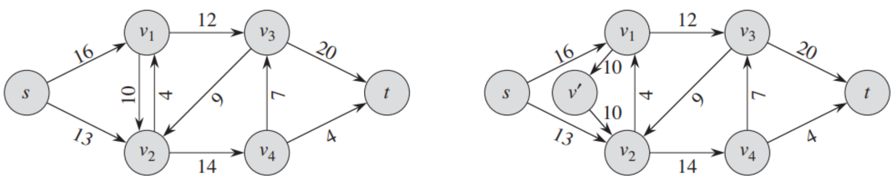
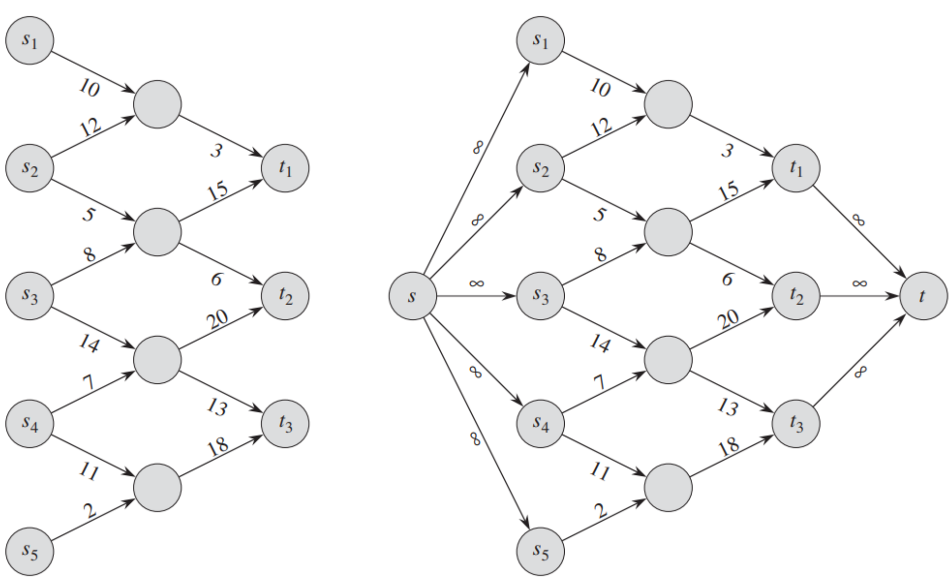
Flujo Máximo

- El problema de flujo máximo consiste en obtener la cantidad de flujo máximo que podemos transportar desde el source al sink conservando el flujo de todos los vertices intermedios.
- **Conservación de flujo**: El flujo total que entra a un vertice tiene que ser igual al flujo total que sale del mismo. (Excepto soruce y sink).
- **flow network**: Un grafo dirigido sin ciclos $G = (V, E)$ donde todos los $(u, v) \in G.E$ tienen un capacidad $c(u, v) \geq 0$. Y se distinguen 2 vertices: s (source) y t (sink). Las aristas que no estan en el grafo tienen costo 0. Y para cada vertice v del grafo, hay un camino de s a v y de v a t (grafo conectado, $|E| \geq |V|$). (Tampoco se permiten self-loops).
- *Una restricción adicional*: si $(u, v) \in E, entonces (v, u) \notin E$. Si tenemos la arista de ida y quicieramos agregar la de la vuelta, podríamos agregar un vertice intermedio v' talque $v \to v'$ y $v' \to u$ ambas con el costo deseado.



(a) (b)

- **flujo**: Dado un grafo de flujo con función de capacidad c, soruce s y sink t, se define el *flujo* como $f : V \times V \to \mathbb{R}$ (recibe una arista y devuelve un real), que satisface las restricciones de:
  - Capacidad: $0 \leq f(u, v) \leq c(u, v)$
  - Conservación: $\sum_{v \in V} f(v, u) = \sum_{v \in V} f(u, v)$. Para $(u, v) \in E, sino f(u, v) = 0$
- **Cantidad de flujo**: $|f| = \sum_{v \in V} f(s, v) - \sum_{v \in V} f(v, s)$, (el segundo término es 0). El problema de máximo flujo desea maximizar ésta cantidad.
- **Multiples sources y sinks**: Se quiere obtener el máximo flujo pero ahora se tiene un conjunto $\{s_1, ..., s_m\}$ de sources y un conjunto $\{t_1, ..., t_n\}$ de sinks. Se puede modelar el problema agregando un "supersource" s conectado a todos los sources tq $c(s, s_i) = \infty$, y un "supersink" t conectado a todos los sinks tq $c(t_i, t) = \infty$. Y el problema es equivalente a el de un source y un sink.



(a) (b)

El Método de Ford-Fulkerson

- Se lo llama método ya que puede tener distintas implementaciones con distintas complejidades.

- Iterativamente incrementa el valor del flow. Se inicia con $f(u, v) = 0$ para todas las aristas, dando un valor de flujo inicial de 0. En cada iteración se busca un camino de aumento en el "grafo resudial" que nos permitirá saber que aristas aumentar y disminuir el flujo para aumentar el flujo total. Se repite hasta que en el grafo residual no haya más caminos de aumento.

```
Ford-Fulkerson-Method(G, s, t):
    initialize flow f to 0
    while there exists an augmenting path p in the residual network G_f
        augment flow f along p
    return f
```

Grafos reciduales

- Dado un grafo G de flujo y f un flujo, el grafo residual G_f consiste en los mismos vertices de G pero con aristas que representan los aumentos y decrementos posibles al flujo de las aristas de G. La diferencia es que si en G está la arista (u, v), en G_f también está la arista (v, u).
- **Capacidad residual**: definimos las capacidades de las aristas del grafo residual cómo:

$$c_f(u, v) = \begin{cases} c(u, v) - f(u, v) & \text{if } (u, v) \in E \ , \\ f(v, u) & \text{if } (v, u) \in E \ , \\ 0 & \text{otherwise} \ . \end{cases}$$

- Si $(u, v) \in G.E$, la capacidad residual indica cuanto flujo más podemos enviar desde u hacia v (capacidad de la arista menos el flujo que pasa por dicha arista).
- Si $(u, v) \notin G.E$, entonces si $(v, u) \in G.E$ indica cuanto flujo podemos "devolver" de v hacia u. (que es justamente el flujo que hay en G de v hacia u).
- Caso contrario en G se está enviando el maximo flujo posible o no hay arista de u a v ni v a u.
- $|E_f| \leq 2 * |E|$

El flujo en el grafo recidual indica cómo cambiar el flujo en el grafo original para aumentar el flujo total. Definimos el **aumento** de flujo de f por f' como:

A flow in a residual network provides a roadmap for adding flow to the original flow network. If f is a flow in G and f' is a flow in the corresponding residual network G_f, we define $f \uparrow f'$, the ***augmentation*** of flow f by f', to be a function from $V \times V$ to $\mathbb{R}$, defined by

$$(f \uparrow f')(u, v) = \begin{cases} f(u, v) + f'(u, v) - f'(v, u) & \text{if } (u, v) \in E \ , \\ 0 & \text{otherwise} \ . \end{cases} \qquad (26.4)$$

Lemma 24.1

Let $G = (V, E)$ be a flow network with source s and sink t, and let f be a flow in G. Let G_f be the residual network of G induced by f, and let f' be a flow in G_f. Then the function $f \uparrow f'$ defined in equation (24.4) is a flow in G with value $|f \uparrow f'| = |f| + |f'|$.

Aumentando caminos

- **Camino de aumento**: Dado un grafo de flujo $G = (V, E)$ y un flujo f, un camino de aumento es un camino de s (source) a t (sink) en el grafo recidual G_f.
- Llamamos a la máxima cantidad de flujo que podes aumentar en cada arista de un camino de aumento p, **capacidad residual** de p: $c_f = min\{c_f(u, v) : (u, v) \in p\}$
- Podemos definir un flujo en base a la capacidad residual para poder usar el lema 24.1 y aumentar el flujo de G. Esto es, 0 si no pertenece al camino de aumento, y si sí pertenece el flujo vale la capacidad residual.

Lemma 24.2

Let $G = (V, E)$ be a flow network, let f be a flow in G, and let p be an augmenting path in G_f. Define a function $f_p : V \times V \to \mathbb{R}$ by

$$f_p(u, v) = \begin{cases} c_f(p) & \text{if } (u, v) \text{ is on } p \,, \\ 0 & \text{otherwise} \,. \end{cases} \tag{24.7}$$

Then, f_p is a flow in G_f with value $|f_p| = c_f(p) > 0$. ■

Luego con el lema 24.1, podemos aumentar el flujo de G.

Corollary 24.3

Let $G = (V, E)$ be a flow network, let f be a flow in G, and let p be an augmenting path in G_f. Let f_p be defined as in equation (24.7), and suppose that f is augmented by f_p. Then the function $f \uparrow f_p$ is a flow in G with value $|f \uparrow f_p| = |f| + |f_p| > |f|$.

Proof Immediate from Lemmas 24.1 and 24.2. ■

Cortes en grafo de flujos

- **Corte (S, T)**: Partición de V en S y T (son disjuntos y su unión da V), tal que $s \in S$ y $t \in T$.
- **Flujo neto del corte**: Si f es el flujo de $G = (V, E)$, entonces el flujo neto del corte se define como:

  $\mathbf{f(S, T)} = \sum_{\mathbf{u \in S}} \sum_{\mathbf{v \in T}} \mathbf{f(u, v)} - \sum_{\mathbf{u \in S}} \sum_{\mathbf{v \in T}} \mathbf{f(v, u)}$.

  (La suma del flujo que va desde S hacia T menos las que van de T hacia S).
- **Capacidad del corte**: $\mathbf{c(S, T)} = \sum_{\mathbf{u \in S}} \sum_{\mathbf{v \in T}} \mathbf{c(u, v)}$.
- Un **corte mínimo** es aquel cuya capacidad es mínima entre todos los cortes posibles.

Lemma 24.4

Let f be a flow in a flow network G with source s and sink t, and let (S, T) be any cut of G. Then the net flow across (S, T) is $\underline{f(S, T) = |f|}$.

- **Colorario 24.5**: El valor de cualquier flujo f en G está acotado por arriba por la capacidad de cualquier corte. En consecuencia, el valor del máximo flow está acotado por la capacidad del mínimo corte.
- **Teorema Max-flow Min-cut**: Sea f un flujo en el grafo de flujo $G = (V, E)$ con source s y sink t, las siguientes condiciones son equivalentes (una implica la otra):
  - f es el flujo máximo de G.
  - El grafo residual G_f no tiene ningun camino de aumento (de s a t).
  - $|f| = c(S, T)$ para algún corte (S, T) de G.

Algoritmo de Ford-Fulkerson

- Se empieza con un flujo 0 para todas las aristas.
- Se asume que para las aristas que no perteneces al grafo, tanto la capacidad y el flujo son 0 siempre.
- Iterativamente se busca un camino de aumento en el grafo residual y se calcula la capacidad residual del camino encontrado. Luego se actualizan los valores de las aristas tanto en el grafo original como en el residual. Termina cuando ya no hay caminos posibles de s a t en el grafo residual.
- **Complejidad**: $O(m * F)$ ya que en el peor caso en cada iteración se aumenta en 1 el flujo.

```
FORD-FULKERSON(G, s, t)
for each edge (u, v) ∈ G.E
    (u, v).f = 0
while there exists a path p from s to t in the residual network G_f
    c_f(p) = min {c_f(u, v) : (u, v) is in p}
    for each edge (u, v) in p
        if (u, v) ∈ G.E
            (u, v).f = (u, v).f + c_f(p)
        else (v, u).f = (v, u).f − c_f(p)
return f
```

- **Teorema**: Si las capacidades de los arcos de la red son enteras, entonces el problema de flujo máximo tiene un flujo máximo entero.
- Si las capacidades o el flujo inicial son n´umeros irracionales, el método de Ford y Fulkerson puede no parar (es decir, realizar un número infinito de pasos).

Algoritmo de Edmonds-Karp

Consiste en el método de Ford-Fulkerson en el que se usa BSF para encontrar el camino de aumento. El algoritmo corre en tiempo polinomial y podemos acotarlo sin depender del valor del flujo.

Lemma 24.7
If the Edmonds-Karp algorithm is run on a flow network $G = (V, E)$ with source s and sink t, then for all vertices $v \in V - \{s, t\}$, the shortest-path distance $\delta_f(s, v)$ in the residual network G_f increases monotonically with each flow augmentation.

Theorem 24.8
If the Edmonds-Karp algorithm is run on a flow network $G = (V, E)$ with source s and sink t, then the total number of flow augmentations performed by the algorithm is $O(VE)$.

Sigue que como el costo de cada iteración del algoritmo de ford-fulkenson cuesta $O(m)$ y hacemos $O(nm)$ iteraciones, la **complejidad** del algoritmo queda $O(nm^2)$

Theorem 24.10 (Integrality theorem)
If the capacity function c takes on only integer values, then the maximum flow f produced by the Ford-Fulkerson method has the property that $|f|$ is an integer. Moreover, for all vertices u and v, the value of $f(u, v)$ is an integer.