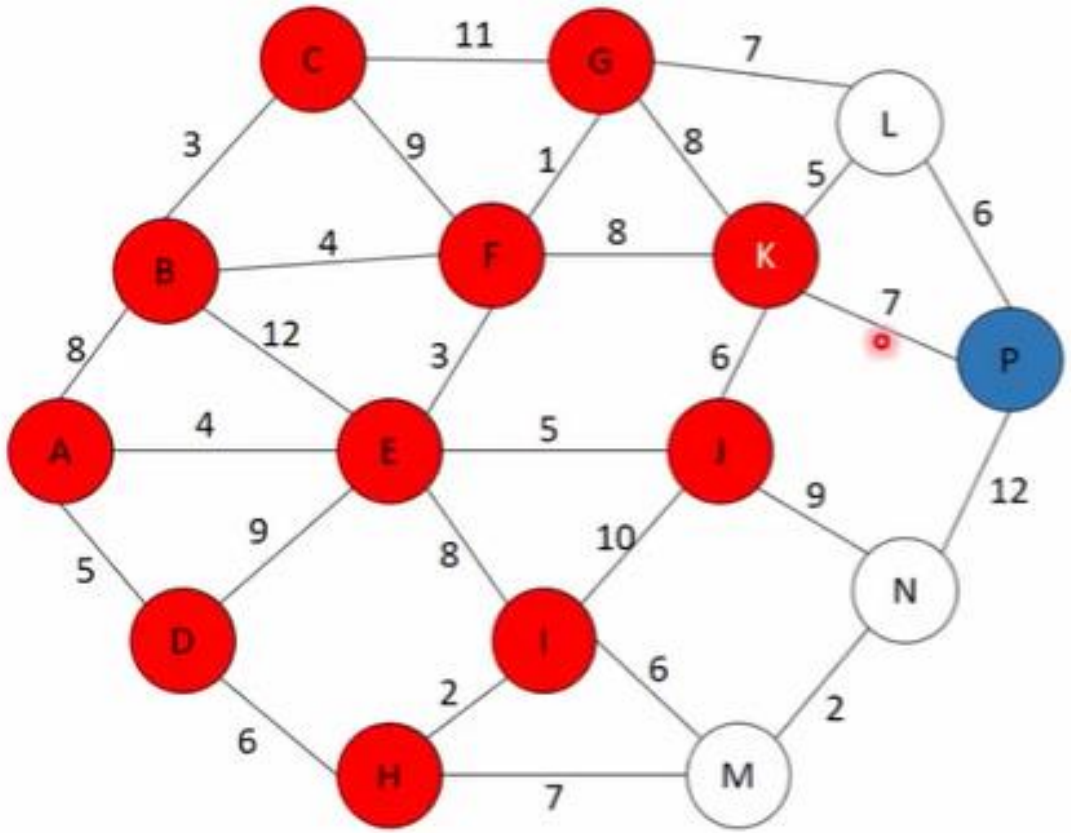
Vertice	Final	Temporal
A	0	0
B	8	8
C	11	11
D	5	5
E	4	4
F	7	7
G	8	8
H	11	11
I		12
J	9	9
K		15
L		15
M		
N		18
P		

Encontrar el camino mas corto de A->P



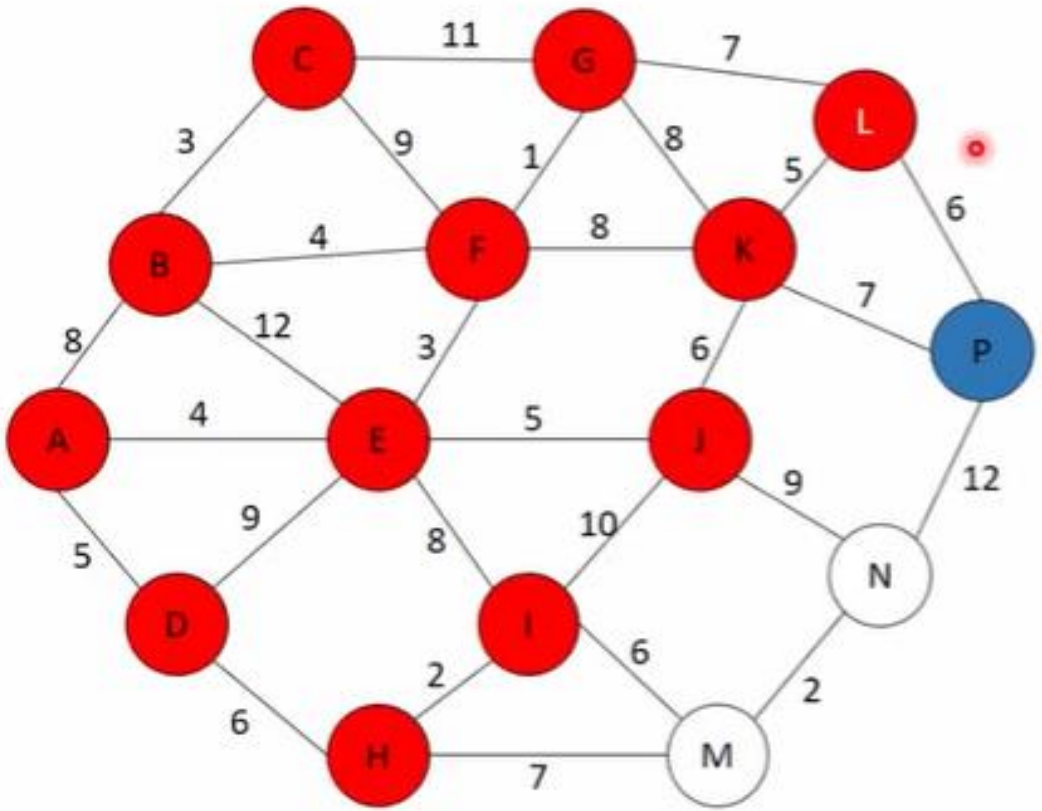
Vertice	Final	Temporal
A	0	0
B	8	8
C	11	11
D	5	5
E	4	4
F	7	7
G	8	8
H	11	11
I	12	12
J	9	9
K		15
L		15
M		18
N		18
P		

Encontrar el camino mas corto de A->P



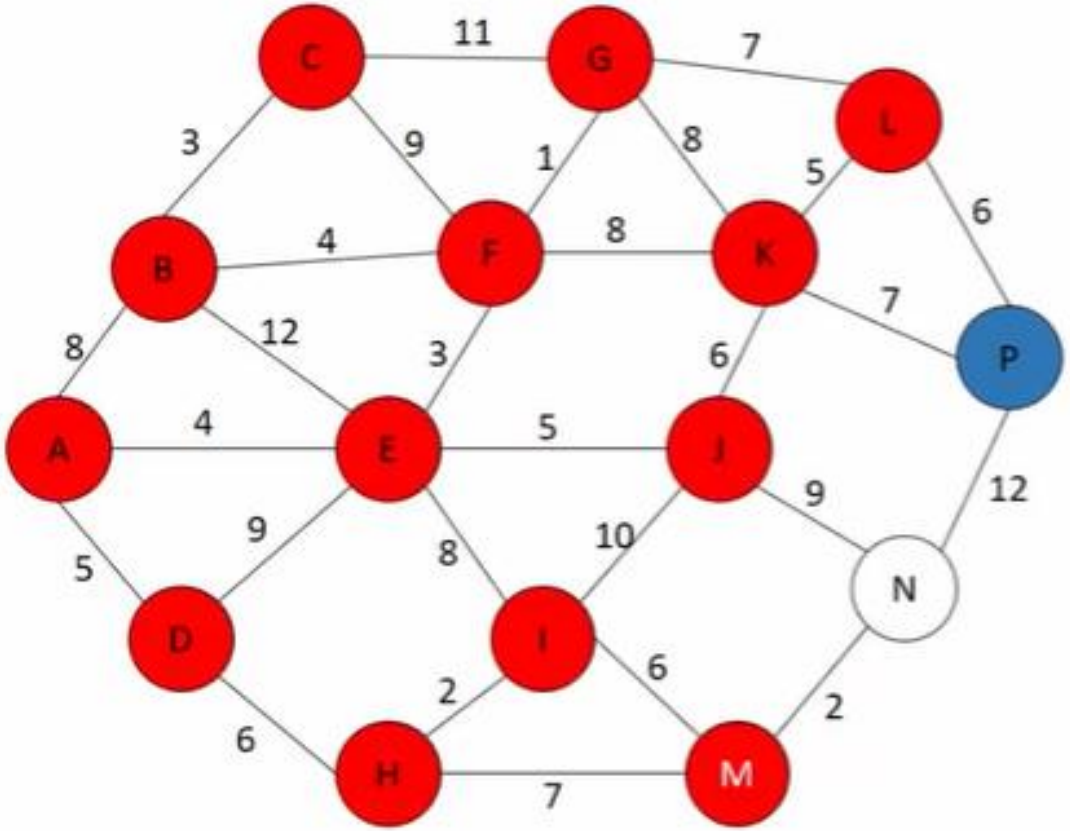
Vertice	Final	Temporal
A	0	0
B	8	8
C	11	11
D	5	5
E	4	4
F	7	7
G	8	8
H	11	11
I	12	12
J	9	9
K	15	15
L		15
M		18
N		18
P		

Encontrar el camino mas corto de A->P



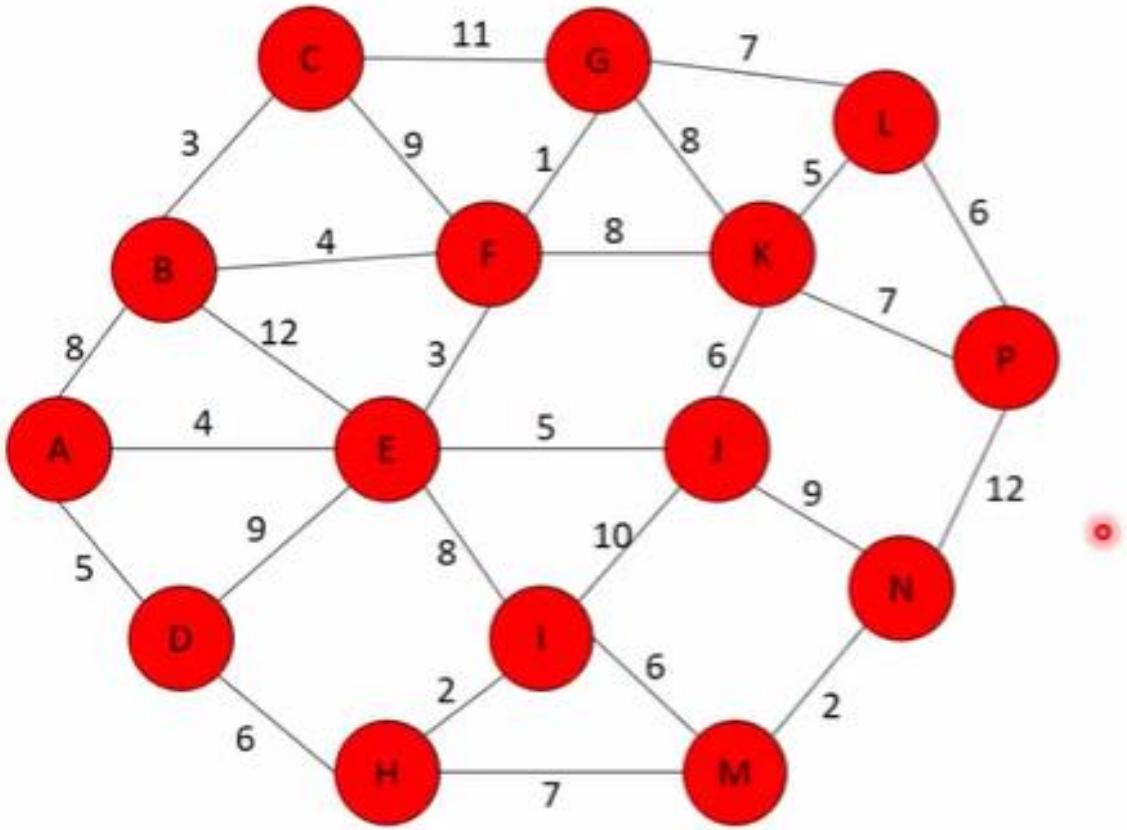
Vertice	Final	Temporal
A	0	0
B	8	8
C	11	11
D	5	5
E	4	4
F	7	7
G	8	8
H	11	11
I	12	12
J	9	9
K	15	15
L	15	15
M		18
N		18
P		22

Encontrar el camino mas corto de A->P



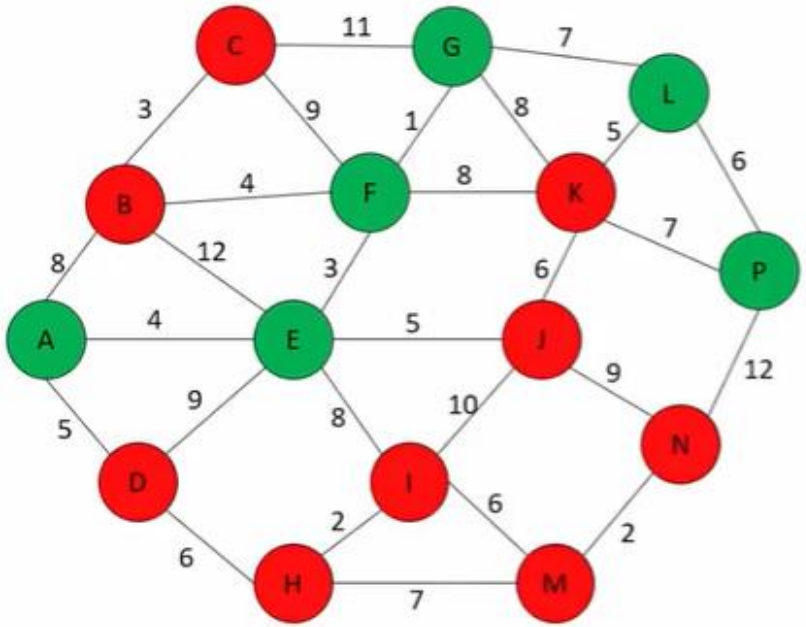
Vertice	Final	Temporal
A	0	0
B	8	8
C	11	11
D	5	5
E	4	4
F	7	7
G	8	8
H	11	11
I	12	12
J	9	9
K	15	15
L	15	15
M	18	18
N		18
P		21

Encontrar el camino mas corto de A->P

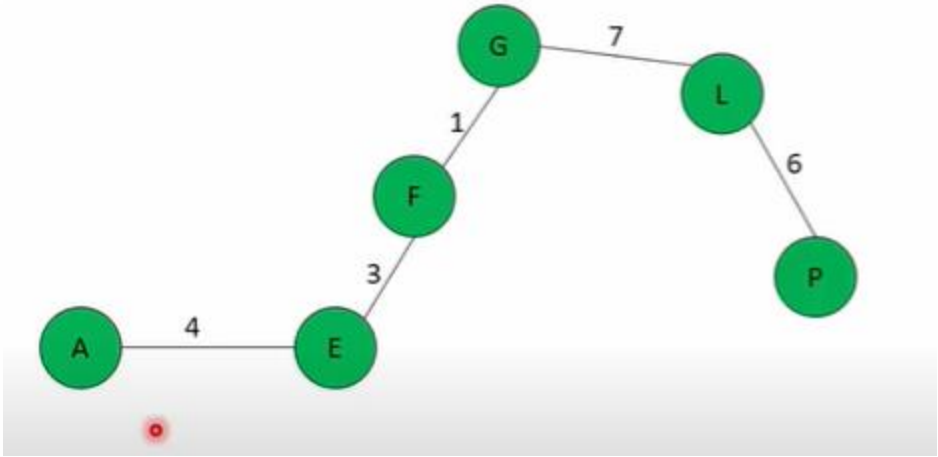


Vertice	Final	Temporal
A	0	0
B	8	8
C	11	11
D	5	5
E	4	4
F	7	7
G	8	8
H	11	11
I	12	12
J	9	9
K	15	15
L	15	15
M	18	18
N	18	18
P	21	21

Costo Minimo: 21



Vertice	Final	Temporal
A	0	0
B	8	8
C	11	11
D	5	5
E	4	4
F	7	7
G	8	8
H	11	11
I	12	12
J	9	9
K	15	15
L	15	15
M	18	18
N	18	18
P	21	21



¡GRACIAS!