Optimización de flujo en redes
Tarea 1: Representación de redes a través de la teoría de grafos

5271

25 de mayo de 2019

1. Grafo simple no dirigido acíclico

Se pueden encontrar varias aplicaciones a la modelación de un grafo simple no dirigido acíclico, una de las más claras son los árboles, "el árbol (árbol libre) que es un grafo no dirigido, conexo y acíclico. Un árbol también puede definirse como un grafo no dirigido en el que hay exactamente un camino entre todo par de vértices"[1].

Un ejemplo de usos de árboles es en topología de red la de árbol, en esta topología los nodos de la red están ubicados en forma de árbol. esta conexión es similar a muchas redes en estrella interconectadas con la diferencia de no poseer un nodo central, en cambio posee un nodo troncal desde el cual se ramifican el resto de los nodos como se muestra en la figura 1 que es una pequeña representación con solo ocho nodos de la red que posee la UEB Rolando Peréz Gollanes entidad dedicada al sacrificio y procesamiento de aves.

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

A=nx.Graph()
A.add_edges_from([('R1','Sw1'),('Sw1','Sw2'),
                ('Sw1','Sw3'),('Sw2','Pc2'),('Sw2','Pc3'),
                ('Sw3','Pc4'),('Sw3','Pc5')])

posicion = nx.spring_layout(A)
nx.draw_networkx_nodes(A, posicion, node_size=800)
nx.draw_networkx_edges(A, posicion, width=2)
nx.draw_networkx_labels(A, posicion, font_size=11, font_family='arial')

plot.axis('off')
plot.savefig("Graf1.eps")
plot.show(A)
```

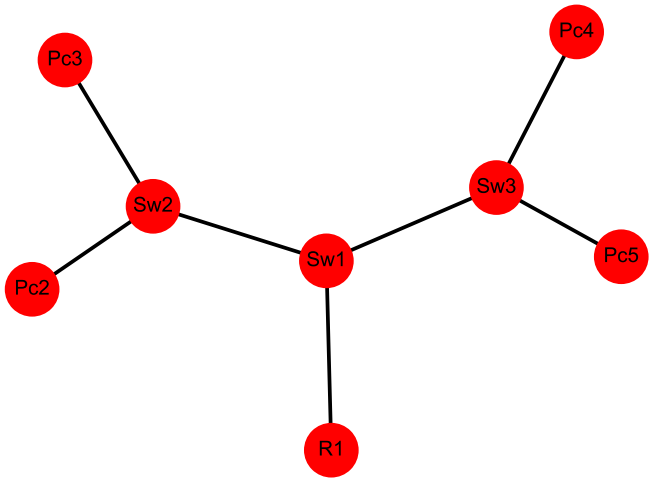

Figura 1: Grafo simple no dirigido acíclico

2. Grafo simple no dirigido cíclico

Un grafo simple no dirigido cíclico puede usarse áreas como geografía. Si se considera un mapa, digamos de Europa, que cada país sea un vértice y conecte dos vértices con una arista si esos países comparten una frontera. Un problema famoso que quedó sin resolver durante más de cien años fue el problema de los cuatro colores. Aproximadamente esto indica que cualquier mapa puede ser coloreado con a lo sumo cuatro colores de tal manera que los países adyacentes no tengan el mismo color. Este problema motivó muchos desarrollos en la teoría de grafos y finalmente se demostró con la ayuda de una computadora en 1976 [4].

Otra aplicación es en la representación de redes sociales, una red social se conceptualiza como un grafo, es decir, un conjunto de vértices (o nodos, unidades, puntos) que representan entidades u objetos sociales y un conjunto de líneas que representan una o más relaciones sociales entre ellos [2].

Por ejemplo, si consideramos un grupo de nueve doctores del núcleo académico de PISIS (Posgrado en Ingeniería de Sistemas) y construimos una red, tomando como vértices a los doctores y la colaboración de ellos en artículos publicados como aristas dará lugar a un grafo simple no dirigido cíclico como se muestra en la figura 2.

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


A=nx.Graph()
A.add_edges_from([('Dr.FL','Dra.AA'),('Dr.FL','Dra.YR'),
                  ('Dr.FL','Dr.RS'),('Dr.FL','Dra.ES'),
                  ('Dra.YR','Dr.VB'),('Dra.YR','Dr.RR'),
```

```
                ('Dra.YR','Dr.RS'),('Dra.AA','Dra.IM'),
                ('Dra.AA','Dr.RR'),('Dra.IM','Dr.VB'),
                ('Dra.IM','Dra.AS'),('Dra.IM','Dr.RS'),
                ('Dra.IM','Dr.VB'),('Dr.VB','Dr.RS'),
                ('Dra.AS','Dr.VB'),('Dra.AS','Dr.RR'),
                ('Dra.AS','Dr.RS'),('Dra.ES','Dr.RR')])


posicion=nx.spring_layout(A)
nx.draw_networkx_nodes(A,posicion, node_size=1700, node_color= 'grey')
nx.draw_networkx_edges(A,posicion, width=2,edge_color='b')
nx.draw_networkx_labels(A,posicion, font_size=11,font_family='arial',
                      font_color='y', font_weight='bold')

plot.axis('off')
plot.savefig("Graf2.eps")
plot.show(A)
```

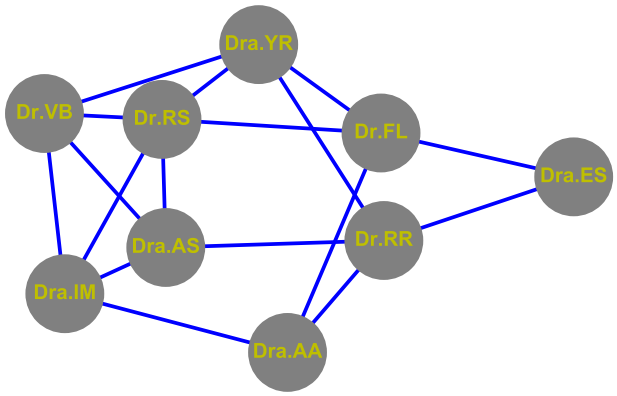

Figura 2: Grafo simple no dirigido cíclico

3. Grafo simple no dirigido reflexivo

Una de las aplicaciones de este tipo de grafo es en la modelación de comportamientos de elementos sociales en la vida real. Por ejemplo, en un estudio sobre el comportamiento sexual de un grupo de adolescentes con edades comprendida entre 15 y 18 años, se representar las relaciones sexuales consentidas(aristas) que existen entre los individuos (vértices) del grupo, así como la satisfacción, para así saber tendencias por edades, posibles esquemas de propagación de enfermedades y promiscuidad entre otros factores del interés de los sexólogos. Este ejemplo se muestra en el grafo de la figura 3.

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


A=nx.Graph()
A.add_edge('Luis(15)','Maria(15)')
A.add_edge('Luis(15)','Luis(15)')
A.add_edge('Maria(15)','Yanet(17)')
A.add_edge('Yanet(17)','Yanet(17)')
A.add_edge('Maria(15)','Ferndo(16)')
A.add_edge('Ferndo(16)','Desisy(18)')
A.add_edge('Desisy(18)','Pedro(17)')
A.add_edge('Pedro(17)','Pedro(17)')
A.add_edge('Pedro(17)','Julia(18)')
A.add_edge('Pedro(17)','Rosi(16)')
A.add_edge('Ferndo(16)','Claudia(16)')
nodes_reflex = {'Luis(15)','Ferndo(16)','Julia(18)'}
nodes_no_reflex = {'Maria(15)','Yanet(17)','Rosi(16)','Desisy(18)',
                   'Pedro(17)','Claudia(16)'}

posicion=nx.spring_layout(A)
nx.draw_networkx_nodes(A,posicion,nodelist=nodes_reflex,
                       node_size=1500, node_color= 'r')
nx.draw_networkx_nodes(A,posicion,nodelist=nodes_no_reflex,
                       node_size=1500, node_color= 'y')
nx.draw_networkx_edges(A,posicion, width=2,edge_color='b')
nx.draw_networkx_labels(A,posicion, font_size=11,font_family='arial',
                        font_color='Black', font_weight='bold')


plot.axis('off')
plot.savefig("Graf3.eps")
plot.show(A)
```

4. Grafo simple dirigido acíclico

Encontraos entre las aplicaciones de los grafos dirigidos o dígrafos acíclico las siguientes:

- Una red bayesiana queda especificada formalmente por una dupla $B = (G, O)$, donde G es un grafo dirigido acíclico (GDA) y O es el conjunto de distribuciones de probabilidad. Definimos un grafo como un par $G = (V, E)$, donde V es un conjunto finito de vértices nodos o variables y E es un subconjunto del producto cartesiano $V \times V$ de pares ordenados de nodos que llamamos enlaces o aristas [5].
- Los arboles dirigidos son un ejemplo clásico de grafos dirigidos acíclico, estos tienen múltiples en la cotidianidad como pueden ser los arbole genealógicos, los organigramas de una empresa (referido a las jerarquías entre los empleados) y el árbol de directorios de Windows.

En la figura 4 se muestra un ejemplo de red bayesiana (topología de red para el cáncer de pulmón),

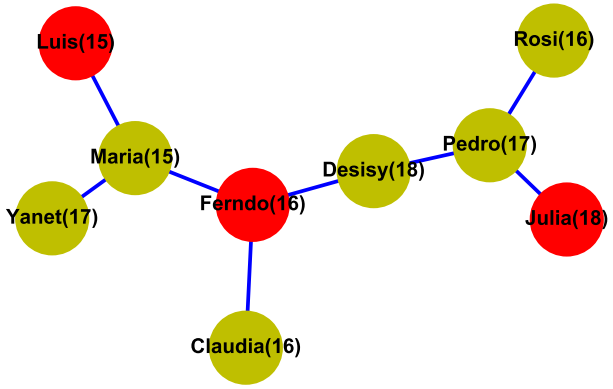

Figura 3: Grafo simple no dirigido reflexivo,donde los vértices rojos representan los nodos donde están las aristas reflexivas

donde C (cáncer), Con (Contaminación), F (Fumador), D (disnea), Rx (rayos-x) son los vértices y sus relaciones son las aristas.

```
import networkx as nx
import matplotlib.pyplot as plot
import numpy as nup
from time import time
A=nx.DiGraph()
A.add_edge('Con','C')
A.add_edge('F','C')
A.add_edge('C','D')
A.add_edge('C','R x')

tiempo=[]

for i in range(30):
    tiempo_inicial = time()
    for j in range(1600):
      d= list(nx.topological_sort(A))
    tiempo_final = time()
    tiempo_ejecucion = tiempo_final - tiempo_inicial
    tiempo.append(tiempo_ejecucion)
media=nup.mean(tiempo)
desv=nup.std(tiempo)
print (tiempo, d, media,desv)#En segundos
posicion=nx.spring_layout(A)
nx.draw_networkx_nodes(A,posicion, node_size=500, node_color= 'grey',alpha=0.9)

nx.draw_networkx_edges(A,posicion, width=2,edge_color='black')
nx.draw_networkx_labels(A,posicion, font_size=11,font_family='arial',
                        font_color='y', font_weight='bold')
```

```
plot.axis('off')
plot.savefig("Graf4.eps")
plot.show(A)
```

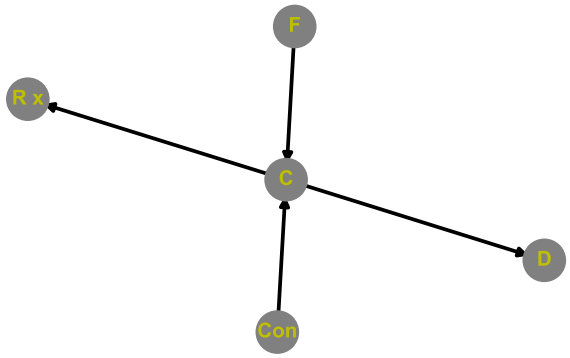

Figura 4: Grafo simple dirigido acíclico

5. Grafo simple dirigido cíclico

Aplicaciones de los grafos dirigidos cíclicos:

- En ingeniería eléctrica se utilizan grafos dirigidos cíclicos en el análisis de circuito desde Kirchoff en los años 1850.
- En redes sociales, otro enfoque de redes sociales en su análisis puede arrojar un grafo dirigido cíclico si te tomamos como vértices a personas y como aristas el sentimiento de amistad de una persona hacia otra, este ejemplo se refleja en el grafo de la figura 5.

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


A=nx.DiGraph()
A.add_edge('Ana','Felix')
A.add_edge('Ana','Alberto')
A.add_edge('Alberto','Yanet')
A.add_edge('Alberto','Fernando')
A.add_edge('Yanet','Roger')
A.add_edge('Teresa','Roger')
A.add_edge('Felix','Marta')
```

```
A.add_edge('Marta','Fernando')
A.add_edge('Yanet','Ana')
A.add_edge('Marta','Ana')



posicion=nx.spring_layout(A)

nx.draw_networkx_nodes(A,posicion,
                    node_size=500, node_color= 'y')
nx.draw_networkx_edges(A,posicion, width=2,edge_color='b')
nx.draw_networkx_labels(A,posicion, font_size=11,font_family='arial',
                     font_color='Black', font_weight='bold')


plot.axis('off')
plot.savefig("Graf5.eps")
plot.show(A)
```

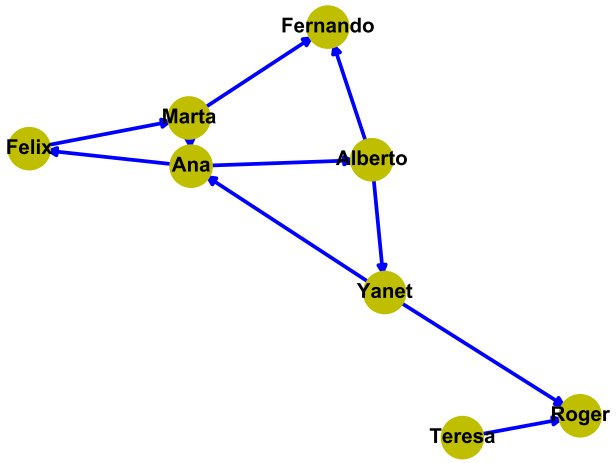

Figura 5: Grafo simple dirigido cíclico

6. Grafo simple dirigido reflexivo

Las aplicaciones de estos grafos van desde la representación del funcionamiento de las páginas web, el modelado de una empresa de servicio que brinda el mismo tanto a sus clientes como a ella misma, hasta representar un grafo que modele la comprobación de una red de computadoras. Por ejemplo, suponiendo que se está comprobando la conectividad entre los nodos (equipos) de una red de computadoras, es

decir que un nodo puede dar ping a cualquier otro nodo, significa que dicho nodo está conectado con el resto, además se debe asegurar que el mismo nodo pueda recibir un auto-ping, si esto no ocurre existe un problema de conectividad. La figura 6 muestra dicho ejemplo.

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


A=nx.DiGraph()
A.add_edge('Pc1','Pc1')
A.add_edge('Pc1','Pc2')
A.add_edge('Pc1','Pc3')
A.add_edge('Pc1','Pc4')
A.add_edge('Pc1','Pc5')
A.add_edge('Pc1','Pc6')
A.add_edge('Pc1','Pc7')
nodes_reflex = {'Pc1'}
nodes_no_reflex = {'Pc2','Pc3','Pc4','Pc5','Pc6','Pc6','Pc7'}
posicion=nx.spring_layout(A)
nx.draw_networkx_nodes(A,posicion,nodelist=nodes_reflex,
                       node_size=500, node_color= 'r')
nx.draw_networkx_nodes(A,posicion,nodelist=nodes_no_reflex,
                       node_size=500, node_color= 'y')
nx.draw_networkx_edges(A,posicion, width=2)
nx.draw_networkx_labels(A,posicion, font_size=11,font_family='arial')


plot.axis('off')
plot.savefig("Graf6.eps")
plot.show(A)
```

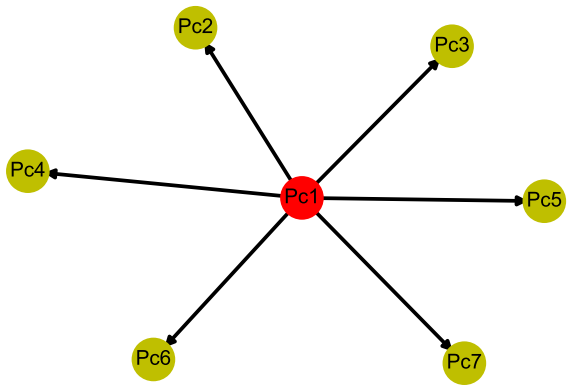

Figura 6: Grafo simple dirigido reflexivo, donde el vértice rojo representa la arista reflexiva

7. Multigrafo no dirigido acíclico

Una de las aplicaciones de este tipo de grafo es en el tasado de rutas. Por ejemplo, considerando que se quiere tranzar los diferentes caminos (sin importar el sentido de estos) que comunican a varios Municipios de La Habana en un orden específico (Lisa, Marianao, Playa, Vedado, Habana Vieja, Habana del Este). Tomando como vértices los municipios y como aristas las carreteras que los unen, esto da lugar al grafo que muestra la figura 7.

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


A=nx.MultiGraph()
A.add_edge('Lisa','Marianao', weight=2)
A.add_edge('Marianao','Playa', weight=2)
A.add_edge('Marianao','Playa', weight=4)
A.add_edge('Playa','Vedado', weight=2)
A.add_edge('Vedado','Habana_Vieja', weight=2)
A.add_edge('Vedado','Habana_Vieja', weight=4)
A.add_edge('Vedado','Habana_Vieja', weight=5)
A.add_edge('Habana_Vieja','Habana_del_Este', weight=2)
A.add_edge('Habana_Vieja','Habana_del_Este', weight=4)

black=[('Lisa','Marianao' ),('Playa','Vedado'),('Vedado','Habana_Vieja'),
   ('Habana_Vieja','Habana_del_Este'),('Marianao','Playa')]
red=[('Marianao','Playa'),('Vedado','Habana_Vieja'),
   ('Habana_Vieja','Habana_del_Este')]
bl=[('Vedado','Habana_Vieja')]
posicion=nx.spring_layout(A)

nx.draw_networkx_nodes(A,posicion,
                    node_size=500, node_color= 'y')
nx.draw_networkx_edges(A, posicion, edgelist=bl, width=10, alpha=0.5,
edge_color='b')
nx.draw_networkx_edges(A, posicion, edgelist=red, width=5, alpha=0.5,
edge_color='r')
nx.draw_networkx_edges(A, posicion, edgelist=black, width=2,
edge_color='Black')
nx.draw_networkx_labels(A,posicion, font_size=11,font_family='arial',
                     font_color='Black', font_weight='bold')


plot.axis('off')
plot.savefig("Graf7.eps")
plot.show(A)
```

8. Multigrafo no dirigido cíclico

Un ejemplo donde podemos utilizar este tipo de grafos es: se quiere representar cuantas llamadas (aristas)se realizaron entre un grupo de personas (vértices), donde nos importa saber la duración de

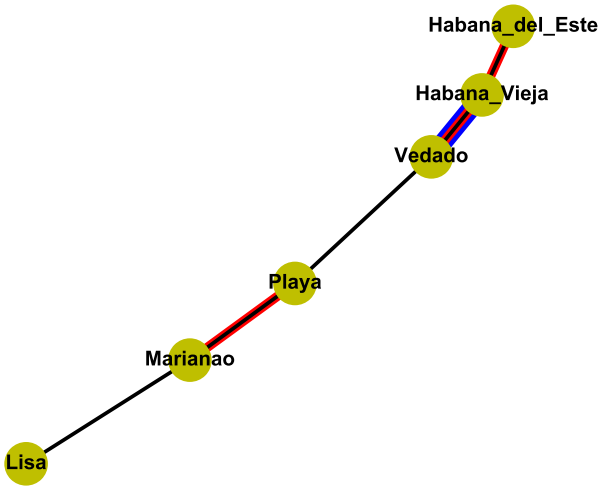

Figura 7: Multigrafo no dirigido acíclico, donde los diferente colores de los arcos representan los diferentes caminos

cada llamada durante un día determinado. Este grafo se ve representado en la figura 8.

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


A=nx.MultiGraph()
A.add_edge('Tel_1','Tel_2', weight=2)
A.add_edge('Tel_1','Tel_2', weight=3)
A.add_edge('Tel_1','Tel_2', weight=4)
A.add_edge('Tel_1','Tel_3', weight=2)
A.add_edge('Tel_1','Tel_4', weight=4)
A.add_edge('Tel_2','Tel_4', weight=2)
A.add_edge('Tel_2','Tel_4', weight=3)
A.add_edge('Tel_2','Tel_3', weight=2)
A.add_edge('Tel_3','Tel_5', weight=2)
A.add_edge('Tel_3','Tel_5', weight=3)
A.add_edge('Tel_3','Tel_5', weight=4)
A.add_edge('Tel_5','Tel_6', weight=2)
A.add_edge('Tel_5','Tel_6', weight=3)
A.add_edge('Tel_1','Tel_6', weight=2)
A.add_edge('Tel_1','Tel_6', weight=3)
black=[('Tel_1','Tel_2' ),('Tel_1','Tel_3'),('Tel_2','Tel_4'),
   ('Tel_2','Tel_3'),('Tel_3','Tel_5'),('Tel_3','Tel_5'),
   ('Tel_5','Tel_6'),('Tel_1','Tel_6')]

red=[('Tel_1','Tel_2'),('Tel_2','Tel_4'),
   ('Tel_1','Tel_2'),('Tel_1','Tel_6'),('Tel_5','Tel_6')]
```

```
bl=[('Tel_1','Tel_2'),('Tel_3','Tel_5')]

posicion=nx.spring_layout(A)

nx.draw_networkx_nodes(A,posicion,
                    node_size=1000, node_color= 'y')
nx.draw_networkx_edges(A, posicion, edgelist=bl, width=10, alpha=0.5,
edge_color='b')
nx.draw_networkx_edges(A, posicion, edgelist=red, width=5, alpha=0.5,
edge_color='r')
nx.draw_networkx_edges(A, posicion, edgelist=black, width=2,
edge_color='Black')
nx.draw_networkx_labels(A,posicion, font_size=11,font_family='arial',
                     font_color='Black', font_weight='bold')


plot.axis('off')
plot.savefig("Graf8.eps")
plot.show(A)
```

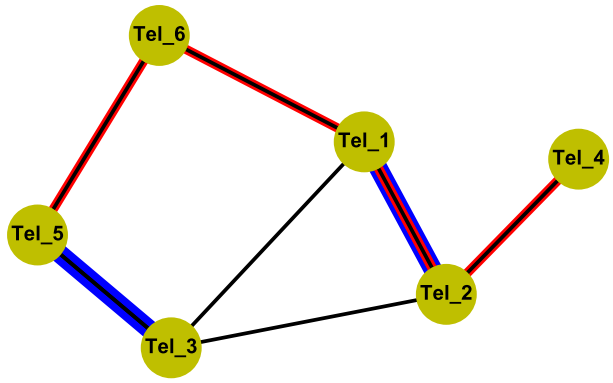

Figura 8: Multigrafo no dirigido cíclico, donde las aristas de diferentes colores representan la existencia de las llamadas con distinta duración

9. Multigrafo no dirigido reflexivo

Partiendo del ejemplo de la sección 3 se pretende construir un grafo más informativo a la hora de su análisis, por lo que se le agrega como arista (el hecho que las relaciones sexuales mantenidas fueran dentro de una relación formal). Este grafo se muestra en la figura 9.

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

```
A=nx.MultiDiGraph()


A.add_edge('Luis(15)','Luis(15)' ,weight=2)
A.add_edge('Maria(15)','Yanet(17)' ,weight=2)
A.add_edge('Yanet(17)','Yanet(17)' ,weight=2)


A.add_edge('Maria(15)','Fernado(16)' ,weight=2)
A.add_edge('Fernado(16)','Desisy(18)',weight=3)
A.add_edge('Fernado(16)','Desisy(18)',weight=2)
A.add_edge('Pedro(17)','Pedro(17)' ,weight=2)
A.add_edge('Pedro(17)','Julia(18)' ,weight=2)
A.add_edge('Pedro(17)','Rosi(16)' ,weight=3)
A.add_edge('Pedro(17)','Rosi(16)' ,weight=2)
A.add_edge('Fernado(16)','Claudia(16)',weight=2)

nodes_reflex = {'Luis(15)','Fernado(16)','Julia(18)'}
nodes_no_reflex = {'Maria(15)','Yanet(17)','Rosi(16)','Desisy(18)',
                   'Pedro(17)','Claudia(16)'}

black=[('Maria(15)','Yanet(17)'),
  ('Maria(15)','Fernado(16)'),('Fernado(16)','Desisy(18)'),
  ('Yanet(17)','Julia(18)'),('Desisy(18)','Pedro(17)'),
  ('Pedro(17)','Julia(18)'), ('Pedro(17)','Rosi(16)'),
  ('Fernado(16)','Claudia(16)')]

bl=[('Fernado(16)','Desisy(18)'),('Pedro(17)','Rosi(16)')]

posicion=nx.spring_layout(A)
nx.draw_networkx_nodes(A,posicion,nodelist=nodes_reflex,
                       node_size=1800, node_color= 'r')
nx.draw_networkx_nodes(A,posicion,nodelist=nodes_no_reflex,
                       node_size=1500, node_color= 'y')
nx.draw_networkx_edges(A, posicion, edgelist=bl, width=5, alpha=0.5,
edge_color='b')
nx.draw_networkx_edges(A, posicion, edgelist=black, width=2,
edge_color='Black')
nx.draw_networkx_labels(A,posicion, font_size=11,font_family='arial',
                        font_color='Black', font_weight='bold')


plot.axis('off')
plot.savefig("Graf9.eps")
plot.show(A)
```

10. Multigrafo dirigido acíclico

Considerando que se quiere trazar las diferentes rutas (teniendo en cuenta el sentido de los mimas) que llevan de una provincia a otra en Cuba (Pinar del Río, Artemisa, La Habana, Mayabeque, Matanza,

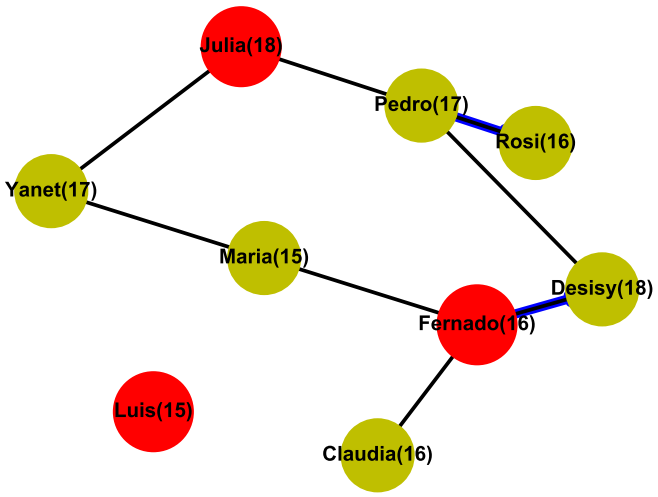

Figura 9: Multigrafo no dirigido reflexivo, donde las aristas de diferentes colores representan la existencia de una relación formal entre los individuos y los vértices de color rojo representa los vértices con aristas reflexivas

Cienfuegos, Villa Clara). Tomando como vértices las provincias y como aristas las carreteras que las unen, como muestra la figura 10.

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


A=nx.MultiDiGraph()
A.add_edge('PR','A',  weight=2)
A.add_edge('A','LH',  weight=2)
A.add_edge('A','LH',  weight=4)
A.add_edge('LH','May', weight=2)
A.add_edge('May','Mat', weight=2)
A.add_edge('May','Mat', weight=4)
A.add_edge('LH','May', weight=4)
A.add_edge('LH','May', weight=5)
A.add_edge('May','Mat', weight=5)
A.add_edge('Mat','C', weight=2)
A.add_edge('Mat','C', weight=4)

black=[('PR','A' ),('LH','May'),('May','Mat'),
   ('Mat','C'),('A','LH')]
red=[('A','LH'),('May','Mat'),
   ('Mat','C'),('LH','May')]
bl=[('May','Mat'),('LH','May')]
posicion=nx.spring_layout(A)
```

```
nx.draw_networkx_nodes(A,posicion,
                       node_size=500, node_color= 'y')
nx.draw_networkx_edges(A, posicion, edgelist=bl, width=10, alpha=0.5,
edge_color='b')
nx.draw_networkx_edges(A, posicion, edgelist=red, width=5, alpha=0.5,
edge_color='r')
nx.draw_networkx_edges(A, posicion, edgelist=black, width=2,
edge_color='Black')
nx.draw_networkx_labels(A,posicion, font_size=11,font_family='arial',
                        font_color='Black', font_weight='bold')


plot.axis('off')
plot.savefig("Graf10.eps")
plot.show(A)
```

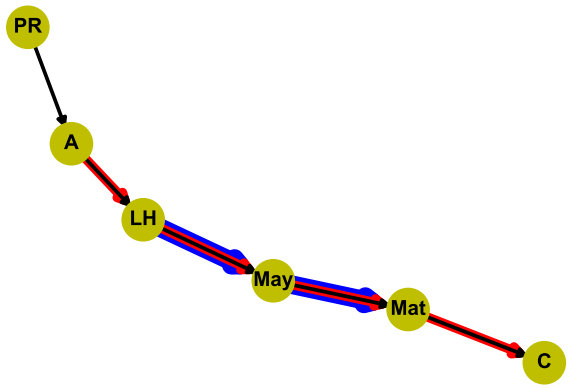

Figura 10: Multigrafo dirigido acíclico, donde las aristas de diferentes colores representan las existencia de más de una ruta entre las diferente provincias

11. Mltigrafo dirigido cíclico

"En el caso particular de que las redes reflejen una realidad social, los nodos pueden representar personas o entidades relacionadas con sus contextos, y las conexiones representarán relaciones sociales existentes entre ellos (amistad, parentesco, membresía, afinidad, etc.). A pesar de que intuitivamente las redes sociales se asemejan a los grafos matemáticos, es más habitual que en ellas se trabaje con distintos tipos de relaciones"[3] por lo que es necesario la utilización de multígrafos, que es la herramienta que contempla más de una relación entre dos nodos, con esto ganamos mayor riqueza en los datos a analizar. Por ejemplo, tenemos un grupo de personas que laboran en un departamento de Informática y se hace una encuesta donde se les pide que marque cuales de tres sentimientos (respeto, afinidad, rechazo) sienten por sus compañeros de trabajo, como muestra la figura 11.

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


A=nx.MultiDiGraph()
A.add_edge('Persona1','Persona2', weight=2)
A.add_edge('Persona1','Persona2', weight=3)
A.add_edge('Persona1','Persona2', weight=4)
A.add_edge('Persona1','Persona3', weight=2)
A.add_edge('Persona1','Persona4', weight=4)
A.add_edge('Persona2','Persona4', weight=2)
A.add_edge('Persona2','Persona4', weight=3)
A.add_edge('Persona4','Persona3', weight=3)
A.add_edge('Persona3','Persona2', weight=2)
A.add_edge('Persona3','Persona5', weight=2)
A.add_edge('Persona3','Persona5', weight=3)
A.add_edge('Persona3','Persona5', weight=4)
A.add_edge('Persona5','Persona6', weight=2)
A.add_edge('Persona5','Persona6', weight=3)
A.add_edge('Persona1','Persona6', weight=2)
A.add_edge('Persona1','Persona6', weight=3)
black=[('Persona1','Persona2' ),('Persona1','Persona3'),('Persona2','Persona4'),
   ('Persona3','Persona2'),('Persona3','Persona5'),('Persona3','Persona5'),
   ('Persona5','Persona6'),('Persona1','Persona6'),('Persona4','Persona3')]

red=[('Persona1','Persona2'),('Persona2','Persona4'),
   ('Persona1','Persona2'),('Persona1','Persona6'),('Persona5','Persona6')]
bl=[('Persona1','Persona2'),('Persona3','Persona5')]

posicion=nx.spring_layout(A)

nx.draw_networkx_nodes(A,posicion,
                    node_size=500
                    , node_color= 'y')
nx.draw_networkx_edges(A, posicion, edgelist=bl, width=10, alpha=0.5,
edge_color='b')
nx.draw_networkx_edges(A, posicion, edgelist=red, width=5, alpha=0.5,
edge_color='r')
nx.draw_networkx_edges(A, posicion, edgelist=black, width=2,
edge_color='Black')
nx.draw_networkx_labels(A,posicion, font_size=11,font_family='arial',
                     font_color='Black', font_weight='bold')


plot.axis('off')
plot.savefig("Graf11.eps")
plot.show(A)
```

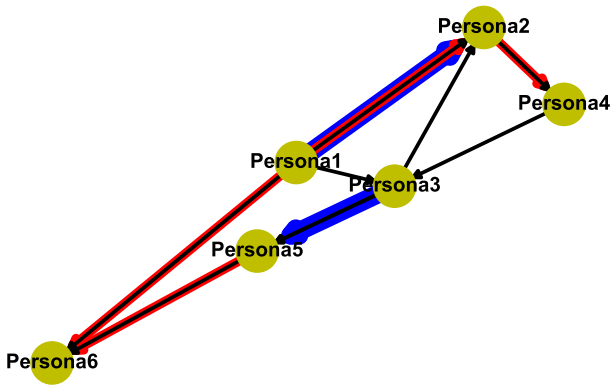

Figura 11: Multigrafo dirigido cíclico, donde las aristas de diferentes colores representan las existencia de más de una opinión sobre sus compañeros

12. Multigrafo dirigido reflexivo

Una de las aplicaciones de estos grafos es en el modelado del funcionamiento de una máquina de estados. Por ejemplo, se desea modelar un grafo que represente el funcionamiento de una máquina muy sencilla de golosinas, que responde a las siguientes reglas:

1 Cada golosina vale $ 0.25.

2 La máquina acepta sólo monedas de $ 0.10 y de $ 0.05.

3 La máquina NO da vuelto.

Para modelar el grafo, debemos considerar distintos "estados de dinero ingresado", y como se va pasando de uno a otro. Por ejemplo, si en un momento tenemos ingresados 5 centavos, y agregamos 5 centavos más, pasamos a otro estado, que es el mismo que si hubiésemos ingresado 10 centavos al principio. Llamemos A, B, C, D, E y F a los estados que representan 0, 5, 10, 15, 20, 25 centavos ingresados respectivamente. Las transiciones de un estado a otro se harán por ingreso de 5 o 10 centavos, o al presionar el botón para obtener los caramelos (G). Como se muestra en la figura 12.

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


A=nx.MultiDiGraph()
A.add_edge('A','A',  weight=2)
A.add_edge('A','B',  weight=2)
A.add_edge('A','C',  weight=2)
A.add_edge('B','B',  weight=2)
A.add_edge('B','C',  weight=2)
```

```
A.add_edge('B','D',  weight=2)
A.add_edge('C','C',  weight=2)
A.add_edge('C','D',  weight=2)
A.add_edge('C','E',  weight=2)
A.add_edge('D','D',  weight=2)
A.add_edge('D','E',  weight=2)
A.add_edge('D','F',  weight=2)
A.add_edge('E','E',  weight=2)
A.add_edge('E','F',  weight=2)
A.add_edge('E','F',  weight=4)
A.add_edge('F','F',  weight=2)
A.add_edge('F','A',  weight=2)


black=[('A','A' ),('A','B'),('A','C'),
   ('B','B'),('B','C'),('B','D'),('C','C'),('C','D'),('C','E'),
   ('D','D'),('D','E'),('D','F'),('E','E'),('E','F'),('E','F'),('F','A')]

bl=[('E','F')]
posicion=nx.spring_layout(A)

nx.draw_networkx_nodes(A,posicion,
                    node_size=500, node_color= 'R')
nx.draw_networkx_edges(A, posicion, edgelist=bl, width=10, alpha=0.5,
edge_color='b')

nx.draw_networkx_edges(A, posicion, edgelist=black, width=2,
edge_color='Black')
nx.draw_networkx_labels(A,posicion, font_size=11,font_family='arial',
                     font_color='Black', font_weight='bold')


plot.axis('off')
plot.savefig("Graf12.eps")
plot.show(A)
```

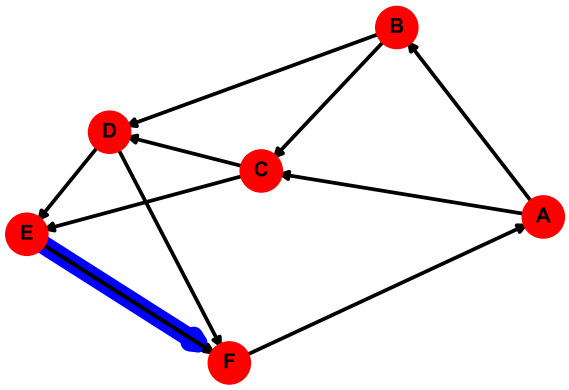

Figura 12: Multigrafo dirigido reflexivo, donde las aristas de diferentes colores representan las existencia de mas de una forma de pasar de un estado a otro

Referencias

[1] Amalia Duch Brown. Grafos. `https://www.cs.upc.edu/~duch/home/duch/grafos.pdf`, Octubre 2007.

[2] Anwesha Chakraborty, Trina Dutta, Sushmita Mondal, and Asoke Nath. Application of graph theory in social media. *International journal of computer sciences and engineering*, 6:722–729, 10 2018.

[3] R. A. Hanneman and M. Riddle. Introduction to social network methods. `http://faculty.ucr.edu/~hanneman/.Consultado`, September 2013.

[4] Kimball Martin. *Graph Theory and Social Networks*. Spring, Abril 2014.

[5] Azahara Muñoz Martínez Mauricio Beltrán Pascuala and Ángel Muñoz Alamillos. Redes bayesianas aplicadas a problemas de credit scoring. una aplicación práctica. `https://www.sciencedirect.com/science/article/pii/S0210026613000083`, Octubre 2013.