ACM Training Camp: Grafos II

Martín D. Safe

6 de agosto de 2010

Motivación

- Supongamos que queremos mandar tantos camiones como sea posible desde un punto r en una cierta red de caminos hacia otro punto s. Además, para cada tramo e de la red hay una cota superior u_e en el número de camiones que pueden recorrer ese tramo.
- Supongamos que la red de caminos viene dado por un digrafo $G = (V, E)$. Entonces nuestro problema es el de encontrar un conjunto de (r, s)-caminos dirigidos $P_1, P_2, \ldots, P_k$ de G tal que cada arco e sea atravesado por a lo sumo u_e caminos distintos y tal que se maximice k.
- Podemos suponer que los caminos son simples y que G también es simple.

Reformulación

Para cada tramo e sea $x_e = \#\{i : P_i \text{ recorre } e\}$.

- Para cada nodo $v \neq r, s$, cada P_i que entra a v sale de él. Por lo tanto:

$$\sum_{w \in V : wv \in E} x_{wv} - \sum_{w \in V : vw \in E} x_{vw} = 0, \quad \text{para todo } v \in V \setminus \{r, s\}$$

$$0 \leqslant x_{vw} \leqslant u_{vw}, \quad \text{para todo } vw \in E$$

$$x_{vw} \text{ es entero}, \quad \text{para todo } vw \in E$$

- El número k de caminos dirigidos satisface

$$k = \sum_{w \in V : ws \in E} x_{ws} = \sum_{w \in V : rw \in E} x_{rw}$$

Flujo

- Una función $x : E \to \mathbb{R}$ es un flujo si para cada nodo $v \in V \setminus \{r, s\}$ se satisface la condición de conservación de flujo:

$$f_x(v) := \sum_{\{w \in V : vw \in E\}} x_{wv} - \sum_{\{w \in V : vw \in E\}} x_{vw} = 0$$

  La conservación no se requiere ni en r (fuente) ni en s (sumidero).
- Se dice que el flujo x es factible si además $0 \leqslant x_e \leqslant u_e$ para todo $e \in E$. Permitiremos que cada $u_e \geqslant 0$ ó $u_e = +\infty$.
- Si cada x_e es enteros ó $+\infty$, se dice que x es un flujo entero.
- El valor de un flujo x es igual al flujo neto hacia el sumidero s:

$$f_x(s) := \sum_{\{w \in V : ws \in E\}} x_{ws}.$$

- Un flujo máximo (resp. entero) es un flujo factible (resp. entero) de valor máximo.

Relación entre flujo máximo y "empaquetamiento" de caminos

Si todas las capacidades u_e son enteras entonces existe la siguiente relación con el problema original:

Proposición

Existe una familia $P_1, P_2, \ldots, P_k$ de (r, s)-caminos dirigidos tales que $\#\{i : P_i \text{ recorre } e\} \leqslant u_e$ para cada $e \in E$ si y sólo si existe un flujo factible entero de valor k.

Aún cuando los u_e no son enteros vale lo siguiente:

Proposición

Cada flujo de valor no negativo es la suma de a lo sumo $\#E$ flujos, cada uno de los cuales es un flujo un valor único no negativo a lo largo de un camino o circuito dirigido.

Corte

- Para cada $R \subseteq V$ definimos $\delta(R)$ como el conjunto de arcos que van desde un nodo de R a un nodo fuera de R. O sea:

$$\delta(R) = \{vw \in E : \ v \in R, \ w \notin R\}.$$

- Para cada $R \subseteq V$ tal que $r \in R$ y $s \notin R$ decimos que $\delta(R)$ es un corte.
- La capacidad del corte $\delta(R)$ es

$$u(\delta(R)) = \sum_{e \in \delta(R)} u_e.$$

- Para cualquier corte $\delta(R)$ y cualquier flujo x vale que

$$f_x(s) = x(\delta(R)) - x(\delta(\bar{R})) = \sum_{e \in \delta(R)} x_e - \sum_{e \in \delta(\bar{R})} x_e.$$

- Si x es un flujo factible y $\delta(R)$ es cualquier corte entonces:

$$f_x(s) \leqslant x(\delta(R)) \leqslant u(\delta(R)).$$

Flujo máximo y corte mínimo

Teorema Max-Flow Min-Cut

Si existe un flujo máximo entonces

$$\text{máx}\{f_x(s) : x \text{ es un flujo factible}\} = \text{mín}\{u(\delta(R)) : \delta(R) \text{ es un corte}\}$$

Algunas consecuencias que nos importan son las siguientes:

- Podemos demostrar que un flujo factible x es máximo exhibiendo un corte $\delta(R)$ cuya capacidad iguala el valor de x.
- Si x es un flujo factible y $\delta(R)$ es un corte entonces x es máximo y $\delta(R)$ es mínimo si y sólo si

$$x_e = u_e \text{ para todo } e \in \delta(R) \quad \text{y} \quad x_e = 0 \text{ para todo } e \in \delta(\bar{R}).$$

Algoritmos push-relabel para flujo máximo

- Vamos a introducir un tipo de algoritmos conocidos como algoritmos push-relabel (Goldberg y Tarjan) que resuelve el problema de flujo máximo. En particular, vamos a mostrar que puede implementarse para correr en tiempo $O(n^3)$.
- Los algoritmos de camino aumentante trabajan enviando flujo a través de caminos dirigidos que van desde la fuente hasta el sumidero.
- Dos algoritmos de camino aumentante muy conocidos son el de Edmonds-Karp $O(m^2 n)$ y el de Dinits $O(n^2 m)$.
- La dificultad de los algoritmos de camino aumentante es que pueden requerir un tiempo $\Theta(n)$ para pasar flujo a lo largo de un (r, s)-camino aumentante.
- Los algoritmos push-relabel no tienen esta dificultad porque envían flujo a lo largo de arcos individualmente en lugar de a lo largo de caminos aumentantes.
- Pero por esta misma razón durante su ejecución no se respeta la conservación de flujo en los nodos...

Preflujo

- A los fines del desarrollo teórico convenimos que si $vw \notin E$ entonces $x_{vw} = u_{vw} = 0$.
- Decimos que x es un preflujo si

$$f_x(v) \geqslant 0 \quad \text{para todo nodo } v \in V \setminus \{r, s\}.$$

- El valor $f_x(v)$ se llama el exceso en v.
- Un nodo $v \in V \setminus \{r, s\}$ se dice activo si su exceso $f_x(v) > 0$.
- Si además $0 \leqslant x \leqslant u$ entonces x es un preflujo factible.
- Notese que un preflujo (factible) x es un flujo (factible) si y sólo si x no tiene nodos activos.
- Dado un preflujo x, el digrafo auxiliar $G(x)$ tiene los mismos nodos y $vw \in E(G(x))$ si y sólo si $x_{vw} < u_{vw}$ ó $x_{wv} > 0$.
- Notemos que

$$vw \in E(G(x)) \text{ si y sólo si } \bar{u}_{vw} := u_{vw} - x_{vw} + x_{wv} > 0.$$

Push en un arco

- Si vw es un arco de $G(x)$ entonces hasta $\bar{u}_{vw}$ unidades de flujo pueden "empujarse" a lo largo de vw manteniendo la restricción de no negatividad y de capacidad (pero no necesariamente la definición de preflujo).
- Si $\bar{u}_{vw} > 0$ y $f_x(v) > 0$ entonces $\varepsilon = \text{mín}(\bar{u}_{vw}, f_x(v)) > 0$ unidades de flujo pueden "empujarse" a lo largo de vw.

push en vw

$\varepsilon \leftarrow \text{mín}(\bar{u}_{vw}, f_x(v))$
decrementar x_{wv} tanto como $\varepsilon' \leftarrow \text{mín}(\varepsilon, x_{wv})$
incrementar x_{vw} tanto como $\varepsilon - \varepsilon'$

¿Hacia donde empujamos el flujo?

- La idea básica de los algoritmos push-relabel es elegir un nodo activo v y un arco $vw \in G(x)$ y empujar flujo a través de vw.
- Pero nos falta especificar de qué manera elegimos el nodo v y el arco vw a través del cual empujar flujo.
- Si no, podríamos empujar flujo a través de vw, luego a través de wv, luego de nuevo a través de vw, ...
- La estrategia que se utiliza sigue la idea de empujar el flujo hacia los nodos estén más "cerca" del sumidero.
- En posible que llegado un punto no se pueda empujar más flujo hacia el sumidero y en ese caso queremos empujar el exceso hacia los nodos más "cerca" de la fuente para volver a restablecer la conservación de flujo.
- Vamos a estimar la cercanía de un nodo al sumidero o a la fuente mediante los etiquetados válidos.

Etiquetado válido e inicialización

- Si x es un preflujo factible, una función $d : V \to \mathbb{N} \cup \{0, +\infty\}$ es un etiquetado válido para x si

$$d(r) = n, \quad d(s) = 0 \quad \text{y} \quad d(v) \leqslant d(w) + 1 \ \forall vw \in E(G(x)).$$

- Notemos que $d(v) = d_x(v, s)$ (la distancia en $G(x)$ de v a s) satisface la definición de etiquetado válido, con excepción de $d(r) = n$.
- No siempre es posible dar un etiquetado válido para cualquier preflujo factible.
- Pero es posible dar *un* preflujo factible y *un* etiquetado válido.

inicializar x y d

$d(r) \leftarrow n$
$d(v) \leftarrow 0$ para todos los $v \in V \setminus \{r\}$
$x_e \leftarrow u_e$ para los arcos saliente de r
$x_e \leftarrow 0$ para todos los demás $e \in E$

Criterio de parada

Proposición

Si x es un preflujo factible y d es un etiquetado válido para x entonces existe un corte $\delta(R)$ tal que

$x_{vw} = u_{vw}$ para todo $e \in \delta(R)$ y $x_{vw} = 0$ para todos los $vw \in \delta(\bar{R})$.

Demostración.

Alcanza con considerar $R = \{v \in V : d(v) > k\}$ donde k se elige de manera que $0 < k < n$ y $d(v) \neq k$ para todo $v \in V$. □

Este corolario nos da una criterio de parada:

- Por el Teorema Max-Flow Min-Cut: *Si x es un flujo factible y x admite un etiquetado válido entonces x es un flujo máximo.*
- Los algoritmos push-relabel mantienen un preflujo factible y un etiquetado válido (y por la Proposición un corte saturado por el preflujo).
- Si en algún momento el preflujo factible se vuelve un flujo, hemos hallado un flujo máximo (porque satura un corte).

Etiquetado válido como aproximación de la distancia

Lema

Si x es un preflujo factible y d es un etiquetado válido para x entonces

$$d_x(v, w) \geqslant d(v) - d(w) \quad \text{para todo los } v, w \in V.$$

Demostración.

Sean $v, w \in V$. Si $d_x(v, w) = +\infty$ no hay nada que probar. Si no, sumando las desigualdades $d(p) - d(q) \leqslant 1$ para cada arco pq de un (v, w)-camino dirigido en $G(x)$ de longitud mínima se obtiene $d(v) - d(w) \leqslant d_x(v, w)$. □

Algunas consecuencias:

- $d(v)$ es una cota inferior de la distancia $d_x(v, s)$. En particular, si $d(v) \geqslant n$ entonces $d_x(v, s) = +\infty$.
- $d(v) - n$ es una cota inferior de la distancia $d_x(v, r)$.

¿Hacia donde empujamos el flujo? (2)

- Siempre trataremos de empujar flujo a lo largo de arcos $vw \in E(G(x))$ tales que v es activo y $d(w) < d(v)$.
- Por definición de etiquetado válido, si $vw \in E(G(x))$ entonces $d(w) < d(v)$ es equivalente a $d(w) = d(v) - 1$.
- Por lo tanto, siempre intentaremos empujar flujo a lo largo de arcos admisibles, que son aquellos arcos $vw \in G(x)$ tales que v es activo y $d(w) = d(v) - 1$.
- Después de empujar flujo a lo largo de un arco admisible, el etiquetado d sigue siendo válido. En efecto, el único nuevo arco que puede entrar en $G(x)$ es wv y para el cual $d(w) \leqslant d(v) + 1$ ya se satisface.

Reetiquetado

- Puede suceder que para un determinado nodo activo v no exista un arco admisible vw.
- En ese caso reetiquetamos:

reetiquetar v

$$d(v) \leftarrow \text{mín}\{d(w) + 1 : vw \in E(G(x))\}$$

- La idea detrás de todo esto es la siguiente:
  - Imaginemos que el flujo es en realidad un líquido que siempre fluye hacia abajo y $d(v)$ representa la altura de v.
  - Si el algoritmo encuentra un nodo que tiene exceso de líquido y un arco descendente que sale de él entonces permite que algo de líquido fluya por ese arco.
  - Pero si hay un exceso de líquido en un nodo y no hay ningún arco que vaya hacia abajo por el cual se pueda hacer fluir líquido entonces se eleva el nodo hasta que uno de sus arcos salientes se vuelva descendente.

Procesar un nodo activo

- Nuestro algoritmos push-relabel va a considerar un nodo activo por vez y lo va a procesar.

procesar v

mientras haya un arco admisible vw
    push en vw
si v es activo entonces
    reetiquetar v

Aunque hay algoritmos más generales del tipo push-relabel, nos restringimos a algoritmos que cada vez que se elige un nodo activo v para procesar se sigue pasando flujo a través de arcos admisibles vw hasta que no haya arcos admisibles vw y si v sigue activo entonces es reetiquetado.

Algoritmo push-relabel para flujo máximo

push-relabel

inicializar x y d
mientras x no sea un flujo
    elegir un nodo activo v
        procesar v

Teorema

El algoritmo push-relabel realiza $O(n^2)$ reetiquetados y $O(n^2m)$ pushes.

Teorema (para al algoritmo de distancia máxima)

El algoritmo push-relabel que siempre elige un nodo activo que maximiza d realiza $O(n^2)$ reetiquetados y $O(n^3)$ pushes.

Número de reetiquetados

Lema

Si x es un preflujo y w es un nodo activo entonces existe un (w, r)-camino dirigido en $G(x)$.

Demostración.

Sea R el conjunto de nodos v tales que existe un (v, r)-camino dirigido en $G(x)$. Alcanzará con probar que si $w \in \bar{R}$ entonces w no es activo.

- Por definición de R, no hay arcos de $G(x)$ que dejen $\bar{R}$, i.e., $x(\delta(R)) = 0$.
- Por otro lado, si sumamos $f_x(w) \geqslant 0$ para todos los $w \in \bar{R}$ obtenemos

$$x(\delta(R)) - x(\delta(\bar{R})) = \sum_{w \in \bar{R}} f_x(w) \geqslant 0. \qquad (*)$$

- Como $x(\delta(R)) = 0$ entonces $x(\delta(\bar{R})) = 0$ y la desigualdad (*) vale con igualdad por lo que $f_x(w) = 0$ para todo $w \in \bar{R}$. □

Número de reetiquetados (2)

Lema

En cada etapa del algoritmo de push-relabel vale que

$$d(v) \leqslant 2n - 1 \quad \text{para cada } v \in V.$$

Demostración.

Como solo los nodos activos son reetiquetados, alcanza con probarlo para nodos activos. Sea v un nodo activo.

- Como x es un preflujo entonces existe un (v, r)-camino dirigido en $G(x)$ y, en particular, $d_x(v, r) \leqslant n - 1$.
- Como d es un etiquetado válido entonces, por un lema anterior, $d_x(v, r) \geqslant d(v) - d(r) = d(v) - n$.
- Concluimos que $d(v) \leqslant (n - 1) + n = 2n - 1$. □

Cada uno de los n nodos se reetiqueta a lo sumo $2n - 1$ veces:

Corolario

El algoritmo push-relabel realiza $O(n^2)$ etiquetados.

Número de pushes saturantes

Un push en vw es saturante si $\bar{u}_{vw} \leqslant f_x(v)$, de modo que $\bar{u}_{vw}$ unidades de flujo son empujadas.

Lema

El número de push saturantes en un algoritmo push-relabel es a lo sumo $2mn$.

Demostración.

Consideremos un par fijo de nodos (v, w) tales que $vw \in E$ o $wv \in E$.

- Luego de un push saturante en vw, el arco vw sale de $G(x)$.
- Antes de un nuevo push en vw, debe haber uno en wv.
- Como $d(v) = d(w) + 1$ cuando hay un push vw y $d(w) = d(v) + 1$ cuando hay uno en wv entonces, entre dos push saturantes de vw, $d(w)$ se incrementa en al menos 2.
- Luego, hay a lo sumo n push saturantes en vw y a lo sumo $2mn$ en total.

□

Número de pushes no saturantes

Un push en vw es no saturante si, por el contrario, $f_x(v) < \bar{u}_{vw}$, de modo que vuelve a v inactivo.

Lema

El número de pushes no saturantes de un algoritmo push-relabel es $O(n^2m)$.

Demostración.

Sea $D = \sum_{f_x(v)>0} d(v)$. D es inicialmente 0 y siempre $D \geqslant 0$.

- Un reetiquetado aumenta D y un push saturante que incremente D lo hace en a lo sumo $2n-1$ unidades (si w se vuelve activo y v sigue activo).
- Un push no saturante decrementa D o bien en 1 si w pasa de inactivo a activo, o en $d(v)$ en caso contrario .
- Como todos los incrementos de D se deben a reetiquetados y pushes saturantes y todos los decrementos a pushes no saturantes, el número de pushes no saturantes es a lo sumo:

$$(n-2)(2n-1) + 2mn(2n-1) = O(n^2m). \qquad \square$$

Número de pushes no saturantes para distancia máxima

Lema

El número de pushes no saturantes del algoritmo push-relabel de distancia máxima es $O(n^3)$.

Demostración.

- Un push no saturante desde un nodo v termina con v inactivo y con $d(w) \leqslant d(v)$ para todos los w. Así que para que v vuelva a ser activo debe haber algún reetiquetado.
- Si hubiera más de n pushes no saturantes sin reetiquetado el algorimo terminaría porque no habría más nodos activos. Luego, hay a lo sumo n pushes no saturantes entre cada par de etiquetados. Como hay $O(n^2)$ etiquetados, hay $O(n^3)$ pushes no saturantes. □

Observación

La cota no es ajustada. Tunçel (1991) probó que los algoritmos push-relabel de distancia máxima realizan a lo sumo $O(n^2\sqrt{m})$ pushes (y probó que esa cota se alcanza).

Implementación de los algoritmos push-relabel

- Para cada uno de los nodos v de la red mantenemos una lista L_v de los pares vw tales que $vw \in E$ o $wv \in E$ (o ambos).
- Junto a cada $vw \in L_v$ mantenemos $\bar{u}_{vw}$ y un link a wv en L_w (de manera que al hacer un push podamos actualizar $\bar{u}_{wv}$ en tiempo constante).
- Además en cada nodo mantenemos los valores de $f_x(v)$ y $d(v)$.
- Con estas estructuras podemos hacer un reetiquetado en tiempo $O(|L_v|)$ y un push en tiempo $O(1)$.
- ¿Cuánto tiempo nos vamos a pasar buscando arcos admisibles, decidiendo cuándo reetiquetar o eligiendo el siguiente nodo activo?

Arcos válidos y reetiquetado

Lema

Supóngase que $v \in V$ es un nodo activo y que el arco vw no es admisible. Entonces vw seguirá siendo no admisible hasta que v no sea reetiquetado.

Demostración.

Si vw no es admisible es porque $\mathrm{d}(w) \geqslant \mathrm{d}(v)$ o porque $\bar{\mathrm{u}}_{vw} = 0$. En el primer caso es claro que v tiene que ser reetiquetado para que vw se vuelva activo. Para el segundo, hace falta un push de w a v, para lo cual se requiere que en algún momento $\mathrm{d}(w) > \mathrm{d}(v)$ y luego para poder hacer el push en vw se requiere que $\mathrm{d}(v) > \mathrm{d}(w)$ y para eso es necesario que v sea reetiquetado. $\square$

Implementación de los algoritmos push-relabel (2)

- Al procesar cada nodo v recorremos la lista L_v realizando pushes en los arcos admisible hasta que v se vuelva inactivo o se nos acabe L_v.
- El procesamiento de v siempre termina con push que lo hace inactivo o con un reetiquetado.
- Si se da el primer caso, en el siguiente procesamiento de v no tiene sentido revisar en L_v los arcos que eran inactivos durante el procesamiento anterior (porque siguen siendo inactivos). Por lo que recomenzamos el procesamiento donde lo dejamos. (Para eso debemos recordar un arco *actual* dentro de la lista L_v).
- Si alcanzamos el final de L_v sin reetiquetar v entonces será posible reetiquetar v, porque todos los arcos que salen de v serán inadmisibles. Por lo tanto reetiquetaremos v al finalizar el recorrido de L_v.
- Esto no solo simplifica el algoritmo sino que es de gran ayuda al calcular la complejidad.

Análisis de complejidad

- Tiempo buscando arcos admisibles: Como v es reetiquetado a lo sumo $2n-1$ veces entonces habrá a lo sumo $O(n)$ recorridos de L_v. El tiempo buscando arcos admisibles es:
$$O\left(\sum_{v \in V \setminus \{r,s\}} n|L_v|\right) = O(mn).$$
- Tiempo reetiquetando: Como los $O(n)$ reetiquetados de v consumen $O(|L_v|)$ tiempo entonces es también $O(mn)$.
- Tiempo haciendo pushes: $O(N)$ con N la cantidad de pushes.
- Número de veces que elegimos un nodo activo: El número de veces que se busca un nuevo nodo activo es $O(N + n^2)$ porque cada vez que se elige un nodo hay un push o en un reetiquetado.

Como en el general se puede elegir un nodo activo en $O(1)$ (usando una cola de nodos activos) y $N = O(n^2m)$, vale lo siguiente.

Teorema

El algoritmo push-relabel para flujo máximo puede implementarse para correr en tiempo $O(n^2m)$.

Implementación de push-relabel de distancia máxima

- No es del todo obvio como mantener la lista de nodos activos.
- Queremos que el tiempo de corrida siga siendo $O(N)$ por lo que queremos poder elegir cada nodo en $O(1)$ promedio.
- Si usamos una sola lista nos costaría $O(n)$ por vez.
- Entonces para cada k mantenemos una lista D_k de todos los nodos *activos* con $d(v) = k$.
- Esta lista es doblemente enlazada permitiendo inserciones y borrados en tiempo constante.
- Además que para cada v mantenemos un link que nos permite acceder (si está activo) a su posición dentro de la lista D_k que le corresponde.
- Cada vez que un nodo es reetiquetado podemos borrarlo de una lista y moverlo a la otra.
- Cuando un push en vw vuelve activo a w lo agregamos a su lista y si v se vuele inactivo lo eliminamos de su lista.
- Nada de esto cambia la complejidad del algoritmo...

¿Cuánto tardamos en encontrar el siguiente nodo activo?

- Después de que v es reetiquetado, sigue siendo máximo, así que en ese caso no hay que consumir tiempo buscando un nuevo nodo activo de etiqueta máxima.
- Si no, sea $k = d(v)$ y recorremos $D_k, D_{k-1}, D_{k-2}, \ldots$ y paramos en $D_{k'}$ que será la primera no vacía y ahora $k = k'$.
- Notemos que el valor k de la etiqueta más grande de un nodo activo se incrementa con los reetiquetados por lo que todos los incrementos de k son a lo sumo $(n-2)(2n-1) = O(n^2)$ y la suma total de los decrementos $k - k'$ es también $O(n^2)$.
- Por lo tanto esta búsqueda del siguiente nodo activo no es un cuello de botella para el algoritmo.

Teorema

El algoritmo push-relabel de distancia máxima para flujo máximo puede implementarse para correr en tiempo $O(n^3)$.

De hecho con la cota de Tunçel nos queda que corre en tiempo $O(n^2\sqrt{m})$. (Ver Ahuja et el., *Network Flows* (1993), pp. 234–237.)

¡Gracias!