MA3705. Algoritmos Combinatoriales. 2014.
Profesor: José Soto
Escriba(s): Francisco Fernández, Emilien Garcia y Alberto Rojas.
Fecha: 24 de Octubre 2014 .

Cátedra 21

1. Flujos máximos y Algoritmo de Edmonds-Karp (II)

Algoritmo 1 Edmonds-Karp

$f \leftarrow \emptyset$
while (exista camino aumentante) **do**
  Aumentar en el camino P más corto ($f \leftarrow f + \overline{u_p^f \chi^p}$)
end while
return f

(Con P s-t camino en G_+^f)

Lema 1. *En cada iteración i, definimos las siguientes* capas*:*

$$L_j^{(i)} = \{v \in V : dist_{G_+^{f_i}}(s, v) = j\}$$

Si $v \in L_j^{(i)} \implies$ para toda iteración posterior $i' \geq i$, se tiene que $v \in L_{j'}^{(i')}$ con $j' \geq j$.
En otras palabras, la distancia entre s y v en el grafo $G_+^{f_i}$ es una función creciente en i.

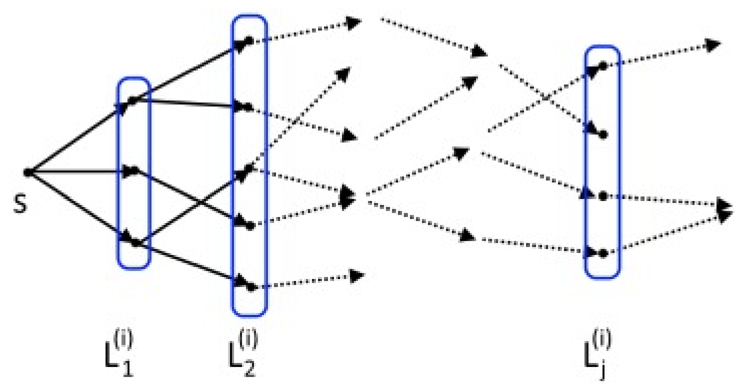

Figura 1: Representación de las capas $L_j^{(i)}$ en una iteración i.

Demostración. Notar que en la iteración i, aumentamos f_i con un s-t camino P_i.
¿Qué pasa con $G_+^{f_i} \mapsto G_+^{f_{i+1}}$? ¿Qué arco puede salir de G_+^f?.
Si $e \in E$, $u_e^{f_i} = u(e) - f_i(e)$. Al aumentar, este arco podría saturarse y su nueva capacidad residual ser 0.
Lo mismo para si $e \in E^{\leftarrow}$, $u_e^{f_i} = f_i(e)$, los únicos arcos que pueden salir de G_+^f son arcos de P_i
¿Qué arcos pueden entrar a $G_+^{f_i}$?.
Solo los arcos en $P^{\leftarrow}$ (los reversos de arcos de P).
Si $e = uv$ es un arco que sale de $G_+^{f_i}$ (hacia adelante), entonces $u \in L_j^{(i)}$, $v \in L_k^{(i)}$ *con* $j + 1 = k$.
Si $e = uv$ entra, entonces $k + 1 = j$ (hacia atrás).

En cada iteración al menos 1 arco sale de G_+^f, este arco que sale no puede volver a entrar en iteraciones futuras muchas veces. De hecho si $e = uv$ sale, y u está en la capa j, entonces cuando vuelva a entrar u debe estar en una capa de índice mayor o igual que $j + 2$.
Luego e no puede entrar a G_+^f más que $n/2$ veces. Así, cada arco sale a lo más $n/2$ veces. Como en cada etapa sale un arco y hay $2m$ arcos en total, se concluye que el número de etapas es a lo más $2m(n/2) = nm$
Por lo tanto hay a lo más:

$$O(mn) \ iteraciones$$

□

Teorema 1. *Edmonds-Karp tiene complejidad*

$$O(mn(m+n)) \ = \ O(m^2n)$$

Este algoritmo es fuertemente polinomial.

Corolario 1. *Podemos encontrar flujos máximos y s-t cortes mínimos en tiempo* $O(nm^2)$
(Recuerdo: $S = \{v : \exists s\text{-}v$ *camino en* G_+^f *para* f^* *máximo, en un s-t corte mínimo*$\}$*)*

Observación 1. Este no es el algoritmo más rápido. Se puede mejorar a $O(mn)$ por ejemplo, usando resultados muy recientes (2012) de Orlin y de King, Rao y Tarjan.

2. Problema del Corte de peso mínimo

En adelante trabajaremos en un grafo no dirigido conexo $G = (V, E)$ dotado de una función de peso $w : E \to \mathbb{R}_+$. Buscamos encontrar un conjunto de aristas F de peso mínimo que desconecte al grafo, es decir:

$$F \in \arg\min\{w(F) : (V, E \setminus F) \text{ es disconexo}\}$$

Problema Controlable 1. Demuestre la existencia de una solución de este problema de la forma $F = \delta(S)$ con $\emptyset \subsetneq S \subsetneq E$.

Por el controlable, sabemos que este problema se reduce a encontrar un corte $\emptyset \subsetneq S \subsetneq E$ con $w(\delta(S))$ mínimo. Este problema se puede resolver utilizando técnicas de flujo ya conocidas, pues Edmonds-Karp nos entrega un flujo máximo, lo que equivale a encontrar un *s*-*t* corte mínimo. Por lo tanto podemos encontrar un corte mínimo simplemente iterando Edmonds-Karp.

Una primera estrategia consiste en buscar un *s*-*t* corte entre dos vértices cualquiera y minimizar el peso del corte sobre todos los pares *s*-*t*. Así encontraremos el corte mínimo tal que al menos dos vértices no pertenezcan a la misma componente conexa, es decir, que desconecte el grafo.

Algoritmo 2 Calculando todos los s-t cortes

```
Dirigir el grafo: G' ← (V, {uv, vu : {u, v} ∈ E}) (donde los arcos heredan el peso de las aristas de las que provienen).
for ( s, t ∈ V :  s ≠ t ) do
    Calcular F(s, t) mínimo s-t corte de G'
end for
Calcular el corte F mínimo sobre todos los F(s, t)
return  F
```

Una mejora sobre el algoritmo anterior ocupa la transitividad de la conexidad en un grafo. Con esto basta que analicemos un único vértice s. Pues si desconectamos dos vértices entre sí, entonces necesariamente alguno de ellos no debe estar conectado con el vértice raíz s, de lo contrario ambos vértices estarían conectados entre sí por un camino que pasa por s. Luego un corte entre dos vértices cualquiera es necesariamente un corte entre alguno de ellos y s.

Algoritmo 3 Calculando todos los cortes a partir de una raíz

```
Dirigir el grafo: G' ← (V, {uv, vu : {u, v} ∈ E}) (donde los arcos heredan el peso de las aristas de las que provienen).
Fijar s ∈ V
for ( t ∈ V \ {s} ) do
    Calcular F(s, t) mínimo s-t corte de (G')
end for
Calcular el corte F mínimo sobre todos los F(s, t)
return  F
```

Revisemos la complejidad de estos algoritmos. En el primero se hacen $O(n^2)$ iteraciones, en cada una de ellas se debe encontrar un *s*-*t* corte mínimo en un grafo conexo, lo que con Edmonds-Karp 2 demora $O(m^2 n)$. Por lo tanto el primer algoritmo corre en $O(n^3 m^2)$, lo cual es demasiado lento, pues dependiendo del grafo esto puede llegar a $O(n^7)$. El segundo algoritmo es un poco mejor, ya que sólo realiza $O(n)$ iteraciones, demorándose en total $O(n^2 m^2)$ que sigue siendo demasiado lento, ya que puede llegar a ser $O(n^6)$.

Como estos algoritmos son muy ineficientes se vuelve necesario encontrar un algoritmo más eficiente. Para encontrarlo será necesario trabajar con propiedades de la función que queremos minimizar. Para esto introduciremos algunas definiciones.

Definición 1 (*Submodular*)**.** $f : 2^V \to \mathbb{R}$ se dice *Submodular* si:

$$f(A \cap B) + f(A \cup B) \leq f(A) + f(B)$$

Definición 2 (*Modular*)**.** $f : 2^V \to \mathbb{R}$ se dice *Modular* si:

$$f(A \cap B) + f(A \cup B) = f(A) + f(B)$$

Definición 3 (*Supermodular*). $f : 2^V \to \mathbb{R}$ se dice *Supermodular* si:

$$f(A \cap B) + f(A \cup B) \geq f(A) + f(B)$$

Observación 2. La función $|\,.\,|$ módulo de un conjunto (cardinal) es modular:

$$|A \cap B| + |A \cup B| = |A| + |B|$$

Proposición 1. *Las funciones submodulares cumplen la propiedad de retornos decrecientes. Es decir dado $X \subseteq Y$, $z \notin Y$ se cumple:*

$$f(Y + z) - f(Y) \leq f(X + z) - f(X)$$

Demostración: Tomando $A = X + z$ y $B = Y$ se tiene que $A \cap B = X$ y $A \cup B = Y + z$. luego ocupando submodularidad:

$$f(X) + f(Y + z) \leq f(X + z) + f(Y)$$

$$f(Y + z) - f(Y) \leq f(X + z) - f(X)$$

Problema Controlable 2. La recíproca también es cierta y queda propuesta su demostración. Es decir pruebe que si una función cumple la propiedad de retornos decrecientes, es entonces una función submodular.

Muchas funciones son submodulares. Veamos algunos ejemplos :

1. La función de rango de matroides.
2. El Teorema de Intersección de Matroides nos dice que :

$$\max_{I \in \mathcal{I}_1 \cap \mathcal{I}_2} I = \min_{X \subseteq S} r_1(X) + r_2(S \setminus X)$$

   En esta igualdad, el lado derecho que minimizamos es una función submodular, pues :

   - La suma de funciones submodulares es una función submodular.
   - Si f es submodular sobre V, entonces g dada por $g(X) = f(V \setminus X)(= f(X^c))$ también es submodular :

$$g(A \cup B) + g(A \cap B) = f(A^c \cap B^c) + f(A^c \cup B^c) \leq f(A^c) + f(B^c) = g(A) + g(B)$$

     donde la desigualdad se tiene por submodularidad de f, y las igualdades por definición de g.

3. Las funciones de cortes y de cortes dirigidos son submodulares :
   Sean $\omega : E(G) \longrightarrow \mathbb{R}_+$ y $f : 2^V \longrightarrow \mathbb{R}$ tal que $f(S) = \omega(\delta(S))\ \forall S \subseteq V$; f es submodular ya que como lo muestra el siguiente dibujo se cumple : $f(A \cup B) + f(A \cap B) \leq f(A \cup B) + f(A \cap B) + f(E(A : B)) = f(A) + f(B)$, donde $E(A : B)$ representa las aristas que van de A en B (o equivalentemente de B en A ya que el grafo no es dirigido).

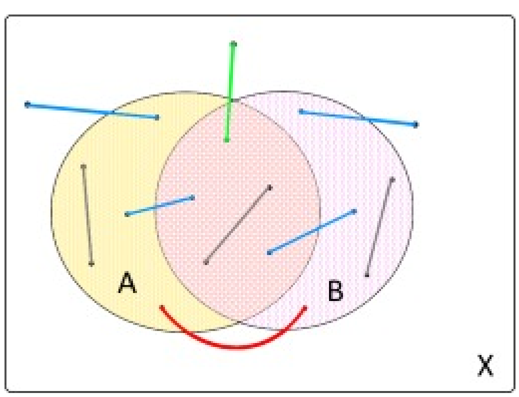

Figura 2: Representación del conteo de las aristas asociadas a $f(A) + f(B)$ y a $f(A \cup B) + f(A \cap B)$.

Cada tipo de aristas aparece en este dibujo : las que no contamos en ninguna cantidad son grises y las que contamos una vez en ambas cantidades son azules ; por otro lado, las que contamos dos veces aparecen en verde y rojo.
Ojo : los dos colores corresponden a dos casos de conteo distintos. En efecto, la arista verde la contamos dos veces en $f(A)+f(B)$ (está en el corte de A y en el corte de B) y dos veces en $f(A\cup B)+f(A\cap B)$ (está en el corte de $A\cup B$ y en el corte de $A\cap B$). Sin embargo, la arista roja se cuenta dos veces en $f(A)+f(B)$ (está en el corte de A y en el corte de B, pues sale de A y sale de B) pero no se cuenta en $f(A\cup B)+f(A\cap B)$ (no sale de $A\cap B$ ni sale $A\cup B$).
Con esto se entiende bien que las aristas contadas en $f(A)+f(B)$ pero no en $f(A\cup B)+f(A\cap B)$ son las que van de A en B, es decir las contadas en $f(E(A:B))$.

4. En cierto sentido, las funciones submodulares son el análogo a las funciones convexas o cóncavas en optimización continua :
   - Sea f con dominio $2^V=\{0,1\}^V$. Su dominio se puede ver como los vértices de un hipercubo, con coordenadas 0 o 1 en cada dimensión. Podemos extender su dominio a $[0,1]^V$ de manera multilinear, donde $\{0,1\}^V$ son los vértices del cubo. Se puede verificar que f así extendida es convexa si y solo si la función original es submodular.
   - Si $g:\mathbb{R}\longrightarrow\mathbb{R}$ es cóncava, entonces f dada por $f(X)=g(|X|)$ es submodular.

Concentrémonos ahora en las funciones de corte. Sean entonces el grafo (no dirigido) $G=(V,E)$, las funciones $\omega : E(G)\longrightarrow\mathbb{R}_+$ y $f:2^V\longrightarrow\mathbb{R}$ tal que $f(S)=\omega(\delta(S))\ \forall S\subseteq V$. Como lo acabamos de ver, f es una función submodular, pero además es una función simétrica, en el sentido que $f(X)=f(S\setminus X)\ \forall X\subseteq S$.
Por eso estudiaremos un algoritmo para minimizar funciones a la vez submodulares y simétricas.

Definición 4 (Separador de dos elementos)**.** $S\subseteq V$ separa dos elementos $s,t\in V$ si $|S\cap\{s,t\}|=1$.

Definición 5 (Trío Factible)**.** Sean $s,t\in V$ y $S\subseteq V$ tal que S separa s y t, (S,s,t) se dirá trío factible si es el mejor separador de s y t, es decir si $f(S)\leq f(X)$ para todo X s-t separador.

Imaginemos que somos capaces de encontrar un trío factible. La idea es la siguiente : si (S,s,t) es un trío factible y X es un mínimo para f,

- o bien X separa s y t, entonces $f(S)=f(X)$, luego S es óptimo.
- o bien X no separa s y t, y entonces todo óptimo tiene a s y t juntos.

Definición 6 (Fusión de dos elementos)**.** Sea $f:2^V\longrightarrow\mathbb{R}$ submodular, sean $s,t\in V$. La función $f_{st}:2^{V-s-t+st}\longrightarrow\mathbb{R}$ es la fusión de los dos elementos s y t, dada por :

$$\forall X\subseteq(V-s-t+st), f_{st}=\begin{cases} f(X) & si \quad st\notin X\\ f(X\cup\{s,t\}-st) & si \quad st\in X\end{cases}$$

Veamos ahora la idea principal del algoritmo que minimiza una función submodular y simétrica :

Algoritmo 4 Minimizar f submodular simétrica

```
U ⟵ ∅
f' ⟵ f
while (existe (S, s, t) trío factible de f') do
    U ⟵ U + S', donde S' es el conjunto obtenido al expandir los elementos de S.
    f' ⟵ función fusión de s y t
end while
return S ∈ arg mín{f(X) ; X ∈ U}
```

La próxima clase, veremos un algoritmo más explícito que sigue este procedimiento para minimizar una función submodular y simétrica : el algoritmo de Queyranne.