Recorrido Mínimo Uno a Todos - Soluciones

Compilado: 23 de octubre de 2024

1. Policías 👮

Problema

La nueva reglamentación de una ciudad establece que toda esquina debe estar a lo sumo a 5 cuadras de una estación de policía. Dada la lista de esquinas $\{v_1, \ldots, v_n\}$ de la ciudad, la lista $\{p_1, \ldots, p_k\}$ de esquinas donde hay policías[a], y la lista E de calles, debemos indicar si la normativa se cumple, y en caso contrario cuáles esquinas son las que quedan "desprotegidas".

[a] O sea, asumimos que la policía siempre se ubica en una esquina.

1.1. Resolución

Primero, modelemos el problema con un grafo G, donde $V(G) = \{v_1, \ldots, v_n\}$ (las esquinas) y $E(G) = E$ (las calles). Traduzcamos el problema a este grafo: necesitamos encontrar los vértices que estén a distancia mayor a 5 de algún vértice del conjunto $P = \{p_1, \ldots, p_k\}$.

Podríamos probar primero haciendo un BFS desde cada uno de los vértices v de $V(G)$, encontrando para cada uno la distancia mínima a un vértice de P. Si la mínima distancia de v a un vértice de P es mayor a 5, sabemos que esa esquina está desprotegida. Esto tendría complejidad $O((n+m)*n)$, siendo $n = |V(G)|$ y $m = |E(G)|$.

Pero podemos hacer mejor que esto. En vez de tomar las distancias desde cada uno de los vértices, tomamos las distancias desde las estaciones de policía. Y una cosa a notar es la siguiente: no nos importa a qué estación de policía están más próximas las esquinas, sino solamente si tienen una a menos de 6 cuadras. Si pudiésemos de alguna manera correr BFS desde cada estación de policía al mismo tiempo, estaríamos marcando todos los vértices a distancia 1 de cualquier estación, después a distancia 2, etc.

Bueno, esto sí se puede hacer: agregamos un vértice "fantasma" 👻 que sea adyacente a todos los vértices con estación de policía, y empezamos el BFS desde ahí. Así, vamos a marcar todos los vértices con la distancia a 👻. Como todo camino desde 👻 tiene como segunda posición una estación de policía, cada camino de distancia d entre 👻 y otro vértice v contiene un camino de distancia $d-1$ entre una estación de policía y v. Por lo tanto, v está a distancia como máximo $d-1$ de alguna estación de policía. Por otro lado, un vértice v está a distancia $d-1$ de una estación de policía sólo si está a distancia d de 👻 (¿Por qué?). Luego, restarle uno a las distancias de cada vértice a 👻 resulta en las distancias a alguna estación de policía.

Recapitulando entonces:

1. Definimos el grafo $G = (\{v_1, \ldots, v_n\}, E)$ como lista de adyacencias. Complejidad: $O(n+m)$.
2. Definimos el grafo G' como el resultado de agregar un vértice 👻 a G adyacente a todos los vértices $\{p_1, \ldots, p_k\}$. Complejidad: $O(k)$.
3. Corremos BFS en G' desde 👻, guardando las distancias a cada vértice. Complejidad: $O(n+m)$.
4. Restamos 1 a todas las distancias. Complejidad: $O(n)$.

5. Si no hay vértices con distancia mayor a 5, decimos que la normativa se cumple. De lo contrario, enumeramos los vértices con distancia mayor a 5. Complejidad: $O(n)$.

Complejidad total: $O(n + m + k) = O(n + m)$.

2. Martín y los Mares 🌊

Problema

Martín es un conocido comerciante marítimo. Planea cruzar el Mar Tortuğoso empezando desde la isla *Anastasia*, parando en algunas islas para comerciar y descansar, y llegando a la isla *Betuna*. Sabe que la mayor amenaza a su barco no son los piratas: son los percebes que se enganchan y pudren la madera. También sabe qué camino entre las piedras debe usar para pasar de una isla a otra, y tiene tan estudiadas las rutas que sabe exactamente cuántos percebes van a engancharse a su barco en cada ruta.

En algunas de esas rutas, sin embargo, ocurre lo contrario: tortugas marinas vienen y se comen un porcentaje de los percebes que tenga el barco. Lamentablemente, Martín sabe que es muy peligroso pasar por dos de estas rutas, ya que las tortugas a veces muerden de más. Por lo tanto, quiere pasar por máximo una sola de las rutas con tortugas.

Martín cursó TDA pero hace muuucho tiempo, y entonces les pregunta a ustedes cuál es la mínima cantidad de percebes con la que puede alcanzar el otro lado del Mar Tortuğoso.

2.1. Resolución

Por simplicidad, llamemos A y B a *Anastasia* y *Betuna*, respectivamente. Podríamos empezar modelando un digrafo D con vértices A, B, y las islas intermedias, y con aristas correspondiendo a las rutas, con costo $c(u, v)$ para los que suman percebes, y con porcentaje $t(u, v)$ (representado como fracción) para los que multiplican por un porcentaje. Uno entonces pensaría quizá encontrar el camino mínimo desde A hasta B con algún algoritmo que conozcamos. Esto tiene dos problemas:

1. Los costos no son todos sumados al costo de un camino: algunos son un producto por una fracción.
2. El camino puede pasar por máximo una arista de tortugas, y no estamos restringiendo eso de ninguna manera acá.

Tenemos que pensar un poco más.

Antes de poder usar alguno de los algoritmos que conocemos tenemos que pensar qué hacemos con las aristas de tortugas, que me dividen. En principio vamos a sacarlas y despues vemos cómo las agregamos nuevamente. Trabajamos sobre el digrafo D' que es igual al D sacando las aristas con tortugas.

Usando Dijkstra vamos a aplicar una técnica que se usa bastante para este tipo de problemas: en lugar de correr una sola vez el algoritmo de camino mínimo vamos a correrlo dos veces. Pero, ¿desde qué nodos? Estamos intentando llegar de un vértice A a otro B así que seguramente tengamos que correrlo desde A. ¿Podemos correr Dijkstra desde B? Quizás así nomás no pero si damos vuelta la orientación de las aristas del digrafo (trasponemos el digrafo) puede servir. Veámoslo de a pasitos:

Primero corremos Dijkstra desde A, por lo que obtenemos los pesos de los caminos desde A a todos los nodos del digrafo D'. Luego trasponemos el digrafo y hacemos un Dijkstra pero desde B, lo que nos da los pesos de todos a B en el digrafo D'. Pero, ¿para qué nos sirve esto? En este momento es cuando vuelven a aparecer las aristas que sacamos, ya que para cada arista (u, v) que sacamos podemos calcular cuál sería el costo mínimo de un recorrido que tiene a esa arista. La cuenta que hacemos para cada arista va a ser: $w(A, u) * (1 - t(u, v)) + w(v, B)$ (siendo $w(\cdot, \cdot)$ el peso del camino mínimo y $t(\cdot, \cdot)$ el porcentaje removido en una arista tortuga). Recordemos que el costo de las aristas con tortugas es el porcentaje de percebes que comen, por eso no es solo sumar el costo de la arista.

Repasemos los pasos del algoritmo:

1. Corremos Dijkstra desde A. Complejidad: $O(min\{m \cdot log(n),\, n^2\})$.
2. Trasponemos el digrafo. Complejidad: $O(m + n)$.
3. Corremos Dijkstra desde B. Complejidad: $O(min\{m \cdot log(n),\, n^2\})$.
4. Para cada "arista tortuga" calculamos el peso del recorrido que la usa con $w(A, u) * (1 - t(u, v)) + w(v, B)$ y devolvemos el camino con menor peso. Complejidad: $O(m)$.

Entonces la complejidad del algoritmo es $O(min\{m \cdot log(n),\, n^2\} + n)$.

2.2. Demostración de correctitud

Para la demostración queremos ver que el algortimo devuelve la solución óptima, donde la solución óptima y lo que devuelve el algoritmo son:

Solución:

$$min(\{\text{Pesos de todos los caminos que no tienen aristas tortuga}\},\\ \{\text{Pesos de los caminos con exactamente una arista tortuga}\}).$$

Para facilitar la notación será $\min(P_{\text{sin🐢}}, P_{\text{con🐢}})$.

Algoritmo:

$$min(\{\text{Peso mínimo de camino sin aristas tortuga}\},\\ \{w(A, u) * (1 - t(u, v)) + w(v, B) \mid (u, v) \text{ es una arista tortuga}\}).$$

Para facilitar la notación será $\min(P'_{\text{sin🐢}}, P'_{\text{con🐢}})$. $P'_{\text{sin🐢}}$ debo considerarlo para el caso que convenga pasar por 0 aristas tortuga.

Ver que $\min(P_{\text{sin🐢}}) = \min(P'_{\text{sin🐢}})$ es trivial ya que $\min(P_{\text{sin🐢}})$ es exactamente el peso mínimo de camino sin aristas tortuga, pero ¿cómo veo que $\min(P_{\text{con🐢}}) = \min(P'_{\text{con🐢}})$?

Lo vamos a probar viendo que $\min(P'_{\text{con🐢}}) \geq \min(P_{\text{con🐢}})$ y que $\min(P'_{\text{con🐢}}) \leq \min(P_{\text{con🐢}})$:

- $\min(P'_{\text{con🐢}}) \geq \min(P_{\text{con🐢}})$: como en $P'_{\text{con🐢}}$ tenemos los pesos de los caminos con una arista tortuga tal que los costos de A a u y de v a B son mínimas, sabemos que estará incluido en los pesos de todos los caminos que tienen exactamente 1 arista tortuga (aunque no tengamos los mínimos) entonces $P'_{\text{con🐢}} \subseteq P_{\text{con🐢}}$.

Además, sabemos que $P'_{\text{con🐢}} \subseteq P_{\text{con🐢}} \implies \min(P'_{\text{con🐢}}) \geq \min(P_{\text{con🐢}})$ pues como todo elemento en $P'_{\text{con🐢}}$ está en $P_{\text{con🐢}}$, el mínimo de $P_{\text{con🐢}}$ será al menos tan chico como el mínimo de $P'_{\text{con🐢}}$ que tiene menos elementos, con lo que probamos que $\min(P'_{\text{con🐢}}) \geq \min(P_{\text{con🐢}})$.

- $\min(P'_{\text{con🐢}}) \leq \min(P_{\text{con🐢}})$: sabemos que $w(A,u)$ es mínimo y $w(v,B)$ también, por lo que $w(A,u)$ es menor o igual al peso de cualquier camino de A a u que no use aristas tortuga, y $w(v,B)$ es menor o igual al peso de cualquier camino de v a B que no use aristas tortuga. Como $t(u,v) \geq 0$, tenemos que $\min(w(A,u) * (1 - t(u,v)) + w(v,B)) \leq \min(\{\text{Pesos de los caminos con exactamente una arista tortuga}\})$ (ya que los costos de las aristas tortugas son los mismos en $P_{\text{con🐢}}$ y $P'_{\text{con🐢}}$), que es equivalente a $\min(P'_{\text{con🐢}}) \leq \min(P_{\text{con🐢}})$.

$\implies$ Solución = algoritmo.