Práctica BFS, Dijkstra y AGM

Compilado: 14 de mayo de 2025

Recorridos en anchura

1. ⋆Un árbol generador T de un grafo G es *v-geodésico* si la distancia entre v y w en T es igual a la distancia entre v y w en G para todo $w \in V(G)$. Demostrar que todo árbol BFS de G enraizado en v es v-geodésico. Dar un contraejemplo para la vuelta, i.e., mostrar un árbol generador v-geodésico de un grafo G que no pueda ser obtenido cuando BFS se ejecuta en G desde v.

2. Diseñar un algoritmo de tiempo $O(n+m)$ que, dado un grafo conexo G con pesos en sus aristas y un vértice v, determine el árbol de menor peso de entre todos los árboles v-geodésicos de G. **Justificar** que el algoritmo propuesto es correcto. **Ayuda:** pensar cuáles aristas pueden pertenecer a un árbol v-geodésico cualquiera, para elegir las que minimicen el peso total.

3. Queremos diseñar un algoritmo que, dado un digrafo G y dos vértices s y t, encuentre el recorrido de longitud par de s a t que use la menor cantidad de aristas.

   a) Sea H el digrafo bipartito que tiene dos vértices v^0, v^1 por cada vértice $v \in V(G)$, donde v^0 es adyacente a w^1 en H si y solo si v y w son adyacentes en G. (Notar que $\{v^i \mid v \in V(G)\}$ es un conjunto independiente para $i \in \{0,1\}$.) Demostrar que $v_1, \ldots, v_k$ es un recorrido de G si y sólo si $v_1^1, v_2^0, \ldots, v_k^{k \bmod 2}$ es un recorrido de H.

   b) Sea $G^{=2}$ el digrafo que tiene los mismos vértices de G tal que v es ayacente a w en $G^{=2}$ si y solo si existe $z \in G$ tal que $v \to z \to w$ es un camino de G. Demostrar que G tiene un recorrido de longitud $2k$ si y solo si $G^{=2}$ tiene un recorrido de longitud k.

   c) Diseñar dos algoritmos basados en las propiedades anteriores para resolver el problema de encontrar el recorrido de longitud par de s a t que use la menor cantidad de aristas.

   d) Justifique cuál de los dos algoritmos es mejor, considerando: la complejidad temporal y espacial, la dificultad de la implementación y la posibilidad de modificar el algoritmo para encontrar recoorídos de longitud impar.

4. ⋆Se tiene una grilla con $m \times n$ posiciones, cada una de las cuales tiene un número entero en $[0, k)$, para un $k \in \mathbb{N}$ dado. Dado un valor objetivo $w \in \mathbb{N}$ y una posición inicial (x_1, y_1), que tiene un valor inicial v_1, queremos determinar la mínima cantidad de movimientos horizontales y verticales que transformen v_1 en w, teniendo en cuenta que el i-ésimo movimiento transforma a v_i por $v_{i+1} = (v_i + z) \bmod k$, donde z es el valor que se encuentra en la casilla de destino del movimiento. Por ejemplo, para la siguiente grilla y $k = 10$, se puede transformar $v_1 = 1$ en $w = 0$ con tres movimientos $1 \to 6 \to 4 \to 9$, aunque la solución óptima es vía el camino $1 \to 3 \to 6$.

1	3	6
6	7	4
4	9	3

   Modelar este problema como un problema de grafos que se resuelva usando BFS en $O(kmn)$ tiempo.

5. Proponer un algoritmo de complejidad temporal $O(ds)$ que, dados dos naturales $d > 0$ y $s > 0$, o bien encuentre el mínimo n divisible por d y cuyos dígitos sumen exactamente s o bien reporte que dicho n no existe. **Ayuda:** recuerde que n se puede decomponer en dígitos $n_1 \dots n_k$. Para cada $i = 1, \dots, k$, el número con digitos $n_1 \dots n_i$ tiene un resto d_i módulo d y una sumatoria s_i. Sabiendo que k debe ser mínimo, piense cómo cambian d_i y s_i al agregar el dígito n_{i+1}.

Algoritmo de Dijkstra

6. Dado un digrafo D con pesos $c\colon E(D) \to \mathbb{N}$ y dos vértices s y t, decimos que una arista $v \to w$ es *st-eficiente* cuando $v \to w$ pertenece a algún camino mínimo de s a t. Sea $d(\cdot, \cdot)$ la función que indica el peso de un camino mínimo entre dos vértices.
   a. Demostrar que $v \to w$ es st-eficiente si y sólo si $d(s, v) + c(v \to w) + d(w, t) = d(s, t)$.
   b. Usando el inciso anterior, proponga un algoritmo eficiente que encuentre el mínimo de los caminos entre s y t que no use aristas st-eficientes. Si dicho camino no existe, el algoritmo retorna $\perp$.
7. Diseñar un algoritmo eficiente que, dado un digrafo G con pesos no negativos, dos vértices s y t y una cota c, determine una arista de peso máximo de entre aquellas que se encuentran en algún recorrido de s a t cuyo peso (del recorrido, no de la arista) sea a lo sumo c. **Demostrar** que el algoritmo propuesto es correcto.
8. Diseñar un algoritmo eficiente que, dado un digrafo pesado G y dos vértices s y t, determine el recorrido minimo de s a t que pasa por a lo sumo una arista de peso negativo. **Demostrar** que el algoritmo propuesto es correcto.
9. Sea G un digrafo con pesos positivos que tiene dos vértices especiales s y t. Para una arista $e \notin E(G)$ con peso positivo, definimos $G + e$ como el digrafo que se obtiene de agregar e a G. Decimos que e mejora el camino de s a t cuando $d_G(s, t) > d_{G+e}(s, t)$. Diseñar un algoritmo eficiente que, dado un grafo G y un conjunto de aristas $E \notin E(G)$ con pesos positivos, determine cuáles aristas de E mejoran el camino de s a t en G. **Demostrar** que el algoritmo es correcto.
10. Sea G un digrafo con pesos positivos que tiene dos vértices especiales s y t. Decimos que una arista $e \in E(G)$ es *crítica* para s y t cuando $d_G(s, t) < d_{G-e}(s, t)$. Diseñar un algoritmo eficiente que, dado G, determine las aristas de G que son críticas para s y t. **Demostrar** que el algoritmo es correcto. **Ayuda:** pensar en el subgrafo P de G que está formado por las aristas de caminos mínimos de G (el "grafo de caminos mínimos").
11. En muchas aplicaciones se necesita encontrar caminos de *peso multiplicativo* mínimo en un digrafo D pesado con una función positiva $c\colon E(G) \to \mathbb{R}_{>1}$. Formalmente, el peso multiplicativo de un camino $v_1, \dots, v_k$ es la multiplicatoria de los pesos de sus aristas. Este tipo de caminos se buscan, por ejemplo, cuando los pesos de las aristas representan probabilidades de eventos independientes y se quiere encontrar una sucesión de eventos con probabilidad máxima/mínima. Modelar el problema de camino de peso multiplicativo mínimo como un problema de camino mínimo. **Demostrar** que el algoritmo es correcto. **Ayuda:** transformar el peso de cada arista usando una operación conocida que sea creciente y transforme cualquier multiplicatoria en una sumatoria.

Árbol generador mínimo, camino minimax y maximin

12. ⋆Se define la distancia entre dos secuencias de naturales $X = x_1, \dots, x_k$ e $Y = y_1, \dots, y_k$ como $d(X, Y) = \sum_{i=1}^{k} |x_i - y_i|$. Dado un conjunto de secuencias $X_1, \dots, X_n$, cada una de tamaño k, su grafo asociado G tiene un vértice v_i por cada $1 \leq i \leq n$ y una arista v_iv_j de peso $d(X_i, X_j)$ para cada $1 \leq i < j \leq n$. Proponer un algoritmo de complejidad $O(kn^2)$ que dado un conjunto de secuencias encuentre el árbol generador mínimo de su grafo asociado.

13. ⋆Una empresa de comunicaciones modela su red usando un grafo G donde cada arista tiene una capacidad positiva que representa su *ancho de banda*. El *ancho de banda* de la red es el máximo k tal que G_k es conexo, donde G_k es el subgrafo generador de G que se obtiene de eliminar las aristas de peso menor a k (Figura 1).

    a) Proponer un algoritmo eficiente para determinar el ancho de banda de una red dada.

    La empresa está dispuesta a hacer una inversión que consiste en actualizar algunos enlaces (aristas) a un ancho de banda que, para la tecnología existente, es virtualmente infinito. Antes de decidir la inversión, quieren determinar cuál es el ancho de banda que se podría obtener si se reemplazan i aristas para todo $0 \leq i < n$.

    b) Proponer un algoritmo que dado G determine el vector $a_0, \dots, a_{n-1}$ tal que a_i es el ancho de banda máximo que se puede obtener si se reemplazan i aristas de G.

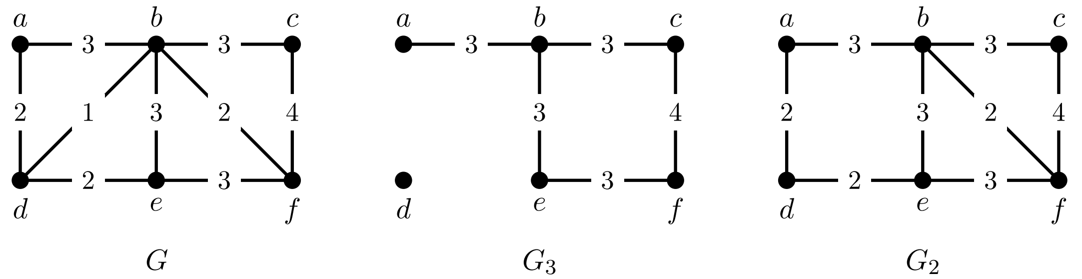

Figura 1. *El grafo G tiene ancho de banda 2 porque G_2 es conexo y G_3 no. Por otra parte, el ancho de banda del camino c, b, d es 1 mientras que el ancho de banda del camino c, b, e, d es 2. En general,* $\mathrm{bwd}(c, d) = 2$ *mientras que* $\mathrm{bwd}(a, e) = \mathrm{bwd}(b, f) = 3$.

14. Dado un grafo G con capacidades en sus aristas, el *ancho de banda* $\mathrm{bwd}_G(C)$ de un camino C es la mínima de las capacidades de las aristas del camino (Figura 1). El *ancho de banda* $\mathrm{bwd}_G(v, w)$ entre dos vértices v y w es el máximo entre los anchos de banda de los caminos que unen a v y w (Figura 1). Un árbol generador T de G es *maximin* cuando $\mathrm{bwd}_T(v, w) = \mathrm{bwd}_G(v, w)$ para todo $v, w \in V(G)$. Demostrar que T es un árbol maximin de G si y solo si T es un árbol generador máximo de G. Concluir que todo grafo conexo G tiene un árbol maximin que puede ser computado con cualquier algoritmo para computar árboles generadores máximos. **Ayuda:** para la ida, tomar el AGM T' que tenga más aristas en común con T y suponer, para obtener una contradicción, que T' tiene una arista e' que no está en T. Luego, buscar una arista e en T que no este en T' tal que $(T' - e') + e$ sea también AGM para obtener la contradicción. Para la vuelta, tomar v y w en el AGM T' y considerar la arista xy de peso mínimo en el único camino

de T' que los une. Luego, mostrar que xy tiene un peso al menos tan grande como cualquier otra arista que une las componentes conexas de $T' \setminus \{xy\}$ que contienen a v y w.

15. $\star$El algoritmo de Kruskal (resp. Prim) con orden de selección es una variante del algoritmo de Kruskal (resp. Prim) donde a cada arista e se le asigna una prioridad $q(e)$ además de su peso $p(e)$. Luego, si en alguna iteración del algoritmo de Kruskal (resp. Prim) hay más de una arista posible para ser agregada, entre esas opciones se elige alguna de mínima prioridad.

    a) Demostrar que para todo árbol generador mínimo T de G, si las prioridades de asignación están definidas por la función
    $$q_T(e) = \begin{cases} 0 & \text{si } e \in T \\ 1 & \text{si } e \notin T \end{cases}$$
    entonces se obtiene T como resultado del algoritmo de Kruskal (resp. Prim) con orden de selección ejecutado sobre G (resp. cualquiera sea el vértice inicial en el caso de Prim).

    b) Usando el inciso anterior, demostrar que si los pesos de G son todos distintos, entonces G tiene un único árbol generador mínimo.

16. Sea $q\colon V(G) \to \mathbb{Z}$ una función inyectiva para un grafo G. Demostrar que G tiene un único árbol generador mínimo si y solo si el algoritmo de Kruskal con prioridad q retorna el mismo árbol que el algoritmo de Kruskal con prioridad $-q$.

Ejercicios integradores

17. ¡Un virus está atacando el país! Es un virus tan fuerte que hasta puede **controlar mentes**. Se tiene una matriz cuadrada M de $n \times n$ con valores en $\{C, -, +, V\}$. Los valores C representan ciudades, en donde el virus puede infectar a todos los habitantes que quiera, y V es la ciudad en la que apareció el virus por primera vez. El virus puede controlar a cualquier persona infectada para que vaya de una celda de la matriz a cualquiera de las 4 adyacentes: arriba, abajo, izquierda, y derecha. El tema es que es un virus vago, y cada vez que hace caminar a alguien de una celda a otra, gasta un poco de su código genético. Por eso, el virus quiere minimizar la cantidad de pasos totales que tiene que hacer con personas controladas para infectar a todos los habitantes del país.

    Los habitantes se dieron cuenta, y comenzaron el contraataque. Los valores $+$ de la matriz son zonas protegidas, por las que si pasa un individuo infectado, se cura automáticamente. Los valores $-$ son posiciones que no contienen nada, por donde el virus puede mover a una persona libremente.

    Proponer un algoritmo con complejidad temporal $O(kn^2)$ para resolver este problema, donde k es la cantidad de posiciones de M con valor 1. Por ejemplo, en la siguiente matriz la cantidad de pasos que tiene que hacer el virus es 13.

$-$	C	C	V	$-$
$-$	$+$	$+$	$-$	$-$
$-$	$-$	C	$+$	C
$-$	$-$	$-$	$-$	$-$
C	$-$	C	$-$	$-$

18. Sea F un bosque generador de un grafo G pesado con una función $c\colon E(G) \to \mathbb{R}$. Decimos que una arista vw es *segura* si v y w pertenecen a distintos árboles de F. La arista vw es *candidata para el árbol T de F que contiene a v* cuando vw es segura y $c(vw) \leq c(xy)$ para toda arista segura xy tal que $x \in T$. Considere el siguiente *(meta-)*algoritmo que computa un árbol generador mínimo de (G, c):

    1: Sea $F = (V(G), \emptyset)$ un bosque generador de G.
    2: Para $i = 1, \ldots, n-1$:
    3: Agregar a F una arista candidata de algún árbol T.

    a) Demostrar que el algoritmo anterior retorna un árbol generador mínimo T de G. **Ayuda:** hacer inducción en i y demostrar en cada paso que el bosque F_i es un subgrafo de un árbol generador mínimo de G.

    b) Mostrar que tanto Prim como Kruskal son casos particulares de este algoritmo en las que la arista candidata se determina usando una política específica. Concluir que la demostración anterior prueba la corrección de Prim y Kruskal en forma conjunta.

    Sin pérdida de generalidad, se puede suponer que todas las aristas de G tienen un peso diferente. En efecto, alcanza con extender $c(\cdot)$ a una función de pesos que $c'(\cdot)$ tal que $c'(v) = (c(v), v)$, donde v es el identificador del vértice (i.e., un numero en $[1, n] \cap \mathbb{N}$). Bajo la hipótesis de que todos los pesos son distintos, el algoritmo que consiste en insertar todas las aristas candidatas posibles a F en cada iteración computa un árbol generador mínimo por el inciso *a*). Este algoritmo fue propuesto por Borůvka en 1926, mucho antes de que Prim y Krukal propusieran los suyos.

    c) Describir una implementación simple del algoritmo de Borůvka que requiera $O(m \log n)$ tiempo cuando un grafo G con n vértices y m aristas es dado.