Universidad industrial de Santander

Optimización en los recorridos dentro del mapa de la univeridad

Dylan Francisco Jimenez Sandoval
2202049
Dilan Esteban Rey Sepulveda
2192097
Juan Camilo Mantilla Ramirez
2202050

Planteamiento del problema

Al iniciar como estudiante de la Universidad Industrial de Santander (UIS) o al visitar esta institución educativa lo que más llega a preocupar es la movilización, no solo se busca la manera mas eficaz de moverse por la universidad si no también, la manera mas rápida, para esto se busca crear un grafo el cual pueda dar esta información de forma eficaz el como movilizarse por la universidad en el tiempo mas corto, de esta manera, logrando que el usuario se siente mas tranquilo dentro de la universidad.

Al principio se pensó en un método mas ortodoxo como es tomar las medidas de un punto a otro, pero además de gastar mucho tiempo, los datos pueden llegar a tener un margen de error al momento de emplear la toma de estos, por ende se decidió el crear un programa el cual pueda crear un grafo mostrando las rutas usando como guía la plataforma de Google Maps, para la creación y toma de distancias entre la entrada y los distintos sitios de interés.

Analisis del problema

Para abordar este problema es necesario tener un mapa de la universidad, en donde se ubiquen sus respectivos lugares de interés para de esta manera generar u grafo dirigido.

A cada lugar de interes se e asigna un numero, el cual sera su representacion en el conjunto de vertices.

Mapa usado


N
O
E
S
16
48
29
27
28
22
15
8
7
33
21
14
26
25
20
13
46
6
32
12
4
24
19
40
42
31
3
18
11
44
36
34
45
23
30
17
10
2
9
39
41
43
47
38
37
1
35

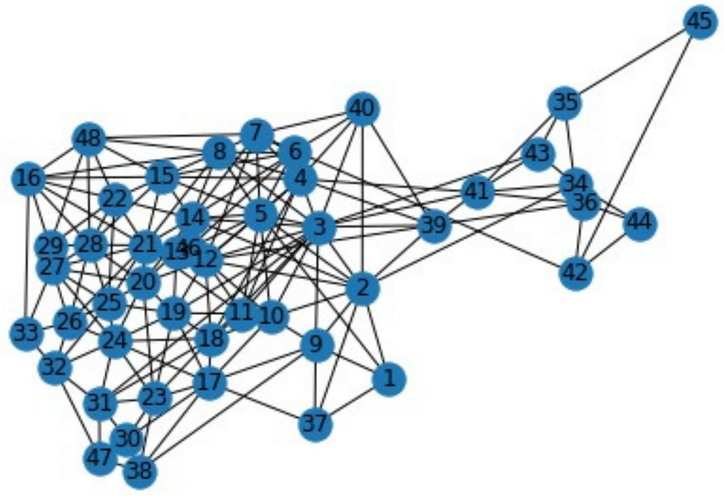
Generacion del grafo

Con la ayuda de la libreria NetworkX se genera el grafo G = (V,E)
Dónde V representa el conjuntos de vértices del grafo (Que son cada edificio de la UIS)
y E representa el conjunto de Aristas que une estos vértices.

La generación del grafo fue mendiante el comando:
G = nx.Graph() #dónde nx reprenseta la libreria NetworkX

Y se añadieron las aristas mediante el comando:
G.add_weighted_edges_from(edges) #dónde edges es una lista de tuplas en la cual cada tupla es de la forma (<verticeInicial>,<verticeFinal>,<peso>).

La lista edges contiene toda la información de las aristas, esta lista se obtiene de un archivo de excel denominado edges.xlsx que se encuentra en el repositorio de github del proyecto.



Asignacion del peso de las aristas

El peso de las aristas es igual a la distancia entre cada lugar de interés, esta distancia se obtuvo gracias a la herramienta de medir distancia de google maps

Seleccion de algoritmo

```
dist[s] ←0                                  (distance to source vertex is zero)
for all v ∈ V–{s}
      do dist[v] ←∞                         (set all other distances to infinity)
S←∅                                         (S, the set of visited vertices is initially empty)
Q←V                                         (Q, the queue initially contains all vertices)
while Q ≠∅                                  (while the queue is not empty)
do  u ← mindistance(Q,dist)                 (select the element of Q with the min. distance)
    S←S∪{u}                                 (add u to list of visited vertices)
     for all v ∈ neighbors[u]
          do if  dist[v] > dist[u] + w(u, v)          (if new shortest path found)
                then     d[v] ←d[u] + w(u, v)        (set new value of shortest path)
                   (if desired, add traceback code)
return dist
```