ETIQUETAS: MD2::(02)Networks

¿Que problema queremos resolver en los networks y flujos?

Dados los productores, los consumidores, y la red de transporte: ¿Cómo llevar la mayor cantidad posible de bienes desde los productores a los consumidores?

¿Cómo se modeliza la red de transporte?

Con una network

¿Que es un grafo dirigido?

Es un par $G = (V, E)$ donde V es un conjunto cualquiera y $E \subseteq V \times V$

En los grafos {{c2::no dirigidos}} los lados son {{c1::conjuntos}}

En los grafos {{c2::dirigidos}} los lados son {{c1::pares ordenados}}

¿En un grafo dirigido es lo mismo (x, y) que (y, x)?

No

¿En un grafo dirigido pueden existir los lados (x, y) y (y, x)?

Sí

¿Cómo se denota al lado (x, y) de un grafo dirigido?

$\overrightarrow{xy}$

¿Cómo se denotan los vecinos hacia adelante del vértice x en un grafo dirigido?

$$\Gamma^+(x)$$

¿Cómo se define $\Gamma^+(x)$ con x vértice de un grafo dirigido?

$$\{y \in V | \overrightarrow{xy} \in E\}$$

¿Cómo se denotan los vecinos hacia atras del vértice x en un grafo dirigido?

$$\Gamma^-(x)$$

¿Cómo se define $\Gamma^-(x)$ con x vértice de un grafo dirigido?

$$\{y \in V | \overrightarrow{yx} \in E\}$$

¿Que es una network? (Formal)

un triple (V, E, c) donde (V, E) es un grafo dirigido y $c : E \to \mathbb{R}_{>0}$

Una network es un grafo {{c1::dirigido}} con pesos {{c2::positivos}} en los lados

¿Cómo se llama a $c(\overrightarrow{xy})$?

Capacidad del lado $\overrightarrow{xy}$
Extra: Máxima capacidad de bienes que se pueden enviar por ese lado

¿Cuantos productores y cuantos consumidores hay en un flujo?

1 de cada uno

¿Cómo se modelizan los bienes que tendremos que mandar?

Con el flujo

Si g es {{c1::una función definida en los lados}} y A y B son {{c1::subconjuntos de vértices}}, entonces {{c1::$g(A, B)$}} denotará:
{{c2::

$$\sum_{\substack{x \in A \\ y \in B \\ (x,y) \in E}} g(\overrightarrow{xy})$$

}}

¿Cómo se define $out_g(x)$ dada una función g sobre lados y un vértice x? (Texto)

Es todo lo que “sale” de x por medio de g

¿Cómo se define $out_g(x)$ dada una función g sobre lados y un vértice x? (Formula)

$$g(\{x\}, V)$$

Extra: O también $g(\{x\}, \Gamma^+(x))$

¿Cómo se define $in_g(x)$ dada una función g sobre lados y un vértice x? (Texto)

Es todo lo que “entra” a x por medio de g

¿Cómo se define $in_g(x)$ dada una función g sobre lados y un vértice x? (Formula)

$$g(V, \{x\})$$

Extra: O también $g(\Gamma^-(x), \{x\})$

¿Que es un flujo?

Dado un network (V, E, c), y un par de vertices $s, t \in V$:
Un flujo de s a t es una función $f : E \to \mathbb{R}$ con 3 propiedades

¿Cuales son las 3 propiedades de un flujo?

- $0 \leq f(\overrightarrow{xy}) \leq c(\overrightarrow{xy}) \ \ \forall \overrightarrow{xy} \in E$
- $in_f(x) = out_f(x) \ \forall x \in V - \{s, t\}$
- $in_f(s) = out_f(t) = 0$

¿Que significa la propiedad $0 \leq f(\overrightarrow{xy}) \leq c(\overrightarrow{xy}) \ \ \forall \overrightarrow{xy} \in E$ de un flujo?

No vamos a transportar una cantidad negativa de un bien ni vamos a tranportar por encima de la capacidad de
transporte de un lado.

¿Que significa la propiedad $in_f(x) = out_f(x) \ \forall x \in V - \{s, t\}$ de un flujo?

El network no tiene “pérdidas”, cualquier vértice que no sea ni s ni t no consume ni produce bienes.

¿Que significa la propiedad $in_f(s) = out_f(t) = 0$?

El productor no consume y el consumidor no produce

¿Cómo se llama s en un flujo?

Fuente

¿Cómo se llama t en un flujo?

Resumidero

¿Cual es el valor de un flujo f?

$$out_f(s)$$

¿Cómo se denota el valor de un flujo f?

$v(f)$

¿Que es un flujo maximal de s a t dado un network N y vertices s, t?

Es un flujo f de s a t tal que $v(g) \leq v(f)$ para todo flujo g de s a t

¿Puede haber mas de un flujo maximal?

Sí

¿Cual es el flujo que siempre existe?

El flujo cuyo valor es 0

¿Puede haber infinitos flujos maximales?

Sí

¿Que es un camino dirigido de x a y?

Es una sucesión de vertices $x_0, x_1, x_2, \ldots, x_r$ con

- $x_0 = x$
- $x_r = y$
- $\overrightarrow{x_i x_{i+1}} \in E \;\; \forall i = 0, \ldots, r-1$

¿Cómo es el Algoritmo Greedy para hallar flujo maximal?

1. Comenzar con un f definida como $f(\overrightarrow{xy}) = 0 \ \forall \ \overrightarrow{xy} \in E$
2. Buscar un camino dirigido de s a t tal que $f(\overrightarrow{xy}) < c(\overrightarrow{xy})$ para todo $\overrightarrow{xy} \in E$ lado del camino
3. Calcular $\varepsilon = min\{c(\overrightarrow{xy}) - f(\overrightarrow{xy})\}$ entre todos los $\overrightarrow{xy} \in E$ lados del camino
4. Aumentar el valor de f en ε
5. Repetir 2 hasta que no se puedan hallar mas caminos con esas condiciones.

¿El Algoritmo Greedy para hallar flujo maximal siempre halla el flujo maximal?

No

¿Que es la capacidad residual de un lado $\overrightarrow{xy}$?

$$c(\overrightarrow{xy}) - f(\overrightarrow{xy})$$

¿Que es un corte?

Es un subconjunto de los vértices que tiene a s pero no tiene a t

¿Cómo se denota la capacidad de un corte?

$cap(S)$

¿Cómo se define la capacidad de un corte?

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

¿Cuando un corte es minimal?

Si su capacidad es la menor de las capacidades de todos los cortes.
Extra: S es un corte minimal si $cap(S) \leq cap(T)$ para todo corte T

¿Que es un camino aumentante o camino de Ford-Fulkerson?

Es una sucesión de vértices $x_0, x_1, \dots, x_r$ tales que:

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

¿Cuales son los lados de tipo I o lados forward?

Lados $\overrightarrow{x_i x_{i+1}} \in E$ tales que $f(\overrightarrow{x_i x_{i+1}}) < c(\overrightarrow{x_i x_{i+1}})$

¿Cuales son los lados de tipo II o lados backward?

Lados $\overrightarrow{x_{i+1} x_i} \in E$ y $f(\overrightarrow{x_{i+1} x_i}) > 0$

¿Cómo es el algoritmo de Ford-Fulkerson para hallar flujo maximal?

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

¿Que dice el teorema de que FordFulkerson mantiene “flujicidad”?

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 $v + \varepsilon$

¿Que dice el teorema Max Flow Min Cut?

- Si f es un flujo y S es un corte, entonces $v(f) = f(S, \overline{S}) - f(\overline{S}, S)$
- El valor de todo flujo es menor o igual que la capacidad de todo corte
- 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

¿Cual es la complejidad de Greedy para encontrar flujo maximal?

$O(m^2)$

Si f es un flujo y S es un corte, entonces $v(f) =$ {{c1::$f(S, \overline{S}) - f(\overline{S}, S)$}}

{{c3::El valor de todo flujo}} es {{c1::menor o igual}} que {{c2::la capacidad de todo corte}}

Si el algoritmo de Ford-Fulkerson termina, termina {{c1::con un flujo maximal}}

¿Que dice el teorema de la integralidad?

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

En un network {{c2::donde todas las capacidades sean enteros}}, Ford-Fulkerson {{c1::siempre termina}} y {{c1::el flujo maximal resultante es un flujo entero}}

¿Cual es la complejidad de Edmonds-Karp?

La complejidad del algoritmo de Edmonds-Karp es $O(nm^2)$

¿Cuando un lado $f(\overrightarrow{xy})$ se vuelve crítico durante la construcción de uno de los flujos intermedios (f_{k+1})?

Si para la construcción de f_{k+1} pasa una de las dos cosas siguientes:

1. Se usa el lado en forma forward, saturandolo ($f_k(\overrightarrow{xy}) < c(\overrightarrow{xy})$, pero luego $f_{k+1}(\overrightarrow{xy}) = c(\overrightarrow{xy})$)
2. Se usa el lado en forma backward, vaciandolo ($f_k(\overrightarrow{xy}) > 0$, pero luego $f_{k+1}(\overrightarrow{xy}) = 0$)

¿Que es la distancia entre x y z relativa a f?

La longitud del menor $f-$camino aumentante entre x y z

¿Cual es la distancia entre x y z relativa a f si no existe ese camino?

∞

¿Cual es la distancia entre x y z relativa a f si $x = z$?

0

¿Cómo se denota la distancia entre x y z relativa a f?

$$d_f(x, z)$$

¿Que denota $d_k(x)$?

$$d_{f_k}(s, x)$$

¿Que denota $b_k(x)$?

$$d_{f_k}(x,t)$$

¿Que dice el lema de las distancias?

- $d_k(x) \leq d_{k+1}(x) \ \forall x$
- $b_k(x) \leq b_{k+1}(x) \ \forall x$

¿Cual es el esquema básico de Dinitz?

1. Construir un network auxiliar (usando BFS).
2. Encontrar un flujo bloqueante en el network auxiliar.
3. Usar ese flujo bloqueante del network auxiliar para modificar el flujo en el network original.
4. Repetir [1] con el nuevo flujo, hasta que, al querer construir un network auxiliar, no llegamos a t

¿Cuando un flujo f es bloqueante?

Si todo camino dirigido desde s a t tiene al menos un lado saturado ($c(\overrightarrow{xy}) = f(\overrightarrow{xy})$)
Extra: si cuando queremos usar Greedy en el network, no llegamos a t.

¿Que característica tiene el network auxiliar?

Es "por niveles"

¿Que es un network por niveles?

Es un network tal que el conjunto de vértices esta dividido en subconjuntos V_i (los “niveles”) tales que sólo existen lados entre un nivel y el siguiente.
Extra: $\overrightarrow{xy} \in E \Rightarrow \exists i : x \in V_i, y \in V_{i+1}$

¿Cómo se define el conjunto de vértices del network auxiliar de Dinic?

Es $V = \bigcup_{i=0}^{r} V_i$, donde los V_i son:

- Sea $r = d_f(s,t)$
- Para $i = 0, 1, \ldots, r-1$, definimos $V_i = \{x : d : f(s,x) = i\}$
- Definimos $V_r = \{t\}$
  Extra: $V_0 = \{s\}$

¿Cómo se define el conjunto de lados del network auxiliar de Dinic?

$\overrightarrow{xy}$ es un lado del network auxiliar si:
$x \in V_i, y \in V_{i+1}$ y cumple alguna de estas propiedades:

1. $\overrightarrow{xy}$ es un lado del network original con $f(\overrightarrow{xy}) < c(\overrightarrow{xy})$
2. $\overrightarrow{yx}$ es un lado del network original con $f(\overrightarrow{yx}) > 0$

¿Cómo se definen las capacidades de los lados del network auxiliar de Dinic?

La capacidad del lado $\overrightarrow{xy}$ será:

- $c(\overrightarrow{xy}) - f(\overrightarrow{xy})$ si $\overrightarrow{xy}$ es un lado del network original con $f(\overrightarrow{xy}) < c(\overrightarrow{xy})$
- $f(\overrightarrow{yx})$ si es un lado del network original con $f(\overrightarrow{yx}) > 0$

Como el network auxiliar es un network por niveles, todos los caminos de un mismo network auxiliar deben tener {{c1::la misma longitud}}

Al usar Dinic el network auxiliar debe tener {{c1::mayor cantidad de niveles}} que el network auxiliar anterior
Extra: Excepto que sea el último network, el que no llega a t

¿Cual es la diferencia entre Dinic y Dinic-even?

El método que se utiliza para podar el network auxiliar

¿Que orden tiene la cantidad de networks auxiliares de Dinic?

$O(n)$

¿Cuando y como se poda el network auxiliar en el Dinic original?

Al principio y luego de cada camino usado, borrando todos los vertices que no tengan lados de salida

¿Cuando y como se poda el network auxiliar en Dinic-Even?

Al correr DFS, cuando llegamos a un vértice que no tiene salida se borran todos los lados por los cuales tenemos que retroceder.

La distancia entre s y t {{c1::aumenta}} entre networks auxiliares consecutivos.

En cualquier corrida de un algoritmo “tipo” Dinic, hay a lo sumo {{c1::n}} {{c2::networks auxiliares}}

La complejidad de un algoritmo "tipo" Dinic es {{c1::
$O(n \cdot (m + \text{complejidad de hallar flujo bloqueante}))$}}

¿Cual es la complejidad de los algoritmos de Dinitz original y de la versión de Even?

Es $O(n^2 m)$

¿Que habrá siempre en los ejercicios de parcial de FF?

Al menos un lado backward

Los caminos de EK no {{c1::disminuyen}} en cantidad de vertices

¿Un network sin lados sigue siendo network?

Sí

¿Cómo hay que resolver un problema de caja negra de max flow?

1. Dar una forma de transformer el seudo-network en un network usual
2. Dar una forma de convertir un flujo usual sobre ese network usual en un seudo-flujo sobre el seudo-network
3. Probar que si el flujo es maximal, el suedo-flujo también es seudo-maximal sobre el seudo-network

¿Cómo probar que si el flujo es maximal, el suedo-flujo también es seudo-maximal sobre el seudo-network en un problema de caja negra de max flow?

Por el absurdo

1. Suponer que existe un seudo-flujo con valor mayor al obtenido
2. Usando ese seudo-flujo construir un flujo que tenga mayor valor que el obtenido de la caja negra
3. Contradicción porque estamos suponiendo que la caja negra funciona bien

¿Cual es el total de cortes que tiene un network?

2^{n-2}, siendo n cantidad de vertices del network

¿Cómo esta definido el corte minimal S asumiendo que ya no existen caminos aumentantes entre s y t?

$S = \{s\} \cup \{x : \exists\ f\text{-camino aumentante entre s y x}\}$

¿Que checkeos se pueden hacer para verificar si el valor del flujo maximal es correcto?

- $\sum$ todo el flujo que sale de $s = v(f)$ y $\sum$ todo el flujo que entra a $t = v(f)$
- $\sum \varepsilon = v(f)$, osea, sumar todos los ε que fueron agregandose a f
- $cap(S) = v(f)$ con S corte minimal

En el algoritmo de EK, cantidad de {{c2::vértices de los caminos hallados en la iteración i}} {{c1::$\leq$}} cantidad de {{c2::vértices de los caminos hallados en la iteración $i+1$}}

¿Cuando 2 flujos f y g son distintos?

Cuando existe $\overrightarrow{xy} \in E$ tal que $f(\overrightarrow{xy}) \neq g(\overrightarrow{xy})$

¿Como es la idea que utiliza de manera genérica los algoritmos que conocemos para una demostración de flujos?

1. Dar un network genérico/con cierta caracteristica
2. Correr 1 iteración de X algoritmo
3. El resultado de eso es flujo aún y el flujo satisface Y propiedad

¿A partir de cual búsqueda de camino aumentante en EK podemos tener un lado backward?

A partir de la segunda

¿Cuales son los pasos para hacer un ejercicio con el algoritmo de EK?

1. Escribir las capacidades de cada lado
2. Realizar las búsquedas de caminos aumentantes para encontrar ε hasta no poder más
3. Chequear que $v(f)$ es correcto
4. Definir $f(\overrightarrow{xy}) = c(\overrightarrow{xy}) - (\text{lo que quedo sin tachar})$

¿Cual es el corte minimal al correr EK?

El conjunto de vertices que quedan al ejecutar el último BFS
Extra: No se puede seguir avanzando y no llego a t

¿Que realiza la función `AVANZAR(x)` de dinic-even?

Elegimos algún vecino y de $\Gamma^+(x)$, agregamos $\overrightarrow{xy}$ al camino y cambiamos $x = y$

¿Que realiza la función `RETROCEDER(x)` de dinic-even?

Tomamos z el vértice anterior a x en la pila, borramos $\overrightarrow{zx}$ del camino y del network auxiliar, y hacemos $x = z$

¿Que realiza la función `INCREMENTAR` de dinic-even?

Calcula cuanto se puede mandar por el camino construido, manda eso y borra del network cualquier lado que haya sido saturado

¿Cual es la complejidad de la función `INCREMENTAR` de dinic-even?

$O(n)$

¿Cual es la complejidad de la función `RETROCEDER(x)` de dinic-even?

$O(1)$

¿Cual es la complejidad de la función `AVANZAR(x)` de dinic-even?

$O(1)$

¿Que tipo de flujo encuentra greedy?

Un flujo bloqueante

En Dinic el invariante de las funciones parciales g obtenidas durante la construcción del flujo bloqueante en un network auxiliar es {{c1::que sea flujo}}

En Dinic detenemos la construcción de las funciones parciales g cuando g es {{c1::bloqueante}}

En Wave el invariante de las funciones parciales g obtenidas es {{c1::que sea bloqueante}}

En Wave detenemos la construcción de las funciones parciales g cuando g es {{c1::flujo}}

¿Que es un vértice bloqueado en el algoritmo de Wave?

Vértices que no pueden mandar mas flujo hacia adelante

¿Que vértices devuelven flujo en una ola hacia atras de Wave?

Los bloqueados

¿Que hace con los vértices bloqueados la ola hacia atras de Wave?

Los balancea

¿En que orden los vértices la ola hacia atras de Wave?

Desde t hacia s

¿En que orden recorre los vértices la ola hacia adelante de Wave?

Desde s hacia t

¿Que intenta hacer la ola hacia adelante de Wave?

Balancear los vertices

¿Cómo balancea los vértices la ola hacia atras de Wave?

Devolviendo flujo

¿Cómo balancea los vértices la ola hacia adelante de Wave?

Mandando flujo hacia adelante

¿Que hace la ola hacia adelante de Wave al no poder balancear un vertice?

Lo bloquea

¿Cual es la complejidad de Wave?

$O(n^3)$

En el algoritmo de Dinic, cantidad de {{c2::vértices de los caminos hallados en la iteración i }} {{c1::$<$}} cantidad de {{c2::vértices de los caminos hallados en la iteración $i+1$}}

¿Cómo modificar avanzar en el algoritmo de Dinic para que en networks con todas capacidades 1 corra en tiempo $O(nm)$? (Eliminar incrementar)

Avanza 1 vértice en el camino, asigna 1 al valor del flujo en ese lado y elimina el lado del network auxiliar

¿Cómo modificar retroceder en el algoritmo de Dinic para que en networks con todas capacidades 1 corra en tiempo $O(nm)$? (Eliminar incrementar)

Asigna 0 al valor del flujo en ese lado y retrocede en la pila

¿Cómo es el orden de envío de flujo en la ola hacia adelante en un ejercicio de wave de Tarjan?

Por niveles
Extra: Primero el nivel de s, después el nivel 1, después el 2 y así

¿Cómo es el orden de envío de flujo en la ola hacia atras en un ejercicio de wave de Tarjan?

Por niveles de manera inversa
Extra: Primero el nivel de t, después el nivel $r-1$, después el $r-2$ y así (Recordarque solo los bloqueados devuelven flujo)