.UBAfiuba
FACULTAD DE INGENIERÍA

Redes de Flujo: Ford Fulkerson

Teoría de Algoritmos I (75.29 / 95.06)

Ing. Víctor Daniel Podberezski
vpodberezski@fi.uba.ar

Algoritmo Ford-Fulkerson

Propuesto en 1956 por L.R. Ford y D. R. Fulkerson en su paper:

"Maximal Flow Through a Network"

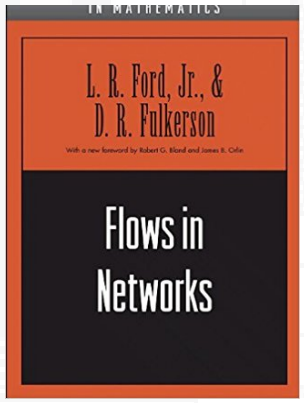
http://www.cs.yale.edu/homes/lans/readings/routing/ford-max_flow-1956.pdf

MAXIMAL FLOW THROUGH A NETWORK

L. R. FORD, JR. AND D. R. FULKERSON

Introduction. The problem discussed in this paper was formulated by T. Harris as follows:

"Consider a rail network connecting two cities by way of a number of intermediate cities, where each link of the network has a number assigned to it representing its capacity. Assuming a steady state condition, find a maximal flow from one given city to the other."



Libro publicado en 1962

Grafo residual

Dado

una red de flujo G, y un flujo f en G,

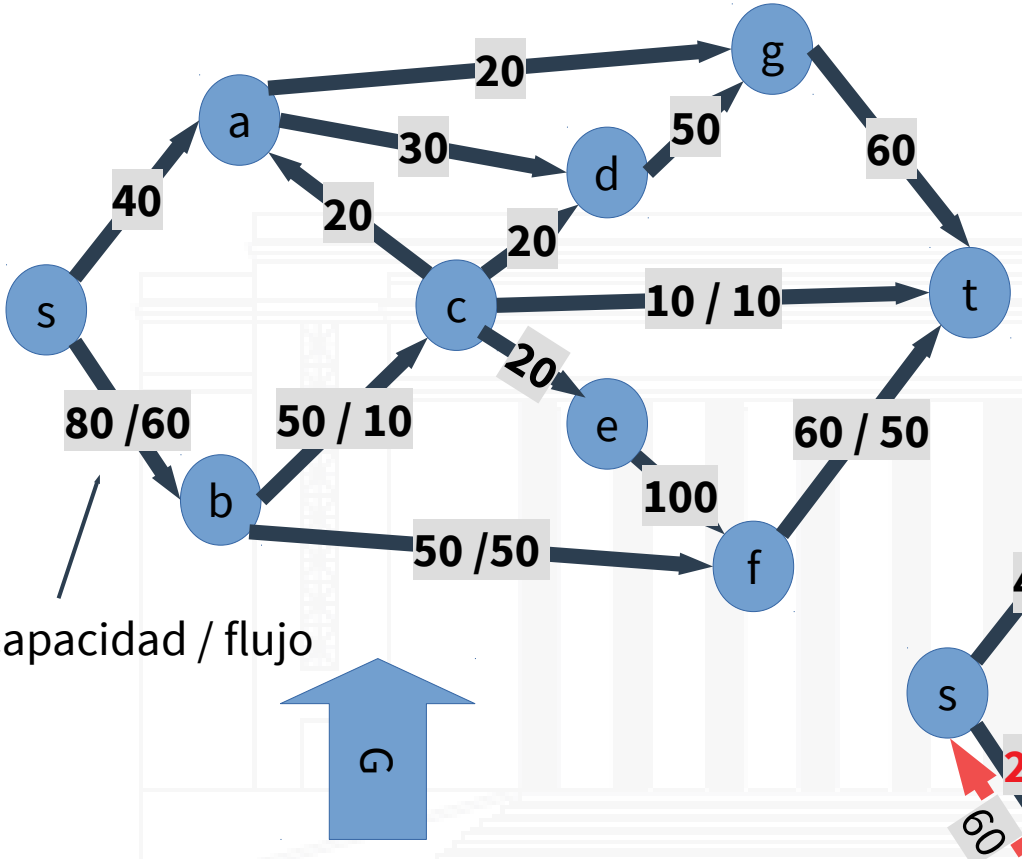
Definimos el grafo residual G_f (de G con respecto a f) a:

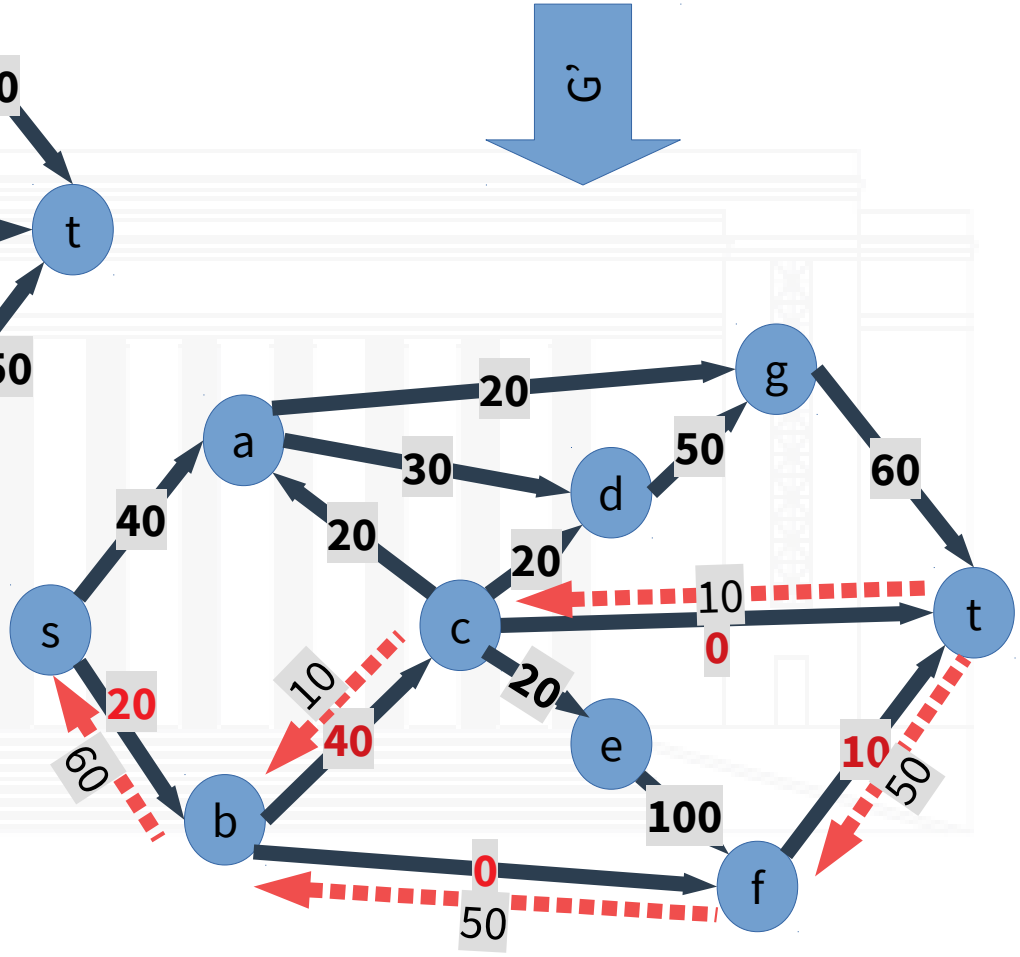
Los mismos Vértices de G

Ejes hacia adelante: Para cada e=(u,v) ϵ E en el que f(e)<C_e. Lo Incluiremos en G_f con capacidad C_e-f(e) [“capacidad residual” de flujo]

Ejes hacia atrás: Para cada e=(u,v) ϵ E en el que f(e)>0. Incluiremos e’=(v,u) con capacidad f(e).

Visualmente





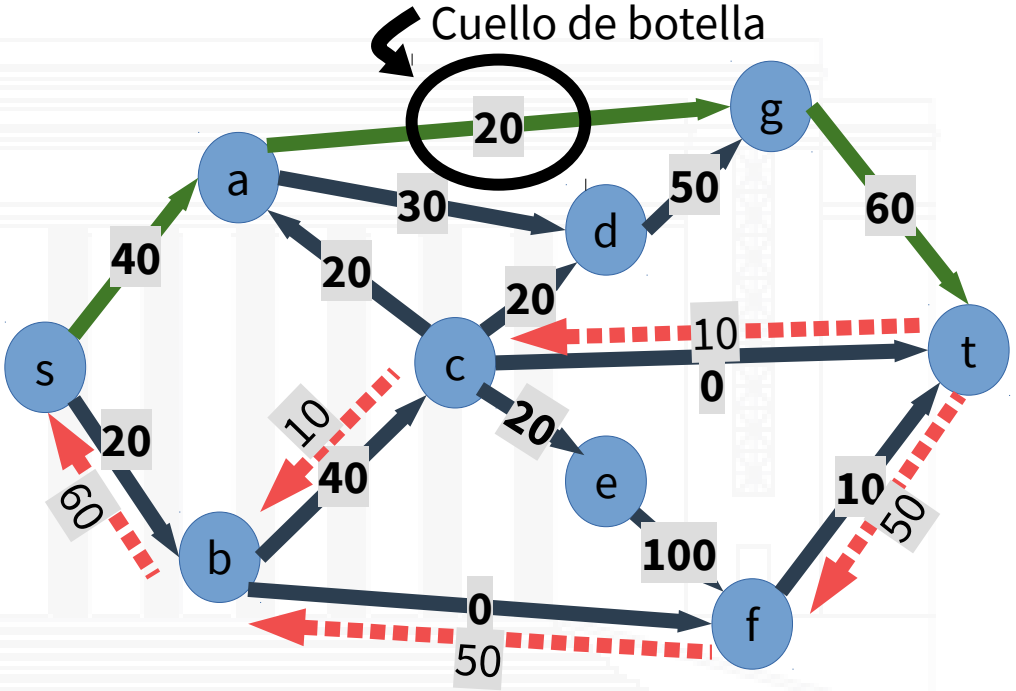
Cuellos de botellas

Sea P un camino simple s-t en G_f

P no visita más de una vez el mismo vértice

Definimos bottleneck(P,f)

a la capacidad residual mínima de cualquier eje de P con respecto al flujo f



Camino de aumento

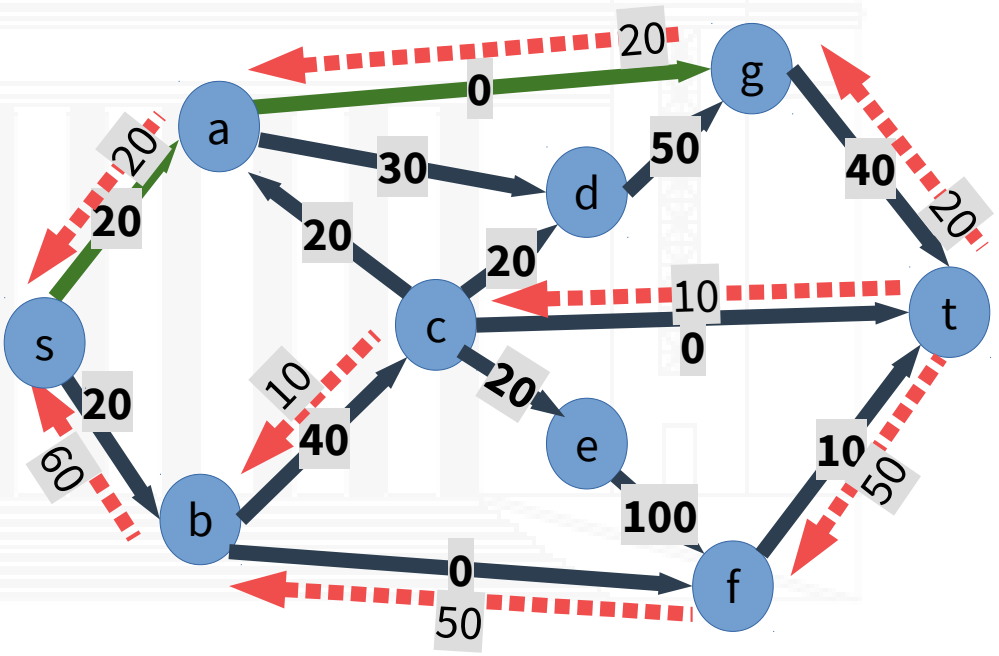
Llamaremos a P como “augmenting path”.

Podemos incrementar botlteneck(P,f) al flujo de P

```
augment(f , P)
    Sea b = bottleneck(P, f )

    Para cada eje e=(u, v) ∈ P
        Si e = (u, v) eje hacia adelante
            f(e)+=b en G
        Sino si es eje para atrás
            e’ = (v, u)
            f(e’)-=b en G

    Retornar f
```



¿Es valido el nuevo flujo?

Se debe verificar que las condiciones de conservación aun se cumplen.

Solo se modificaron los ejes del camino de aumento entre f' y f

Si e=(u,v) es un eje hacia adelante su capacidad residual es C_e-f(e)

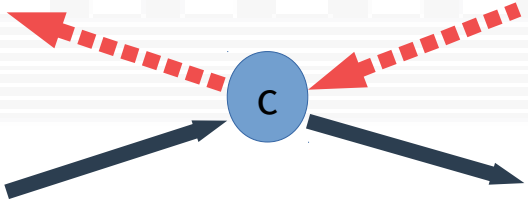
$$0 \leq f(e) \leq f'(e) = f(e) + \mathbf{bottleneck}(P, f) \leq f(e) + (ce - f(e)) = ce$$

Si e=(u,v) es un eje hacia atras su capacidad residual es f(e)

$$ce \geq f(e) \geq f'(e) = f(e) - \mathbf{bottleneck}(P, f) \geq f(e) - f(e) = 0$$

Cada nodo dentro del camino de aumento se

modificará en 2 pares de ejes (uno de entrada y salida)*



* excepto para s y t

Pseudocódigo

```
Max-Flow

    Inicialmente f(e) = 0 Para todo e en G

    Mientras haya un canino s-t en Gf

        Sea P un camino s-t simple en Gf
        f' = augment(f , P)

        Actualizar f para ser f'

        Update Gf para ser Gf'

    Retornar f
```

Análisis del algoritmo: ¿Termina?

A.1 En cada instancia intermedias el valor de los flujos y capacidades residuales son números enteros.

Inicialmente los valores son enteros. Para cada iteración j se resta o suma el valor entero de bottleneck. Por lo tanto siempre serán valores enteros.

A.2 En cada instancia el valor del flujo aumenta.

Lema: Sea f el flujo en G, P camino simple s-t en Gf. Entonces v(f´) = V(f) + bottleneck(P,f). y como bottleneck(P,f)>0 => v(f')>v(f)

proof: El camino s-t sale de s (y no regresa). Por lo tanto el eje

e que sale de s es un camino hacia adelante y aumento el

F(e) en bottleneck(P,f).

Análisis del algoritmo: ¿Termina? (cont.)

A.3 El algoritmo terminará en un numero finito de iteraciones.

Si llamamos C a la suma de todas las C_e de los ejes que salen de la fuente. Vemos que v(f) ≤ C. En cada iteracion el v(f) crece, en el peor de los casos en 1.

Por lo tanto en C iteraciones como limite terminará la ejecución.

La complejidad del algoritmo es O(|E|C)

la cantidad de vértices |V| es menor a los ejes |E|.

El grafo residual tiene a los sumo 2|E| ejes.

Buscar los caminos de aumento (BFS o DFS) → O(|V|+|E|) ≈ O(|E|).

`augment(f , P)` Tomara a lo sumo O(|V|).

Construir un grafo residual tomará O(|E|)

El proceso se iterará a lo sumo C veces

Análisis del Algoritmo: ¿Es óptimo?

A.4 Sea f un flujo s-t y (A,B) cualquier corte, entonces $v(f) = f_{out}(A) - f_{in}(A)$

Proof: Sabemos que $v(f) = f^{out}(s)$, $f^{in}(s)=0 \Rightarrow v(f) = f^{out}(s) - f^{in}(s)$, Como todos los vértices de A (menos s) son internos: $f^{out}(v) - f^{in}(v) = 0$. Entonces:

$$v(f) = \sum_{v \in A} (f^{\text{out}}(v) - f^{\text{in}}(v)),$$

Si los ejes de de un vertice en A solo se conectan con otro en A entonces su $f^{out}(e)$ y $f^{in}(e)$ se cancelan. Solo quedara el aportes de los ejes que entran de B y que salen a B.

$$\sum_{v \in A} f^{\text{out}}(v) - f^{\text{in}}(v) = \sum_{e \text{ out of } A} f(e) - \sum_{e \text{ into } A} f(e) = f^{\text{out}}(A) - f^{\text{in}}(A).$$

Análisis del Algoritmo: ¿Es óptimo? (cont.)

A.5 Sea f un flujo s-t, (A,B) un corte s-t. Entonces v(f)≤ c(A,B)

$$
\begin{aligned}
v(f) &= f^{\text{out}}(A) - f^{\text{in}}(A) \\
&\leq f^{\text{out}}(A) \\
&= \sum_{e \text{ out of } A} f(e) \\
&\leq \sum_{e \text{ out of } A} c_e \\
&= c(A, B).
\end{aligned}
$$

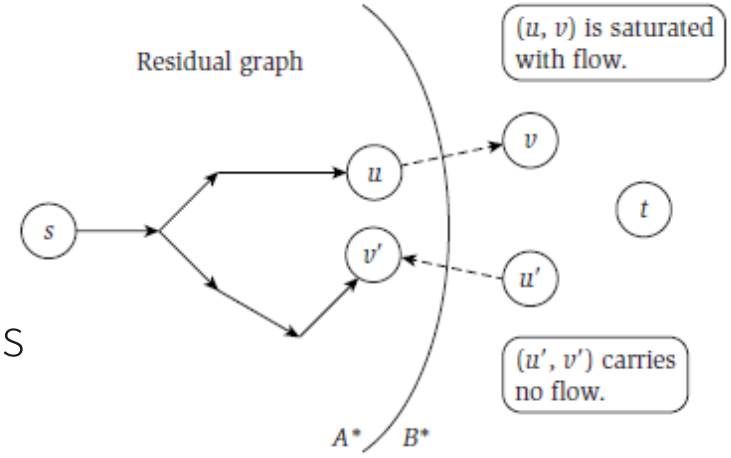
Análisis del Algoritmo: ¿Es óptimo? (y sigue ...)

A.6 Si f es un flujo s-t , tal que no hay un camino s-t en el grafo residual G_f, entonces existe un corte s-t (A*,B*) en G, en el que v(f) = c(A*,B*). Por lo tanto f tiene el máximo valor de cualquier flujo en G y (A*,B*) tiene la mínima capacidad de cualquier corte s-t en G.

Sea A* el set de todos los nodos v en G que tienen un camino s-v en Gf. Sea B* el complemento. Si asumimos que no existe un camino s-t entonces t ϵ B*.

Supongamos que e=(u,v) ϵ E, u ϵ A y v ϵ B. Entonces f(e)=Ce (de lo contrario existe en G_f un camino hacia adelante).

Supongamos que e'=(u',v') ϵ E, u' ϵ B y v' ϵ A. Entonces f(e')=0. (Sino existiria en Gf un camino hacia atrás)



Análisis del Algoritmo: ¿Es óptimo? (y sigue ...)

Todos los ejes de A* a B* están saturados
Todos los ejes de B* a A* están sin utilizar

Por lo tanto partiendo de un resultado previo:

$$
\begin{aligned}
v(f) &= f^{\text{out}}(A^*) - f^{\text{in}}(A^*) \\
&= \sum_{e \text{ out of } A^*} f(e) - \sum_{e \text{ into } A^*} f(e) \\
&= \sum_{e \text{ out of } A^*} c_e - 0 \\
&= c(A^*, B^*). \quad \blacksquare
\end{aligned}
$$

Análisis del Algoritmo: Si, es optimo!

Dado que el Algoritmo termina cuando no hay caminos s-t en el Gf,

Dado (A.6)

El flujo retornado por el algoritmo Ford-Fulkerson es el flujo máximo

Ademas podemos mediante BFS en G_f construir el corte mínimo s-t (A,B) obteniendo A y luego por diferencia B

Algunas consideraciones

Que pasa si las capacidades NO son enteras?

Si son racionales: multiplicar por mínimo común múltiplo

Si son irracionales: no esta asegurado que el algoritmo termine (una selección de caminos desafortunada puede llevarnos a seleccionar cada vez valores menores de flujo, asintóticamente al cero)

Sin embargo, si existe un flujo máximo (las demostraciones utilizadas no usan la hipótesis de número entero)

.UBAfiuba

FACULTAD DE INGENIERÍA

Presentación realizada en Mayo de 2020