# Tarea 3
# Optimización de flujo en redes

Beatriz Alejandra García Ramos

9 de abril de 2018

## 1. Introducción

En esta práctica se realizaron modificaciones en el código hecho para la segunda práctica [1] en donde se crea un grafo, ya sea dirigido y ponderado, solo uno de ellos o ninguno de los dos. Estas modificaciones fueron incluir dos algoritmos, el de Ford-Fulkerson [2] y el de Floyd-Warshall [3], los cuales nos ayudan a encontrar el flujo máximo de un nodo inicio a un nodo fin y el camino más corto de un nodo a otro, respectivamente.

Se obtuvieron resultados de los algoritmos y además se realizó un análisis de los tiempos de ejecución con variante en la cantidad de nodos del grafo, éstos se graficaron en cajas de bigotes utilizadas en estadística para visualizar la distribución del conjunto de datos y se realizó una comparación entre la manera en que los tiempos se comportan y la función exponencial.

## 2. Modificación del grafo en `python`

Dado que los algoritmos que se incluyeron en el código de `python` están de acuerdo a nodos y aristas con propiedad de conjunto y diccionario respectivamente, se hizo la modificación del código implementado para la segunda práctica llamada `Tarea2` de modo que los nodos fueran guardados en una variable `set`() para que no se incluyan dos nodos iguales y al conectarse las aristas estuvieran en una variable `dict`() que permite etiquetar a cada una de las aristas con la capacidad correspondiente a la conexión de un nodo con otro.

Dado que se realizaron modificaciones en el tipo de las variables se realizaron también los cambios necesarios al criterio de conexión de nodos y a las instrucciones que se le daban a `gnuplot` para realizar el grafo correspondiente. Además se creó una función en donde se guardan los nodos correspondientes al grafo, así pueden identificarse el nodo inicio, indicado con naranja, y el nodo fin, indicado con rojo, que se utilizan para el algoritmo de Ford-Fulkerson.

Anteriormente la visualización del grafo era poco apropiada, así que se realizaron modificaciones para que el grafo que se crea tenga colores cuando existen capacidades en las aristas, estos colores están dados por intervalos como se muestran en el cuadro 1, para las capacidades que van entre uno y tres se tiene el color morado en las aristas, cuando se tiene capacidad en la arista ente cuatro y seis entonces ésta se colorea de verde y si se tienen aristas de color azul sus capacidades pueden ir entre siete y diez.

Cuadro 1: Asignación de colores para las capacidades de las aristas.

| Color | Conjuntos |
|---|---|
| Morado | {1, 2, 3} |
| Verde | {4, 5, 6} |
| Azul | {7, 8, 9, 10} |

## 3. Inclusión de los algoritmos de Floyd-Warshall y Ford-Fulkerson

El algoritmo de Floyd-Warshall busca todos los caminos posibles de todos los nodos hacia todos los demás. En la práctica anterior se tenía un bosque, sin embargo, para que existieran más posibilidades de encontrar caminos de un nodo a otro se crearon aristas aleatorias que conectan nodos con otros aunque esto ya no forme un bosque, así cuando se tiene el algoritmo de Floyd-Warshall se toman en cuenta los caminos posibles de un nodo a otro y se guarda el de menos distancia.

En el caso en que se tiene el algoritmo de Ford-Fulkerson se toman en cuenta todas las capacidades de las aristas desde el nodo inicio hasta el nodo fin y de acuerdo al proceso de éste se calcula la mayor cantidad de flujo que puede enviarse sin importar qué camino se tome para llegar de un nodo a otro.

Para poder lograr que los algoritmos de Floyd-Warshall y de Ford-Fulkerson funcionen en cualquier tipo de grafo, ya sea dirigido, con pesos, con alguno de éstos, o con ninguno, se realizaron modificaciones en el código y una de las importantes para los algoritmos es si el grafo es dirigido o no, cuando es dirigido solo se toma el camino del primer nodo seleccionado al segundo nodo, sin embargo, cuando no es dirigido la dirección se toma del primer nodo al segundo nodo y del segundo al primero y la capacidad que se les asigna es la misma.

En la figura 1 se tienen grafos sin dirección, puede visualizarse que cuando los grafos tienen capacidades se involucran los colores en las aristas, cuando el grafo no es dirigido el algoritmo de Floyd-Warshall y el de Ford-Fulkerson realizan su trabajo, pero si se tiene que la capacidad de las aristas es cero entonces ambos obtienen resultados de cero en distancias y cero en flujo máximo. Por otro lado, cuando las aristas tienen capacidades, se tiene que el algoritmo de Floyd-Warshall obtiene la distancia mínima de un nodo a otro tomando en cuenta distintos caminos para llegar a él y el algoritmo de Ford-Fulkerson obtiene el flujo máximo entre el nodo inicio y el nodo fin y devuelve el valor de ese flujo, si es que existen aristas entre el nodo inicio y el nodo fin.

En el caso en que sí existe una dirección, como se muestra en la figura 2, y que además existen capacidades en las aristas se tiene que el algoritmo de Ford-Fulkerson tiene menos posibilidad de obtener un resultado, ya que los nodos inicio y fin podrían no tener acceso entre ellos de acuerdo a las direcciones que se tomaron aunque existan aristas entre ellos, de la misma manera afecta al algoritmo de Floyd-Warshall ya que se tiene que no todos los nodos pueden llegar a los demás. Cuando no hay capacidades entonces todos los resultados son iguales a cero.

## 4. Tiempos de ejecución de los algoritmos

Para el caso en que no se tienen capacidades los algoritmos tienen como resultado cero, sin importar si existe dirección o no.

Cuando se tiene que sí existen capacidades en las aristas mayores que cero entonces se tiene que el algoritmo de Floyd-Warshall encuentra una distancia mínima entre un nodo y otro siempre y cuando éstos estén conectados por aristas, pero cuando se tiene que existen conexiones pero con direcciones es posible que para algún nodo no existan entradas, solo salidas y ésto complica el resultado del algoritmo, aunque existan aristas, si no hay entradas a algún nodo entonces la distancia será cero y es por eso que en la teoría se tiene que la complejidad computacional del algoritmo de Floyd-Warshall es $O(V^3)$ donde $V$ es el número de nodos del grafo.

En el caso en que se tiene que sí existen capacidades para las aristas y no hay dirección el algoritmo de Ford-Fulkerson no tiene ningún problema en encontrar el flujo máximo cuando existen conexiones entre el nodo inicio y el nodo fin ya que se puede llegar de uno a otro en ambos sentidos, pero cuando existen direcciones se tiene que es probable que no haya una entrada al nodo fin desde el nodo inicio aunque haya aristas conectadas entre ellos, es por eso que la posibilidad de que exista un resultado de este algoritmo es mayor cuando no hay direcciones, es por eso que la complejidad computacional está dada por $O(E|f^*|)$ donde $f$ es el máximo flujo y $E$ es la cantidad de aristas en el grafo.

De acuerdo al funcionamiento y la complejidad computacional de cada uno de los algoritmos se tiene que se analizó el tiempo de ejecución del código en `python`, los tiempos fueron guardados en archivos `.csv` que después fueron importados al programa `R` para lograr hacer cajas de bigote que nos ayudan a visualizar el comportamiento de los tiempos de ejecución cuando se realizan repeticiones para

cantidades de nodos distintas. Además se verificó que las repeticiones para cada una de las cantidades de nodos no se distribuyen normalmente, es por eso que las cajas de bigote varían en su proporción.

De esta manera, como se puede ver en la figura 3 se tienen las repeticiones de las cantidades de nodos variantes cuando se hacen corridas de grafos que no tienen dirección. Cuando se tienen grafos con direcciones se obtuvieron los datos que se muestran en la figura 4.

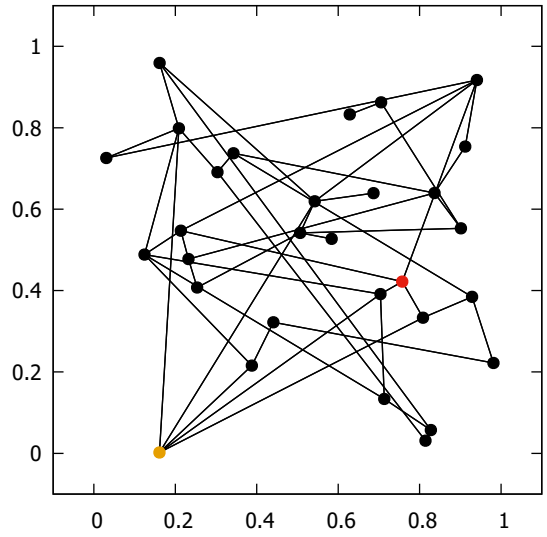
Es claro que cuando se tienen grafos sin dirección el tiempo de ejecución aumenta, ya que existen más posibilidades de encontrar caminos entre un nodo y otro, por lo cual la complejidad de ambos algoritmos aumenta y hace que el tiempo de ejecución aumente también. Aún así los tiempos de ejecución tanto cuando hay dirección como cuando no la hay no tienen gran diferencia entre ellos y en ambos casos se comportan de manera exponencial, como se muestra en la curva azul de las figuras, y este comportamiento es común cuando se aumentan los nodos ya que las aristas también aumentan y es un trabajo mayor para el sistema computacional, además se ve afectado por las demás aplicaciones que se utilizan en el momento en que se están ejecutando los grafos.

# 5. Conclusiones

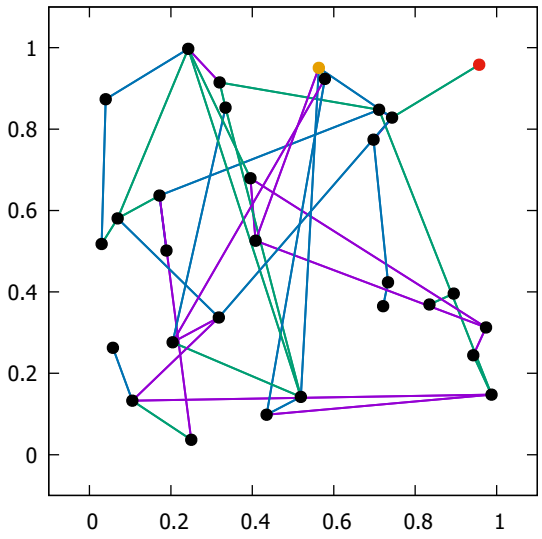
Las modificaciones que se hicieron en el grafo son de gran ayuda para que no existan tantas instrucciones dentro del código y se tenga una mejora en el tiempo de ejecución de los grafos. Además adaptando los algoritmos de Ford-Fulkerson y Floyd-Warshall no se les hacen modificaciones a éstos por lo que se puede asegurar que funcionan de manera correcta.

Cuando no se tienen capacidades en las aristas es claro que los algoritmos no pueden arrojar una respuesta distinta a cero y eso no es en lo que se enfoca el estudio de éstos, es por eso que siempre que se utilicen debemos asegurarnos de que sí existan capacidades en las aristas para que cuando se tenga un grafo dirigido o no se obtengan buenos resultados, aunque éstos sean cero en algunos casos por el hecho de que no exista conexiones entre los nodos con algunos otros.

Los tiempos de ejecución aumentan de manera significativa entre una cantidad de nodos y otra, en esta práctica se tienen variaciones de cinco nodos entre un análisis de tiempo y otro, y aún así logra verse que los tiempos aumentan de acuerdo a su tamaño y cada vez son más complejos de acuerdo a la complejidad computacional de ambos algoritmos.

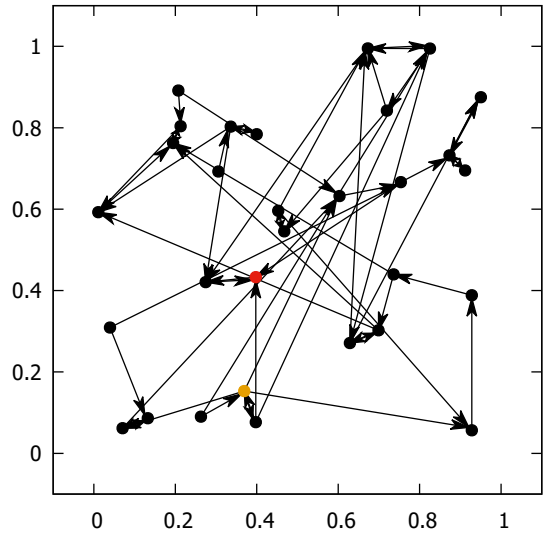



(a) Sin capacidad.

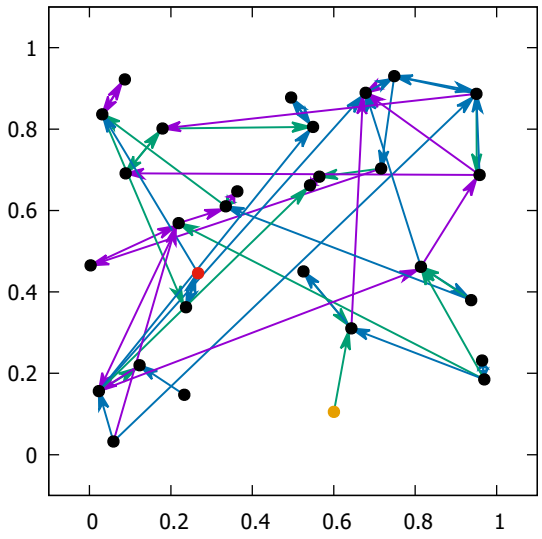



(b) Con capacidad.

Figura 1: Grafos sin dirección.

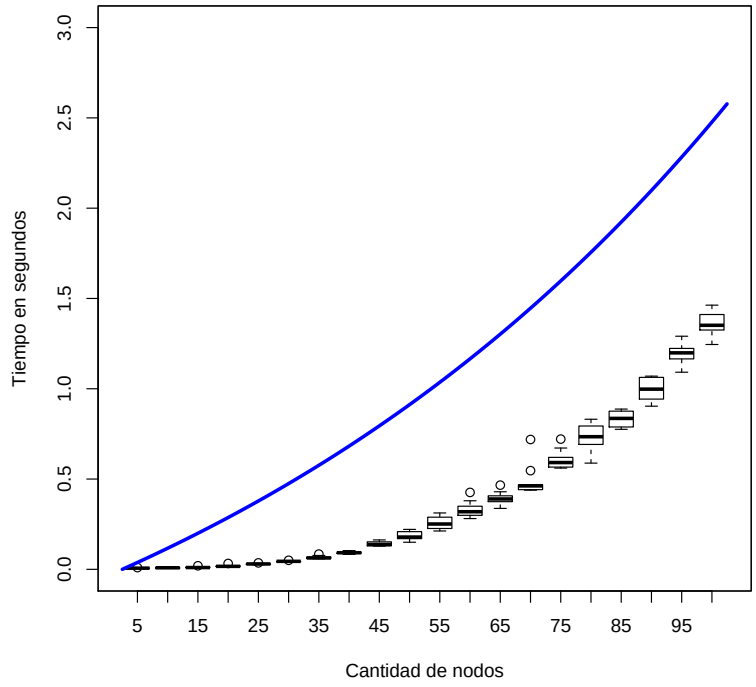



(a) Sin capacidad.




(b) Con capacidad.

Figura 2: Grafos con dirección.



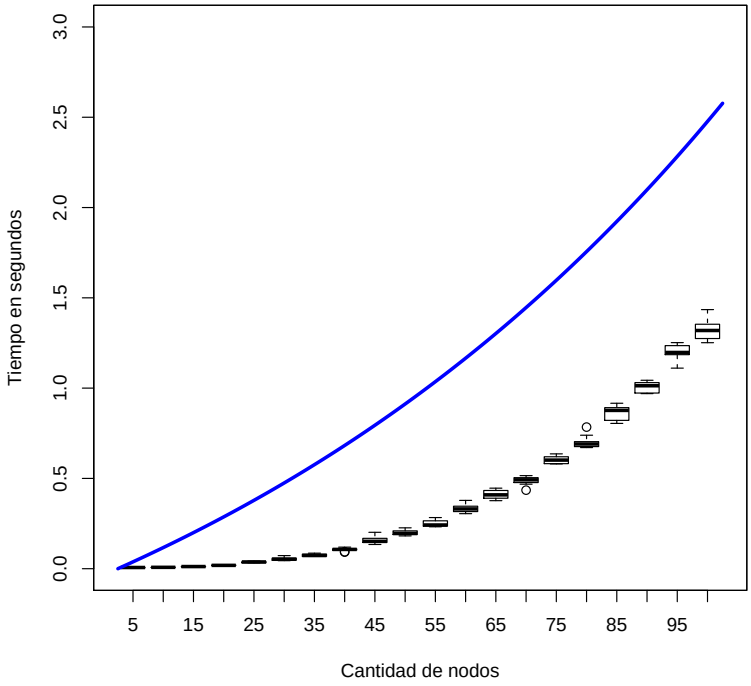
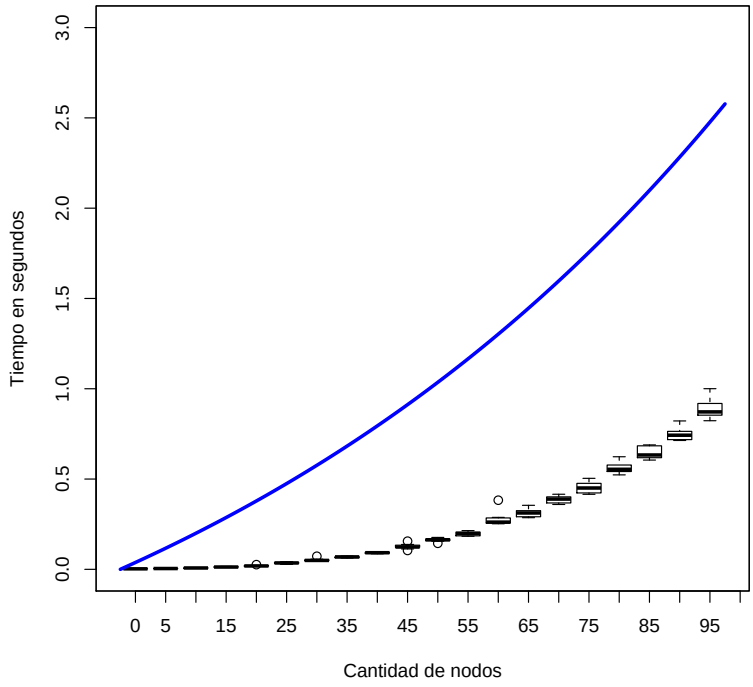




Figura 3: Grafos sin dirección.



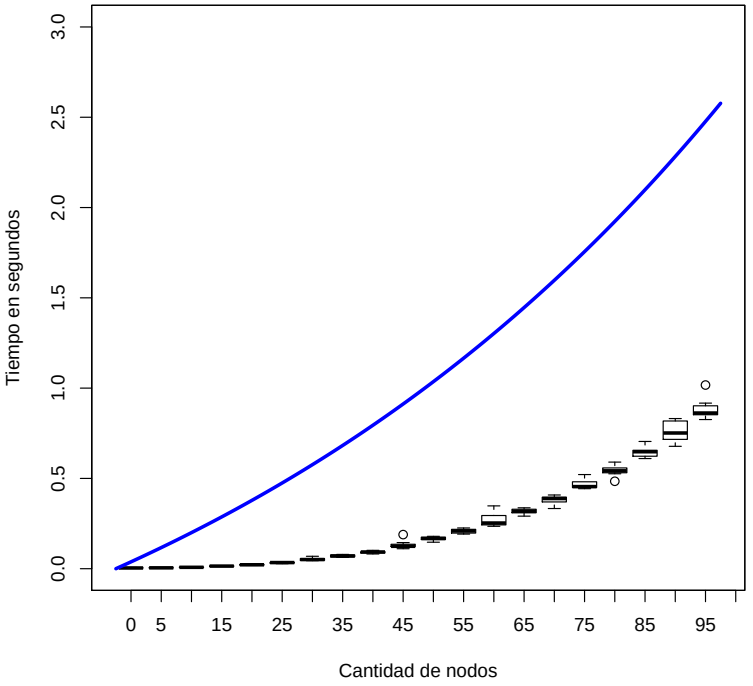




Figura 4: Grafos con dirección.

# Referencias


[1] Beatriz García. Tarea 2. Optimización de flujo en redes. Marzo de 2018. https://github.com/BeatrizGarciaR/FlujoEnRedes/blob/master/Tarea2/DirPeso.py

[2] Wikipedia. La enciclopedia libre. Algoritmo de Ford Fulkerson. Diciembre de 2017. https://es.wikipedia.org/wiki/Algoritmo_de_Ford_Fulkerson

[3] Wikipedia. La enciclopedia libre. Algoritmo de Floyd Warshall. Diciembre de 2017. https://es.wikipedia.org/wiki/Algoritmo_de_Floyd_Warshall