ALGORITMOS Y ESTRUCTURAS DE DATOS III
1$^{\underline{\text{er}}}$ Recuperatorio / 12-DIC-2012

Espacio reservado para los docentes:

Nota (Numérica)	Nota (Letras)	Docente

Completar los siguientes datos antes de entregar:

Nº Orden	Apellido y nombre	L.U.	Cantidad de hojas[1]

Fecha de entrega de notas: a determinar.
Por favor entregar esta hoja junto al examen.

1. Dados dos grafos $G_1 = (V_1, E_1)$ y $G_2 = (V_2, E_2)$, un homomorfismo de G_1 a G_2 es una función $f : V_1 \to V_2$ tal que $(v, w) \in E_1 \Rightarrow (f(v), f(w)) \in E_2$. Decimos que G_1 es homomorfo a G_2 si y sólo si existe un homomorfismo de G_1 a G_2.

   Sea G un grafo. Demostrar que G es bipartito o trivial si y sólo si es homomorfo a K_2.

2. Sea G un grafo. Sean v y w dos vértices de G tales que $G - v$ es conexo, $G - w$ es conexo, y $G - \{v, w\}$ no es conexo. Demostrar que existe un ciclo simple que contiene a v y w.

3. Se tiene una red de puertos conectados por cables, que se utilizan para mandar información de un puerto a otro, pasando eventualmente por puertos intermedios. El tiempo que demora la información para atravesar cada cable es proporcional al grosor del cable y a su longitud. Es decir, si un cable tiene grosor g_i y longitud l_i, el tiempo es $g_i \times l_i$. Sabemos que en algún momento no muy lejano, se van a cambiar *todos* los cables que conectan con el puerto 1 porque es el más antiguo y está en mal estado. Queremos saber por dónde mandar la información para que llegue lo antes posible desde un puerto cualquiera a otro cualquiera, tanto AHORA como LUEGO del cambio de cables. Modelar el problema como un problema de grafos. Diseñar dos algoritmos eficientes basado en grafos para resolver el problema planteado (antes del cambio, y después del cambio de cables). Calcular su complejidad. Mostrar la correctitud y determinar la complejidad del algoritmo propuesto. Justificar por qué el algoritmo planteado resuelve el problema.

4. Dadas dos cadenas de caracteres $s = (s[1], s[2], \ldots s[p])$ y $t = (t[1], t[2], \ldots t[q])$, se dice que t es una subsecuencia de s si y sólo si existen q valores $i_1 < i_2 \cdots < i_q$ tales que $t[j] = s[i_j]$ para $j = 1, 2, \ldots q$; por ejemplo, "`hoyeselfindelmundo`" tiene como subsecuencias a "`oye`", "`mudo`" y "`feudo`", entre otras. Un palíndromo es una cadena de caracteres capicúa; por ejemplo, "`SMS`", "`rallar`" y "`@`" son palíndromos. Diseñar un algoritmo eficiente que dada una cadena de n caracteres, indique la longitud de la subsecuencia más larga que sea un palíndromo. Mostrar la correctitud y determinar la complejidad del algoritmo propuesto. Justificar. El mejor algoritmo que conocemos tiene complejidad $O(n^2)$ y utiliza la técnica de programación dinámica. SUGERENCIA: considerar los palíndromo en la la porción de cadena cadena $s = (s[i], s[i+1], \ldots \ldots s[j])$

5. Sea $G = (V, E)$ un grafo conexo con pesos asociados a sus ejes. Dada cualquier partición de V en (V_1, V_2), se dice que un eje es liviano respecto de esa partición si y sólo si tiene peso mínimo dentro del conjunto de ejes que tienen un extremo en V_1 y otro en V_2. Este concepto está bien definido ya que al ser G conexo, existe al menos un eje entre V_1 y V_2.

   (a) Sea e cualquier eje de G. Demostrar que e está en *algún* árbol generador mínimo de G si y sólo si existe una partición de V tal que e es *liviano* respecto de ella.

[1] incluyendo a esta hoja

(b) Sea e cualquier eje de G. Demostrar que e está en *todo* árbol generador mínimo de G si y sólo si existe una partición de V tal que e es *el único eje liviano* respecto de ella.

(c) Demostrar que si para toda partición de V existe un único eje liviano respecto de ella, entonces G tiene un único árbol generador mínimo. ¿Vale la recíproca? Justificar.