El problema del agente viajero

Este programa permite al usuario descubrir la ruta mas directa por atravesar varios objetos y terminar el recorrido con la menor distancia posible, esto es recorrer todos los puntos sin repetir y a la vez lo mas económico.

El siguiente fragmento de código nos sirve para representar nuestro graficar de tal manera que el ordenador lo pueda graficar y entender.

class Grafo:

def \_\_init\_\_(self):

self.V = set() #un conjunto

self.E = dict() # un mapeo de pesos de aristas

self.vecinos = dict() # un mapeo

def agrega(self, v):

self.V.add(v)

if not v in self.vecinos: # vecindad de v

self.vecinos[v] = set() # inicialmente no tiene nada

def conecta(self, v, u, peso=1):

self.agrega(v)

self.agrega(u)

self.E[(v,u)] = self.E[(u,v)] = peso # en ambos sentidos

self.vecinos[v].add(u)

self.vecinos[u].add(v)

Este es mi grafo de países:

3

Inglaterra

España

7

4

Alemania

2

Rusia

2

1

Portugal

3

Gales

4

8

5

Francia

Irlanda

6

2

2

Suecia

Suiza

Este es el programa :

El algoritmo que vamos a utilizar para determinar la ruta más corta se llama el “algoritmo de Dijkstra”. El algoritmo de Dijkstra es un algoritmo iterativo que nos proporciona la ruta más corta desde un nodo inicial particular a todos los otros nodos en el grafo. De nuevo, esto es similar a los resultados de una búsqueda en anchura.

**from** **pythoned.grafos** **import** ColaPrioridad, Grafo, Vertice

**def** dijkstra(unGrafo,inicio):

cp = ColaPrioridad()

inicio.asignarDistancia(0)

cp.construirMonticulo([(v.obtenerDistancia(),v) **for** v **in** unGrafo])

**while** **not** cp.estaVacia():

verticeActual = cp.eliminarMin()

**for** verticeSiguiente **in** verticeActual.obtenerConexiones():

nuevaDistancia = verticeActual.obtenerDistancia() \

+ verticeActual.obtenerPonderacion(verticeSiguiente)

**if** nuevaDistancia < verticeSiguiente.obtenerDistancia():

verticeSiguiente.asignarDistancia( nuevaDistancia )

verticeSiguiente.asignarPredecesor(verticeActual)

cp.decrementarClave(verticeSiguiente,nuevaDistancia)

Este seria mi grafo en Python:

g= Grafo()

g.conecta('España','Alemania',5)

g.conecta('España','inglaterra',3)

g.conecta('España','Rusia',4)

g.conecta('Inglaterra','Gales',2)

g.conecta('Alemania','Portugal',2)

g.conecta('Portugal','Rusia',1)

g.conecta('Portugal','Francia',4)

g.conecta('Portugal','Irlanda',3)

g.conecta('Portugal','Suiza',8)

g.conecta('Francia','Suecia',2)

g.conecta('Suecia','Suiza',2)

g.conecta('Gales','Irlanda',5)

g.conecta('Portugal','Inglaterra',7)

print(dijkstra(g,'España'))

['España', 'Rusia', 'Portugal', 'Suiza', 'Suecia', 'Francia', 'Irlanda', 'Gales', 'Inglaterra', 'Alemania', 'inglaterra']

print(5+3+4+2+2+1+4+3+8+2+2+5+7)

48

print(dijkstra(g,'Inglaterra'))

['Inglaterra', 'Portugal', 'Suiza', 'Suecia', 'Francia', 'Rusia', 'España', 'Alemania', 'inglaterra', 'Irlanda', 'Gales']

print(dijkstra(g,'Alemania'))

['Alemania', 'Portugal', 'Suiza', 'Suecia', 'Francia', 'Rusia', 'España', 'inglaterra', 'Irlanda', 'Gales', 'Inglaterra']

print(dijkstra(g,'Portugal'))

['Portugal', 'Suiza', 'Suecia', 'Francia', 'Rusia', 'España', 'Alemania', 'inglaterra', 'Irlanda', 'Gales', 'Inglaterra']

print(dijkstra(g,'Rusia'))

['Rusia', 'Portugal', 'Suiza', 'Suecia', 'Francia', 'Irlanda', 'Gales', 'Inglaterra', 'Alemania', 'España', 'inglaterra']

print(dijkstra(g,'Gales'))

['Gales', 'Irlanda', 'Portugal', 'Suiza', 'Suecia', 'Francia', 'Rusia', 'España', 'Alemania', 'inglaterra', 'Inglaterra']

print(dijkstra(g,'Francia'))

['Francia', 'Portugal', 'Suiza', 'Suecia', 'Rusia', 'España', 'Alemania', 'inglaterra', 'Irlanda', 'Gales', 'Inglaterra']

print(dijkstra(g,'suiza'))

'Suiza', 'Portugal', 'Rusia', 'España', 'Alemania', 'inglaterra', 'Irlanda', 'Gales', 'Inglaterra', 'Francia', 'Suecia']

print(dijkstra(g,'suecia'))

['Suecia', 'Suiza', 'Portugal', 'Rusia', 'España', 'Alemania', 'inglaterra', 'Irlanda', 'Gales', 'Inglaterra', 'Francia']

print(dijkstra(g,'Irlanda'))

['Irlanda', 'Portugal', 'Suiza', 'Suecia', 'Francia', 'Rusia', 'España', 'Alemania', 'inglaterra', 'Inglaterra', 'Gales']