MA4701. Optimización Combinatorial. 2013.
Profesor: José Soto
Escriba(s): Ivana Bachmann y Abner Turkieltaub.
Fecha: 4 de noviembre de 2013.

Cátedra 19

1. Flujo máximo en (G, u, s, t)

Previamente vimos un algoritmo genérico para encontrar flujo máximo, el algoritmo de Ford-Fulkerson. Este consistía en buscar caminos aumentantes en (G_0^f, u^f, s, t), sin embargo este algoritmo no era polinomial. La clase anterior vimos el algoritmo de Edmond-Karp, este algoritmo seguía la idea de Ford-Fulkerson, pero eligiendo en cada iteración el camino aumentante más gordo, es decir, elige P un $s-t$ camino en el grafo residual tal que $\min_{e \in P} u_e^f$ sea lo mayor posible. En cuanto a la complejidad de este algoritmo tenemos que:

- Si $u \in \mathbb{Z}$, el número de iteraciones (aumentos) es $O(m \ln(\text{valor}(f_{max})))$.
- Cada aumento se puede realizar en $O(Dijkstra) = O(m + n \ln n)$

Por lo tanto E-K demora $O((m \ln(\text{valor}(f_{max})))(m+n \ln n)) = O(m^2 \ln n \ln |\text{valor}(f_{max})|)$. Podemos acotar $\ln |\text{valor}(f_{max})|)$ como sigue:

$$\ln |\text{valor}(f_{max})|) \leq \ln \sum u_e \leq \#\text{ de bits de la entrada.}$$

Por lo tanto, este algoritmo depende de los números de la entrada, no solo de la cantidad de datos, es decir, es débilmente polinomial.

En esta clase presentaremos un algoritmo para encontrar flujo máximo que sí es fuertemente polinomial, la idea del algoritmo es la misma que en los anteriores, pero esta vez el criterio para elegir caminos aumentantes será elegir aquellos con menos arcos, en vez de los más gordos.

2. Algoritmo de Edmond-Karp 2

Algorithm 1 Algoritmo de Edmond-Karp 2.

```
Dado (G, u, s, t) una red.
f ← 0.
Calcular (G_0^f, u^f, s, t).
while Existe un camino aumentante en (G_0^f) do
    P ← camino aumentante con menos arcos.
    Aumentar f en P.
    Calcular (G_0^f, u^f, s, t).
end while
Devolver f.
```

Observación. *E-K 2 es correcto, lo único que falta es ver su complejidad.*

Analicemos cada iteración. Para encontrar el camino con menos aristas podemos usar BFS, notemos que al aplicar BFS desde el nodo s podemos agrupar los nodos por capas, definiendo los conjuntos:

$$L_i = \{v : \ s - v \text{ camino tiene } i \text{ arcos}\}.$$

En otras palabras, $v \in L_i \Leftrightarrow$ el camino con menos arcos de s a v tiene i arcos.

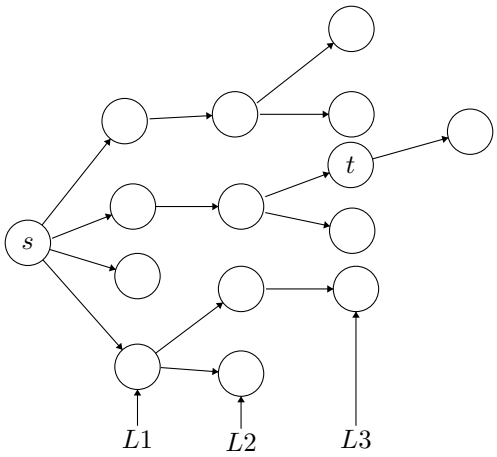

Figura 1: Ubicación de capas de nodos L_i

Veamos qué pasa con G_0^f cuando aumentamos con P.

$$V(P) = \{s, v_1, v_2, \ldots, v_k\} \text{ con } v_k = t,\ v_i \in L_i.$$

Aumentemos $\delta = \min_{e \in P} u_e^f$ en P y veamos como cambia G_0^f.

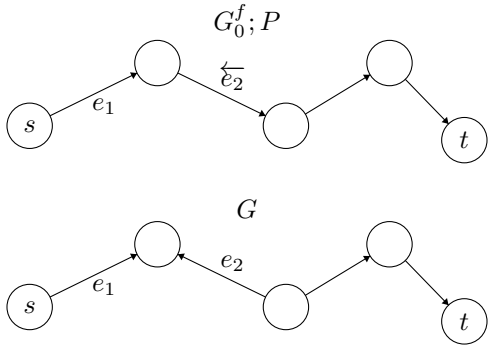

Figura 2: Relación de G_0^f con G

- Solo pueden salir de $E(G_0^f)$ aquellos arcos e en P con $u_e^f = \delta$.
- Solo pueden entrar a $E(G_0^f)$ los arcos reversos de P, (pues ellos aumentan su capacidad en δ).

Lema 1. *Si $v \in L_i$ en una iteración, entonces $v \in L_j$ con $j \geq i$ para toda iteración posterior.*

Demostración. *En el BFS, tendremos que:*

($\diamond$) *Los arcos de G_0^f que salen, van de L_k a L_{k+1}.*

($\diamond\diamond$) *Los arcos que entran a G_0^f van de L_{k+1} a L_k.*

Si agregamos los arcos en ($\diamond\diamond$) uno a uno en G_0^f, la distancia entre s y v no puede disminuir (se mantiene). Y si luego borramos los arcos en ($\diamond$) de G_0^f, la distancia de s a v solo puede aumentar o mantenerse. ■

Teorema 1. *E-K toma a lo más nm iteraciones.*

Demostración. *Cada arco entra como máximo $n/2$ veces, y en cada iteración un arco sale. Luego:*

$$\#\text{ de iteraciones } \leq \frac{n}{2}|E \cup \overleftarrow{E}| = nm$$

. ■

Corolario 1. *E-K encuentra un flujo máximo en* $O(nm^2)$.

Veamos algunas consecuencias:

1. Podemos encontrar flujo máximo en tiempo $O(nm^2)$.
2. Si $u \in \mathbb{Z}^E$, el flujo máximo integral puede encontrarse en $O(nm^2)$.
3. Podemos encontrar un $s-t$ corte mínimo en tiempo $O(nm^2)$.

Observación. *Existen algoritmos más rápidos para* f_{max} *y* c_{min}.

3. Corte Mínimo

La pregunta que queremos abordar ahora es, dado G conexo, no dirigido y $u : E \to \mathbb{R}_+$, ¿Cómo podemos encontrar $F \subseteq E$ con $u(F)$ mínimo tal que $G \setminus F$ sea disconexo? Una opción para encontrar F sería la siguiente:

1. Bidirigir E. $(\star)$
2. Para cada par de nodos (u, v), $S_{uv} \leftarrow minu - v$ corte en G'.
3. Tomamos $S \in \{S_{uv} : \ (u, v) \in V \times V, \ u \neq v\}$ que minimice $u(\delta^+(S))$.
4. Devolver $\delta_E S$.

$(\star)$ Bidirigir E:

$\forall e = uv \in E$:

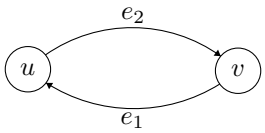

Figura 3: conversión a arcos dirigidos

Observación. *Problema Controlable:*
$\forall F \subseteq E, \ \exists S \subseteq V, S \neq \phi, \ S \neq V$, *tal que* $u(F) \geq u(\delta^+(S))$.

Con la observación anterior, $\delta_E(S)$ es corte mínimo.

Corolario 2. *Podemos encontrar* F *corte mínimo en tiempo* $O(n^2)O(s-t$ *corte mín), en nuestro caso, esto es* $O(n^3m^2)$.

Observación. *Lo anterior puede hacerse en* $O(n)O(s-t$ *corte mín), en vez de revisar cada par* (u, v)*, dejamos* u *fijo y vemos todos los pares* (u, v) *con* $v \neq u$. *(Esto funciona ya que* u *siempre estará en alguna de las mitades que deja el corte). En cualquier caso esto nos daría* $O(n^2m^2)$ *que sigue siendo demasiado.*

En la siguiente clase veremos un algoritmo para encontrar cortes mínimos en grafos no dirigidos más eficiente.