Definiciones

miércoles, 13 de abril de 2022 13:54

Teoremas

miércoles, 13 de abril de 2022 14:52

Algoritmo de Dinitz:

Input:

(V, E, c) un network

$s, t \in V$

Output:

$f : E \to \mathbb{R}_{\geq 0}$

Algoritmo:

$f(e) \leftarrow 0$

while el f-network auxiliar llega a t

$(V', E', c') \leftarrow f$-network auxiliar que llega a t

$f' \leftarrow$ flujoBloqueante(V', E', c')

$$f(\overrightarrow{xy}) \leftarrow \begin{cases} \overrightarrow{xy} \in E' \to f(\overrightarrow{xy}) + f'(\overrightarrow{xy}) \\ \overrightarrow{yx} \in E' \to f(\overrightarrow{xy}) - f'(\overrightarrow{xy}) \\ \text{si no} \to f(\overrightarrow{xy}) \end{cases}$$

La distancia a t en los networck auxiliares va aumentando en cada iteración

Algoritmo flujoBloquanteDFS: 🛠️ (incompleto)

input:

(V, E, c) un network por niveles

$s, t \in V$

Output:

$f : E \to \mathbb{R}_{\geq 0}$

Algoritmo:

$f(e) \leftarrow 0$

while(true)

camino $\leftarrow$ [s]

$x \leftarrow s$

while $t \notin$ camino

if $\Gamma_f^+(x) \neq \emptyset$

sea $y \in \Gamma_f^+(x)$

camino $=$ camino $+ [y]$

else

Demostraciones

viernes, 24 de junio de 2022 16:26

La distancia en networks auxiliares aumenta:

Sea:

(V, E, c) un network

$s, t \in V$

f_i el flujo en la iteración i

N_i el network auxiliar de la iteración i de Dinitz

(o sea, f_i se usa para costruir N_i)

$$d_{f_i}(t) < d_{f_{i+1}}(t)$$

Demostración:

① $d_{f_i}(t) \leq d_{f_{i+1}}(t)$

Esto es cierto por la demostración de Ford-Fulkerson, y por que el flujo se puede escribir como suma de caminos aumentantes.

② $\forall x \in V - \{t\} : x \in V(N_i) \Leftrightarrow d_{f_i}(x) < d_{f_i}(t)$

Esto es así porque se usa BFS para construir el network auxiliar

Si $t \notin N_{i+1}$:

Si existe la iteración $i + 1$ es porque $d_{f_i}(t) < \infty$, $t \notin N_{i+1} \Rightarrow \mathrm{d}_{f_{i+1}}(t) = \infty$

Así que en este caso es trivialmente cierto

Supongamos ahora que $t \in N_{i+1}$

Sea:

$X = x_0, x_1, x_2, \ldots, x_r$ un camino de s a t en N_{i+1}

Notar que:

$x_0 = \mathrm{s}$

$x_r = t$

$r = d_{f_{i+1}}(t)$

③ X no es un camino en N_i

Ya que si lo fuera, se habría saturado en f_i y por ende no podría serlo en N_{i+1}

④ $\langle \exists j \in \{0, \dots, r\} : x_j \notin V(N_i) \rangle \vee \langle \exists j \in \{0, \dots, r-1\} : \overrightarrow{x_j x_{j+1}} \notin E(N_i) \rangle$

Ya que si no se contradeciría ②

Divido en los casos de ③:

Caso $\langle \exists j \in \{0, \dots, r\} : x_j \notin V(N_i) \rangle$:

Sea j que satisface el $\exists$

$$
\begin{aligned}
& d_{f_i}(t) \\
\le & \{ ②, x_j \notin V(N_i) \Rightarrow d_{f_i}(x_j) \ge d_{f_i}(t) \} \\
& d_{f_i}(x_j) \\
\le & \{\text{Por teorema de las distancias en la demostración de Ford-Fulkerson}\} \\
& d_{f_{i+1}}(x_j) \\
= & \{X \text{ es un camino en en } N_{i+1}\text{, que es un network por niveles}\} \\
& j \\
< & \{t \in V(N_i) \Rightarrow x_j \ne t\} \\
& d_{f_{i+1}}(t)
\end{aligned}
$$

Caso $\langle \exists j \in \{0, \dots, r-1\} : \overrightarrow{x_j x_{j+1}} \notin E(N_i) \rangle$:

Sea j el mínimo elemento que satisface el $\exists$:

⑤ $d_{f_i}(x_{j+1}) < d_{f_{i+1}}(x_{j+1})$

El $\le$ vale por la demostración de Ford-Fulkerson, pruebo que es $<$ por absurdo suponiendo que es $=$:

(5.1) $d_{f_{i+1}}(x_{j+1}) = j + 1$

Esto es así porque X es el camino mas corto en N_{i+1}

(5.2) $d_{f_i}(x_{j+1}) = j + 1$

Esto es cierto por (5.1) y por que $d_{f_i}(x_{j+1}) = d_{f_{i+1}}(x_{j+1})$

(5.3) $d_{f_i}(x_j) = j$

Esto es cierto porque $\overrightarrow{x_j x_{j+1}}$ es el primer lado del camino que no está en N_i, así que los anteriores si están en N_i (y N_i es por capas)

(5.4) $\overrightarrow{x_{j+1} x_j} \in E(N_i)$

Esto es cierto que $\overrightarrow{x_j x_{j+1}} \in E(N_{i+1}) \wedge \overrightarrow{x_j x_{j+1}} \notin E(N_i)$ implica que el lado

$\overrightarrow{x_j x_{j+1}}$ se des-saturo, y por ende el lado contrario tiene que haber estado

Como N_i es un network por capas, $\overrightarrow{x_{j+1} x_j} \in E(N_i)$ significa que x_{j+1} está en la capa enterior que x_j, pero eso contradice a (5.2) y (5.3)
Queda probado $d_{f_i}(x_{j+1}) < d_{f_{i+1}}(x_{j+1})$

Ahora hago la prueba:

$$
\begin{aligned}
& d_{f_i}(t) \\
= \; & \\
& d_{f_i}(x_{j+1}) + b_{f_i}(x_{j+1}) \\
< \; & \{5\} \\
& d_{f_{i+1}}(x_{j+1}) + b_{f_i}(x_{j+1}) \\
\leq \; & \{\text{Por la preba de Ford-Fulkerson}\} \\
& d_{f_{i+1}}(x_{j+1}) + b_{f_{i+1}}(x_{j+1}) \\
= \; & \\
& d_{f_{i+1}}(t)
\end{aligned}
$$

Complejidad de encontrar flujo bloqueante con Dinitz:

$O(nm)$

Demostración:

El algoritmo consiste en primero podar, y luego, mientras haya un camino, agregar el camino y podar.
Para podar se recorre el network por niveles, y se van borrando los vértices que no lleven a ningún lado.

Pongámosle nombres:
$B(x) =$ borrar el vértice x
$P =$ podar
$EC =$ encontrar un camino
$A =$ agregar un camino

Sea:

PS_i = el conjunto de vértices a podar en la iteración i

Voy probando algunas cosas:

1 $O(B(x)) = O(|\Gamma(x)|)$

Esto porque para borrar un vértice hay que borrar todos los lados en los que está

2 $O(EC) = O(n)$

Esto porque se usa DFS, y el haber podado asegura que el camino se encuentre sin tener que retroceder

3 Hay a lo sumo m caminos

Esto es cierto porque en cada camino se satura por lo menos un lado, y cada lado se puede saturar una sola vez

4 $O(A) = O(n)$

Esto porque es solo recorrer el camino cambiando los valores del flujo, y el camino tiene a lo sumo n elementos

5 $O(P) = O(n) * \sum_{x \in PS_i} O(|\Gamma(x)|)$

Esto es porque para saber que vértices borrar hay que recorrer todo el network (eso es $O(n)$), y además, hay que borrar algunos vértices (por 1, $O(B(x)) = O(|\Gamma(x)|)$)

6 La complejidad de cada iteración es $O(EC) + O(A) + O(P)$

Ya que en cada iteración hay que encontrar un camino aumentante, sumarlo y podar el network

Con esto la complejidad es:
Sea r la cantidad de iteraciones

$$\sum_{i=0}^{r-1} O\big(O(EC) + O(A) + O(P)\big)$$

= {2, 4, 5}

$$\sum_{i=0}^{r-1} \left(O(n) + O(n) + O(n) \sum_{x \in PS_i} O(|\Gamma(x)|) \right)$$

=

$$rO(n) + O(n)\sum_{i=0}^{r-1}\left(\sum_{x \in PS_i} O(|\Gamma(x)|)\right)$$

= {Como cada vértice se borra a lo sumo una vez, los PS_i son disjuntos, y su union es a lo sumo V}

$$rO(n) + O(n)\sum_{x \in V} O(|\Gamma(x)|)$$

= {Lema del apretón de manos}

$$rO(n) + O(n)O(2m)$$

= {Por 3, $r < m$}

$$O(nm) + O(nm)$$

=

$$O(nm)$$

Complejidad de encontrar el flujo bloqueante con Dinitz-Even:

$$O(nm)$$

Demostración:

El algoritmo consiste en hacer DFS, pero borrando los lados cuando se ve que no llegan a ningún lado, hasta que no se pueda llegar a t

Pongámosle nombre a las cosas:
A = abanzar en el DFS (o sea, agregar un vértice nuevo)
R = retroceder en el DFS (incullendo el borrar el lado, ya que si hace falta retroceder es por que no lleva a ningún lado)
C = encontrar el camino aumentante (o sea, hacer el DFS)
I = sumar al flujo el camino aumentante (I de incrementar)

Sea:
C_i = el C de la iteración i
CR_i = el conjunto de lados borrados en la iteración i

Pruebo algunas cosas:

1 $O(A) = O(1)$

Ya que solo consiste en agregar un vértice en DFS

② $O(B) = O(1)$
B consiste en tachar un vértice en DFS y borrar un lado.
Tachar un vértice en DFS es $O(1)$
Borrar un lado requiere borrarlo en ambos vértices, pero también es $O(1)$
Así que en total es $O(1)$

③ $O(I) = O(n)$
Porque el camino aumentante tiene a lo sumo n elementos

④ R se llama a lo sumo m veces entotal
Esto es así porque R borra un lado, cada lado puede ser borrado una sola ves, y hay m lados en total

⑤ Hay a lo sumo m caminos
Esto es así, porque con cada camino se satura al menos un lado, el cuál ya no se puede volver a usar, y por ende, después de m caminos se tienen que haber agotado todos los lados

⑥ $O(C_i) = O(n + |CR_i|)$
Encontrar un camino requiere como base la cantidad de capas del network, pero ademas, por cada ves que se retrocede hay que avanzar uno extra
Entonces, si k es la cantidad de capas del network, esto es:
$O(k + |CR_i| + |CR_i|)$
Y como $k \leq n$, tengo $O(C_i) = O(n + 2|CR_i|) = O(n + |CR_i|)$

⑦ Cada iteración es $O(C_i) + O(I)$
En cada iteración hay que encontrar el camino y luego sumarlo

Con esto calculo la complejidad total:
Sea r la cantidad total de iteraciones

Por ⑦:
$$\sum_{i=0}^{r-1} O\big((C_i) + O(I)\big)$$
$= \{$③ y ⑥$\}$

$$\sum_{i=0}^{r-1} \mathrm{O}\big((n + |CR_i|) + \mathrm{O}(n)\big)$$

$=$

$$\sum_{i=0}^{r-1} \mathrm{O}(n + |CR_i|)$$

$=$

$$r\mathrm{O}(n) + \mathrm{O}\left(\sum_{i=0}^{r-1} |CR_i|\right)$$

$=$ {Como por ④ cada lado se borra a lo sumo una vez, los CR_i son todos disjuntos}

$$r\mathrm{O}(n) + \mathrm{O}\left(\left|\bigcup_{i=0}^{r-i} CR_i\right|\right)$$

$=$ {Los CR_i son a lo sumo todos los lados}

$$r\mathrm{O}(n) + \mathrm{O}(m)$$

$=$ {Por ⑤, $r < m$}

$$\mathrm{O}(mn + m)$$

$=$

$$\mathrm{O}(mn)$$

1a)

miércoles, 13 de abril de 2022 15:13

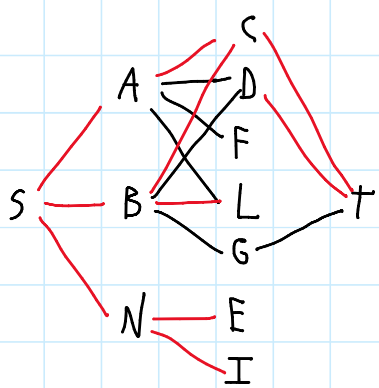
(1) En los siguientes networks, hallar un flujo maximal y un corte minimal usando el algoritmo de Dinic. Si tiene elección, use el orden alfabetico $s < t < A < B < ....$ Calcular el valor del flujo maximal obtenido y la capacidad del corte minimal obtenido.

(a) sA 144 sB 96 sN 150 AC 100 AD 70 AF 85 AL 17 BC 10 BD 17 BG 102 BL 35 Ct 80 CJ 5 Dt 80 EC 100 ED 15 EG 10 FH 15 FK 100 Gt 80 Ht 20 HM 40 ID 22 IG 10 JA 5 Kt 120 LH 30 LK 60 Mt 30 NE 110 NI 40

	C		1	2	3	4	5	6	7	8	9	10	11	12	13	F
sA	144	~~0~~	~~80~~	144												144
sB	96	~~0~~			~~76~~	96										96
sN	150	~~0~~					~~15~~	~~80~~	~~85~~	~~90~~	~~95~~	~~105~~	~~112~~	~~127~~	137	137
AC	100	~~0~~	~~80~~				~~65~~	0								0
AD	70	~~0~~		~~64~~					~~59~~	~~44~~	~~39~~		32			32
AF	85	~~0~~					~~75~~	~~80~~	85							85
AL	17	~~0~~								~~5~~	~~10~~		17			17
BC	10	0														0
BD	17	~~0~~			~~16~~									1		1
BG	102	~~0~~				80						~~70~~			60	60
BL	35	~~0~~										~~10~~		~~25~~	35	35
Ct	80	~~0~~	80													80
CJ	5	0														0
Dt	80	~~0~~		~~64~~	80											80
EC	100	~~0~~					~~15~~	80								80
ED	15	~~0~~							~~5~~	~~10~~	15					15
EG	10	~~0~~										10				10
FH	15	~~0~~					15									15
FK	100	~~0~~						~~65~~	70							70
Gt	80	~~0~~				80										80
Ht	20	~~0~~					~~15~~			20						20
HM	40	0														0
ID	22	~~0~~											~~7~~	22		22

HM	40	0		0
ID	22	~~0~~	~~7~~ 22	22
IG	10	~~0~~	10	10
JA	5	0		0
KT	120	~~0~~	~~65~~ ~~70~~ ~~75~~ ~~85~~ ~~92~~ ~~107~~ 117	117
LH	60	~~0~~	5	5
LK	60	~~0~~	~~5~~ ~~15~~ ~~22~~ ~~37~~ 47	47
MT	30	0		0
NE	110	~~0~~	~~75~~ ~~80~~ ~~85~~ ~~90~~ ~~95~~ 100	100
NI	40	~~0~~	~~7~~ ~~22~~ 32	32

Network auxilior 1:



Primer camino: SACT
$\varepsilon = 80$

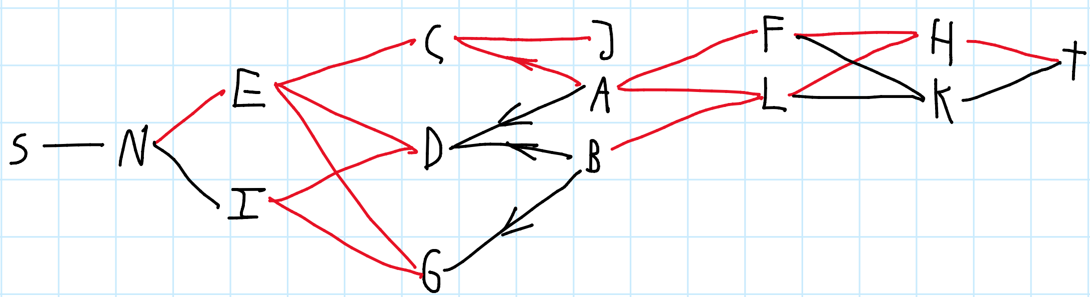
Segundo camino: SADT
$\varepsilon = 64$

Tercer camino: SBDT
$\varepsilon = 16$

Cuarto camino: S B G t
$\varepsilon = 80$

No hay mas caminos

Network auxilar 2:



Quinto camino: S N E $\overleftarrow{CA}$ F H t
$\varepsilon = 15$

Sexto camino: S N E $\overleftarrow{CA}$ F K t
$\varepsilon = 65$

Septimo camino: S N E $\overleftarrow{DA}$ F K t
$\varepsilon = 5$

Octavo camino: S N E $\overleftarrow{DA}$ L H t
$\varepsilon = 5$

Noveno camino: S N E $\overleftarrow{DA}$ L K t
$\varepsilon = 5$

Decimo camino: S N E $\overleftarrow{GB}$ L K t
$\varepsilon = 10$

Onceabo camino: $SNI\overleftarrow{D}ALKt$

$\varepsilon = 7$

Doceabo camino: $SNI\overleftarrow{D}BLKt$

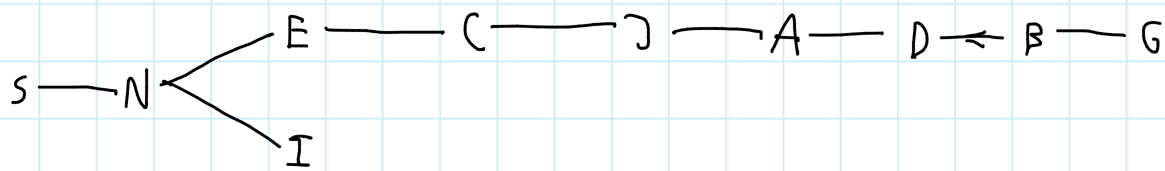
$\varepsilon = 15$

Treceabo camino: $SNI\overleftarrow{G}BLKt$

$\varepsilon = 10$

No hay mas caminos

Network auxilar 3:



No se puede llegar a t

corte minimal:

$$S = \{s, N, E, I, C, J, A, D, B, G\}$$

$$v(f) = out_f(s)$$

$$= f(\overrightarrow{sA}) + f(\overrightarrow{sB}) + f(\overrightarrow{sN})$$

$$= 144 + 96 + 737$$

$= 377$

$$\mathrm{cap}(S) = c(S, \bar{S})$$

$$= \sum_{\overrightarrow{xy} \in S \times \bar{S} \cap E} c(\overrightarrow{xy})$$

$$= c(\overrightarrow{CT}) + c(\overrightarrow{AF}) + c(\overrightarrow{AL}) + c(\overrightarrow{DT}) + c(\overrightarrow{BL}) + c(\overrightarrow{GT})$$

$$= 80 + 85 + 17 + 80 + 35 + 80$$

$$= 377$$

$$v(f) = \mathrm{cap}(S)$$

$\Rightarrow$ f es maximal

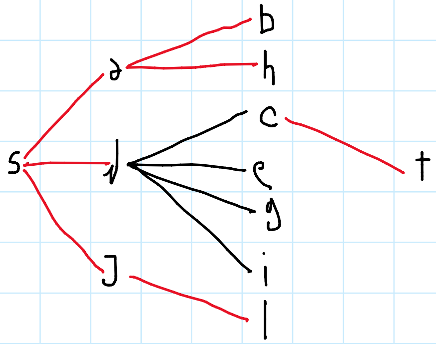
1b)

sábado, 16 de abril de 2022 9:41

(b) sa 15 sd 20 sj 7 ab 17 ah 5 bc 15 ct 20 dc 26 de 5 dg 10 di 6 ef 5 ek 2 ft 5 gk 10 gm 3 go 1 hn 4 if 4 jl 7 kt 10 lb 5 ln 4 mt 1 nc 6 ot 10

sa	15	~~0~~		~~5~~	15			15
sd	20	~~0~~	20					20
sj	7	~~0~~				~~1~~	2	2
ab	17	~~0~~		~~5~~	15			15
ah	5	0						0
bc	15	~~0~~		~~5~~	15			15
ct	20	~~0~~	20					20
dc	26	~~0~~	~~20~~	~~15~~	~~5~~	~~4~~	3	3
de	5	~~0~~		5				5
dg	10	~~0~~			10			10
di	6	~~0~~				~~1~~	2	2
ef	5	~~0~~		~~5~~		~~4~~	3	3
ek	2	~~0~~				~~1~~	2	2
ft	5	~~0~~		5				5
gk	10	~~0~~			~~10~~	~~9~~	8	8
gm	3	~~0~~				1		1
go	1	0					1	1
hn	4	0						0
if	4	~~0~~				~~1~~	2	2
jl	7	~~0~~				~~1~~	2	2
kt	10	~~0~~			10			10
lb	5	0						0
ln	4	~~0~~				~~1~~	2	2
mt	1	~~0~~				1		1
nc	6	~~0~~				~~1~~	2	2
ot	10	~~0~~					1	1

1º network auxiliar:

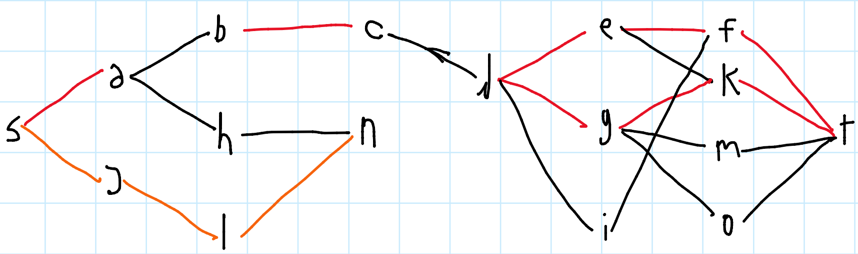


1º camino: $sdct$

$\varepsilon = \min\{20, 26, 20\} = 20$

No hay mas caminos

2º network auxilar:



2º camino: $sab\overleftarrow{cd}eft$

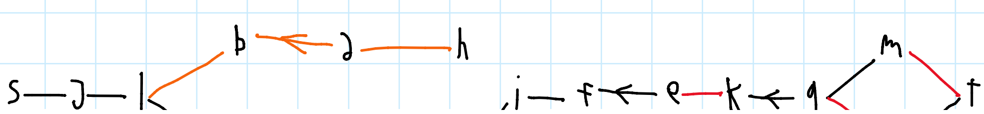
$\varepsilon = \min\{15, 17, 15, 20, 5, 5, 5\} = 5$

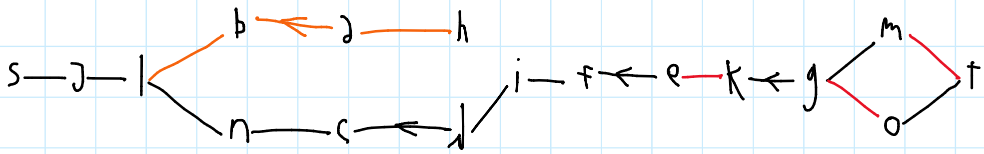
3º camino: $sab\overleftarrow{cd}gkt$

$\varepsilon = \min\{10, 12, 10, 15, 10, 10, 10\} = 10$

No hay mas caminos

3º network auxilar





4^{o} camino: $s\,j\,l\,n\,c\,\overleftarrow{d}\,i\,f\,\overleftarrow{e}\,\overleftarrow{k}\,g\,m\,t$

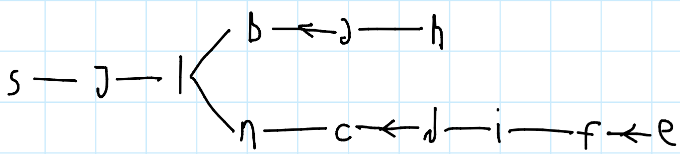
$\varepsilon = \min\{7, 7, 4, 6, 5, 6, 4, 5, 2, 10, 3, 1\} = 1$

5^{o} camino: $s\,j\,l\,n\,c\,\overleftarrow{d}\,i\,f\,e\,\overleftarrow{k}\,g\,o\,t$

$\varepsilon = \min\{6, 6, 3, 5, 4, 5, 3, 4, 1, 9, 1, 10\} = 1$

No hay mas caminos

4^{o} network auxilar:



No hay mas caminos

corte minimal:

$$S = \{s, j, l, b, a, h, n, c, d, i, f, e\}$$

$$v(f) = out_f(s)$$

$$= f(\overrightarrow{sa}) + f(\overrightarrow{sd}) + f(\overrightarrow{sj})$$

$$= 15 + 20 + 2$$

$$= 37$$

$$\mathrm{cap}(S) = c(S, \bar{S}) \quad S = \{s, a, l, b, a, h, n, c, d, i, f, e\}$$

$$= \sum_{\overrightarrow{xy} \in S \times \bar{S} \cap E} c(\overrightarrow{xy})$$

$$= c(\overrightarrow{cT}) + c(\overrightarrow{dg}) + c(\overrightarrow{fT}) + c(\overrightarrow{eK})$$

$$= 20 + 10 + 5 + 2$$

$$= 37$$

$$v(f) = \mathrm{cap}(S)$$

$\Rightarrow$ f es maximal

1c)

miércoles, 20 de abril de 2022 15:07

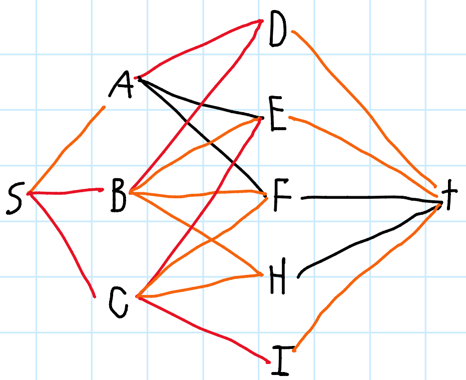
(c) sA 20 sB 69 sC 145 AD 14 AE 19 AF 18 BD 9 BE 4 BF 14 BH 1 CE 190 CF 4 CH 20 CI 20 Dt 9 DH 8 DI 1 DJ 7 Et 16 EH 2 EI 16 EJ 7 Ft 146 GI 5 Ht 25 It 15 Jt 7

versiones de f

	C		1	2	3	4	5	6	7	8	9	10	11	12			
sA	20	~~0~~	~~9~~	20													
sB	69	0			~~4~~	~~18~~	~~19~~					~~23~~	28				
sC	145	~~0~~						~~1~~	~~5~~	~~25~~	~~40~~			~~42~~	~~53~~	~~55~~	60
AD	14	~~0~~	~~9~~														4
AE	19	~~0~~		~~11~~											0		
AF	18	0													~~11~~	~~13~~	18
BD	9	~~0~~										~~4~~	~~8~~			7	
BE	4	0			4												
BF	14	~~0~~				14											
BH	1	0					1										
CE	190	~~0~~						~~1~~						~~3~~	~~14~~	~~16~~	18
CF	4	0							4								
CH	20	0								20							
CI	20	0									15						
Dt	9	0	9														
DH	8	0										~~4~~				2	
DI	1	0															
DJ	7	0											~~3~~				0
Et	16	~~0~~		~~11~~	~~15~~			16									
EH	2	0														2	
EI	16	0															
EJ	7	~~0~~												~~2~~			7
Ft	146	~~0~~				~~14~~			~~18~~						~~29~~	~~31~~	36
GI	5	0															
Ht	25	~~0~~					~~1~~			~~21~~		25					
It	15	~~0~~									15						
Jt	7	0											~~5~~	7			

It	15	~~0~~	15		
Jt	7	~~0~~		~~5~~	7

1º network auxiliar:



1º camino: sADt
$\varepsilon = \min\{20, 14, 9\} = 9$

2º camino: sAEt
$\varepsilon = \min\{11, 19, 16\} = 11$

3º camino: sBEt
$\varepsilon = \min\{69, 4, 5\} = 4$

4º camino: sBFt
$\varepsilon = \min\{65, 14, 146\} = 14$

5º camino: SBHt
$\varepsilon = \min\{57, 1, 25\} = 1$

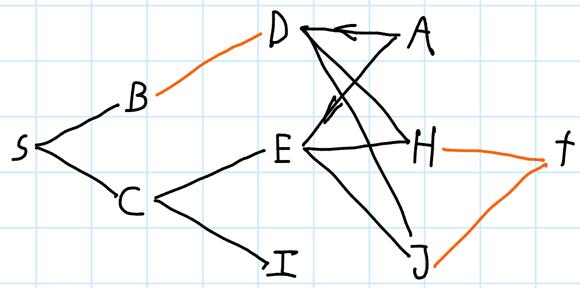
6º camino: sCEt
$\varepsilon = \min\{145, 190, 7\} = 1$

7º camino: sCFt
$\varepsilon = \min\{144, 4, 132\} = 4$

8º camino: sCHt
$\varepsilon = \min\{140, 20, 25\} = 20$

9º camino: sCIt
$\varepsilon = \min\{120, 20, 15\} = 15$

2º network auxiliar



10º camino: sBDHt

$\varepsilon = \min\{50, 9, 8, 4\} = 4$

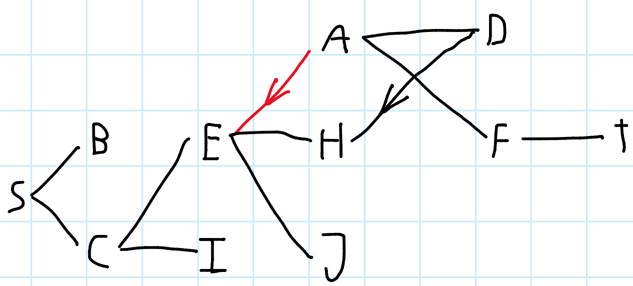
11º camino: sBDJt

$\varepsilon = \min\{46, 5, 7, 7\} = 5$

12º camino: sCEJt

$\varepsilon = \min\{105, 189, 7, 2\} = 2$

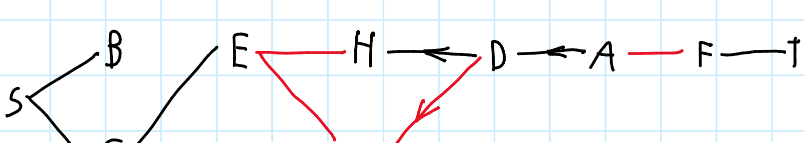
3º network auxiliar:

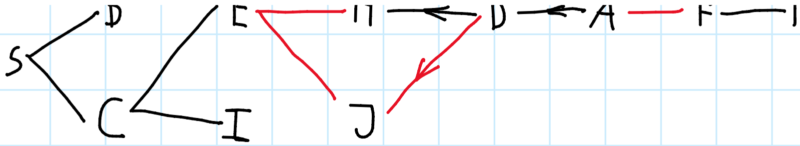


13º camino: sC$\overleftarrow{EA}$Ft

$\varepsilon = \min\{103, 187, 11, 18, 12\} = 11$

4º Network auxiliar:





14º camino: $sCE\overline{HDA}Ft$

$\varepsilon = \min\{97, 176, 2, 4, 9, 7, 117\} = 2$

15º camino: $sCE\overline{JDA}Ft$

$\varepsilon = \min\{95, 174, 5, 5, 9, 5, 115\} = 5$

1d) ❌ 🛠️ (muy largo)

sábado, 23 de abril de 2022 17:13

(d) su 160 sv 50 sw 100 ah 20 an 10 bh 10 bi 20 ci 10 cj 20 dj 15 dk 20 ek 15 el 20 fl 15 fm 20 gm 15 gn 10 ht 20 it 20 jt 20 kt 20 lt 20 mt 20 nt 20 pa 20 pb 20 pc 20 pd 20 pe 20 pf 20 pg 25 qx 50 qz 50 rx 50 rz 50 ux 60 uy 200 uz 100 vq 70 vr 20 wq 70 wr 30 xt 60 yp 200 zt 100

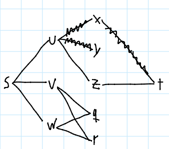
0 1 2 3 4 5 6 7 8

su	160	~~0~~ ~~60~~ 160
sv	50	~~0~~ ~~20~~ ~~40~~ 50
sw	100	~~0~~ ~~5~~ ~~25~~ ~~45~~ ~~50~~ ~~60~~ 70
ah	20	~~0~~ 20
an	10	0
bh	10	0
bi	20	~~0~~ 20
ci	10	0
cj	15	~~0~~ ~~10~~ 15
dj	15	0
dk	20	~~0~~ 20
ek	15	0
el	20	~~0~~ 20
fl	15	0
fm	20	~~0~~ ~~5~~ 15
gm	15	0
gn	10	~~0~~ 10
ht	20	~~0~~ 20
it	20	~~0~~ 20
jt	20	~~0~~ ~~10~~ 15
kt	20	~~0~~ 20
lt	20	~~0~~ 20
mt	20	~~0~~ ~~5~~ 15

nt	20	~~0~~ 10
pa	20	~~0~~ 20
pb	20	~~0~~ 20
pc	20	~~0~~ ~~10~~ 15
pd	20	~~0~~ 20
pe	20	~~0~~ 20
pf	20	~~0~~ ~~5~~ 15
pg	25	~~0~~ 10
qx	50	~~0~~ ~~20~~ ~~40~~ 50
qz	50	~~0~~ ~~5~~ ~~25~~ ~~45~~ 50
rx	50	~~0~~ 10
rz	50	~~0~~ 10
ux	60	~~0~~ ~~60~~ ~~40~~ ~~20~~ ~~10~~ 0
uy	200	~~0~~ ~~20~~ ~~40~~ ~~50~~ ~~55~~ ~~75~~ ~~95~~ ~~100~~ ~~110~~ 120
uz	100	~~0~~ ~~100~~ ~~95~~ ~~75~~ ~~55~~ ~~50~~ ~~45~~ 40
vq	70	~~0~~ ~~20~~ ~~40~~ 50
vr	20	0
wq	70	~~0~~ ~~5~~ ~~25~~ ~~45~~ 50
wr	30	~~0~~ ~~10~~ 20
xt	60	~~0~~ 60
yp	200	~~0~~ ~~20~~ ~~40~~ ~~50~~ ~~55~~ ~~75~~ ~~95~~ ~~100~~ ~~110~~ 120
zt	100	~~0~~ 100

~~10~~ ~~25~~ ~~45~~ 50	
~~15~~ 35	~~10~~ 25
	20
	~~5~~ 20
	20
10	~~10~~ ~~25~~ ~~45~~ 50
15	~~75~~ 35
20	
	~~50~~ ~~35~~ ~~15~~ 10
~~5~~ 20	~~50~~ ~~65~~ ~~85~~ ~~90~~ ~~105~~ 120
	~~85~~ 70
20	~~10~~ ~~25~~ ~~45~~ 30
	~~75~~ 35
10	
15	~~50~~ ~~65~~ ~~85~~ ~~90~~ ~~105~~ 120
20	
~~5~~ 20	
20	

1º network auxilar



1º camino: s u x t

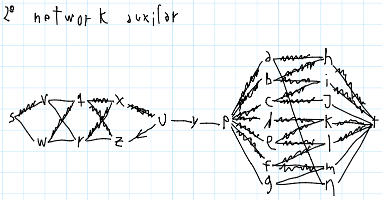
$\varepsilon = \min\{160, 60, 60\} = 60$

2º camino: s u z t

$\varepsilon = \min\{100, 100, 100\} = 100$

No hay mas caminos

2º network auxilar



3º camino: s v q $\overleftarrow{x\,u}$ y p a h t

$\varepsilon = \min\{50, 70, 50, 60, 200, 200, 20, 20, 20\} = 20$

4º camino: s v q $\overleftarrow{x\,u}$ y p b i t

$\varepsilon = \min\{30, 50, 30, 40, 180, 180, 20, 20, 20\} = 20$

4º camino: s v q $\overleftarrow{x\,v}$ y p c j t

$\varepsilon = \min\{40, 60, 40, 50, 190, 190, 190, 15, 20\} = 15$

5º camino: s v q $\overleftarrow{x\,u}$ y p c j t
$\varepsilon = \min\{10, 30, 10, 20, 160, 160, 20, 15, 20\} = 10$

6º camino: s w q $\overleftarrow{z\,u}$ y p c j t
$\varepsilon = \min\{100, 70, 50, 100, 150, 150, 10, 5, 10\} = 5$

7º camino: s w q $\overleftarrow{z\,u}$ y p d k t
$\varepsilon = \min\{95, 65, 45, 95, 145, 145, 20, 20, 20\} = 20$

8º camino: s w q $\overleftarrow{z\,u}$ y p e l t
$\varepsilon = \min\{75, 45, 25, 75, 125, 125, 20, 20, 20\} = 20$

9º camino: s w q $\overleftarrow{z\,u}$ y p f m t
$\varepsilon = \min\{55, 25, 5, 55, 105, 105, 20, 20, 20\} = 5$

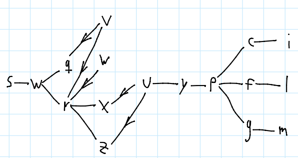
10º camino: s w r $\overleftarrow{x\,u}$ y p f m t
$\varepsilon = \min\{50, 30, 50, 10, 100, 100, 15, 15, 15\} = 10$

11º camino: s w r $\overleftarrow{z\,u}$ y p f m t
$\varepsilon = \min\{40, 20, 50, 50, 90, 90, 5, 5, 5\} = 5$

12º camino: s w r $\overleftarrow{z\,u}$ y p g n t
$\varepsilon = \min\{35, 15, 45, 45, 85, 85, 25, 10, 20\} = 10$

No hay mas caminos

3º network auxilar



5º camino: s v q $\overleftarrow{x\,u}$ y p d k t
$\varepsilon = \min\{25, 45, 25, 35, 175, 175, 20, 20, 20\} = 20$

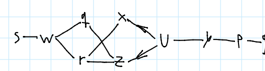
6º camino: s v q $\overleftarrow{x\,u}$ y p e l t
$\varepsilon = \min\{5, 25, 5, 15, 155, 155, 20, 20, 20\} = 5$

7º camino: s w q $\overleftarrow{z\,u}$ y p e l t
$\varepsilon = \min\{100, 70, 50, 100, 110, 110, 15, 15, 15\} = 15$

8º camino: s w q $\overleftarrow{z\,u}$ y p f m t
$\varepsilon = \min\{85, 55, 35, 85, 95, 95, 20, 20, 20\} = 20$

No hay mas caminos

3º network auxilar:



2)

miércoles, 20 de abril de 2022 16:02

(2) Vimos que la complejidad de Dinic para encontrar un blocking flow es $O(nm) = O(n^3)$ en networks densos. Probar que es $\Omega(n^3)$ usando los siguientes networks para $r > 2$: $n = 3r$ vertices que son x_i, z_i, y_i, $i = 1, ..., r$. x_1 es s, x_r es t. Los lados son $x_i x_{i+1}$ todos con capacidad r^2 para todo $i < r - 1$, lados $x_{r-1}z_j$ de capacidad r^2 para todo j, lados $z_j y_k$ de capacidad 1 para todo j, k, lados $y_k t$ de capacidad r^2 para todo k.

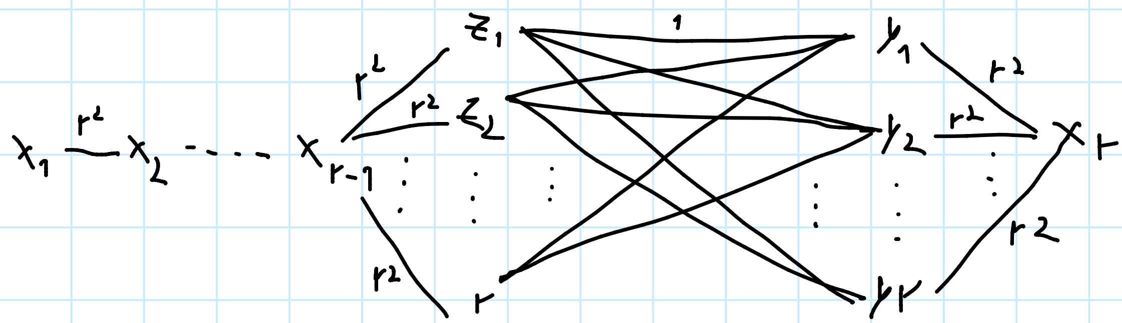
$N = (V, E, c)$

$V = \{x_1, z_i, y_i : i \in \{1, \dots, r\}\}$

$s = x_1$

$t = x_r$

$E = \{\overrightarrow{x_i x_{i+1}} : i \in \{1, \dots, r-1\}\} \cup \{\overrightarrow{x_{r-1} z_i}, \overrightarrow{z_i y_j}, \overrightarrow{y_i x_r} \ : i, j \in \{1, \dots, r\}\}$



El network auxiliar va a coincidir con el original, y por ende va a ser uno solo

En el network auxiliar se van a encontrar exactamente r^2 caminos, cada uno con capacidad 1. Van a tener capacidad 1, por el lado $\overrightarrow{z_1 y_j}$, y van a ser r^2, todos por los vertices $\overrightarrow{x_1 x_2}, \dots, \overrightarrow{x_{r-2} x_{r-1}}$, r por $\overrightarrow{x_{r-1} z_i}$ y por $\overrightarrow{y_i x_r}$, y 1 por cada lado $\overrightarrow{z_1 y_j}$, ya que si se suma 1 en cada 1, se va a llegar a sumar lo necesario.

Encontrar cada camino (hacer un DFS) en este caso va a recorrer exactamente $r + 2$ vértices, por ende la complejidad va a ser $\Omega((r+2)r^2) = \Omega(r^3) = \Omega\left(\left(\frac{n}{3}\right)^3\right) = \Omega(n^3)$

3)

viernes, 22 de abril de 2022 17:42

(3) Modificar ligeramente el algoritmo de Dinic para que en networks en donde todas las capacidades son 1 corra en tiempo $O(nm)$ (i.e. halla un blocking flow en tiempo $O(m)$), probando esta cota. (ayuda: cambiar el avanzar y el retroceder inteligentemente para poder eliminar incrementarFlujo).

Hay que ir borrando del network auxiliar las aristas a medida que se usan. Y ademas para mas eficiencia se puede ir sumando el ε (que siempre va a ser 1) a medida que se recorre el network auxiliar