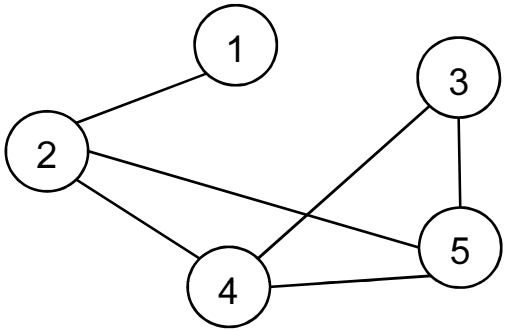
UNIVERSIDAD PONTIFICIA
ICAI ICADE
COMILLAS
MADRID
ESCUELA TÉCNICA SUPERIOR DE INGENIERÍA
DEPARTAMENTO DE ORGANIZACIÓN INDUSTRIAL

Teoría de grafos y optimización en redes

José María Ferrer Caja
Universidad Pontificia Comillas

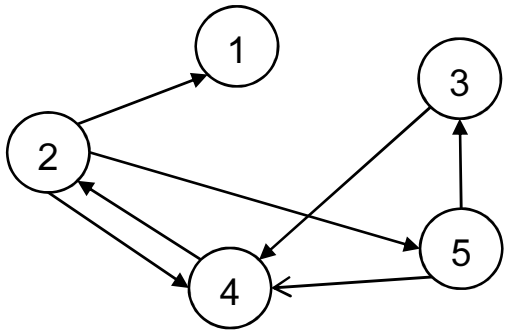
Definiciones básicas

- Grafo: Conjunto de nodos (o vértices) unidos por aristas → $G = (V,E)$
  - ✓ Ejemplo



$V = \{1, 2, 3, 4, 5\}$
$E = \{\{1,2\}, \{2,4\}, \{2,5\}, \{3,4\}, \{3,5\}, \{4,5\}\}$

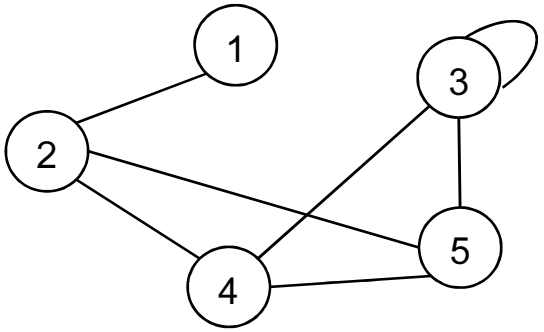
- Grafo dirigido o red: Conjunto de nodos (o vértices) unidos por arcos $R=(V,A)$
  - ✓ Ejemplo



$V = \{1, 2, 3, 4, 5\}$
$A = \{(2,1), (2,4), (2,5), (3,4), (4,2), (5,3), (5,4)\}$

Definiciones básicas

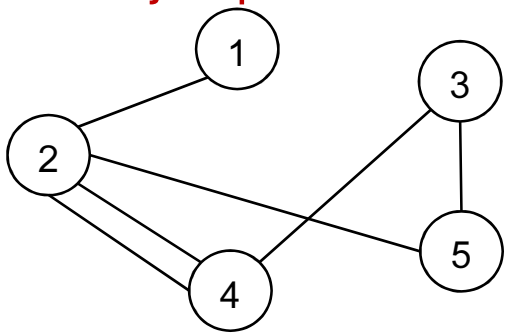
- Pseudografo: "Grafo" en el que se permiten bucles → aristas que unen un nodo consigo mismo
  - ✓ Ejemplo



$V = \{1, 2, 3, 4, 5\}$

$E = \{\{1,2\}, \{2,4\}, \{2,5\}, \{3,3\}, \{3,4\}, \{3,5\}, \{4,5\}\}$

- Multigrafo: "Grafo" en el que se permite más de una arista entre cada par de nodos
  - ✓ Ejemplo



$V = \{1, 2, 3, 4, 5\}$

$E = \{\{1,2\}, \{2,4\}, \{2,4\}, \{2,5\}, \{3,4\}, \{3,5\}, \{4,5\}\}$

Representación matricial

- Matriz de adyacencia de un grafo (o red) → *A*
  - ✓ Cada fila asociada a un nodo
  - ✓ Cada columna asociada a un nodo

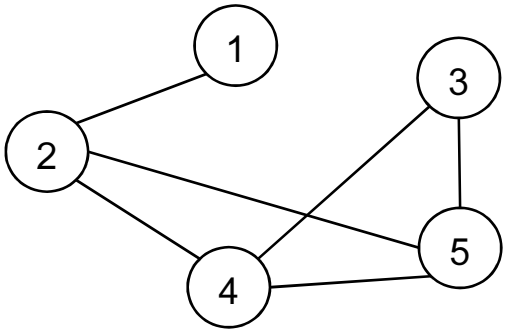
$$a_{ij} = \begin{cases} 1 & \text{si existe arista (arco) del nodo } i \text{ al nodo } j \\ 0 & \text{en otro caso} \end{cases}$$

  - ✓ Simétrica para el caso de grafo
- Matriz de incidencia de un grafo (o red) → *B*
  - ✓ Cada fila asociada a un nodo
  - ✓ Cada columna asociada a una arista
  - ✓ Grafo $b_{ij} = \begin{cases} 1 & \text{si el nodo } i \text{ pertenece a la arista } j \\ 0 & \text{en otro caso} \end{cases}$
  - ✓ Red (totalmente unimodular)

$$b_{ij} = \begin{cases} -1 & \text{si el nodo } i \text{ es el comienzo del arco } j \\ 1 & \text{si el nodo } i \text{ es el final del arco } j \\ 0 & \text{en otro caso} \end{cases}$$

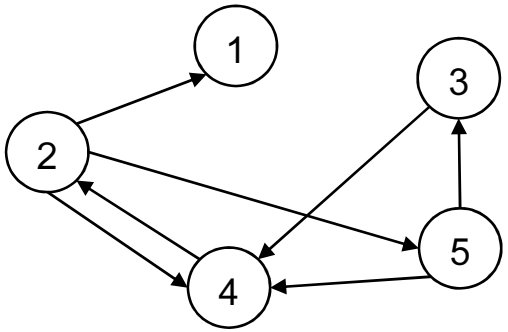
Representación matricial

❑ Ejemplo



$$A = \begin{pmatrix} 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 \\ 0 & 1 & 1 & 0 & 1 \\ 0 & 1 & 1 & 1 & 0 \end{pmatrix} \quad B = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 & 1 & 1 \end{pmatrix}$$

❑ Ejemplo

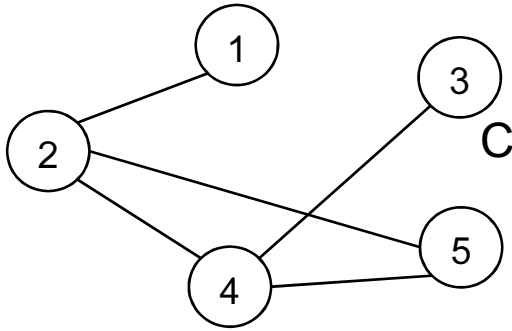


$$A = \begin{pmatrix} 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 \end{pmatrix} \quad B = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ -1 & -1 & -1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & -1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 & -1 & 0 & 1 \\ 0 & 0 & 1 & 0 & 0 & -1 & -1 \end{pmatrix}$$

Conexión

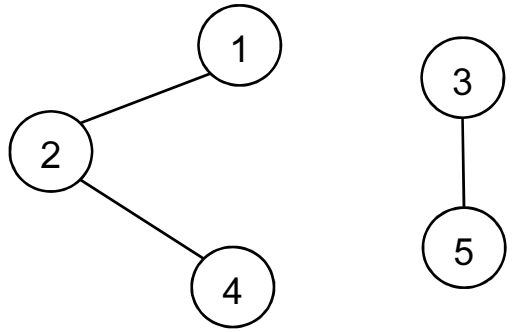
Conexión en un grafo

- ✓ Una cadena es una secuencia de aristas de forma que el nodo final de cada arista coincide con el nodo inicial de la siguiente. Cada cadena conecta dos nodos
- ✓ Si dos nodos están conectados, se puede encontrar una cadena entre ellos que no repita nodos
- ✓ Un ciclo es una cadena que conecta un nodo consigo mismo
- ✓ Un grafo es conexo si cada par de nodos está conectado por medio de alguna cadena
- ✓ Ejemplos



Cadena: {2,5}, {5,4}, {4,3}
Ciclo: {2,5}, {5,4}, {4,2}

Grafo conexo



Grafo no conexo

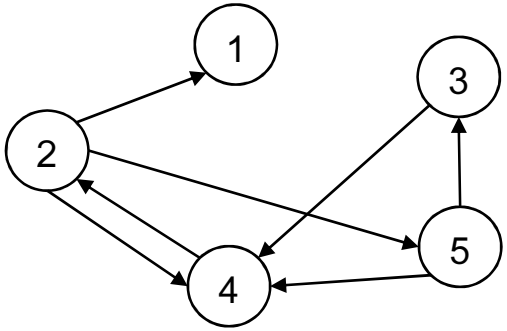
Conexión

Conexión en una red

- ✓ Cadena y ciclo: Igual que en un grafo. No influye la orientación de los arcos
- ✓ Un camino es una secuencia de arcos de forma que el nodo final de cada arco coincide con el nodo inicial del siguiente. Cadena en la que todos los arcos están orientados en el mismo sentido
- ✓ Un circuito es un ciclo y un camino al mismo tiempo
- ✓ Una red es conexa si cada par de nodos está conectado por medio de alguna cadena
- ✓ Una red es fuertemente conexa si cada par de nodos está conectado por medio de algún camino
- ✓ Toda red fuertemente conexa es conexa

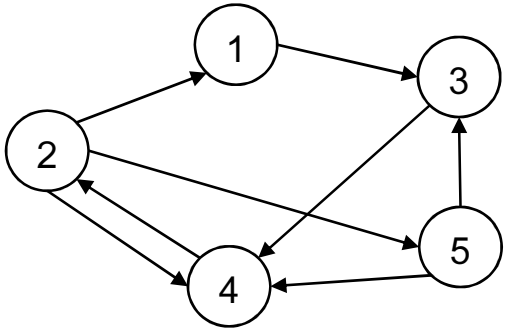
Conexión

- ✓ Ejemplos



Cadena: {2,5}, {5,4}, {4,3}
Ciclo: {2,5}, {5,4}, {4,2}
Camino: (2,5), (5,3), (3,4)
Circuito: (5,4), (4,2), (2,5)

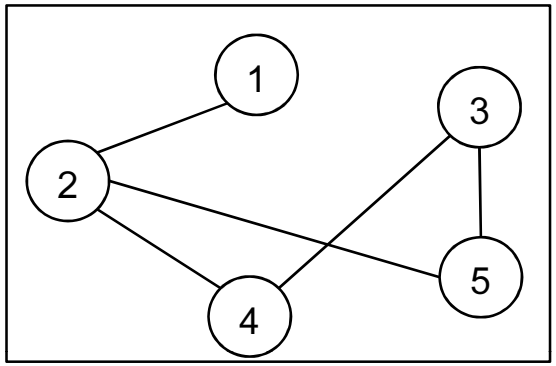
Red conexa
Red no fuertemente conexa, ya que no existe camino que conecte el nodo 1 con el nodo 2



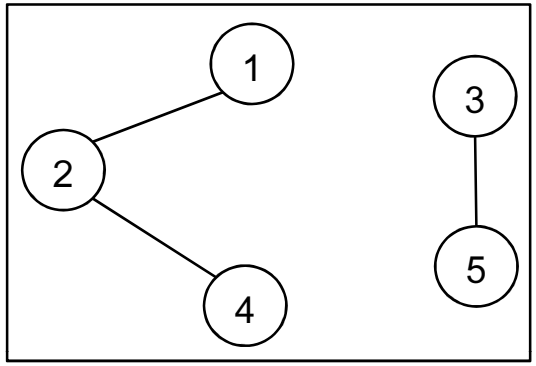
Red fuertemente conexa

Árboles

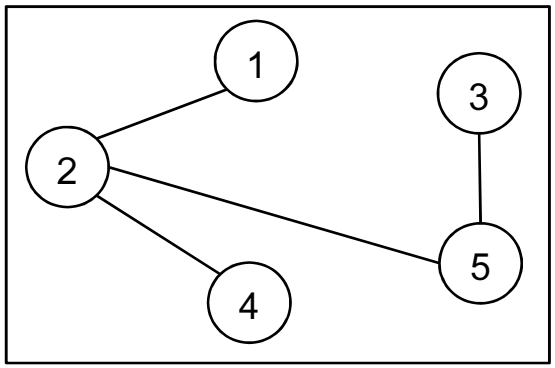
- Un árbol es un grafo conexo y sin ciclos
  - ✓ Ejemplos



No es árbol, pues existen ciclos



No es árbol, pues no es conexo

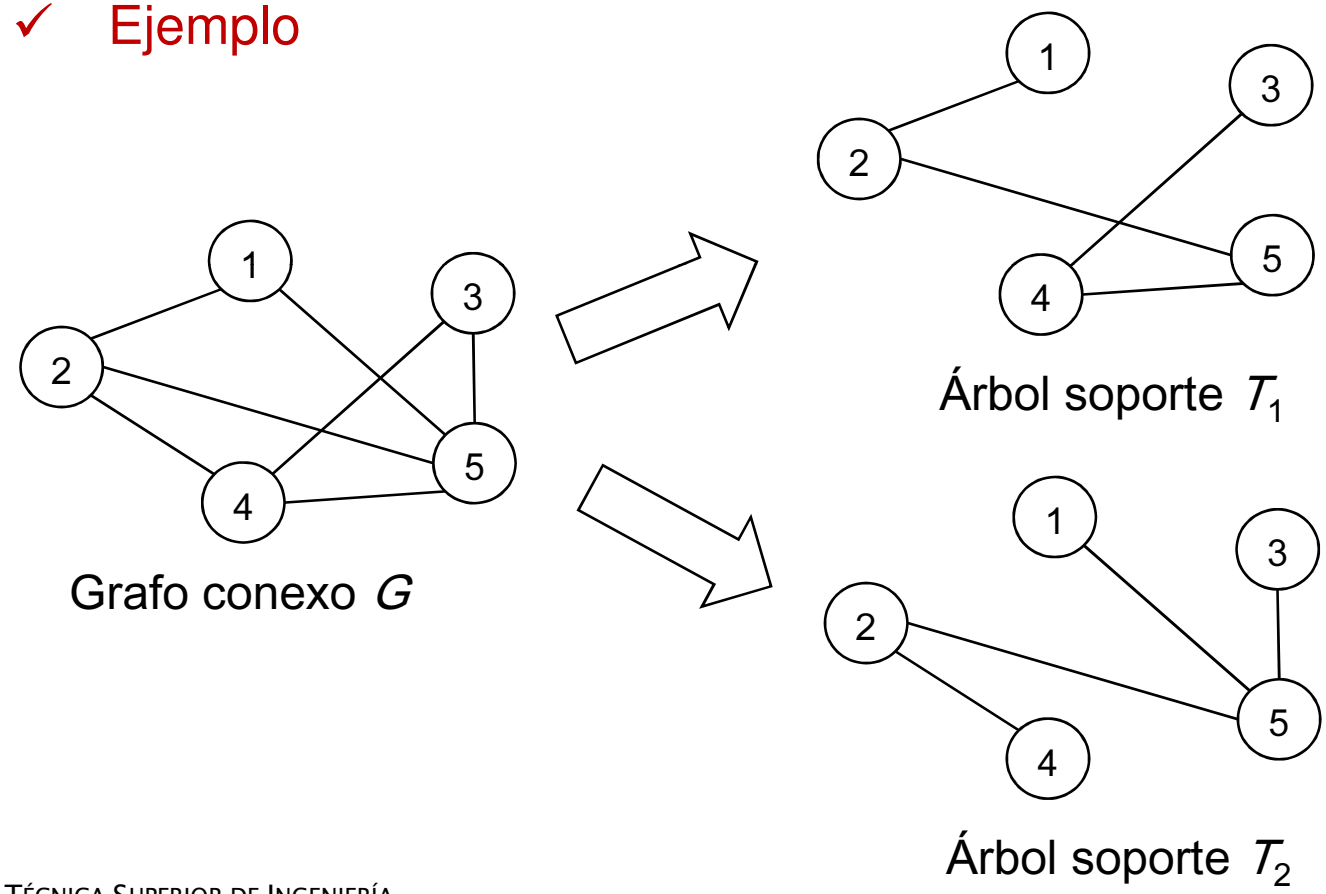


Árbol

- En un árbol se cumple card(E) = card(V) - 1

Árbol soporte

- Dado un grafo conexo $G = (V,E)$, un árbol soporte (árbol generador, árbol de expansión, árbol de extensión) es un árbol $T = (V,E')$ donde $E' \subseteq E$
  - ✓ Ejemplo



Árbol soporte de mínimo peso

- Planteamiento: Dado un grafo conexo $G = (V,E)$ con pesos asociados a las aristas, el objetivo es obtener un árbol soporte $T = (V,E')$ en el que la suma de los pesos de las aristas elegidas sea mínima
- Algoritmo de Prim

PASO 1. INICIALIZACIÓN

Elegir un NODO cualquiera $i \in V$. Hacer $C_1 = \{i\}, \bar{C}_1 = V - \{i\}$

Sea $k = 2, E' = \varnothing$

PASO 2. SELECCIÓN DE LA ARISTA DE MENOR PESO QUE UNA UN NODO DE LOS CONECTADOS CON UN NODO NO CONECTADO

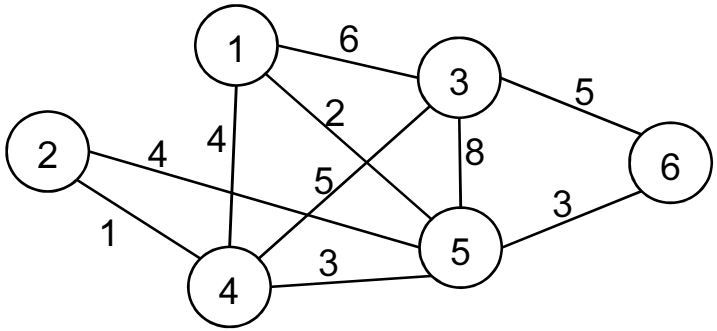
Seleccionar la arista $e_{k-1} = \{i, j\}$ con $i \in C_k, j \in \bar{C}_k$ de peso mínimo entre todas las posibles. Hacer $C_k = C_{k-1} \cup \{j\}, \bar{C}_k = \bar{C}_{k-1} - \{j\}$ y $E' = E' \cup \{e_{k-1}\}$

Si $k = car(V)$ PARAR.

Si no, hacer $k = k + 1$ y volver al PASO 2.

- ✓ Algoritmo tipo *greedy* de complejidad polinomial

Árbol soporte de mínimo peso. Ejemplo

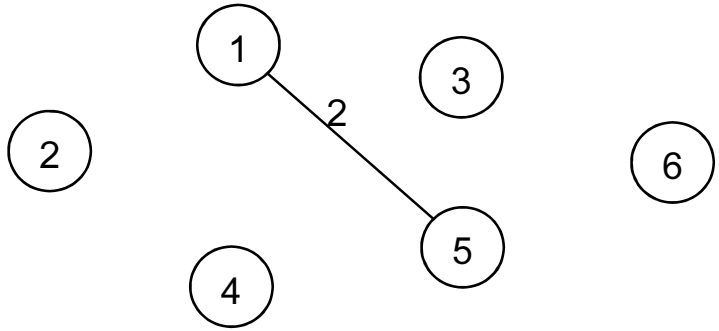


PASO 1

Elegimos por ejemplo $i=1: C_1=\{1\}, \bar{C}_1=\{2,3,4,5,6\}. k=2, E'=\varnothing$

PASO 2

Elegimos de las aristas que unen nodos de C_1 con nodos de $\bar{C}_1$, la de menor peso: $e_1=\{1,5\}$ con peso 2. $C_2=\{1,5\}, \bar{C}_2=\{2,3,4,6\}$. $E'=\{\{1,5\}\}$



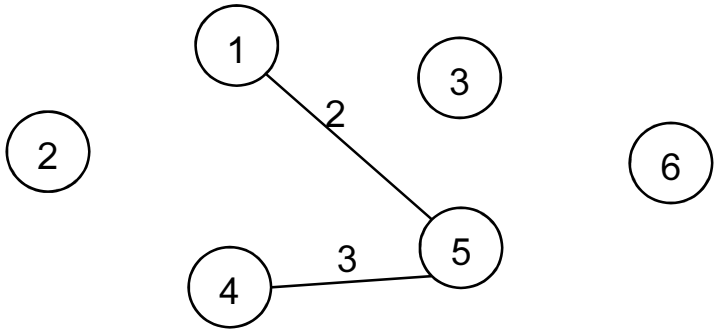
$k<6 \Rightarrow k=3$, volver al PASO 2

Árbol soporte de mínimo peso. Ejemplo

PASO 2

Elegimos por ejemplo $e_2 = \{4,5\}$ con peso 3. $C_3 = \{1,4,5\}, \bar{C}_3 = \{2,3,6\}$

$E' = \{\{1,5\},\{4,5\}\}$

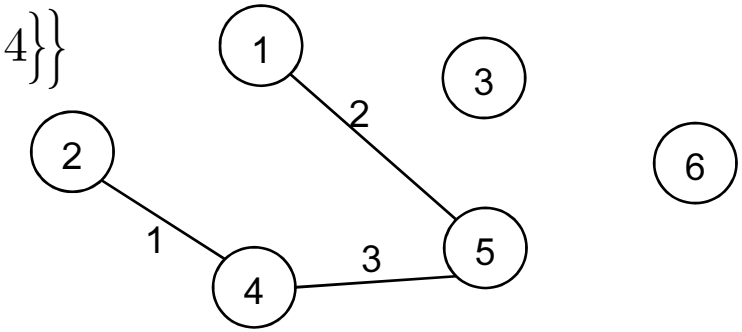


$k < 6 \Rightarrow k = 4$, volver al PASO 2

PASO 2

Elegimos $e_3 = \{2,4\}$ con peso 1. $C_4 = \{1,2,4,5\}, \bar{C}_4 = \{3,6\}$

$E' = \{\{1,5\},\{4,5\},\{2,4\}\}$



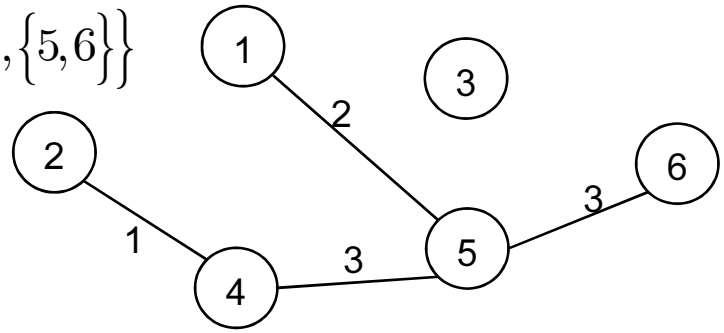
$k < 6 \Rightarrow k = 5$, volver al PASO 2

Árbol soporte de mínimo peso. Ejemplo

PASO 2

Elegimos $e_4 = \{5,6\}$ con peso 3. $C_5 = \{1,2,4,5,6\}, \bar{C}_5 = \{3\}$

$E' = \{\{1,5\},\{4,5\},\{2,4\},\{5,6\}\}$

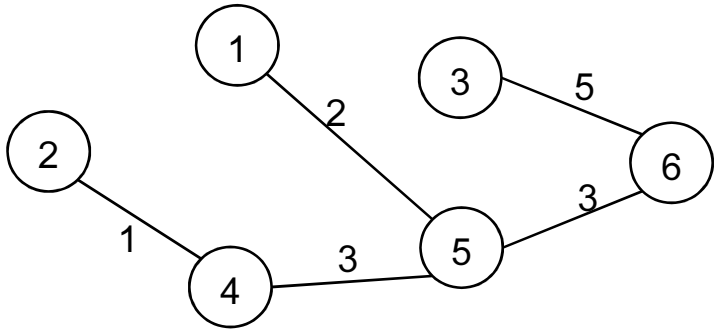


$k < 6 \Rightarrow k = 6$, volver al PASO 2

PASO 2

Elegimos por ejemplo $e_5 = \{3,6\}$ con peso 5. $C_6 = \{1,2,3,4,5,6\}, \bar{C}_6 = \varnothing$

$E' = \{\{1,5\},\{4,5\},\{2,4\},\{5,6\},\{3,6\}\}$.$k = 6 \Rightarrow$ Parar



Peso total=14

Problema del camino mínimo

- Planteamiento: Dada una red $R = (V,A)$ con distancias d_{kj} asociadas a los arcos, el objetivo es obtener los caminos más cortos de un nodo origen a todos los demás
  - ✓ Las distancias pueden ser también costes, tiempos, etc.
  - ✓ Si no existe arco del nodo *k* al nodo *j* se asigna $d_{kj} = \infty$
  - ✓ Existen otros planteamientos similares: obtener el camino mínimo de un nodo origen a un nodo destino, obtener los caminos mínimos de todos a todos los nodos, etc.
  - ✓ En el caso de un grafo, se considera cada arista como un par de arcos de sentidos opuestos
- Ecuaciones de Bellman

  Sea 1 el nodo origen, y u_j la longitud del camino mínimo del origen al nodo *j*

$$u_1 = 0$$

$$u_j = \min_{k \neq j} \left\{ u_j, u_k + d_{kj} \right\} \qquad j = 2, \ldots, n$$

Algoritmo de Dijkstra

- ✓ Es necesario que todas las distancias sean positivas
- ✓ Su complejidad es polinomial
- ✓ En cada etapa hay dos conjuntos de nodos: *P* formado por nodos permanentes y *T* por nodos transitorios
- ✓ Se denota *pred*(*j*) al nodo predecesor de *j* en el camino del origen a *j*

❑ Algoritmo

PASO 0. INICIALIZACIÓN

$u_1 = 0,\ u_j = d_{1j}, j = 2,\ldots,n.\ P = \{1\},\ T = \{2,\ldots,n\},\ pred(j) = 1, j = 2,\ldots,n$

PASO 1. HACER PERMANENTE EL NODO CON LONGITUD MÍNIMA

Elegir $k \in T$ tal que $u_k = \min_{j \in T}\{u_j\}$. Hacer $P = P \cup \{k\},\ T = T - \{k\}$

Si $T = \varnothing$ PARAR

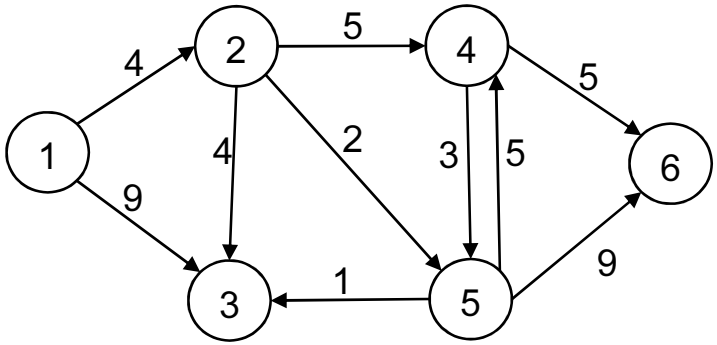
PASO 2. REVISIÓN DE LOS NODOS TRANSITORIOS

Para cada $j \in T$ calcular $u_j = \min\{u_j, u_k + d_{kj}\}$.

Si se modifica u_j hacer $pred(j) = k$

Volver al PASO 1

Algoritmo de Dijkstra. Ejemplo



PASO 0

$u_1 = 0,\ u_2 = 4, u_3 = 9, u_4 = u_5 = u_6 = \infty.\ P = \{1\},\ T = \{2,3,4,5,6\},\ pred(j) = 1, j = 2,3,4,5,6$

PASO 1

$k = 2, u_2 = 4.\ P = \{1,2\},\ T = \{3,4,5,6\}$

PASO 2

$$u_3 = \min\{u_3, u_2 + d_{23}\} = \min\{9, 4+4\} = 8 \Rightarrow pred(3) = 2$$

$$u_4 = \min\{u_4, u_2 + d_{24}\} = \min\{\infty, 4+5\} = 9 \Rightarrow pred(4) = 2$$

$$u_5 = \min\{u_5, u_2 + d_{25}\} = \min\{\infty, 4+2\} = 6 \Rightarrow pred(5) = 2$$

$$u_6 = \min\{u_6, u_2 + d_{26}\} = \min\{\infty, 4+\infty\} = \infty$$

Algoritmo de Dijkstra. Ejemplo

PASO 1

$k = 5, u_5 = 6.\ P = \{1,2,5\},\ T = \{3,4,6\}$

PASO 2

$u_3 = \min\{u_3, u_5 + d_{53}\} = \min\{8, 6+1\} = 7,\ pred(3) = 5$

$u_4 = \min\{u_4, u_5 + d_{54}\} = \min\{9, 6+5\} = 9$

$u_6 = \min\{u_6, u_5 + d_{56}\} = \min\{\infty, 6+9\} = 15,\ pred(6) = 5$

PASO 1

$k = 3, u_3 = 7.\ P = \{1,2,5,3\},\ T = \{4,6\}$

PASO 2

$u_4 = \min\{u_4, u_3 + d_{34}\} = \min\{9, 7+\infty\} = 9$

$u_6 = \min\{u_6, u_3 + d_{36}\} = \min\{15, 7+\infty\} = 15$

PASO 1

$k = 4, u_4 = 9.\ P = \{1,2,5,3,4\},\ T = \{6\}$

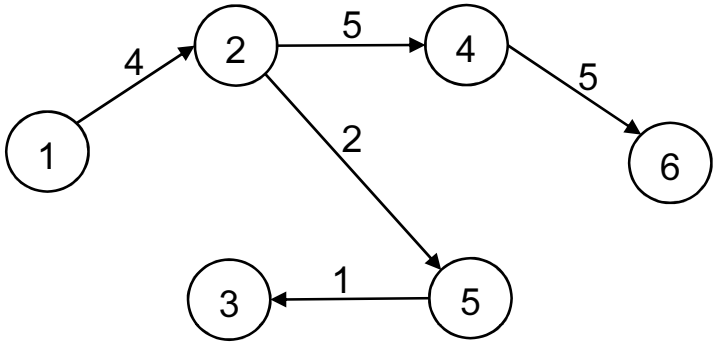
PASO 2

$u_6 = \min\{u_6, u_4 + d_{46}\} = \min\{15, 9+5\} = 14,\ pred(6) = 4$

Algoritmo de Dijkstra . Ejemplo

PASO 1

$k = 6, u_6 = 14.\ P = V, T = \varnothing \Rightarrow$ Parar



Algoritmo de Bellman-Ford

- ✓ Apropiado cuando existen distancias negativas
- ✓ Su complejidad es mayor que la de Dijkstra
- ✓ Si existen circuitos de longitud negativa, no hay solución
- ✓ $u_j^m \rightarrow$ Longitud del camino mínimo del nodo origen al nodo *j* usando a lo sumo *m* arcos
- ✓ No puede fijarse ningún nodo como permanente hasta el final

❑ Algoritmo

PASO 1. INICIALIZACIÓN

$u_1^1 = 0,\ u_j^1 = d_{1j}, j = 2,\ldots,n.\ m = 1$

PASO 2. CALCULAR LOS CAMINOS MÍNIMOS CON *m*+1 ARCOS

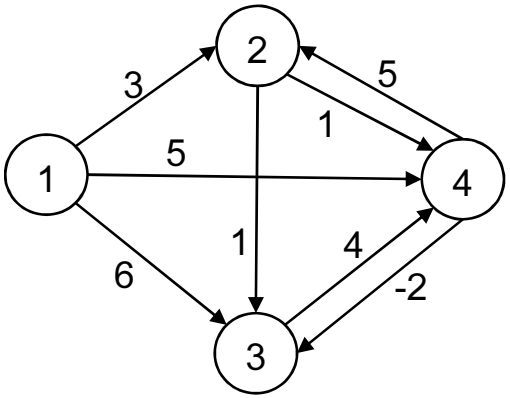
Calcular, para cada $j = 2,\ldots,n,\ u_j^{m+1} = \min\left\{u_j^m, \min_{k\neq j}\left\{u_k^m + d_{kj}\right\}\right\}$

Si $u_j^{m+1} = u_j^m\ \forall j$ o $m+1 = n-1$ PARAR.

Si no, hacer $m = m+1$ y volver al PASO 2

- ✓ Pueden almacenarse los predecesores para construir el árbol final

Algoritmo de Bellman-Ford. Ejemplo



Hay una distancia negativa
No hay circuitos de longitud negativa

PASO 1

$u_1^1 = 0,\ u_2^1 = 3, u_3^1 = 6, u_4^1 = 5.\ m = 1$

PASO 2

$$u_2^2 = \min\left\{u_2^1, \min\left\{u_3^1 + d_{32}, u_4^1 + d_{42}\right\}\right\} = \min\left\{3, 6+\infty, 5+5\right\} = 3$$

$$u_3^2 = \min\left\{u_3^1, \min\left\{u_2^1 + d_{23}, u_4^1 + d_{43}\right\}\right\} = \min\left\{6, 3+1, 5-2\right\} = 3$$

$$u_4^2 = \min\left\{u_4^1, \min\left\{u_2^1 + d_{24}, u_3^1 + d_{34}\right\}\right\} = \min\left\{5, 3+1, 6+4\right\} = 4$$

Como $u_3^2 \neq u_3^1$ y $m+1 = 2 \neq 3 = n-1$, hay que seguir: $m = 2$ y volver al PASO 2

Algoritmo de Bellman-Ford. Ejemplo

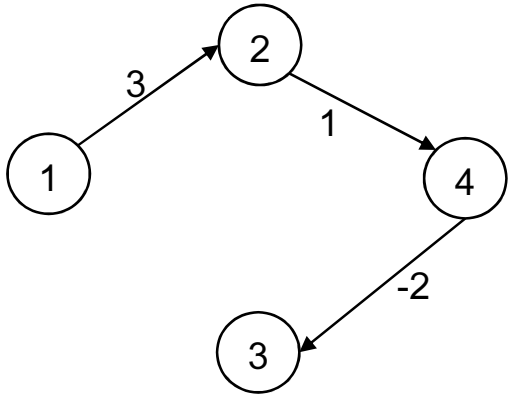
PASO 2

$$u_2^3 = \min\left\{u_2^2, \min\left\{u_3^2 + d_{32}, u_4^2 + d_{42}\right\}\right\} = \min\left\{3, 3+\infty, 4+5\right\} = 3$$

$$u_3^3 = \min\left\{u_3^2, \min\left\{u_2^2 + d_{23}, u_4^2 + d_{43}\right\}\right\} = \min\left\{6, 3+1, 4-2\right\} = 2$$

$$u_4^3 = \min\left\{u_4^2, \min\left\{u_2^2 + d_{24}, u_3^2 + d_{34}\right\}\right\} = \min\left\{4, 3+1, 3+4\right\} = 4$$

Como $m+1 = 3 = n-1 \Rightarrow$ Parar



Flujo en redes

- Red de transporte: Grafo dirigido $R = (V,A)$ que cumple
  - ✓ Existe una fuente *s*, nodo al que no llega ningún arco y desde el que se puede alcanzar el resto de nodos
  - ✓ Existe un sumidero *t*, nodo del que no sale ningún arco y que puede alcanzarse desde el resto de nodos
  - ✓ Cada arco tiene una capacidad (no negativa) asociada → c_{ij}
- Flujo compatible: Vector φ con tantas componentes como arcos que cumple dos propiedades
  1. No supera la capacidad de los arcos

$$\varphi_{ij} \leq c_{ij} \ \forall (i,j) \in A$$

  2. Verifica la ley de conservación de flujo

$$\forall i \neq s,t \quad \sum_{j/(j,i)\in A} \varphi_{ji} = \sum_{j/(i,j)\in A} \varphi_{ij}$$

- Valor del flujo → $\sum_{j/(s,j)\in A} \varphi_{sj}$, que coincide con $\sum_{j/(j,t)\in A} \varphi_{jt}$

Problema de flujo máximo

- Planteamiento: Dada una red de transporte $R = (V,A)$, determinar un flujo compatible para el que el valor del flujo sea máximo
- Capacidad residual de un arco $\rightarrow c^*_{ij} = c_{ij} - \varphi_{ij}$
- Camino de aumento: Cadena de la fuente al sumidero que cumple
  - ✓ Para cada arco orientado positivamente $\rightarrow c^*_{ij} > 0$
  - ✓ Para cada arco orientado negativamente $\rightarrow \varphi_{ij} > 0$
- Método de Ford-Fulkerson o del camino de aumento
  - ✓ Comenzar por un flujo compatible cualquiera
  - ✓ Buscar un camino de aumento y aumentar el flujo por ese camino todo lo posible
  - ✓ Parar cuando no exista ningún camino de aumento

Algoritmo de Ford-Fulkerson

PASO 1. COMENZAR CON UN FLUJO COMPATIBLE. CALCULAR LAS CAPACIDADES RESIDUALES

Hacer $\varphi_{ij} = 0 \; \forall (i,j) \in A$ y $c^*_{ij} = c_{ij} \; \forall (i,j) \in A$. Etiquetar la fuente $s \rightarrow (-, \infty)$

PASO 2. BUSCAR UN CAMINO DE AUMENTO

Sean $i, j \in V$ dos nodos adyacentes, etiquetado y sin etiquetar respectivamente.

a) Si $(i,j) \in A$ y $c^*_{ij} > 0$ se puede aumentar flujo: calcular $\delta_j = \min\{\delta_i, c^*_{ij}\}$. Etiquetar el nodo $j \rightarrow (i+, \delta_j)$

b) Si $(j,i) \in A$ y $\varphi_{ji} > 0$ se puede disminuir flujo: calcular $\delta_j = \min\{\delta_i, \varphi_{ji}\}$. Etiquetar el nodo $j \rightarrow (i-, \delta_j)$

Repetir hasta que el sumidero esté etiquetado (se habrá logrado obtener un camino de aumento formado por nodos etiquetados, cuya capacidad residual global es δ_t) e ir al PASO 3.

Si no puede etiquetarse el sumidero, PARAR. El flujo actual es el máximo.

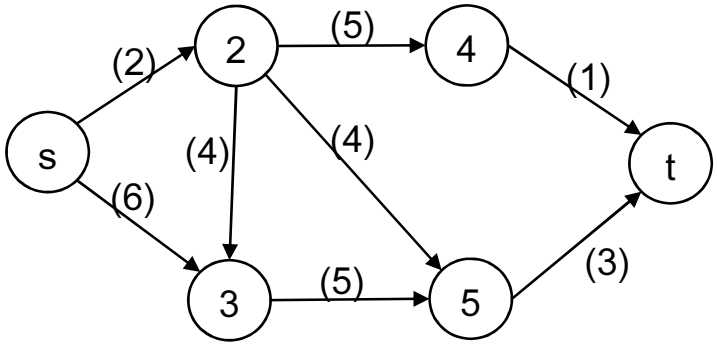
PASO 3. AUMENTAR EL FLUJO EN EL CAMINO DE AUMENTO OBTENIDO

En cada arco del camino orientado positivamente, hacer $\varphi_{ij} = \varphi_{ij} + \delta_t, \; c^*_{ij} = c_{ij} - \varphi_{ij}$

En cada arco del camino orientado negativamente, hacer $\varphi_{ij} = \varphi_{ij} - \delta_t, \; c^*_{ij} = c_{ij} - \varphi_{ij}$

Borrar todas las etiquetas, excepto la de la fuente, y volver al PASO 2.

Algoritmo de Ford-Fulkerson. Ejemplo



Las capacidades de los arcos aparecen entre paréntesis

PASO 1

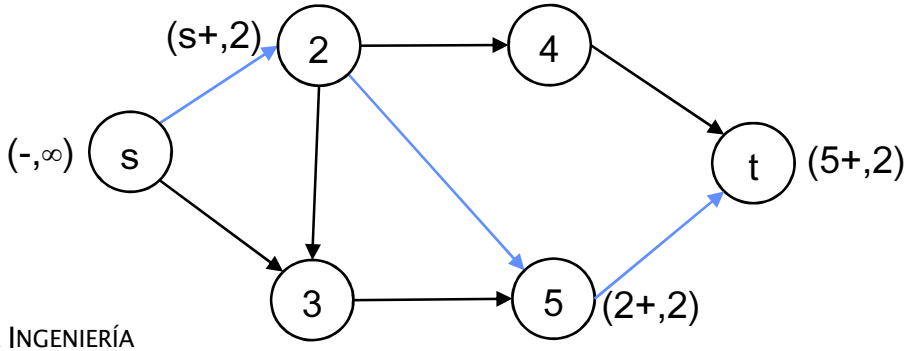
$\varphi_{ij} = 0, c^*_{ij} = c_{ij}\ \forall (i,j) \in A$. Etiquetar la fuente $s \rightarrow (-,\infty)$

PASO 2

$i = s, j = 2 \rightarrow$ Caso a) $\delta_2 = \min\{\infty, 2\} = 2$. Etiquetar $2 \rightarrow (s+,2)$

$i = 2, j = 5 \rightarrow$ Caso a) $\delta_5 = \min\{2, 4\} = 2$. Etiquetar $5 \rightarrow (2+,2)$

$i = 5, j = t \rightarrow$ Caso a) $\delta_t = \min\{2, 3\} = 2$. Etiquetar $t \rightarrow (5+,2)$. Ir al PASO 3

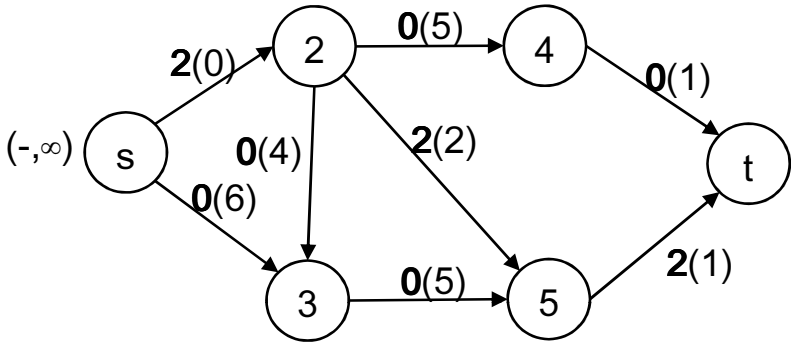


Algoritmo de Ford-Fulkerson. Ejemplo

PASO 3

Nuevo flujo $\varphi_{s2} = 0+2 = 2,\ \varphi_{25} = 0+2 = 2,\ \varphi_{5t} = 0+2 = 2$

Nuevas capacidades residuales $c^*_{s2} = 2-2 = 0,\ c^*_{25} = 4-2 = 2,\ c^*_{5t} = 3-2 = 1$

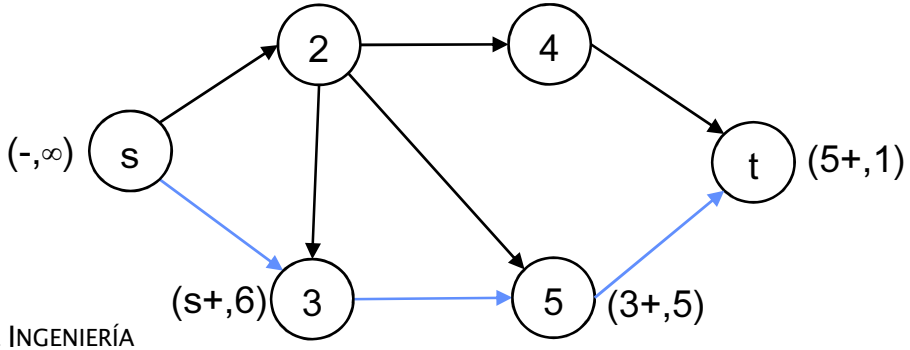


PASO 2

$i = s,\ j = 3 \rightarrow$ Caso a) $\delta_3 = \min\{\infty, 6\} = 6$. Etiquetar $3 \rightarrow (s+, 6)$

$i = 3,\ j = 5 \rightarrow$ Caso a) $\delta_5 = \min\{6, 5\} = 5$. Etiquetar $5 \rightarrow (3+, 5)$

$i = 5,\ j = t \rightarrow$ Caso a) $\delta_t = \min\{5, 1\} = 1$. Etiquetar $t \rightarrow (5+, 1)$. Ir al PASO 3

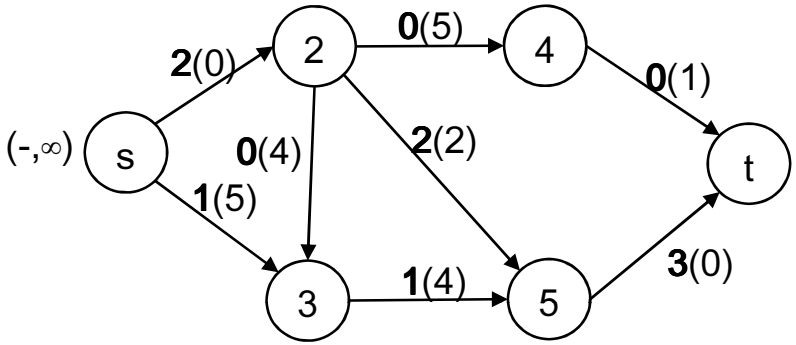


Algoritmo de Ford-Fulkerson. Ejemplo

PASO 3

Nuevo flujo $\varphi_{s3} = 0+1 = 1,\ \varphi_{35} = 0+1 = 1,\ \varphi_{5t} = 2+1 = 3$

Nuevas capacidades residuales $c^*_{s3} = 6-1 = 5,\ c^*_{35} = 5-1 = 4,\ c^*_{5t} = 3-3 = 0$



PASO 2

$i = s,\ j = 3 \rightarrow$ Caso a) $\delta_3 = \min\{\infty, 5\} = 5$. Etiquetar $3 \rightarrow (s+, 5)$

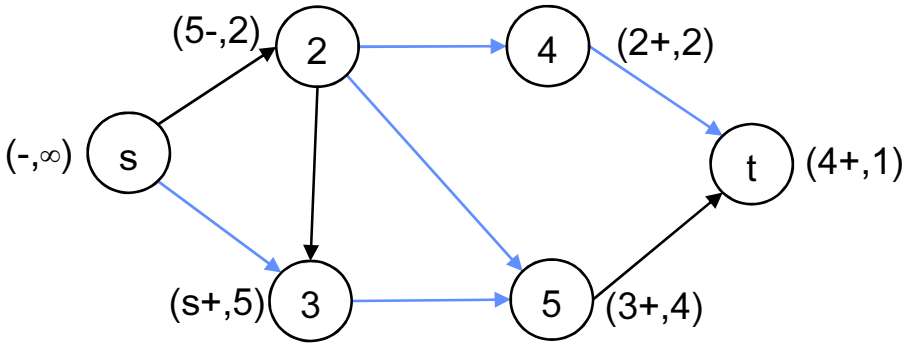
$i = 3,\ j = 5 \rightarrow$ Caso a) $\delta_5 = \min\{5, 4\} = 4$. Etiquetar $5 \rightarrow (3+, 4)$

$i = 5,\ j = 2 \rightarrow$ Caso b) $\delta_2 = \min\{4, 2\} = 2$. Etiquetar $2 \rightarrow (5-, 2)$

$i = 2,\ j = 4 \rightarrow$ Caso a) $\delta_4 = \min\{2, 5\} = 2$. Etiquetar $4 \rightarrow (2+, 2)$

$i = 4,\ j = t \rightarrow$ Caso a) $\delta_t = \min\{2, 1\} = 1$. Etiquetar $t \rightarrow (4+, 1)$. Ir al PASO 3

Algoritmo de Ford-Fulkerson. Ejemplo

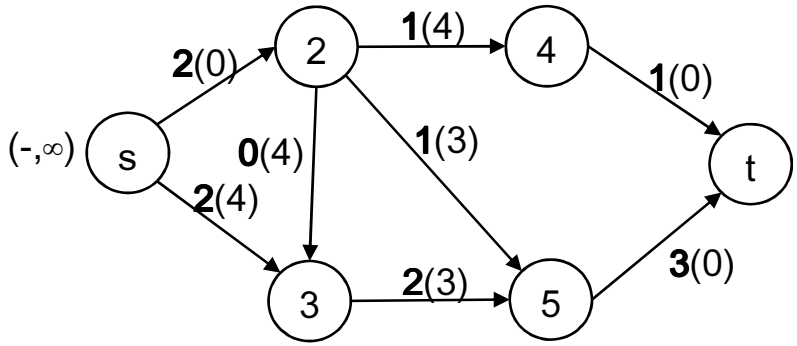


PASO 3

Nuevo flujo $\varphi_{s3}=1+1=2,\ \varphi_{35}=1+1=2,\ \varphi_{52}=2-1=1,\ \varphi_{24}=0+1=1,\ \varphi_{4t}=0+1=1$

Nuevas capacidades residuales

$c^*_{s3}=6-2=4,\ c^*_{35}=5-2=3,\ c^*_{52}=4-1=3,\ c^*_{24}=5-1=4, c^*_{4t}=1-1=0$



Algoritmo de Ford-Fulkerson. Ejemplo

PASO 2

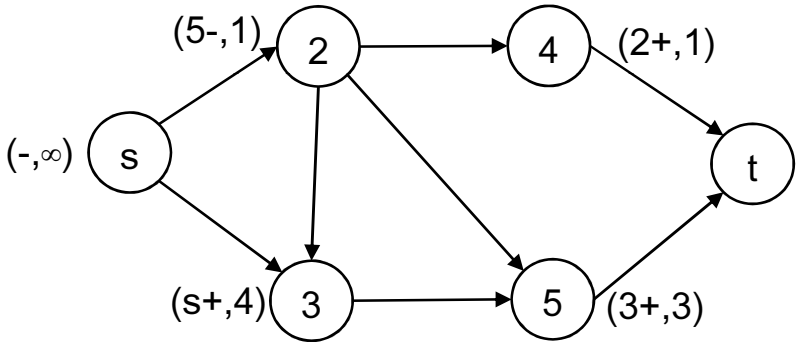
$i = s, j = 3 \rightarrow$ Caso a) $\delta_3 = \min\{\infty, 4\} = 4$. Etiquetar $3 \rightarrow (s+, 4)$

$i = 3, j = 5 \rightarrow$ Caso a) $\delta_5 = \min\{4, 3\} = 3$. Etiquetar $5 \rightarrow (3+, 3)$

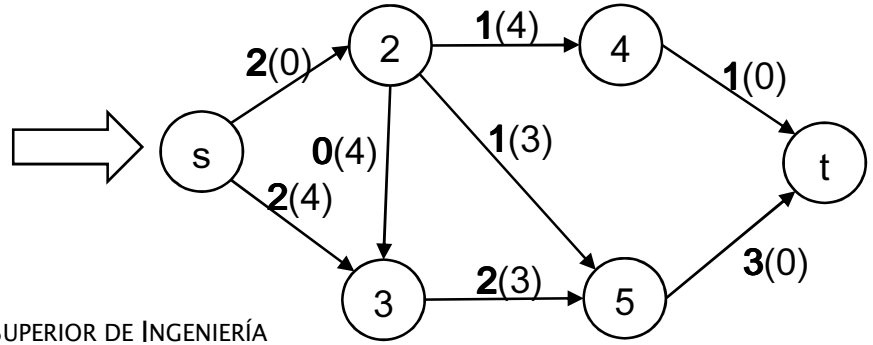
$i = 5, j = 2 \rightarrow$ Caso b) $\delta_2 = \min\{3, 1\} = 1$. Etiquetar $2 \rightarrow (5-, 1)$

$i = 2, j = 4 \rightarrow$ Caso a) $\delta_4 = \min\{1, 4\} = 1$. Etiquetar $4 \rightarrow (2+, 1)$

No es posible etiquetar más nodos, el sumidero no ha sido etiquetado. PARAR



Solución óptima

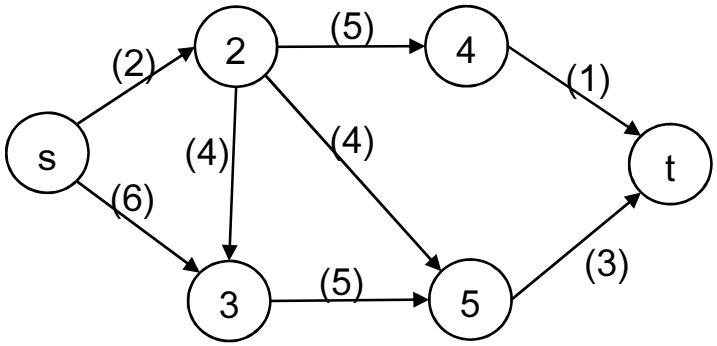


Valor del flujo

$$V(\varphi) = \varphi_{s2} + \varphi_{s3} = 2 + 2 = 4$$

Problema del corte mínimo

- Corte: Subconjunto de arcos de una red de transporte cuya eliminación desconecta la fuente del sumidero. Cada corte genera una partición de V en dos componentes: P a la que pertenece la fuente y $\overline{P}$ a la que pertenece el sumidero
- Capacidad de un corte: Suma de las capacidades de los arcos que forman el corte
- Ejemplo



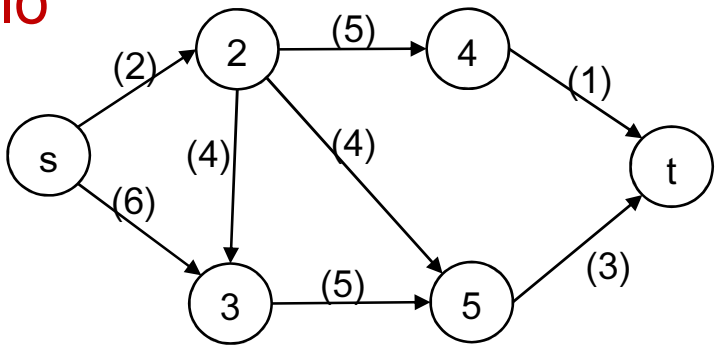
Los arcos (2,4) y (5,t) forman un corte
Las componentes son $P = \{s,2,3,5\}$ y $\overline{P} = \{4,t\}$
La capacidad del corte es $C(P,\overline{P}) = 5 + 3 = 8$

Problema del corte mínimo

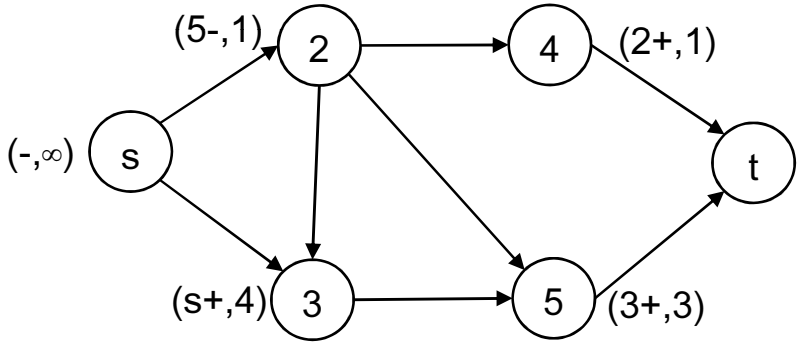
- Planteamiento del problema: Dada una red de transporte $R = (V,A)$, determinar un corte de capacidad mínima
- Relación con el flujo máximo
  - ✓ Dado un flujo compatible φ y un corte $(P,\overline{P})$ $\quad V(\varphi) \leq C(P,\overline{P})$
  - ✓ φ es flujo máximo y $(P,\overline{P})$ es corte mínimo $\Leftrightarrow V(\varphi) = C(P,\overline{P})$
- Resolución: se usa el algoritmo de Ford-Fulkerson. Una vez realizada la última iteración
  - ✓ P está formado por los nodos etiquetados
  - ✓ $\overline{P}$ está formado por los nodos no etiquetados

Problema del corte mínimo. Ejemplo

- Ejemplo



El algoritmo de Ford-Fulkerson lleva a



Corte mínimo: $P = \{s,2,3,4,5\}$ y $\overline{P} = \{t\}$

La capacidad del corte es $C(P,\overline{P}) = c_{4t} + c_{5t} = 1 + 3 = 4$

Problema de flujo compatible con coste mínimo

- Planteamiento del problema: Dada una red de transporte $R = (V,A)$, con costes unitarios de flujo d_{ij} asociados a los arcos, determinar un flujo compatible de valor fijo ϑ, con coste total mínimo

$$\min \quad \sum_{(i,j)\in A} d_{ij}\varphi_{ij}$$

$$\sum_{i/(s,i)\in A} \varphi_{ij} = \theta$$

$$\sum_{i/(i,t)\in A} \varphi_{ij} = \theta$$

$$\sum_{i/(j,i)\in A} \varphi_{ji} - \sum_{i/(i,j)\in A} \varphi_{ij} = 0 \quad \forall j \neq s,t$$

$$0 \leq \varphi_{ij} \leq c_{ij} \quad \forall (i,j) \in A$$