Universitat Oberta de Catalunya

Estudios de Informática, Multimedia y Telecomunicación

ASIGNATURA: Grafos y Complejidad

Segunda PEC. Módulos 4 y 5.

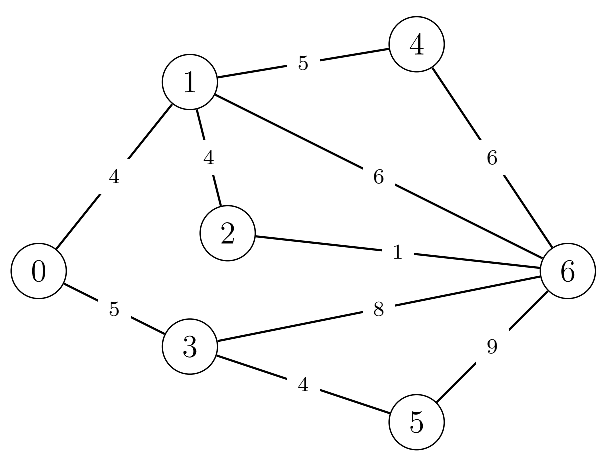
Semestre de primavera de 2012 (del 12 de abril al 2 de mayo).

Por favor, seguid las instrucciones siguientes:

- Enviad la solución en un fichero con el nombre:
  PEC1_Apellido1Apellido2Nombre.pdf
- Debe enviarse al apartado “Entrega y registro de EC” del aula.
- Enumerad las respuestas de acuerdo con la numeración de las preguntas y los apartados.
- Las respuestas a los problemas propuestos no deben limitarse a dar un resultado. Añadid, también, una explicación que justifique la respuesta.

1. (Valoración de un 20 %)

   El *Spanning Tree Protocolo* es un protocolo de red que evita la formación de lazos (*loops*) en una red formada por nodos (*bridges*) y segmentos de la red. Suponemos que representamos la red como un grafo donde los vértices representan nodos y las aristas son las conexiones entre los nodos. El peso de una arista representa el coste de cada segmento.



a) Utilizar el algoritmo más eficiente para calcular un árbol generador de este grafo. Representadlo gráficamente.

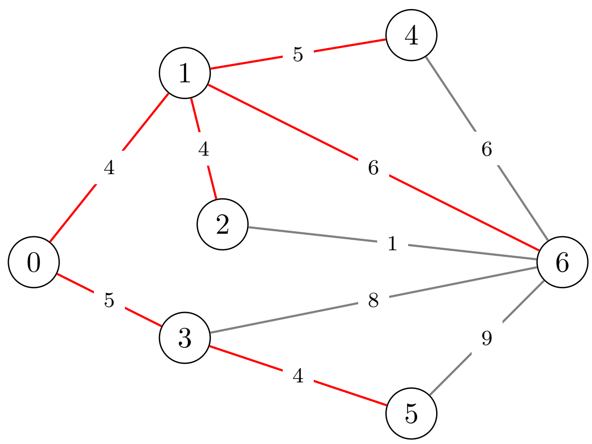
b) ¿El árbol obtenido en el apartado anterior es un árbol generador minimal? Justificar la respuesta.

c) Si la arista $\{4, 6\}$ queremos que forme parte de la solución, ¿el árbol resultante será un árbol generador minimal? Justificar la respuesta.

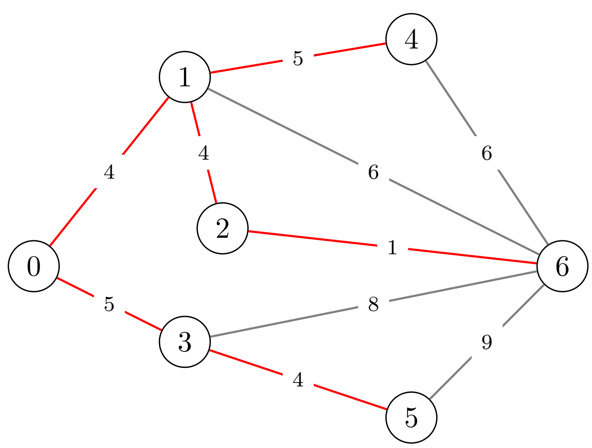
d) Demostrar que si las aristas de un grafo conexo G de orden n tienen todas un peso diferente entonces G tiene un único árbol generador minimal.

Solución:

a) Utilizando el algoritmo BFS empezando por el vértice 0 se obtiene el árbol generador:



b) El árbol obtenido con el algoritmo BFS tiene un peso de 28. Si aplicamos, por ejemplo, el algoritmo de Kruskal obtendremos el árbol,



con un peso de 23. Por lo tanto, el árbol obtenido con el algoritmo BFS no es un árbol generador minimal.

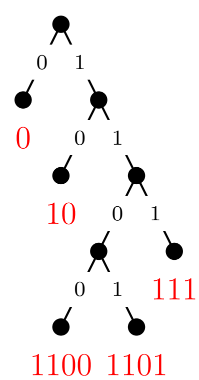
c) Si añadimos la arista $\{4, 6\}$ al árbol generador minimal obtenido en el apartado anterior, entonces formaremos el ciclo $\{4, 6, 2, 1, 4\}$. Por lo tanto, para obtener un árbol generador deberíamos sacar alguna de las aristas del ciclo y, como que todas tienen un peso más pequeño que la arista $\{4, 6\}$, el árbol obtenido no podrá ser un árbol generador minimal.

d) Si todas las aristas tienen un peso diferente entonces, aplicando el algoritmo de Kruskal, siempre elegiremos las $n-1$ aristas de peso más pequeño que determina el árbol generador minimal de forma única.

2. (Valoración de un 20 %)

Un código C se llama *prefijo* si ninguna sucesión de símbolos de C forma parte de otra sucesión de símbolos de C. Los códigos prefijos se utilizan en la codificación y en la compresión de datos (método de Huffman) puesto que permiten una descodificación eficiente. Por ejemplo, el código siguiente $C = \{0, 111, 1100, 1101, 10\}$ es un código prefijo mientras que el código $D = \{0, 10, 01, 101, 010\}$ no es un código prefijo.

Los códigos prefijos se pueden representar mediante árboles con raíz. Por ejemplo, el árbol siguiente representa el código prefijo C:



Las hojas del árbol se identifican con los símbolos (palabras-códigos) del código. Los códigos representados pueden ser sobre cualquier alfabeto: binarios, ternarios,...

a) Indicar cuáles de los códigos siguientes son prefijos y en caso afirmativo, representad el árbol correspondiente a cada código.

1) $C_1 = \{00, 01, 11, 100, 101\}$.
2) $C_2 = \{0, 01, 011, 0111\}$.
3) $C_3 = \{02, 12, 01, 120, 012\}$.
4) $C_4 = \{00, 01, 02, 10, 11, 12, 20, 21, 22\}$.

b) La longitud media de un código prefijo de n palabras-código se calcula por la fórmula

$$\bar{L} = \frac{1}{n} \sum_{i=1}^{n} L_i$$

donde L_i es la longitud (número de símbolos) de la palabra-código i. Por ejemplo, para el código C la longitud media es $\bar{L} = \frac{1+2+3+4+4}{5} = 2{,}8$. Los mejores códigos son los que tienen la menor longitud media.

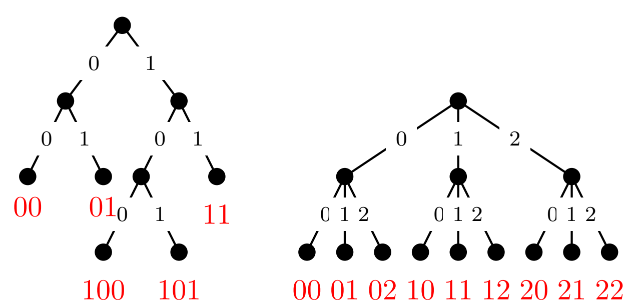
Relacionad la longitud media de un código prefijo con las propiedades del árbol que lo representa.

c) En el código ASCII cada carácter está representado por un código de 8 bits. ¿Es un código prefijo? En caso afirmativo decir qué altura tiene el árbol que lo representa, cuántas hojas tiene y qué clase de árbol es.

d) Queremos construir un código ternario para representar 243 caracteres. Queremos que sea un código prefijo de la menor longitud media posible. Utilizando su representación en forma de árbol, indicad como se construiría este código y cual sería su longitud media.

Solución:

a) Sólo C_1 y C_4 son códigos prefijos. Su representación gráfica es:



b) La longitud de cada palabra-código coincide con el nivel de la hoja que lo representa. Así, la longitud media del código sería igual a la suma de los niveles de todas las hojas dividido por el número de hojas.

c) El código ASCII es un código prefijo porque cada palabra-código tiene 8 bits y ninguna palabra-código puede coincidir con otra palabra-código. El árbol tendría altura 8, 256 hojas y sería un árbol binario completo.

d) Tendríamos que construir un árbol ternario de 243 hojas y con la mínima altura. Es decir, $243 \leq 3^h$ donde h es la altura del árbol. De aquí, $h \geq \log_3 243 = 5$. Por lo tanto, tomando $h = 5$ podemos construir un árbol ternario completo de exactamente 243 hojas. El código correspondiente tendría longitud media 5.

3. (Valoración de un 20 %)

Cuatro amigos, *Anna*, *Bob*, *Carlos* y *Diana*, van de vacaciones a una isla del caribe. La tabla siguiente representa los 9 puntos de interés cultural de la isla junto con las carreteras que los unen (de doble sentido) y la distancia en kilómetros entre ellos:

	$P0$	$P1$	$P2$	$P3$	$P4$	$P5$	$P6$	$P7$	$P8$
$P0$		12		6		5		4	
$P1$			7				8		2
$P2$				7				5	
$P3$					2				1
$P4$						9			
$P5$							6		15
$P6$								3	
$P7$									5

a) Dibujad el mapa de carreteras de la isla.

b) *Anna* quiere hacer diferentes circuitos (empezando y acabando en el mismo punto) que incluyan siempre un número par de puntos diferentes. ¿Lo podrá hacer?

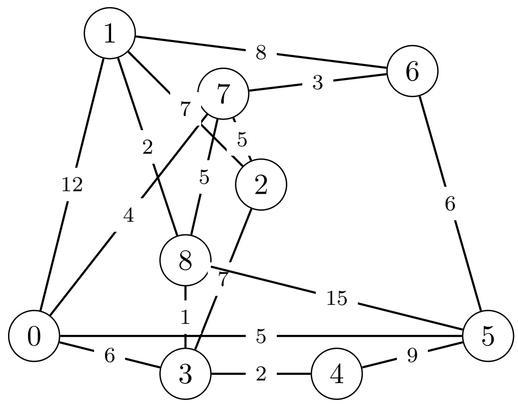
c) *Bob* quiere visitar todos los puntos de interés sin repetir ninguno y volver al punto de partida. ¿Podrá hacerlo? En caso afirmativo dar el recorrido que tendrá que seguir.

d) *Carlos* quiere recorrer todas las carreteras de la isla, independientemente del sentido, una sola vez y volver al punto de partida. ¿Lo podrá hacer? En caso afirmativo dar el recorrido que tendrá que seguir. En caso negativo, decir si lo podrá hacer aunque el punto de llegada sea diferente del de partida.

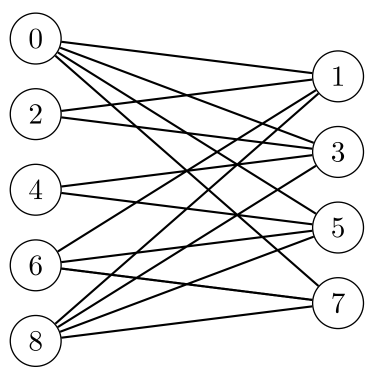
e) *Diana*, como Bob, quiere visitar todos los puntos de interés sin repetir ninguno y volver al punto de partida pero recorriendo el menor número de kilómetros posible. ¿Lo podrá hacer? En caso afirmativo explicad como se encontraría la solución. En caso negativo, calculad el número mínimo de kilómetros que tendría que recòrrer para visitar todos los puntos.

Solución:

a) El mapa de carreteras de la isla será:



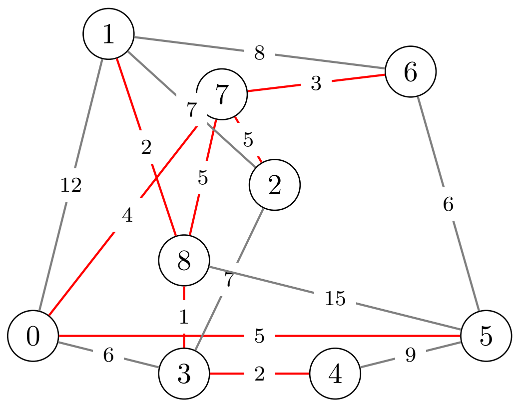
b) *Anna* lo podrá hacer si todos los lugares pertenecen a ciclos de longitud par. Este es el caso de nuestro grafo puesto que es bipartito y, por lo tanto, no contiene ciclos de longitud impar.



c) *Bob* quiere encontrar un ciclo hamiltoniano en el grafo. Esto no es posible puesto que el grafo es bipartito y los conjuntos de vértices de la partición no tienen el mismo cardinal.

d) *Carlos* quiere hacer un circuito euleriano. El grafo es conexo y tiene exactamente dos vértices de grado impar. Por lo tanto, no es posible encontrar un circuito euleriano pero sí un recorrido euleriano.

e) *Diana* quiere resolver el problema TSP sobre el grafo. Pero cómo que el grafo no es hamiltoniano no podrá resolver el problema. El número mínimo de kilómetros que tiene que recorrer será el peso del árbol generador minimal,



que tiene un peso mínimo 27.

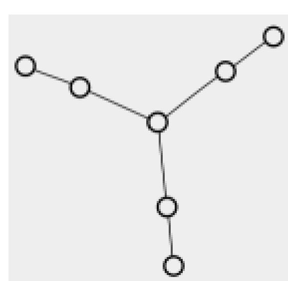
4. (Valoración de un 20 %)

a) Demostrar que un árbol con máximo grado k tiene, como mínimo, k hojas.

b) Un hidrocarburo saturado es una molécula C_iH_j, donde cada carbono (C) tiene cuatro enlaces y cada hidrógeno (H) tiene uno, y que no contiene ningún ciclo. Demostrar que una molécula de este tipo sólo puede existir si se verifica que $j = 2i + 2$.

c) Demostrar que un grafo G es un árbol si y sólo si no contiene ciclos y tiene un único árbol generador.

d) Un árbol A se denomina *oruga* cuando contiene como subgrafo a un grafo trayecto T, de manera que todo vértice de A, o bien pertenece a T, o bien es adyacent a un vértice de T. Demostrar que un árbol es una oruga si y sólo si no contiene como subgrafo al grafo Y de la figura.



Solución:

a) Lo demostraremos por inducción sobre el orden n. Evidentemente para $n = 1$ es cierto. Para ver el caso inductivo, consideramos un árbol T de orden $n+1$.
En primer lugar observamos que T tiene alguna hoja: eliminando un vértice cualquiera obtenemos un grafo con n vértices, que por hipótesis de inducción tiene al menos n hojas. Por lo tanto, T tiene también al menos n hojas, donde $n \geq 1$.
Ahora eliminamos una hoja cualquiera de T. El árbol resultante T' es un árbol con un cierto grado máximo k. Por hipótesis de inducción contiene, al menos, k hojas. Entonces T tiene, como mínimo, $k+1$ hojas, mientras que el grado máximo será k o $k+1$ (en función de si la hoja que tiene de más T que T' es o no adyacente a un vértice de grado k). En conclusión, hemos visto que el árbol T tiene grado máximo k o $k+1$, y $k+1$ hojas como mínimo, por lo tanto, tiene al menos tantas hojas como el grado máximo.
También puede razonarse de manera más directa considerando un vértice de grado máximo k y sus vecinos: aquéllos que no sean hojas tendrán, a su vez, vecinos que serán o no hojas. De nuevo, consideramos los vecinos de los que no lo sean. Este proceso acabará hallando k hojas. (Observemos que es necesario tener un árbol, en particular que no haya ciclos: en un grafo cualquiera podría haber ramas que se uniesen).

b) Aplicando la fórmula de los grados, $2m = 4i + j$. Como $m = n - 1$ por ser un árbol, y $n = i + j$, tenemos que $2m = 2(n-1) = 2(i + j - 1)$. De la primera y última de las igualdades obtenemos $4i + j = 2(i + j - 1)$. Operando se obtiene $j = 2i + 2$.

c) Sea G un árbol, debemos demostrar que el único árbol generador es él mismo: G tiene $n-1$ aristas, y todo árbol generador debe tener ese mismo número de aristas, por lo que debe contenerlas todas.
Por otro lado, supongamos que un grafo G de orden n tiene al menos dos árboles generadores. Como cada uno tiene $n-1$ aristas, y al menos tienen alguna arista diferente uno respecto al otro, agrupando las aristas de ambos tenim como mínimo n aristas de G, por lo que G no puede ser un árbol.

d) Si A contiene a Y, no puede ser una oruga, ya que, independientemente del trayecto T que consideremos, una de las hojas de Y ni pertenecerá al trayecto ni será adyacente a el.
Por otro lado, supongamos que A no es una oruga. Veremos que necesariamente debe contener al grafo Y. Consideremos cualquier subgrafo T de A que sea un trayecto. Sea $(x_1, ..., x_k)$ la succesión de vértices de este trayecto. Como A no es una oruga, hay un vértice y_0 que no pertenece a T, ni es adjacent. Como A es conexo, hay un camino desde y_0 hasta T, formado por una sucesión de vértices $(y_0, y_1, y_2, ..., y_l, x_s)$, donde x_s pertenece al trayecto T, y $l \geq 1$. Entonces los vértices $\{x_{s-2}, x_{s-1}, x_s, x_{s+1}, x_{s+2}, y_l, y_{l-1}\}$ nos determinen un subgrafo que es Y. Para

garantizar que existen x_{s-2} y x_{s+2} es necesario que s sea distinto de $1, 2, k-1$ y k, pero si no existiese un y_0 que cumpliese esto, entonces A sería una oruga.

5. (Valoración de un 20 %)

   a) Un húngaro quiere visitar las seis ciudades europeas de la tabla partiendo y acabando en Budapest. Justificar que podemos usar *TSP-aproximado* y aplicarlo.

	Bar	*Bel*	*Ber*	*Bru*	*Buc*	*Bud*
Barcelona		1528	1498	1063	1968	1499
Belgrado			999	1373	447	316
Berlín				652	1293	689
Bruselas					1770	1132
Bucarest						640
Budapest						

   b) Hallar una cota inferior del circuito óptimo. Demostrar que el circuito hallado en el apartado anterior no es el óptimo.

   c) Aplicar el algoritmo de Hierholzer a K_7.

Solución:

a) Aplicando el algoritmo de Prim desde Budapest, obtenemos estas aristas (en el orden en que se indican): $Bud - Bel, Bel - Buc, Bud - Ber, Ber - Bru, Bru - Bar$. Recorriendo el árbol obtenido en preorden obtenemos: $Bud, Bel, Buc, Ber, Bru, Bar$. Añadiendo Bud al final obtenemos el circuito hamiltoniano: Bud, Bel, Buc, Ber, Bru, Bar, Bud, con un total de 5270 Km.

b) La mitad del *TSP-aproximado* nos da una cota inferior, o sea 2635 Km (también sería una cota inferior la suma de las aristas del árbol generador minimal). Un circuito más corto al hallado sería, por ejemplo, $Bud, Ber, Bru, Bar, Buc, Bel, Bud$, con un total de 5135 Km.

c) Si numeramos los vértices del 1 al 7, una posibilidad sería la siguiente:

Iteración	v	C'	C
0	1		{1}
1	1	{1,2,3,4,5,6,7,1}	{1,2,3,4,5,6,7,1}
2	1	{1,3,5,7,2,4,6,1 }	{1,3,5,7,2,4,6,1,2,3,4,5,6,7,1 }
3	1	{1,4,7,3,6,2,5,1 }	{1,4,7,3,6,2,5,1,3,5,7,2,4,6,1,2,3,4,5,6,7,1 }