Algorítmica
Flujos sobre Redes y Programación Lineal

Conrado Martínez
U. Politècnica Catalunya

ALG Q1-2022–2023

Temario

Parte V

Flujos sobre Redes

Parte V

Flujos sobre Redes

Parte V

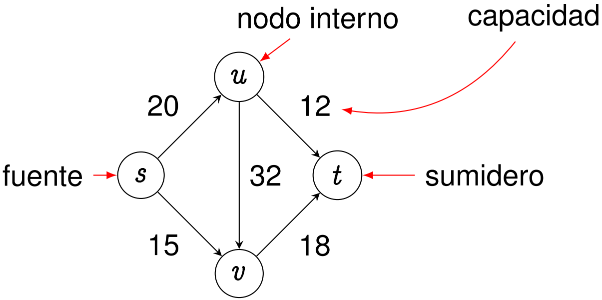
Flujos sobre Redes

Flujos sobre Redes

Definición

Una red s-t es un grafo dirigido $G = \langle V, E \rangle$ donde

1. Cada arco e tiene una capacidad $c_e > 0$.
2. El vértice $s \in V$, denominado fuente (ing: *source*) es el único vértice de G con grado de entrada 0.
3. El vértice $t \in V$, denominado sumidero (ing: *sink*) es el único vértice de G con grado de salida 0.



Flujos sobre Redes

Definición

Un flujo f sobre una red G es una función $f : E(G) \to \mathbb{R}$ tal que:

1. Para todo arco $e \in E(G)$,

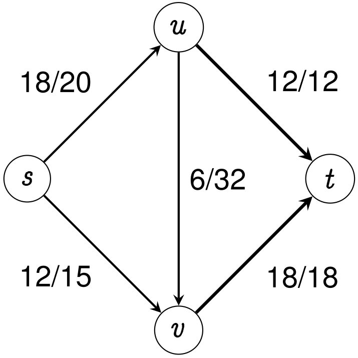
$$0 \leq f(e) \leq c_e \qquad \text{(condiciones de capacidad)}$$

2. Para todo vértice $v \in V(G) \setminus \{s, t\}$,

$$\sum_{e=(v,w)} f(e) - \sum_{e=(w,v)} f(e) = 0 \qquad \text{(conservación del flujo)}$$

Flujos sobre Redes

Un ejemplo de flujo



Las etiquetas x/y sobre los arcos denotan el flujo acarreado por el arco (x) y su capacidad (y); empleamos un trazo más grueso para indicar que un arco lleva flujo, y muy grueso para indicar que está saturado.

Flujos sobre Redes

Si definimos

$$f^{\text{out}}(v) = \sum_{e=(v,w)} f(e) \qquad \text{flujo saliente}$$

$$f^{\text{in}}(v) = \sum_{e=(w,v)} f(e) \qquad \text{flujo entrante}$$

las condiciones de conservación del flujo las podemos reescribir

$$f^{\text{out}}(v) - f^{\text{in}}(v) = 0$$

para todo $v \in V(G) \setminus \{s, t\}$.

Salvo que el flujo f sea nulo ($f(e) = 0$ para todo e), la fuente s emite flujo ($f^{\text{out}}(s) > 0$ y $f^{\text{in}}(s) = 0$) y el sumidero t lo absorbe ($f^{\text{in}}(t) > 0$ y $f^{\text{out}}(t) = 0$).

Si para un arco e se cumple que $f(e) = c_e > 0$ se dice que el arco está saturado.

Flujos sobre Redes

Dado un conjunto de vértices A,

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

El valor $v(f)$ de un flujo f es el flujo saliente de s, que coincide con el flujo entrante en t

$$v(f) = f^{\text{out}}(s) = f^{\text{in}}(t)$$

Flujos sobre Redes

El problema a resolver se puede entonces formular entonces en los siguientes términos:

> *"Dada G, una red s-t, hallar un flujo f máximo, es decir, un flujo sobre G cuyo valor $v(f)$ es máximo."*

Obs: una red G puede admitir varios flujos máximos, todos de idéntico valor.

Un cota trivial al máximo flujo alcanzable:

$$\begin{aligned} v(f) = f^{\text{out}}(s) &= \sum_{e=(s,v)} f(e) \\ &\leq \sum_{e=(s,v)} c_e \end{aligned}$$

Flujos sobre Redes

Definición

Un corte s-t de una red G es una partición $\langle A, B \rangle$ del conjunto de vértices $V(G)$ tal que:

- $A \cup B = V(G)$
- $A \cap B = \emptyset$
- $s \in A$ y $t \in B$

Definición

Dado $\langle A, B \rangle$, un corte s-t de G, su capacidad es

$$c(A, B) = \sum_{\substack{e=(u,v) \\ u \in A, v \in B}} c_e$$

Flujos sobre Redes

Lema

Sea $\langle A, B\rangle$ un corte s-t de una red G, y f un flujo cualquiera sobre la red. Entonces

$$f^{out}(A) - f^{in}(A) = v(f)$$

Demostración

Puesto que $f^{\text{out}}(s) = v(f)$ y $f^{\text{in}}(s) = 0$

$$\begin{aligned} f^{\text{out}}(A) &= f^{\text{out}}(s) + \sum_{v \in A \setminus \{s\}} f^{\text{out}}(v) \\ &= v(f) + \sum_{v \in A \setminus \{s\}} f^{\text{out}}(v) \end{aligned}$$

$$f^{\text{in}}(A) = f^{\text{in}}(s) + \sum_{v \in A \setminus \{s\}} f^{\text{in}}(v) = \sum_{v \in A \setminus \{s\}} f^{\text{in}}(v)$$

Flujos sobre Redes

Demostración (continúa)

Luego

$$
\begin{aligned}
f^{\text{out}}(A) - f^{\text{in}}(A) &= v(f) + \sum_{v \in A \setminus \{s\}} f^{\text{out}}(v) - \sum_{v \in A \setminus \{s\}} f^{\text{in}}(v) \\
&= v(f) + \sum_{v \in A \setminus \{s\}} \left(f^{\text{out}}(v) - f^{\text{in}}(v)\right) \\
&= v(f)
\end{aligned}
$$

puesto que $f^{\text{out}}(v) - f^{\text{in}}(v) = 0$ para todo $v \notin \{s, t\}$. $\square$

Se puede demostrar de manera parecida que para cualquier corte $\langle A, B\rangle$ y cualquier flujo f

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

Flujos sobre Redes

Teorema

Dados un corte s-t cualquiera $\langle A, B\rangle$ y un flujo f cualquiera

$$v(f) \leq c(A, B)$$

"El valor de un flujo no puede exceder la capacidad de ningún corte de la red"

La cota trivial $v(f) \leq \sum_{e=(s,v)} c_e$ es un caso particular del teorema, tomando $A = \{s\}$ y $B = V(G) \setminus \{s\}$.

Demostración

Comenzaremos demostrando un resultado intermedio importante:

$$f^{\text{out}}(A) - f^{\text{in}}(A) = \sum_{e \text{ sale de } A} f(e) - \sum_{e \text{ entra en } A} f(e)$$

Por el lema demostrado anteriormente

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

Flujos sobre Redes

Demostración (continúa)

Tenemos

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

$$f^{\text{in}}(A) = \sum_{v \in A} f^{\text{in}}(v) = \sum_{v \in A} \sum_{e=(w,v)} f(e)$$

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

Flujos sobre Redes

Demostración (continúa)

Consideraremos tres casos:

1. Arcos que salen de v hacia un vértice fuera de A, es decir, de la forma (v, w) con $w \in B$: contribuyen el término
$$\sum_{\substack{e=(v,w)\\ w\in B}} f(e)$$
2. Arcos que llegan a v desde un vértice fuera de A (entran en A), es decir, de la forma (w, v) siendo $w \in B$: contribuyen el término
$$-\sum_{\substack{e=(w,v)\\ w\in B}} f(e)$$

Flujos sobre Redes

Demostración (continúa)

3. Arcos "internos" a A, de la forma (x, y) con $x, y \in A$: cuando sumamos sobre todos los vértices v de A contribuyen 0

$$
\begin{aligned}
\sum_{v \in A} \left(\sum_{\substack{e=(v,w) \\ w \in A}} f(e) - \sum_{\substack{e=(w,v) \\ w \in A}} f(e) \right) \\
= \sum_{\substack{e=(v,w) \\ v \in A, w \in A}} f(e) - \sum_{\substack{e=(w,v) \\ v \in A, w \in A}} f(e) = 0
\end{aligned}
$$

Flujos sobre Redes

Demostración (continúa)

Recapitulando

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

$$= \sum_{e \text{ sale de } A} f(e) - \sum_{e \text{ entra en } A} f(e)$$

Como todos los flujos son positivos o cero,

$$v(f) = f^{\text{out}}(A) - f^{\text{in}}(A) = \sum_{e \text{ sale de } A} f(e) - \sum_{e \text{ entra en } A} f(e)$$

$$\leq \sum_{e \text{ sale de } A} f(e) \leq \sum_{e \text{ sale de } A} c_e = c(A, B)$$

$\square$

Parte V

Flujos sobre Redes

Algoritmo de Ford-Fulkerson

Lester R. Ford Jr (1927–2017)

Delbert R. Fulkerson (1924–1976)

El algoritmo de Ford-Fulkerson (1954) calcula un flujo máximo a través de una serie de etapas sucesivas. Se parte de un flujo nulo y cada etapa envía flujo adicional de s a t, incrementando el valor del flujo en curso.

Para ello se busca, en cada etapa, un camino de s a t en el que se pueda incrementar el flujo que atraviesa los arcos del camino. El valor del flujo se va acercando progresivamente al valor máximo. El algoritmo finaliza cuando no podemos encontrar un camino que nos lleve de s a t en el que podamos incrementar el flujo.

Algoritmo de Ford-Fulkerson

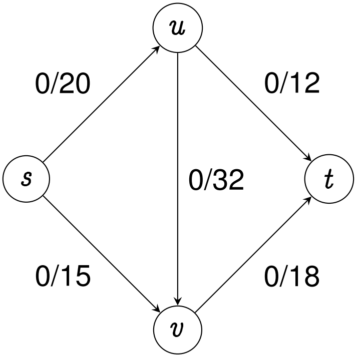
De ahora en adelante asumiremos que todas las capacidades c_e de la red son enteros positivos, y nuestro análisis de la corrección del algoritmo y de su tiempo de ejecución se basarán en esta hipótesis.

Algoritmo de Ford-Fulkerson

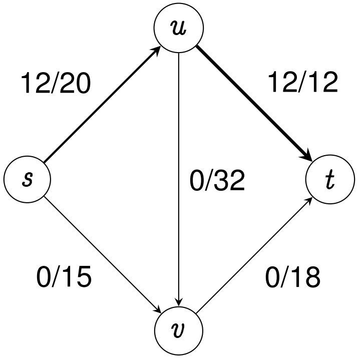
El algoritmo de Ford-Fulkerson va calculando flujos sucesivos $f_0, f_1, f_2, \ldots$ de manera que $v(f_i) < v(f_{i+1})$, con f_0 el flujo nulo; entre dos flujos sucesivos f_i y f_{i+1} sólo son diferentes los flujos enviados a través de los arcos de un cierto camino, todos los restantes arcos llevan el mismo flujo en f_i y en f_{i+1}. A medida que evoluciona el algoritmo podemos pensar en sucesivas redes $G_0 = G, G_1, \ldots$ donde el flujo asignado a los arcos e de G_i les deja una capacidad remanente o residual $c'_e = c_e - f_i(e)$.

Este procedimiento nos puede llevar a un callejón sin salida si elegimos "mal" el camino que nos lleva de s a t, pues podemos saturar arcos vitales para llegar de s a t bloqueando otros posibles flujos de mayor valor. Por ello debemos considerar la posibilidad de retornar el flujo o parte de él que hayamos enviado a través de un arco.

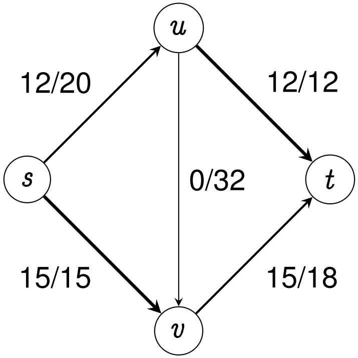
Algoritmo de Ford-Fulkerson



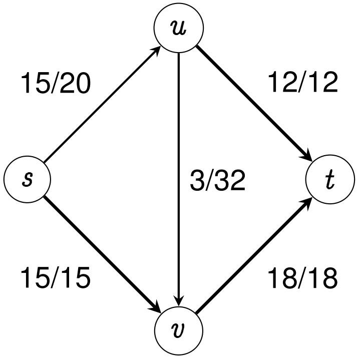
Algoritmo de Ford-Fulkerson



Algoritmo de Ford-Fulkerson



Algoritmo de Ford-Fulkerson



Grafo residual

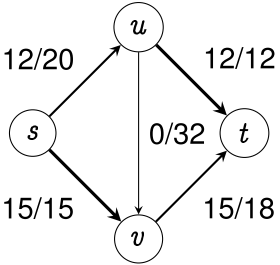
Definición

Dada una red $G = \langle V, E \rangle$ y un flujo f sobre G, el grafo residual (o *red residual*) $G_f = \langle V_f, E_f \rangle$ se define como sigue:

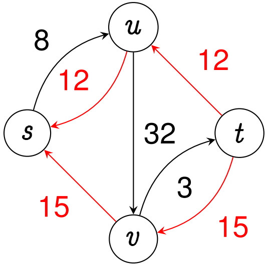
1. $V_f = V$
2. Si $e = (u, v) \in E$ y $f(e) < c_e$ entonces $e = (u, v) \in E_f$ y $\overline{c}_e = c_e - f(e)$ (arcos de avance; ing: *forward edges*)
3. Si $e = (u, v) \in E$ y $f(e) > 0$ entonces $e' = (v, u) \in E_f$ y $\overline{c}_{e'} = f(e)$ (arcos de retroceso; ing: *backward edges*)

Observación: en el grafo residual todas las capacidades residuales $\overline{c}$ son números enteros positivos si los flujos lo son

Grafo residual



Red con flujo



Grafo residual

Los arcos de avance del grafo residual se muestran con línea sólida y etiquetados por la capacidad residual ($\bar{c}_e = c_e - f(e)$); los arcos de retroceso se muestran en color rojo y etiquetados por su capacidad residual ($\bar{c}_{(v,u)} = f(u,v)$)

Caminos de aumentación

Dada una red G y un flujo f, un camino de aumentación (ing: *augmenting path*) es un camino simple entre s y t en el grafo residual G_f.

```
procedure BOTTLENECK(P)
    return  la capacidad residual mínima de un arco de P
end procedure
procedure PUSHFLOW(P, f)
    b := BOTTLENECK(P)
    for e = (u, v) ∈ P do
        if e es un arco de avance en G_f then
            f(e) := f(e) + b
        else ▷ e es un arco de retroceso
            e' := (v, u)
            f(e') := f(e') − b
        end if
    end for
    return  f
end procedure
```

Caminos de aumentación

Observación: Si para todos los arcos e el flujo $f(e)$ es un número entero positivo, lo mismo ocurre con $f' = \textsc{PushFlow}(P, f)$, es decir, $f'(e)$ es un número entero positivo también.

Proposición

Sea $f' = \textsc{PushFlow}(P, f)$. Entonces f' es un flujo válido sobre la red G y $v(f') = v(f) + \textsc{Bottleneck}(P) > v(f)$

Caminos de aumentación

Demostración

Sea $b = \textsc{Bottleneck}(P)$.

1. Condiciones de capacidad: Sólo se modifica el flujo en los arcos de P.
   Supongamos que e es un arco de avance.
   Entonces $f'(e) = f(e) + b$. Pero $\overline{c}_e = c_e - f(e) \geq b$ pues b es la capacidad residual mínima, luego $c_e \geq b + f(e) = f'(e)$. Como $f(e) \geq 0$ y $b > 0$ tenemos que $f'(e) > 0$.
   Supongamos ahora que $e = (u, v)$ es un arco de retroceso. Entonces $f'((v,u)) = f((v,u)) - b$. La capacidad residual de e es $\overline{c}_e = f((v,u))$ y por tanto
   $c_{(u,v)} \geq f((v,u)) > f'((v,u)) = f((v,u)) - b \geq 0$, ya que $b \leq \overline{c}_e$.

Caminos de aumentación

Demostración (continúa)

2. Conservación del flujo: Consideremos cualquier vértice que no está en el camino P. Como el flujo f' es igual al flujo f en los arcos que entran y que salen de v, la conservación del flujo se cumple para dichos vértices.
   Para los vértices $v \notin \{s, t\}$ que forman parte del camino P tenemos que considerar cuatro posibles casos (FF,FR,RF,RR), según que el camino P llegue al vértice por un arco de avance o uno de retroceso, y que el camino salga de v por un arco de avance o uno de retroceso.

Caminos de aumentación

Demostración (continúa)

Por ejemplo, supongamos el caso FR. Los arcos $e_1 = (u,v)$ de avance y $e_2 = (v,w)$ de retroceso forman parte del camino P. Ningún arco de salida de v en el grafo original cambia, luego $f^{\text{out}}(v) = f'^{\text{out}}(v)$. Pero los arcos de entrada (u,v) y (w,v) sí cambian su flujo. No obstante como $f'((u,v)) = f((u,v)) + b$ y $f'((w,v)) = f((w,v)) - b$, $f^{\text{in}}(v) = f'^{\text{in}}(v)$, de lo cual deducimos que el flujo se conserva en v. Los otros tres casos (FF,RF,RR) se demuestran razonando de manera análoga.

Demostrado que f' es válido y puesto que P necesariamente debe comenzar con un arco de avance que sale de s (G_f no puede tener arcos de retroceso que salgan de s!),

$$v(f') = f'^{\text{out}}(s) = f^{\text{out}}(s) + b = v(f) + b$$

□

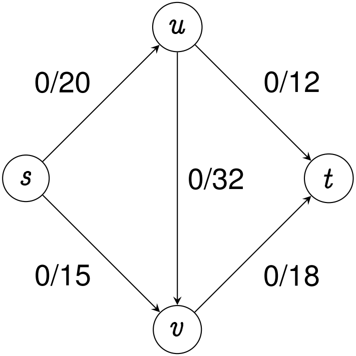
Algoritmo de Ford-Fulkerson (FF)

Require: G una red s-t

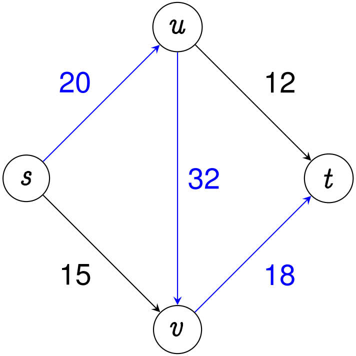
Ensure: $f = \textsc{Ford-Fulkerson}(G)$ es un flujo máximo sobre G

procedure $\textsc{Ford-Fulkerson}(G)$
  for $e \in E(G)$ **do**
    $f(e) := 0$
  end for
  $G_{\mathrm{res}} := G$
  while $\exists$ caminos de aumentación entre s y t en G_{res} **do**
    $P :=$ un camino de aumentación
    $f := \textsc{PushFlow}(P, f)$
    $\textsc{ActualizarResidual}(G_{\mathrm{res}}, P, f)$
  end while
  return f
end procedure

Algoritmo de Ford-Fulkerson (FF)

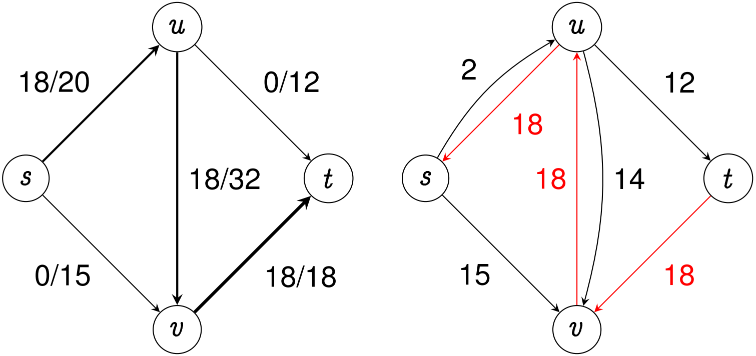


Red



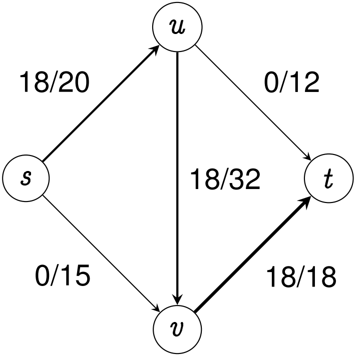
Red residual

Algoritmo de Ford-Fulkerson (FF)

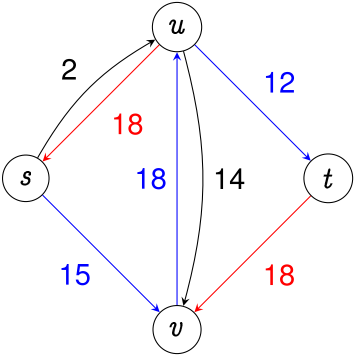


Red Red residual

Algoritmo de Ford-Fulkerson (FF)

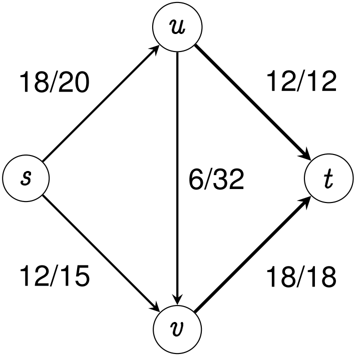


Red

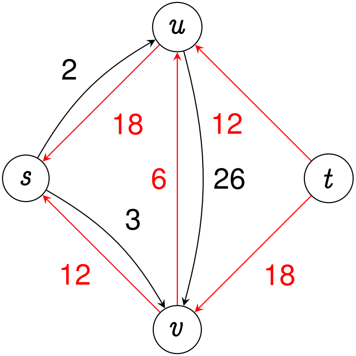


Red residual

Algoritmo de Ford-Fulkerson (FF)



Red



Red residual

Terminación y tiempo de ejecución de FF

En cada iteración del bucle principal del algoritmo el valor del flujo f aumenta al menos en una unidad. Por tanto el número de iteraciones del algoritmo está acotado superiormente por

$$C = \sum_{e=(s,v)} c_e,$$

y la terminación del algoritmo está garantizada.

Cada iteración puede implementarse con coste $\mathcal{O}(m+n)$ ya que el grafo residual nunca tiene más de $2m$ arcos ($m = |E(G)|$) y un camino de aumentación contiene a lo sumo $n = |V(G)|$ vértices.

El coste total del algoritmo es $\mathcal{O}((m+n) \cdot C)$. En caso peor, este coste puede llegar a ser muy elevado; es posible construir redes en las que C sea muy grande, y en cada iteración sólo aumente una unidad el valor del flujo f.

Corrección de FF

El algoritmo de Ford-Fulkerson se termina cuando el grafo residual G_f no contiene un camino de s a t. ¿Porqué eso nos garantiza que el flujo f es entonces máximo?
Sea f_{FF} el flujo hallado por el algoritmo. En el correspondiente grafo residual $G_{f_{FF}}$ no hay ningún camino de s a t. Definamos dos conjuntos de vértices:

$$A_{FF} = \{v \in G_{f_{FF}} \mid v \text{ es accesible desde } s\}$$
$$B_{FF} = \{v \in G_{f_{FF}} \mid v \text{ no es accesible desde } s\}$$

Corrección de FF

El par $\langle A_{FF}, B_{FF}\rangle$ es un corte s-t del grafo G: todo vértice de G pertenece a A_{FF} o a B_{FF}, s pertenece a A_{FF} (porque siempre podemos acceder a s desde s!) y t pertenece a B_{FF}, puesto que $G_{f_{FF}}$ no podemos acceder a t desde s.
Ya hemos visto antes que

$$\begin{aligned} v(f_{FF}) &= f^{\text{out}}(A_{FF}) - f^{\text{in}}(A_{FF}) \\ &= \sum_{e \text{ sale de } A_{FF}} f(e) - \sum_{e \text{ entra } A_{FF}} f(e) \end{aligned}$$

Corrección de FF

Sea e un arco que sale de A_{FF} (sale de un cierto $v \in A_{FF}$ y llega a un vértice $w \in B_{FF}$). No puede existir un arco $e = (v, w)$ en $G_{f_{FF}}$, porque sino tendríamos la posibilidad de acceder a w desde s, lo cual es una contradicción. La única forma de que (v, w) no sea un arco en $G_{f_{FF}}$ es que el flujo sature e, dejando una capacidad residual nula (y por tanto e no aparece en el residual). Así que para todo arco e que sale de A_{FF}, $f(e) = c_e$.

Corrección de FF

Sea $e = (w, v)$ un arco que entra en A_{FF} (sale de un cierto $w \in B_{FF}$ y llega a un cierto $v \in A_{FF}$). Razonando como antes, no puede existir un arco de retroceso $e' = (v, w)$ en el grafo residual $G_{f_{FF}}$, porque sino w sería accessible desde s y por definición w no es accessible. Para que no exista el arco de retroceso $e' = (v, w)$ en $G_{f_{FF}}$ tiene que ocurrir que el arco $e = (w, v)$ del grafo original no lleve flujo, porque si $f(e) > 0$ entonces $e' = (v, w)$ estaría en el residual y su capacidad $\overline{c}_{e'} = f(e)$. Para todo arco e que entra en A_{FF} se tiene que cumplir que $f(e) = 0$.

Corrección de FF

Por lo tanto

$$\begin{aligned} v(f_{FF}) &= \sum_{e \text{ sale de } A_{FF}} f(e) - \sum_{e \text{ entra } A_{FF}} f(e) \\ &= \sum_{e \text{ sale de } A_{FF}} c_e - 0 \\ &= \sum_{e \text{ sale de } A_{FF}} c_e = c(A_{FF}, B_{FF}) \end{aligned}$$

Por un teorema que hemos visto anteriormente ningún flujo puede tener un valor que exceda la capacidad de ningún corte s-t. Como $v(f_{FF})$ coincide con $c(A_{FF}, B_{FF})$ la única conclusión posible es que $v(f_{FF})$ es máximo y que $c(A_{FF}, B_{FF})$ es mínima.
Acabamos de demostrar un importante teorema.

Teorema *MaxFlow-MinCut*

Teorema (MaxFlow-MinCut)

Para toda red s*-*t *con capacidades enteras, existe un flujo máximo* f^* *cuyo valor coincide con la capacidad de un corte* $\langle A^*, B^* \rangle$ *de capacidad mínima.*

$$v(f^*) = c(A^*, B^*)$$

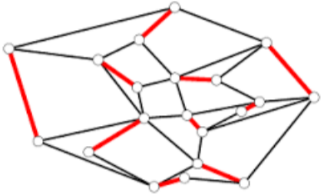
El algoritmo de Ford-Fulkerson nos retorna un flujo f_{FF} *de valor máximo; el corte* $\langle A_{FF}, B_{FF} \rangle$ *tiene capacidad mínima.*

Parte V

Flujos sobre Redes

Problema del Emparejamiento Máximo

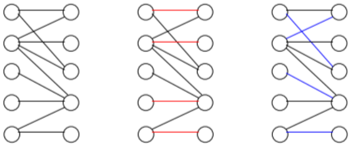
Dado un grafo no dirigido $G = \langle V, E \rangle$ un emparejamiento (eng: *matching*) es un subconjunto $M \subseteq E$ de aristas tales que no hay ningún par de aristas que compartan extremos; dicho de otro modo, todo vértice aparece en a lo sumo una arista de M
El problema del emparejamiento máximo (maximum matching) es hallar un matching de G que tenga máxima cardinalidad.



Maximum matchings en Grafos Bipartidos

Un grafo $G = \langle V, E \rangle$ es bipartido si existe una partición $\langle L, R \rangle$ de V ($L \cup R = V$, $L \cap R = \emptyset$) tal que toda arista $e \in E(G)$ conecta un vértice de L con un vértice de R.

El problema del maximum bipartite matching consiste en hallar un matching de cardinalidad máxima en un grafo bipartido G dado.

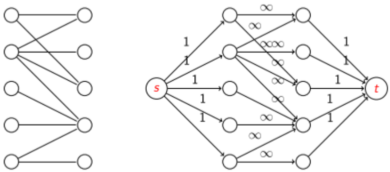


Max matchings = 4

Maximum matchings en Grafos Bipartidos

Dado un grafo bipartido $G = \langle L \cup R, E\rangle$ construiremos la siguiente red s-t $\mathcal{N} = (\hat{V}, \hat{E}, c, s, t)$:

- Añadimos una fuente s y un sumidero t: $\hat{V} = L \cup R \cup \{s, t\}$
- Añadimos arcos entre s y todos los vértices de L con capacidad 1. Añadimos arcos entre todos los vértices de R y t con capacidad 1.
- A toda arista $e = (u, v) \in E$ se le da dirección de L a R y capacidad ≥ 1 (p.e. $c(u, v) = \infty$)
- $\hat{E} = \{(s, u) \mid u \in L\} \cup \vec{E} \cup \{(v, t) \mid v \in R\}$.



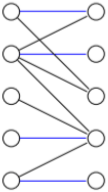
Maximum matchings en Grafos Bipartidos

Teorema

Flujo máximo en $\mathcal{N}$ =Max bipartite matching en G

Demostración

Sea M un matching de G con k aristas. Estas k aristas se ponen en correspondencia biunívoca con k caminos disjuntos en $\mathcal{N}$ y podemos enviar un unidad de flujo a travñes de cada uno de esos k caminos, el valor del flujo es k.



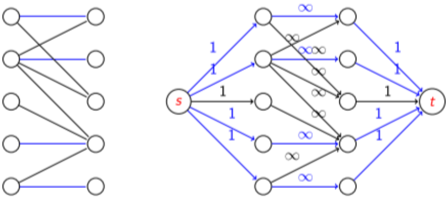
Maximum matchings en Grafos Bipartidos

Teorema

Flujo máximo en $\mathcal{N}$ =Max bipartite matching en G

Demostración

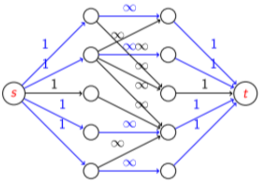
Sea M un matching de G con k aristas. Estas k aristas se ponen en correspondencia biunívoca con k caminos disjuntos en $\mathcal{N}$ y podemos enviar un unidad de flujo a travñes de cada uno de esos k caminos, el valor del flujo es k.



Maximum matchings en Grafos Bipartidos

Demostración (continúa)

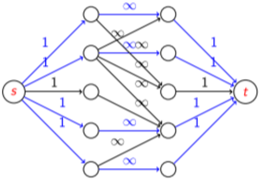
- Consideremos un flujo entero en $\mathcal{N}$. Como todos los arcos que salen de s y los que llegan a t tienen capacidad 1, el flujo en cualquier arco de $\mathcal{N}$ tiene que ser 0 o 1.
- $C = \langle \{s\} \cup L, R \cup \{t\} \rangle$: un s-t corte en $\mathcal{N}$
- Sea M el conjunto de arcos que atraviesan el corte C y llevan flujo 1; entonces $|M| = |f|$.
- Claramente M se corresponde a un emparejamiento en $G = \langle L \cup R, E \rangle$ y su cardinalidad es igual al valor del flujo f en $\mathcal{N}$

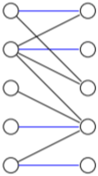


Maximum matchings en Grafos Bipartidos

Demostración (continúa)

- Consideremos un flujo entero en $\mathcal{N}$. Como todos los arcos que salen de s y los que llegan a t tienen capacidad 1, el flujo en cualquier arco de $\mathcal{N}$ tiene que ser 0 o 1.
- $C = \langle \{s\} \cup L, R \cup \{t\} \rangle$: un s-t corte en $\mathcal{N}$
- Sea M el conjunto de arcos que atraviesan el corte C y llevan flujo 1; entonces $|M| = |f|$.
- Claramente M se corresponde a un emparejamiento en $G = \langle L \cup R, E \rangle$ y su cardinalidad es igual al valor del flujo f en $\mathcal{N}$





Maximum matchings en Grafos Bipartidos

Demostración (continúa)

Dado que todas las capacidades de $\mathcal{N}$ son enteros existe un flujo máximo entero f^* y el matching asociado tendrá cardinalidad f^*. En sentido inverso, si M es un matching máximo con k arcos entonces podemos asociar un flujo en $\mathcal{N}$ con valor k. Por lo tanto todo matching máximo en G se corresponde con un flujo máximo en $\mathcal{N}$ y viceversa. $\square$

Maximum matchings en Grafos Bipartidos

El algoritmo consiste por lo tanto en:

1. Construir la red $\mathcal{N}$
2. Aplicar un algoritmo de flujo máximo sobre $\mathcal{N}$
3. Extraer el matching máximo.

Sea $n_L = |L|$, $n_R = |R|$, $n = n_L + n_R = |V|$ y $m = |E|$. El coste es:

1. Crear $n_L + n_R = n$ nuevos arcos y dirigir y dar capacidad a los arcos de E, y añadir dos vértices: $\mathcal{O}(n + m)$
2. El valor de un flujo máximo en $\mathcal{N}$ es $\leq \min\{n_L, n_R\} \leq n$, y el coste de FF sobre $\mathcal{N}$ será $\mathcal{O}((n + m)n) = \mathcal{O}(n^2 + nm)$
3. Basta seleccionar los arcos $(u, v) \in L \times R$ que llevan flujo 1, se hace con coste $\mathcal{O}(m)$

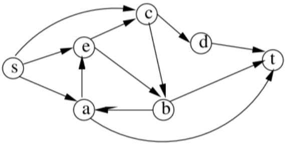
Parte V

Flujos sobre Redes

Problema de los Caminos Disjuntos

Dado un digrafo $G = \langle V, E \rangle$ y un conjunto de caminos se dice que son arco-disjuntos (*disjuntos*, para simplificar) si ningún arco es parte de más de un camino del conjunto. Los caminos pueden compartir vértices.

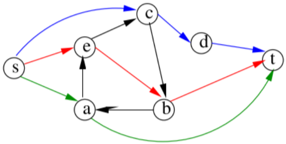
El problema de los caminos disjuntos consiste en dados un digrafo G y dos vértices s y t en V hallar un conjunto de caminos $s \rightsquigarrow t$ arco-disjuntos de máxima cardinalidad.



Problema de los Caminos Disjuntos

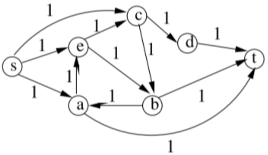
Dado un digrafo $G = \langle V, E \rangle$ y un conjunto de caminos se dice que son arco-disjuntos (*disjuntos*, para simplificar) si ningún arco es parte de más de un camino del conjunto. Los caminos pueden compartir vértices.

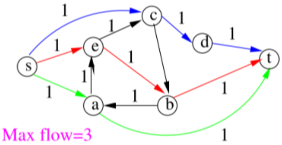
El problema de los caminos disjuntos consiste en dados un digrafo G y dos vértices s y t en V hallar un conjunto de caminos $s \rightsquigarrow t$ arco-disjuntos de máxima cardinalidad.



Problema de los Caminos Disjuntos

En términos de flujos, un camino de s a t podemos verlo como el medio a través del cual transportamos una unidad de flujo. Constuiremos una red s-t $\mathcal{N}$ asignando capacidad unidad a cada arco del digrafo, y convirtiendo los vértices s y t en fuente y sumidero.



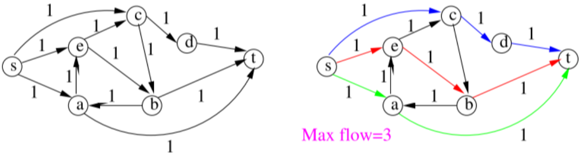


Teorema (Teorema de Menger)

El número máximo de caminos disjuntos $s \rightsquigarrow t$ en G es igual al número de arcos mínimo que atraviesa un s-t corte (= valor del máximo flujo en $\mathcal{N}$)

Problema de los Caminos Disjuntos

En términos de flujos, un camino de s a t podemos verlo como el medio a través del cual transportamos una unidad de flujo. Constuiremos una red s-t $\mathcal{N}$ asignando capacidad unidad a cada arco del digrafo, y convirtiendo los vértices s y t en fuente y sumidero.



Teorema (Teorema de Menger)

El número máximo de caminos disjuntos $s \rightsquigarrow t$ en G es igual al número de arcos mínimo que atraviesa un s-t corte (= valor del máximo flujo en $\mathcal{N}$)

Problema de los Caminos Disjuntos

Demostración

1. El número máximo de caminos disjuntos $s \rightsquigarrow t$ en G es $\leq$ al valor del máximo flujo en $\mathcal{N}$:
   Si tenemos k caminos arco-disjuntos $s \rightsquigarrow t$ en G entonces podemos pasar una unidad de flujo por cada arco e que pertenezca a alguno de los k caminos; ningún arco puede pertenecer a más de un camino, así que se respeta la restricción $f(e) \leq c_e$ para todo arco e. El valor de dicho flujo será $k \leq |f^*|$

Problema de los Caminos Disjuntos

Demostración (continúa)

3. El número máximo de caminos disjuntos $s \rightsquigarrow t$ en G es $\geq$ al valor del máximo flujo en $\mathcal{N}$:
   - Sea f^st un flujo máximo con $|f^*| = k$; como en $\mathcal{N}$ todas las capacidades son unitarias el flujo será entero, para cada arco e tendremos $f(e) = 0$ o $f(e) = 1$.
   - Sea $G' = \langle V, E' \rangle$ el subgrafo G que solo contiene las aristas con flujo $f(e) = 1$
   - Iterativamente calculamos un camino $s \rightsquigarrow t$ y eliminamos sus arcos de G'
   - Esto podrá hacerse k veces; cada uno de los caminos disjuntos que hemos eliminado llevaba una unidad de flujo de s a t.

$\square$

Problema de los Caminos Disjuntos

El algoritmo consiste por lo tanto en:

1. Construir la red $\mathcal{N}$
2. Aplicar un algoritmo de flujo máximo sobre $\mathcal{N}$
3. Extraer los k caminos arco-disjuntos.

Sea $n = |V|$ y $m = |E|$. El coste es:

1. Quitar (o poner capacidad 0) los arcos entrantes a s, los salientes de t y dar capacidad unidad a todos arcos restantes: $\mathcal{O}(n + m)$
2. El valor de un flujo máximo en $\mathcal{N}$ es $\leq n - 1$ (¿por qué?), y el coste de FF sobre $\mathcal{N}$ será $\mathcal{O}((n + m)n) = \mathcal{O}(n^2 + nm)$
3. Eliminar los arcos que no llevan flujo ($\mathcal{O}(m)$) y extraer iterativamente los $|f^*|$ caminos —como en la demostración del teorema de Menger—; extraer un camino tiene coste $\mathcal{O}(n + m)$, de manera que el coste de esta parte también es $\mathcal{O}(|f^*|(n + m)) = \mathcal{O}(n^2 + nm)$

Problema de los Caminos Disjuntos

- Ejercicio #1: obtener el número máximo de caminos vértice-disjuntos $s \rightsquigarrow t$, todos los vértices intermedios tienen que ser distintos
- ¿Número max. de caminos vértice-disjuntos $\leq$ número max. de caminos arco-disjuntos? En caso afirmativo, ¿porqué?
- Ejercicio #2: obtener el número máximo de caminos disjuntos en un grafo no dirigido

Parte V

Flujos sobre Redes

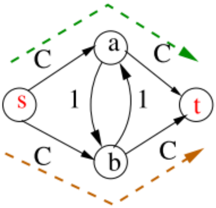
Algoritmo de Edmonds-Karp

J.Edmonds, R. Karp: *Theoretical improvements in algorithmic efficiency for network flow problems*. Journal ACM 1972.

Una elección apropiada de los caminos de aumentación en FF nos da un algoritmo más eficiente! Usar BFS para encontrar los caminos de aumentación en G_f.

Algoritmo de Edmonds-Karp

El algoritmo de Edmonds-Karp es como Ford-Fulkerson pero hay que usar BFS para encontrar los caminos de aumentación de s a t en el grafo residual G_f. Es decir, escoger un camino de aumentación con un número mínimo de arcos.



Algoritmo de Edmonds-Karp

Sea $\mathcal{N} = (V, E, c, s, t)$ una red s-t y f un flujo válido en $\mathcal{N}$. Si G_f admite un camino de aumentación denotaremos por f' el flujo en la siguiente iteración del algoritmo EK.

- El camino de aumentación usado para pasar de f a f' es EK es de longitud mínima entre s y t
- Para todo $v \in V$, $\delta_f(s, v)$ denotará la distancia mínima de s a v en G_f.

Algoritmo de Edmonds-Karp

Es posible que $(u,v) \in E_{f'}$ pero $(u,v) \notin E_f$?

- (u,v) es un arco de avance saturado en f y no en f': luego (v,u) está en el camino de aumentación y hemos "devuelto" flujo a través de él
- (u,v) es un arco de retroceso en $G_{f'}$ pero no en G_f porque $f(v,u) = 0$: luego (v,u) está en el camino de aumentación y hemos pasado flujo

En ambos casos se ha modificado el flujo de v a u y (v,u) forma parte del camino de aumentación que nos lleva de f a f'

Algoritmo de Edmonds-Karp

Lema

Si el algoritmo EK se ejecuta sobre $\mathcal{N} = (V, E, c, s, t)$*, para todo vértice* $v \neq s$*,* $\delta_f(s, v)$ *es una función monótona creciente con cada aumentación.*

Demostración

Por contradicción.
Sea f el primer flujo para el cual existe algún $u \neq s$ tal que

$$\delta_{f'}(s, u) < \delta_f(s, u).$$

Algoritmo de Edmonds-Karp

Demostración (continúa)

Sea v el vértice con mínimo $\delta_{f'}(s, v)$ de entre los que han decrecido su distancia a s.

- Sea $p = s, \ldots, u, v$ un camino de longitud mínima entre s y v en $G_{f'}$
- Por lo tanto $\delta_{f'}(s, v) = \delta_{f'}(s, u) + 1$ y $\delta_{f'}(s, u) \geq \delta_f(s, u)$.
- Si $(u, v) \in E_f$,

$$\delta_f(s, v) \leq \delta_f(s, u) + 1 \leq \delta_{f'}(s, u) + 1 = \delta_{f'}(s, v)!$$

  Por lo tanto $(u, v) \notin E_f$

Algoritmo de Edmonds-Karp

Demostración (continúa)

- $(u, v) \in E_{f'}$ pero $(u, v) \notin E_f$
- Por lo tanto (v, u) aparece en el camino de aumentación escogido De manera que el camino de longitud mínima de s a u en G_f tiene a (v, u) como último arco.

$$\delta_f(s, v) \leq \delta_f(s, u) - 1 \leq \delta_{f'}(s, u) - 1 = \delta_{f'}(s, v) - 1 - 1$$

- En contradicción con la hipótesis: $\delta_{f'}(s, v) < \delta_f(u, v)$.

$\square$

Algoritmo de Edmonds-Karp

Sea P un camino de aumentación en G_f. Diremos que $(u, v) \in P$ es crítico si bottleneck(P) $= c_f(u, v)$. Los arcos críticos de f no aparecen en $G_{f'}$.

- Si (u, v) es de avance, $f'(u, v) = c(u, v)$
- Si (u, v) es de retroceso, $f'(v, u) = 0$

Algoritmo de Edmonds-Karp

Lema

En el algoritmo EK, cada arco puede ser crítico a lo sumo en $|V|/2$ iteraciones.

Demostración

- Sea $(u, v) \in E$, cuando (u, v) es crítico por primera vez, $\delta_f(s, v) = \delta_f(s, u) + 1$
- Tras esa iteración (u, v) desaparece del grafo residual hasta que el flujo en (v, u) cambia
- En ese momento es porque (v, u) forma parte del camino de aumentación en $G_{f'}$, y $\delta_{f'}(s, u) = \delta_{f'}(s, v) + 1$, luego

$$\delta_{f'}(s, u) = \delta_{f'}(s, v) + 1 \geq \delta_f(s, v) + 1 \geq \delta_f(s, u) + 2$$

Algoritmo de Edmonds-Karp

Demostración (continúa)

- Luego la distancia de u se ha incrementado en 2 unidades al menos. Pero $\delta_f(s,u) \leq n$ o $\delta_f(s,u) = +\infty$, de manera que que el arco (u,v) solo puede participar en $n/2$ aumentaciones.

$\square$

Algoritmo de Edmonds-Karp

Teorema

El algoritmo de Edmonds-Karp tiene coste $\mathcal{O}(mn(n+m)) = \mathcal{O}(m^2n)$.

Demostración

- Cada iteración tiene coste $O(m+n)$ para encontrar un camino de aumentación, si lo hay, usando BFS, y para actualizar el grafo residual
- Por el lema anterior solo pueden hacerse $O(mn)$ aumentaciones.

$\square$

Maximización de flujos: Algoritmos

- Ford-Fulkerson (1956): $O(mC)$, donde C es una cota del valor del flujo máximo
- Dinic (1970): (blocking flow) $O(n^2m)$
- Edmonds-Karp (1972): (shortest augmenting path) $O(nm^2)$
- Karzanov (1974): $O(n^2m)$; Goldberg-Tarjant (1986): (push re-label preflow + dynamic trees) $O(nm\lg(n^2/m))$
- King-Rao-Tarjan (1998): $O(nm\log_{m/n\lg n} n)$
- J. Orlin (2013): $O(nm)$ (basado en KRT 1988)

Parte V

Flujos sobre Redes

Problemas de Asignación Generalizados

- Consideremos un problema de asignación generalizado $\mathcal{GP}$ donde se nos dan como entrada d conjuntos finitos, cada uno representando un conjunto distinto de recursos (máquinas, personas, localizaciones, tiempos, medios de transporte, ...)
- Nuestro objetivo es escoger el mayor número posible de d-tuplas $(x_1, \ldots, x_d)$ donde $x_i \in X_i$ y tales que satisfacen una serie de restricciones:
  - Para toda i, $1 \leq i \leq d$ el elemento $x_i \in X_i$ puede aparecer en a lo sumo $c(x_i)$ de las tuplas escogidas.
  - Para toda i, $1 \leq i < d$, y cualesquiera $x \in X_i$ e $y \in X_{i+1}$ el par (x, y) puede aparecer a lo sumo en $c(x, y)$ de las tuplas escogidas.
  - Los valors de $c(x)$ y $c(x, y)$ son enteros o $+\infty$.
- Observación: solo los pares de objetos x e y de conjuntos X_i y X_{i+1} consecutivos están sujetos a posibles restricciones ($c(x, y) = +\infty$ cuando no lo están!)

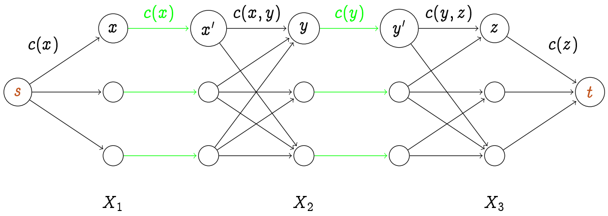
Problemas de Asignación Generalizados

Podemos reducir el problema $\mathcal{GP}$ a uno de máximo flujo construyendo la siguiente red $\mathcal{N}$:

- V contiene un vértice x para cada elemento x de cada conjunto X_i, $1 \leq i \leq d$; además contendrá una copia x' de cada elemento $x \in X_i$, para $1 \leq i < d$, y finalmente los vértices s y t.
- Añadiremos un arco (s, x) para todo $x \in X_1$ y un arco (y, t) para todo $y \in X_d$. Estos arcos tendrán capacidades $c(s, x) = c(x)$ y $c(y, t) = c(y)$, respectivamente.
- Añadiremos un arco (x', y) para todo par $x \in X_i$ e $y \in X_{i+1}$. La capacidad de (x', y) será $c_{(x',y)} = c(x, y)$.
- Todo arco de capacidad 0 se omite.
- Para todo $x \in X_i$, $1 \leq i < d$, se añade el arco (x, x') con capacidad $c_{(x,x')} = c(x)$.

Todo camino $s \rightsquigarrow t$ en $\mathcal{N}$ identifica una d-tupla factible; a la inversa toda d-tupla factible determina un camino $s \rightsquigarrow t$.

Problemas de Asignación Generalizados



- Para resolver $\mathcal{GP}$, se construye $\mathcal{N}$ y se busca un flujo f^* de valor máximo en $\mathcal{N}$
- El subgrafo formado por los arcos e con $f^*(e) > 0$, buscamos un camino $s \rightsquigarrow t$, representando una d-tupla válida, se decrementa el flujo en los arcos del camino y se eliminan los arcos donde el flujo es 0
- El procedimiento se repite $|f^*|$ veces, de manera que se obtiene un conjunto de d-tuplas factibles (satisfacen todas las restricciones) de cardinalidad máxima.

Problemas de Asignación Generalizados

Ejemplo

1. En una facultad se imparten n cursos y para cada uno de ellos ha de hacerse un examen parcial. El curso C_i, $1 \leq i \leq n$, tiene e_i estudiantes matriculados.
2. Se dispone de r aulas para realizar los examenes. Cada examen se ha de hacer en una única aula. El aula A_j, $1 \leq j \leq r$, tiene capacidad para un máximo de s_j estudiantes.
3. La semana de examenes parciales está dividida en τ slots de tiempo. Se ha de asignar un slot de tiempo y un aula a cada examen. No puede haber dos examenes en el mismo slot de tiempo y en la misma aula.

Problemas de Asignación Generalizados

Ejemplo

4. Para cada examen parcial necesitamos un profesor que lo vigile.
5. Cada profesor tiene unas restricciones de disponibilidad: $D[\ell, k] =$ **true** ssi el profesor P_ℓ, $1 \leq \ell \leq p$ está disponible en el slot de tiempo T_k, $1 \leq k \leq \tau$. Ningún profesor puede vigilar más de 6 exámenes.

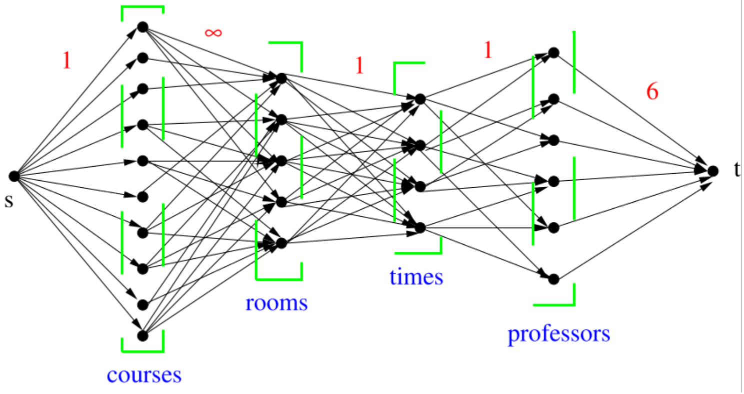
Nuestro objetivo: diseñar un algoritmo eficiente que nos dé una planificación válida que asigna a cada examen (=curso) un aula, un slot de tiempo y un vigilante, respetando todas las restricciones o bien que reporte que no existe tal planificación.

Problemas de Asignación Generalizados

Construimos la siguente red $\mathcal{N}$:

- Vértices: $V = \{s, t, \{C_i\}, \{A_j\}, \{T_k\}, \{P_\ell\}\}$; un total de $2 + n + r + \tau + p$ vértices
- Arcos y capacidades:
  - (s, C_i) con capacity $1 \leftarrow$ cada curso tiene un examen
  - (C_i, A_j), si $e_i \leq s_j$, con capacidad $+\infty \leftarrow$ los estudiantes del curso C_i caben en el aula A_j
  - Para toda j y k, (A_j, T_k), con capacidad $1 \leftarrow$ solo un examen por aula en un slot de tiempo dado
  - Para toda k y ℓ, (T_k, P_ℓ), si $D[\ell, k] =$ **true** con capacidad 1 $\leftarrow P_\ell$ puede vigilar un parcial en el slot T_k si está disponible, pero solo un parcial
  - (P_ℓ, t), con capacity $6 \leftarrow$ cada profesor puede vigilar 6 examenes como mucho

Problemas de Asignación Generalizados



Problemas de Asignación Generalizados

1. El tamaño del problema es $N = n + r + \tau + p$; el tamaño de $\mathcal{N}$ es $N + 2 = \mathcal{O}(N)$ vértices y $\mathcal{O}(N^2)$ aristas $\leftarrow$ ¿porqué?
2. Todo camino $s \rightsquigarrow t$ nos proporciona una asignación aula-tiempo-vigilante para un examen parcial (el del curso C_i correspondiente); análogamente cualquier asignación (C_i, A_j, T_k, P_ℓ) se representa mediante un camino $s \rightsquigarrow t$ en $\mathcal{N}$
3. Todo flujo entero f de valor $|f|$ identifica un conjunto de caminos entre s y t que a su vez representa una asignación válida para $|f|$ examenes parciales, y viceversa.

Problemas de Asignación Generalizados

1. Queremos tener asignaciones para todos los n examenes parciales: por lo tanto neecsitamos saber si $\mathcal{N}$ admite un flujo de valor n; para ello calculamos el flujo máximo f^* en $\mathcal{N}$
2. Si $|f^*| = n$ entonces podremos planificar todos los examens parciales, en caso contrario no $\leftarrow$ $|f^*| \leq n$ necesariamente, ¿porqué?
3. Para recobrar la planificación extraemos n caminos de manera iterativa considerando el subgrafo con arcos en los que hay flujo $f(e) > 0$

Problemas de Asignación Generalizados

Coste del algoritmo:

- Para construir $\mathcal{N}$, el coste es $\mathcal{O}(N^2)$.
- El coste de aplicar Ford-Fulkerson para calcular f^* es $\mathcal{O}((N + N^2)n) = \mathcal{O}(nN^2)$
- La recuperación de las n asignaciones también necesita coste $\mathcal{O}((N + N^2)n) = \mathcal{O}(nN^2)$
- En total: $\mathcal{O}(nN^2) = \mathcal{O}(n(n + r + \tau + p)^2)$

Parte V

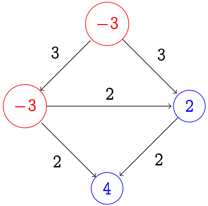
Flujos sobre Redes

Circulación con demandas

- Un nuevo tipo de problema de flujo en redes: hay suministro u oferta y hay demanda
- En vez de un par fuente/sumidero, consideramos un escenario con vértices productores y vértices consumidores
- Habrá nodos que pueden producir flujo y nods que pueden consumir flujo
- La cuestión básica es si todo el flujo producido podrá ser enrutado y consumido; una asignación de flujo qu elo hace posible se denomina una circulación

Circulación con demandas

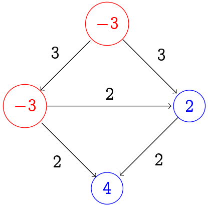
Una red con demandas $\mathcal{N}$ es una tupla $\langle V, E, c, d\rangle$ donde $c : E \to \mathbb{R}^+$ son capacidades positivas asignadas a los arcos y $d : V \to \mathbb{R}$ asocia a cada vértice su demanda $d(v)$.



- Si $d(v) > 0$ significa que v debe recibir $d(v)$ unidades de flujo que consumirá, el resto del flujo recibido debe fluir hacia sus vértices sucesores; un vértice con $d(v) > 0$ se denomina sumidero
- Si $d(v) < 0$ significa que v produce $d(v)$ unidades de flujo, que junto a las recibidas de sus predecesores fluyen hacia los vértices sucesores; un vértice con $d(v) < 0$ se denomina fuente

Circulación con demandas

Una red con demandas $\mathcal{N}$ es una tupla $\langle V, E, c, d \rangle$ donde $c : E \to \mathbb{R}^+$ son capacidades positivas asignadas a los arcos y $d : V \to \mathbb{R}$ asocia a cada vértice su demanda $d(v)$.



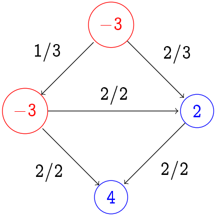
- Los vértices con $d(v) = 0$ no son fuentes ni sumideros, simplemente enrutan flujo y deben cumplir la condición de conservación del flujo $fin(v) = fout(v)$
- $S = \{v \mid d(v) < 0\}, T = \{v \mid d(v) > 0\}$

Circulación con demandas

Dada un red con demandas $\mathcal{N} = \langle V, E, c, d \rangle$, una circulación en $\mathcal{N}$ es una asignación de flujo $f : E \to R^+$ tal que

1. Condiciones de capacidad: para todo arco $e \in E$, $0 \leq f(e) \leq c(e)$
2. Condiciones de circulación: para todo $v \in V$,

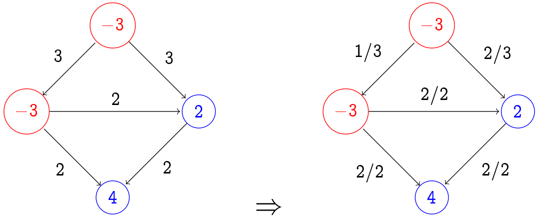
$$\sum_{(u,v)\in E} f(u,v) - \sum_{(v,w)\in E} f(v,w) = d(v)$$



N.B. Una circulación puede **no** existir

Circulación con demandas

Problema de la circulación: Determinar si para $\mathcal{N} = \langle V, E, c, d \rangle$ existe una circulación, y en su caso, obtener una.



Circulación con demandas

Si f es una circulación en $\mathcal{N} = \langle V, E, c, d\rangle$ entonces

$$\sum_{v \in V} d(v) = \sum_{v \in V} \left(\underbrace{\sum_{(u,v) \in E} f(u,v)}_{\text{arcos entrantes a } v} - \underbrace{\sum_{(v,w) \in E} f(v,w)}_{\text{arcos salientes de } v} \right).$$

Para todo $e = (u, v) \in E$, $f(e)$ aparece en la suma de los arcos entrantes a v y en la suma de los arcos saliente de u, luego sus dos contribuciones se cancelan en la suma de arriba!

Por lo tanto $\sum_{v \in V} d(v) = 0$.

Circulación con demandas

Si f es una circulación en $\mathcal{N} = \langle V, E, c, d \rangle$ entonces

$$\sum_{v \in V} d(v) = \sum_{v \in V} \left(\underbrace{\sum_{(u,v) \in E} f(u,v)}_{\text{arcos entrantes a } v} - \underbrace{\sum_{(v,w) \in E} f(v,w)}_{\text{arcos salientes de } v} \right).$$

Para todo $e = (u, v) \in E$, $f(e)$ aparece en la suma de los arcos entrantes a v y en la suma de los arcos saliente de u, luego sus dos contribuciones se cancelan en la suma de arriba!

Por lo tanto $\sum_{v \in V} d(v) = 0$.

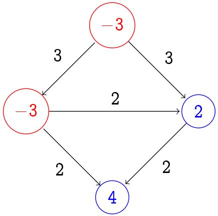
Circulación con demandas

Para que $\mathcal{N}$ tenga una circulación es condición necesaria que $\sum_{v \in V} d(v) = 0$.

Recordemos:
$S = \{v \in V | d(v) < 0\}$ y
$T = \{v \in V | d(v) > 0\}$.

Definamos
$D = -\sum_{v \in S} d(v) = \sum_{v \in T} d(v)$.

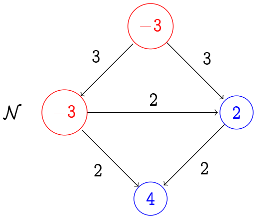
D es la cantidad total extra producida por las fuentes (S) que debe ser enrutada para llegar y ser consumida por los sumideros (T)



Reducción a Max-Flow

Dada $\mathcal{N} = \langle V, E, c, d \rangle$, definimos una red de flujo s–t $\mathcal{N}' = \langle V', E', c', s, t \rangle$ como sigue:

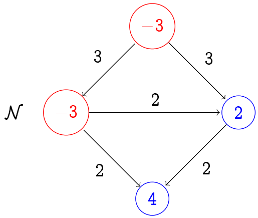
- $V' = V \cup \{s, t\}$, añadimos una fuente s y un sumidero t
- Para todo $v \in S$ $(d(v) < 0)$, se añade el arco (s, v) con capacidad $-d(v)$.
- Para todo $v \in T$ $(d(v) > 0)$, se añade (v, t) con capacidad $d(v)$.
- En E' tendremos todos los arcos de E; si $e \in E$, su capacidad en $\mathcal{N}'$ es la original: $c'(e) = c(e)$.

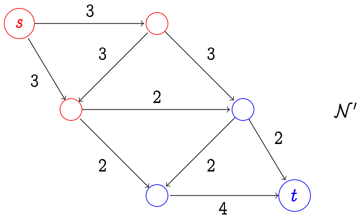


Reducción a Max-Flow

Dada $\mathcal{N} = \langle V, E, c, d\rangle$, definimos una red de flujo s–t $\mathcal{N}' = \langle V', E', c', s, t\rangle$ como sigue:

- $V' = V \cup \{s, t\}$, añadimos una fuente s y un sumidero t
- Para todo $v \in S$ ($d(v) < 0$), se añade el arco (s, v) con capacidad $-d(v)$.
- Para todo $v \in T$ ($d(v) > 0$), se añade (v, t) con capacidad $d(v)$.
- En E' tendremos todos los arcos de E; si $e \in E$, su capacidad en $\mathcal{N}'$ es la original: $c'(e) = c(e)$.





Reducción a Max-Flow

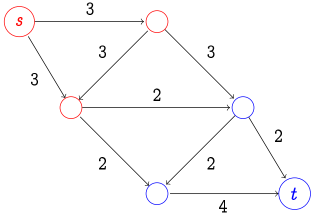
Observación #1: todo flujo f' en $\mathcal{N}'$ satisface $|f'| \leq D$.

La capacidad del corte $c'(\{s\}, V \cup \{t\}) = D$, por tanto $|f'| \leq D$.

$\mathcal{N}'$

Reducción a Max-Flow

Observación #1: todo flujo f' en $\mathcal{N}'$ satisface $|f'| \leq D$.
La capacidad del corte $c'(\{s\}, V \cup \{t\}) = D$, por tanto $|f'| \leq D$.



$\mathcal{N}'$

Reducción a Max-Flow

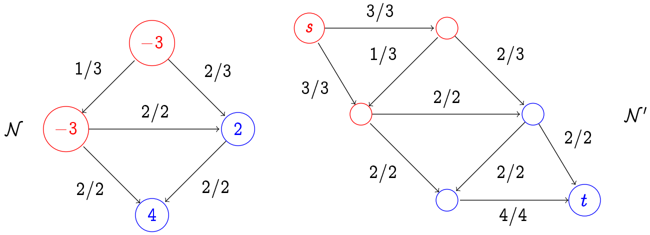
Observación #2: si hay una circulación f en $\mathcal{N}$ entonces el valor del flujo máximo f^* en $\mathcal{N}'$ es $|f^*| = D$.

Demostración

Convertimos la circulación f en un flujo f^* haciendo que el flujo producido por las fuentes de S sea flujo originado en s en $\mathcal{N}'$: $f^*(s, v) = -d(v)$ para toda $v \in S$; análogamente el flujo consumido en los sumideros de T será flujo absorbido en t en $\mathcal{N}'$: $f^*(v, t) = d(v)$.

Por las condiciones de capacidad y de circulación f^* es un flujo válido en $\mathcal{N}'$ y puesto que el valor de f^* es D, se deduce que f^* es un flujo máximo. $\square$

Reducción a Max-Flow



Reducción a Max-Flow

Obervación #3: si existe un flujo máximo f^* en $\mathcal{N}$ tal que $|f'| = D$, entonces $\mathcal{N}$ tiene una circulación

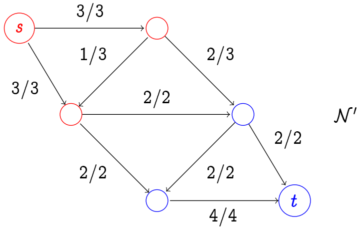
Demostración

Para todo arco $e \in E$, fijamos $f(e) = f^*(e)$.

- Puesto que $|f^*| = D$, todos los arcos $(s, v) \in E'$ y $(u, t) \in E'$ tienen que ser saturados por f^*.
- Por la conservación del flujo, f satisface
$$d(v) = \underbrace{\sum_{(u,v)\in E} f(u, v)}_{\text{arcos entrates en } v} - \underbrace{\sum_{(v,w)\in E} f(v, w)}_{\text{arcos salientes de } v}$$
- Luego f es una circulación para $\mathcal{N}$.

$\square$

Reducción a Max-Flow



Circulaciones

Teorema (Condición necesaria y suficiente)

La red con demandas $\mathcal{N} = \langle V, E, c, d \rangle$ *admite una circulación si y solo si* $\mathcal{N}'$ *admite un flujo máximo de valor* D,

$$D = \sum_{v : d(v) > 0} d(v) = \sum_{v : d(v) < 0} -d(v)$$

Teorema (Integridad de las circulaciones)

Si todas las capacidades y demandas en $\mathcal{N}$ *son enteras y si* $\mathcal{N}$ *admite una circulación entonces existe una circulación* f *para* $\mathcal{N}$ *tal que* $f(e) \in \mathbb{Z}$ *para todo arco* e

Demostración

Observaciones #1 a #3 + integralidad del flujo máximo si las capacidades son enteras. □

Circulaciones

Teorema (Condición necesaria y suficiente)

La red con demandas $\mathcal{N} = \langle V, E, c, d\rangle$ admite una circulación si y solo si $\mathcal{N}'$ admite un flujo máximo de valor D,

$$D = \sum_{v:d(v)>0} d(v) = \sum_{v:d(v)<0} -d(v)$$

Teorema (Integridad de las circulaciones)

Si todas las capacidades y demandas en $\mathcal{N}$ son enteras y si $\mathcal{N}$ admite una circulación entonces existe una circulación f para $\mathcal{N}$ tal que $f(e) \in \mathbb{Z}$ para todo arco e

Demostración

Observaciones #1 a #3 + integralidad del flujo máximo si las capacidades son enteras. $\square$

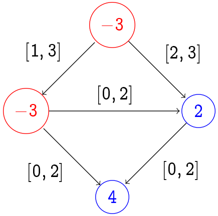
Circulaciones

Teorema

El problema de la circulación puede resolverse en tiempo polinómico. Si demandas y capacidades son enteras, el problema se puede resolver en tiempo $\mathcal{O}(m \cdot D)$ *usando FF.*

Redes con demandas y cotas inferiores

Generalizaremos el problema anterior: además de las demandas (de producción y de consumo) podremos poner restricciones de flujo mínimo en ciertos arcos.
Para cada arco e tendremos un nuevo valr asociado $\ell(e)$ que indica que el flujo en dicho arcos ha de ser como mínimo $\ell(e)$
Una red con demandas y cotas inferiores $\mathcal{N}$ es una tupla $\langle V, E, c, \ell, d\rangle$ tal que $c(e) \geq \ell(e) \geq 0$, para todo arco $e \in E$

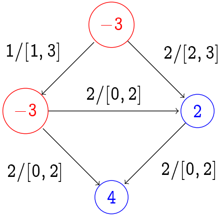


Redes con demandas y cotas inferiores

Dada una red $\mathcal{N} = \langle V, E, c, \ell, d \rangle$ una circulación es una asignación de flujo $f : E \to R^+$ tal que

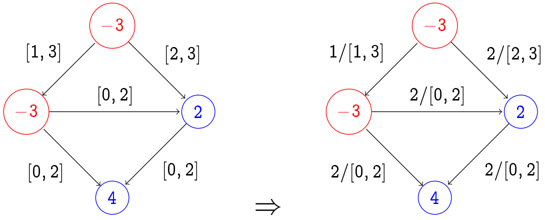
1. Capacidad: para todo $e \in E$, $\ell(e) \leq f(e) \leq c(e)$
2. Circulación: para todo $v \in V$,

$$\sum_{(u,v) \in E} f(u,v) - \sum_{(v,w) \in E} f(v,w) = d(v).$$



Redes con demandas y cotas inferiores

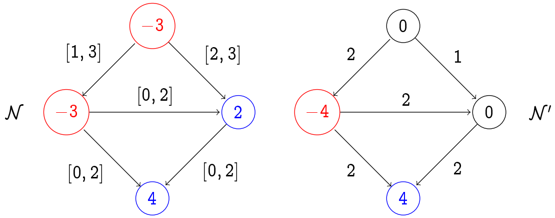
Problema de la circulación con demandas y cotas inferiores:
Dado $\mathcal{N} = \langle V, E, c, \ell, d \rangle$, obtene una circulación f para $\mathcal{N}$, si existe:



Reducción a circulaciones con demandas

Sea $\mathcal{N} = \langle V, E, c, \ell, d \rangle$, construímos una nueva red $\mathcal{N}' = \langle V, E, c', d' \rangle$ donde solo hay demandas (y no cotas inferiores ℓ):

1. Fijar $c' = c$ and $d' = d$ inicialmente.
2. Para todo arco $e = (u, v) \in E$, tal que $\ell(e) > 0$:
   - $c'(e) := c(e) - \ell(e)$.
   - Actualizar las demandas en ambos extremos de e: $d'(u) := d(u) + \ell(e)$ y $d'(v) := d(v) - \ell(e)$

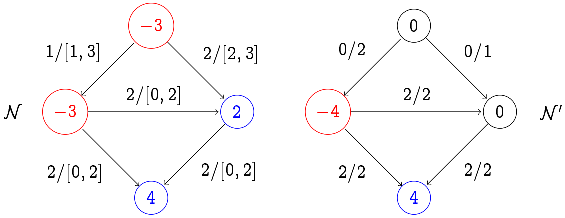


Reducción a circulaciones con demandas

1. Si f es una circulación en $\mathcal{N}$ entonces $f'(e) = f(e) - \ell(e)$, para todo $e \in E$, es una circulación en $\mathcal{N}'$.

Demostración

Por construcción de $\mathcal{N}'$, f' verifica la condición de capacidad.
Además si (u, v) tiene la cota fija $\ell(u, v) > 0$, el flujo de salida de u y el de entrada en v se decrementan en $\ell(u, v)$.
f es una circulación en $\mathcal{N}$ luego el balance $f^{\text{in}}(v) - f^{\text{out}}(v)$ de f' es d' en cada nodo. $\square$

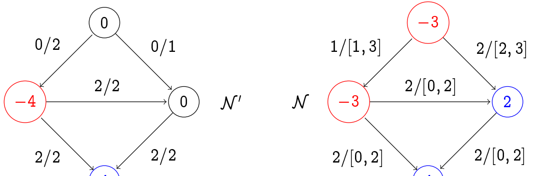


Reducción a circulaciones con demandas

2. Si f' es una circulación en $\mathcal{N}'$, entonces $f(e) = f'(e) + \ell(e)$, para todo $e \in E$, es una circulación en $\mathcal{N}$.

Demostración

f' verifica las condiciones de capacidad: $f'(e) \geq 0$, luego $f(e) \geq \ell(e)$. Como f' es una circulación el balance $f^{\text{in}}(u) - f^{\text{out}}(u)$ de f' en u es $d'(u)$.
Sea (u, v) un arco $\ell(u, v) > 0$, entonces incrementamos el flujo en (u, v) para compensar $\ell(u, v)$ unidades de flujo que salen de u con $\ell(u, v)$ unides de flujo recibidas en v. Y el balance final $f^{\text{in}}(u) - f^{\text{out}}(u)$ de f será $d(u) = d'(u) - \ell(u, v)$. □



Circulaciones con demandas y cotas inferiores

Teorema

$\mathcal{N}$ admite una circulación con demandas y cotas inferiores si y solo si $\mathcal{N}'$ admite una circulación con demandas.

Teorema

Si todas las demandas, capacidades y cotas inferiores son enteros entonces existe una circulación válida para $\mathcal{N}$ tal que todos los flujos $f(e)$ son enteros.

Circulaciones con demandas y cotas inferiores

Teorema

La circulación en la red con demandas y cotas inferiores, si existe, puede obtenerse en tiempo polinómico. Si c, d y ℓ toman valores enteros la circulación tomará valores enteros y se puede obtenr con FF en tiempo $\mathcal{O}((D + L)m)$, donde $D = \sum_{v:d(v)>0} d(v) = \sum_{v:d(v)<0} -d(v)$ es la demanda de consumo total o la producción total, y $L = \sum_{e \in E} \ell(e)$

Parte V

Flujos sobre Redes

Reducciones: Ejemplos Adicionales