# Laboratorio No. 5: Implementación de grafos

**Cristian Camilo Rendón Cardona**
Universidad Eafit
Medellín, Colombia
crendo11@eafit.edu.co

**Jhesid Steven Suerez Berrio**
Universidad Eafit
Medellín, Colombia
jssuarezb@eafit.edu.co

**3) Simulacro de preguntas de sustentación de Proyectos**

**1-**

**Ejersicio 1.3 =**
**[0, 2, 1, 3] → List**
**[0, 2, 1, 3] → Matrix**

**2-** En el ejercicio 1.1 para ambas implementaciones se utiliza el dato pareja, que contiene el vértice y el peso. A continuación se llama la clase DigraphAl para crear una LinkedList con el tamaño que pide el parámetro del constructor. Para DigraphAM se crea una matriz en el constructor del tamaó que el parámetro lo pida, y se llena la matriz con las posiciones [origen][destino].

**3-** Es más conveniente usar matrices cuando se tienen grafos pequeños ya que cuando se tienen grafos grandes las matrices ocuparían mucho espacio incensario.

**4-** Es mejor usar listas de adyacencia porque, al tener pocos nodos que se conectan entre sí, una matriz ocuparía mucho espacio innecesario ya que se trataría de una matriz dispersa.

**5-** Finalmente es posible decir que es mejor usar las Listas de adyacencia porque para grandes datos, las listas ocupan menos espacio y no vale la pena cambiar a matrices cuando se tienen pocos datos porque la velocidad no sería un problema.

**6-**

```
public static boolean ejercicio21(Digraph graph){
   int[] color = new int[graph.size];                                   //c1
   boolean[] visited = new boolean[graph.size];                         //c2
   color[0]=1;                                                          //c3
   return ejercicio21aux(graph, 0, color, visited);                     //c4
}
public static boolean ejercicio21aux(Digraph graph, int n, int[] color, boolean[] visited){
   if(!visited[n]){                                                     //c5
      visited[n]=true;                                                  //c6
      for(int i : graph.getSuccessors(n)){                              //c7*n
         if(color[i]==color[n]) return false;                           //c8*n
         if(color[i]==0){                                               //c9*n
            color[i]=3-color[n];                                        //c10*n
            return ejercicio21aux(graph,i,color,visited);               //c11*T(m-1)*n
         }
      }
   }
   return true;                                                         //c12
}
```

Numero de acciones en el peor de los casos es

$$T(n) = c1*n + c2*T(n-1)$$

$$T(n) = c_1\, c2^{n-1} + \frac{c1\left(c2^{n+1} - c2\,(n+1) + n\right)}{(c2-1)^2}$$

Por lo tanto, la complejidad es

O(2^n)

**7-** N es el número de vértices y m es el número de vértices visitados. Si se visitan todos los vértices, entonces m = n.

**DOCENTE MAURICIO TORO BERMÚDEZ**
**Teléfono: (+57) (4) 261 95 00 Ext. 9473. Oficina: 19 - 627**
**Correo: mtorobe@eafit.edu.co**

***4) Simulacro de Parcial***

1- a.

| | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
|---|---|---|---|---|---|---|---|---|
| 0 | | | | 1 | 1 | | | |
| 1 | 1 | | | | | 1 | | |
| 2 | | | | | 1 | | 1 | |
| 3 | | | | | | | | 1 |
| 4 | | | 1 | | | | | |
| 5 | | | | | | | | |
| 6 | | | 1 | | | | | |
| 7 | | | | | | | | |

2-

- 0 →[3,4]
- 1 →[1,5]
- 2 →[4,6]
- 3 →[7]
- 4 →[2]
- 5 →[]
- 6 → [2]
- 7 →[]

3- b

DOCENTE MAURICIO TORO BERMÚDEZ
Teléfono: (+57) (4) 261 95 00 Ext. 9473. Oficina: 19 - 627
Correo: mtorobe@eafit.edu.co