Tarea 5 Optimización de Flujo de Redes

Evely Gutiérrez Noda

6 de mayo de 2018

1. Introducción

En el reporte se abordará el tema estudiado en la clase de Optimización de Flujo de Redes. En dicha clase se continuó 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.

En la clase se realizaron experimentos con un tipo de grafo en particular, estos grafos fueron construidos basados en la **distancia de Manhattan** o la **geometría del taxista** como también suele llamarse.

2. Descripción del trabajo en clase

El trabajo realizado fue partiendo de un grafo cuyas características son similares a la geometría que describe la distancia de Manhattan. Esta es una forma de geometría en la que la métrica usual de la geometría euclidiana es reemplazada por una nueva métrica en la que la distancia entre dos puntos es la suma de las diferencias (absolutas) de sus coordenadas.

La **métrica del Taxista** también se conoce como **distancia rectilínea**, **distancia L**, **distancia de ciudad, distancia o longitud Manhattan**, con las correspondientes variaciones en el nombre de la geometría. El último nombre hace referencia al diseño en cuadrícula de la mayoría de las calles de la isla de Manhattan, lo que causa que el camino más corto que un auto puede tomar entre dos puntos de la ciudad tengan la misma distancia que dos puntos en la geometría del taxista de modo que el resultado de un camino desde un lugar a otro en esta ciudad quedaría como se muestra en la figura 1, donde las líneas azul y verde representan la distancia de Manhattan y la línea roja es para representar, en esa misma situación, la distancia euclidiana.

Figura 1: Distancia Manhattan y Distancia Euclidiana.

Para el trabajo con esta distancia se crearon grafos a partir de varios parámetros que definen sus nodos, sus aristas y en la manera que se conectan. Uno de los parámetros a considerar en este experimento es el parámetro de distancia **L**, el cual especifica el número de conexiones que va a tener un nodo con otro, teniendo en cuenta que si $\mathbf{L = 0}$, no existen conexiones en el grafo.

Otro parámetro considerado fue **K**, el cual define la cantidad de vértices resultantes en el grafo, de modo que si $\mathbf{K = 5}$, entonces la cantidad de nodos sería $\mathbf{N = 25}$, es decir **K** al cuadrado.

Se trabajó con una valor de $\mathbf{K = 5}$ en principio, para poder vizualizar correctamente las conexiones establecidas, luego este parámetro varió hasta 10. Por tanto la cantidad de nodos **N** varió entre **25 y 100**. Además se utilizó un parámetro **P** con valores muy bajos entre **0.0001 y 0.0010** para conectar aleatoriamente algunos nodos que no estén bajo la distancia de Manhattan (**L**), es decir que estarán conectados bajo la distancia euclidiana. Como resultado de lo antes descrito se crearon los grafos que se muestran en las figuras 2, 3 y 4, donde las aristas de color negro indican las conexiones dependiendo del valor de **L**, y las aristas de color azul, son las conexiones bajo la probabilidad de conexión **P**.

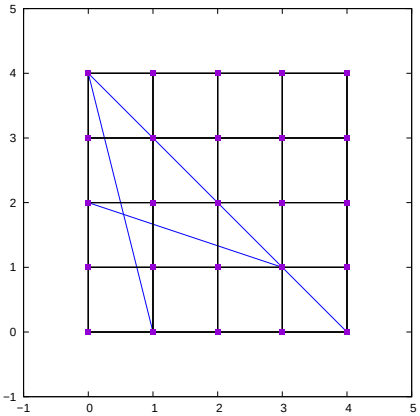

Figura 2: Grafo con L = 1 y P = 0.008

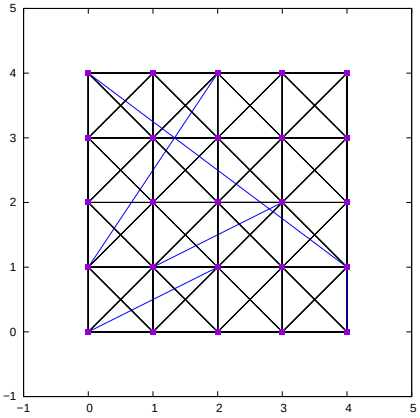

Figura 3: Grafo con L = 2 y P = 0.008

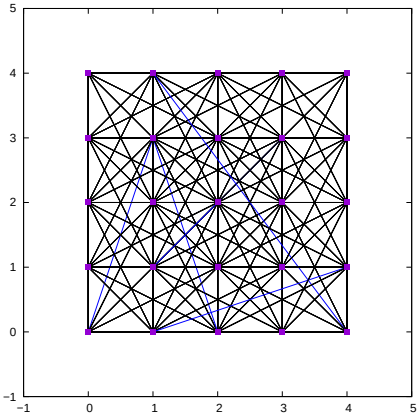

Figura 4: Grafo con L = 3 y P = 0.008

Para cada uno de estos grafos se calculó el flujo máximo, a medida que el valor de **L** iba aumentando desde 1 hasta 10, de este modo se iban aumentando las conexiones entre los nodos, por lo cual iba aumentando a su vez el flujo máximo en el grafo creado. Este flujo máximo se calculó utililzando el algoritmo **FordFulkerson** descrito en la Tarea 3 [3]. Este proceso se describe en el diagrama que se muestra en la figura 5.

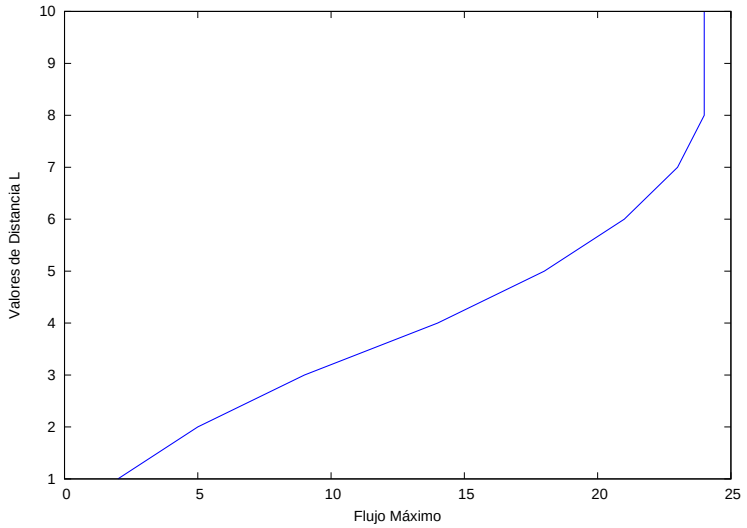

Figura 5: Distancia Manhattan L VS Flujo Máximo

Luego se realizó otro experimento, esta vez se programó una función para eliminar aristas al azar y de este modo, calcular el flujo máximo al quitar estas aristas, lo cual demostró que a pesar de quitar aristas el flujo máximo no vario notablemente, es decir, solo varió en pocas unidades.

Por último, se programó una función para eliminar nodos al azar del grafo, y como en el experimento anterior, calcular el flujo máximo que corre por el grafo, y ver qué sucede con el mismo. Al eliminar un nodo, también se eliminarán todas las aristas que estén conectadas a este nodo, de modo que el cambio en el flujo máximo puede ser más drástico, pero en este experimento el flujo se comportó sin muchas variaciones, indicando que el echo de que disminuyan las aristas o los nodos no influye directamente en la disminución del flujo máximo. También se midió el tiempo de ejecución del algoritmo utilizado para calcular el flujo máximo, el cual fue aumentando muy ligeramente. Algunos de los valores de como van disminuyendo los nodos y las aristas conectadas a estos nodos, se muestran en la salida de Python de la figura 6, así como también se ven los valores del flujo máximo cada vez que se elimina un nodo y sus aristas, y el tiempo de ejecución de este proceso completo.

Las funciones programadas para elimiar aristas y nodos al azar se muestran a continuación.

```
def EliminaArista(self, u,v):

    del self.aristas[u]
    del self.aristas[v]

    self.vecinos[u].remove(v)
    if not f:
        self.vecinos[v].remove(u)
```

```
def EliminaNodo(self, u):

    vecino = self.vecinos[u].copy()
    for i in vecino:

        self.EliminaArista(u,i)
    for n in self.nodos:
        if u in self.vecinos[n]:

            self.EliminaArista(n,u)

    h = self.nodos.pop(u)
```

```
Nodos iniciales******
888
Cant de Nodos
887
Cant Aristas
14632
Flujo Maximo
12
Tiempo de ejecucion:
18.854243169027654
Nodos iniciales******
887
Cant de Nodos
886
Cant Aristas
14602
Flujo Maximo
13
Tiempo de ejecucion:
18.863767674482858
```

Figura 6: Salida de Python.

Referencias

[1] Python Software Fundation, www.python.org/

[2] Gnuplot, www.gnuplot.info

[3] Tarea3, https://github.com/EvelyGutierrez/Optimizacion-de-flujo-de-redes/blob/master/Tarea2/VersionFinal/Tarea%202.pdf