Problema de flujo máximo con costo mínimo.

Se nos presenta con el siguiente problema: Dado un grupo de ciudades, que incluye una ciudad de origen O y una ciudad destino D, y una serie de caminos que conectan dichas ciudades que tienen de atributo una capacidad máxima de personas que pueden circular y además un costo por persona, encontrar cual es la máxima cantidad de personas que pueden ir desde O hasta D al menor costo posible. Podemos asumir que la capacidad de personas es siempre positiva y entera, al igual que los costos por persona.

Este problema, se puede representar mediante una red de flujo donde la ciudad de origen es la fuente, el destino es el sumidero, las ciudades serían los vértices, los caminos las aristas y las personas el flujo. A diferencia de las redes de flujo con las que veníamos trabajando, las aristas en este caso van a tener dos atributos. (Quizás deberíamos explicar qué es un grafo residual) La capacidad máxima de flujo que puede pasar por la arista, que representa a la capacidad máxima de personas que recorren el camino y un costo por unidad de flujo, en este modelo, el costo por persona que utiliza el camino. Donde el costo de un flujo cualquiera resulta la sumatoria del producto entre el flujo resultante en cada arista y el costo por unidad de de cada arista . Sea f el flujo sobre una red, E el conjunto de las aristas y $c(x)$ una función costo, tenemos que el costo del flujo es:

$$c(f) = \sum_{e \in E} f(e).c(e) \qquad (1)$$

El hecho de que tengamos un costo en las aristas va a impactar en el grafo residual de la forma que el par del grafo residual de una arista en la red de flujo va a tener el mismo costo pero negativo[1]. Es decir, notando a una arista e como los vértices u, v que conecta, en orden de la dirección, esto lo representamos como:

$$c(u, v) = -c(v, u)$$

Entonces, el problema que debemos resolver es el de flujo máximo con costo mínimo, teniendo todo esto en consideración.

Cycle cancelling algorithm

El problema posee 2 puntos de optimización. La primera es maximizar el flujo limitado por las capacidades y la segunda es minimizar el costo del flujo (1). Un teorema[2] para redes de grafos indica que un flujo f es el de costo mínimo si y solo si no hay ciclos negativos en los costos del grafo residual. Esto es porque[3]:

[1] (Klein, 1967)
[2] (Busacker & Saaty, 1965)
[3] (Erickson, 2017)

Sea γ un ciclo de costo negativo en el grafo residual G_f y llamando C_m a la capacidad residual mínima presente en γ. Podemos aumentar el flujo sobre este ciclo de forma que las aristas que no pertenezcan al ciclo se vean inafectadas y las que si pertenezcan al ciclo se modifiquen así:

$$f_{nuevo}(u,v) = \begin{cases} f(u,v) + C_m & cuando\ (u,v) \in \gamma \\ f(u,v) - C_m & cuando\ (v,u) \in \gamma \end{cases}$$

Entonces, el costo del nuevo flujo se obtiene de:

$$c(f_{nuevo}) = c(f) + C_m . c(\gamma)$$

De acá, es evidente que mientras $c(\gamma)$ sea negativo, es decir que existan ciclos de costos negativos va a existir un flujo más barato que el actual. [4]

Entonces, ahora entendemos que es necesario que el grafo residual final no tenga ciclos negativos. Teniendo esto en consideración, el algoritmo para calcular el flujo máximo de costo mínimo, que fue originalmente propuesto por Morton Klein en 1967, propone 5 pasos para resolver el problema[5].

1) Obtener el flujo máximo de la red sin considerar el costo. (Klein propone hacerlo con Ford-Fulkerson.)
2) Actualizar el grafo residual G_f con el costo por unidad de flujo negativo, como se mencionó previamente.
3) Probar si hay ciclos negativos dirigidos en el grafo G_f. En caso de no haber, se terminó el problema. (Se propone hacer con un método matricial que desconozco. Claramente es con Bellman-Ford para nosotros.)
4) Se redistribuye el flujo de manera que se satura una de las aristas del ciclo negativo.
5) Repetir desde el punto 2

Entonces, la idea es primero obtener el flujo máximo da la red dada. Luego, por lo presentado anteriormente, mientras existan ciclos de costo negativo sabemos que se puede reducir el costo saturando una de las aristas del ciclo negativo. Como sucede en Ford-Fulkerson, cada vez que aumentamos el flujo reducimos el costo total del flujo por al menos 1. Esto último se cumple porque los costos de cada arista son enteros positivos, pues el precio de los pasajes son números enteros. De esta manera, tenemos que si comenzamos con un costo inicial $C_i(f)$ vamos a ir reduciendo el costo de a por lo menos 1 hasta llegar al costo final $C_f(f)$. Entonces, el algoritmo eventualmente va a finalizar en $C_i(f) - C_f(f)$ iteraciones y obtendremos la solución óptima.

[4] (Erickson, 2017)
[5] (Klein, 1967)

Pseudocódigo

```
cycle-canceling():
    Sea f el flujo máximo
    Sea G(f) el grafo residual
    Obtener f y G(f) mediante Ford-Fulkerson
    Mientras G(f) tenga un ciclo negativo C obtenido con Bellman-Ford
        //Alto capo Ford, está en los dos
        Sea c_m la menor capacidad residual de C
        Aumentar el flujo en el ciclo C en c_m unidades de flujo.
        Actualizar G(f)
    return f y costo mínimo

//Faltaría calcular costo mínimo pero es una papita.
```

Análisis temporal y espacial

Primero aplicamos Ford-Fulkerson para obtener el flujo máximo cuya complejidad temporal es $O(C.E)$, donde C es la capacidad del flujo y E la cantidad de aristas que tiene la red. Luego, se aplica el algoritmo de Bellman-Ford, cuya complejidad es $O(V.E)$. hasta que no haya mas ciclos negativos. Siendo que, como mencionamos antes, cada iteración reduce como mínimo en 1 el costo, se va a llamar a Bellman-Ford un máximo de $C_i(f) - C_f(f)$ veces. Ahora, el costo inicial depende del algoritmo por lo tanto es mejor establecer una cota superior al costo inicial. Llevándolo a un extremo, sea C_{max} la capacidad más alta de una arista, a K_{max} como el costo más alto de una arista, entonces, el costo inicial tiene una cota superior tal que $C_i(f) \leq K_{max}.C_{max}.E$. Pensando que C_f tiene que ser positivo, podríamos maximizar a la resta como $C_i(f) - C_f(f) \leq K_{max}.C_{max}.E$. Por lo tanto, el algoritmo de Bellman-Ford va a ser llamado un máximo de $K_{max}.C_{ma}\ .E$ veces. Entonces, la complejidad temporal del algoritmo resulta $O(V.E^2K_{max}.C_{max})$.

References

Busacker, R., & Saaty, T. L. (1965). *Finite Graphs and Networks.* New York: McGraw-Hill.

C.Crutchfield, K. a. (3 de 5 de 2022). Obtenido de MIT Web Site: https://courses.csail.mit.edu/6.854/06/scribe/s12-minCostFlowAlg.pdf

Erickson, J. (2017). *Algorithms.* Chicago: University of Illinois Press.

Goldberg, & Tarjan. (1989). *Finding Minimum-Cost Circulations.* New Jersey: Murray Hill.

Klein, M. (1967). A PRIMAL METHOD FOR MINIMAL COST FLOWS WITH APPLICATIONS TO THE ASSIGNMENT AND TRANSPORTATION PROBLEMS. In *Management Science, Vol. 14, No. 3* (pp. 205-266). New York: INFORMS.