Tarea 1: Visualización de grafos

Gabriela Sánchez Y.

Introducción

Un grafo como herramienta matemática se define como un par $G = (V, E)$ que consiste en un conjunto finito $V \neq \emptyset$ y un conjunto E de subconjuntos de dos elementos de V. Los elementos de V se llaman vértices y los elementos de E se llaman aristas [1]. La figura 1 muestra un ejemplo de un grafo.

Es una herramienta bastante útil ya que cualquier ambiente real en donde se encuentren grupos de sistemas o elementos interconectados, pueden ser representados mediante grafos.

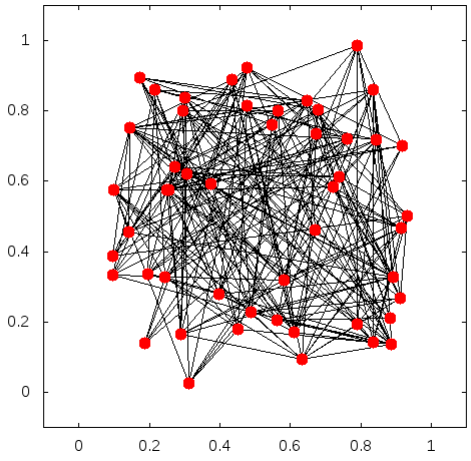


Figura 1: Ejemplo de un grafo.

Tipos de grafos

Existen diferentes tipos de grafos:

- simples: cada par definido de nodos es unido por una sola arista, en caso contrario se denomina multigrafo.
- ponderados: las aristas tienen un valor definido.
- difigidos: las aristas tienen definida una dirección.

El objetivo de la tarea es utilizar el lenguaje de programación `Python` en conjunto con `gnuplot` para graficar y visualizar diferentes tipos de grafos.

Visualización

Nodos

Para crear nodos, se generan al azar pares ordenados dentro del cuadrado unitario, los cuales representan las posiciones de los nodos. El tamaño de los nodos puede ser distinto dependiendo de lo que se desee representar, en la figura 2 podemos observar un ejemplo en el cual el tamaño de los nodos es variante.

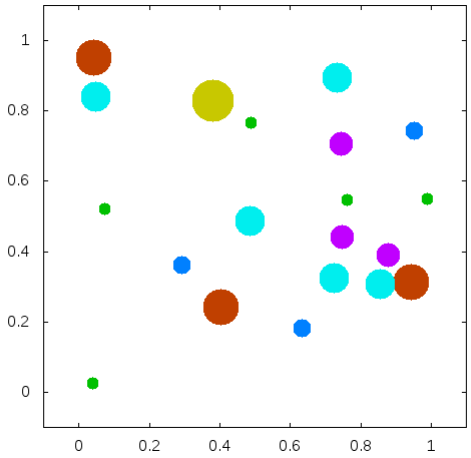
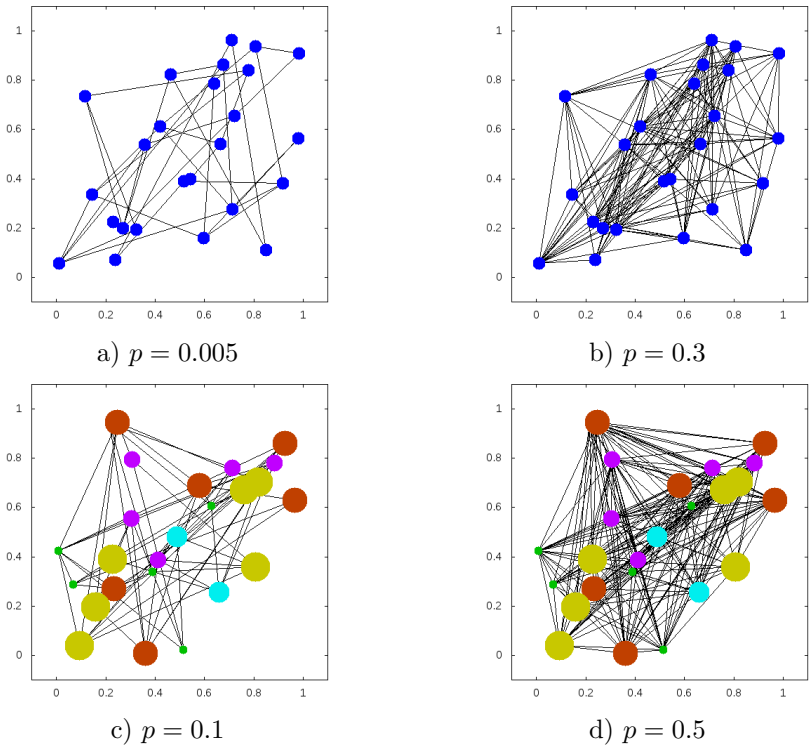


Figura 2: Ejemplo gráfico de nodos.

Aristas

La unión de los nodos puede realizarse de diferentes formas de acuerdo a lo que se desee representar. En este caso experimentaremos diferentes opciones.



a) $p = 0.005$ b) $p = 0.3$

c) $p = 0.1$ d) $p = 0.5$

Figura 3: Grafos con probabilidad de unión entre nodos.

La primera opción es considerar una probabilidad. Para cada par de nodos la unión entre ellos se realiza de acuerdo a una probabilidad p; se probaron cuatro diferentes opciones $p = \{0.005, 0.1, 0.3, 0.5\}$. La figura 3 muestra algunos resultados: en a) y b) el grafo mantiene un tamaño uniforme en sus nodos mientras que en c) y d) los nodos varían sus dimensiones. Observamos que mientras más grande es p, hay un número mayor de aristas en el grafo.

Grafos ponderados

La característica que distingue a los grafos ponderados es el asignar un valor a las aristas. El archivo `grafo_pond.py` permite crear este tipo de grafos.

Una vez que se realiza la creación de los nodos, la unión de los mismos se efectúa de acuerdo a una probabilidad. Si los nodos son del mismo tamaño, la probabilidad es uniforme para todos los nodos y la ponderación se asigna de manera aleatoria; mientras que si los nodos varían en sus dimensiones, la probabilidad y la ponderación de las aristas, depende de éstas dimensiones. La

figura 4 muestra un ejemplo en cada uno de los casos anteriores.

Grafos dirigidos

Como se ha definido anteriormente, los grafos dirigidos asignan a sus aristas una dirección. Modificamos el archivo `grafo_pond.py` agregando la siguiente sentencia que permite asignar una dirección a las aristas:

```
print('set arrow',i, 'from', x1,',',y1,'to',x2,',',y2, 'head
                filled size screen 0.03,15', file = archivo3)
```

En la figura 5 muestra un ejemplo de grafo dirigido para nodos uniformes.

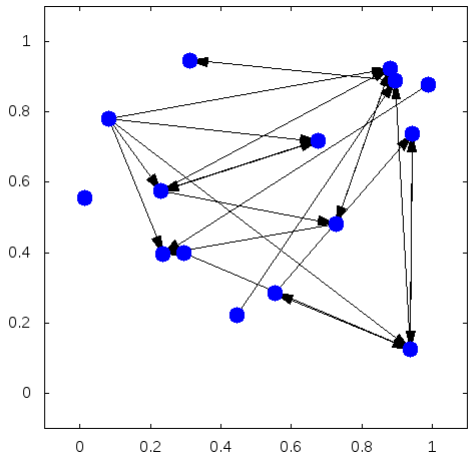


Figura 5: Ejemplo de grafo dirigido.

Referencias

[1] Dieter Jungnickel. *Graphs, Networks and Algorithms*. Springer, tercera edición, 2008.

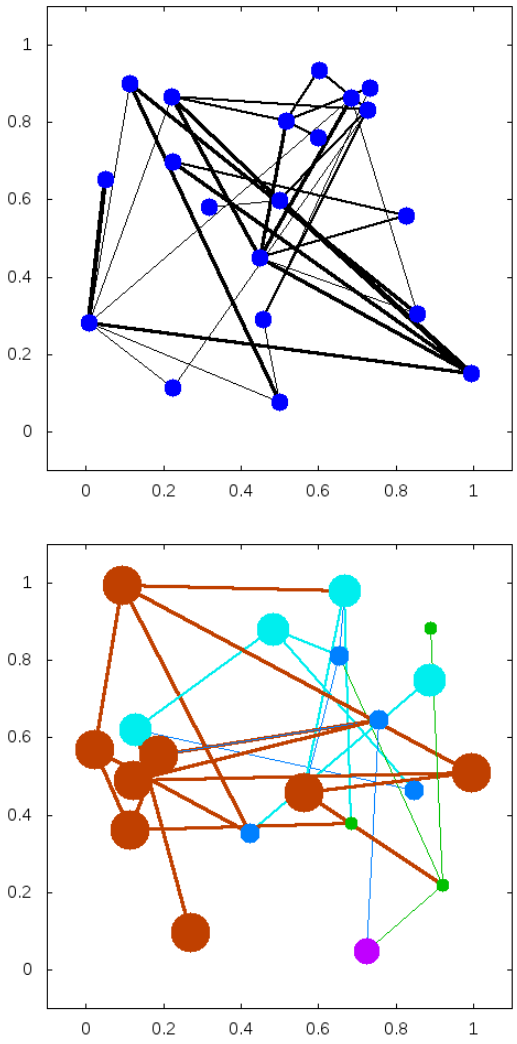


Figura 4: Ejemplos de grafos ponderados.