Experimentos con el algoritmo de Ford-Fulkerson en Python

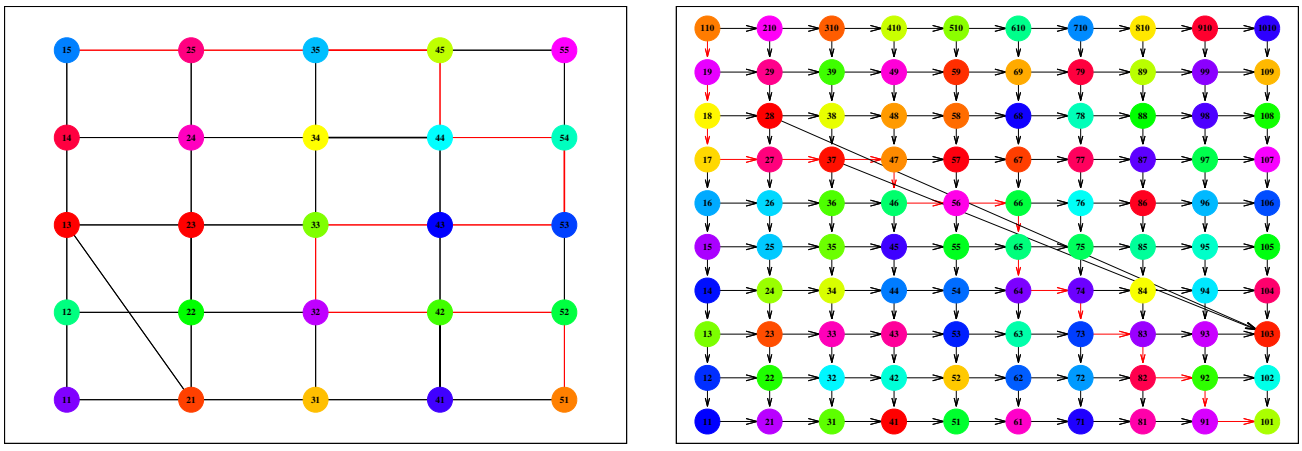
Astrid González

Optimización de flujo en redes

7 de mayo de 2018

Introducción

El algoritmo de Ford-Fulkerson [4] se basa en la existencia de caminos de aumento presentes en el grafo. Este algoritmo consiste en encontrar el flujo máximo de la red, es decir, el flujo máximo que va del nodo inicial o nodo fuente, al nodo final o nodo sumidero. Dado que su complejidad computacional depende del número de conexiones o aristas presentes en el grafo $G(E)$ así como de la capacidad de las mismas, se tiene que su complejidad corresponde a $O(E * f)$ [1].



(a) Grafo ponderado no dirigido de 25 nodos.

(b) Grafo dirigido no ponderado de 100 nodos.

Figura 1: Se presentan grafos dirigidos y ponderados con distintos valores de k, en donde se realizan recorridos estilo Manhattan para encontrar el camino del nodo fuente hacia el nodo sumidero.

Metodología

Para la generación de grafos ponderados y grafos dirigidos se utilizó la versión 3.6.4 del lenguaje de programación Python [5] acorde al procedimiento obtenido previamente [2, 3]. Posteriormente, fue utilizada la versión 4.6 de Gnuplot [6] como visualizador gráfico.

Se procedió a generar grafos con forma de rejilla con la finalidad de representar recorridos estilo Manhattan. Además de las conexiones establecidas, se agregaron por probabilidad hasta un máximo de dos conexiones extra de forma aleatoria. Ésto con el objetivo de representar la existencia de un salto probabilístico en las uniones. Cabe mencionar que en el caso de grafos ponderados la capacidad de las aristas se distribuyo de manera aleatoria dentro de una distribución de tipo normal con un valor medio de 2 y desviación estándar de 1.

Posteriormente se designó al nodo de la esquina superior izquierda como el nodo fuente y al nodo de la esquina inferior derecha como el nodo sumidero. Es decir que en un grafo donde $k = 5$ el nodo fuente corresponde al nodo 15 mientras que el nodo sumidero es el nodo 51.

Se procede a buscar un camino que recorra el grafo desde el nodo fuente hasta el nodo sumidero respetando las propiedades de los recorridos estilo Manhattan, donde las conexiones de dicho camino fueron resaltadas en color rojo. Una vez obtenido un camino del nodo fuente al nodo sumidero, se retiraron de forma recursiva hasta $n = 10$ aristas, donde en cada iteración se procede nuevamente a buscar un camino del nodo fuente al nodo sumidero tomando en cuenta la falta de las aristas removidas en dicho grafo.

Finalmente se registraron los tiempos de ejecución del algoritmo de Ford-Fulkerson y los valores obtenidos de flujo máximo para cada iteración donde se realiza la remoción de aristas.

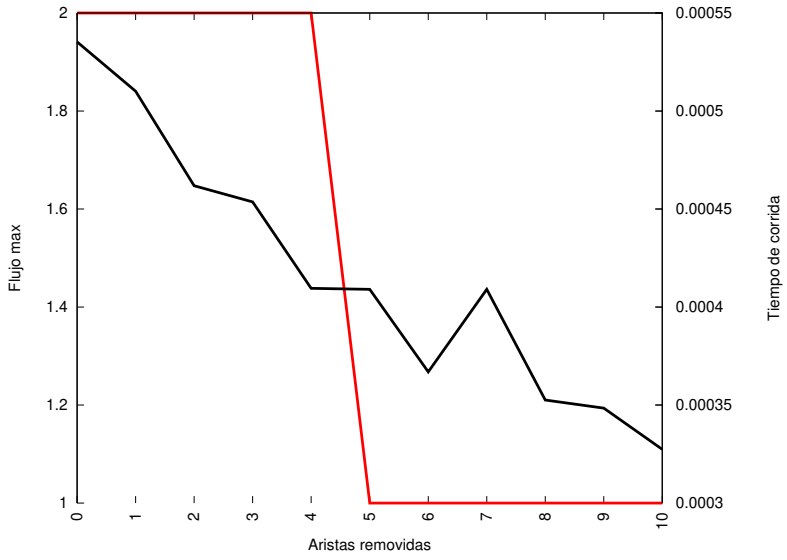


Figura 2: Evolución de los tiempos de corrida y flujo máximo para grafos ponderados donde $k = 5$, $p = 0.3$, $l = 2$, conforme se lleva a cabo la remoción de conexiones en el grafo.

Resultados

En la figura 1 se puede observar una comparativa de los recorridos estilo Manhattan en grafos ponderados y grafos dirigidos. Se puede observar que en los grafos ponderados el recorrido puede tomar cualquiera de las cuatro conexiones presentes dado que no se indica dirección alguna, por lo que se puede subir, bajar, moverse hacia la izquierda o la derecha pero no de forma diagonal, con la excepción de la conexión obtenida por probabilidad. Mientras tanto, en los grafos dirigidos este movimiento se restringe únicamente hacia abajo o la derecha.

Los tiempos de corrida o ejecución y los valores de flujo máximo se representan en la figura 2. En donde se puede apreciar que en grafos ponderados eventualmente ocurrirá una disminución en la cantidad de flujo máximo disponible, y a medida que existen menos conexiones en el grafo se reduce la cantidad de tiempo necesario para encontrar el camino del nodo fuente al nodo sumidero.

Por último, la figura 3 corresponde al recorrido realizado tras un proceso iterativo de remoción o percolación de aristas en un grafo dirigido donde $k = 5$. Se puede observar un camino del nodo fuente al nodo sumidero en cada una de las iteraciones. Cabe mencionar que el camino obtenido tras remover la arista n es completamente aleatorio al camino obtenido previamente tras la remoción de la arista $n - 1$.

Se puede observar que el recorrido obtenido en grafos ponderados es similar a una caminata aleatoria ya que existe al menos un paso t en el cual el nodo seleccionado se encuentra a mayor distancia del nodo sumidero que el nodo seleccionado en el paso $t - 1$. Mientras que en los grafos dirigidos no ocurre ésto debido a la restricción de dirección.

Referencias

[1] Gács, P. and László, L. (1999). Complexity of algorithms. `http://web.cs.elte.hu/~lovasz/complexity.pdf`. Consultado en Abril 2018.

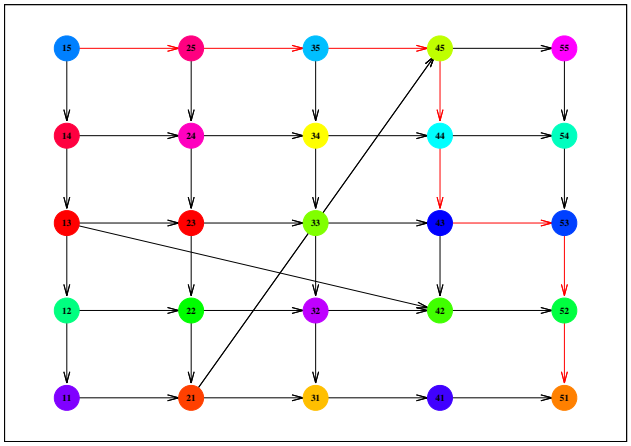
[2] González, A. (2018a). Medición experimental de la complejidad asintótica con python y gnuplot. *Universidad Autónoma de Nuevo León*, Optimización de flujo en redes.

[3] González, A. (2018b). Visualización de grafos simples, ponderados y dirigidos con gnuplot. *Universidad Autónoma de Nuevo León*, Optimización de flujo en redes.

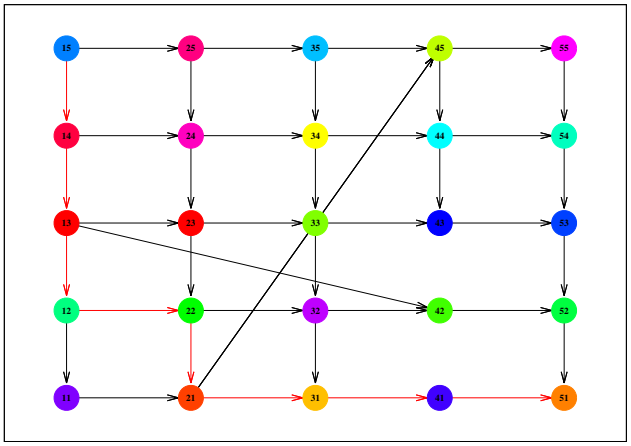
[4] Papadimitriou, C. H. and Steiglitz, K. (1998). *Combinatorial optimization: algorithms and complexity*. Courier Corporation. ISBN: 0-486-40258-4.

[5] Van Rossum, G. and the Python Development Team (2018). The python language reference. release 3.6.4. `https://docs.python.org/3.6/`. Consultado en Febrero 2018.

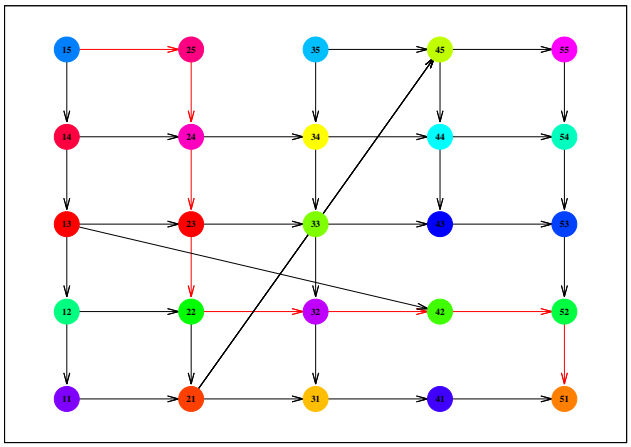
[6] Williams, T. and Kelley, C. (2013). Gnuplot 4.6: an interactive plotting program. `http://gnuplot.info/`. Consultado en Febrero 2018.



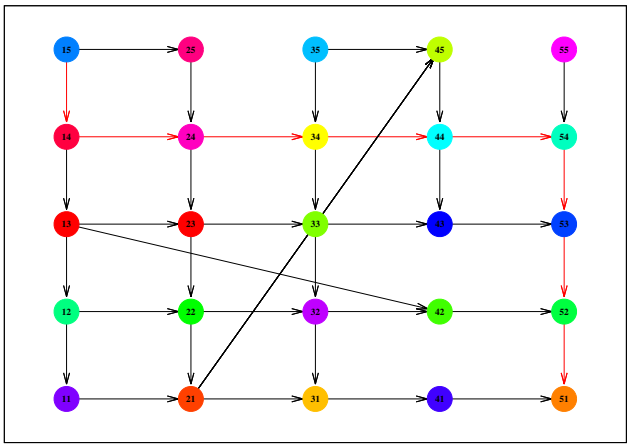
(a) Grafo dirigido $k = 5$, aristas removidas $= 1$.



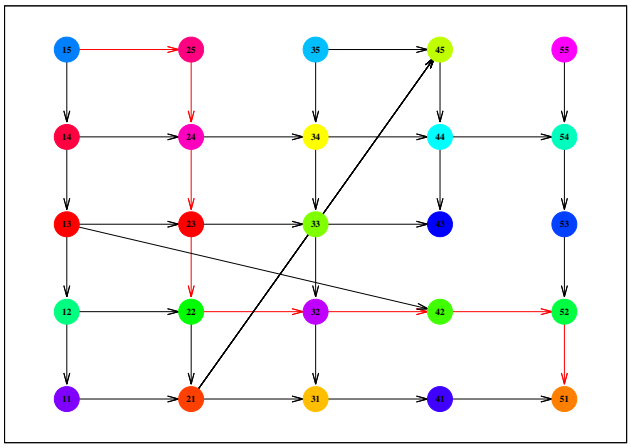
(b) Grafo dirigido $k = 5$, aristas removidas $= 2$.



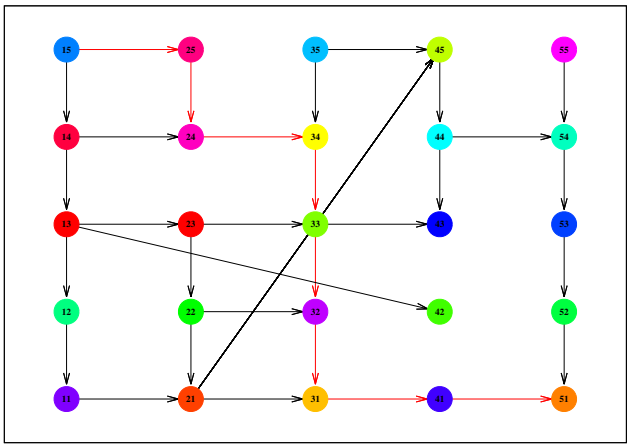
(c) Grafo dirigido $k = 5$, aristas removidas $= 3$.



(d) Grafo dirigido $k = 5$, aristas removidas $= 4$.



(e) Grafo dirigido $k = 5$, aristas removidas $= 5$.



(f) Grafo dirigido $k = 5$, aristas removidas $= 10$.

Figura 3: Proceso de remoción de aristas en un grafo dirigido y posterior recorrido estilo Manhattan.