Problema de evacuación de una ciudad

Brenda Yaneth Sotelo Benítez
5705

3 de junio de 2019

Introducción

Problemas de optimización pueden ser resueltos utilizando el algoritmo de flujo máximo, tales como el problema de corte mínimo o el problema de acoplamiento máximo. En este trabajo se plantea el problema de evacuación de personas o evacuación de una ciudad para el que se muestra la red representativa del problema y la red modificada de tal forma que se resuelva como un problema de flujo máximo.

Los instancias se obtuvieron por medio de la librería Networkx [1] y Matplotlib [2] de `Python` [3] para la generación de grafos y guardar imágenes en el formato *eps* respectivamente. Además cada una de las instancias fueron visualizadas con el algoritmo de acomodo `bipartite_layout` que produce un resultado entendible acorde a la aplicación. Las caracteristicas de los algoritmos mencionados se pueden encontrar en la práctica dos y cuatro del repositorio de Sotelo [4]. El código empleado se obtuvo consultando documentación oficial [5].

Se presenta una breve descripción del problema seleccionado, visualización de las redes, fragmentos relevantes de código y una sección de resultados.

Las pruebas se han realizado en una PC con procesador Intel Core i3 2.00 GHz y 8.00 GB de RAM.

Problema de evacuación de personas

Un plan de evacuación se define como la planificación y organización para utilizar de forma óptima los medios que se disponen con la finalidad de reducir

las posibles consecuencias que pudieran derivarse de una situación de riesgo como desbordamientos de ríos, deslaves de montañas, sismos, terremotos, maremotos, tornados, huracanes, incendios forestales, tsunamis, ciclones, erupción de un volcán, inundaciones, guerras, epidemias, ataques terroristas, incendios, explosiones, etc.

En la figura 1 se muestra la red representativa del problema donde se tienen diez personas, tres vehículos y tres calles. Se desea saber cuál es la mejor asignación de persona–vehículo y vehículo–calle. Cada persona y cada vehículo puede ir a cualquier vehículo y calle respectivamente, solamente una vez.

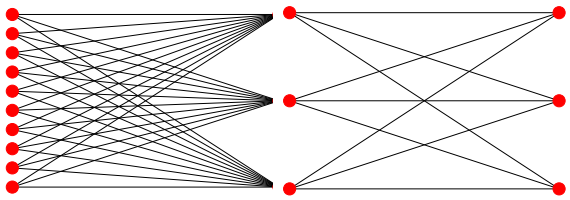


Figura 1: *Red representativa del problema*

Datos de entrada

Los parámetros utilizados para la resolución del problema son los siguientes:

- **Número de personas.** Como su nombre lo indica, este parámetro representa la cantidad de habitantes que hay en la zona afectada que serán evacuados en una cierta cantidad de vehículos que se tiene disponible para cuado ocurre alguna situación de emergencia.
- **Número de vehículos.** Ante una situación de este tipo se cuenta con una cantidad finita de vehículos.
- **Número de calles.** Durante el traslado de las personas existen diferentes rutas que se pueden tomar, de tal forma que se permita tener alternativas ya sea más rápidas o ante situaciones de congestionamiento.
- **Capacidad del vehículo.** Representa la cantidad de personas que pueden trasladarse por vehículo.
- **Capacidad de las calles.** Las calles, carreteras o avenidas pueden ir en un sentido o dos, lo que se busca es que el traslado sea lo más rápido posible por lo que no se quisiera que todos los vehículos siguieran una misma ruta uno tras otro, es por eso que se establece una cantidad de vehículos

por calle para que se tomen rutas alternas y evitar el agrupamiento de vehículos por calle.

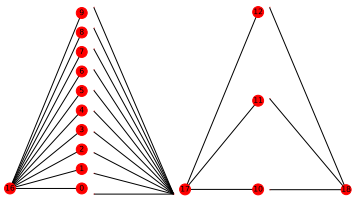
En código 1 se muestra el fragmento de código de los parámetros de la instancia.

Código 1: *Parámetros de la instancia*

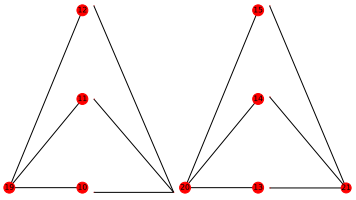
```
#Cardinalidades
numpersonas = 1000
numvehiculos =80
numcalles = 20

#Capacidades
personasxvehiculo=3
vehiculosxcalle=4
```

En la figura 2 se muestra la red utilizada para resolver el algoritmo de flujo máximo, la cual ha sido dividida en dos secciones. La primer sección consta de asignar cada persona a algún vehículo y la segunda consiste en realizar la asignación de cada vehículo a alguna calle, además se consideran las capacidades de personas por vehículo así como vehículo por calle para cumplir con las restricciones deseadas.



(a) Asignación persona vehículo



(b) Asignación vehículo calle

Figura 2: *Red solución*

En el cuadro 1 se tiene una instancia del problema en la que son 1000 personas que necesitan ser evacuadas, aunque es claro que en las ciudades puede haber más habitantes. Se tienen también 250 vehículos y 20 calles, cada una con capacidad 3 y 4 respectivamente.

Cuadro 1: *Datos de entrada de la instancia*

Parámetros	**Unidades**	**Capacidad**
Personas	1000	–
Vehículos	80	3
Calles	20	4

Datos de salida

Como solución se retornan dos valores, el primero de ellos representa la cantidad de personas que son asignadas y evacuadas dependiendo de la instancia y el segundo es la cantidad de vehículos que serán utilizados.

Código 2: *Crear conjuntos de nodos y solución*

```
A1 = [ (u,v,{'cap':1})  if u>v else (v,u,{'cap':1}) for (u,v) in G1
    .edges] #Con menor porque el S es el siguiente
A2 = [ (u,v,{'cap':1})  if u<v else (v,u,{'cap':1}) for (u,v) in G2
    .edges] #Con mayor porque
A3 = [ (u,v,{'cap':personasxvehiculo})  if u<v else (v,u,{'cap':
    personasxvehiculo}) for (u,v) in G3.edges]
A4 = [ (u,v,{'cap':personasxvehiculo})  if u<v else (v,u,{'cap':
    personasxvehiculo}) for (u,v) in G4.edges] #Con mayor porque

A5 = [ (u,v,{'cap':1})  if u>v else (v,u,{'cap':1}) for (u,v) in G5
    .edges] #Con menor porque el S es el siguiente
A6 = [ (u,v,{'cap':1})  if u<v else (v,u,{'cap':1}) for (u,v) in G6
    .edges] #Con mayor porque
A7 = [ (u,v,{'cap':vehiculosxcalle})  if u<v else (v,u,{'cap':
    vehiculosxcalle}) for (u,v) in G7.edges]
A8 = [ (u,v,{'cap':vehiculosxcalle})  if u<v else (v,u,{'cap':
    vehiculosxcalle}) for (u,v) in G8.edges] #Con mayor porque

Gfinal1 = nx.Graph()
Gfinal1.add_edges_from(A1+A2+A3+A4)

Gfinal2 = nx.Graph()
Gfinal2.add_edges_from(A5+A6+A7+A8)

valor1,flujo_arcos1 = nx.maximum_flow(Gfinal1,aux1,aux3,'cap')
valor2,flujo_arcos2 = nx.maximum_flow(Gfinal2,aux4,aux6,'cap')
```

En el cuadro 2 se muestra la solución de la instancia mencionada en el cuadro

1 en la que se tienen 20 calles disponibles y cada una tiene una capacidad de 4 vehículos y donde a lo más transitarían 80 vehículos. Luego si se tienen 80 vehículos y cada uno tiene una capacidad de 3 personas se lograrían transportar como máximo 240 personas.

Cuadro 2: *Datos de salida de la instancia*

Parámetros	**Unidades**	**Capacidad**	**Solución**
Personas	1000	–	240
Vehículos	80	3	80
Calles	20	4	20

Referencias

[1] NetworkX developers Versión 2.0. `https://networkx.github.io/`.

[2] The Matplotlib development team Versión 3.0.2. `https://matplotlib.org/`.

[3] Python Software Foundation Versión 3.7.2. `https://www.python.org/`.

[4] Sotelo B. Repositorio optimización flujo en redes. `https://github.com/BrendaSotelo/Flujo_Redes_BSotelo`.

[5] NetworkX developers con última actualización el 19 de Septiembre 2018. `https://networkx.github.io/documentation/networkx-1.10/reference/drawing.html`.