Teórica 6: Flujo máximo en redes

1. Introducción

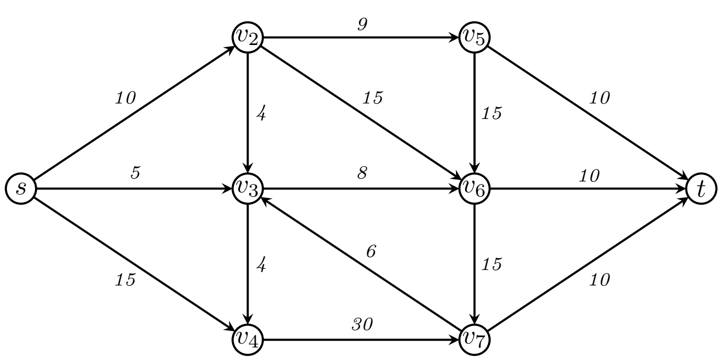
Las redes se presentan en infinidad de formas en la vida cotidiana, como por ejemplo las redes eléctricas y telefónicas, redes de transporte, redes de servicios aéreos, redes de distribución de mercadería, redes de computadores, líneas de producción, etc.

En todos estos ejemplos, el objetivo es mover alguna entidad (líquido a través de cañería, electricidad en una red eléctrica, personas en una red de rutas, mercadería en una red de abastecimiento, datos en una red de comunicación, partes en una línea de ensamblado) desde un punto a otro de la red de la forma más eficiente posible.

Definición 1.

- Una ***red*** $N = (V, X)$ es un grafo orientado conexo que tiene dos vértices distinguidos, una fuente s, con grado de salida positivo, y un sumidero t, con grado de entrada positivo.
- Una ***función de capacidades*** en la red es una función $c : X \to \mathbb{R}^{\geq 0}$.

Ejemplo 1.



Los problemas básicos que se pueden presentar en una red son:

- Problemas de camino mínimo.
- Problema de flujo máximo: Dada una red con capacidades de flujo en los arcos, ¿cómo podemos enviar la máxima cantidad de flujo posible desde un origen a un sumidero determinados respetando las capacidades máximas de los arcos?
- Problema de flujo de costo mínimo: Si, además de la capacidad máxima de flujo que puede atravesar un arco, cada arco tiene asociado un costo por unidad de flujo enviado a través de él, ¿cómo podemos enviar una determinada cantidad de flujo desde un origen a un sumidero determinados respetando las capacidades máximas de los arcos a menor costo?

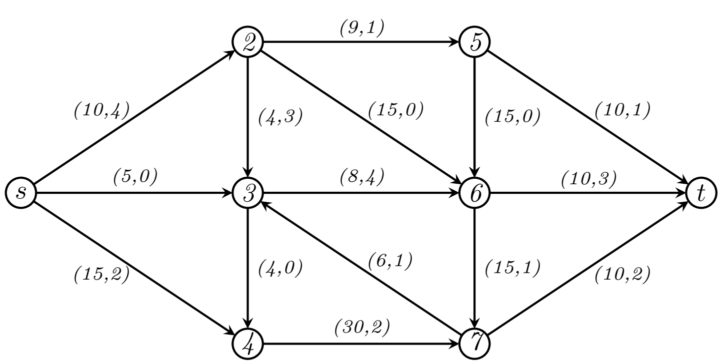
En la clase nos vamos a dedicar al problema de flujo máximo. Las entidades que viajan son producidas en una fuente y son consumidas en un sumidero. Los vértices que no son ni la fuente ni el sumidero, no producen ni consumen material. Es decir, la cantidad de material que entra al vértice debe ser igual a la cantidad que sale del vértice. Cada arco en la red tiene una capacidad máxima de producto que puede atravesarlo.

Definición 2.
Dada una red $N = (V, X)$ con función de capacidad c, fuente s y sumidero t:

- Un ***flujo factible*** es una función $f : X \to \mathbb{R}^+$ que verifica:
    1. $0 \leq f(e) \leq c(e)$ para toda arista $e \in X$.
    2. Ley de conservación de flujo:
    $$\sum_{e \in In(v)} f(e) = \sum_{e \in Out(v)} f(e)$$
    para todo vértice $v \in V \setminus \{s, t\}$, donde
    $$In(v) = \{e \in X, e = (w \to v), w \in V\}$$
    $$Out(v) = \{e \in X, e = (v \to w), w \in V\}$$
- El ***valor del flujo*** es
$$F = \sum_{e \in In(t)} f(e) - \sum_{e \in Out(t)} f(e).$$

Ejemplo 2. *En cada arista, la primera coordenada del par ordenado representa su capacidad, mientras que la segunda el valor del flujo sobre esa arista.*

$$F = 6$$



Es indistinto definir el valor del flujo sobre la fuente o sobre el sumidero, como lo muestra este lema.

Lema 1. *Dada una red $N = (V, X)$ con función de capacidad c, fuente s y sumidero t:*

$$\sum_{e\in Out(s)} f(e) - \sum_{e\in In(s)} f(e) = F$$

Demostración.

$$0 = \sum_{e\in X} f(e) - \sum_{e\in X} f(e) = \sum_{v\in V} \sum_{e\in Out(v)} f(e) - \sum_{e\in In(v)} f(e) = \sum_{e\in Out(s)} f(e) - \sum_{e\in In(s)} f(e) + \sum_{e\in Out(t)} f(e) - \sum_{e\in In(t)} f(e)$$

$$\implies \sum_{e\in Out(s)} f(e) - \sum_{e\in In(s)} f(e) = F.$$

■

2. Problema de flujo máximo

Definición 3. El ***problema de flujo máximo*** consiste en, dada una red $N = (V, X)$ con capacidades en los arcos y dos vértices específicos, una fuente s, y un sumidero t, determinar el flujo factible de valor máximo F que se puede enviar de s a t.

Aplicaciones de este problema se presentan de forma directa, como en redes de envío de combustibles, autos en una red de rutas, mensajes en una red de computadoras, electricidad en una red eléctrica, pero también en otras aplicaciones, como machine scheduling, asignación de módulos de un programa a procesadores o conectividad en grafos. Algunas veces también aparece como subproblema de problemas en redes más complicados, como en el caso del problema de flujo de costo mínimo.

Un problema relacionado es el problema de corte mínimo.

Definición 4. Un ***corte*** en la red $N = (V, X)$ es un subconjunto $S \subseteq V \setminus \{t\}$, tal que $s \in S$.

Notación: Dados $S, T \subseteq V$, llamaremos $ST = \{(u \to v) \in X : u \in S \text{ y } v \in T\}$.

Proposición 1. *Sea f un flujo factible definido en una red N con valor F y sea S un corte, entonces*

$$F = \sum_{e\in S\bar{S}} f(e) - \sum_{e\in \bar{S}S} f(e)$$

donde $\bar{S} = V \setminus S$.

Demostración. Sumando las igualdades de conservación de flujo para los vértices en $S\setminus\{s\}$ y $F = \sum e \in Out(s) f(e) - \sum e \in In(s) f(e)$, obtenemos que

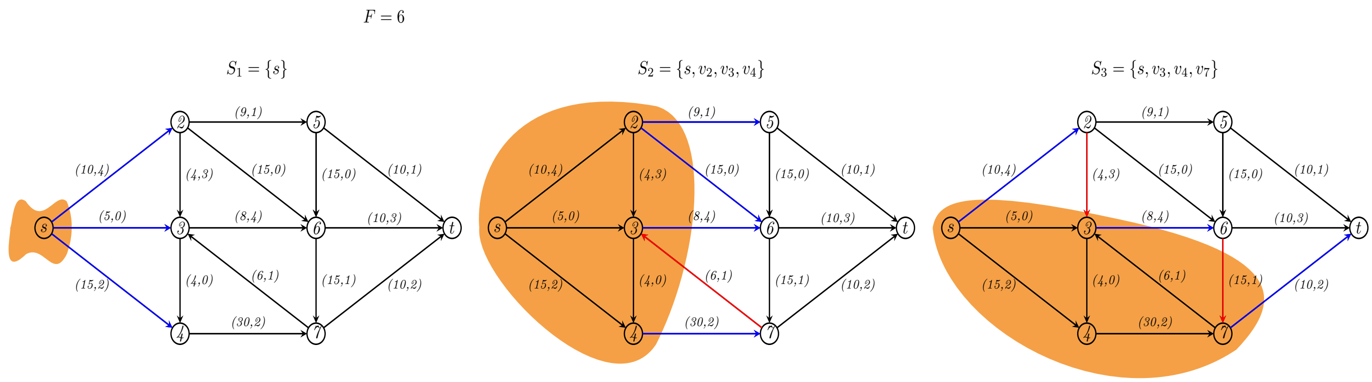
$$F = \sum_{v\in S} (\sum_{e\in Out(v)} f(e) - \sum_{e\in In(v)} f(e)) = \sum_{e\in SV} f(e) - \sum_{e\in VS} f(e))$$

Como los arcos $e \in SS$, pertenecen a ambas sumatorias, podemos simplificarlos, quedando

$$F = \sum_{e\in S\bar{S}} f(e) - \sum_{e\in \bar{S}S} f(e)$$

■

Ejemplo 3. *Se muestran los gráficos para tres posibles cortes, S_1, S_2 y S_3. El valor del flujo F es igual a la suma de los flujos sobre los arcos azules menos la suma de los flujos sobre los arcos rojos.*



Definición 5. Dada una red $N = (V, X)$ con función de capacidad c, la capacidad de un corte S se define como

$$c(S) = \sum_{e \in S\bar{S}} c(e).$$

El problema de optimización asociado a este concepto es el siguiente:

Definición 6. El ***problema de corte mínimo*** consiste en, dada una red $N = (V, X)$ con función de capacidades en los arcos c, determinar un corte capacidad mínima. Es decir, encontrar S corte de N tal que:

$$c(S) = \text{mín}\{c(\bar{S}) | \bar{S} \text{ es un corte de } N\}.$$

Los siguientes resultados relacionan ambos problemas de una forma dual.

Lema 2. *Dados una red $N = (V, X)$ con función de capacidad c, una función de flujo factiblr con valor F y un corte S, se cumple que:*

$$F \leq c(S).$$

Demostración. De la Proposición 1 sabemos que

$$F = \sum_{e \in S\bar{S}} f(e) - \sum_{e \in \bar{S}S} f(e)$$

Como $0 \leq f(e) \leq c(e) \forall e \in X$

$$\sum_{e \in S\bar{S}} f(e) - \sum_{e \in \bar{S}S} f(e) \leq \sum_{e \in S\bar{S}} c(e) - \sum_{e \in \bar{S}S} 0$$

$$\Longrightarrow F \leq \sum_{e \in S\bar{S}} c(e) = c(S)$$

■

Corolario 1 (Certificado de optimalidad)**.** *Si F es el valor de un flujo factible f y S un corte en una red N tal que $F = c(S)$ entonces:*

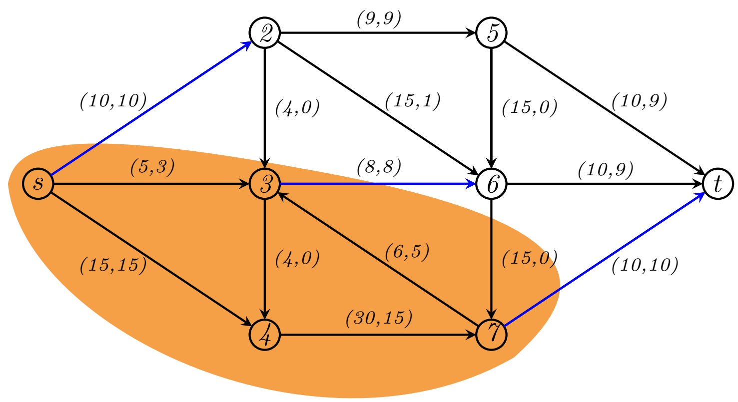
- *f define un flujo factible máximo y*

- *S es un corte de capacidad mínima.*

Ejemplo 4.

$$S = \{s, v_3, v_4, v_7\} \qquad c(S) = 28$$

$$F = 28$$



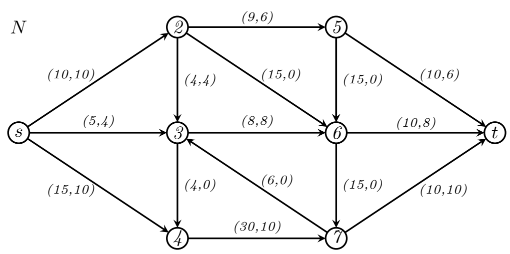
F es flujo factible máximo y S es corte mínimo.

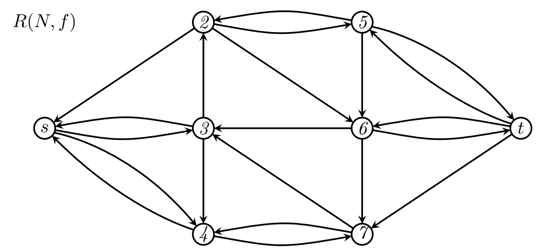
3. Camino de aumento

Definición 7. Dada una red $N = (V, X)$ con función de capacidad c y un flujo factible f,

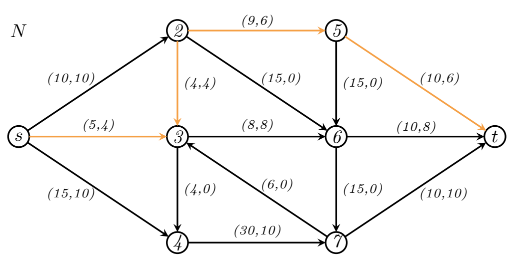
- Definimos la ***red residual***, $R(N, f) = (V, X_R)$ donde $\forall (v \to w) \in X$,
  - $(v \to w) \in X_R$ si $f((v \to w)) < c((v \to w))$
  - $(w \to v) \in X_R$ si $f((v \to w)) > 0$.
- Un ***camino de aumento*** es un camino orientado de s a t en $R(N, f)$.

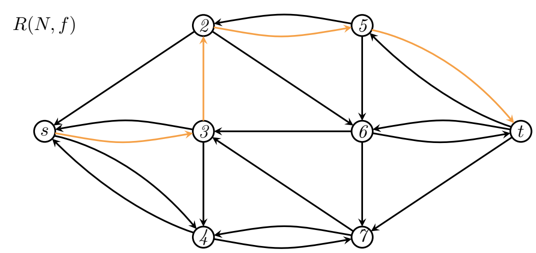
Ejemplo 5. *Por ejemplo, como $f((s \to v_2)) > 0$, $(v_2 \to s) \in X_R$. Como $0 < f((s \to v_4)) < c((s \to v_4))$, $(v_4 \to s) \in X_R$ y $(s \to v_4) \in X_R$.*





En naranja está marcado un camino de aumento:





Dada un red, un flujo factible y su red residual, el siguiente algoritmo encuentra un camino de aumento si existe o, en caso contrario, determina un corte.

```
caminoAumento(N,f,R)
    entrada: N = (V,X), funcion de flujo f, red residual R(N,f) = (V,X_R)
    salida: P camino de aumento o S corte (que sera minimo)

    S ← {s}
    mientras t ∉ S y ∃(v → w) ∈ X_R y v ∈ S y w ∉ S hacer
        ant[w] ← v
        S ← S ∪ {w}
    fin mientras
    si t ∉ S entonces
        retornar S corte de N
    si no
        reconstruir P entre s y t usando ant a partir de t
        retornar P camino de aumento
    fin si
```

Proposición 2. *Dada una red N, un flujo factible f y su residual $R(N,f) = (V, X_R)$, el algoritmo de camino de aumento determina un camino de aumento en $R(N,f)$, P, si existe y, en caso contrario, determina un corte S de N.*

Demostración. Demostremos por inducción que en toda iteración k, los vértices de S_k son alcanzables en $R(N,f)$ desde s siguiendo ant para atrás, donde S_k es el valor de la variable S al salir de la iteración k del ciclo.

Caso base: Antes de ingresar al ciclo, $k = 0$, tenemos que $S_0 = \{s\}$, por lo que se cumple trivialmente.

Paso inductivo: Consideremos S_k, $k \geq 1$.

La hipótesis inductiva es: ***Para $k' < k$, los vértices de $S_{k'}$ son alcanzables desde s siguiendo ant para atrás***.

Sea w el vértice incorporado a S en la iteración k considerando el arco $(v \to w) \in X_R$, es decir, $S_k = S_{k-1} \cup \{w\}$, $v \in S_{k-1}$ y $ant[w] = v$.

Por HI aplicada a S_{k-1}, v es alcanzable desde s siguiendo $ant[v]$ hacia atrás. Sea $P = s \dots v$ el camino definido en $R(N,f)$ por $ant[v]$. Como $(v \to w) \in X_R$, $P' = P + (v \to w)$ es un camino en $R(N,f)$ desde s a w. Por lo tanto w es alcanzable desde s en $R(N,f)$ y, como el algoritmo fija $ant[w] = v$, P' es el camino definido a partir de $ant[w]$.

Nos falta ver que, si al finalizar el ciclo $t \notin S$, es porque no hay camino de aumento en $R(N,f)$. El algoritmo sale del ciclo si sucede que $t \in S$ o $\nexists w \notin S$ tal que $\exists (v \to w) \in X_R$ con $v \in S$.

Si $t \in S$, por lo que vimos anteriormente, el algoritmo retorna P camino de s a t definido por $ant[t]$.

En caso contrario, $\forall w \notin S$, $\nexists (v \to w) \in X_R$ con $v \in S$. Es decir, no hay arco de S a $V \setminus S$. Como $s \in S$ y $t \in V \setminus S$, ésto significa que no hay camino de s a t en $R(N,f)$. Por lo tanto, no hay camino de aumento en $R(N,f)$ y, por definición, S es corte de N. En caso contrario, $\forall w \notin S$, $\nexists (v \to w) \in X_R$ con $v \in S$. Es decir, no hay arco de S a $V \setminus S$. Como $s \in S$ y $t \in V \setminus S$, ésto significa que no hay camino de s a t en $R(N, f)$. Por lo tanto, no hay camino de aumento en $R(N, f)$ y, por definición, S es corte de N.

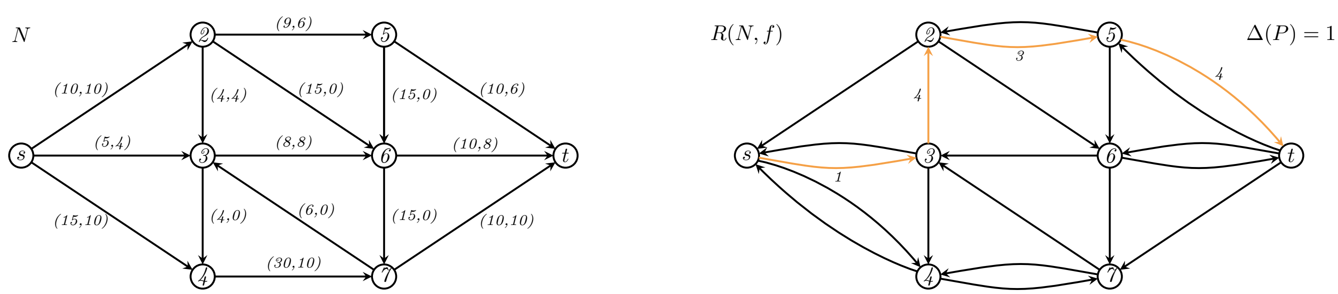
■

El algoritmo de camino de aumento no especifica en qué orden deben incorporarse los vértices a S.

Definición 8. Dada una red $N = (V, X)$ con función de capacidad c, un flujo factible f y un camino de aumento P en $R(N, f)$:

- Para cada arco $(v \to w)$ de P, definimos
$$\Delta((v \to w)) = \begin{cases} c((v \to w)) - f((v \to w)) & \text{si } (v \to w) \in X \\ f((w \to v)) & \text{si } (w \to v) \in X \end{cases}$$
- $\Delta(P) = \text{mín}_{e \in P}\{\Delta(e)\}$

Ejemplo 6.



Proposición 3. *Sea f un flujo factible definido sobre una red N con valor F y sea P un camino de aumento en $R(N, f)$. Entonces el flujo $\bar{f}$, definido por*

$$\bar{f}((v \to w)) = \begin{cases} f((v \to w)) & si\ (v \to w) \notin P \\ f((v \to w)) + \Delta(P) & si\ (v \to w) \in P \\ f((v \to w)) - \Delta(P) & si\ (w \to v) \in P \end{cases}$$

es un flujo factible sobre N con valor $\bar{F} = F + \Delta(P)$.

Demostración. Sabiendo que para $(v \to w) \in X$, $0 \leq f((v \to w)) \leq c((v \to w))$ por ser flujo factible, analicemos los tres casos posibles:

- $(v \to w) \notin P$: $\bar{f}((v \to w)) = f((v \to w))$, por lo que $0 \leq \bar{f}(e) \leq c(e)$.

- $(v \to w) \in P$: Como $0 \leq \Delta(P) \leq \Delta((v \to w)) = c((v \to w)) - f((v \to w))$, se cumple que $0 \leq \bar{f}((v \to w)) = f((v \to w)) + \Delta(P) \leq c((v \to w))$.
- $(w \to v) \in P$: Como $0 \leq \Delta(P) \leq \Delta((w \to v)) = f((v \to w))$, se cumple que $0 \leq \bar{f}((v \to w)) = f((v \to w)) - \Delta(P) \leq c((v \to w))$.

Entonces $0 \leq \bar{f}(e) \leq c(e)$ para todo $e \in X$.

Veamos ahora que $\bar{f}$ cumple la ley de conservación de flujo, sabiendo que f la cumple.

Para los vértices $v \notin P$, sigue valiendo la ley de conservación de flujo, porque $f(e) = f(\bar{e})\ \forall e \in In(v) \cup Out(v)$.

Si $v \in P$, $v \neq s, t$, $P : s \dots uvw \dots t$, tiene que pasar una de estas cuatro configuraciones en N:

1. $u \xrightarrow{+\Delta(P)} v \xleftarrow{-\Delta(P)} w$

   $\sum_{e \in In(v)} f(\bar{e}) = \sum_{e \in In(v)} f(e) + \Delta(P) - \Delta(P)$

   $\sum_{e \in Out(v)} f(\bar{e}) = \sum_{e \in Out(v)} f(e)$

2. $u \xleftarrow{-\Delta(P)} v \xleftarrow{-\Delta(P)} w$

   $\sum_{e \in In(v)} f(\bar{e}) = \sum_{e \in In(v)} f(e) - \Delta(P)$

   $\sum_{e \in Out(v)} f(\bar{e}) = \sum_{e \in Out(v)} f(e) - \Delta(P)$

3. $u \xrightarrow{+\Delta(P)} v \xrightarrow{+\Delta(P)} w$

   $\sum_{e \in In(v)} f(\bar{e}) = \sum_{e \in In(v)} f(e) + \Delta(P)$

   $\sum_{e \in Out(v)} f(\bar{e}) = \sum_{e \in Out(v)} f(e) + \Delta(P)$

4. $u \xleftarrow{-\Delta(P)} v \xrightarrow{+\Delta(P)} w$

   $\sum_{e \in In(v)} f(\bar{e}) = \sum_{e \in In(v)} f(e)$

   $\sum_{e \in Out(v)} f(\bar{e}) = \sum_{e \in Out(v)} f(e) + \Delta(P) - \Delta(P)$

En todos los casos se mantiene la ley de conservación de flujo, es decir $\sum_{e \in In(v)} f(\bar{e}) = \sum_{e \in Out(v)} f(\bar{e})$.

Finalmente calculemos el valor de $\bar{f}$, $\bar{F}$. El único arco incidente a t cuyo flujo ha cambiado es el último arco de P. Este arco puede ser de la forma $(w \to t)$ o de $(t \to w)$. En el primer caso, $\bar{f}((w \to t)) = f((w \to t)) + \Delta(P)$; y en el segundo, $\bar{f}((t \to w)) = f((t \to w)) - \Delta(P)$. En ambos casos

$$\bar{F} = \sum_{e \in In(t)} \bar{f}(e) - \sum_{e \in Out(t)} \bar{f}(e) = \sum_{e \in In(t)} f(e) - \sum_{e \in Out(t)} f(e) + \Delta(P) = F + \Delta(P).$$

■

El siguiente teorema fija la base para el diseño de algoritmos para resolver el problema de flujo máximo.

Teorema 1. *Sea f un flujo factible definido sobre una red $N = (V, X)$. f es un flujo factible de valor máximo $\iff$ no existe camino de aumento en $R(N, f)$.*

Demostración. ($\Longrightarrow$) Sea f un flujo máximo en una red N. Entonces por, Proposición 3 no existe camino de aumento en $R(N, f)$. Ya que si existiera se podría definir un flujo factible de mayor valor ($\Delta(P) > 0 \; \forall P$ camino de aumento).

($\Longleftarrow$) Supongamos que no existe camino de aumento en $R(N, f)$. Sea S el conjunto de vértices alcanzables desde s en $R(N, f)$ (el conjunto de vértices retornado por el algoritmo de camino de aumento cuando no encuentra camino de aumento). Como no hay camino de aumento, $t \notin S$ y S define un corte en N.

Como $\nexists$ arco en $R(N, f)$ entre $v \in S$ y $w \in \bar{S}$, entonces $\forall (v \to w) \in X$ tal que $(v \to w) \in S\bar{S}$, pasa que $f((v \to w)) = c((v \to w))$; y $\forall (v \to w) \in X$ tal que $(v \to w) \in \bar{S}S$ pasa que $f((v \to w)) = 0$. Por Proposición 1 obtenemos:

$$c(S) = \sum_{e \in S\bar{S}} c(e) = \sum_{e \in S\bar{S}} f(e) = \sum_{e \in S\bar{S}} f(e) - \sum_{e \in \bar{S}S} f(e) = F.$$

Por Corolario 1, f es un flujo máximo con valor F y S un corte con capacidad mínima. ∎

Teorema 2. *Dada una red $N = (V, X)$, el valor del flujo máximo es igual a la capacidad del corte mínimo.*

Demostración. Ver demo anterior. ∎

4. Algoritmo de Ford y Fulkerson

La demostración del Teorema 1 sugiere el siguiente algoritmo:

```
Ford&Fulkerson(N)
    entrada: N = (V, X) con funcion de capacidad c
    salida: f flujo maximo
    definir un flujo inicial en N
    (por ejemplo f(e) ← 0 para todo e ∈ X)
    mientras exista P camino de aumento en R(N, f) hacer
        para cada arco (v → w) de P hacer
            si (v → w) ∈ X entonces
                f((v → w)) ← f((v → w)) + Δ(P)
            si no      comentario: ((w → v) ∈ X)
                f((w → v)) ← f((w → v)) − Δ(P)
            fin si
        fin para
    fin mientras
```

Proposición 4. *Si las capacidades de las arcos de la red $N = (V, X)$ son enteras, el problema de flujo máximo tiene una función de flujo máximo f entero. Es decir, $f(e) \in \mathbb{Z}_{\geq 0}$ para todo $e \in X$.*

Demostración. Mostraremos que, si las capacidades son enteras, la función de flujo al finalizar el algoritmo de F&F, tomando como flujo inicial $f(e) = 0 \; \forall e \in X$, tiene valores enteros. Por Teorema 1, sabemos que ese flujo es máximo.

Haremos inducción en las iteraciones del algoritmo de F&F.

Caso base: Como el algoritmo comienza con un flujo incial cero, $f_0(e) \in \mathbb{Z}_{\geq 0}$ $\forall e \in X$.

Paso inductivo: Analicemos f_k el flujo al finalizar la iteración k del algoritmo de F&F.

Hipótesis inductiva: ***Para $k' < k$, $f_{k'}(e) \in \mathbb{Z}_{\geq 0}$ $\forall e \in X$, donde $f_{k'}$ es la función de flujo al finalizar la iteración k' del F&F.***

Como las capacidades son enteras y, por HI, $f_{k-1}(e) \in \mathbb{Z}_{\geq 0}$ $\forall e \in X$, los valores $\Delta(e)$ serán enteros en $R(N, f_k) = (V, X_R)$ $\forall e \in X_R$. Luego, $\Delta(P)$ es entero, y entonces $f_k(e) \in \mathbb{Z}_{\geq 0}$ $\forall e \in X$.

■

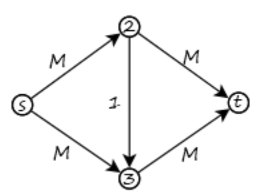
Si las capacidades son racionales, se pueden escalar convenientemente para transformarlos en enteros. Ésto engloba lo que suele suceder en la práctica.

Proposición 5. *Si los valores del flujo inicial y las capacidades de los arcos de la red son enteros, el método de F&F realiza a lo sumo F iteraciones, siendo F el valor del flujo máximo.*

Demostración. Como las capacidades y el flujo inicial son enteros, entonces, si P es un camino de aumento, $\Delta(P) > 0$ y entero. Ésto implica que $\Delta(P) \geq 1$ en toda iteración del algoritmo. Por lo tanto, el algoritmo incrementa el valor del flujo en, por lo menos, 1 en cada iteración, realizando, a lo sumo, F iteraciones. ■

Hay redes donde esta cota se cumple por igualdad, como en el caso de la siguiente red.

Ejemplo 7. *En esta red, donde se muestran las capacidades, el flujo máximo es $F = 2M$. Si los caminos de aumento P de todas las iteraciones incluyen el arco $(v_2 \rightarrow v_3)$, $\Delta(P) = 1$ en toda iteración, debiendo hacer $F = 2M$ iteraciones si se comienza con flujo inicial igual a 0.*



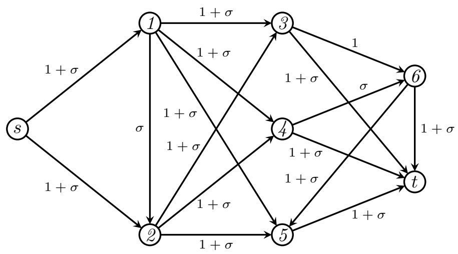
Proposición 6. *Si los valores del flujo inicial y las capacidades de los arcos de la red son enteros, el método de Ford y Fulkerson es $\mathcal{O}(nmU)$, donde U es una cota superior finita para el valor de las capacidades.*

Demostración. Si las capacidades están acotadas por un número finito U, la capacidad del corte $S = \{s\}$ es a lo sumo nU, y entonces el valor del flujo máximo es a lo sumo nU. Entonces, por la Proposición 5, el algoritmo a lo sumo realiza nU iteraciones. Como el algoritmo de camino de aumento es $\mathcal{O}(m)$, porque examina cada arco a lo sumo una vez, el orden total del algoritmo es $\mathcal{O}(nmU)$. ■

Si las capacidades o el flujo inicial son números irracionales, el método de Ford y Fulkerson puede no parar (realizar un número infinito de pasos), como lo muestra este ejemplo.

Ejemplo 8.

$\sigma = (\sqrt{5} - 1)/2$



Iteración	*Camino de aumento*
$6k+1$	$s, 1, 2, 3, 6, t$
$6k+2$	$s, 2, 1, 3, 6, 5, t$
$6k+3$	$s, 1, 2, 4, 6, t$
$6k+4$	$s, 2, 1, 4, 6, 3, t$
$6k+5$	$s, 1, 2, 5, 6, t$
$6k+6$	$s, 2, 1, 5, 6, 4, t$

Si no se especifica el orden en el que se eligen los arcos y vértices al marcar en el algoritmo de camino de aumento, el número de iteraciones de F&F, aún con capacidades enteras, puede no ser polinomial respecto del tamaño de la entrada del problema.

Edmonds y Karp definieron una implementación del método de F&F utilizando BFS para el cálculo del camino de aumento. El procedimiento encontrará el camino entre s y t en la red residual de menor cantidad de arcos. Esta implementación es $\mathcal{O}(nm^2)$, mostrando un algoritmo polinomial para el problema de flujo máximo con capacidades enteras. Existen algoritmos más eficientes para este problema, pero más complicados, por ejemplo el algoritmo de Dinic [1].

Para ver la complejidad de E&K, analicemos la cantidad de iteraciones que realiza, a lo sumo, este algoritmo.

Teorema 3. *Dada una red $N = (V, X)$ con n vértices y m arcos, el algoritmo de E&K realiza, a lo sumo, nm iteraciones.*

Para organizar esta demostración, primero demostraremos un lema. Llamemos $d_{R(N,f)}(s, v)$ a la distancia de s a v (cantidad de aristas mínima de un camino orientado de s a v) en la red residual de N relativa al flujo factible f, $R(N, f)$.

Lema 3. *Sea f_i el flujo de la iteración i de algoritmo de E&K. Entonces, para todo $v \in V$, $d_{R(N,f_i)}(s, v) \leq d_{R(N,f_{i+1})}(s, v)$ para toda iteración i del algoritmo.*

Demostración (esta demostración es opcional). Por el absurdo, supongamos que existe $v \in V \setminus \{s\}$ tal que $d_{R(N,f_i)}(s, v) > d_{R(N,f_{i+1})}(s, v)$ para alguna iteración i. Sea w un vértice con menor $d_{R(N,f_{i+1})}(s, w)$ entre los vértices para los cuales esta distancia disminuyó entre la iteración i e $i+1$, $w = \arg\min\{d_{R(N,f_{i+1})}(s, v) : d_{R(N,f_i)}(s, v) > d_{R(N,f_{i+1})}(s, v)\}$.

Sea P un camino de s a w de $d_{R(N,f_{i+1})}(s, w)$ aristas, y u el vértice justo anterior a w en P, $P : s \dots uw$. Entonces $(u \to w) \in X_{R(N,f_{i+1})}$ y $d_{R(N,f_{i+1})}(s, w) = d_{R(N,f_{i+1})}(s, u) + 1$.

Por como elegimos w y dado que $d_{N(G,f_{i+1})}(s,u) < d_{N(G,f_{i+1})}(s,w)$, se cumple que $d_{R(N,f_i)}(s,u) \leq d_{R(N,f_{i+1})}(s,u)$.

Si $(u \to w) \in X_{R(N,f_i)}$, tendríamos que:

$$d_{R(N,f_i)}(s,w) \leq d_{R(N,f_i)}(s,u) + 1 \leq d_{R(N,f_{i+1})}(s,u) + 1 = d_{R(N,f_{i+1})}(s,w)$$

contradiciendo que $d_{R(N,f_i)}(s,w) > d_{R(N,f_{i+1})}(s,w)$, que se cumple por la elección de w.

Por lo tanto, $(u \to w) \notin X_{R(N,f_i)}$ y $(u \to w) \in X_{R(N,f_{i+1})}$. Ésto significa que $(w \to u)$ está en el camino de aumento seleccionado en $R(N,f_i)$, $P_{f_i} : s \dots wu \dots t$.

Como este camino se calculó usando BFS, es un camino de menor cantidad de aristas. Por lo tanto $d_{R(N,f_i)}(s,u) = d_{R(N,f_i)}(s,w) + 1$. Entonces:

$$d_{R(N,f_i)}(s,w) = d_{R(N,f_i)}(s,u) - 1 \leq d_{R(N,f_{i+1})}(s,u) - 1 = d_{R(N,f_{i+1})}(s,w) - 2.$$

Ésto contradice que $d_{R(N,f_i)}(s,w) > d_{R(N,f_{i+1})}(s,w)$. Por lo tanto no existe $w \in V$ tal que $d_{R(N,f_i)}(s,w) > d_{R(N,f_{i+1})}(s,w)$ para alguna iteración i de E&K, demostrando el lema.

■

Ahora sí demostremos el Teorema que limita la cantidad de iteraciones de E&K.

Demostración Teorema 3 (esta demostración es opcional). Llamamos crítico a un arco e de una red residual $R(N,f) = (V, X_R)$ si $\Delta(P) = \Delta(e)$, donde P es el camino de aumento encontrado. En la iteración siguiente, el arco e no estará en la red residual, ya que el flujo sobre este arco será incrementado a $c(e)$ o disminuido a 0, según sea el caso. Veremos que cada arco puede ser crítico a lo sumo $n/2$ veces. Como hay $2m$ arcos potenciales a ser críticos y en cada iteración por lo menos habrá un arco crítico, ésto implica que el algoritmo puede realizar, a lo sumo, mn iteraciones.

Veamos ahora cuántas veces un arco puede ser crítico. Sea $(u \to w) \in X_R$ un arco crítico en la i-ésima iteración del algoritmo, en la red residual de N con respecto a f_i, $R(N,f_i)$. Por lo tanto, $d_{R(N,f_i)}(s,w) = d_{R(N,f_i)}(s,u) + 1$.

En la iteración siguiente ese arco no estará en la red residual y sí estará $(w \to u)$ (puede ser que ya estuviese). Para que el arco $(u \to w)$ vuelva a estar en una red residual, $(w \to u)$ debe ser elegido en un camino de aumento, para así disminuir o aumentar, según sea el caso, el flujo que atraviesa a $(u \to w)$. Supongamos que ésto sucede en la iteración j, con $j > i$. Entonces $d_{R(N,f_j)}(s,u) = d_{R(N,f_j)}(s,w) + 1$.

Por Lema 3, sabemos que $d_{R(N,f_j)}(s,u) = d_{R(N,f_j)}(s,w) + 1 \geq d_{R(N,f_i)}(s,w) + 1 = d_{R(N,f_i)}(s,u) + 2$.

Ésto muestra que entre dos iteraciones que el arco $(u \to w) \in X_R$ es crítico, la distancia de s a u aumenta en por lo menos dos arcos. Como lo mínimo que puede ser esta distancia es 1 y lo máximo $n-2$, el arco $(u \to w)$ a lo sumo puede ser crítico $n/2$ veces.

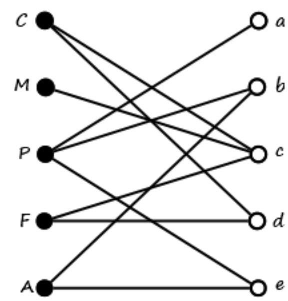
■

5. Matching máximo en grafos bipartitos

Analicemos la siguiente situación:

Hay un conjunto de cinco personas y un conjunto de 5 trabajos para realizar. Sean las personas Carlos, Marcela, Pedro, Fernando y Andrea, y los trabajos a, b, c, d y e. Carlos está capacitado para realizar los trabajos c y d, Marcela para c, Pedro para a, b y e, Fernando para c y d, y Andrea para b y e. ¿Es posible realizar una distribución del trabajo de modo que se puedan realizar todos los trabajos simultáneamente?

Podemos modelar el problema mediante un grafo $G = (V, X)$ donde el conjunto de vértices representa a las personas y los trabajos, y hacemos un vértice respresentando a una persona adyacente a los vértices correspondientes a los trabajos para los que está capacitado. Obtenemos el siguiente grafo:

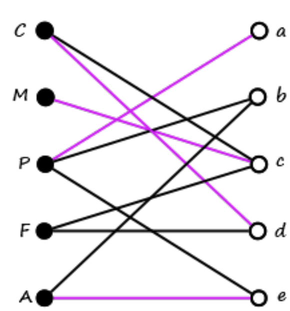


Para poder realizar todos los trabajo simultáneamente, debemos asignarle exáctamente un trabajo a cada persona. Intentemos hacerlo:

- El trabajo a sólo puede realizarlo Pedro, así que no tenemos elección: le asignamos a Pedro el trabajo a.
- Marcela sólo sabe realizar el trabajo c, tampoco tenemos elección: le asignamos a Marcela el trabajo c.
- Carlos está capacitado para realizar los trabajos c y d, pero el c ya lo tiene asignado Marcela: le asignamos el trabajo d a Carlos.
- Fernando puede realizar los trabajos c y d, pero ambos trabajos ya están asignados, por lo que no tenemos trabajo para asignarle Fernando.

Esto ya muestra que no será posible realizar todos los trabajos simultáneamente.

Andrea puede realizar los trabajos b o e. Ambos trabajos todavía están libres y le podemos asignar alguno de los dos, por ejemplo el trabajo e. Entonces sí podemos realizar simultáneamente 4 de los 5 trabajos.



En esta aplicación, que se conococe como problema de asignación de personal, estamos interesados en encontrar un conjunto de aristas de cardinal máximo, tal que no haya dos aristas en el conjunto que incidan sobre el mismo vértice.

Este tipo de escenarios da origen a las siguientes definiciones y estudios.

Definición 9. Dado un grafo $G = (V, X)$

- Un ***matching o correspondencia*** entre los vértices de G, es un conjunto $M \subseteq X$ de aristas de G tal que para todo $v \in V$, v es incidente a lo sumo a una arista $e \in M$.
- El problema de optimización relacionado a este concepto es el problema de ***matching máximo***: consiste en encontrar un matching de cardinal máximo entre todos los matchings de G.

El problema de matching máximo es computacionalmente fácil (se conocen algoritmos polinomiales) para grafos en general. Pero en el caso de grafo bipartitos, podemos enunciar un algoritmo más simple transformándolo en un problema de flujo máximo en una red.
Para esto, dado el grafo bipartito $G = (V_1 \cup V_2, X)$ definimos la siguiente red $N(V', X')$:

- $V' = V_1 \cup V_2 \cup \{s, t\}$, con s y t dos vértices ficticios representando la fuente y el sumidero de la red.
- $X' = \{(u \to w), u \in V_1, w \in V_2, (u, w) \in X\} \cup \{(s, u), u \in V_1\} \cup \{(w \to t), w \in V_2\}$.
- $c(e) = 1$ para todo $e \in X'$.

El cardinal del matching máximo de G será igual al valor del flujo máximo en la red N.

6. Variantes del problema de flujo máximo

Múltiples fuentes y sumideros: Un problema de flujo máximo puede tener múltiples fuentes y sumideros. Por ejemplo sería el caso si estamos modelando el traslado de mercadería proveniente de un conjunto de fábricas, $s_1, \ldots, s_p$, o varios depósitos o clientes, $t_1, \ldots, t_q$. Podemos reducir este problema al problema clásico de máximo flujo que hemos estudiado, agregando una fuente s y un sumidero t ficticios y los arcos (s, s_i), $i = 1, \ldots, p$, y (t_j, t), $j = 1, \ldots, q$ con capacidad infinita. Un flujo en la red original se corresponderá a un flujo en la red ficticia y viceversa. La fuente ficticia s generará tanto flujo como requieran las fuentes s_i y el sumidero ficticio t consumirá tanto flujo como lo hagan los sumideros t_j.

Capacidades en los vértices: En este caso, cada vértice tiene un límite del flujo que lo puede atravesar. Ver ejericio de la práctica.

Flujo en redes no dirigidas: El flujo puede atravesar un enlace en cualquiera de las dos direcciones. Ver ejercicios de la práctica.

Flujo máximo con costo mínimo o acotado:

- el costo es proporcional al flujo transportado a través del arco (polinomial).
- el costo es fijo y se cobra por el uso del arco (no se conocen algoritmos polinomiales).

7. Bibliografía recomendada

- Sección 26 de T. Cormen, C. Leiserson, R. Rivest, C. Stein, *Introduction to Algorithms*, The MIT Press, McGraw-Hill, 2001.
- Sección 12 de J. Gross and J. Yellen, *Graph theory and its applications*, CRC Press, 1999.
- Capítulo 1 y 6 de R. Ahuja, T. Magnanti and J. Orlin, *Network Flows: Theory, Algorithms, and Applications*, Prentice-Hall, 1993.

Referencias

[1] J. McHugh. *Algorithmic Graph Theory*. Prentice Hall, 1990.