Ford-Fulkerson

Daniel Penazzi

29 de marzo de 2023

Tabla de Contenidos

Complejidad de Greedy

- Al final de la clase anterior habiamos visto el algoritmo Greedy para encontrar flujos maximales.
- Si bien vimos que no siempre encuentra un flujo maximal, al menos es polinomial:
  1. En cada camino dirigido que Greeedy usa, al menos un lado se satura.
  2. Como en Greedy los lados nunca se des-saturan, entonces Greedy puede hacer a lo sumo $O(m)$ incrementos de flujo antes de que forzosamente deba terminar si o si.
  3. Encontrar un camino dirigido no saturado es $O(m)$ (ya sea usando DFS o BFS)
  4. Asi que la complejidad total de Greedy es $O(m^2)$.

Greedy

- Si bien vimos que Greedy no siempre da un flujo maximal, dije que se podia modificar Greedy para lograr eso, generalizando el concepto de camino dirigido no saturado.
- En el ejemplo que vimos, que era chico, podiamos ver “a ojo”lo que habria que corregir pero en un caso cuando hayamos creado miles de caminos aumentantes no es posible examinar cada uno a ver si estaba bien o mal.
- Asi que la idea no es esa.
- Para motivar la idea, imaginemos como antes que nuestro network representa una red de cañerias de agua por la cual estamos queriendo mandar agua desde s a t.

Greedy

- Pero ademas ahora supongamos que en cada vértice $x \neq s, t$ hay un operador, que controla cuanto flujo de agua puede mandar desde x hacia $\Gamma^+(x)$.
- Cuando estamos queriendo aumentar el flujo usando Greedy, podemos modelar la busqueda de un camino dirigido de la siguiente forma:
  - s le pregunta a alguno de sus vecinos x de $\Gamma^+(s)$ con lado (s, x) no saturado si le puede mandar flujo.
  - Ese vértice x le pregunta a alguno de sus vecinos de su Γ^+ si le puede mandar flujo,
  - etc, hasta llegar a t

Generalización de caminos dirigidos no saturados

- Cada vértice debe “preguntarle” al siguiente si le puede mandar flujo, pues un vértice $\neq s, t$ no puede aceptar que le manden flujo sin saber si se lo puede “sacar” de encima.
- Por ejemplo s le pregunta a A si le puede mandar flujo, A se fija que le puede mandar flujo a B, asi que le pregunta, este se fija que le puede mandar flujo a t, asi que le responde que si, A le dice que si a s, y se crea el camino $sABt$.
- Pero podria pasar que B no pueda mandar flujo a t por estar el lado Bt saturado , y que B no tenga mas vecinos.
- En Greedy, B le va a decir a A que no puede recibir mas flujo.
- Pero supongamos que la razón por la cual (B, t) esta saturado es que B está recibiendo flujo de un vértice C.

Caminos dirigidos no saturados

- Entonces B, antes de decirle que no a A podria pensar: “yo podria recibir flujo de A y mandarlo a t si C no me mandara flujo, o al menos no tanto”.
- Y entonces B preguntarle a C si no hay alguna posibilidad que C en vez de mandarle flujo a B se lo mande a otro vértice.
- Supongamos que si, que C tiene un vecino D en $\Gamma^+(C)$ al cual le puede mandar flujo.
- Le pregunta a D si puede mandarle flujo, y supongamos que D puede llegar a t, le dice que si a C, C le dice que si a B, B le dice que si a A y listo.
- Lo que queda es la sucesión de vértices s, A, B, C, D, t pero esa sucesión NO ES un camino dirigido, porque (B, C) no es un lado, el que es lado es (C, B).
- Un camino asi se llama “camino aumentante”(augmenting path).

Caminos aumentantes

- En un camino dirigido, “mandar”ϵ unidades de flujo significaba aumentar el flujo en cada lado en ϵ.
- Pero fijemosnos que aca no es eso lo que ocurre:
  - A le va a mandar flujo a B, C le va a mandar flujo a D y D a t...
  - pero B no le manda flujo a C, sino que C deja de mandarle flujo a B (o le manda menos).
- Entonces en algunos lados vamos a DISMINUIR el flujo en vez de aumentarlo.

Idea de FF

- Esa es la base para la idea de Ford y Fulkerson:
- Al pararnos en un vértice x, en vez de limitar la busqueda a vértices a los cuales x les pueda "mandar" flujo.
  - Permitir que x tambien busque vértices a los cuales les pueda pedir que no le manden mas flujo, o al menos no tanto flujo como le estan mandando.
- Claro que para pedirle a alguien que no te mande mas flujo, ese "alguien" te tiene que haber mandado flujo.
- Entonces, técnicamente, lo que Ford y Fulkerson hacen es que:
- en vez de limitar la busqueda a $y \in \Gamma^+(x)$ con $f(\overrightarrow{xy}) < c(\overrightarrow{xy})$
- permiten ademas buscar $y \in \Gamma^-(x)$ con $f(\overrightarrow{yx}) > 0$

Camino aumentante

Definición:

Un camino aumentante (o f-camino aumentante si necesitamos especificar f) o camino de Ford-Fulkerson, es una sucesión de vértices $x_0, x_1, ..., x_r$ tales que:

- $x_0 = s$, $x_r = t$.
- Para cada $i = 0, ..., r-1$ ocurre una de las dos cosas siguientes:
  1. $\overrightarrow{x_i x_{i+1}} \in E$ y $f(\overrightarrow{x_i x_{i+1}}) < c(\overrightarrow{x_i x_{i+1}})$ o:
  2. $\overrightarrow{x_{i+1} x_i} \in E$ y $f(\overrightarrow{x_{i+1} x_i}) > 0$.

Si en vez de comenzar en s y terminar t el camino es como arriba pero con $x_0 = x$,$x_r = z$ diremos que es un camino aumentante desde x a z

Lados forward y backward

- A los lados en 1) los llamaremos
  - "lados de tipo I" o "lados forward"
- A los lados en 2) los llamaremos
  - "lados de tipo II" o "lados backward"
- Observemos que el orden en que se listan los vértices en el camino aumentante es $...x_i, x_{i+1}...$ pero como JUSTAMENTE estamos generalizando la noción de camino dirigido, no siempre (x_i, x_{i+1}) será un lado, sino que el lado será $\overrightarrow{x_{i+1}x_i}$ en el caso de los lados backward.

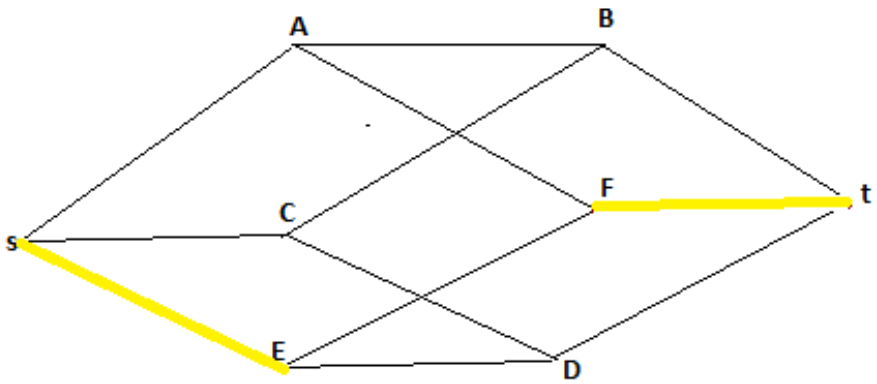
Ford-Fulkerson

Algoritmo de Ford-Fulkerson para hallar flujo maximal

1. Comenzar con $f = 0$ (es decir, $f(\overrightarrow{xy}) = 0 \forall \overrightarrow{xy} \in E$).
2. Buscar un f-camino aumentante $s = x_0, x_1, ..., x_r = t$.
3. Definir ε_i de la siguiente manera:
   - $\varepsilon_i = c(\overrightarrow{x_i x_{i+1}}) - f(\overrightarrow{x_i x_{i+1}})$ en los lados forward.
   - $\varepsilon_i = f(\overrightarrow{x_{i+1} x_i})$ en los lados backward.
4. Calcular $\varepsilon = \min\{\varepsilon_i\}$.
5. Cambiar f a lo largo del camino de [2] en ε, de la siguiente forma:
   - $f(\overrightarrow{x_i x_{i+1}}) += \varepsilon$ en los lados forward.
   - $f(\overrightarrow{x_{i+1} x_i}) -= \varepsilon$ en los lados backwards.
6. Repetir [2] hasta que no se puedan hallar mas caminos aumentantes.

Ej

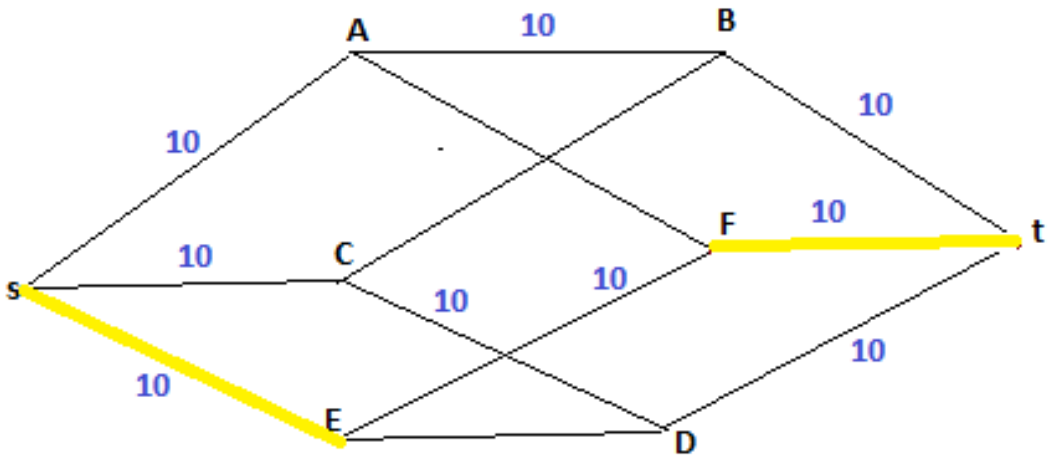
- Recordemos uno de los ejemplos que vimos.



- Capacidades todas 10 excepto los lados marcados de distinto color, de capacidad 20.

Ejemplo de Greedy

- Greedy aumentaba $f = 0$ por medio de los siguientes caminos, en sucesión:
- sABt:10 sCDt:10 sEFt:10 y quedaba:



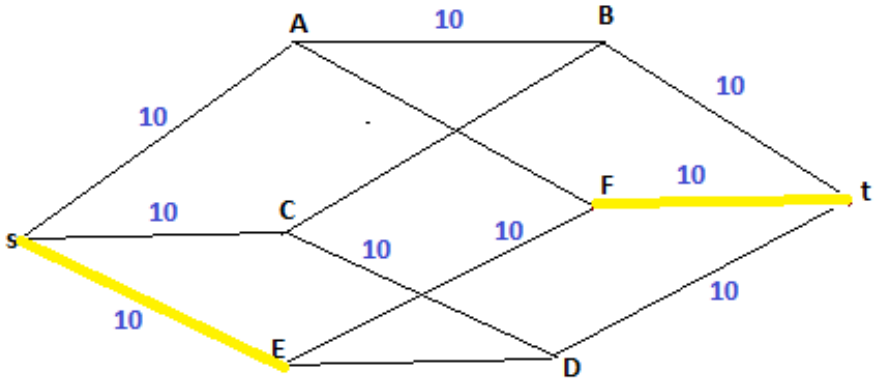
- Y no podiamos seguir, pero no era maximal porque vimos que habia un flujo con valor 40.

Ejemplo de FF

- Con FF, luego de sABt:10, sCDt:10,sEFt:10
- Podemos continuar con el proceso pues ahora podemos usar caminos aumentantes en vez de solo dirigidos.
- Como $f(\overrightarrow{st}) = 10 < 20 = c(\overrightarrow{st})$ entonces s puede mandar flujo a E
- Como $f(\overrightarrow{ED}) = 0 < 10 = c(\overrightarrow{ED})$ entonces E puede mandarle flujo a D.
- $f(\overrightarrow{Dt}) = 10 = c(\overrightarrow{Dt})$ asi que D no puede mandar flujo a t.
- Pero $f(\overrightarrow{CD}) > 0$ asi que D puede pedirle a C que no le mande flujo.

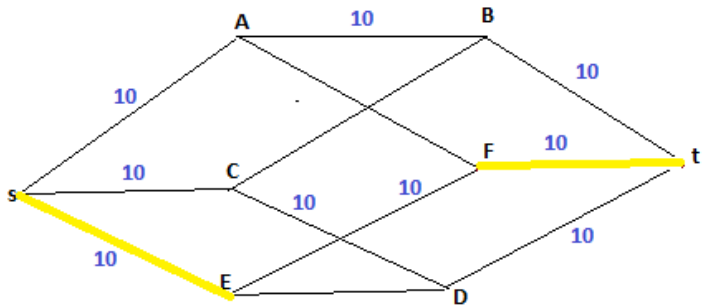
Ej de FF

- Como $f(\overrightarrow{CB}) < c(\overrightarrow{CB})$
- entonces se puede seguir la busqueda desde C a B.
- $f(\overrightarrow{Bt}) = 10 = c(\overrightarrow{Bt})$ pero $f(\overrightarrow{AB}) > 0$ asi que B puede agregar a A a la busqueda, como “backward”.



Ej de FF

- $f(\overrightarrow{AF}) = 0 < 10 = c(\overrightarrow{AF})$ asi que podemos ir de A a F
- y $f(\overrightarrow{Ft}) = 10 < 20 = c(\overrightarrow{Ft})$ y podemos completar el camino.



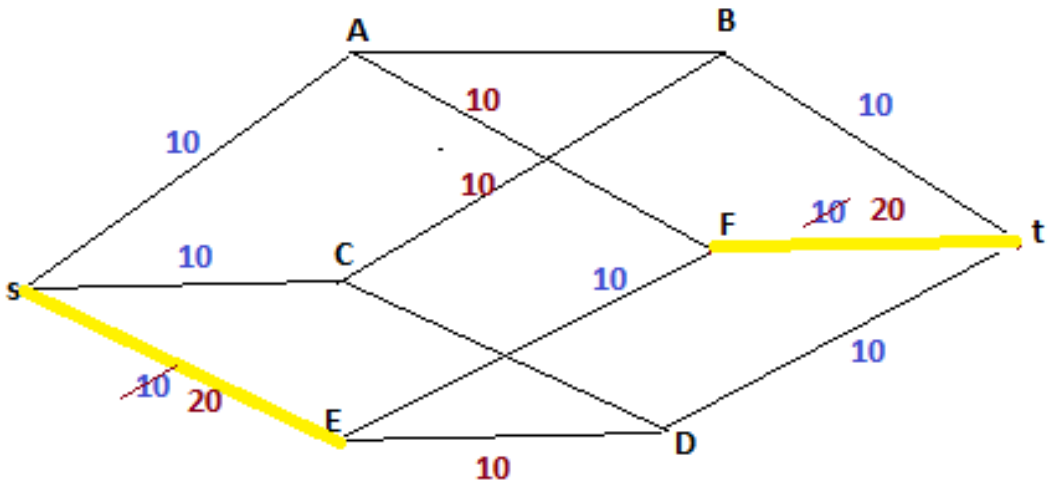
- s, E, D, C, B, A, F, t es un camino aumentante, con un par de lados backward.

Ejemplo y notación

- Denotaremos la acción de mandar 10 unidades de flujo (u otra cantidad) por ese camino asi:
- $sE\overleftarrow{DC}\overleftarrow{BA}Ft : 10$
- Las flechas de derecha a izquierda indicando cuales son los lados "backward"
- pues $\overrightarrow{DC}$ y $\overrightarrow{BA}$ no existen en el network.
- Los que existen son $\overrightarrow{CD}$ y $\overrightarrow{AB}$
- pero el orden en que listamos los vértices es D primero, luego C; y B primero, luego A.

Notación (cont.)

- Con esa notación sabemos que esos lados deben “devolver” flujo.
- Es decir, el flujo f cambia de la siguiente forma, usando la notación de C:
- $f+ = 10$ en los lados $\overrightarrow{sE}, \overrightarrow{ED}, \overrightarrow{CB}, \overrightarrow{AF}, \overrightarrow{Ft}$.
- $f- = 10$ en los lados $\overrightarrow{CD}, \overrightarrow{AB}$.



Idea de FF

- Esto lo podemos pensar como que:
- A redirige los 10 que le mandaba a B a F.
- F le manda 10 unidades mas de flujo a t.
- C le manda a B los 10 que le mandaba a D, pues ahora B no tiene problemas en aceptarlos.
- B sigue mandandole 10 a t.
- s le manda 10 a E.
- E se los manda a D.
- y D sigue mandandole 10 a t.
- (lo que queda es equivalente a haber hecho los caminos $sAFt, sCBt, sEFt, sEDt$ de entrada).

Un problema con Ford-Fulkerson

- Un problema que tiene Ford-Fulkerson que no tiene Greedy es el siguiente:
- Con un camino dirigido no saturado que usa Greedy, es obvio que al mandar ε unidades de flujo por el camino, si el ε no "reventaba" ningún caño, lo que quedaba era flujo.
- Con Ford-Fulkerson no es completamente obvio que lo que quede luego de aumentar el flujo siga siendo un flujo.
- Hemos dado en el ejemplo introductorio una racionalidad por la cual esto también debería pasar en un camino de Ford-Fulkerson, pero debemos probarlo.

FordFulkerson mantiene “flujicidad”

Teorema:

Si f es un flujo de valor v y aumentamos f con un f-camino aumentante con ε calculado como se explica en el algoritmo de Ford-Fulkerson, entonces lo que queda sigue siendo flujo y el valor del nuevo flujo es $v + \varepsilon$

- La prueba es mas o menos directa de escribir, pero un poco larga y engorrosa asi que la dejamos para mas adelante para ir adelantando temas.

Ford-Fulkerson vs Greedy

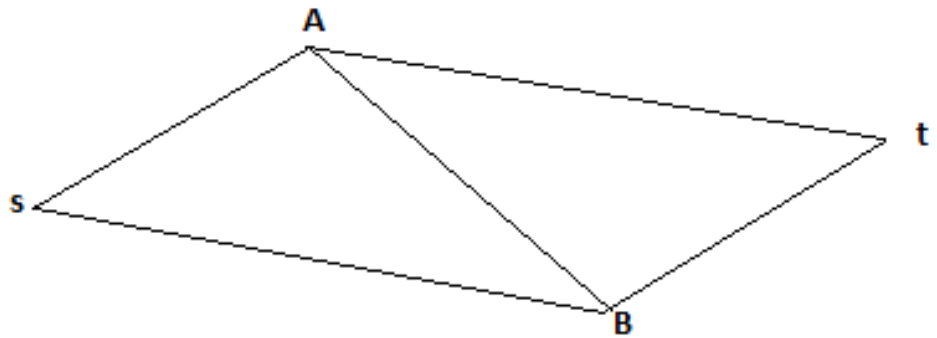
- Recordemos que vimos que Greedy tiene complejidad $O(m^2)$ pues:
  1. Encontrar un camino dirigido no saturado es $O(m)$.
  2. En cada camino dirigido no saturado se satura al menos un lado.
  3. Greedy puede hacer a lo sumo $O(m)$ incrementos de flujo pues los lados nunca se des-saturan, y por [2] cada camino satura al menos un lado.
  4. Asi que la complejidad total de Greedy es $O(m^2)$.
- En Ford-Fulkerson vale [1], pues encontrar un camino aumentante tambien es $O(m)$, pej modificando DFS o BFS.
- Y tambien vale el equivalente de [2] pues en todo camino que use Ford-Fulkerson va a haber al menos un lado que se sature, o un lado que se vacie.

Complejidad de Ford-Fulkerson

- Pero no vale [3]:
  - Como podemos “devolver”flujo a través de los lados backward, ahora los lados pueden des-saturarse, como vimos en el ejemplo. (o vaciarse y volverse a llenar)
- Asi que no queda claro que la complejidad sea $O(m^2)$ o siquiera si es polinomial.
- De hecho NO ES polinomial:

FF ej 1

- Ejemplo:



Ejemplo 1 cont

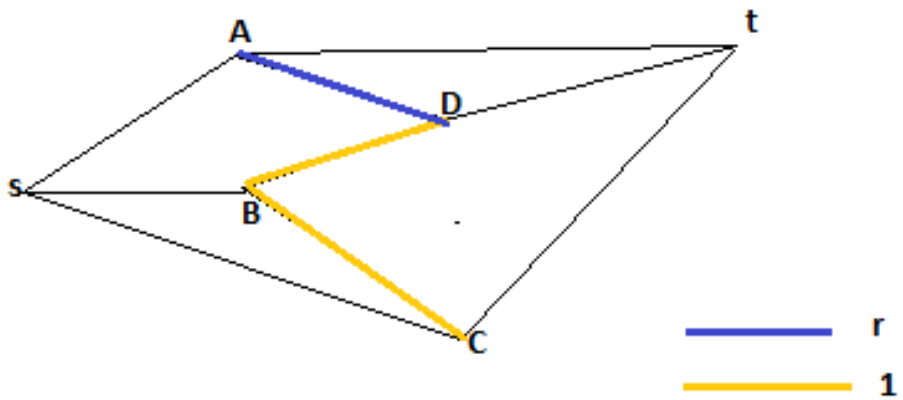
- Ford-Fulkerson encuentra el siguiente camino aumentante, comenzando con $f = 0$: $sABt : 1$.
- Luego, a partir del flujo obtenido, como se ha mandado una unidad de flujo por $\overrightarrow{AB}$, ahora se puede devolver, encuentrando el camino aumentante: $s\overleftarrow{BA}t : 1$
- Continuamos iterando con caminos $sABt : 1$ y luego $s\overleftarrow{BA}t : 1$.
- Luego de dos mil trillones de caminos, llegamos a nuestro flujo maximal.

Ejemplo 1 cont

- En vez de mil trillones podemos poner el número que queramos, lo cual muestra que no se puede acotar la complejidad con ninguna función de n, m.
- Obviamente, se puede acotar con una función que dependa de n, m y las capacidades.
- Pues vimos que hay a lo sumo $cap(\{s\})$ flujos intermedios.
- Pero puede ser peor:

Ejemplo II

- Con capacidades todas 10, excepto $\overrightarrow{BC}$, $\overrightarrow{BD}$ de capacidades 1 y $\overrightarrow{AD}$ de capacidad r.



- Donde $r = \frac{\sqrt{5}-1}{2} \simeq 0,62 < 1$.

Ejemplo II

- Ya que $10 > 1 > r$ nos concentraremos en las 3 capacidades mas chicas, $1, 1, r$
- Y especificamente miraremos las capacidades “sobrantes”es decir, capacidad-flujo en cada uno de esos lados.
- Empezamos con el camino $sBDt$. Como pasa por $\overrightarrow{BD}$, solo podemos mandar 1.
- Las capacidades sobrantes de los lados $\overrightarrow{BC}, \overrightarrow{BD}, \overrightarrow{AD}$ quedan $1, 0, r$
- Recordemos esto, digamos con un $(\star)$.

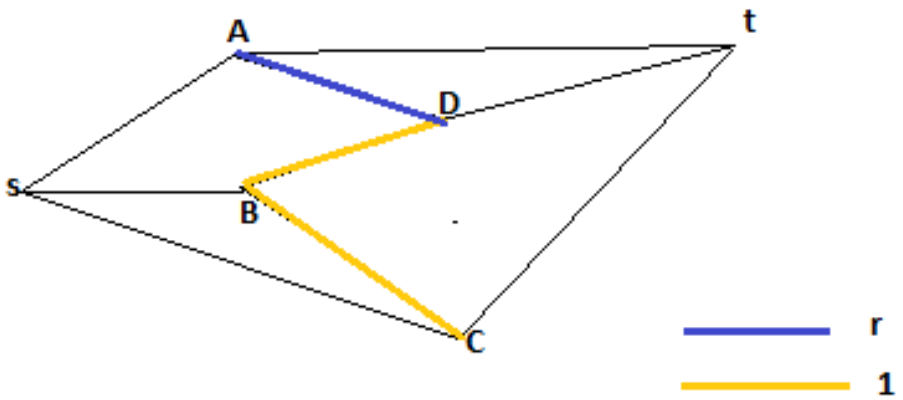
Ejemplo II

- Luego podemos ir por s, A, D y al llegar a D, ya que hay flujo desde B a D, podriamos devolver una unidad de flujo, y luego seguir por C, teniendo $sA\overleftarrow{DB}Ct$.
- ¿Cuanto podemos mandar?
- Como usamos el lado $\overrightarrow{AD}$ de capacidad r y $r < 1$, entonces podemos mandar sólo r.

Ejemplo II

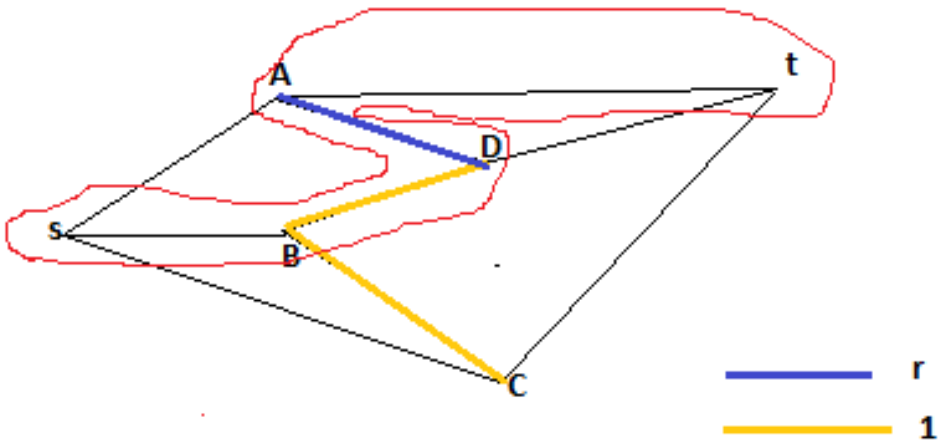
- ¿Como quedan las capacidades sobrantes de los lados $\overrightarrow{BC}, \overrightarrow{BD}, \overrightarrow{AD}$?
- Sumamos flujo r en $\overrightarrow{BC}, \overrightarrow{AD}$ y restamos r (pues devolvemos flujo) en $\overrightarrow{BD}$.
- Por lo tanto como las capacidades sobrantes son $c - f$, se le resta r en $\overrightarrow{BC}, \overrightarrow{AD}$ y se suma r en $\overrightarrow{BC}$.
- Quedan: $1 - r, r, 0$
- Pero $r = \frac{\sqrt{5}-1}{2}$ es raiz de la ecuación $r^2 + r - 1 = 0$.
- Por lo tanto $1 - r = r^2$ y las capacidades sobrantes quedan:$r^2, r, 0$

Ejemplo II



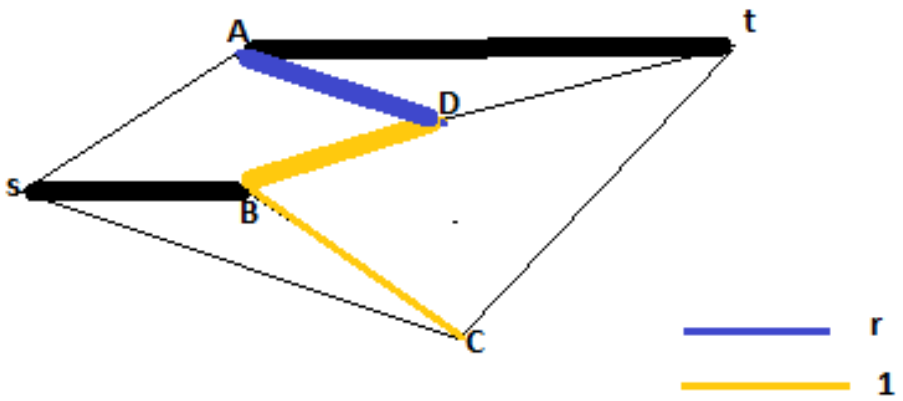
- Ahora podemos usar el camino $s\overleftarrow{BDA}t$

Ejemplo II



- Ahora podemos usar el camino $sB\overleftarrow{DA}t$ pues $\overrightarrow{BD}$ quedó con capacidad sobrante r asi que se puede mandar flujo por ahi, y como hemos mandamos flujo por $\overrightarrow{AD}$, entonces podemos devolver flujo por el.

Ejemplo II



- Ahora podemos usar el camino $s\overleftarrow{BDA}t$
- Capacidades sobrantes de los lados $\overrightarrow{BC}, \overrightarrow{BD}, \overrightarrow{AD}$ $\quad r^2, r, 0$

Ejemplo II

- Podemos mandar r por $\overrightarrow{BD}$ y devolver r por $\overrightarrow{AD}$ asi que el camino es:$sB\overleftarrow{DA}t : r$
- Capacidades sobrantes de los lados $\overrightarrow{BC}, \overrightarrow{BD}, \overrightarrow{AD}$ quedan $r^2, 0, r$

Ejemplo II

- Como las capacidades sobrantes son $r^2, 0, r$ podemos otra vez hacer el camino $s\overleftarrow{ADB}Ct$.
- Esta vez podemos mandar r^2 por $\overrightarrow{BC}$, devolver hasta 1 por el $\overrightarrow{BD}$ y mandar r por el $\overrightarrow{AD}$.
- Como $r < 1$ entonces $r^2 < r$ asi que solo podemos mandar r^2.
- Capacidades sobrantes quedan $0, r^2, r^3$

Ejemplo II

- Como las capacidades sobrantes son $0, r^2, r^3$ podemos tomar el camino $s\overleftarrow{CBD}t$.
- Esta vez podemos devolver 1 por $\overrightarrow{BC}$, y mandar r^2 por el $\overrightarrow{BD}$, asi que el minimo es r^2.
- Capacidades sobrantes quedan $r^2, 0, r^3$
- Comparemos esto con $(\star)$ que era $1, 0, r$ y vemos que multiplicamos las capacidades sobrantes por r^2.
- Esto lo podemos ir repitiendo todo el tiempo que querramos:

Ejemplo II

- Si suponemos que las capacidades sobrantes son $r^{2k}, 0, r^{2k+1}$
-
- $sA\overleftarrow{DB}Ct : r^{2k+1}$ $\quad$ $sB\overleftarrow{DA}t : r^{2k+1}$
- $sA\overleftarrow{DB}Ct : r^{2k+2}$ $\quad$ $sC\overleftarrow{BD}t : r^{2k+2}$
- Capacidades sobrantes: $r^{2k+2}, 0, r^{2k+3}$
- Y hemos pasado de $r^{2k}, 0, r^{2k+1}$ a $r^{2k+2}, 0, r^{2k+3}$ asi que podemos repetir el ciclo.
- Como el ciclo lo podemos repetir indefinidamente, Ford-Fulkerson no termina nunca.

Ejemplo II

- Mas aun, es trivial ver que esta secuencia de flujos tiene valores que convergen a 3 pero el valor del flujo maximal es 21, asi que ni siquiera converge al flujo maximal.
- En estos ejemplos estamos usando DFS para encontrar caminos aumentantes.
- En vez de usar DFS, se puede usar una variación de Dijkstra para buscar un camino de máximo aumento de flujo.
- Esto da un algoritmo polinomial pero no en n, m solamente sino en n, m y la mayor capacidad. (no probaremos esto).
- Otra posibilidad es usar BFS en vez de DFS.
- Eso equivale a decir que, de entre todos los caminos aumentantes, elegimos uno que tenga la longitud mínima.
- Veremos que esto da origen a un algoritmo polinomial en n, m.

Correctitud de Ford-Fulkerson

- De todos modos, todavia no hemos visto la correctitud de Ford-Fulkerson.
- Vimos una parte de la correctitud de Ford-Fulkerson: en todo momento, las funciones intermedias que produce son flujos.
- Por lo tanto, si termina, lo que devuelve es un flujo. (aunque vimos que a veces no termina).
- Lo que queremos ver a continuación es que, si termina, ese flujo que devuelve es maximal. (esta es la ventaja sobre Greedy).
- Pero para eso necesitaremos otro concepto y un teorema.

Definición de Corte

Definición:

Un Corte es un subconjunto de los vertices que tiene a s pero no tiene a t.

Por ejemplo, $S = \{s\}$ es un corte.
$S = V - \{t\}$ tambien es un corte.
Otro ejemplo: si los vértices son $s, A, B, C, D, E, F, G, t$
Entonces $S = \{s, A, C, D, E, G\}$ también es un corte.

Capacidad de un Corte

Definición:

La capacidad de un corte es $cap(S) = c(S, \overline{S})$, donde $\overline{S} = V - S$

Definición:

Un corte es MINIMAL si su capacidad es la menor de las capacidades de todos los cortes.

Es decir, S es un corte minimal si $cap(S) \leq cap(T)$ para todo corte T.

Sobre cortes minimales

- Cortes minimales existen porque hay una cantidad finita de cortes.
- Es fácil ver que hay 2^{n-2} cortes, donde n, como siempre, es la cantidad de vértices.
- En algunas aplicaciones es importante encontrar cortes minimales. (luego veremos una).
- Pero una complejidad de 2^{n-2} no es adecuada.
- Afortunadamente un teorema relaciona cortes minimales con flujos maximales.

Max Flow Min Cut

Teorema de Ford-Fulkerson(Max Flow Min Cut):

A. Si f es un flujo y S es un corte, entonces $v(f) = f(S, \overline{S}) - f(\overline{S}, S)$.

B. El valor de todo flujo es menor o igual que la capacidad de todo corte.

C. Si f es un flujo, las siguientes afirmaciones son equivalentes:

1. Existe un corte S tal que $v(f) = cap(S)$.
2. f es maximal.
3. No existen f-caminos aumentantes.

- Y si se cumplen, el S de [1] es minimal.

- En realidad suele llamarse Max Flow Min Cut a la parte [C] especificamente.
- o incluso mas restrictivo, a la implicación $2 \Rightarrow 1$.

Corolario

- La demostración la daremos tambien mas adelante, para ir avanzando en la practica de los algoritmos.
- Una consecuencia del teorema es:

Corolario:

Si el algoritmo de Ford-Fulkerson termina, termina con un flujo maximal

- Prueba: si Ford-Fulkerson termina, entonces termina con un flujo f para el cual no hay mas f-caminos aumentantes.
- Por lo tanto f es maximal, por el Max Flow Min Cut Theorem. Fin

Corte minimal.

- Si bien dejamos la prueba para mas adelante, para correr el algoritmo es importante conocer un detalle de la prueba.
- Si el algoritmo de FF termina, entonces termina cuando no se encuentren mas caminos aumentantes y por el MFMC es maximal.
- Ademas obtenemos un corte minimal con $cap(S) = v(f)$, y eso se obtiene de la implicación $3 \Rightarrow 1$.
- En algunas aplicaciones no solo es importante encontrar f sino tambien S.

S como certificado

- Ademas S sirve como “certificado”: para demostrar que f es maximal no hace falta volver a correr todo el algoritmo, basta exhibir S, calcular $v(f)$ y $cap(S)$ en forma independiente, y verificar si son iguales.
- Por la parte $1 \Rightarrow 2$ del teorema, si son iguales sabemos que f es maximal.
- Ademas en los ejercicios prácticos esto sirve para chequear que uno no haya cometido algún error tal que el flujo calculado no sea en realidad un flujo maximal, pues si $cap(S) \neq v(f)$ entonces hay un error en algun lado.

Construcción de S

- Y cual es el S que se construye en la demostración?
- Es
  $S = \{s\} \cup \{x \in V : \text{exista un } f\text{-camino aumentante desde } s \text{ a } x\}$.
- Se puede ir construyendo el S definido antes mientras se corre Ford-Fulkerson, durante la busqueda de un camino aumentante.
- Si en algún momento $t \in S$, no hace falta seguir: existe un f-camino aumentante y podemos aumentar el flujo.
- Si terminamos con $t \notin S$, el algoritmo se detendrá, el flujo será maximal, y el último S construido será el S que queremos.

Flujos enteros maximales

- Aún con las limitaciones que tiene Ford-Fulkerson, es suficiente para probar un teorema importante.
- Ya habiamos hablado antes sobre lo de abajo, pero lo vuelvo a recordar para plantear el teorema.
- En la definición de flujo, no se requiere que $f(\overrightarrow{xy})$ sea un entero para todo lado. Llamabamos "entero" a tales flujos.
- Algunos libros, como el Biggs, si requieren eso.
- Si uno limita los flujos a flujos enteros, entonces como hay una cantidad finita de ellos es obvio que hay flujos enteros maximales, es decir, flujos que son maximales entre los flujos enteros.

Teorema de la Integralidad

- Pero no necesariamente serán maximales entre todos los flujos.
- Pej si tenemos un sólo lado *st* de capacidad 10,4 entonces el flujo entero maximal tiene valor 10, pero el flujo maximal tiene valor 10,4.
- Esto es consecuencia directa de que hay lados con capacidades no enteras.
- Pero ¿qué pasa si todas las capacidades son enteras?
- Ese es el Teorema de la Integralidad:

Teorema de la integralidad.

En un network con capacidades enteras, todo flujo entero maximal es un flujo maximal.

Teorema de la Integralidad

- El Teorema de la Integralidad sale directo del siguiente Teorema:

Teorema

En un network donde todas las capacidades sean enteros, Ford-Fulkerson siempre termina y el flujo maximal resultante es un flujo entero.

- Este teorema demuestra el Teorema de la Integralidad porque ya vimos que si Ford-Fulkerson termina, entonces termina con un flujo maximal.
- Para demostrarlo demostraremos primero que todos los flujos intermedios que se producen en el algoritmo de Ford-Fulkerson son enteros.
- Asi que obviamente el flujo final será entero.

Prueba del Teorema de la Integralidad

- Como empezamos con el flujo $f = 0$, que es entero, basta probar que si f es entero y lo aumentamos con un camino aumentante de Ford-Fulkerson, entonces el flujo resultante sigue siendo entero.
- Asumiendo f entero, tomemos entonces un f-camino aumentante $s = x_0, x_1, ..., x_r = t$.
- Los ε_i que calculamos son:
  - $\varepsilon_i = c(\overrightarrow{x_i x_{i+1}}) - f(\overrightarrow{x_i x_{i+1}})$ en los lados forward.
    - Como tanto c como f son enteros, entonces ese ε_i es entero.
  - o bien $\varepsilon_i = f(\overrightarrow{x_{i+1} x_i})$ en los lados backward.
    - Como f es entero por hipotesis, entonces ε_i también es entero en este caso.
- Concluimos que todos los ε_i son enteros.

Prueba del Teorema de la Integralidad

- Como ε es el mínimo de los ε_i y estos son todos enteros, entonces ε es entero.
- Como f es cambiado sumando o restando ε en los lados, entonces el nuevo f tambien es entero.
- Esto concluye con la primera parte de la prueba. Para poder terminar la prueba debemos ver que efectivamente Ford-Fulkerson siempre termina si las capacidades son enteras.
- Vimos que el ε es entero, y sabemos que es positivo, porque se calcula en un camino aumentante.
- Por lo tanto, en cada aumento de flujo, el valor del flujo aumenta en AL MENOS UNO.

Prueba del Teorema de la Integralidad

- Como el valor de todo flujo es menor o igual que la capacidad de cualquier corte, todo flujo debe tener un valor acotado por pej, $cap(\{s\})$.
- Como en cada calculo de un nuevo flujo el valor aumenta en al menos uno,
- concluimos que Ford-Fulkerson puede calcular a lo sumo $cap(\{s\})$ flujos intermedios, y luego si o si debe terminar en el caso de capacidades enteras.
- Fin.