Práctica 6 - Árboles (Práctica complementaria)

Árboles recubridores

1. Determinar si cada una de las siguiente afirmaciones es verdadera o falsa, justificando adecuadamente.

   a) Si G es un grafo conexo y T un árbol recubridor de G, entonces existe un orden de los vértices de G tal que el algoritmo BFS tiene como salida al árbol recubridor T.

   b) Si G es un grafo conexo y T un árbol recubridor de G, entonces existe un orden de los vértices de G tal que el algoritmo DFS tiene como salida al árbol recubridor T.

2. *a*) Escribir un algoritmo basado en el BFS para determinar si un grafo es o no conexo.

   b) Escribir un algoritmo basado en el DFS para determinar si un grafo es o no conexo.

3. *a*) Sea G un grafo conexo ponderado, $v \in V(G)$ y e una arista con peso mínimo incidente en v. Probar que e está contenida en algún árbol recubridor de peso mínimo.

   b) Sea G un grafo conexo ponderado, $v \in V(G)$. Suponer que los pesos de las aristas incidentes en v son distintos. Sea e la arista con peso mínimo incidente en v. ¿Debe estar e contenida en todo árbol recubridor de peso mínimo?

4. *a*) Sea G un grafo ponderado. Probar que si, mientras sea posible, se elimina una arista de G con peso máximo cuya eliminación no desconecta a G, el resultado es un árbol recubridor de peso mínimo para G.

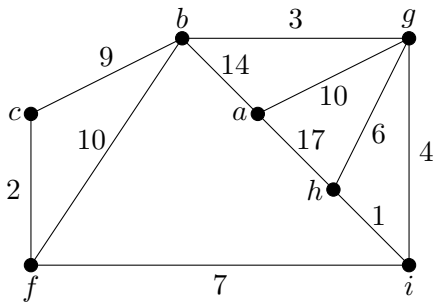
   b) Escribir un algoritmo que encuentre un árbol recubridor de peso mínimo en un grafo conexo ponderado utilizando el resultado del ítem anterior.

Algoritmo de Dijkstra

5. *a*) Modificar el algoritmo de Dijkstra y escribir un algoritmo que dado un grafo conexo ponderado G y un vértice $v \in V(G)$, devuelva $d(v, w)$ y dé un (v, w)-camino de longitud mínima para todo $w \in V(G)$.

   b) Modificar el algoritmo de Dijkstra y escribir un algoritmo que dado un grafo conexo ponderado G y dos vértices $v, w \in V(G)$, devuelva $d(v, w)$.

6. *a*) Aplicar el algoritmo de Dijkstra al grafo ponderado de la figura y determinar la distancia del vértice a a cada uno de los otros vértices.



   b) Dar un camino de longitud mínima desde el vértice a a los vértices c, f e i.

7. Determinar si cada una de las siguiente afirmaciones es verdadera o falsa, justificando adecuadamente.

   a) Sea G un grafo ponderado con $V(G) = \{v_1, v_2, \ldots, v_n\}$ y $w(e^*) < w(e)$ para toda arista $e \in E(G)$, con $e \neq e^*$ (donde $w(e)$ denota el peso asignado a la arista e). Si aplicamos el algoritmo de Dijkstra a G y calculamos la distancia de v_1 a cada uno de los vértices v_i para $2 \leqslant i \leqslant n$, entonces existe un vértice v_j, para algún $2 \leqslant j \leqslant n$, tal que la arista e^* se use en el camino más corto de v_1 a v_j.

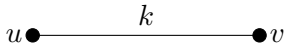
   b) El algoritmo de Dijkstra encuentra la longitud de una ruta más corta en un grafo conexo ponderado incluso si algunos pesos son negativos.

   c) Dados un grafo conexo ponderado G con $w(e) \geqslant 0$ para cada $e \in E(G)$ y $a, z \in V(G)$, el siguiente algoritmo devuelve la distancia entre a y z.

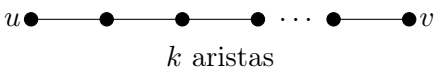
```
algo (w, a, z) {
    longitud = 0
    v = a
    T = conjunto de todos los vertices
    while (v not in z) {
        T = T - {v}
        seleccionar x in T con w(v,x) minimo
        longitud = longitud + w(v,x)
        v = x
    }
    return longitud
}
```

8. *a*) Escribir un algoritmo basado en el BFS que, dado un grafo no ponderado G y $v \in V(G)$, devuelva la distancia desde v a todos los demás vértices del grafo.

   b) Sea G un grafo ponderado en el que el peso de cada arista es un entero positivo. Sea G' el grafo obtenido a partir de G al sustituir cada arista



   en G de peso k por un camino P_k de aristas no ponderadas.



k aristas

   Demostrar que el algoritmo de Dijkstra para encontrar la distancia en el grafo ponderado G desde un vértice fijo v a todos los demás y realizar la búsqueda en anchura en el grafo no ponderado G' comenzando en el vértice v son el mismo proceso.

Árboles binarios

9. *a*) Sea T un árbol binario. Si $|V(T)| = n$, ¿cuál es la máxima altura posible de T?

   b) Sea T un árbol binario completo. Si $|V(T)| = n$, ¿cuál es la máxima altura posible de T en este caso?

   c) Repita los ítems anteriores para un árbol m-ario y m-ario completo respectivamente.

10. *a*) ¿Cuál es el número máximo de vértices internos que puede tener un árbol cuaternario completo de altura 8?

ESCUELA DE CIENCIAS EXACTAS Y NATURALES
DEPARTAMENTO DE MATEMÁTICA
Av. Pellegrini 250. Rosario
+54 0341 - 480 2649 internos 216 - 119

b) ¿Cuál es el número máximo de vértices internos que puede tener un árbol m-ario completo de altura h?

11. Sea N_h el número mínimo de vértices en un árbol binario balanceado de altura h, y sea $f_1, f_2, \ldots$ la sucesión de Fibonacci.

    a) Probar que $N_0 = 1$, $N_1 = 2$ y $N_2 = 4$.

    b) Probar que $N_h = 1 + N_{h-1} + N_{h-2}$, para $h \geqslant 2$.

    c) Probar que $N_h = f_{h+3} - 1$, para $h \geqslant 0$.

12. *a*) Cuatro monedas son idénticas en apariencia, pero una es defectuosa y es más pesada o más liviana que las otras, que pesan lo mismo. Dibujar un árbol de decisiones para proporcionar un algoritmo que identifique la moneda defectuosa (pero no necesariamente que determine si es más o menos pesada que las otras) usando sólo una balanza de cruz.

    b) Demostrar que se requiere pesar al menos dos veces para resolver el problema del ítem anterior.

    c) Repetir el ítem (a) pero ahora determinando si la moneda defectuosa es más pesada o más liviana que las otras.

    d) Determinar la menor cantidad de pesadas requeridas (en el peor caso) para resolver el problema del ítem (c).

13. Ocho monedas son idénticas en apariencia, pero una es defectuosa y es más pesada o más liviana que las otras, que pesan lo mismo. Dibujar un árbol de decisiones para proporcionar un algoritmo que identifique (usando la menor cantidad de pesadas en el peor caso) la moneda defectuosa y determine si es más pesada o más liviana que las otras usando sólo una balanza de cruz.

14. Dibujar un árbol de decisiones para proporcionar un algoritmo que ordene de menor a mayor cuatro números reales, usando la menor cantidad comparaciones en el peor caso.

15. Dibujar un árbol de decisiones para proporcionar un algoritmo que dados cuatro números reales determine cuál es el mayor de ellos, usando la menor cantidad de comparaciones en el peor caso.