ALGORITMOS Y ESTRUCTURAS DE DATOS III
1er Parcial / 13-OCT-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.

   (a) ¿Existe algún grafo G tal que para todo grafo H se cumple que G es homomorfo a H?
   (b) ¿Existe algún grafo H tal que para todo grafo G se cumple que G es homomorfo a H?
   (c) ¿Existe algún grafo H tal que para todo grafo bipartito G se cumple que G es homomorfo a H?

   En caso afirmativo, caracterizar en términos simples todos los grafos que cumplen lo pedido y justificar. En caso negativo, indicarlo y justificar.

2. Sea T un árbol no trivial, y por lo tanto bipartito. Sea $\{V_1, V_2\}$ una partición de los vértices de T tal que todos los ejes tienen un extremo en V_1 y otro en V_2. Supongamos sin pérdida de generalidad que $|V_1| \geq |V_2|$. Demostrar que V_1 contiene al menos una hoja.

3. Sea G un grafo o digrafo con longitudes no negativas asociadas a sus ejes. Sean v_1, v_2 y v_3 vértices de G. Diseñar un algoritmo eficiente para encontrar un camino de v_1 a v_3 que pase por v_2 y que tenga longitud total mínima entre todos los caminos de ese tipo. Mostrar la correctitud y determinar la complejidad del algoritmo propuesto. Justificar. El mejor algoritmo que conocemos tiene la misma complejidad que el algoritmo de Dijkstra.

4. Se tiene una expresión lógica formada por n constantes lógicas, con $n - 1$ operadores lógicos intercalados entre ellas. Cada una de las constantes es TRUE (verdadero) o FALSE (falso). La cantidad de operadores distintos que aparece en la expresión es acotada, y para cada uno de ellos se conoce su tabla de verdad. El resultado de evaluar la expresión, en general depende del orden en que se aplican los operadores. Por ejemplo, si la expresión es $\text{TRUE} \vee \text{FALSE} \Rightarrow \text{TRUE} \wedge \text{FALSE}$, dos formas posibles de evaluarla son

$$\begin{aligned} \text{TRUE} \vee ((\text{FALSE} \Rightarrow \text{TRUE}) \wedge \text{FALSE}) &= \text{TRUE} , \\ (\text{TRUE} \vee \text{FALSE}) \Rightarrow (\text{TRUE} \wedge \text{FALSE}) &= \text{FALSE} . \end{aligned}$$

   Diseñar un algoritmo eficiente que indique si existe un orden de aplicación de los operadores que hace que el resultado de evaluar la expresión sea TRUE. Mostrar la correctitud y determinar la complejidad del algoritmo propuesto (temporal y espacial). Justificar. Sólo se considerarán eficientes los algoritmos que tengan complejidad temporal $O(n^3)$.

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.
   (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.

[1] incluyendo a esta hoja