Recorrido Mínimo Uno a Todos

Técnicas de Diseño de Algoritmos

FCEyN UBA

Mayo 2024

¿Dónde estamos?

Hasta ahora vimos:

- Qué son los grafos y cómo representarlos

¿Dónde estamos?

Hasta ahora vimos:

- Qué son los grafos y cómo representarlos
- Algoritmos para **recorrer** y obtener **árbol generador** de un grafo (BFS y DFS)

¿Dónde estamos?

Hasta ahora vimos:

- Qué son los grafos y cómo representarlos
- Algoritmos para **recorrer** y obtener **árbol generador** de un grafo (BFS y DFS)
- Algoritmos para obtener el **árbol generador mínimo** de un grafo (Kruskal y Prim)

¿Dónde estamos?

Hasta ahora vimos:

- Qué son los grafos y cómo representarlos
- Algoritmos para **recorrer** y obtener **árbol generador** de un grafo (BFS y DFS)
- Algoritmos para obtener el **árbol generador mínimo** de un grafo (Kruskal y Prim)

Hoy vamos a ver:

- Algoritmos para obtener el **camino mínimo de uno a todos los nodos** (Dijkstra y Bellman-Ford)

Plan de hoy

Recorrido mínimo

Problema

Sea $G = (V, E)$ un (di)grafo con una función de costo para las aristas $c\colon E \to \mathbb{R}$, y v un vértice de G. Para todo $w \in V$, encontrar la mínima suma del costo de las aristas de un recorrido desde v a w.

BFS

Problema

Sea $G = (V, E)$ un (di)grafo con una función de costo para las aristas $c \colon E \to \mathbb{R}$, y v un vértice de G. Para todo $w \in V$, encontrar la mínima suma del costo de las aristas de un recorrido desde v a w.

- ¿Se puede resolver con BFS?

BFS

Problema

Sea $G = (V, E)$ un (di)grafo con una función de costo para las aristas $c \colon E \to \mathbb{R}$, y v un vértice de G. Para todo $w \in V$, encontrar la mínima suma del costo de las aristas de un recorrido desde v a w.

- ¿Se puede resolver con BFS? **Así nomás, no.**
- ¿Qué tengo que cambiar?

BFS

Problema

Sea $G = (V, E)$ un (di)grafo con una función de costo para las aristas $c\colon E \to \mathbb{R}$, y v un vértice de G. Para todo $w \in V$, encontrar la mínima suma del costo de las aristas de un recorrido desde v a w.

- ¿Se puede resolver con BFS? **Así nomás, no.**
- ¿Qué tengo que cambiar? **El costo tiene que ser el mismo para todas las aristas.**[1]
- ¿Complejidad?

[1] Podemos relajar esto un poco, ver acá.

BFS

Problema

Sea $G = (V, E)$ un (di)grafo con una función de costo para las aristas $c\colon E \to \mathbb{R}$, y v un vértice de G. Para todo $w \in V$, encontrar la mínima suma del costo de las aristas de un recorrido desde v a w.

- ¿Se puede resolver con BFS? **Así nomás, no.**
- ¿Qué tengo que cambiar? **El costo tiene que ser el mismo para todas las aristas.**[1]
- ¿Complejidad? $\boldsymbol{O(|V| + |E|)}$ **sobre lista de adyacencias.**

[1] Podemos relajar esto un poco, ver acá.

Dijkstra

Problema

Sea $G = (V, E)$ un (di)grafo con una función de costo para las aristas $c \colon E \to \mathbb{R}$, y v un vértice de G. Para todo $w \in V$, encontrar la mínima suma del costo de las aristas de un recorrido desde v a w.

- ¿Se puede resolver con Dijkstra?

Dijkstra

Problema

Sea $G = (V, E)$ un (di)grafo con una función de costo para las aristas $c\colon E \to \mathbb{R}$, y v un vértice de G. Para todo $w \in V$, encontrar la mínima suma del costo de las aristas de un recorrido desde v a w.

- ¿Se puede resolver con Dijkstra? **Así nomás, no.**
- ¿Qué tengo que cambiar?

Dijkstra

Problema

Sea $G = (V, E)$ un (di)grafo con una función de costo para las aristas $c\colon E \to \mathbb{R}$, y v un vértice de G. Para todo $w \in V$, encontrar la mínima suma del costo de las aristas de un recorrido desde v a w.

- ¿Se puede resolver con Dijkstra? **Así nomás, no.**
- ¿Qué tengo que cambiar? **El costo tiene que ser $\geq$ 0.**
- ¿Complejidad?

Dijkstra

Problema

Sea $G = (V, E)$ un (di)grafo con una función de costo para las aristas $c\colon E \to \mathbb{R}$, y v un vértice de G. Para todo $w \in V$, encontrar la mínima suma del costo de las aristas de un recorrido desde v a w.

- ¿Se puede resolver con Dijkstra? **Así nomás, no.**
- ¿Qué tengo que cambiar? **El costo tiene que ser $\geq$ 0.**
- ¿Complejidad? $\boldsymbol{O(\min\{|E| \log |V|, |V|^2\})}$ **sobre lista de adyacencias.**

Bellman-Ford

Problema

Sea $G = (V, E)$ un (di)grafo con una función de costo para las aristas $c\colon E \to \mathbb{R}$, y v un vértice de G. Para todo $w \in V$, encontrar la mínima suma del costo de las aristas de un recorrido desde v a w.

- ¿Se puede resolver con Bellman-Ford?

Bellman-Ford

Problema

Sea $G = (V, E)$ un (di)grafo con una función de costo para las aristas $c\colon E \to \mathbb{R}$, y v un vértice de G. Para todo $w \in V$, encontrar la mínima suma del costo de las aristas de un recorrido desde v a w.

- ¿Se puede resolver con Bellman-Ford? **Sí!**
- ¿Qué pasa si hay un ciclo con suma de costos negativa?

Bellman-Ford

Problema

Sea $G = (V, E)$ un (di)grafo con una función de costo para las aristas $c\colon E \to \mathbb{R}$, y v un vértice de G. Para todo $w \in V$, encontrar la mínima suma del costo de las aristas de un recorrido desde v a w.

- ¿Se puede resolver con Bellman-Ford? **Sí!**
- ¿Qué pasa si hay un ciclo con suma de costos negativa? **Te lo dice, es crack.**
- ¿Complejidad?

Bellman-Ford

Problema

Sea $G = (V, E)$ un (di)grafo con una función de costo para las aristas $c\colon E \to \mathbb{R}$, y v un vértice de G. Para todo $w \in V$, encontrar la mínima suma del costo de las aristas de un recorrido desde v a w.

- ¿Se puede resolver con Bellman-Ford? **Sí!**
- ¿Qué pasa si hay un ciclo con suma de costos negativa? **Te lo dice, es crack.**
- ¿Complejidad? $\boldsymbol{O(|V||E|)}$ **sobre lista de adyacencias.**

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[2], y la lista E de calles, debemos indicar si la normativa se cumple, y en caso contrario cuáles esquinas son las que quedan "desprotegidas".

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

Policías - Resolución

- Primero el modelado: en este caso es bastante directo, las esquinas son los nodos, y las calles los ejes.
- Entonces lo que nos pide el enunciado es que todo nodo esté a una distancia menor o igual a 5 de un policía.
- ¿Conocemos algún algoritmo que nos permita resolver este problema?

Policías - Resolución

- Podemos correr un BFS desde cada vértice v y ver la distancia de cada uno a la estación de policía más cercana.
- La complejidad de esto es cuadrática. ¿Se puede mejorar?
- Si pudiésemos correr un BFS desde todas las estaciones de policia a la vez y cortarlo cuando la distancia es mayor a 5, guardándonos los nodos a los que puedo llegar nos estaríamos guardando todos los vértices que cumplen.
- Pero hay una manera mucho más simple.

Policías - Resolución

- Cambiemos un poquito nuestro modelo: agregamos un **nodo fantasma z** al grafo y lo conectamos con todos los policías.
- En este nuevo grafo el problema va a ser un poco distinto. Corriendo BFS desde z y viendo qué nodos están a distancia mayor a 6 nos alcanza para resolverlo (y con complejidad $O(m+n)$!!).
- Ahora solo nos queda demostrar que lo que hicimos vale y da la misma respuesta que en el grafo original. Es decir, hay que probar el siguiente lema:

Un nodo v está a distancia menor o igual a 5 de un policía si y solamente si v está a distancia menor o igual a 6 de z

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. (sigue . . .)

Martín y los Mares

Problema (cont.)

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.

Martín y los Mares - Resolución

- ¿Cómo podemos modelar con grafos este problema?

Martín y los Mares - Resolución

- ¿Cómo podemos modelar con grafos este problema?
- Modelamos como un digrafo D con A, B y las islas intermedias como vértices y con aristas correspondiendo a las rutas, con costo $p(u, v)$ para los que suman percebes y con porcentaje $t(u, v)$, representado como fracción, para los que multiplican por un porcentaje (las que tienen tortugas).

Martín y los Mares - Resolución

- ¿Nos alcanza con encontrar el camino mınimo de A a B con un algoritmo de los que conocemos?

Martín y los Mares - Resolución

- ¿Nos alcanza con encontrar el camino mınimo de A a B con un algoritmo de los que conocemos?
- No, vamos a tener dos problemas:
  - Los costos no son todos sumados al costo de un camino: los que contienen tortugas son un producto por una fracción.
  - El camino puede pasar por máximo una arista de tortugas, y no estamos restringiendo eso de ninguna manera acá.

Martín y los Mares - Resolución

- ¿Como resolvemos el problema de tener aristas que dividen?

Martín y los Mares - Resolución

- ¿Como resolvemos el problema de tener aristas que dividen?
- Por ahora podemos sacarlas y después vemos cómo las agregamos.
- Llamamos D' al digrafo sin aristas tortuga.

Martín y los Mares - Resolución

- Ahora en D' sí podemos correr algoritmos de camino mínimo. ¿Cuál conviene usar? ¿Alcanza con correr desde un solo nodo?

Martín y los Mares - Resolución

- Ahora en D' sí podemos correr algoritmos de camino mínimo. ¿Cuál conviene usar? ¿Alcanza con correr desde un solo nodo?
- Las aristas no tienen costos negativos así que podemos usar Dijkstra. Estamos intentando llegar desde A hasta B así que posiblemente tengamos que llamar a Dijkstra desde A. ¿Servirá correrlo también desde B?

Martín y los Mares - Resolución

- Ahora en D' sí podemos correr algoritmos de camino mínimo. ¿Cuál conviene usar? ¿Alcanza con correr desde un solo nodo?
- Las aristas no tienen costos negativos así que podemos usar Dijkstra. Estamos intentando llegar desde A hasta B así que posiblemente tengamos que llamar a Dijkstra desde A. ¿Servirá correrlo también desde B?
- Así nomás no pero llamemos H al digrafo D' con las orientaciones dadas vuelta (traspuesto). Si en H corremos Dijkstra desde B podemos obtener los recorridos mínimos de todos a B en D'.

Martín y los Mares - Resolución

- Ahora volvamos a considerar las *aristas tortuga* que sacamos de D. Podemos calcular para cada arista tortuga (u, v) cuál sería el costo mínimo de un camino que la contiene.
- Para cada arista hacemos $w(A, u) * t(u, v) + w(v, B)$ (siendo $w(\cdot, \cdot)$ el peso del camino y $t(\cdot, \cdot)$ la fracción que queda después de pasar por la arista).
- $w(A, u)$ y $w(v, B)$ lo tenemos precalculado con Dijkstra y multiplicamos $w(A, u)$ con $t(u, v)$ porque el costo de las aristas tortuga es un porcentaje de los percebes que tenía el barco.

Martín y los Mares - Resolución

Recapitulando, los pasos del algoritmo y su complejidad son:

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 $w(A, u) * t(u, v) + w(v, B)$ y devolvemos el camino con menor peso. Complejidad: $O(m)$.

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

Martín y los Mares - Bonus

Tareíta

¿Se puede resolver el mismo problema con Bellman-Ford?
Demostrar que sí o mostrar un contraejemplo.

Manuel y los Monstruos

Problema

Manuel estaba aburrido en la clase de TDA, así que se puso a programar un juego. Quiere hacer que el jugador pase por distintos mundos pasando por portales unidireccionales, matando los monstruos de los mundos por los que pasa. Cada portal i por el que logra pasar le da $p_i \in \mathbb{R}$ puntos. El objetivo del juego es llegar desde el mundo m_1 hasta el mundo m_n con la mayor cantidad de puntos posible.

A Manu le preocupa que el juego esté balanceado, así que quiere saber cuál es la máxima cantidad de puntos que puede generar un jugador dada la cantidad de mundos, los portales que hay, y cuántos puntos da cada portal. Además, quiere saber cuál es la mínima cantidad de portales podría estar tomando el jugador que hiciese la máxima cantidad de puntos posible.

Manuel y los Monstruos - Resolución Puntaje Máximo

Empecemos con el modelado:

- Tenemos mundos que se conectan entre otros a través de portales, donde cada portal tiene un puntaje asignado ¿Cómo podemos modelar el problema?

Manuel y los Monstruos - Resolución Puntaje Máximo

Empecemos con el modelado:

- Tenemos mundos que se conectan entre otros a través de portales, donde cada portal tiene un puntaje asignado ¿Cómo podemos modelar el problema?
- Armamos un digrafo G donde los nodos son los mundos, los ejes dirigidos los portales que llevan de un mundo a otro, y los pesos de las aristas son el puntaje del portal.

Manuel y los Monstruos - Resolución Puntaje Máximo

Algunas consideraciones:

- ¿Qué sería una partida desde el punto de vista del grafo?

Manuel y los Monstruos - Resolución Puntaje Máximo

Algunas consideraciones:

- ¿Qué sería una partida desde el punto de vista del grafo? Un recorrido R que empieza desde el nodo m_1 y termina en el m_n.
- ¿Hay alguna cota para el máximo puntaje?

Manuel y los Monstruos - Resolución Puntaje Máximo

Algunas consideraciones:

- ¿Qué sería una partida desde el punto de vista del grafo? Un recorrido R que empieza desde el nodo m_1 y termina en el m_n.
- ¿Hay alguna cota para el máximo puntaje? Depende de si existe un ciclo C de longitud positiva alcanzable por m_1 y que luego alcanza m_n. Esto tenemos que demostrarlo.
- Para el resto del problema asumamos que no hay ejes salientes de m_n (¿Por qué es esto válido?).

Manuel y los Monstruos - Resolución Puntaje Máximo

- ¿Cómo podemos hacer ahora para resolver un problema de recorrido máximo?

Manuel y los Monstruos - Resolución Puntaje Máximo

- ¿Cómo podemos hacer ahora para resolver un problema de recorrido máximo?
- ¡Si multiplicamos todos los pesos por -1 lo convertimos en un problema de recorrido mínimo!
- Además los "ciclos positivos" pasan a ser ciclos negativos.
- ¿Esto ya se parece más a un problema que podemos resolver, no?

Manuel y los Monstruos - Resolución Puntaje Máximo

- No del todo, porque a nosotros solo nos importan los ciclos negativos que están en medio de un recorrido de m_1 a m_n, y no sabemos cómo detectar solamente estos casos.
- ¿Cómo lo podemos solucionar?

Manuel y los Monstruos - Resolución Puntaje Máximo

- No del todo, porque a nosotros solo nos importan los ciclos negativos que están en medio de un recorrido de m_1 a m_n, y no sabemos cómo detectar solamente estos casos.
- ¿Cómo lo podemos solucionar?
- Necesitamos olvidarnos la parte de que nos ciclos "alcancen a m_n", para ello lo que podemos hacer es borrar todos los nodos que no alcazan a m_n y ya. (¿Por qué?)
- Para ello alcanza con correr DFS desde m_n en G^t para ver que nodos no son alcanzables y eliminarlos.

Manuel y los Monstruos - Resolución Puntaje Máximo

Ahora sí, definamos cómo quedaría el algoritmo:

1. Armamos el modelo.
2. Multiplicamos los pesos de las aristas por -1.
3. Eliminamos todos los nodos que no llegan a m_n mediante DFS.
4. Ejecutamos Bellman-Ford desde m_1, y devolvemos infinito si encuentra un ciclo, y sino el opuesto del valor del recorrido mínimo entre m_1 y m_n.

Manuel y los Monstruos - Resolución Recorrido Más Corto

- Para empezar, notemos que si el puntaje máximo no fuera acotado, no tendría sentido buscar el recorrido más corto que lo alcance.
- ¿Cómo podemos hacer para decidir cuán largo es el recorrido mínimo más corto?

Manuel y los Monstruos - Resolución Recorrido Más Corto

- Para empezar, notemos que si el puntaje máximo no fuera acotado, no tendría sentido buscar el recorrido más corto que lo alcance.
- ¿Cómo podemos hacer para decidir cuán largo es el recorrido mínimo más corto?
- Si todas las aristas del grafo pertenecieran a un recorrido mínimo sería un problema más fácil, ¿no?

Manuel y los Monstruos - Resolución Recorrido Más Corto

- Para empezar, notemos que si el puntaje máximo no fuera acotado, no tendría sentido buscar el recorrido más corto que lo alcance.
- ¿Cómo podemos hacer para decidir cuán largo es el recorrido mínimo más corto?
- Si todas las aristas del grafo pertenecieran a un recorrido mínimo sería un problema más fácil, ¿no?
- Esto no lo podemos garantizar, pero tal vez podamos construirnos un nuevo grafo a partir del original que solo contenga las aristas que pertenezcan a algún recorrido mínimo. ¿Cómo?

Manuel y los Monstruos - Resolución Recorrido Más Corto

Construyámonos el "DAG de recorridos mínimos" de m_1 a m_n en G. Esto es un digrafo acíclico que contiene todas las aristas pertenecientes a un recorrido mínimo entre dichos nodos del grafo

- Corremos Bellman-Ford desde m_1 y guardamos el resultado en un vector "DistanciaAm_1"
- Corremos Bellman-Ford desde m_n en G^t y guardamos el resultado en un vector "DistanciaAm_n"
- Para cada arista $(u, v) \in G$, si cumple:

$$\text{DistanciaA}m_1[u] + \text{peso}(u, v) + \text{DistanciaA}m_n[v] = \text{DistanciaA}m_1[m_n],$$

la agrego al DAG de recorridos mínimos.

Manuel y los Monstruos - Resolución Recorrido Más Corto

Finalmente el algoritmo quedaría:

- Construimos el DAG de recorridos mínimos de m_1 a m_n en G
- Corremos BFS en G desde m_1, y la altura de m_n va a ser el largo del recorrido mínimo más corto

Solo quedaría demostrar que este algoritmo es correcto.

Resumen

Hoy vimos varias herramientas que nos pueden ser útiles a la hora de trabajar con problemas de camino mínimo. Estas fueron:

- Agregar un nodo fantasma.
- Al buscar caminos estre dos nodos, correr Dijkstra o BF desde ambos para tener más información con la que trabajar.
- Si buscamos camino máximo, multliplicar los pesos por -1.
- Armar el DAG de caminos mínimos.

Siempre se nos pueden ocurrir maneras distintas para resolver un problema, estas son solo algunas técnicas que nos pareció interesante mostrarles. Otras pueden ser:

- Tener el grafo repetido y usar esos "niveles" para representar estados.

Conclusiones

- Como pasaba con los modelos de AGM, el objetivo del los problemas es **abstraer una propiedad y traducirla a grafos** para resolverlo.
- Luego, **aplicamos los algoritmos** comunes, calculando bien la complejidad en función de los tamaños de los modelos.
  - ! Hay que demostrar que el algoritmo es correcto, es decir, que resuelve el problema.
  - ! Por eso está bueno **evitar modificar los algoritmos que sabemos que funcionan**, ya que simplifica las demostraciones. (Sino, de modificar por ejemplo Dijkstra, deberían volver a demostrar que el algoritmo es correcto).