Segundo Problema DAA

Karlos Alejandro Alfonso Rodríguez
Karel Camilo Manresa Leon

May 7, 2023

Problema

Lázaro presidente del PCC.
Han pasado 20 años desde que Lázaro se graduó de Ciencias de la Computación (haciendo una muy buena tesis) y las vueltas de la vida lo llevaron a convertirse en el presidente del Partido Comunista de Cuba. Una de sus muchas responsabilidades consiste en visitar zonas remotas. En esta ocasión debe visitar una ciudad campestre de Pinar del Río.
También han pasado 20 años desde que Marié consiguió su título en MATCOM. Tras años de viaje por las grandes metrópolis del mundo, en algún punto decidió que prefería vivir una vida tranquila, aislada de la urbanización, en una tranquila ciudad de Pinar del Río. Las vueltas de la vida quisieron que precisamente Marié fuera la única universitaria habitando la ciudad que Lázaro se dispone a visitar. Los habitantes de la zona entraron en pánico ante la visita de una figura tan importante y decidieron reparar las calles de la ciudad por las que transitaría Lázaro. El problema está en que nadie sabía qué ruta tomaría el presidente y decidieron pedirle ayuda a Marié.
La ciudad tiene n puntos importantes, unidos entre sí por calles cuyos tamaños se conocen. Se sabe que Lázaro comenzará en alguno de esos puntos s y terminará el viaje en otro t. Los ciudadanos quieren saber, para cada par s, t, cuántas calles participan en algún camino de distancia mínima entre s y t.

Problema en términos matemáticos

Dado un grafo no dirigido y ponderado, donde no hay aristas de costo negativo. Se desea saber para cada par de vértices s y t, cuantas aristas participan en algún camino de costo mínimo entre s y t.

Modelación

Supongamos que para cada par de vétices s y d tenemos el conjunto $U_{s,d}$, que contiene los vértices que participan en algún camino de costo mínimo entre s y d. Entonces, si para cada vértice $v \in U_{s,d}$, tenemos la cantidad de aristas que llegan en un camino de costo mínimo entre s y v (partiendo de s), tendríamos la cantidad de aristas que llegan a cada vértice de un camino de costo mínimo entre s y d, llamémosle $W_{s,v}$ a este conjunto. Por tanto, para s y d, la cantidad de aristas que participan en algún camino de costo mínimo es $\sum_{v \in U_{s,d}} W_{s,v}$.
Luego para resolver el problema basta con calcular los conjuntos $U_{s,d}$ y $W_{s,v}$ para cada par de vértices s y d, y efectuar $\sum_{v \in U_{s,d}} W_{s,v}$.

Como obtener $W_{s,v}$

Sea F la matriz de distancias obtenida luego de aplicar el algoritmo de Floyd-Warshall. Sea e una arista del grafo. Sea u un vértice del grafo, v y w los vértices que forman la arista e. Podemos saber si e participa en un camino

de costo mínimo de u a v si $F[u,w] + peso(e) = F[u,v]$ (para el caso de w si $F[u,v] + peso(e) = F[u,w]$). Demostremos esto.

Lema

Sea F la matriz resultante de aplicar Floyd-Warshall en un grafo no dirigido y ponderado. Sea u, v y w vértices del grafo, y e una arista que une a v y w. Si $F[u,v] + peso(e) = F[u,w]$ entonces e participa en algún camino de costo mínimo entre u y w.

Proof. Si $F[u,v] + peso(e) = F[u,w]$ entonces el algoritmo de Floyd-Warshall encontró un camino de costo mínimo entre u y v con costo $F[u,v]$ y un camino de costo mínimo entre u y w con costo $F[u,w]$. Por tanto, como $e =< v,w >$, si e no participara en algún camino de costo mínimo entre u y w, entonces el algoritmo de Floyd-Warshall habría encontrado un camino con costo menor que $F[u,w]$, lo cual es una contradicción. Luego e participa en algún camino de costo mínimo entre u y w. □

Entonces para obtener $W_{s,v}$ para cada s y v basta con iterar por las aristas y vértices del grafo comprobando lo anterior.

Como obtener $U_{s,d}$

Sea F la matriz de distancias obtenida luego de aplicar el algoritmo de Floyd-Warshall. Sea u, v y w vértices del grafo. Si $F[u,w]+F[w,v] = F[u,v]$ entonces w participa en algún camino de costo mínimo entre u y v. Demostremos esto.

Lema

Sea F la matriz resultante de aplicar Floyd-Warshall en un grafo no dirigido y ponderado. Sea u, v y w vértices del grafo. Si $F[u,w] + F[w,v] = F[u,v]$ entonces w participa en algún camino de costo mínimo entre u y v.

Proof. $F[u,w]$ es el costo de un camino de costo mínimo de u a w, $F[w,v]$ es el costo de un camino de costo mínimo de w a v, y si la suma del costo de ambos caminos es igual al costo de un camino de costo mínimo entre u y v, entonces existe un camino $u...w...v$ que tiene costo mínimo, por tanto w participa en algún camino de costo mínimo de u a v. □

Solución

Teniendo en cuenta lo anterior, para cada par de vértices s y d, se puede obtener $U_{s,d}$ y $W_{s,v}$ ($v \in V$), y como se planteaba al inicio de la modelación del problema, calcular la solución para $[s,d]$ sería efectuar $\sum_{v \in U_{s,d}} W_{s,v}$.

Pseudocódigo

```
1       def solve(graph):
2           F = Floyd-Warshall(graph)
3           W = {u -> {v -> int}}
4           solution = {u -> {v -> int}}
5
6           for e in edges:
7               for u in nodes:
8                   if F[u,e.v1] + e.weight == F[u,e.v2]:
9                       W[u,e.v2] += 1
10                  if F[u,e.v2] + e.weight == F[u,e.v1]:
11                      W[u,e.v1] += 1
12
13          for u in nodes:
14              for v in nodes:
15                  for w in nodes:
16                      if F[u,w] + F[w,v] == F[u,v]:
17                          solution[u,v] += W[u,w]
18
19          return solution
```

Complejidad temporal

Para calcular la complejidad temporal del algoritmo se analizarán las complejidades de los bloques de código ***2-4***, ***6-11*** y ***13-17***.
El bloque de código ***2-4*** tiene complejidad $O(|V|^3)$ ya que el costo de ejecutar Floyd-Warshall es $O(|V|^3)$ y el costo de inicializar W y *solution* es $O(|V|^2)$ y $O(|V|^3) + O(|V|^2) = O(|V|^3)$.
En el bloque de código ***6-11*** se itera por todas las aristas y vértices del grafo, por tanto tiene complejidad $O(EV)$ (el interior del ciclo es $O(1)$). En caso de ser denso el grafo, se cumple que $|E| \approx |V|^2$ por lo que el costo sería $O(|V|^3)$.
Luego en el bloque ***13-17*** se realiza un triple for por los vértices del grafo, por tanto tiene costo $O(|V|^3)$(el interior del ciclo es $O(1)$).
Por tanto la complejidad temporal del algoritmo es:

$$O(|V|^3) + O(|V|^2) + O(|V|^3) = O(|V|^3). \tag{1}$$