Flujo Máximo: parte 2

Tecnología Digital V: Diseño de Algoritmos

Universidad Torcuato Di Tella

Repaso

- Un corte en la red $G = (N, A)$ es un subconjunto $S \subseteq N \setminus \{t\}$ tal que $s \in S$.

Repaso

- Un corte en la red $G = (N, A)$ es un subconjunto $S \subseteq N \setminus \{t\}$ tal que $s \in S$.
- Dados $S, T \subseteq N$, definimos $ST = \{ij : i \in S \text{ y } j \in T\}$

Repaso

- Un corte en la red $G = (N, A)$ es un subconjunto $S \subseteq N \setminus \{t\}$ tal que $s \in S$.
- Dados $S, T \subseteq N$, definimos $ST = \{ij : i \in S \text{ y } j \in T\}$

Proposición

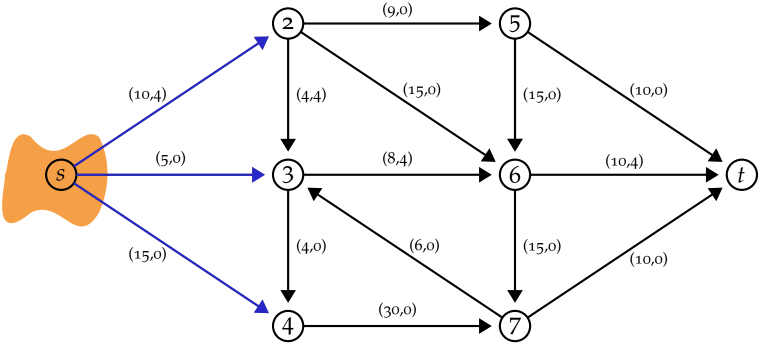
Sea x un flujo definido en una red $G = (N, A)$ y sea S un corte. Entonces

$$F = \sum_{ij \in S\bar{S}} x_{ij} - \sum_{ij \in \bar{S}S} x_{ij}$$

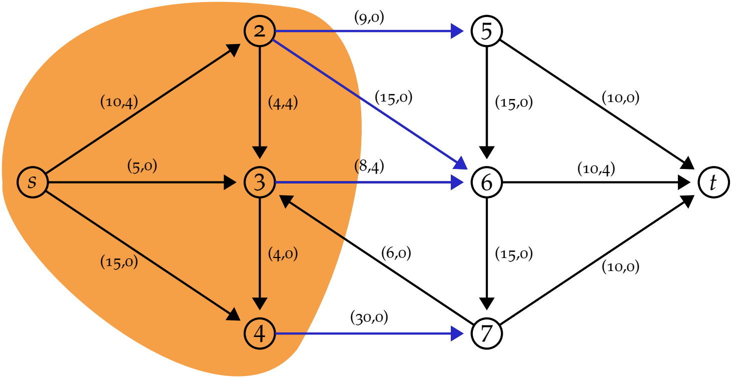
donde $\bar{S} = N \setminus S$.

Flujo en redes

$$F = 4$$

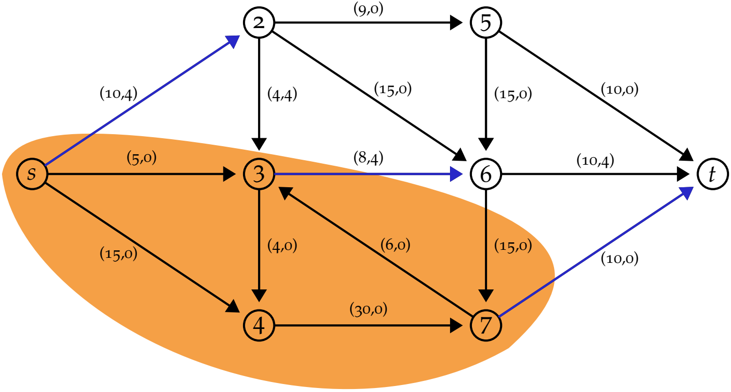


$$F = 4$$



Flujo en redes

$$F = 4$$



- La capacidad de un corte S se define como
$$u(S) = \sum_{ij \in S\bar{S}} u_{ij}.$$

- La capacidad de un corte S se define como

$$u(S) = \sum_{ij \in S\bar{S}} u_{ij}.$$

Proposición

Si x es un flujo con valor F y S es un corte en N, entonces $F \leq u(S)$.

- La capacidad de un corte S se define como

$$u(S) = \sum_{ij \in S\bar{S}} u_{ij}.$$

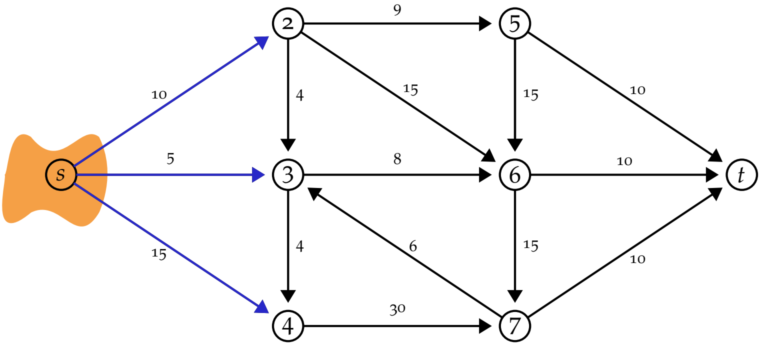
Proposición

Si x es un flujo con valor F y S es un corte en N, entonces $F \leq u(S)$.

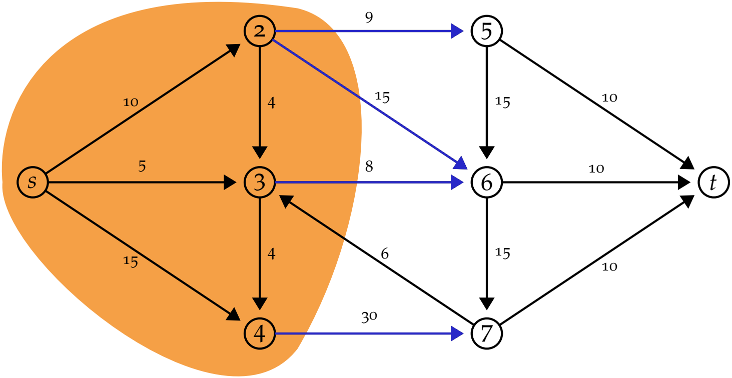
Corolario (certificado de optimalidad)

Si F es el valor de un flujo x y S un corte en G tal que $F = u(S)$ entonces x define un flujo máximo y S un corte de capacidad mínima.

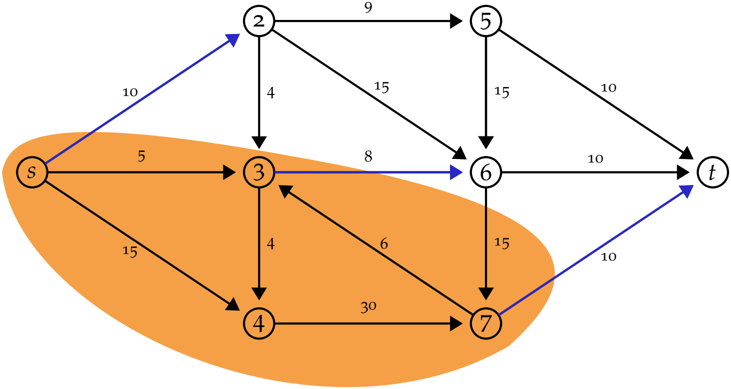
$$U = 30$$



$$U = 62$$

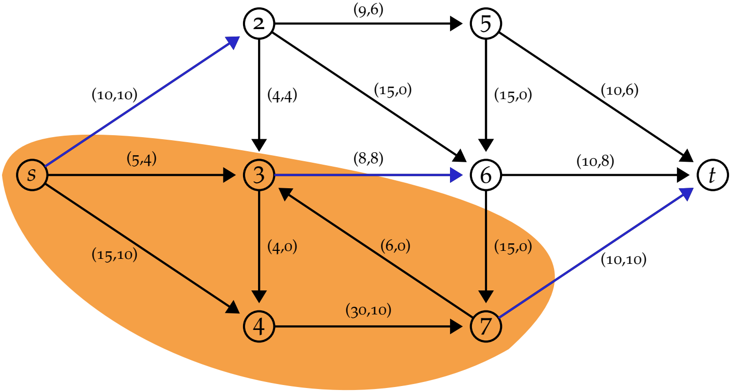


$$U = 28$$



$$U = 28$$

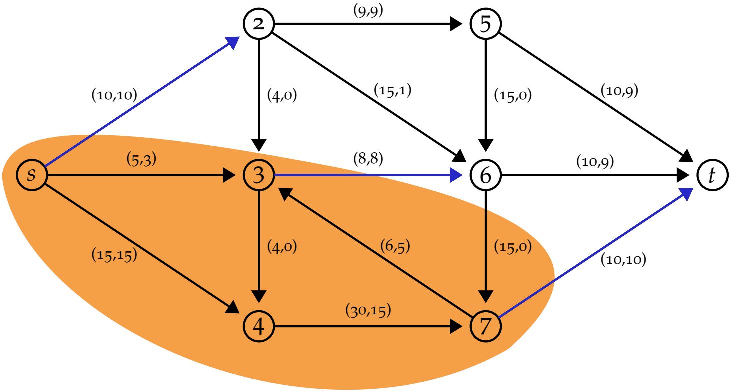
$$F = 24$$



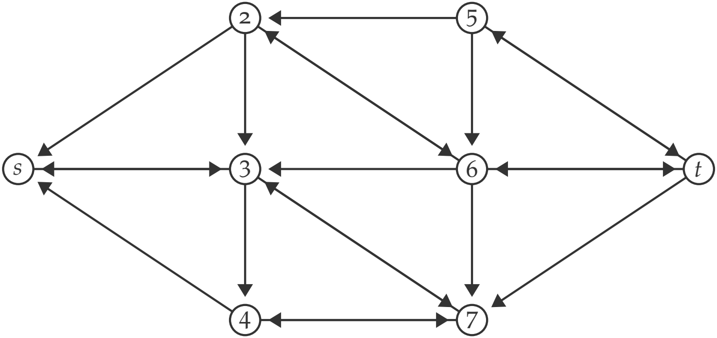
2
5
s
3
6
t
4
7
(9,6)
(10,10)
(4,4)
(15,0)
(15,0)
(10,6)
(5,4)
(8,8)
(10,8)
(15,10)
(4,0)
(6,0)
(15,0)
(10,10)
(30,10)

$$U = 28$$

$$F = 28$$


2
5
3
6
s
t
4
7
(9,9)
(10,10)
(4,0)
(15,1)
(15,0)
(10,9)
(5,3)
(8,8)
(10,9)
(15,15)
(4,0)
(6,5)
(15,0)
(10,10)
(30,15)

Recuperacion el corte mínimo

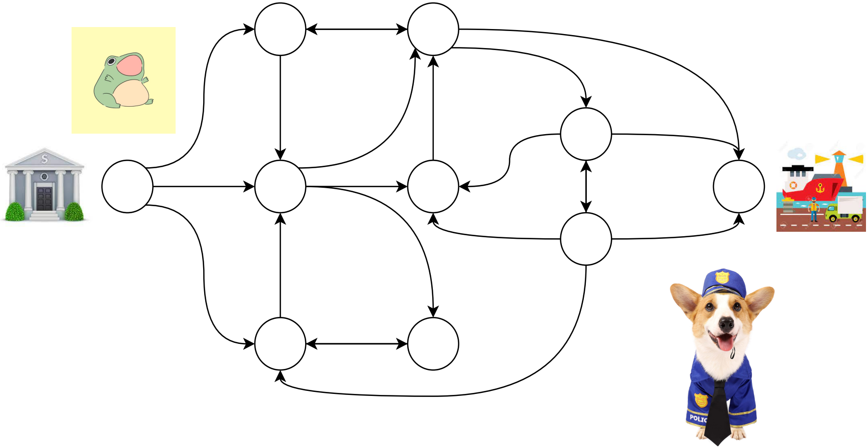


Pepe el sapo robo un banco y se esta dirigiendo al puerto para escaparse. Sin embargo, Bob el perro lo quiere detener cerrando algunas calles de la ciudad.

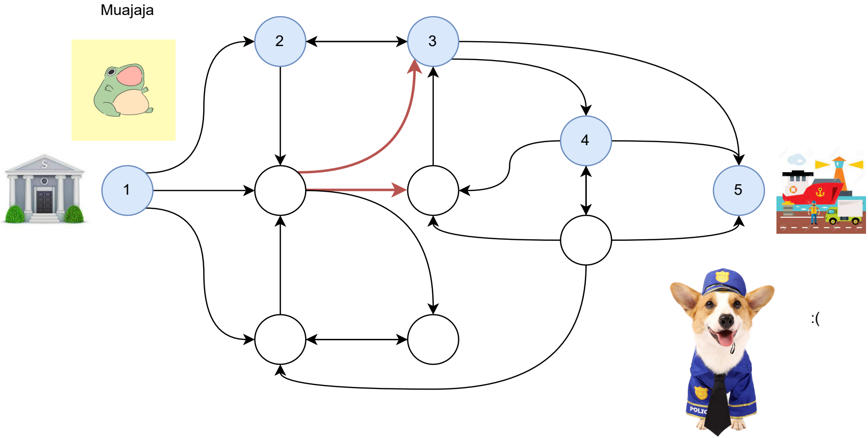
Cerrar una calle es una tarea que lleva tiempo, por ende Bob nos pidio ayuda para decidir la menor cantidad de calles que debemos cerrar para que el malhechor Pepe no se escape.

(Se puede asumir que la ciudad es un grafo dirigido)

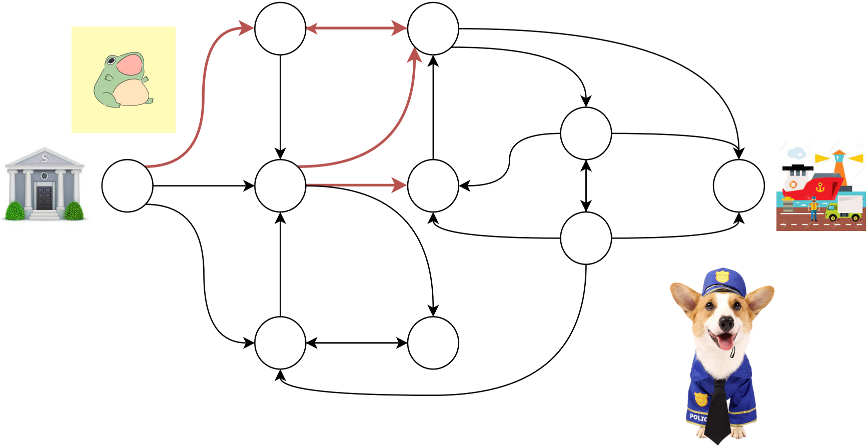
Dibujito



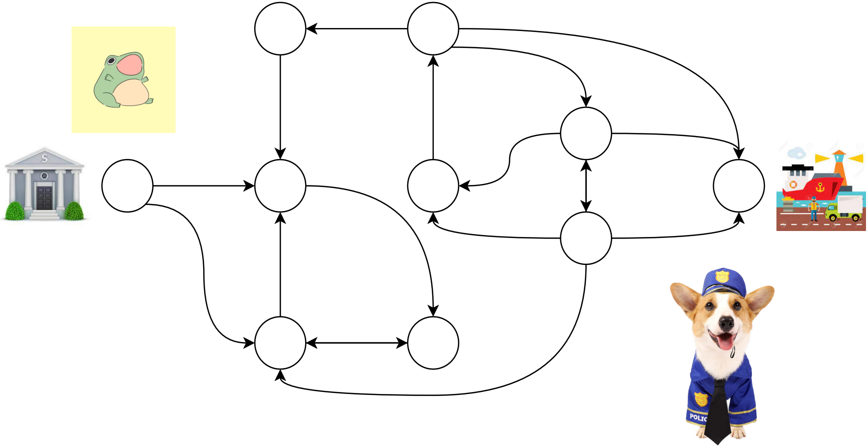
Dibujito



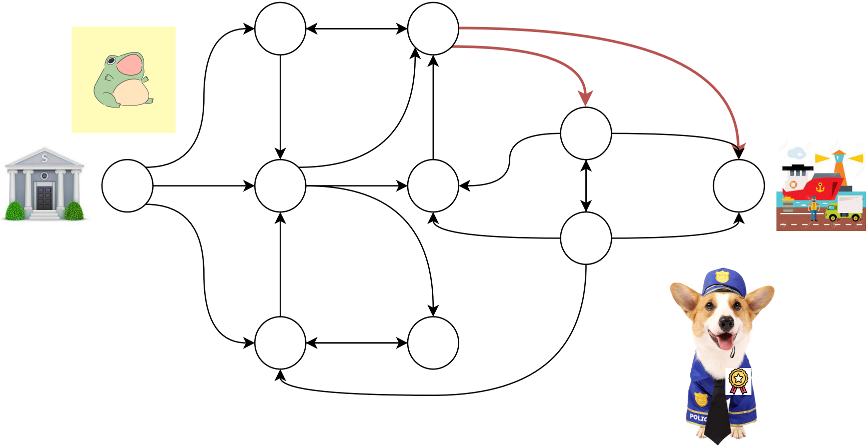
Dibujito



Dibujito



Dibujito



Sea s el nodo correspondiente al banco y t el nodo correspondiente al puerto. Nos interesa sacar la menor cantidad de aristas tal que no exista un camino entre s y t.

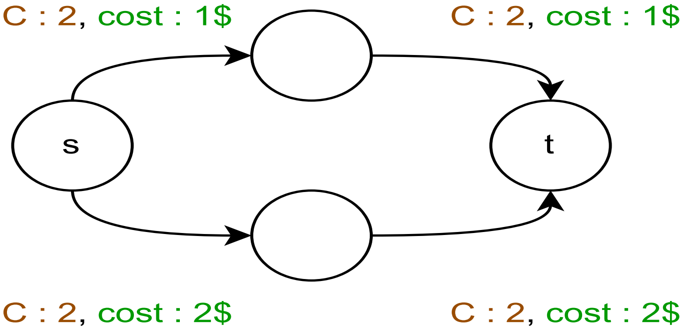
Podemos pensar que una vez que saquemos las aristas el grafo va a quedar divido en dos componentes, una que contiene a s y otra que contiene a t y desde la primera no debería ser posible llegar a la segunda. Esto se va pareciendo al problema del corte mínimo.

Como nos interesa la cantidad de calles a cerrar, podemos colocar a todos los arcos capacidad 1.

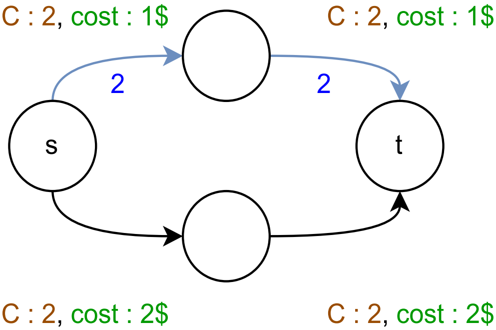
Con los algoritmos visto en la teórica podemos modelar el problema de flujo máximo de mínimo costo.

Coloquialemente, este consiste en encontrar el flujo máximo en una red y luego de todos los flujos con dicho valor tomar el de menor costo. El costo se encuentra definido en las aristas y es por unidad de flujo.

Ejemplo

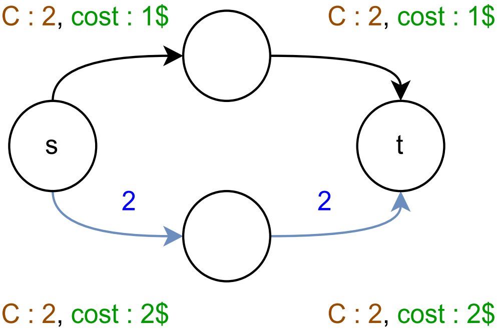


Ejemplo



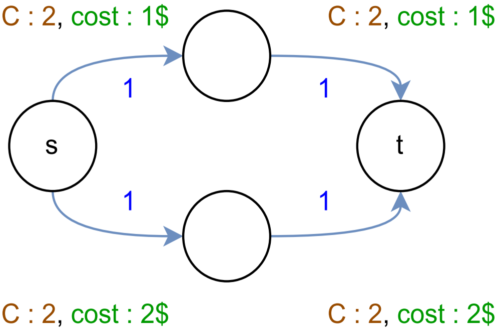
Costo total: 4

Ejemplo



Costo total: 8

Ejemplo



Costo total: 6

Problema

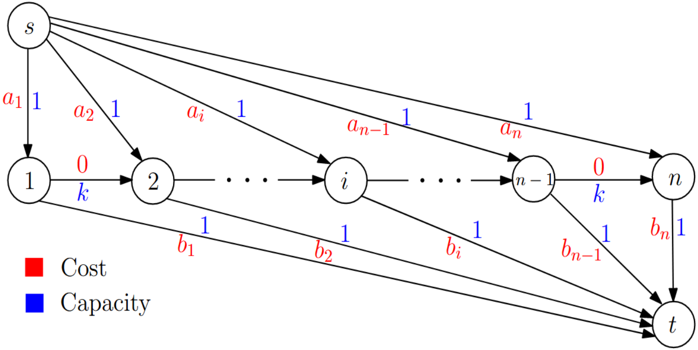
Javier y Juan tienen que preparar k parciales para los proximos n días. Una vez preparado el parcial hay que imprimirlo.

En cada día pueden preparar a lo sumo 1 parcial, porque sino se cansan, y pueden imprimir a lo sumo 1 parcial, porque tienen Windows 11.

Cada día tiene asociado un costo de preparar un parcial y un costo de imprimir un parcial. Obviamente no podemos imprimir un parcial si todavía no fue preparado.

Podemos ayudar a Javier y a Juan a preparar todos los parciales con el mínimo costo?

Solución



s
1
2
i
n − 1
n
t
a_1 1
a_2 1
a_i 1
a_{n-1} 1
a_n 1
0
k
0
k
1 b_1
1 b_2
1 b_i
1 b_{n-1}
b_n 1
Cost
Capacity