Optimización de flujo en redes

175
Tarea #2

26 de febrero de 2019

1. Acomodo bipartito

Este tipo de acomodo divide el grafo en dos conjuntos ajenos, luego une los dos conjuntos con las aristas correspondientes, muy util para observar relacion entre elementos de los conjuntos. En seguida el codigo utilizando dicho acomodo y la figura 1 muestra el grafo.

```
import networkx as nx
import matplotlib.pyplot as plt

G = nx.Graph()

G.add_nodes_from(['A', 'B', 'C'], bipartite = 0)
G.add_nodes_from(['D', 'E', 'F'], bipartite = 1)

G.add_edges_from([('A', 'D'), ('A', 'E')])
G.add_edges_from([('B', 'E')])
G.add_edges_from([('C', 'D'), ('C', 'F')])

nx.draw(G, pos = nx.bipartite_layout(G, ['A', 'B', 'C']), with_labels = True)

plt.show()
```

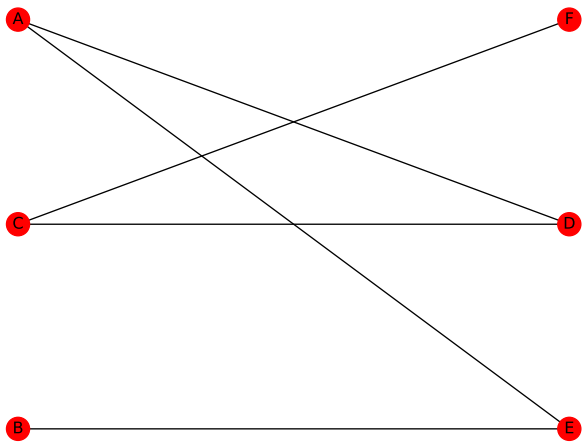

Figura 1: Grafo bipartito.

2. Acomodo circular

Los nodos son acomodados de tal forma para crear una circunferencia al unir los nodos. La figura 2 muestra un ejemplo del acomodo circular.

```
import networkx as nx
import matplotlib.pyplot as plt

G=nx.Graph()
G.add_nodes_from(['A', 'B', 'C', 'D', 'E', 'F', 'G'])

G.add_edges_from([('A', 'B'), ('B', 'C'), ('C', 'D'), ('D', 'E'), ('E', 'F'), ('
    F', 'G'), ('G', 'A')])
G.add_edges_from([('A', 'D'), ('A', 'G')])
G.add_edges_from([('E', 'C'), ('E', 'G')])


nx.draw(G, pos = nx.circular_layout(G), with_labels=True)

plt.show()
```

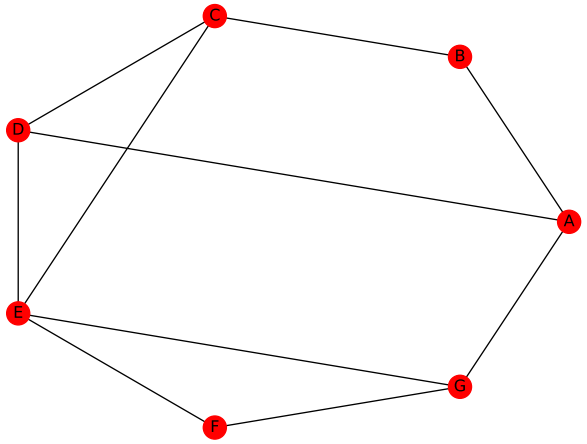

Figura 2: Ejemplo de acomodo circular de un grafo.

3. Acomodo kamada-kawai

Este acomodo hace que la distancia teórica del gráfico entre vértices en un gráfico está relacionada con la distancia geométrica entre ellos en el dibujo. El algoritmo tiene muchas buenas propiedades; dibujos simétricos, un número relativamente pequeño de cruces de bordes y dibujos casi congruentes de gráficos isomorfos. La figura 3 lo demuestra.

```
import networkx as nx
import matplotlib.pyplot as plt

G=nx.Graph()
G.add_nodes_from(['A', 'B', 'C', 'D', 'E', 'F', 'G'])

G.add_edges_from([('A', 'B'), ('B', 'C'), ('C', 'D'), ('D', 'E'), ('E', 'F'), ('
    F', 'G'), ('G', 'A')])
G.add_edges_from([('A', 'D'), ('A', 'G')])
G.add_edges_from([('E', 'C'), ('E', 'G')])

nx.draw(G, pos = nx.kamada_kawai_layout(G), with_labels=True)

plt.show()
```

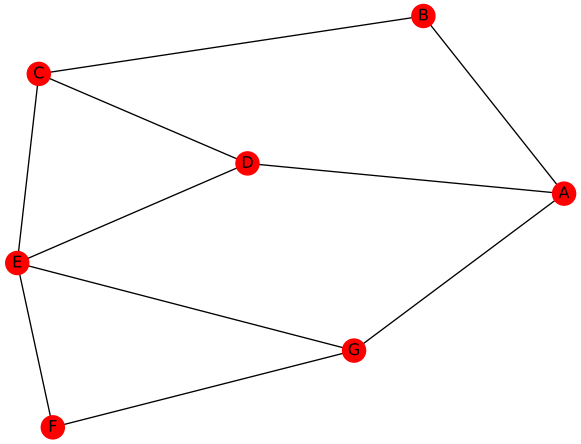

Figura 3: Ejemplo del acomodo kamada-kawai.

4. Acomodo aleatorio

El acomodo de los nodos es aleatoria, en base a una distribucion dada. Un ejemplo se muestra en la figura 4.

```
import networkx as nx
import matplotlib.pyplot as plt

G=nx.DiGraph()

G.add_node("A")
G.add_nodes_from(["B", "C", "D", "E"])

G.add_edges_from([("A","B"), ("A","C"), ("A","D"), ("A","E")])

nx.draw(G, pos = nx.random_layout(G), with_labels = True)

plt.show()
```

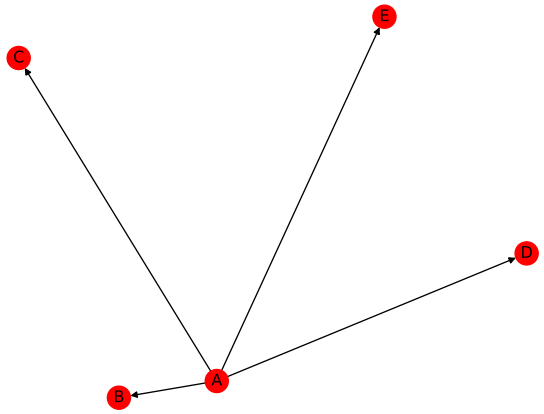

Figura 4: Ejemplo de acomodo aleatorio de un grafo.

5. Acomodo de cascara

Los nodos son acomodados de tal forma que se encuentren concentricos (como un cascaron). Se muestra un ejemplo enla figura 5.

```
import networkx as nx
import matplotlib.pyplot as plt

G = nx.DiGraph()

G.add_node("A")
G.add_nodes_from(["B", "C", "D", "E", "F"])
G.add_nodes_from(["G", "H", "I", "J", "K", "L"])

G.add_path(["B", "C", "D", "E", "F", "B"])
G.add_path(["G", "H", "I", "J", "K", "L", "G"])

G.add_edges_from([("H", "C"), ("J", "E"), ("D", "A"), ("F", "A")])

nx.draw(G, pos = nx.shell_layout(G), with_labels=True)

plt.show()
```

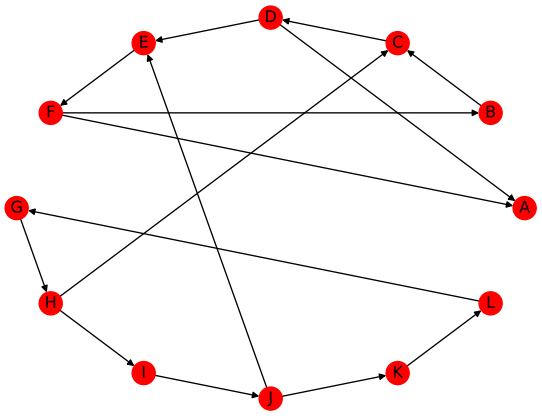

Figura 5: Grafo dibujado con el acomodo de cascara.

6. Acomodo espectral

El acomodo espectral coloca los nodos del gráfico en función de los vectores propios del gráfico Laplaciano. La figura 6 muestra un ejemplo.

```
import networkx as nx
import matplotlib.pyplot as plt

M = nx.MultiGraph()

M.add_nodes_from(['X','Y','Z'])
M.add_edges_from([('X','Y'),('X','Y'),('Y','Z')])

nx.draw_spectral(M, with_labels=True)
plt.show()
```

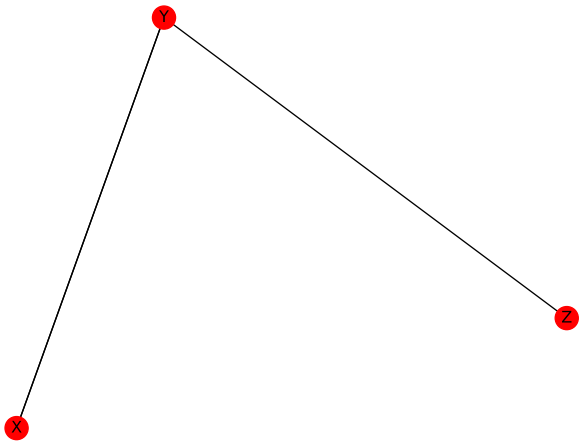

Figura 6: Ejemplo de acomodo espectral.

7. Acomodo de verano

Introduce los nodos utilizando el algoritmo dirigido por fuerza de Fruchterman-Reingold. En la figura 7 se observa el ejemplo.

```
import networkx as nx
import matplotlib.pyplot as plt

G = nx.DiGraph()

G.add_edges_from([("A", "B", {'myweight':1}), ("B", "C", {'myweight':2}),
                  ("A", "D", {'myweight':3}), ("B", "D", {'myweight':10})])

nx.draw_networkx(G, nx.spring_layout(G, weight = "myweight"))

plt.axis('off')
plt.show()
```

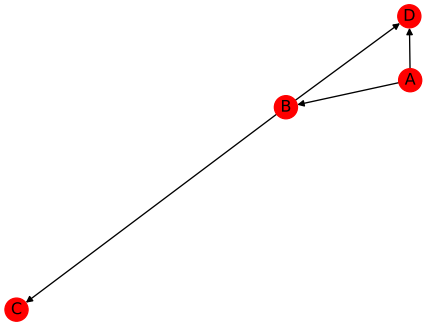

Figura 7: Ejemplo de acomodo de verano.

Referencias

[1] Schaeffer E. *Optimización de flujo en redes, 2019.*
`https://elisa.dyndns-web.com/teaching/opt/flow/`

[2] Tomihisa Kamada, Satoru Kawai. *An Algorithm for Drawing General Undirected Graphs*
`Information Processing Letters, 1988.`