Universidad Simón Bolívar
Departamento de Computación y Tecnología de la Información
CI-2693 - Laboratorio de Algoritmos y Estructuras III
Trimestre Abril-Julio 2016

Prelaboratorio de la semana 5: DFS y BFS

1. Actividad a realizar

Se quiere que implemente los algoritmos sobre grafos de Búsqueda en Amplitud (*Depth-first search*, DFS) y Búsqueda en Profundidad (*breadth-first search*, BFS) [1] . Los algoritmos deben ser implementados como clases de Java y deben trabajar sobre grafos dirigidos. El Listado 1 muestra el esqueleto que debe ser la base de su código de DFS. De la misma forma, el Listado 2 presenta el esqueleto del código de BFS que debe completar.

```
1  import java.util.*;
2
3  public class DepthFirstSearchDirected {
4
5      public DepthFirstSearchDirected(Digraph G, int s) {
6         dfs(G, s);
7      }
8
9      public void dfs(Digraph G, int s) {
10     }
11
12     public Integer[] arcsVisited() {
13     }
14
15     public LinkedList<Integer> getDirectedPathTo(int v) {
16     }
17
18     public LinkedList<LinkedList<Integer>> getAllDirectedPath() {
19     }
20 }
```

Listado 1: Esqueleto del código de DFS a implementar.

```
1  import java.util.*;
2
3  public class BreadthFirstSearchDirected {
4
5      public BreadthFirstSearchDirected(Digraph G, int s) {
6         bfs(G, s);
7      }
8
9      public void bfs(Digraph G, int s) {
10     }
11
12     public Integer[] arcsVisited() {
13     }
14
15     public LinkedList<Integer> getDirectedPathTo(int v) {
16     }
```

```
17
18      public LinkedList<LinkedList<Integer>> getAllDirectedPath() {
19      }
20  }
```

Listado 2: Esqueleto del código de BFS a implementar.

A continuación se explican los métodos obligatorios que deben contener las clases mostradas en el Listado 1 y en el Listado 2.

dfs: Implementación del algoritmo de Búsqueda en Profundidad, dado un vértice de inicio.

bfs: Implementación del algoritmo de Búsqueda en Amplitud, dado un vértice de inicio.

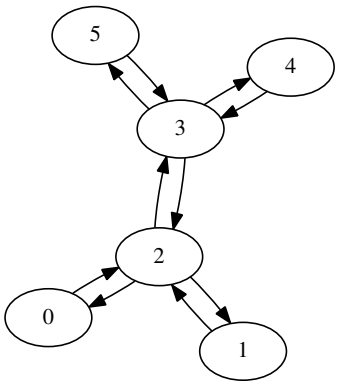
arcsVisited: Retorna un arreglo que indica para cada vértice, cual fue el vértice anteriormente visitado en la búsqueda. Por ejemplo, dado el grafo de la Figura 1a, si le aplica DFS partiendo desde el vértice 0, el algoritmo hace un recorrido cuya traza se muestra en la Figura 1b, por lo que el arreglo resultante que retorna el método es el siguiente

Vértice	0	1	2	3	4	5
Integer[]	null	2	0	2	3	3

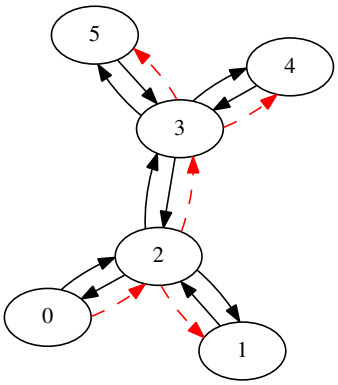
.

getDirectedPathTo: Dado un vértice v, retorna los vértices del camino que encontró el algoritmo búsqueda, desde el vértice fuente s hasta el vértice v, contenidos en una clase LinkedList. Por ejemplo, dado el recorrido mostrado en la Figura 1b que comenzó en el vértice 0, si se llama al método con el vértice 5, entonces el mismo debe retornar un objeto LinkedList conteniendo la siguiente secuencia de vértices: $0 \to 2 \to 3 \to 5$.

getAllDirectedPath: Retorna una lista con todos los caminos desde el vértice fuente s, hasta todos vértices que le son alcanzables.



(a) Grafo dirigido de ejemplo.



(b) Grafo de la Figura 1a, con la traza del recorrido de DFS en líneas rojas discontinuas, partiendo desde vértice 0.

Figura 1: Grafo de ejemplo y la traza del recorrido de DFS

2. Detalles de implementación

Todo el código debe seguir la guía de estilo de Java, y debe estar debidamente documentado siguiendo las reglas para la herramienta Javadoc, ver `http://www.oracle.com/`

`technetwork/articles/java/index-137868.html`. También debe crear un archivo Makefile que compile todo el código a entregar. Tenga cuidado que cuando coloque un archivo Java en el Makefile, sus dependencias se deben haber compilado antes.

3. Condiciones de entrega

Debe entregar el día 11 de Mayo de 2016 antes de las 11:00 am, un archivo comprimido llamado **PreLabSem5-X-Y.tar.gz**, con todos los códigos Java y el archivo Makefile. Las letras **X** y **Y** del archivo comprimido son los número de carné de los integrantes del equipo.

Referencias

[1] Meza, O., and Ortega, M. *Grafos y Algoritmos*, segunda ed. Editorial Equinoccio, 2004.

Guillermo Palma / gvpalma@usb.ve / Mayo 2016