Grafos y Flujo

Marcelo Lemus

marcelo.lemus@progcomp.cl

Importancia de flujo en ICPC

- En casi todas las regionales ICPC hay un problema
- La solución siempre suele ser armar el grafo y correr un algoritmo de flujo (que veremos ahora que posibilidades hay)
- Muchos no saben/manejan flujo
- El problema de flujo no parece que sea de flujo
- Junto con geometria computacional, el equipo que tenga a alguien que se maneje especialmente bien en alguno de estos temas, puede llegar a garantizar la victoria sobre los demas equipos.

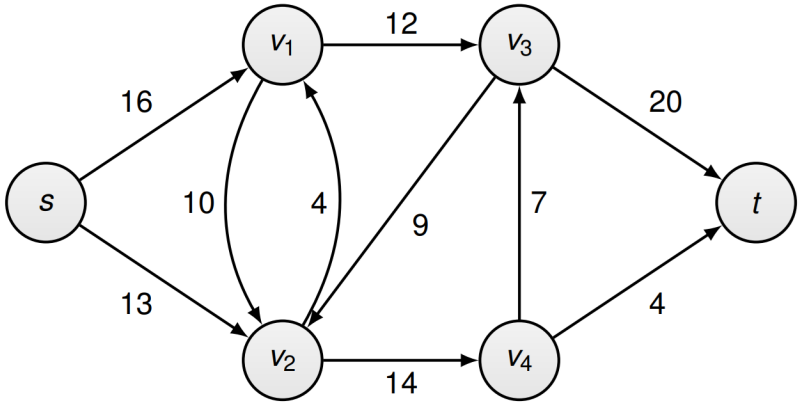
Redes de flujo

Una red flujo consiste en:

- Un grafo dirigido G con pesos en las aristas
- Dos nodos especiales: s (fuente) y t (sumidero)

Nuestro objetivo es mandar la mayor cantidad de unidades de flujo de s a t

- Los pesos en las aristas representan la cantidad maxima de flujo que soportan



Flujo sobre una Red

Es una asignacion de un real no negativo a cada arista tal que:

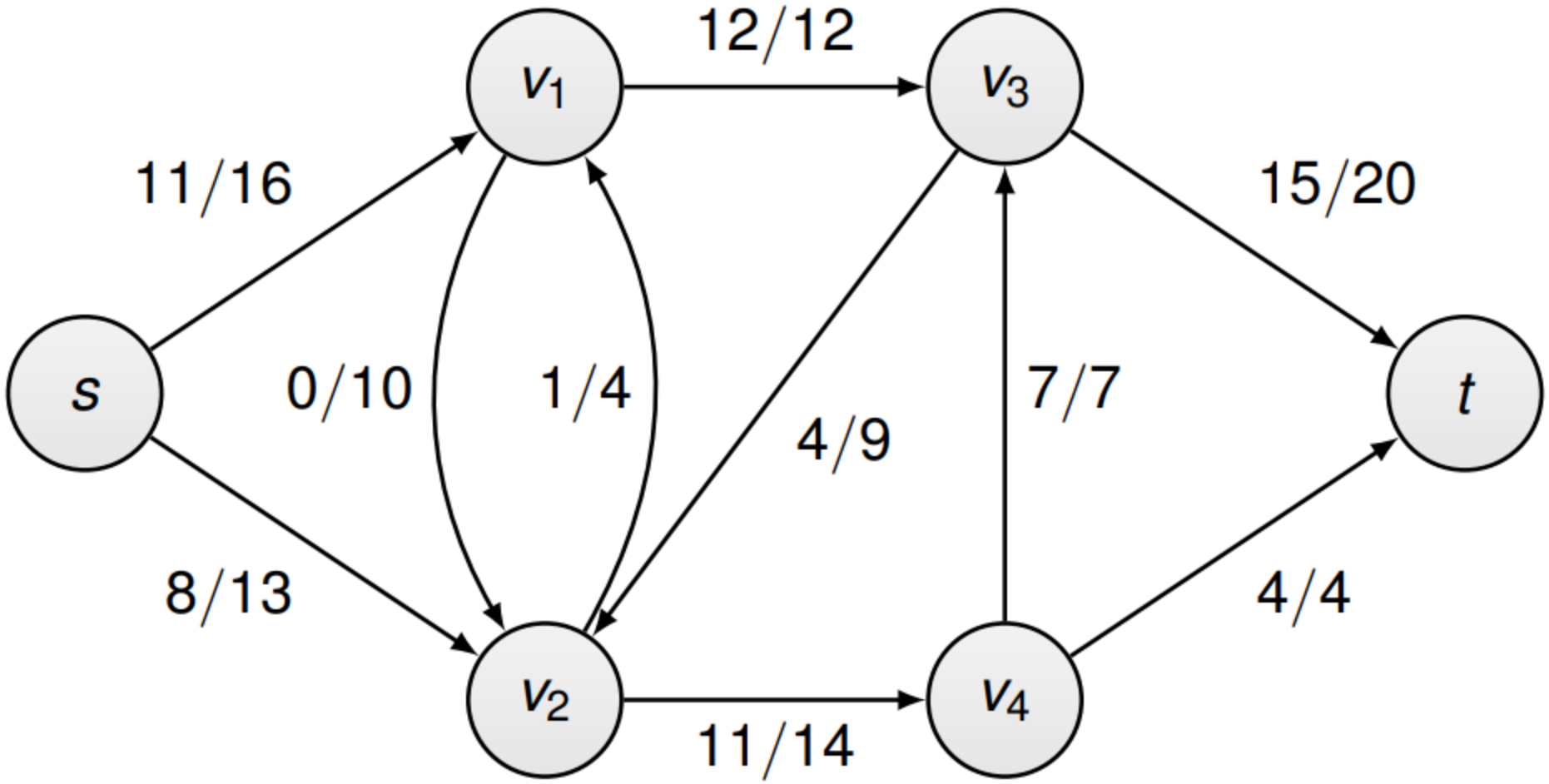
- Para cada arista digo cuantas unidades de flujo pasan por ella.
- No asigno a una arista mas de su capacidad maxima.
- En cada nodo excepto s y t el flujo que entra tiene que ser igual al que sale.

El valor del flujo es:

- La cantidad de flujo que sale de s.
- La cantidad de flujo que entra a t.

Ambos numeros siempre coinciden, queremos encontrar cual es el mayor valor posible de un flujo sobre la red.

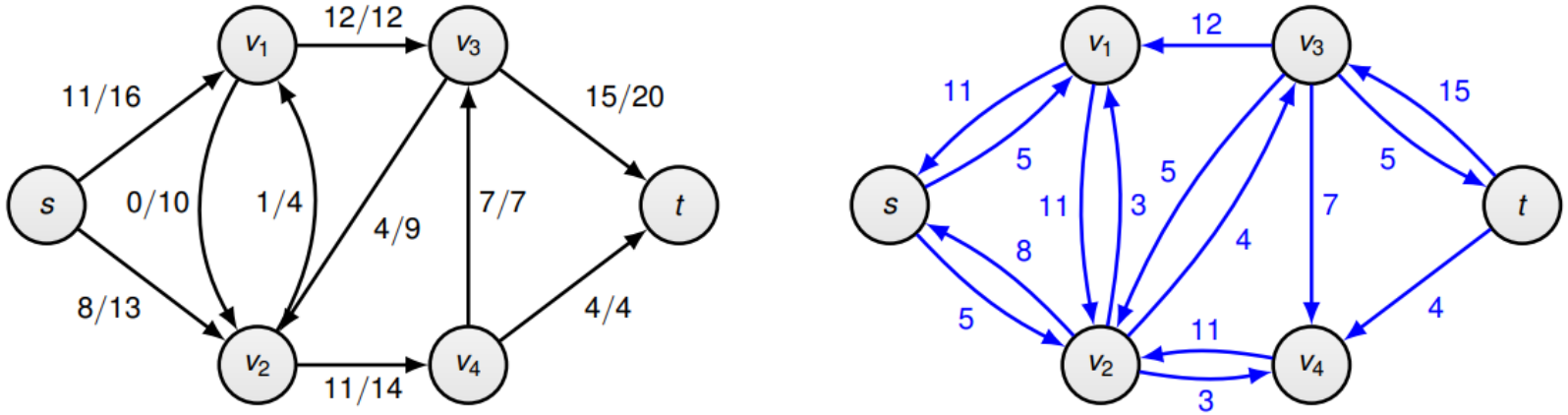
Ejemplo



Red Residual

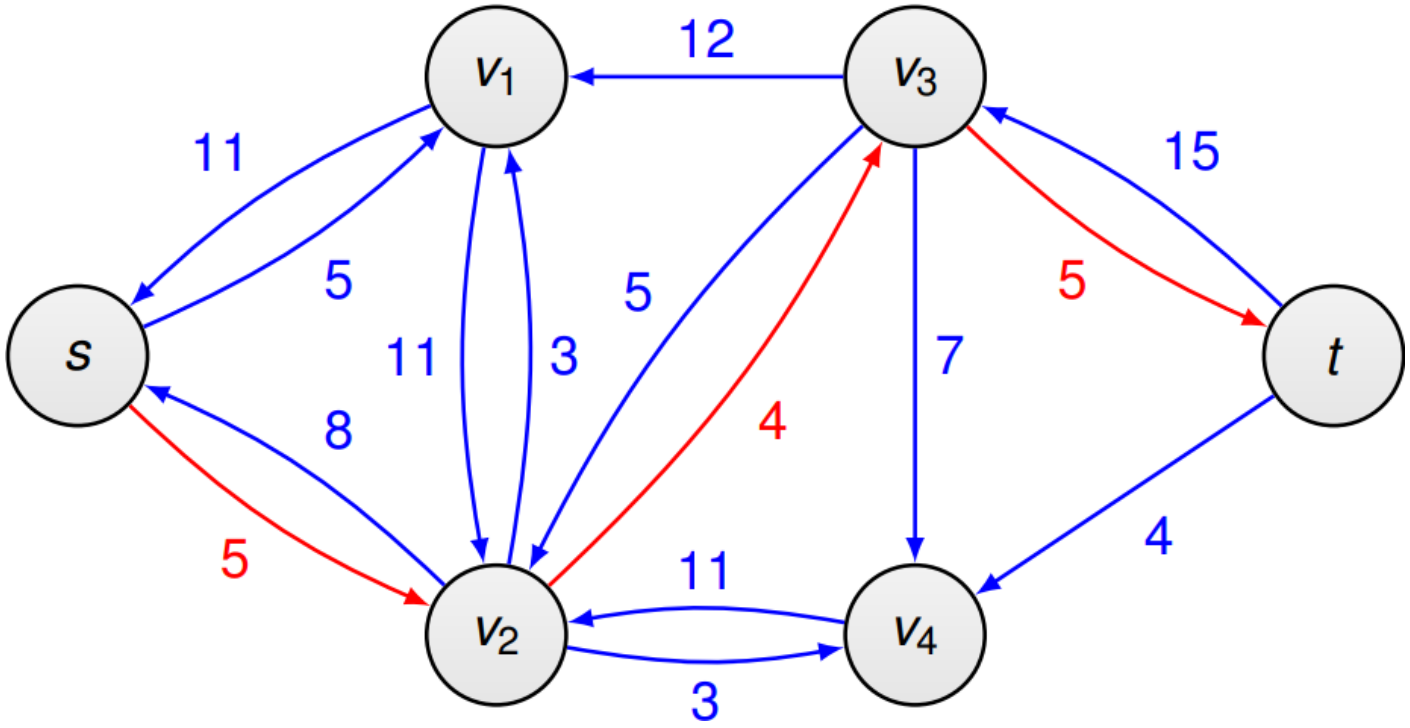
La red residual de un flujo es un grafo dirigido tal que:

- Los nodos son los mismos que en la red original.
- Hay una arista de u a v con capacidad la suma de:
  - Lo que me falta para llenar la capacidad de u a v en el grafo original.
  - El flujo que estoy mandando de v a u.



Augmenting Path (Camino de aumentanto)

Un camino de aumento es un camino de s a t en la red residual.
Dado un camino de aumento podemos conseguir un flujo mayor mandando todo lo posible por este camino.



Ford Fulkerson

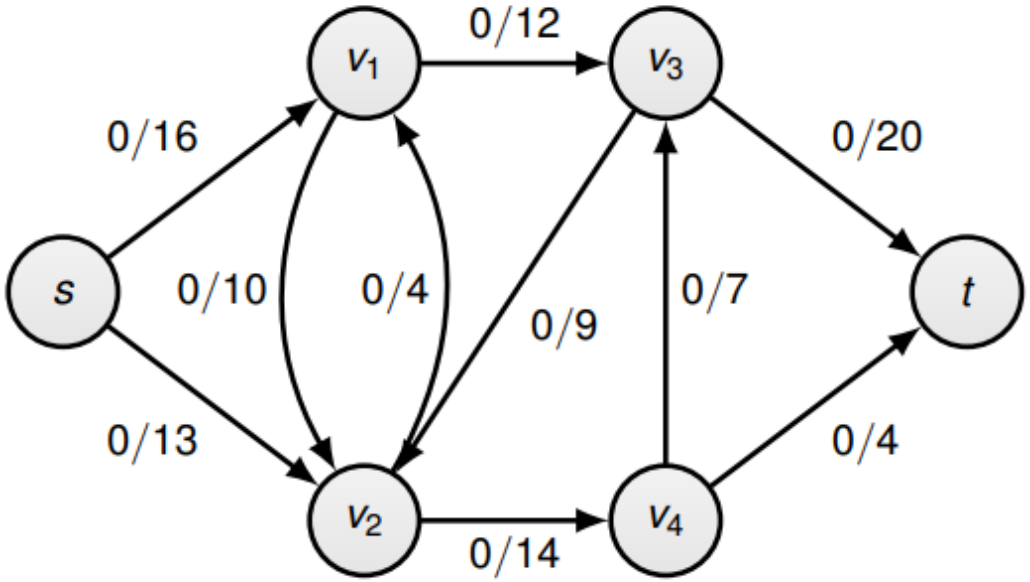
Este seria un posible algoritmo para calcular el flujo maximo.

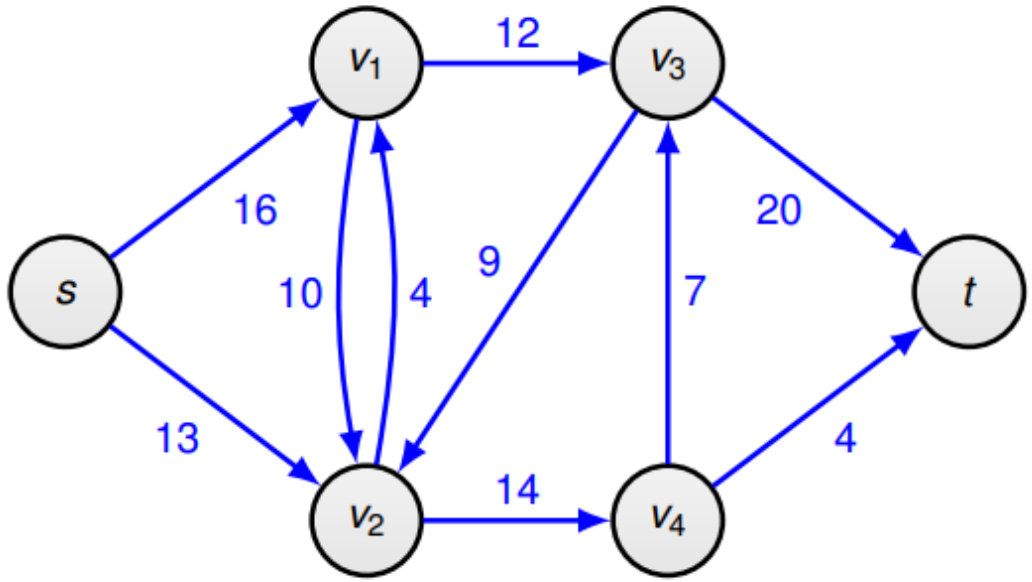
- Empiezo con el flujo vacio (el flujo de cada arista es 0)
- Construyo la red residual
- Mientras haya camino de s a t en la red residual:
  - Busco el la arista con menor peso (x) del camino.
  - Actualizo el flujo para mandar x unidades de flujo por ese camino.
  - Actualizo la red residual.

Devuelvo el flujo obtenido.

Ejemplo de flujo con Ford Fulkerson

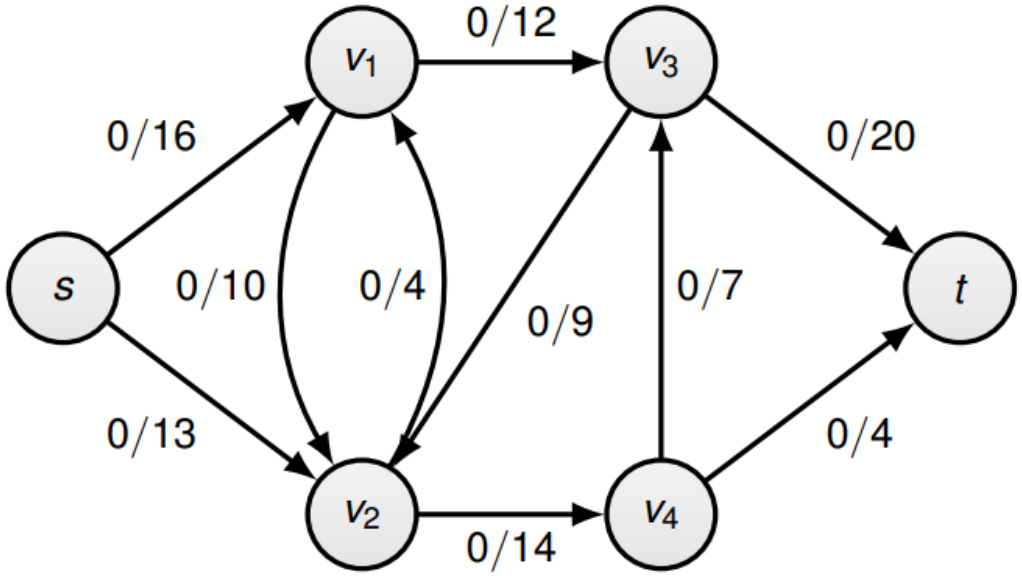
Empezamos con 0 en todas las aristas (flujo enviado)

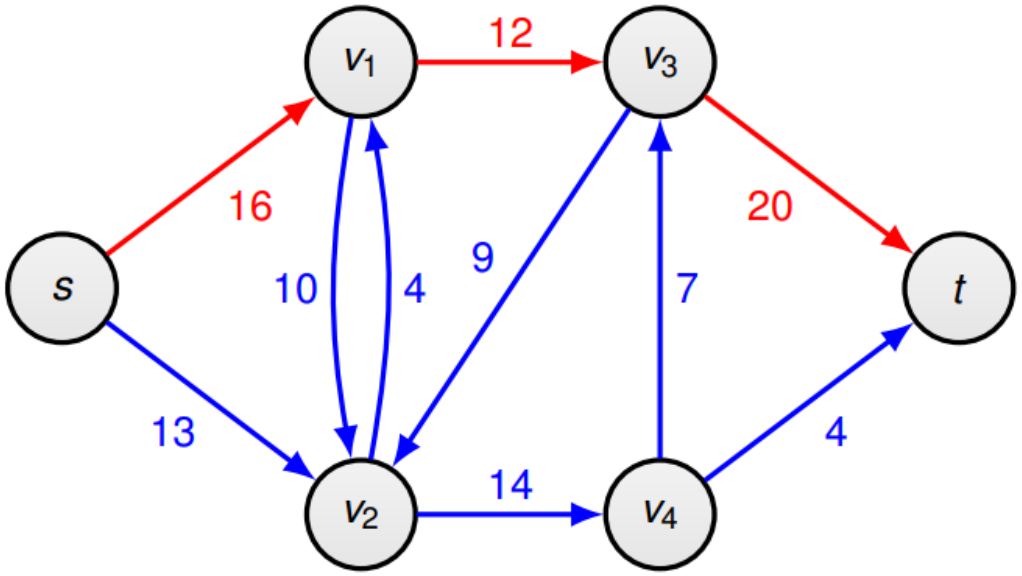




Ejemplo de flujo con Ford Fulkerson

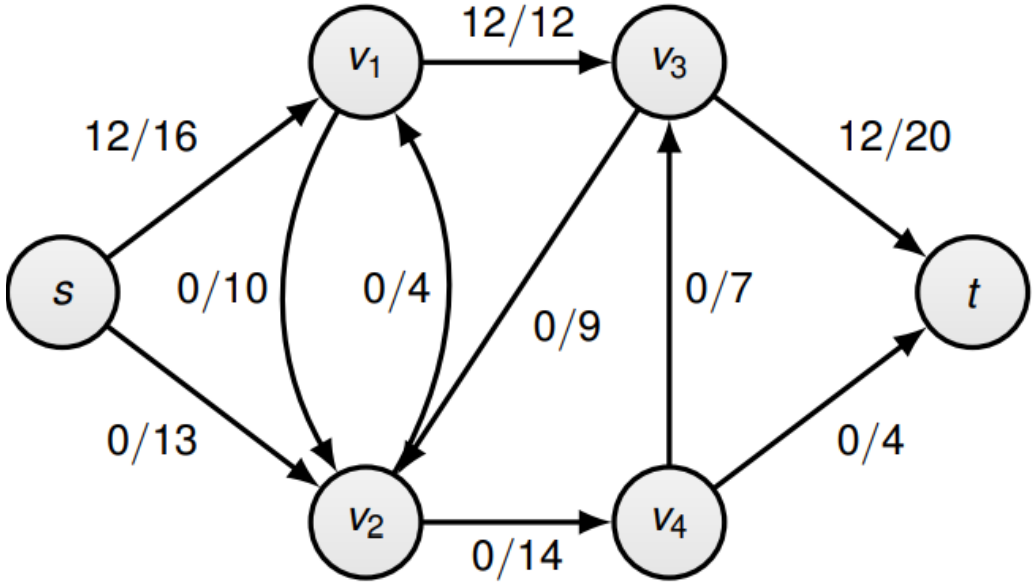
Encontramos un camino aumentante en la red residual.

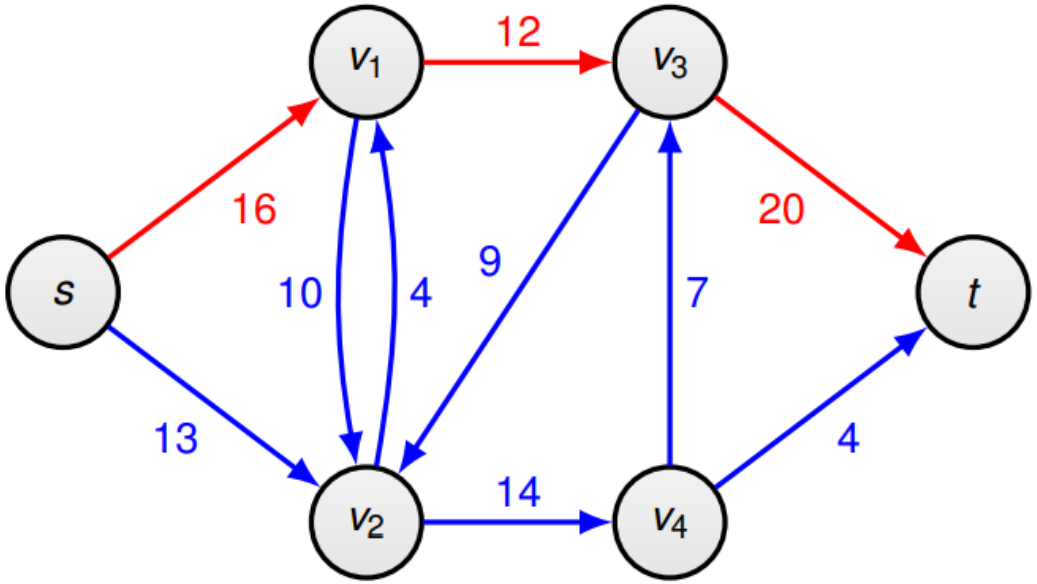




Ejemplo de flujo con Ford Fulkerson

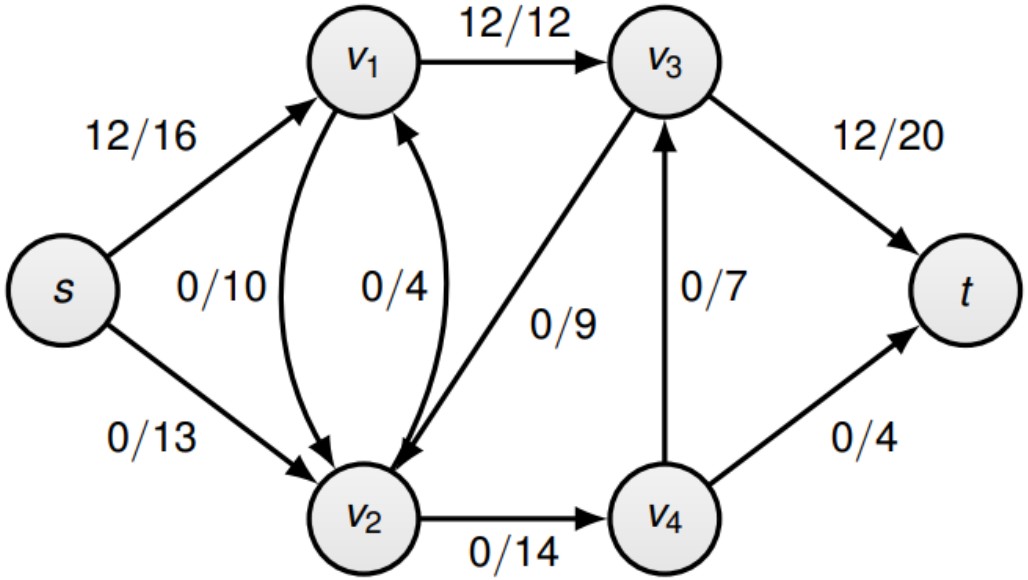
Mandamos todo lo que podemos por ese camino

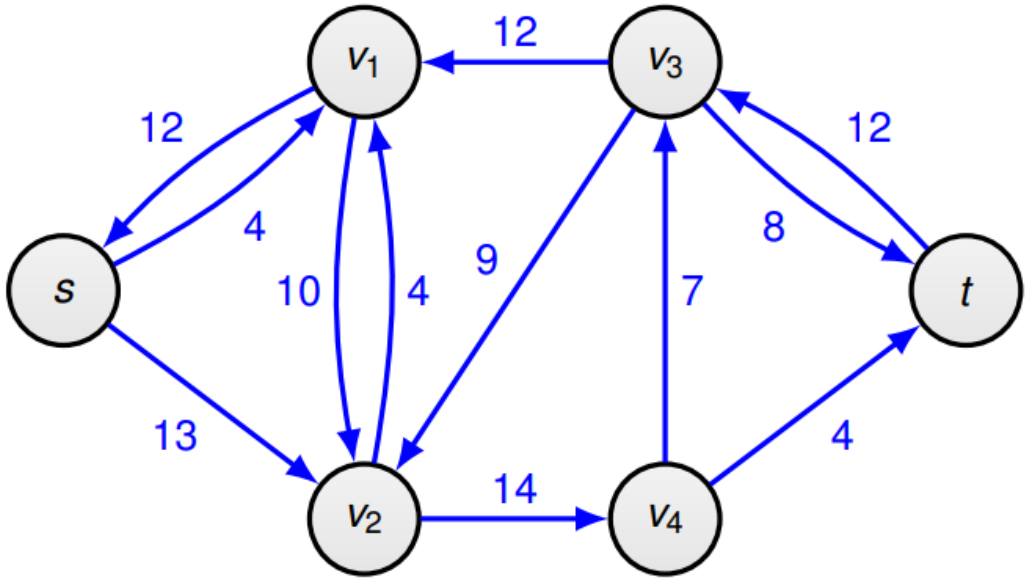




Ejemplo de flujo con Ford Fulkerson

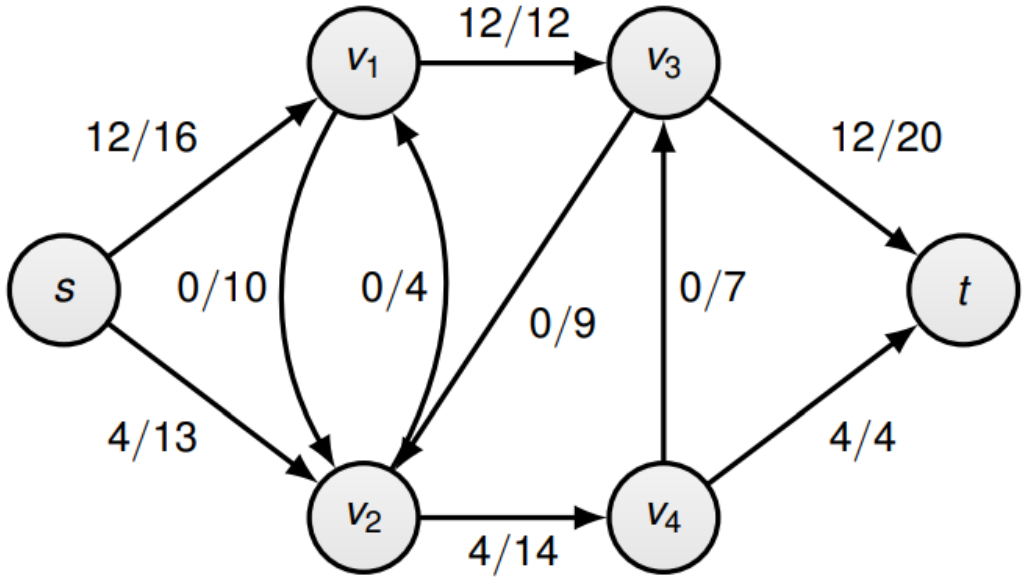
Actualizamos la red residual.

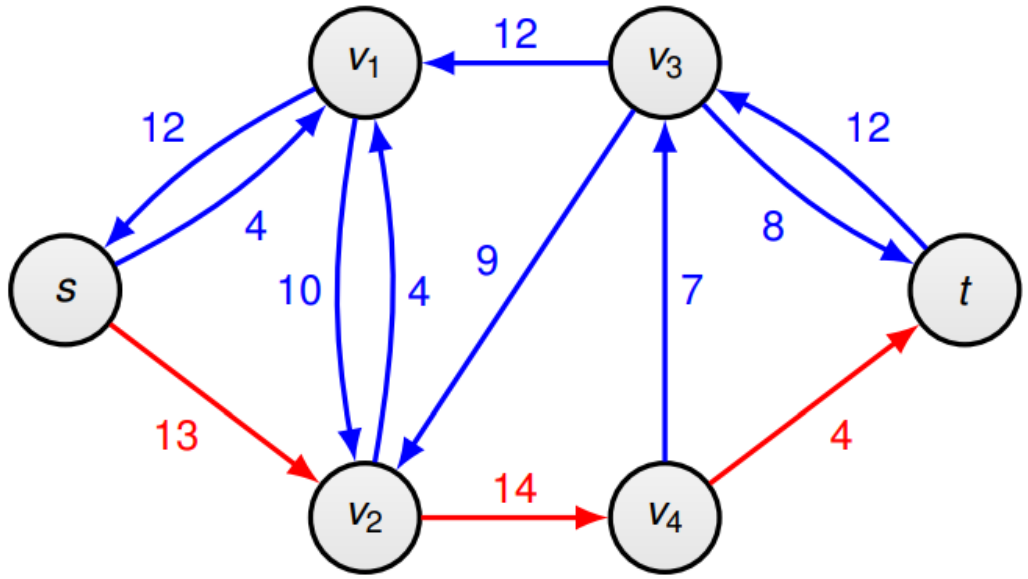




Ejemplo de flujo con Ford Fulkerson

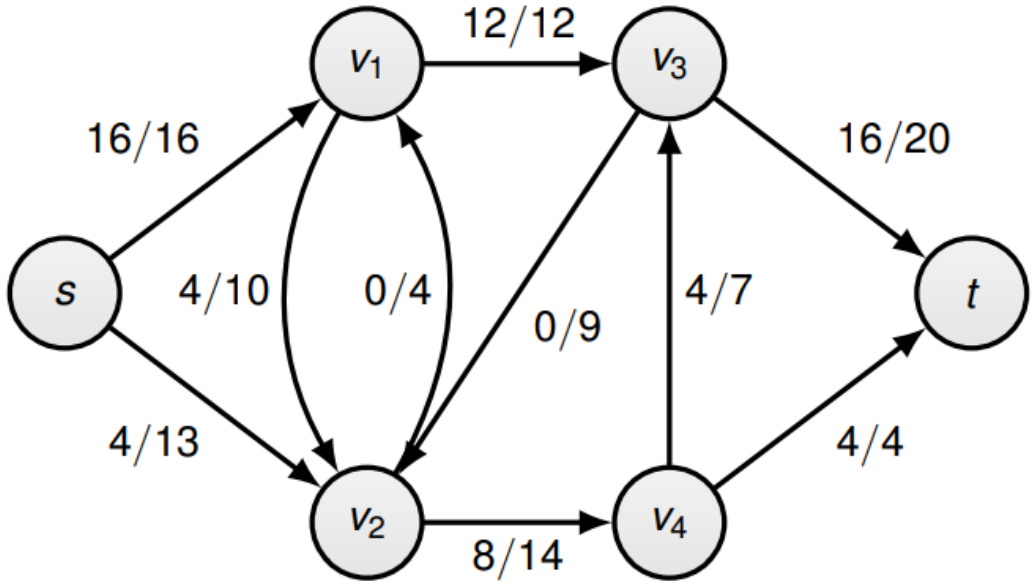
Volvemos a buscar un camino aumentante en la red residual y mandamos flujo por el.

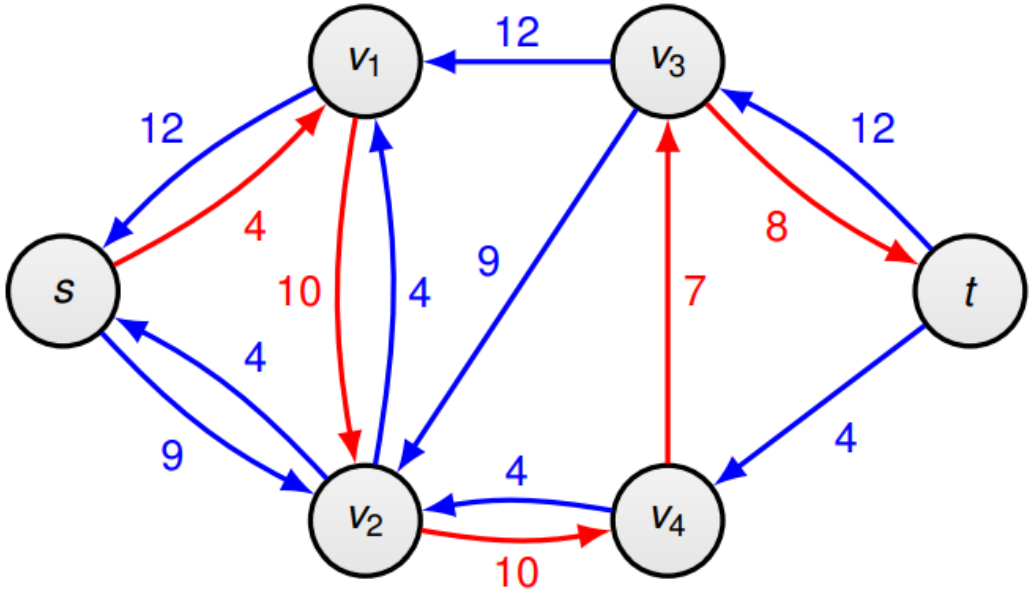




Ejemplo de flujo con Ford Fulkerson

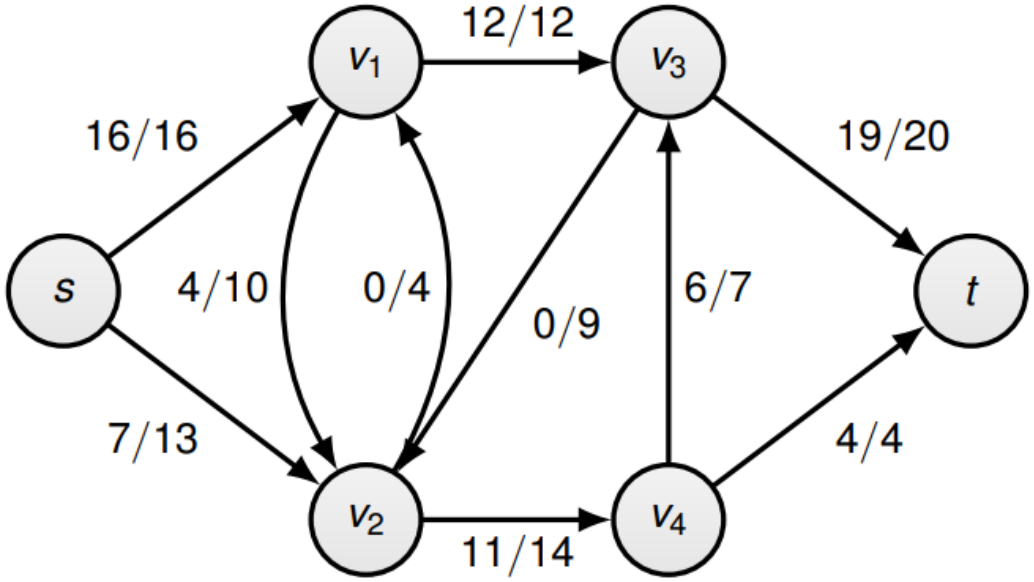
Actualizamos la red residual y repetimos.

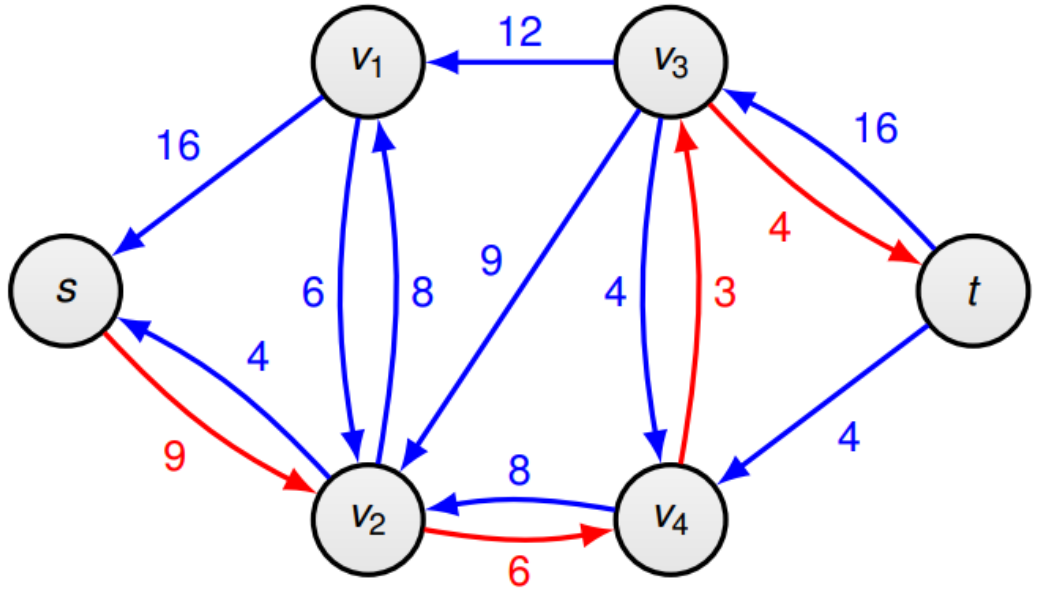




Ejemplo de flujo con Ford Fulkerson

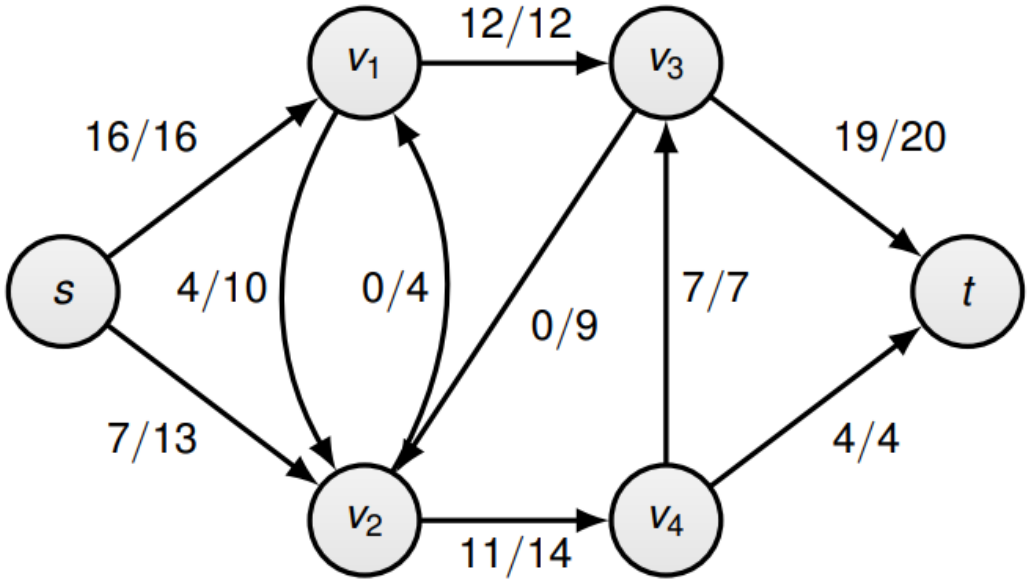
Actualizamos la red residual y repetimos.

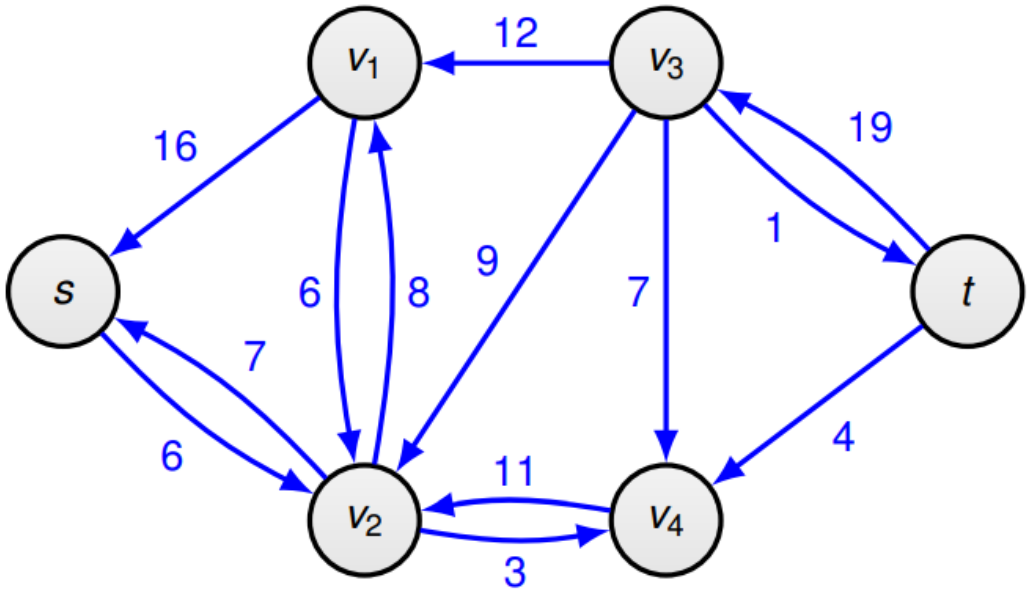




Ejemplo de flujo con Ford Fulkerson

No hay mas caminos aumentantes en la red residual, paramos.





Edmond Karp y Dinitz (Dinic)

Si ademas buscamos el camino de s a t con BFS:

- El algoritmo queda $O(nm^2)$ con n la cantidad de nodos y m la de aristas.
- Este algoritmo es conocido como Edmond Karp.

Tambien existe el algoritmo de Dinic

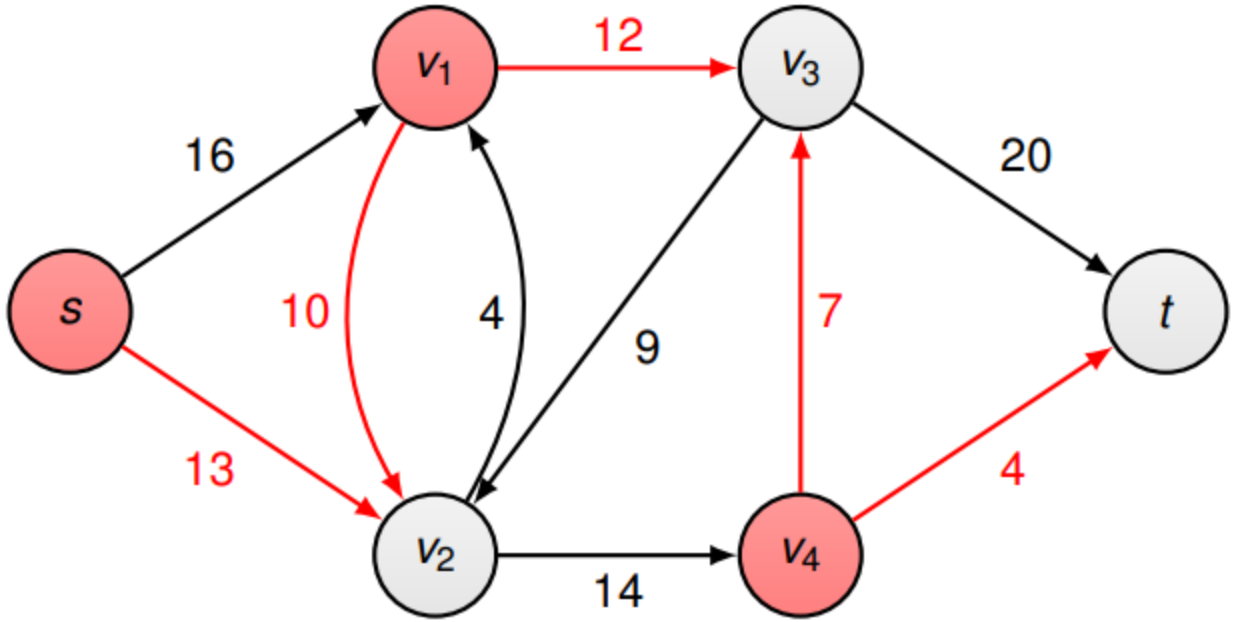
- construye los niveles usando bfs y luego encuentra todas las rutas disjuntas posibles de esa capa utilizando DFS antes de actualizar el grafo.
- El algoritmo queda $O(n^2m)$ por lo que es mas rapido que Edmond Karp.
- Es bastante mas largo de implementar.

Siempre debemos considerar la cota $O(mF)$ para todo algoritmo de maximo flujo, donde F es el valor del flujo máximo.

Hay otros algoritmos situacionales como push relabel y MPM, pero en la practica y en las competencias, entonces no son necesarios.

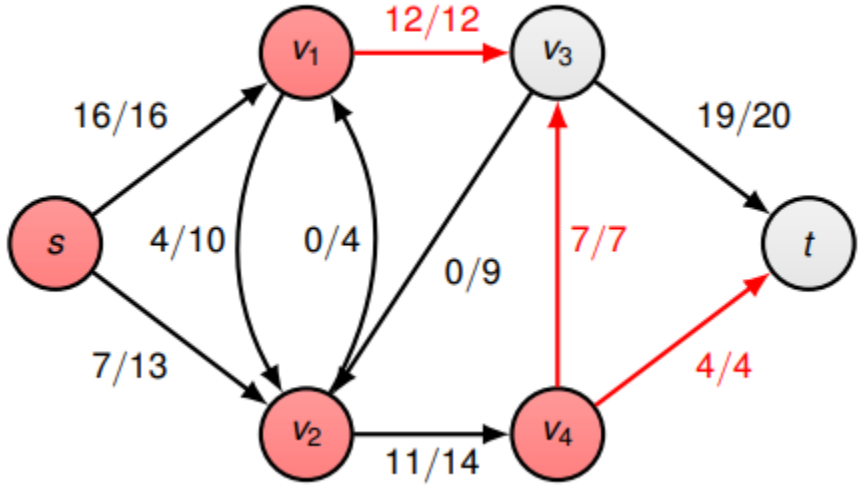
Definicion de corte

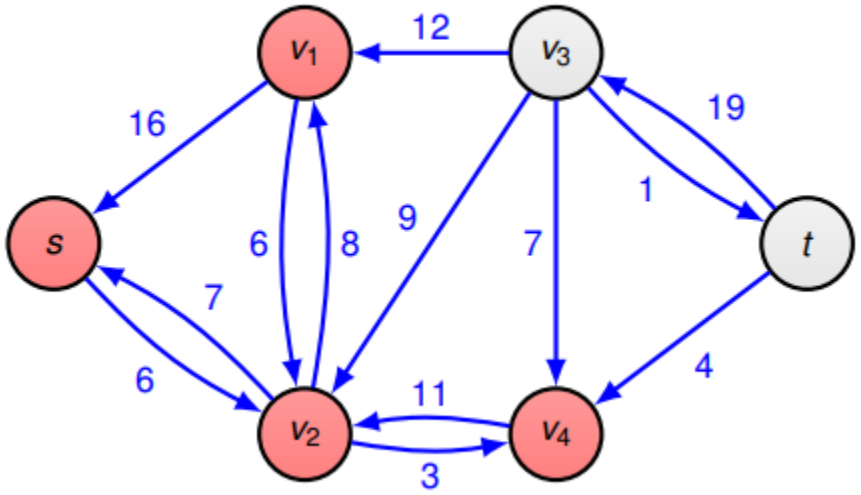
Un corte en una red es una partición de los nodos en dos conjuntos U y V tal que $s \in U$ y $t \in V$
.
Dado un corte en una red su valor es la suma de las capacidades de todas las aristas que van de un nodo de U a uno de V.



Teorema de Max-Flow Min-Cut

- El valor del corte mínimo coincide con el del flujo máximo.
- Podemos encontrar un corte mínimo definiendo:
  - U como los nodos alcanzables desde s en la red residual del flujo máximo.
  - V como el resto de los nodos.



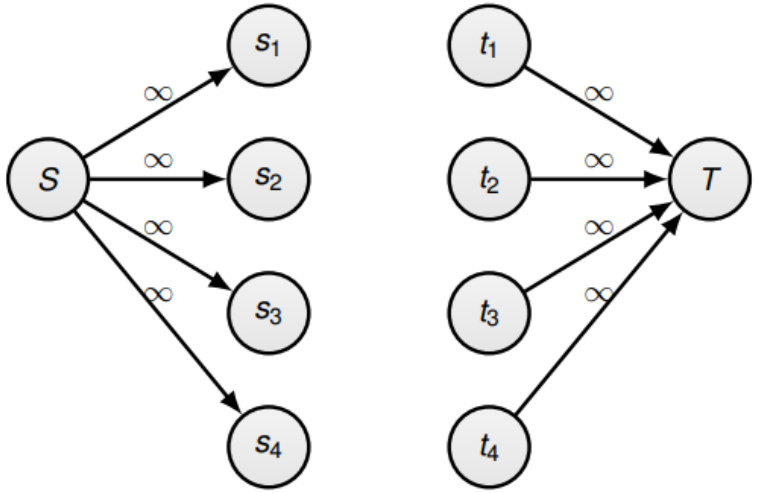


Trucos de Modelacion

Multiples fuentes/sumideros

Supongamos que tenemos un problema donde hay varios nodos de donde ”sale” flujo y/o varios nodos a los que tiene que llegar flujo, ¿Qué hacemos?

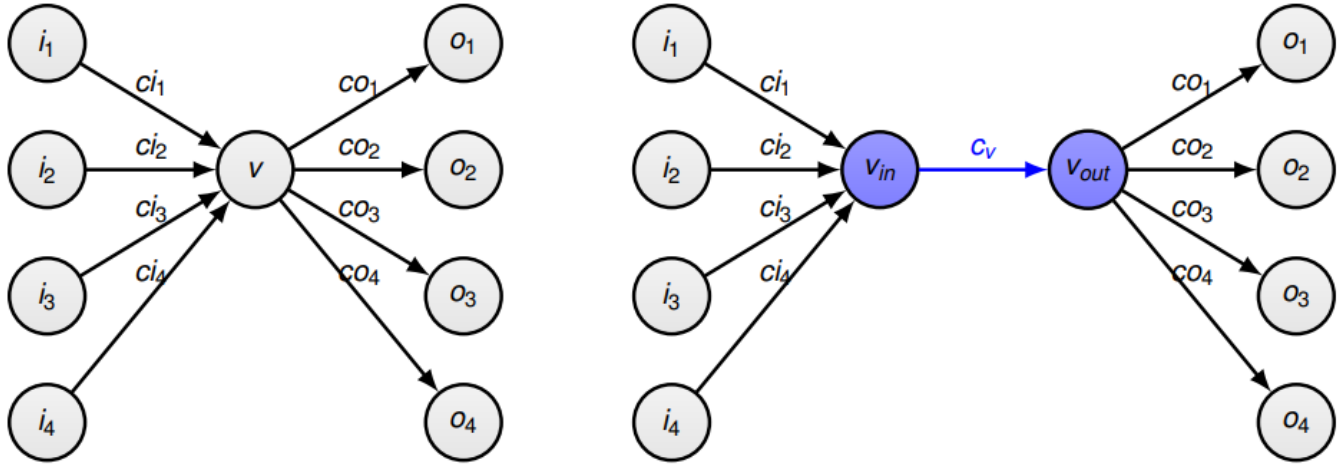
- Creamos dos nodos s y t nuevos.
- Conectamos s a todos los nodos de los que ”sale” flujo con capacidad ∞
- Conectamos todos los nodos a los que llega flujo con t con capacidad ∞.



Restricciones sobre los nodos

¿Qué pasa si tenemos restricciones de flujo que pasa por un nodo v en lugar de por un eje?

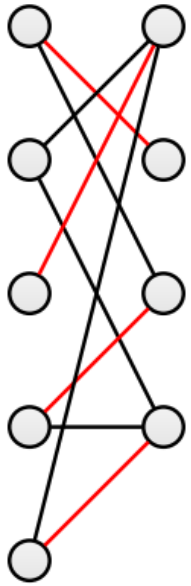
- Duplicamos el nodo en dos nodos v_{in} y v_{out}.
- Conectamos todas las aristas que entraban a v a v_{in}.
- Conectamos todas las aristas que salían de v a v_{out}.
- Conectamos v_{in} a v_{out} con un eje de la capacidad máxima del nodo



Los 4 jinetes del matching bipartito

Matching máximo

Dado un grafo G, un matching es un conjunto de aristas tal que no haya dos que incidan sobre el mismo vértice.
El problema de matching máximo pide el mayor conjunto de aristas que cumpla esto.



Esta version general del problema se puede resolver con el algoritmo de Blossom.

Matching máximo bipartito

Si el grafo es bipartito se puede modelar con un flujo de la siguiente manera:

- Si P y Q son los conjuntos de nodos de cada lado.
- Creo un nodo s y lo conecto con capacidad 1 a cada nodo de P.
- Creo un nodo t y conecto cada nodo de Q a t con capacidad 1.
- Por cada arista (p, q) del grafo uno a p con q con capacidad ∞.

El flujo máximo sobre esta red nos da el matching máximo.

Las aristas del medio por las que enviamos flujo forman el matching.

Minimal Edge cover

Dado un grafo, un cubrimiento por aristas es un conjunto de aristas tal que todo nodo sea extremo de alguna arista del conjunto.

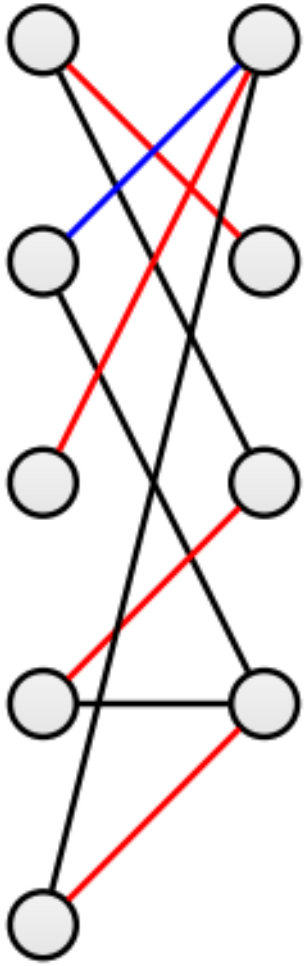
Ahora nos interesa saber el tamaño del cubrimiento de menor tamaño.

Si el grafo es bipartito se puede obtener a partir del matching máximo:

- Agrego al cubrimiento todas las aristas de un matching máximo.
- Por cada nodo que me quedó sin cubrir agrego una arista cualquiera que incida en el nodo.

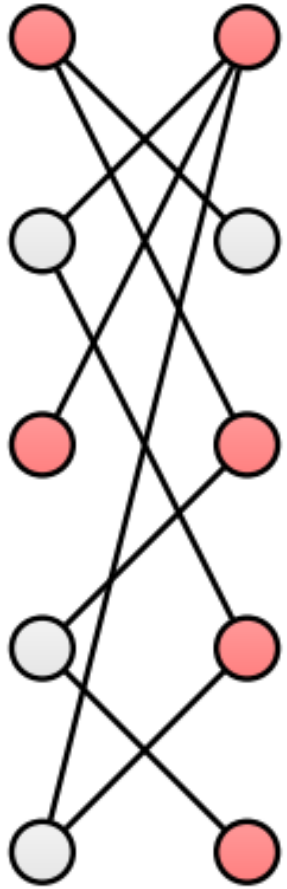
De esta manera, el tamaño del cubrimiento mínimo es $n - MM$ donde MM es el tamaño del matching máximo.

Minimal Edge cover ejemplo



Minimal Vertex Cover

Dado un grafo, un cubrimiento por nodos es un conjunto de nodos tal que toda aristas tenga algún extremo en el conjunto. Nuevamente nos interesa el cubrimiento de menor tamaño.

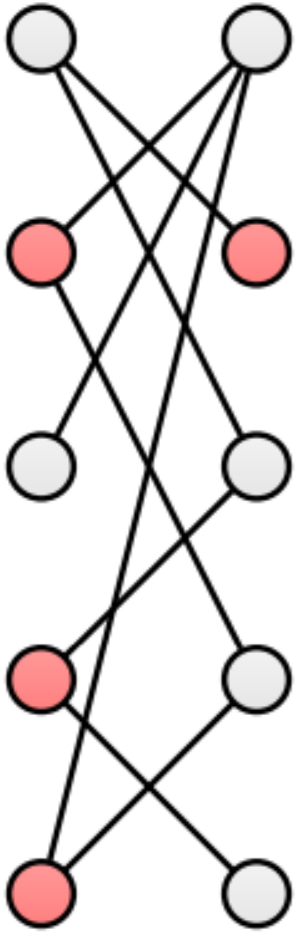


Minimal Vertex Cover caso bipartito

Si el grafo es bipartito, el teorema de Konig nos dice que el tamaño del cubrimiento mínimo coincide con el del matching máximo.

Maximal Independent Set

Dado un grafo, un conjunto independiente es un conjunto de nodos tal que no hay dos adyacentes. Queremos encontrar el conjunto independiente más grande.



Maximal Independent Set caso bipartito

Si el grafo es bipartito:

- Tomo un Vertex Cover mínimo.
- Los nodos que no están en el cubrimiento forman un conjunto independiente máximo.

Nuevamente, el tamaño del conjunto independiente más grande es $n - MM$.

Complejidad de flujo en un grafo bipartito

A la hora de calcular un matching máximo con flujo.

Si uso Edmond-Karp:

- El flujo máximo es a lo mucho $O(n)$.
- Usando la cota $O(mF)$ nos queda que el matching máximo es $O(mn)$.

Si uso Dinitz:

- La complejidad de encontrar el matching maximo en el grafo es de $O(m\sqrt{n})$.

Problemas con Tableros (Ajedrez)

Es muy común tener alguno de estos problemas escondido en algún problema de tableros.

Conviene pensar el siguiente modelo:

- Creo un grafo bipartito.
- Creo un nodo en P por cada fila del tablero.
- Creo un nodo en Q por cada columna del tablero.
- Una arista entre un nodo fila y un nodo columna representa la casilla de esa fila y columna.

Muchas veces en este modelo el problema se reduce a alguno de los 4 que vimos.

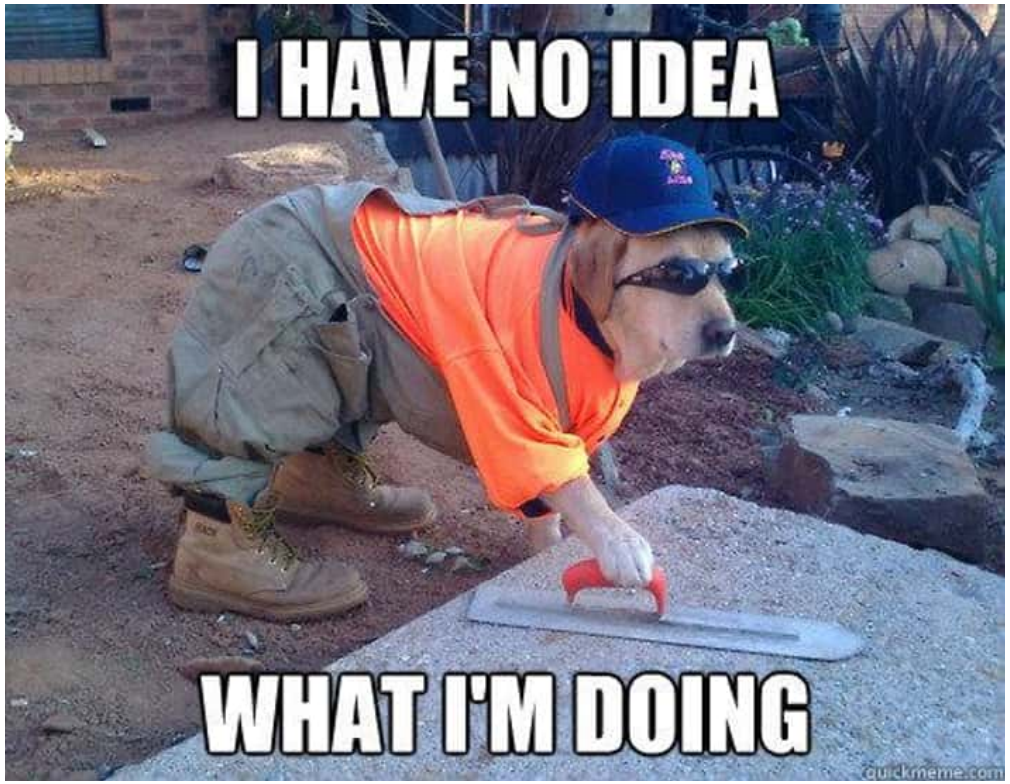
Algunos algoritmos extra

Min-Cost Max-Flow

Una variante del problema de flujo máximo es la siguiente:

- A cada arista, le agregamos un costo de mandar una unidad de flujo por esa arista.
- El costo de un flujo es la suma de los costos de las aristas multiplicado por el flujo que mandamos por cada una.
- Queremos encontrar entre todos los flujos máximos, aquel que minimice el costo total.

no sé



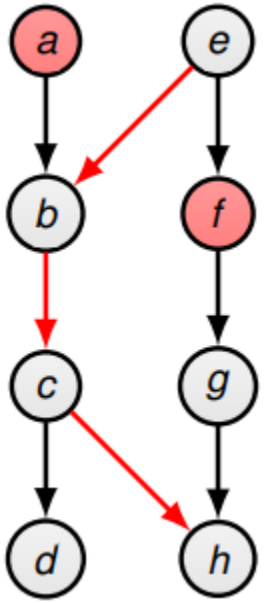
Intentar correr Ford Fulkerson y al momento de buscar un camino aumentante, utilizar Dijkstra o Bellman-Ford si los pesos son negativos.

Y si nos preguntan Min-Cost Max-flow para un matching maximo?

- Podemos modelar el problema exactamente igual y utilizamos el algoritmo de Min-Cost Max-Flow en el grafo
- (Opción más rapida) Utilizar el Hungarian algorithm que lo resuelve en $O(n^3)$

Dado un grafo acíclico dirigido (DAG) tenemos las siguientes definiciones:

- Una cadena es una secuencia de nodos tal que de cada uno puedo llegar al siguiente.
- Una anticadena es un conjunto de nodos tal que no hay camino entre ningún par de ellos.



Teorema de Dilworth

El teorema de Dilworth nos dice que los tamaños de estas cosas coinciden:

- La anticadena de mayor tamaño
- La menor cantidad de cadenas que cubren todos los nodos.

Grafo Auxiliar (Dilworth)

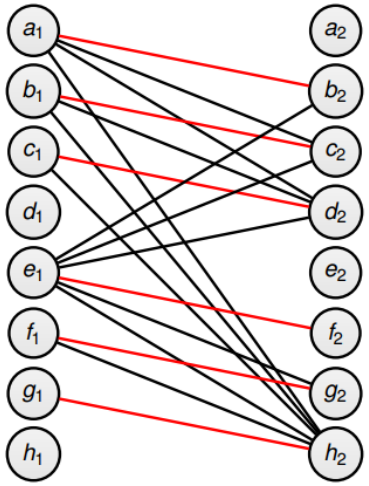
Construimos el siguiente grafo bipartito:

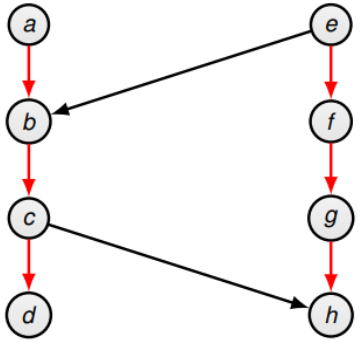
- Ponemos una copia de cada nodo del grafo de cada lado del bipartito.
- Unimos un nodo del lado izquierdo v con uno del derecho w si hay camino de v a w en el grafo original.

Minimal Chain Cover

Para el cubrimiento por cadenas:

- Obtenemos un matching máximo del grafo.
- Si un eje de v a w esta en el matching, agregamos la cadena formada por el camino de v a w en el grafo.
- Si las dos copias de un nodo están en el matching, unimos las dos cadenas con extremos en este nodo.

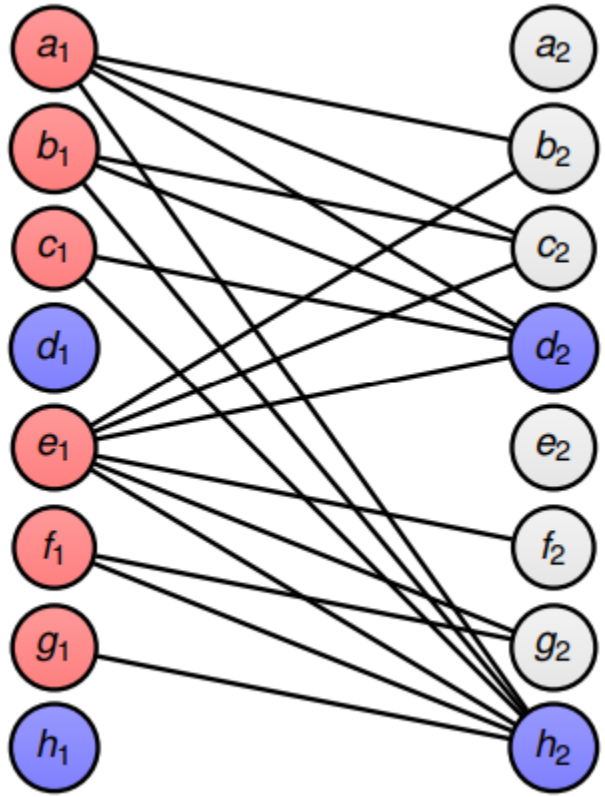




Anticadena máxima

Para la máxima anticadena:

- Obtenemos un Minimal Vertex Cover del grafo.
- Tomamos los nodos que no tienen ninguna copia en el cubrimiento y este es el maximo tamaño de la anticadena.



Felicitaciones a Ustedes

Gracias por asistir al campamento. Estamos muy orgullosos del crecimiento de la comunidad y nos alegra ver su motivación.