Flujo

Contenidos

Contenidos **1**

Definiciones de flujo . . . 1
Corte mínimo . . . 2
Dual . . . 2
Ford-Fulkerson vs Edmonds-Karps . . . 3
Capacidades enteras garantiza flujo entero. . . . 3
Ejercicio 0 - Matching máximo . . . 3
Ejercicio 1 - Linearidad . . . 5
Ejercicio 2 - Medicxs . . . 6
Ejercicio 3 - Satelites . . . 8
Ejercicio 4 - Titanic . . . 8

Definiciones de flujo

Definición: (Red): Una red es un grafo dirigido $G = (V, E)$ con dos vértices especiales, fuente s y sumidero t y dónde cada arco tiene asignada una capacidad no negativa, que vamos a notar como $c(u, v)$.
Definición: (Función de flujo): Dada una red definimos el flujo como una función $f : V \times V \to \mathbb{R}$ que satisface los dos siguientes axiomas:

Restricción sobre la capacidad:

$$0 \leq f(u, v) \leq c(u, v)$$

Conservación de flujo:

$$\sum_{u \in V} f(u,v) = \sum_{u \in V} f(v,u)$$

El valor del flujo $|f|$ se define como

$$\sum_{v \in V} f(s,v) - \sum_{v \in V} f(v,s)$$

Entonces el problema de **flujo máximo** consiste en, dada una red, maximizar f. Poniendo todo junto queremos resolver:

$$\begin{aligned} \max \quad & \sum_{v \in V} f(s,v) - \sum_{v \in V} f(v,s) \\ \text{s.a.} \quad & 0 \leq f(u,v) \leq c(u,v) \\ & \sum_{u \in V} f(u,v) = \sum_{u \in V} f(v,u) \end{aligned}$$

Además de maximizar el flujo, existe algo llamado el dual, esto es nada más que otra forma de pensar el problema, el dual de flujo máximo es el problema de corte de capacidad mínima.

Corte mínimo

Definición: (Corte) Un corte $C = (S,T)$ son dos conjuntos de vértices de G tal que $S \cup T = V$, $s \in S$ y $t \in T$. Definimos la capacidad del corte como

$$\sum_{u \in S, v \in T} c(u,v)$$

Entonces como dijimos, encontrar el flujo máximo es equivalente a encontrar el corte de capacidad mínima, es decir:

$$max\,|f| = \min_{C=(S,T)} \sum_{u \in S, v \in T} c(u,v)$$

Dual

- Dado un problema de flujo máximo se lo puede mirar de otra manera como un problema equivalente llamado **dual**. Este consiste en encontrar un corte de capacidad mínima.
- Un corte $C = (S,T)$ son dos conjuntos de vértices de G tal que $S \cup T = V$, $s \in S$ y $t \in T$. Definimos la capacidad del corte como

$$\sum_{u \in S, v \in T} c(u,v).$$

- Teorema de flujo máximo corte de capacidad mínima: El flujo máximo de la red es equivalente al mínimo de cualquier corte en la red.

- De esto podemos deducir que si f es un flujo en nuestra red y $cap(C)$ la capacidad de un corte cualquiera tenemos

$$f \leq cap(C).$$ [‡]

Ford-Fulkerson vs Edmonds-Karps

Ford-Fulkerson nos decía que si conseguíamos un camino de aumento entonces la complejidad de conseguir el flujo máximo se acotaba por $\mathcal{O}(mF)$ siendo m la cantidad de aristas en la red y F el flujo máximo[‖]. Por otro lado Edmond-Karps nos especifica una forma de buscar caminos de aumento (usando BFS) y además nos garantiza otra complejidad de $\mathcal{O}(nm^2)$ siendo m las aristas y n los vértices de nuestra red.

Esto significa que las dos complejidades conviven, por un lado estamos eligiendo un camino de forma de obtener la segunda cota y por otro lado como mantenemos lo que pide FF también tenemos la primera.

Al final del día podemos tomar el mínimo de los dos como una cota de la ejecución del algoritmo.

En otras palabras si G es una red con n vértices, m aristas y $F_{\max}$ [¶] es el flujo máximo entonces vale que la complejidad de FFEK está en

$$O(\min\{mF_{\max}, nm^2\})$$

Capacidades enteras garantiza flujo entero.

Teorema 0.4: Si la red tiene todas capacidades enteras entonces existe un flujo máximo entero.

Es decir que si tenemos **todas** capacidades enteras podemos garantizar que hay **alguna** asignación de flujo máximo que le asigna flujo entero a todas las aristas.

Ejercicio 0 - Matching máximo

Nos dan una lista de personas y sus preferencias para hacer ciertas tareas, cada persona puede hacer una tarea y además cada tarea solo puede ser hecha por una sola persona. Nos piden calcular la cantidad máxima de tareas que se pueden hacer.

Podemos plantear esto como un grafo bipartito, donde $G = (U, V, E)$, U siendo los vértices de las personas y V el de las tareas, como se ve en el siguiente dibujo.

Entonces lo que nos estan pidiendo es un matching de cardinalidad máxima. [§]

Queremos resolver esto con flujo, tenemos que convertir el grafo anterior en una red de manera que al conseguir un flujo máximo obtengamos el matching máximo deseado.

[‡] Esto nos va a servir para acotar las complejidades

[‖] Que no conocemos a priori pero lo podemos acotar, ver sección anterior

[¶] O una cota.

[§] Hay un algoritmo específico para esto que es más eficiente pero no lo vamos a ver en la materia: https://en.wikipedia.org/wiki/HopcroftKarp_algorithm

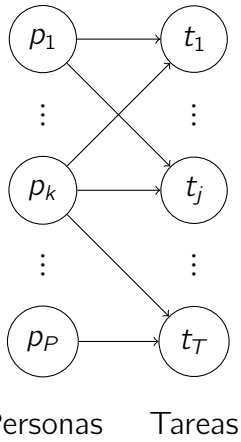

Figure 0.1: Grafo bipartito de personas y tareas.

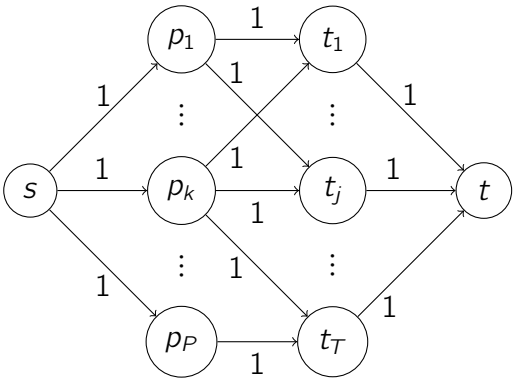

Figure 0.2: Problema representado como una red de flujo.

Para esto podemos poner las aristas con capacidad 1, si esta saturada significa que elegimos la persona-tarea que une la misma, además agregamos los vértices fuente s y sumidero t que van (y salen) a cada bipartición con capacidad 1; de esta manera por preservación de flujo no puede ser que una persona tome más de una tarea o una tarea sea hecha por más de una persona.

Ahora podemos correr FFEK y obtener un flujo máximo que afirmamos que va a representar un matching máximo. Tenemos que probar que esto es correcto, lo vamos a hacer de la siguiente manera

- $\Rightarrow$) Vamos a mostrar que una solución de nuestro problema se puede representar como una función de flujo f válida que además tiene como $F_{\max}$ el matching.

- $\Leftarrow$) Luego al revés mostramos que dado un flujo máximo y una función de flujo podemos recuperar un matching que tiene como cardinalidad el flujo máximo.

- Como todo matching posible se puede convertir a un flujo del valor de la cardinalidad en esta red y viceversa y además tenemos un flujo máximo gracias a FFEK podemos

concluir que tenemos un matching máximo.

⇒) Supongamos que tenemos un matching máximo, esto significa que tenemos un subconjunto de aristas $E' \subseteq E$ tal que no hay dos o más aristas que sea incidente al mismo vértice. Dado $e = (u, v)$ podemos asignar 1 de flujo a la arista esta y 1 de flujo de s a u y de v a t. Veamos que esto es un flujo factible, que respetamos capacidades se ve fácilmente, veamos la conservación, como tenemos un matching válido u es solo incidente a una arista por lo cual le entra 1 de flujo de s y sale solo 1 de flujo por e, lo mismo pasa con v, le entra 1 por e y sale 1 hacia t.

Además como le asignamos 1 de flujo a todas las aristas incidentes a un vértice de nuestro matching hacia el sumidero podemos afirmar que el flujo máximo es $|E'|$.

⇒) Ahora al revés, tenemos un flujo válido en nuestra red y queremos reconstruir un matching válido. Primero **e importante** dado que las capacidades son enteras podemos decir que el flujo máximo es entero y además asigna flujo entero a las aristas. Repitiendo algo que dijimos arriba, como asignamos 1 de capacidad entrante desde s (y lo mismo saliente hacia t) dado un vértice, por conservación de flujo a lo sumo le entra 1 de flujo y sale 1 por lo cual no hay más de una arista que matchee con u o v.

Con esto podemos concluir que el valor de flujo que obtuvimos es es el de un matching de cardinalidad máxima.

Ejercicio 1 - Linearidad

Sea $G = (V, E)$ una red con capacidades en los arcos. Sean f_1 y f_2 dos flujos válidos en G. Sea $\alpha \in [0, 1]$. Definimos la función $f: E \to \mathbb{R}$ como

$$f(e) = \alpha f_1(e) + (1 - \alpha) f_2(e)$$

1. Demostrar que f es un flujo válido en G.

2. ¿Es cierto que si f_1 y f_2 son flujos máximos entonces f también lo es? En caso afirmativo demostrar; en caso negativo dar un contraejemplo y justificar.

Demostración:

1. Veamos que es un flujo válido, tenemos que probar dos cosas, que respeta la capacidad máxima de las aristas y que conserva flujo.

    - Capacidad máxima: como f_1 y f_2 respetan las capacidades tenemos que ambos se pueden acotar por $c(e)$ de lo cual obtenemos

$$f(e) = \alpha f_1(e) + (1 - \alpha) f_2(e) \leq$$

$$\alpha c(e) + (1 - \alpha) c(e) = c(e)$$

- Conservación de flujo: queremos ver que el flujo que sale es igual al que entra en cualquier vértice distinto de s y t

$$\begin{aligned}\sum_{u\epsilon V} f(u,v) &= \sum_{u\epsilon V} \alpha f_1(u,v) + (1-\alpha) f_2(u,v) \\ &= \alpha \sum_{u\epsilon V} f_1(u,v) + (1-\alpha) \sum_{u\epsilon V} f_2(u,v) \\ &= \alpha \sum_{u\epsilon V} f_1(v,u) + (1-\alpha) \sum_{u\epsilon V} f_2(v,u) \\ &= \sum_{u\epsilon V} f(v,u)\end{aligned} \tag{1}$$

2. Veamos que esto es cierto también. Tenemos que tanto f_1 como f_2 son flujos máximos, muy importante es que esto no nos dice que f_1 y f_2 valen lo mismo en cada arista sino que la suma total de los valores que asignamos por ejemplo a las aristas que entran al sumidero suman lo mismo.

   Entonces tenemos que

$$F_{\max} = \sum_{u\epsilon V} f_1(u,t) = \sum_{u\epsilon V} f_2(u,t)$$

   De una forma parecida a la que usamos para ver conservación se puede ver que

$$\sum_{u\epsilon V} f(u,t) = F_{\max}$$

Ejercicio 2 - Medicxs

En un hospital hay K períodos de feriados. Cada período k consiste de $D_j = \{d_{k1} \ldots, d_{kr}\}$ días feriado contiguos. Este hospital tiene N médicxs y cada uno tiene un conjunto S_i de días disponibles para trabajar durante los períodos de vacaciones. Por ejemplo, una médica puede tener disponible viernes y sábado de Semana Santa y el lunes del feriado de Güemes. Queremos encontrar, si existe, una asignación de médicxs que cumpla:

- Nadie tiene asignado más que C días totales para trabajar en vacaciones (y sólo dentro de sus días disponibles).
- Cada día de vacaciones tiene asignada una única persona que trabaje.
- Unx médicx sólo puede tener, como máximo, un día asignado dentro de cada período D_k. Es decir, quizá tiene disponibles jueves, viernes, sábado y domingo de Semana Santa pero sólo se le puede asignar uno de esos días.

Solución: Queremos asignar a cada médico a lo sumo C días, para hacer esto podemos restringir el flujo que sale del sumidero a cada médico con C correspondientes. Además cada médico puede trabajar solo un día en cada período D_j, por lo cual podemos salir del médico con capacidad 1 a cada período, luego desde un período específico para ese médico podemos salir a los días que le corresponden también con capacidad uno. Obtenemos una red de flujo de la siguiente manera.

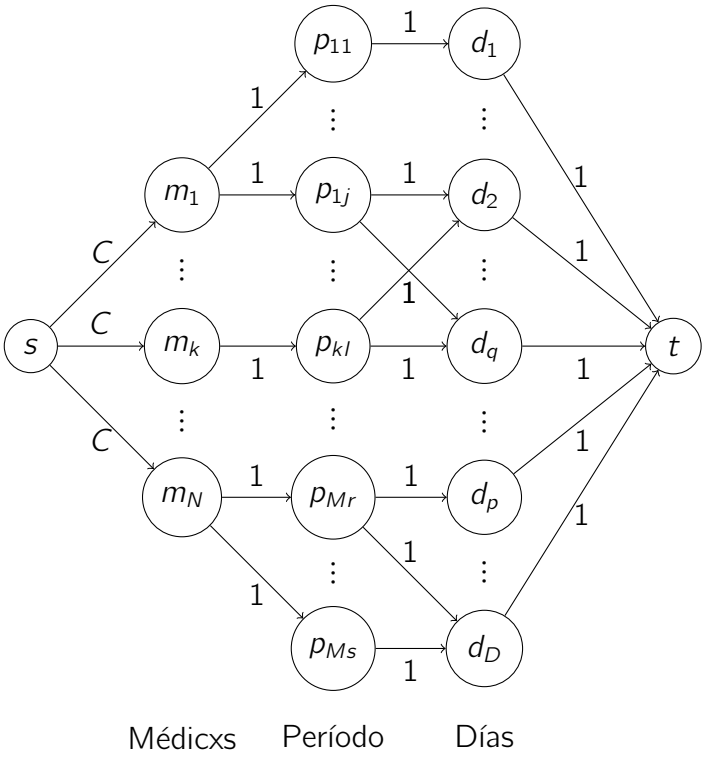

Figure 0.3: Problema de los médicos.

Queremos ver la correctitud de este modelo, de vuelta vamos a ver las implicaciones.

⇒) Dada una solución de nuestro problema queremos dar una función de flujo válida para esta red. Ya vimos parte de la explicación más arriba, falta decir que se conserva el flujo y que respetamos las capacidades. Dada una asignación válida de doctores entonces a lo sumo un doctor va a trabajar $D < C$ días por lo cual respetamos la capacidad del sumidero a m_i, además esto significa que trabaja en D períodos distintos, así que de cada doctor podemos asignar 1 de flujo a cada período correspondiente, de vuelta mantenemos conservación y restricciones. Además cada día de feriado se trabaja por un solo médico así que podemos asignar 1 de flujo desde el período correspondiente al día correspondiente.

Además como asignamos 1 de flujo a todas las aristas correspondientes a días si tomamos el flujo que entra al sumidero tenemos uno igual a la cantidad de médicxs trabajando en feriados.

⇐) Tenemos un flujo válido y queremos ver que obtenemos una asignación válida de los médicos. Es repetir el mismo argumento que recién pero yendo hacia "atrás" en la red.

Complejidad: Veamos cuantos vértices y arcos tenemos en nuestra red. Hay:

- N médicos
- Hay P períodos total, como cada médicx puede trabajar potencialmente en cualquier período tenemos una cota sobre las aristas de médicxs a períodos de NP.

- Hay K feriados totales, como cada feriado puede estar en cualquier período tenemos una cota de las aristas de períodos a feriados de NPK.

- s y t con P y K aristas correspondientemente.

Además tenemos una cota del flujo, tomando un corte sobre s a los médicos de PC y de los feríados al sumidero de K.

Si aplicamos FFEK tenemos posiblemente dos complejidades, la de EK que es

$$O((P+K+N)(NPK)^2)$$

y la de FF

$$O(NPK \min(PC, K)$$

Como en FF tomamos el mínimo de PC, K y además no tenemos $(NPK)^2$ nos podemos quedar con esta última como una mejor cota.

Ejercicio 3 - Satelites

Un satélite necesita mandar datos en megabytes a la Tierra y dispone de N ventanas de tiempo para hacerlo. Capta los datos del espacio a través de R sensores. Cada sensor tiene una cola asignada a la que cargarle datos, donde la cola q puede almacenar hasta c_q megabytes. El sensor r carga a_{rt} datos a su cola en la ventana de tiempo t, que ocurre antes de empezar la ventana de envío hacia la tierra. En cada ventana de tiempo se pueden mandar hasta d_t megabytes entre todas las colas para que finalmente lleguen a la Tierra. Si en un instante t las colas no pudieron mandar todos los datos que contenían, guardan estos datos para intentar mandarlos en el siguiente instante. Científicos de la NASA nos piden escribir un programa para maximizar la cantidad de megabytes que pueden ser transferidos desde el satélite a la Tierra en un período determinado.

Ejercicio 4 - Titanic

Después del hundimiento del Titanic mucha gente quedó varada en el agua esperando que los rescaten. La gente puede moverse en 4 direcciones y además cada posición es de la siguiente forma:

- Quedarse en un pedazo de hielo pequeño con la restricción de que no pueden quedarse para siempre acá y una vez que se muevan el hielo se hunde.

- Intentar ir al agua, lo cual significa una muerte segura por las bajas temperaturas.

- Iceberg grande, este no se hunde si la gente pasa por el pero tampoco pueden quedarse mucho tiempo dada las extremas temperaturas.

- Pedazos de Madera, solo puede subirse una persona y quedarse indefinidamente esperando al rescate.

Nos describen el mapa del área como una matriz con los tipos descrito arriba, la gente empieza en una posición con hielo pequeño. Nos ver cual es el máximo de gente que se puede salvar.

```
*~~#
...@
.~.*
```

En este ejercicio solo vamos a definir un modelo sin probar correctitud. También solo vamos a ver un par de casos del modelado ya que el resto sale de manera parecida.

- Caso $*$:

  Veamos entonces que pasa con $*$ que son los icebergs pequeños con una persona arriba. Queremos ver cuantos se pueden salvar asi que vamos a asignar 1 de capacidad de la fuente a ellos, además como queremos modelar a donde se pueden mover vamos a asignar 1 de capacidad si tiene en sus vecinos una madera u otro tipo de iceberg.

- Caso .:

  En este caso todavía nadie pasó por ellos pero tenemos un problema, si asignamos 1 de capacidad a todas las posiciones que se podrían mover podríamos habilitar a que más de una persona pase por ahi incluso si ya se hundió después de la primera, tenemos entonces que limitarlo de alguna manera. Introducimos un vértice intermedio entre . y el resto de sus vecinos. De . a este vértice solo vamos a tener capacidad uno y de este vértice nuevo podemos ir a los vecinos con cualquier capacidad ≥ 1.

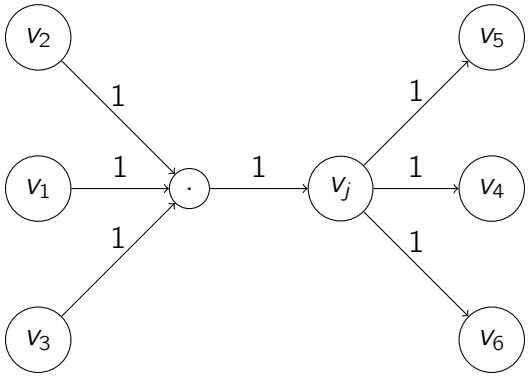

Figure 0.4: v_j es el nuevo vértice que agregamos para limitar el flujo de . a sus vecinos.