Universidad del Valle de Guatemala
Ingeniería de Software
Sección 11
2 de Julio del 2024

GRUPO 2

RUTA MAS CORTA PARA DETERMINAR DISTANCIA DE LUGARES.

MARCO CONCEPTUAL

El algoritmo de Dijkstra se puede aplicar para encontrar la ruta más corta desde la ubicación del usuario a diversas instituciones, utilizando las coordenadas de longitud y latitud. A continuación se explica en detalle cómo se puede implementar este algoritmo en este contexto.

1. Representación del Grafo

Para aplicar el algoritmo de Dijkstra, necesitamos representar nuestras ubicaciones (usuario e instituciones) como un grafo. En este caso, cada nodo en el grafo representará una ubicación (el usuario o una institución), y las aristas entre los nodos estarán ponderadas por la distancia entre ellos.

Distancia entre dos puntos: La distancia euclidiana entre dos puntos

$$d = \sqrt{(lat2 - lat1)^2 + (lon2 - lon1)^2}$$

Para una mayor precisión, especialmente en grandes distancias, se puede utilizar la fórmula de Haversine:

$$a = sen^2(\frac{\Delta lat}{2}) + cos(lat1) \cdot cos(lat2) \cdot sin^2(\Delta lon/2)$$
$$c = 2\, arctan\, 2(\sqrt{a}, \sqrt{1 - a},)$$
$$distancia = R * c$$

ALGORITMO

1. **Inicialización de Parámetros:**
- Coordenadas del usuario (lat_usuario, lon_usuario).
- Lista de instituciones con coordenadas (lat, lon), nombre y dirección.

```
Coordenadas del usuario: (lat_usuario, lon_usuario)
Lista de instituciones: [
  {nombre: 'Institucion A', lat: 37.8715, lon: -122.2730, direccion:
'Direccion A'},
  {nombre: 'Institucion B', lat: 37.8044, lon: -122.2711, direccion:
'Direccion B'}, ...
]
```

2. **Función para Calcular la Distancia Entre dos Puntos (Fórmula de Haversine):**

```
Funcion calcular_distancia(lat1, lon1, lat2, lon2):
    R = 6371  # Radio de la Tierra en km
    dlat = (lat2 - lat1) en radianes
    dlon = (lon2 - lon1) en radianes
    a = seno(dlat / 2)^2 + coseno(lat1 en radianes) * coseno(lat2 en radianes) * seno(dlon / 2)^2
    c = 2 * atan2(raiz_cuadrada(a), raiz_cuadrada(1 - a))
    distancia = R * c
    retornar distancia
```

3. Construcción del Grafo:

- Crear un nodo para el usuario.
- Crear nodos para cada institución.
- Calcular las distancias entre el nodo del usuario y cada nodo de las instituciones.

```
Grafo = {}
Para cada institucion en lista de instituciones:
    distancia = calcular_distancia(lat_usuario, lon_usuario, institucion.lat,
institucion.lon)
    Grafo[nombre_usuario][institucion.nombre] = distancia
```

Aplicación del Algoritmo de Dijkstra

1. Inicialización del Algoritmo:

```
Inicializar conjunto de nodos no visitados con todos los nodos
Inicializar distancias a infinito, excepto la distancia del nodo del usuario
que es 0
Previo = diccionario vacío para rastrear la ruta
```

2. Iteración del Algoritmo:

```
Mientras haya nodos no visitados:
    Nodo_actual = nodo no visitado con la distancia mínima
    Para cada vecino del nodo_actual:
        Si vecino está en nodos no visitados:
            Nueva_distancia = distancia[nodo_actual] + peso de la arista entre actual y vecino
            Si nueva_distancia < distancia[vecino]:
                distancia[vecino] = nueva_distancia
                previo[vecino] = nodo_actual
    Remover nodo_actual de nodos no visitados
```

3. Determinación de la Ruta Más Corta:

```
Ruta = lista vacía
Nodo_actual = nodo_objetivo (nombre de la institución deseada)

Mientras nodo_actual != nodo_inicio (nombre del usuario):
```

```
    Insertar nodo_actual al inicio de Ruta
    Nodo_actual = previo[nodo_actual]

Insertar nodo_inicio al inicio de Ruta
```