

Estructuras de Datos y de la Información II

Examen parcial – 14 de abril 2004

1. Demostrar que todo grafo no dirigido conexo contiene un vértice que puede ser eliminado de forma que el grafo sigue siendo conexo. Diseñar un algoritmo que dado un grafo conexo no dirigido $G = (V, E)$ encuentre dicho vértice en tiempo $O(V+E)$.

Solución

Existe un árbol T , eliminar una hoja del árbol. DFS (o BFS).

Una hoja es un nodo u para el que no existe ningún v con $\pi[v] = u$.

Existe al menos una hoja u_0 , porque si no habría un ciclo en el árbol, tomando antecesores partiendo de cualquier nodo u con $\pi[u] \neq \text{NIL}$.

Los únicos nodos que pueden perder conectividad al eliminar u_0 son los adyacentes a u_0 en el árbol. El único nodo adyacente a u_0 en T , que pierde un arco al eliminar u_0 , es $\pi[u_0]$. Los caminos que conectan $\pi[u_0]$ con todos los demás nodos en T no pasan por u_0 , ya que después del arco $(\pi[u_0], u_0)$ sólo cabría retroceder a $\pi[u_0]$. Luego al eliminar $\pi[u_0]$, $\pi[u_0]$ sigue estando conectado con todos los demás nodos, el árbol resultante es conexo, y por lo tanto lo es G sin u_0 .

2. Escribir un algoritmo que determine si a partir de un nodo inicial r de un grafo G es posible alcanzar todos los nodos de G por un único camino. Justificar cuál es la complejidad del algoritmo.

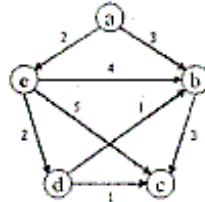
Solución

Variante de BFS: si en el transcurso de BFS (G, r) se encuentra un nodo con estado $\neq N$, el camino no sería único. Si al finalizar BFS (G, r) quedan nodos con estado $= N$, habría nodos no accesibles desde r .

También se puede utilizar DFS de forma parecida.

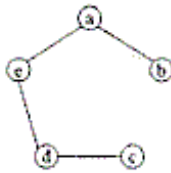
Complejidad: $O(V+E)$ igual que BFS.

3. Hallar los caminos de coste mínimo desde el nodo a hasta todos los demás, utilizando el algoritmo más eficiente aplicable al siguiente grafo:



Mostrar paso a paso la ejecución del algoritmo.

Solución



```
{ (a, 0, NIL), (b, ∞, NIL), (c, ∞, NIL), (d, ∞, NIL), (e, ∞, NIL) }
{ (b, 3, a), (c, ∞, NIL), (d, ∞, NIL), (e, 2, a) }
{ (b, 3, a), (c, 7, c), (d, 4, c) }
{ (c, 6, b), (d, 4, c) }
{ (c, 5, d) }
{ }
```