Tarea 2 Optimización de Flujo de Redes

Evely Gutiérrez Noda

27 de febrero de 2018

1. Introducción

En el siguiente reporte se abordará el tema estudiado en la clase de Optimización de Flujo de Redes. En dicha clase continuamos el estudio sobre la construcción de grafos, utilizando lenguaje Python [1] para la programación cuando queremos crear un grafo, además utilizamos la herramienta Gnuplot [2], la cual vinculamos con Python para representar el grafo antes programado.

Grafo Simple

Grafo Dirigido

Grafo Ponderado

Es el grafo al cual se le atribuye a cada arista un número específico, llamado ponderación o coste, y se obtiene así un grafo ponderado.

En un grafo ponderado se le llama peso de un camino, a la suma de los pesos de las aristas o arcos que lo forman. Ademas se puede determinar caminos mas cortos o mas largos entre dos vertices obteniendo el camino de menor peso entre los vertices o nodos.

Un grafo simple dirigido, se puede decir que es ponderado al atribuirle peso a cada arista.

2. Descripción del trabajo en clase

En la tarea anterior ya quedo programado un grafo con ciertas carecterísticas, ahora se utilizará este grafo que se creo, para convertirlo en un grafo ademas de simple, dirigido y ponderado.

Para realizar esta actividad comenzamos el trabajo con la estructura de clases en Python, la cual permitira crear metodos generales donde solamente se le cambia el valor a la cantidad de nodos y la probabilidad con que se uniran. De este modo se podra representar distintos tipos de grafos.

Se creo la **clase Grafo**, donde se definieron los parametros **n** (nodos), **x, y** (vertice x y vertice y respectivamente), un conjunto **E**, donde mas tarde se almacenara cada par de vertices que seran unidos por una arista, el color de esta arista y el peso correspondiente a la misma. También se creo el parametro **destino**, donde se almacenará el nombre del archivo donde se guardan los nodos y aristas creados, este fragmento de codigo se muestra a continuación.

```
class Grafo:

    def __init__(self):
```

```
        self.n = None # se crean las variables pero aun no se inicializan
        self.x = dict()
        self.y = dict()
        self.E = []
        self.destino = None
```

En la clase aprendimos como programar en Python un grafo a partir de una cantidad de nodos **n**, tomando por ejemplo una cantidad de nodos $\mathbf{n = 10}$, luego utilizando un ciclo **for** creamos un conjunto de nodos con valores aleatorios, los imprimimos en la consola y luego los almacenamos en un archivo de texto llamado **nodos.dat**. Para ello utilizamos las sentencias de código siguientes en la consola de Python:

Referencias

[1] Python, **http://www.python.org/**, 19 de Febrero de 2018

[2] Gnuplot, **http://www.gnuplot.info/**, 19 de Febrero de 2018