DEPARTAMENTO DE
INGENIERÍA
INFORMÁTICA
UNIVERSIDAD DE SANTIAGO DE CHILE

Estructuras de Datos

Clase Coordinada - Unidad nro. III: Grafos

10-05-2022

DEPARTAMENTO DE
INGENIERÍA
INFORMÁTICA
UNIVERSIDAD DE SANTIAGO DE CHILE

Hasta ahora…

- Terminología y algunas propiedades
- Representación de un grafo
  - Matriz de adyacencia
  - Listas de adyacencia

DEPARTAMENTO DE INGENIERÍA INFORMÁTICA
UNIVERSIDAD DE SANTIAGO DE CHILE

Contenido

- Recorridos en grafos
- Camino mínimo
- Flujo máximo
- Tarea

DEPARTAMENTO DE INGENIERÍA INFORMÁTICA UNIVERSIDAD DE SANTIAGO DE CHILE

Recorridos en grafos

Problema:

Dado un grafo, determinar un procedimiento para visitar todos sus vértices y aristas.

- Sea $G(V, A)$ conexo:

  elegir un vértice inicial y desde ahí revisar los vértices y aristas adyacentes, marcándolos a medida que se vayan visitando, hasta que se hayan explorado todos los vértices y sus respectivas aristas.

Recorridos en grafos

- Se denomina:
  - raíz r del recorrido al vértice inicial elegido.
  - arista explorada: arista (v, w) escogida a partir del v seleccionado.
  - vértice alcanzado: vértice w al que se llega con la arista seleccionada.
  - vértice explorado: vértice en el que todas sus aristas incidentes fueron exploradas, visitadas o están marcadas.

DEPARTAMENTO DE INGENIERÍA INFORMÁTICA
UNIVERSIDAD DE SANTIAGO DE CHILE

Recorridos en grafos

- En todo proceso de recorrido se debe escoger:
  - raíz del recorrido: vértice inicial elegido.
  - vértice marcado v a partir del cual continuar el recorrido.
  - arista (v, w) incidente al vértice v seleccionado.
- Criterio de selección del próximo nodo a explorar:
  - Depende del contexto del proceso de recorrido. Nosotros escogeremos el orden léxico entre los nodos.

DEPARTAMENTO DE INGENIERÍA INFORMÁTICA
UNIVERSIDAD DE SANTIAGO DE CHILE

Recorrido en amplitud

- Criterio: entre todos los vértices marcados e incidentes a alguna arista aún no explorada, se escoge aquel menos recientemente alcanzado en el recorrrido.
- Este método se implementa utilizando una cola.
- Complejidad: visita todos los vértices y todas las aristas del grafo, $O(n+m)$ es decir es lineal en el tamaño del grafo.

Recorrido en amplitud

Algoritmo: Recorrido en amplitud

```
    Dado G(V,A) conexo
    Desmarcar todos los vértices
    escoger una raíz s ∈ V
    definir una cola Q vacía
    marcar s
    insertar s en Q
    mientras Q ≠ ∅ hacer
        sea v el primer elemento de Q
        para todo w ∈ Adya(v) hacer
                si w no está marcado entonces
                        marcar (v,w) tipo I
                        marcar w
                        insertar w en Q
                caso contrario
                        si w ∈ Q entonces marcar (v,w) tipo II
        retirar v de Q
```

DEPARTAMENTO DE INGENIERÍA INFORMÁTICA UNIVERSIDAD DE SANTIAGO DE CHILE

Recorrido en amplitud

- Se distinguen dos tipos de aristas marcadas
- Aristas tipo I: aristas (v, w) en que el vértice w no ha sido visitado
- Aristas tipo II: el vértice w ya ha sido visitado

Árbol de anchura:

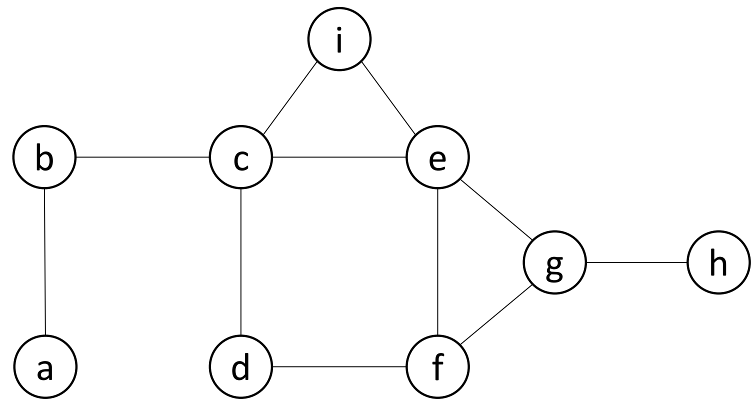
Vértices del grafo inicial y todas las aristas de tipo I, corresponde a un árbol de expansión del grafo.

DEPARTAMENTO DE
INGENIERÍA
INFORMÁTICA
UNIVERSIDAD DE SANTIAGO DE CHILE

Recorrido en amplitud

Algoritmo: Recorrido en amplitud

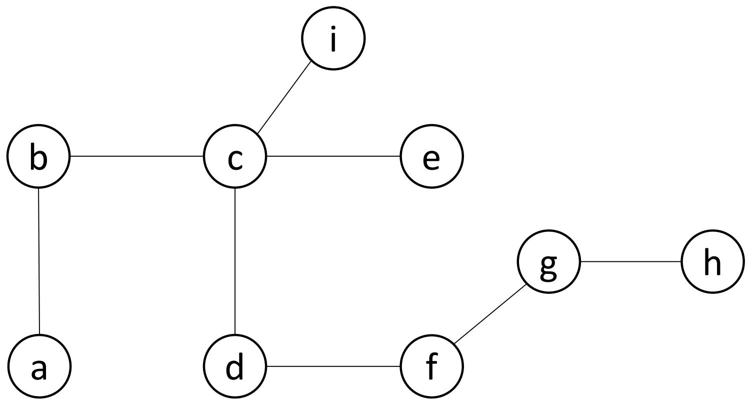
```
Dado G(V,A) conexo
Desmarcar todos los vértices
escoger una raíz s ∈ V
definir una cola Q vacía
marcar s
insertar s en Q
mientras Q ≠ ∅ hacer
      sea v el primer elemento de Q
      para todo w ∈ Adya(v) hacer
                  si w no está marcado entonces
                              marcar (v,w) tipo I
                              marcar w
                              insertar w en Q
                  caso contrario
                              si w ∈ Q entonces marcar (v,w) tipo II
      retirar v de Q
```



DEPARTAMENTO DE INGENIERÍA INFORMÁTICA
UNIVERSIDAD DE SANTIAGO DE CHILE

Recorrido en amplitud

- Árbol de amplitud:



DEPARTAMENTO DE INGENIERÍA INFORMÁTICA
UNIVERSIDAD DE SANTIAGO DE CHILE

Recorrido en profundidad

- Criterio: entre todos los vértices marcados y adyacentes a alguna arista no visitada se escoge aquel más recientemente alcanzado en el recorrido.
- Para su implementación se utiliza una pila.
- Complejidad: visita todos los vértices y todas las aristas del grafo, O(n+m) es decir es lineal en el tamaño del grafo.

Recorrido en profundidad

Algoritmo: Recorrido en profundidad

```
Dado G(V,A) conexo
desmarcar todos los vértices
definir una pila vacía S
escoger una raíz r ∈ V
P(r)

Procedimiento P(v)
marcar v
colocar v en S
para todo w ∈ Adya(v) hacer
    si w no está marcado entonces
            marcar (v,w) tipo I
            P(w)
    caso contrario
            si w ∈ S y v con w no son consecutivos, entonces marcar (v,w) tipo II
    retirar v de S
```

DEPARTAMENTO DE INGENIERÍA INFORMÁTICA
UNIVERSIDAD DE SANTIAGO DE CHILE

Recorrido en profundidad

- El recorrido en profundidad, al igual que el recorrido en amplitud, particiona el conjunto de aristas A en dos tipos.
- Las aristas tipo I corresponden a las aristas del árbol de profundidad que es un árbol de expansión del grafo
- Las aristas tipo II a las aristas de retorno.

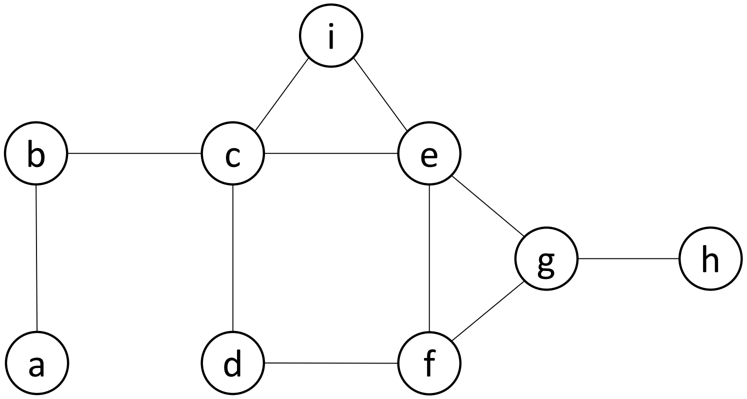
Árbol de profundidad:

Vértices del grafo inicial y todas las aristas de tipo I, corresponde a un árbol de expansión del grafo.

Recorrido en profundidad

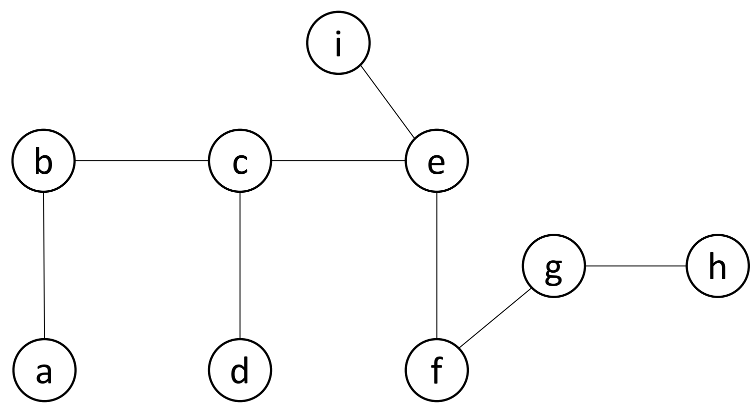
Algoritmo: Búsqueda en Profundidad

```
Dado G(V,A) conexo
desmarcar todos los vértices
definir una pila vacía S
escoger una raíz r ∈ V
P(r)

Procedimiento P(v)
marcar v
colocar v en S
para todo w ∈ Adya(v) hacer
      si w no está marcado entonces
                  marcar (v,w) tipo I
                  P(w)
      caso contrario
                  si w ∈ S y v con w no son consecutivos, entonces marcar (v,w) tipo II
      retirar v de S
```



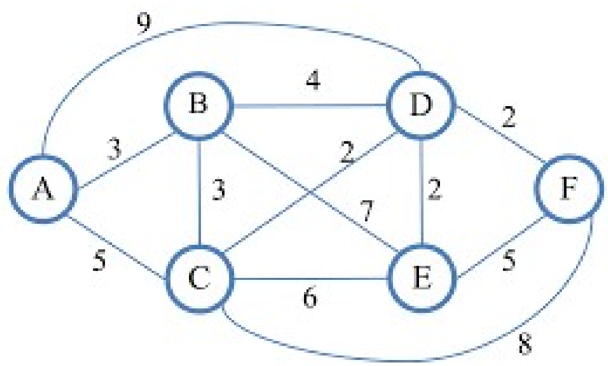
Recorrido en amplitud

- Árbol de profundidad:



Problema del camino mínimo

- En teoría de grafos, el problema del camino mínimo es el problema de encontrar una ruta entre dos vértices (o nodos) en un grafo ponderado de modo que se minimice la suma total de los pesos de sus aristas.



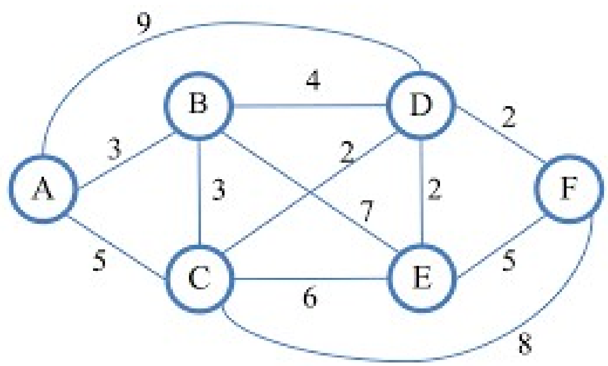
DEPARTAMENTO DE INGENIERÍA INFORMÁTICA
UNIVERSIDAD DE SANTIAGO DE CHILE

Algoritmo de Dijkstra

- Este algoritmo nace de la técnica de programación dinámica, partiendo de cualquier nodo origen para así lograr su optimización.
- Mantiene un conjunto *S* de vértices cuya distancia al nodo origen es mínima y conocida (nodos alcanzables). *S* inicialmente mantiene solo el nodo origen.
- El algoritmo trabaja por medio de iteraciones las cuales agregan un nuevo nodo cuya distancia o costo desde el origen es el menor posible

Algoritmo de Dijkstra

- En cada iteración, el algoritmo encuentra el nodo que no está en S y cuyo camino óptimo tentativo es mínimo. Este nodo se agrega a S y su camino óptimo tentativo se convierte en su camino óptimo. Luego se actualizan los caminos óptimos tentativos para los demás nodos.



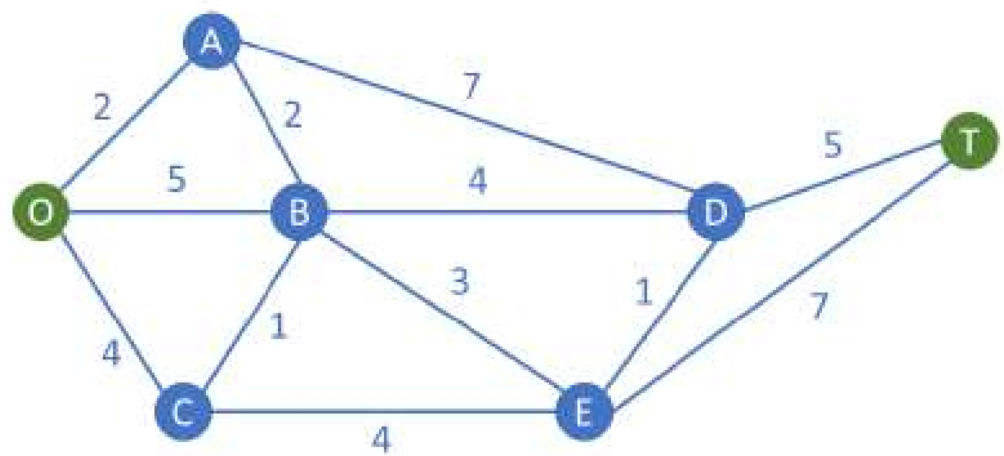
DEPARTAMENTO DE INGENIERÍA INFORMÁTICA
UNIVERSIDAD DE SANTIAGO DE CHILE

Algoritmo de Dijkstra

```
Algoritmo: Dijkstra

Dado G(V,A) un grafo ponderado y conexo
S = {f};
D[f] = 0;
D[v] = costo(f,v) para todo v en V-S // infinito si ∄(f,v)
C[v] = f para todo v // indefinido si ∄(f,v)
mientras(S!=V)
    Encontrar v en V-S tal que D[v] es mínimo
    Agregar v a S
    para todo w tal que (v,w) está en A
            D[w] = min(D[w], D[v] + costo(v,w))
            C[w] = v
Retornar D, C
```

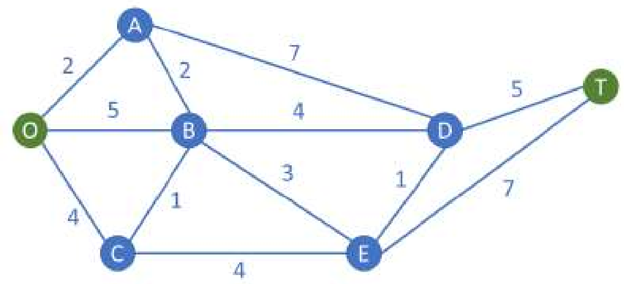
Algoritmo de Dijkstra

- Encontrar el camino mínimo:



DEPARTAMENTO DE
INGENIERÍA
INFORMÁTICA
UNIVERSIDAD DE SANTIAGO DE CHILE

Algoritmo de Dijkstra

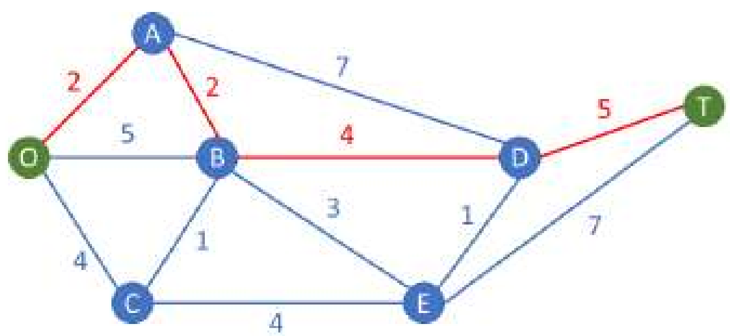


Conjunto *S*	A Pred.	B Pred.	C Pred.	D Pred.	E Pred.	T Pred.

DEPARTAMENTO DE
INGENIERÍA
INFORMÁTICA
UNIVERSIDAD DE SANTIAGO DE CHILE

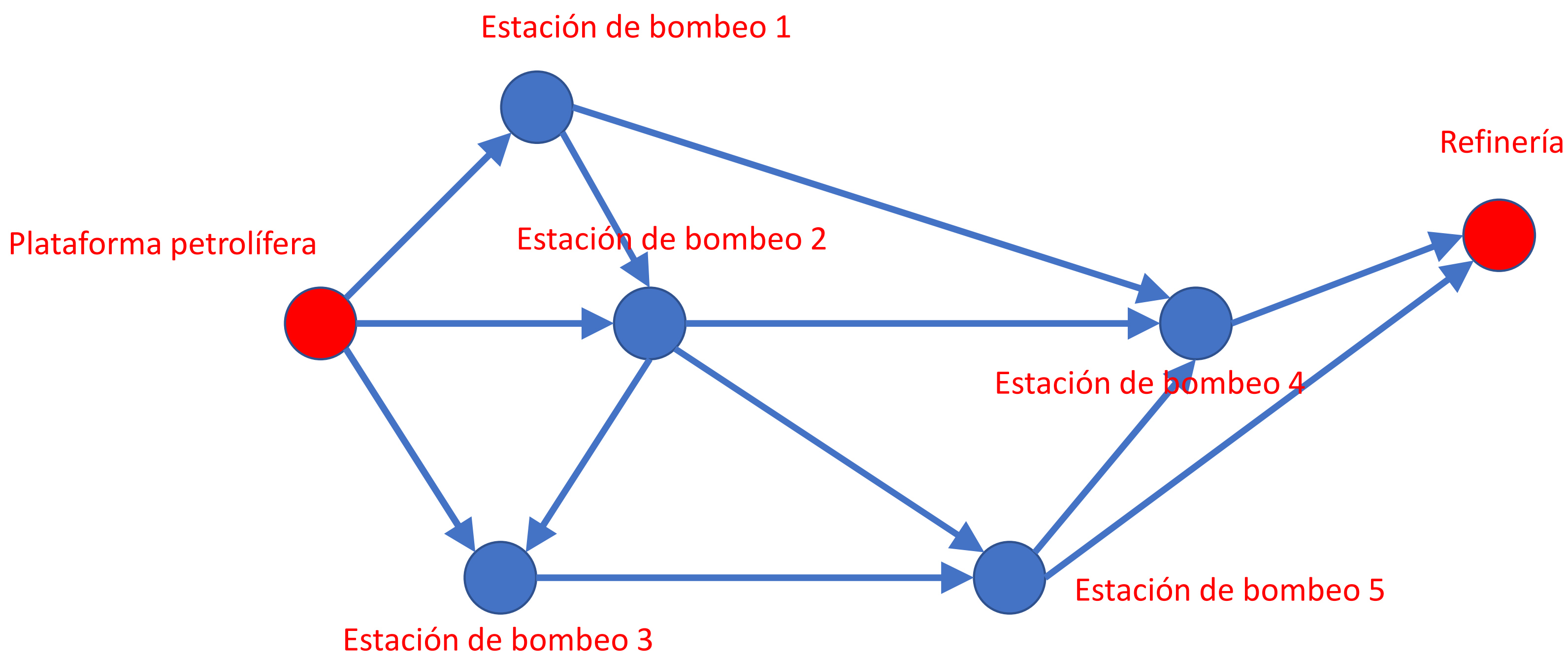
Algoritmo de Dijkstra

Conjunto S	A - Pred.	B - Pred.	C - Pred.	D - Pred.	E - Pred.	T - Pred.
$\{O\}$	2 O	5 O	4 O	∞ -	∞ -	∞ -
$\{O, A\}$	——	4 A	4 O	9 A	∞ -	∞ -
$\{O, A, B, C\}$		——	——	8 B	7 B	∞ -
$\{O, A, B, C, E\}$				8 B	——	14 E
$\{O, A, B, C, E, D\}$				——		13 D
$\{O, A, B, C, E, D, T\}$						——
	2 O	4 A	4 O	8 B	7 B	13 D



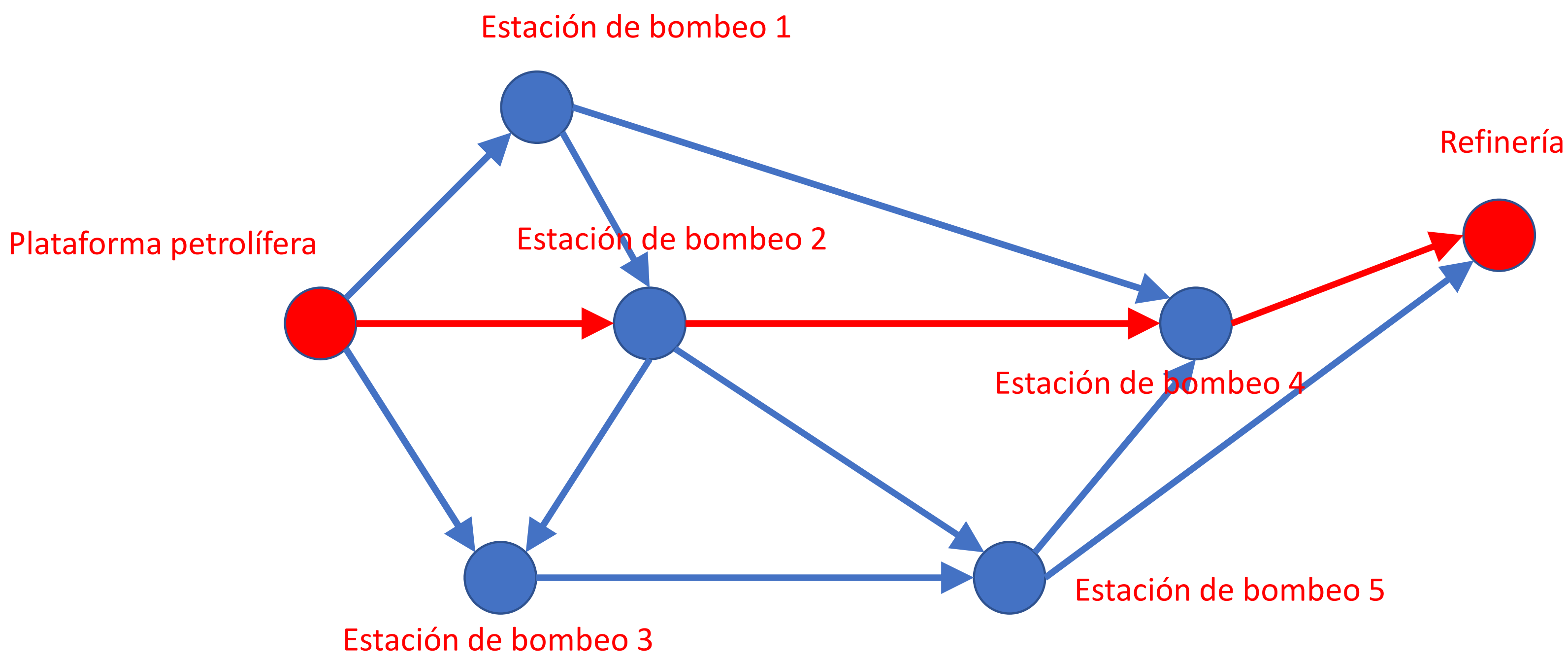
DEPARTAMENTO DE INGENIERÍA INFORMÁTICA UNIVERSIDAD DE SANTIAGO DE CHILE

¿Cómo transportar?

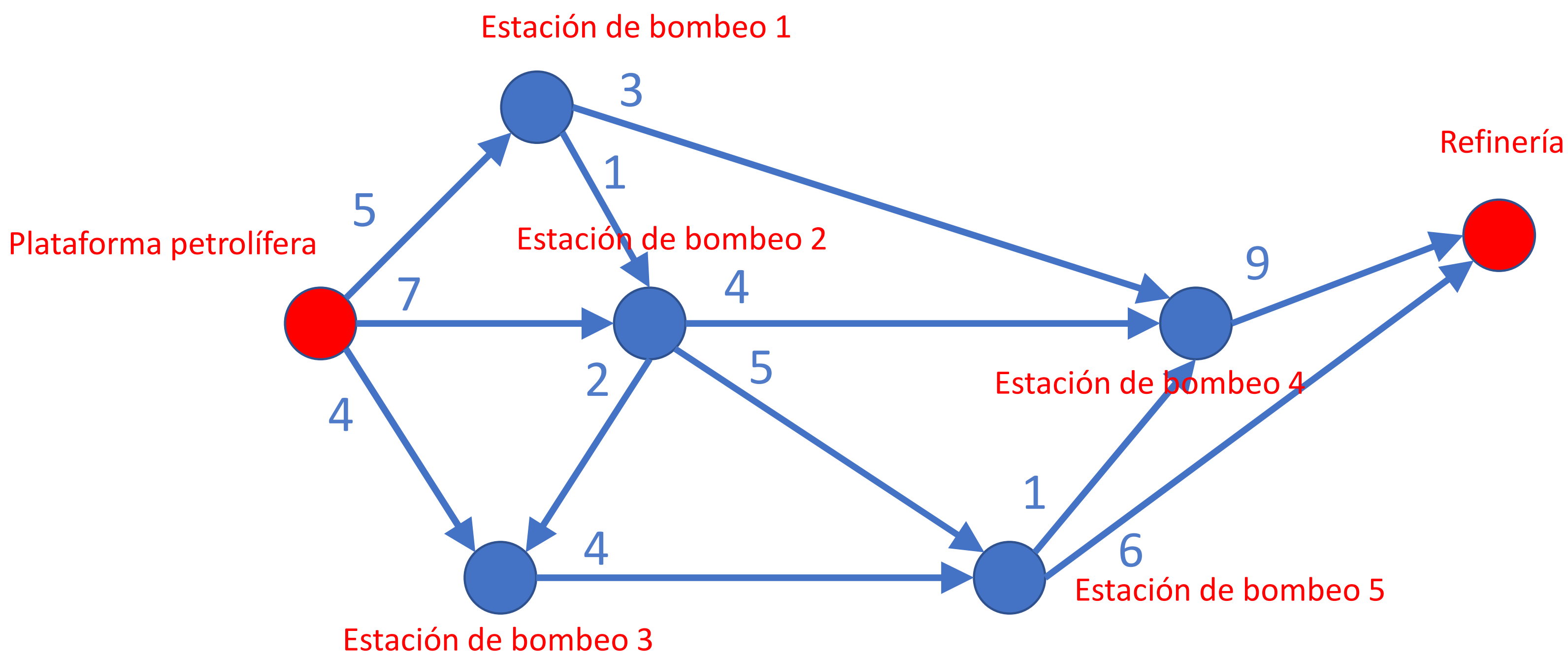


¿Cómo transportar?

- Calidad de la solución, factibilidad.

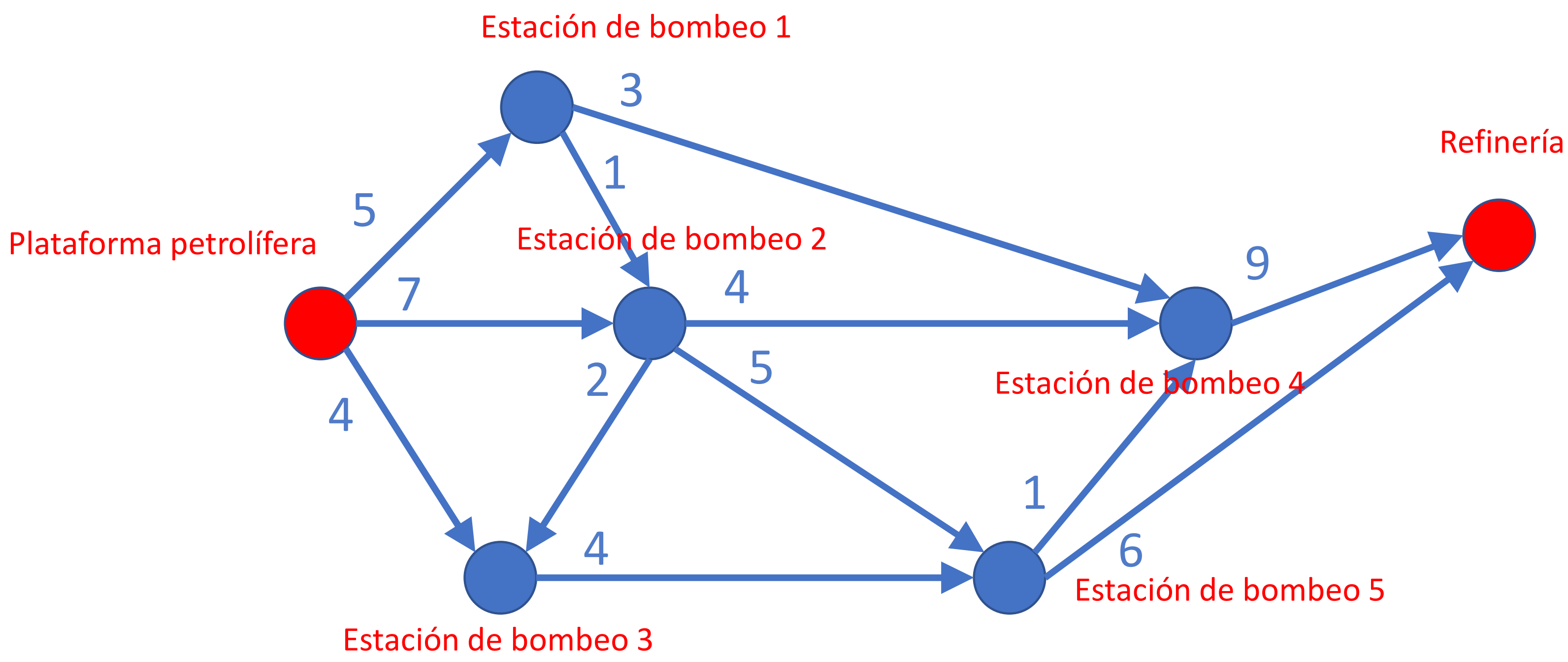


Modelo del problema



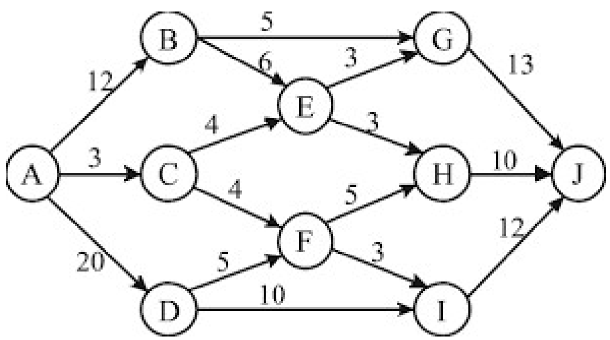
Modelo del problema

- ¿Solución óptima?



Problema de fujo máximo

- Dado un grafo dirigido, determinar el mayor flujo que sea posible enviar desde un nodo fuente a uno de destino. Cada arista tiene asociada una capacidad.



Algoritmo de Ford-Fulkerson

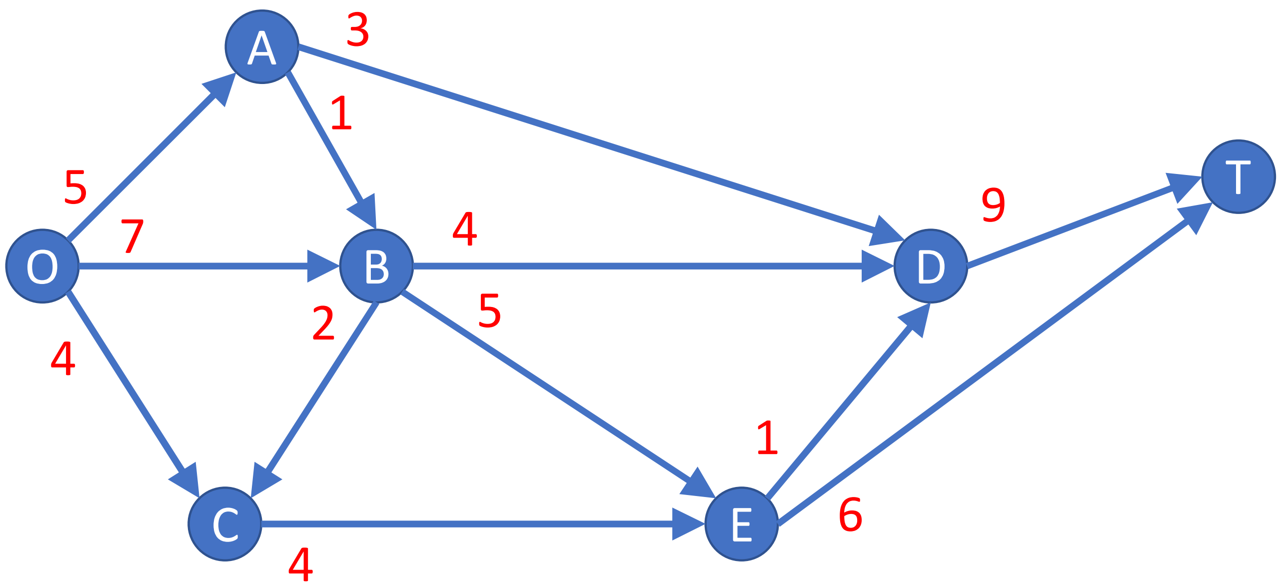
- Propone buscar caminos en los que se pueda aumentar el flujo, hasta que se alcance el flujo máximo
- Camino de aumento: es una camino dirigido desde el nodo origen hasta el nodo destino en la red residual, tal que cada arista sobre ella tenga capacidad residual estrictamente positiva.
- Red residual: muestra las capacidades restantes (capacidades residuales) de las aristas del grafo.

Algoritmo de Ford-Fulkerson

```
Algoritmo: Ford-Fulkerson
     Dado G(V,A) conexo y dirigido

     1 Convertir el grafo dirigido en red residual.

     2 Identificar un camino de aumento, si no existe uno, los
       flujos netos asignados constituyen un patrón de fujo óptimo.

     3 Encontrar el mínimo de las capacidades residuales de las
       aristas sobre el camino de aumento. Se aumenta en el valor
       identificado el flujo asignado de cada una de las aristas del camino.

     4 Se disminuye en el mínimo de las capacidades residuales identificado
       cada arista del camino de aumento. Se regresa al paso 2.
```

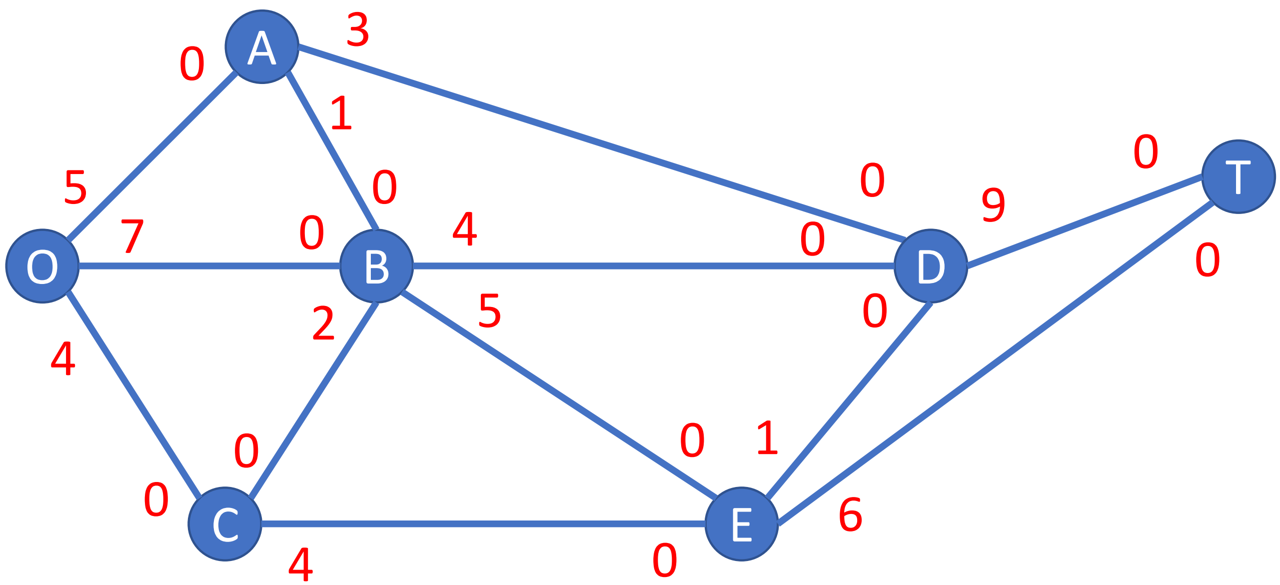
Algoritmo de Ford-Fulkerson

- Encontrar el flujo máximo en el siguiente grafo:



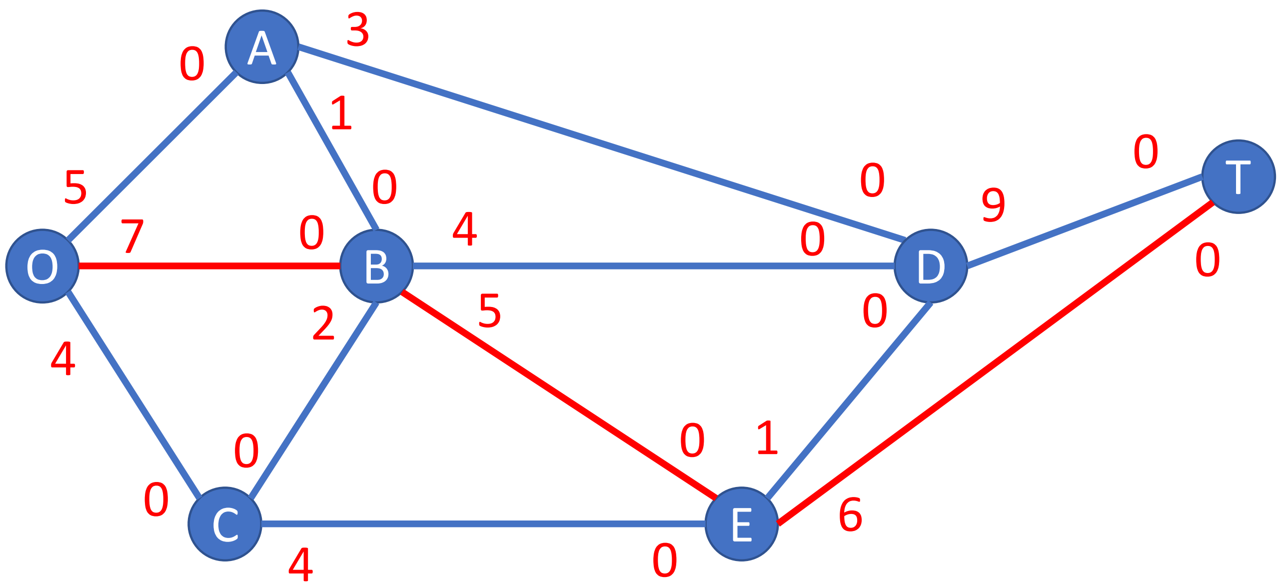
Algoritmo de Ford-Fulkerson

- Red residual inicial



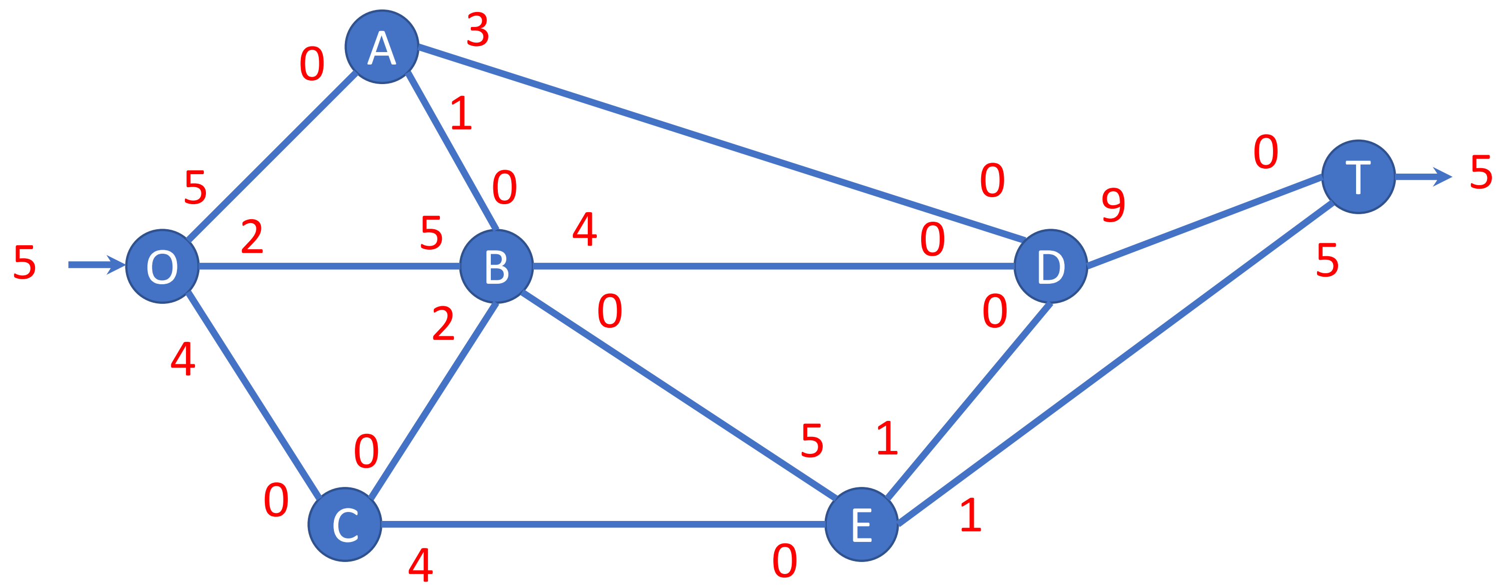
Algoritmo de Ford-Fulkerson

- Iteración 1: trayectoria de aumento es O → B → E → T con capacidad residual igual al min {7, 5, 6} = 5



Algoritmo de Ford-Fulkerson

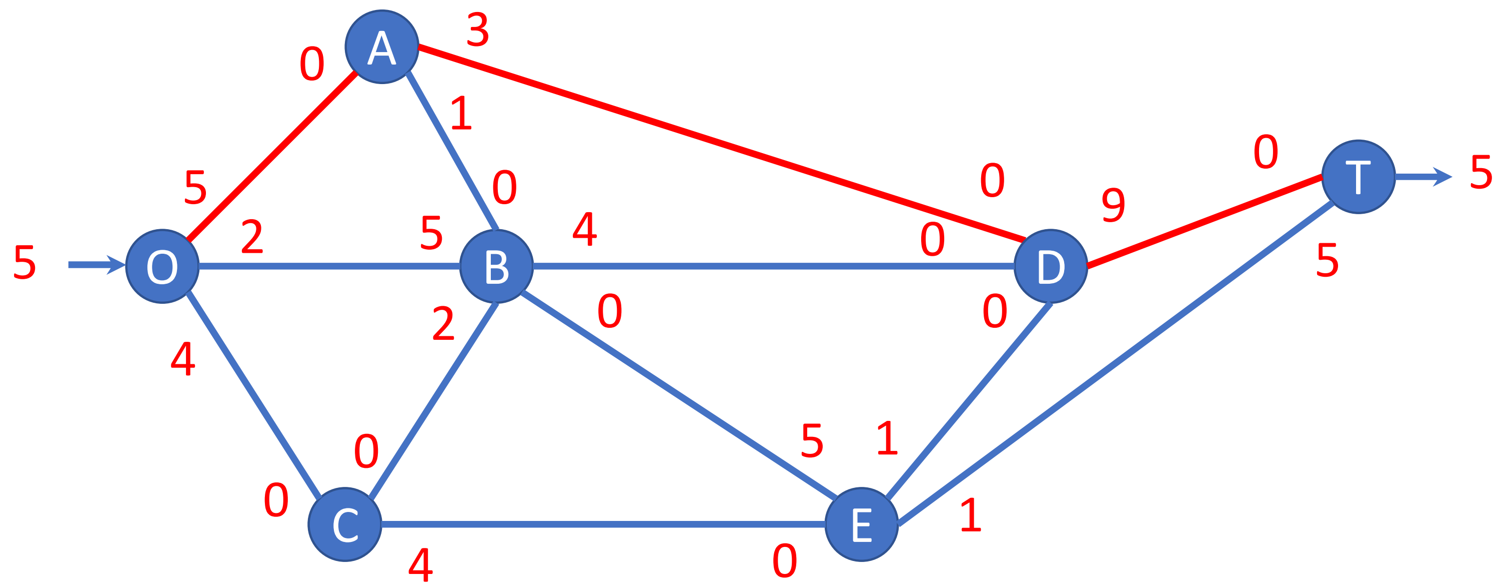
- Iteración 1: trayectoria de aumento es O → B → E → T con capacidad residual igual al min {7, 5, 6} = 5



DEPARTAMENTO DE INGENIERÍA INFORMÁTICA UNIVERSIDAD DE SANTIAGO DE CHILE

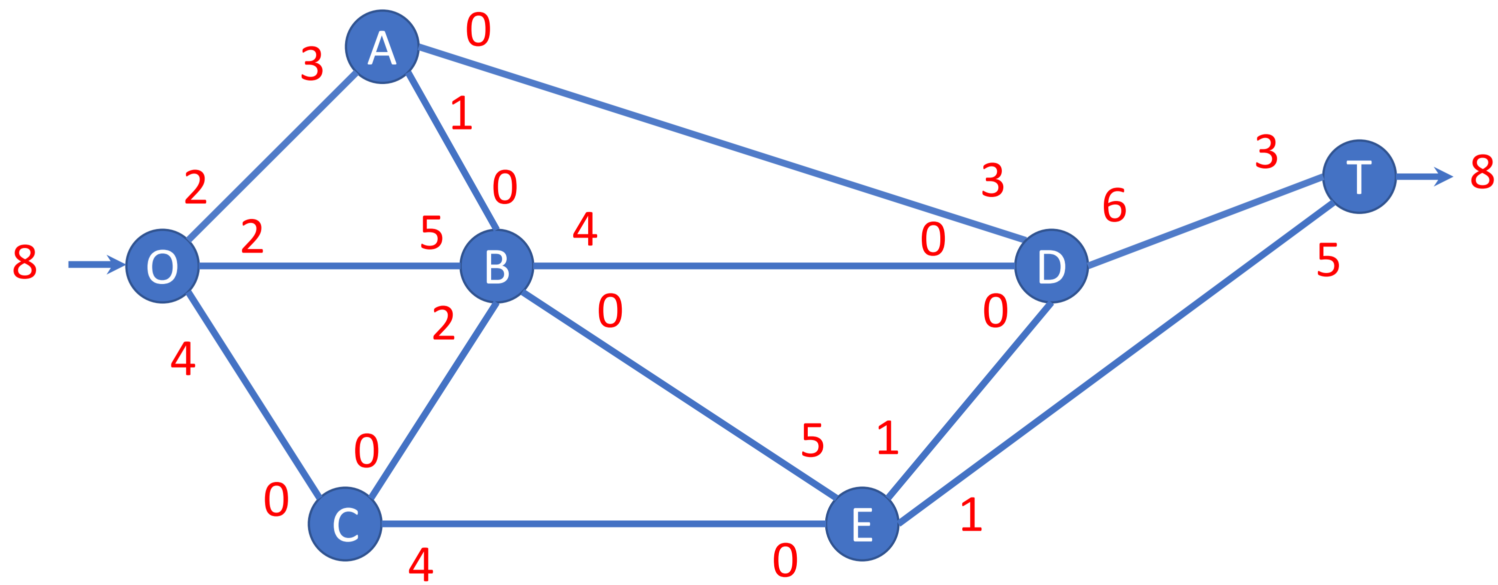
Algoritmo de Ford-Fulkerson

- Iteración 2: trayectoria de aumento es O → A → D → T con capacidad residual igual al min {5, 3, 9} = 3



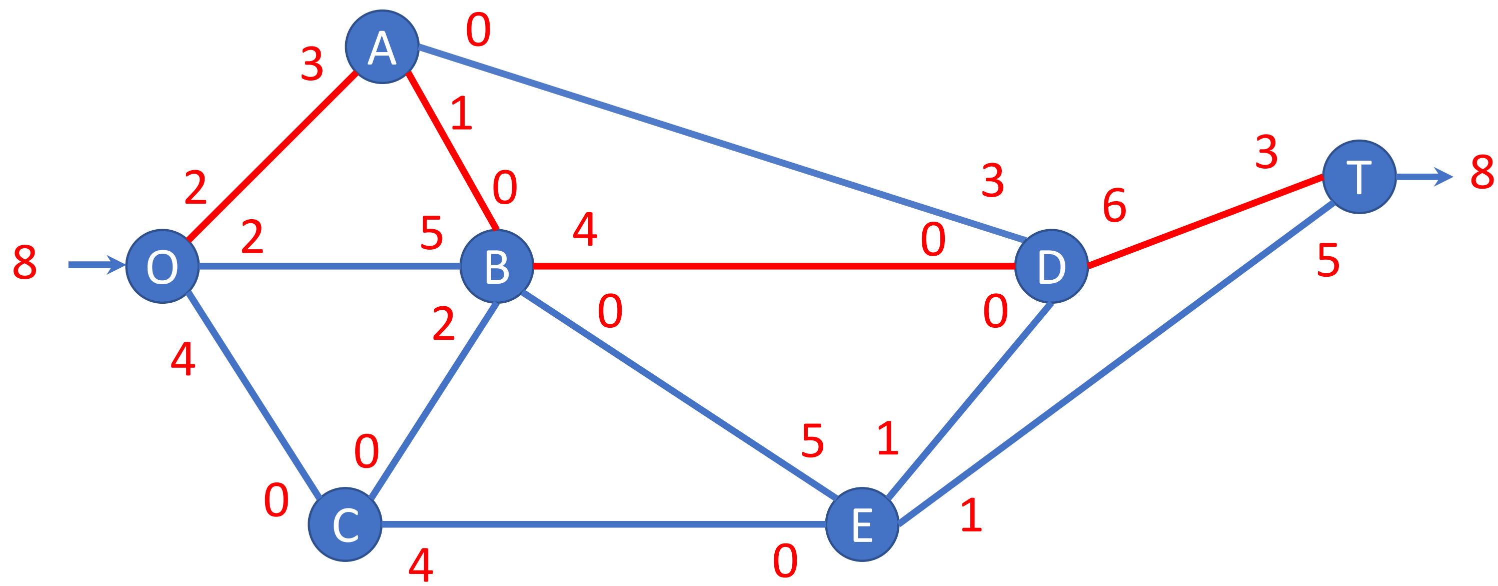
Algoritmo de Ford-Fulkerson

- Iteración 2: trayectoria de aumento es O → A → D → T con capacidad residual igual al min {5, 3, 9} = 3



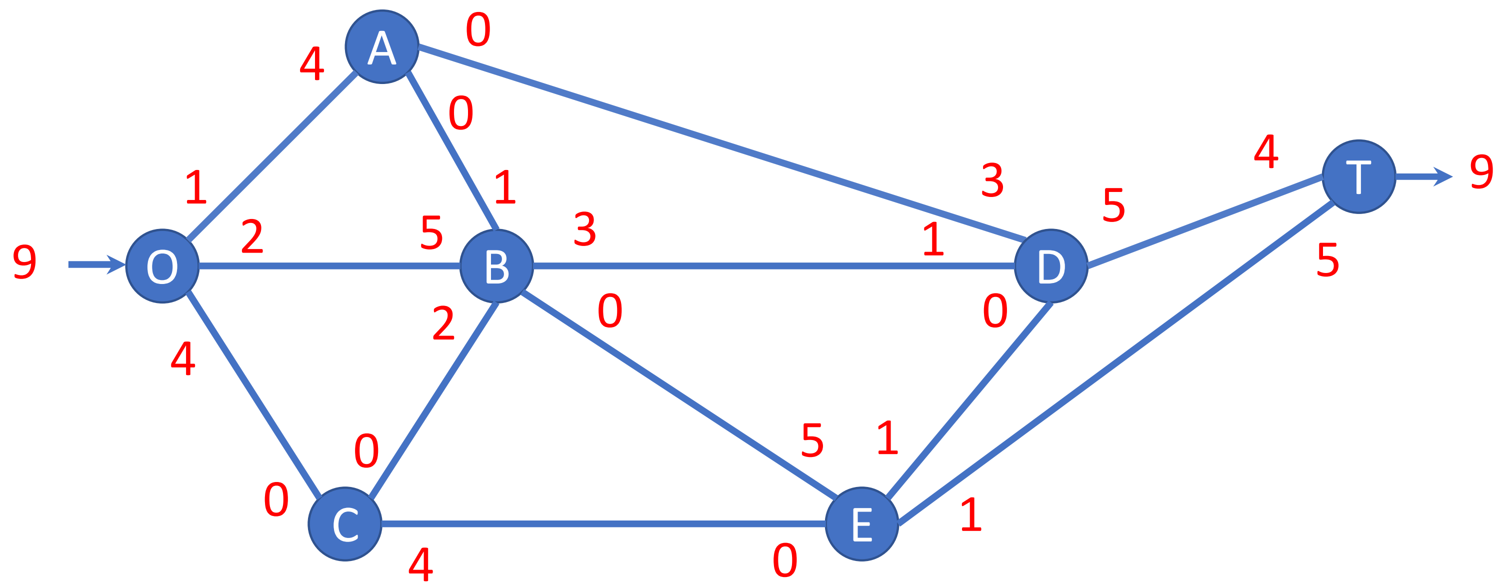
Algoritmo de Ford-Fulkerson

- Iteración 3: trayectoria de aumento es O → A → B → D → T con capacidad residual igual al min {2, 1, 4, 6} = 1



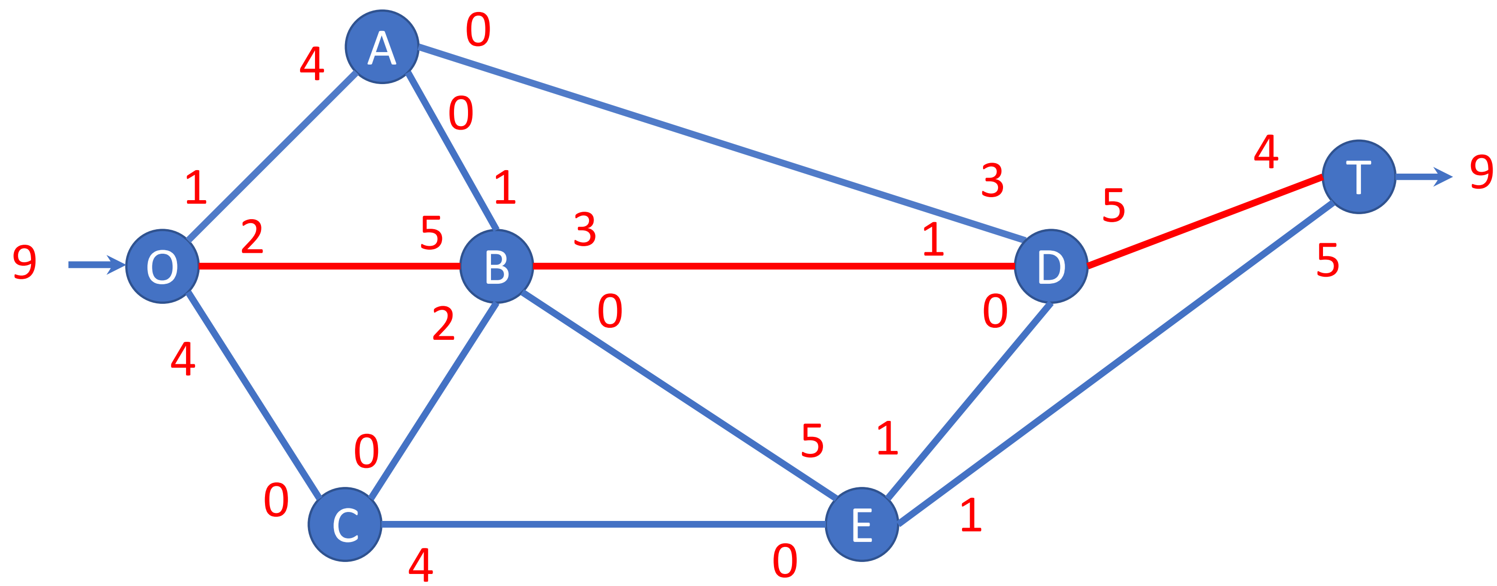
Algoritmo de Ford-Fulkerson

- Iteración 3: trayectoria de aumento es O → A → B → D → T con capacidad residual igual al min {2, 1, 4, 6} = 1



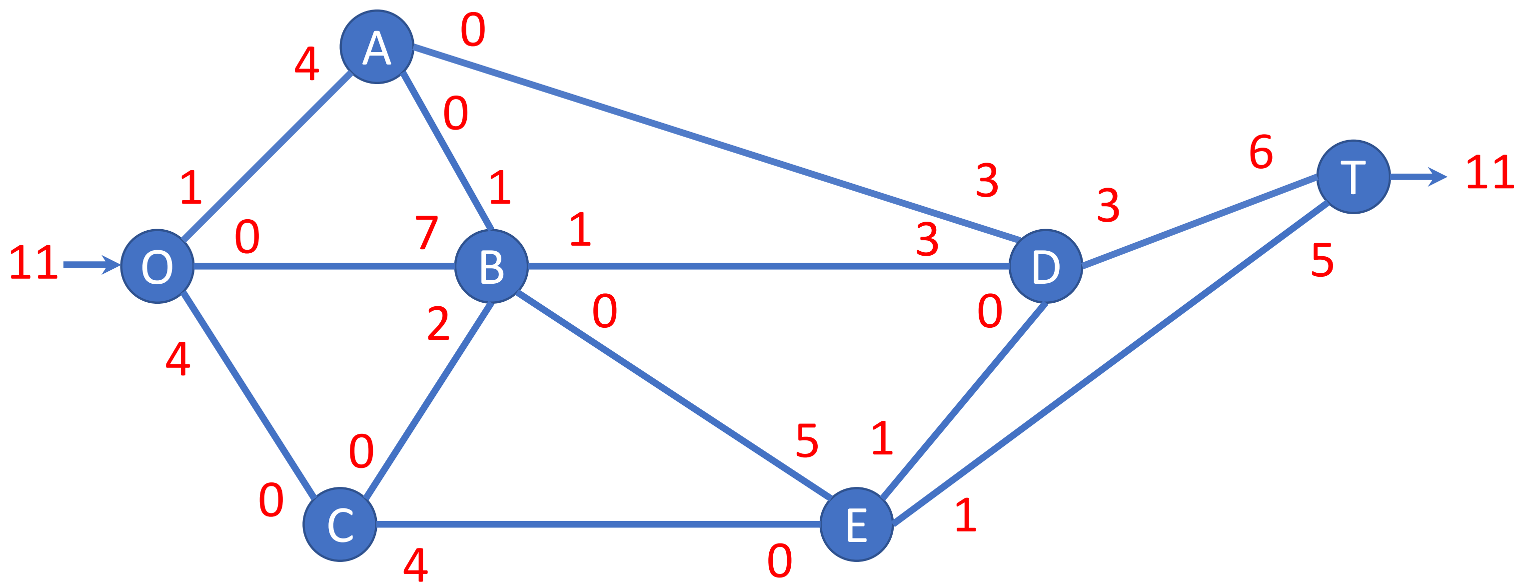
Algoritmo de Ford-Fulkerson

- Iteración 4: trayectoria de aumento es O → B → D → T con capacidad residual igual al min {2, 3, 5} = 2



Algoritmo de Ford-Fulkerson

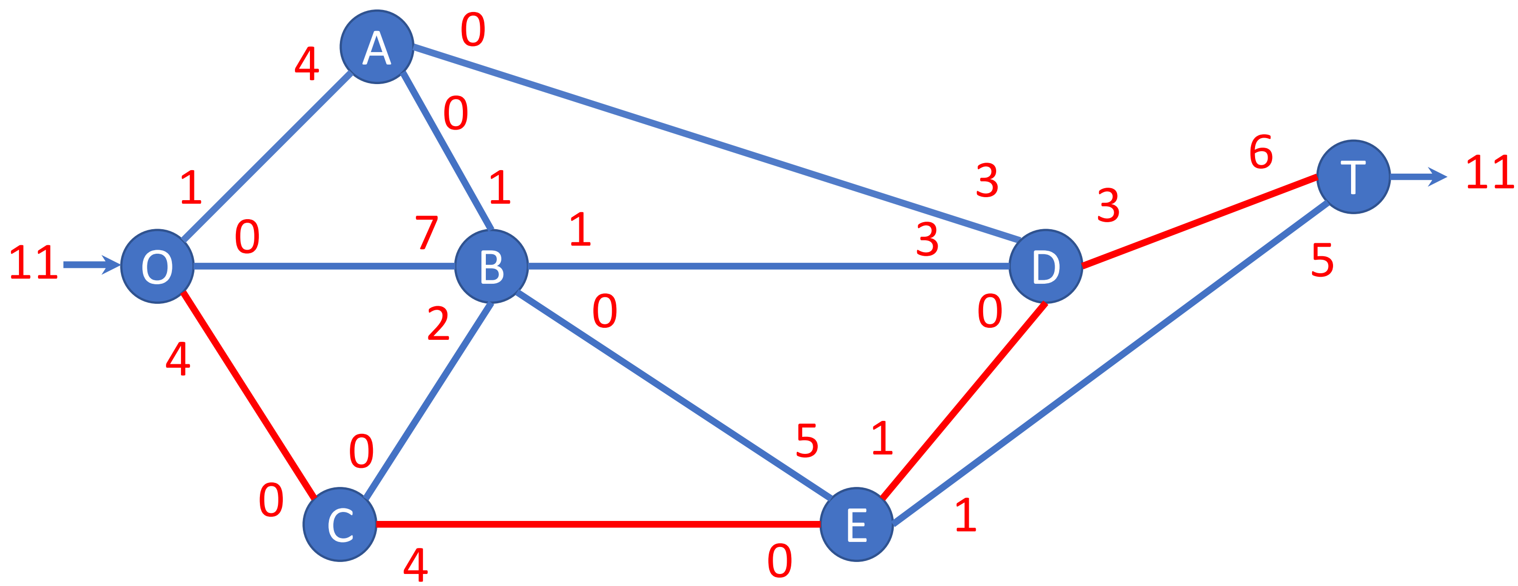
- Iteración 4: trayectoria de aumento es O → B → D → T con capacidad residual igual al min {2, 3, 5} = 2



DEPARTAMENTO DE
INGENIERÍA
INFORMÁTICA
UNIVERSIDAD DE SANTIAGO DE CHILE

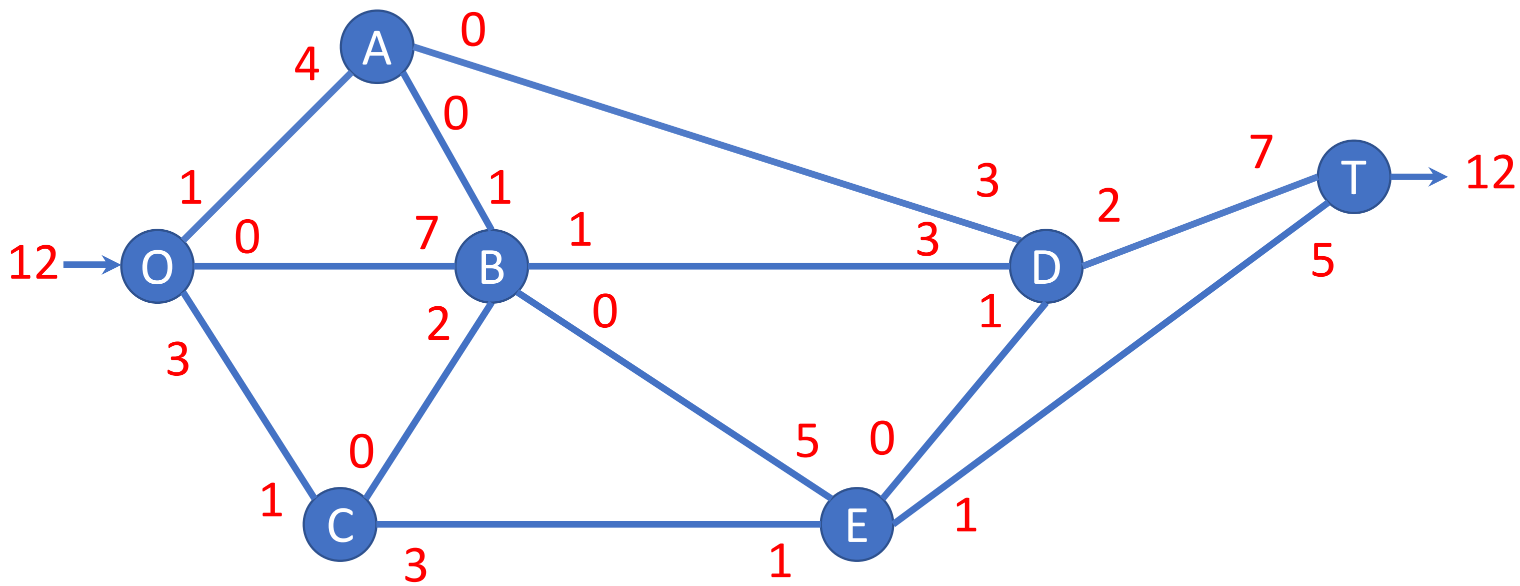
Algoritmo de Ford-Fulkerson

- Iteración 5: trayectoria de aumento es O → C → E → D → T con capacidad residual igual al min {4, 4, 1, 3} = 1



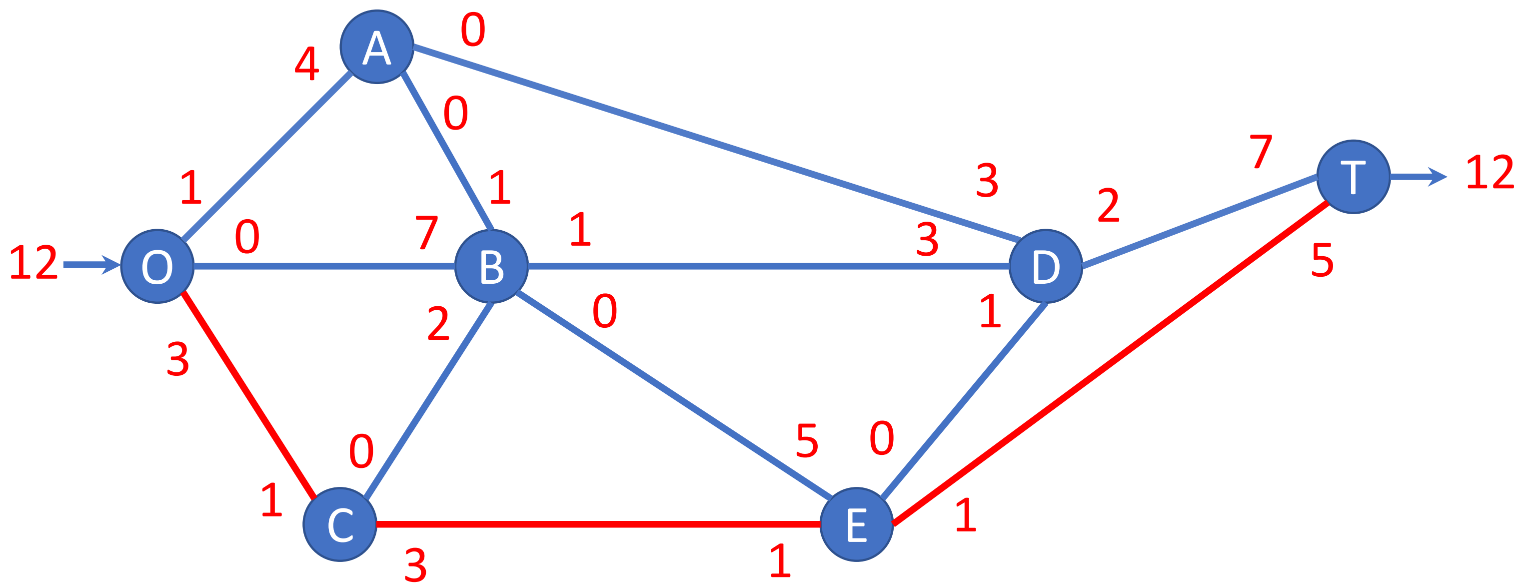
Algoritmo de Ford-Fulkerson

- Iteración 5: trayectoria de aumento es O → C → E → D → T con capacidad residual igual al min {4, 4, 1, 3} = 1



Algoritmo de Ford-Fulkerson

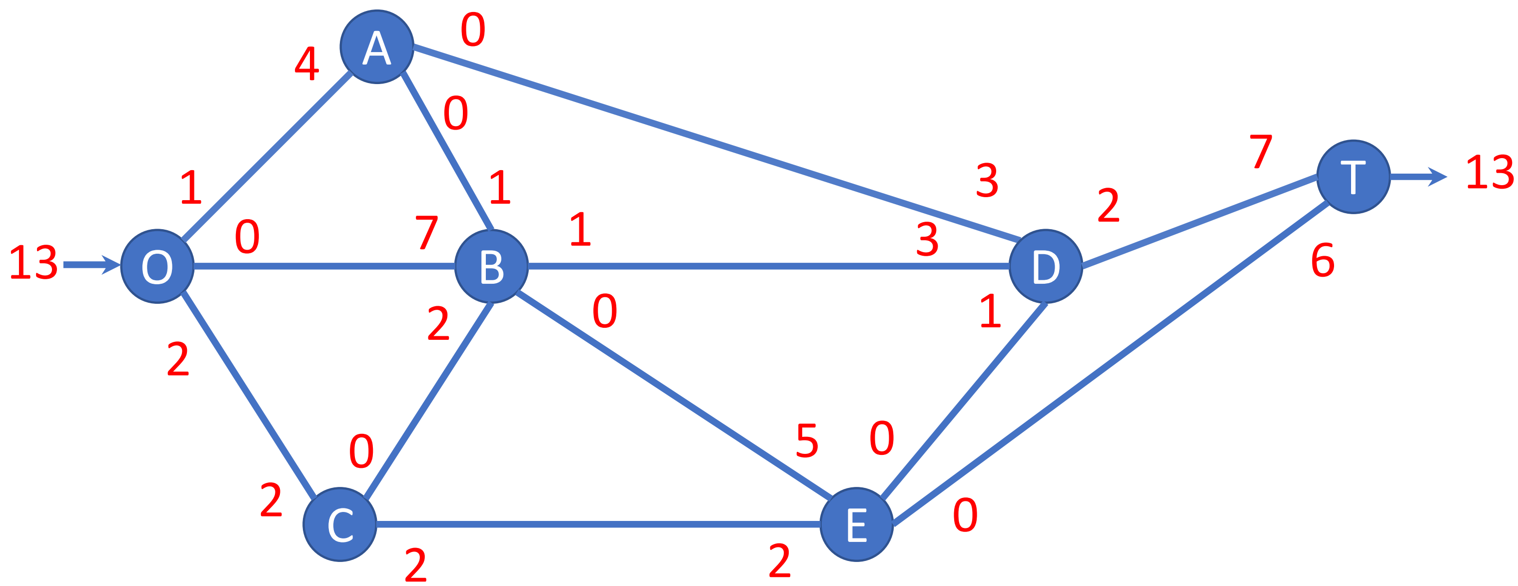
- Iteración 6: trayectoria de aumento es O → C → E → T con capacidad residual igual al min {3, 3, 1} = 1



Algoritmo de Ford-Fulkerson

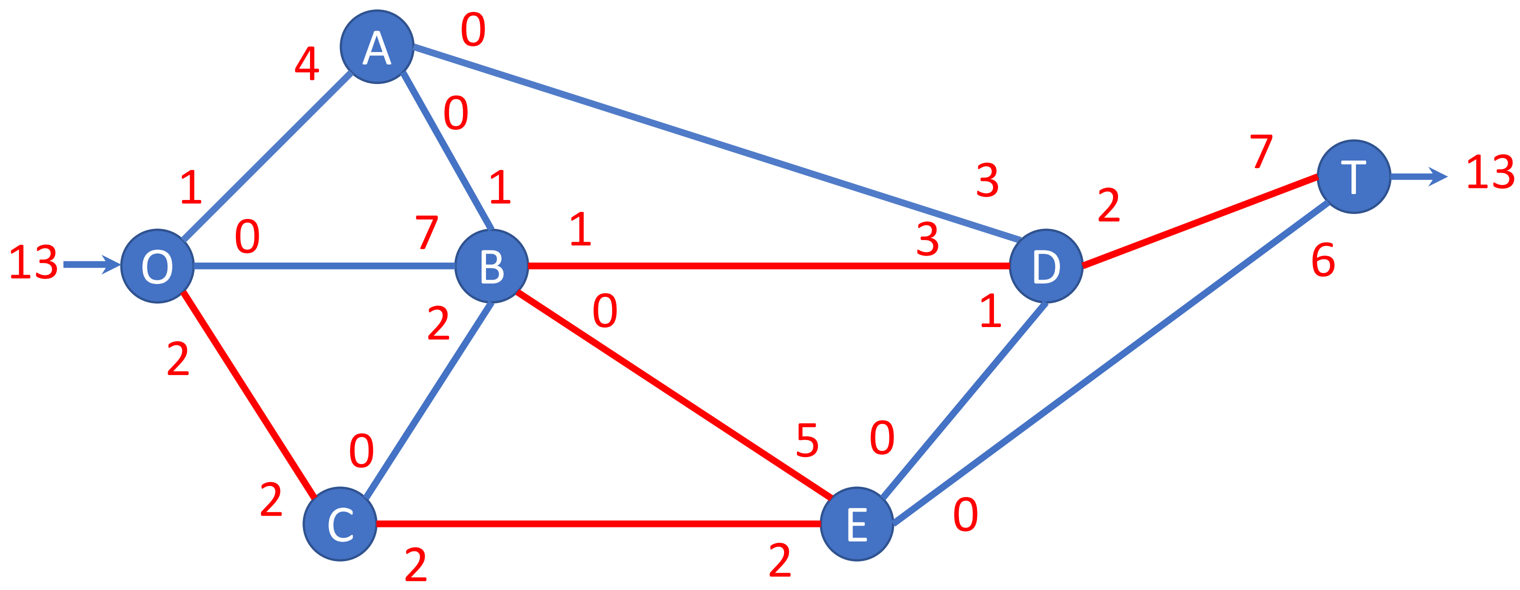
DEPARTAMENTO DE INGENIERÍA INFORMÁTICA
UNIVERSIDAD DE SANTIAGO DE CHILE

- Iteración 6: trayectoria de aumento es O → C → E → T con capacidad residual igual al min {3, 3, 1} = 1



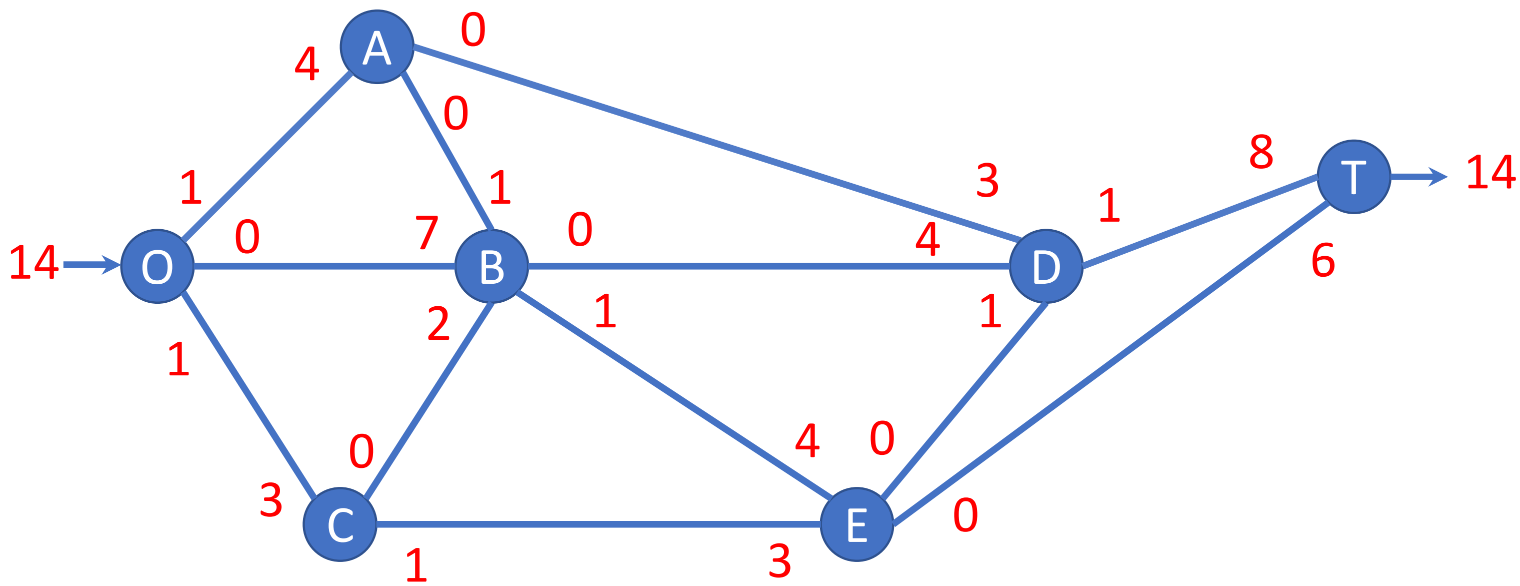
Algoritmo de Ford-Fulkerson

- Iteración 7: trayectoria de aumento es O → C → E → B → D → T con capacidad residual igual al min {2, 2, 5, 1, 2} = 1



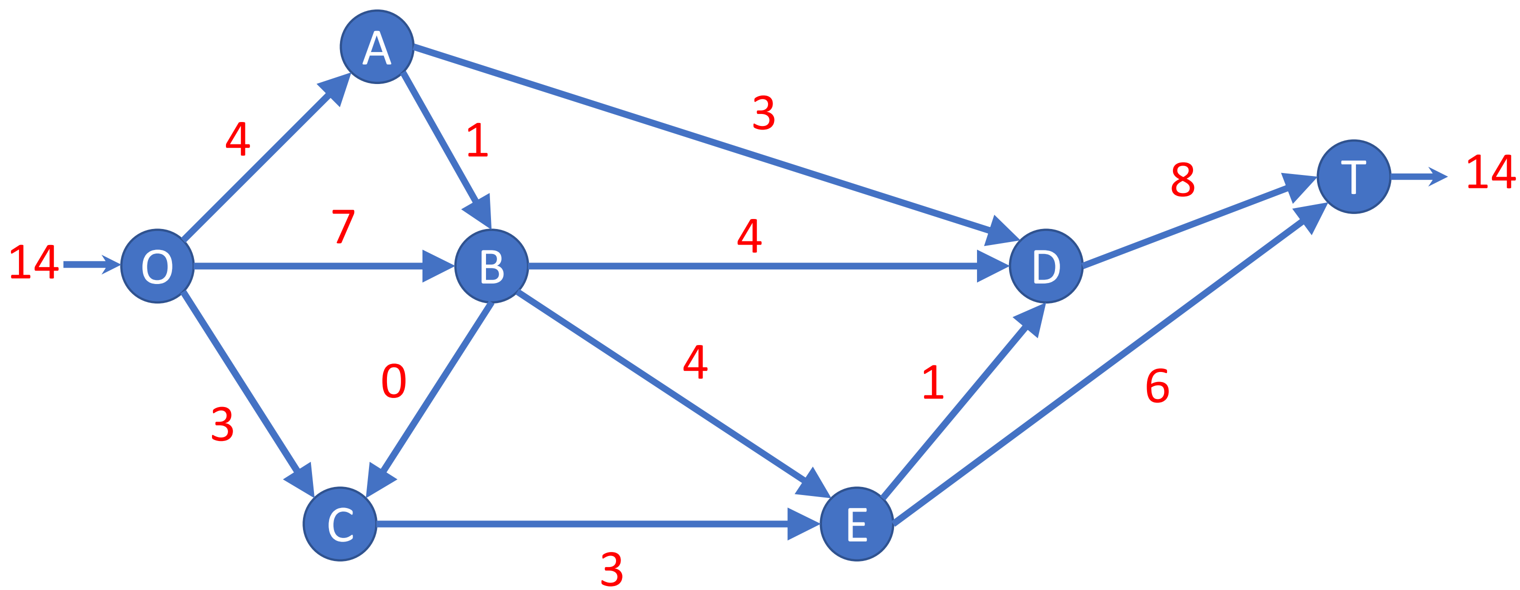
Algoritmo de Ford-Fulkerson

- Iteración 7: trayectoria de aumento es O → C → E → B → D → T con capacidad residual igual al min {2, 2, 5, 1, 2} = 1



Algoritmo de Ford-Fulkerson

- Solución óptima



DEPARTAMENTO DE INGENIERÍA INFORMÁTICA UNIVERSIDAD DE SANTIAGO DE CHILE

Tarea

- Implementar la búsqueda en amplitud y la búsqueda en profundidad

CONSULTAS