Algoritmos y complejidad

Actividad 15

Integrantes:

* Kloster, Matías Nicolas. LU 110873.
* Baschiera, Luciano Manuel. LU 111847.

Profesor:

* Fillottrani, Pablo R.

# Análisis teórico Espacio-Temporal

## Determinar si un grafo no-dirigido y pesado es conexo:

### Implementando un recorrido por niveles (BFS):

Para un grafo . , .

La implementación del algoritmo se encuentra en la clase Conexo, en la carpeta src, bajo el nombre de “conexoBFS”.

Nuestra implementación consiste en realizar un recorrido por niveles en el grafo, y luego recorrer la lista de nodos “padre” para determinar si hay más de un nodo que tenga -1 como padre, lo que indica que este grafo no es conexo.

El tiempo de ejecución de conexoBFS será:

* Θ (n) para setear todos los nodos a color blanco, y poner -1 como padre de cada nodo.
* Θ (1) para crear una nueva cola enlazada.
* Θ (n\*
* (1 para setear el color del nodo actual a gris.
* + 1 para encolar el nodo en Q.
* + n) para VisitarBFS: Para cada nodo en la cola, pregunta por los arcos adyacentes, para con los cuales obtiene el nodo en la otra punta, si es un nodo blanco, le cambia el color y el padre, y lo mete en la cola (Todo Θ(1)). Se puede probar por inducción que cada nodo es metido en la cola una sola vez, por tanto ejecuta n dequeues en total.

Entonces:

El espacio de ejecución de conexoBFS será:

* Θ(3\*n) para los arreglos de color, padre y la cola.
* Θ(n2) para las listas de adyacencia de los n nodos (Puede considerarse n si asumimos que reutilizamos la misma lista para cada nodo)

Entonces: o si tenemos en cuenta la consideración anterior.

### Mediante el uso de la estructura disjoint-set:

Para un grafo . , .

La implementación del algoritmo se encuentra en la clase Conexo, en la carpeta src, bajo el nombre de “conexoDS”.

Nuestra implementación consiste en crear n conjuntos, uno para cada nodo, y luego para cada arco del grafo, unir los conjuntos que contienen a los nodos del arco, hasta que haya un solo conjunto disjunto.

El tiempo de ejecución de conexoDS será analizado en función de la totalidad de las operaciones de la estructura de datos Disjoint Set:

Dado que estamos usando la estructura de datos implementada con las 2 heurísticas que se proponen en el libro Cormen (union by rank & path compression), podemos decir el tiempo total de las m operaciones será del

Dado que es una función de crecimiento suave, ya que representa el logaritmo iterado, podemos decir a entradas de gran tamaño esta función está acotada por 4,

ie  **->**

El análisis de la totalidad de las operaciones se debe a que estamos en un problema