UNIVERSIDAD PONTIFICIA
ICAI ICADE
COMILLAS
MADRID

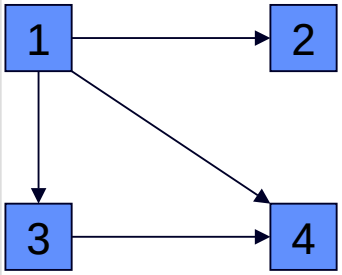
Teoría de redes y optimización en redes

Pedro Sánchez Martín

Contenidos

- Definiciones básicas
- Árbol generador mínimo de expansión
- Camino mínimo
  - Algoritmo Dijkstra
  - Algoritmo Bellman-Ford
- Flujo máximo
- Flujo de coste mínimo

Definiciones básicas (I)

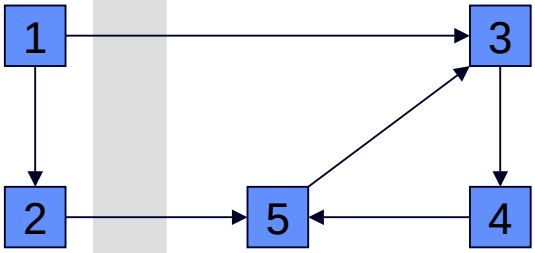


Red		Nodos	Vértices	Nudos
Grafo		Arcos	Aristas	Ramas

N = {1, 2, 3, 4}
A = {(1, 2), (1, 3), (1, 4), (3, 4)}

Arco dirigido	Par orientado de vértices
Red dirigida	Conjunto de nodos unidos por arcos dirigidos
Cadena	Secuencia de arcos no dirigidos que unen dos nodos
Camino	Secuencia de arcos dirigidos que unen dos nodos
Ciclo	Cadena que une un nodo con él mismo
Circuito	Camino que une un nodo con él mismo

Definiciones básicas (II)



Camino NO dirigido (cadena)	(1, 2) (2, 5) (5, 4)
Camino dirigido (camino)	(1, 2) (2, 5) (5, 3) (3, 4)
Ciclo	(1, 2) (2, 5) (5, 3) (3, 1)
Circuito	(5, 3) (3, 4) (4, 5)

Nodos conexos: dos nodos son conexos si la red contiene al menos una cadena entre ellos

Red conexa: red donde cada pareja de nodos es conexa

Árbol: red conexa sin ciclos

Árbol generador: red conexa sin ciclos que tiene exactamente *n*-1 arcos para recorrer todos los *n* nodos

Capacidad de un arco: máximo flujo por un arco

Nudo de generación: aquél cuyo flujo saliente excede al entrante

Nudo de demanda: aquél cuyo flujo entrante excede al saliente

***Nudo de transbordo* (*conexión*):** aquél que conserva el flujo

Definiciones básicas (III)

Matrices

Matriz de **incidencia**: filas nodos, columnas aristas,

elementos $b_{ij} = \begin{cases} 1 & \text{si el nodo } i \text{ pertenece a la arista } j \\ 0 & \text{otro caso} \end{cases}$

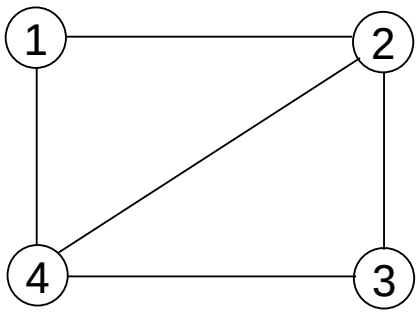
Grafo dirigido: -1 nodo inicial, +1 nodo final.
Dos elementos por columna. Matriz unimodular.

Matriz de **adyacencia**: filas nodos, columnas nodos,

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

Matriz simétrica.

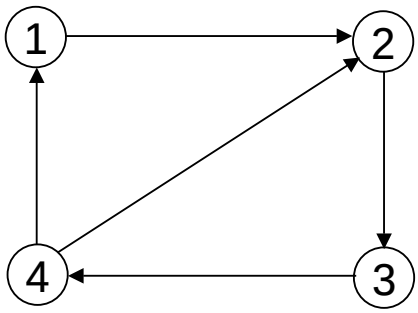
Definiciones básicas (IV)



Matriz de incidencia $B = \begin{pmatrix} 1 & 0 & 0 & 1 & 0 \\ 1 & 1 & 0 & 0 & 1 \\ 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 1 \end{pmatrix}$

Matriz de adyacencia $A = \begin{pmatrix} 0 & 1 & 0 & 1 \\ 1 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 \\ 1 & 1 & 1 & 0 \end{pmatrix}$

Definiciones básicas (IV)



Matriz de incidencia $B = \begin{pmatrix} -1 & 0 & 0 & 1 & 0 \\ 1 & -1 & 0 & 0 & 1 \\ 0 & 1 & -1 & 0 & 0 \\ 0 & 0 & 1 & -1 & -1 \end{pmatrix}$

Matriz de adyacencia $A = \begin{pmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 1 & 1 & 0 & 0 \end{pmatrix}$

Árbol generador mínimo

Aplicaciones:
Redes de telecomunicación, de distribución o de transporte

Hipótesis:
Red conexa no dirigida
Distancia no negativa en cada arco

Objetivo:
Encontrar la cadena de longitud o peso mínimo que recorre todos los nodos sin ciclos (es decir, el árbol generador de mínimo peso).

Árbol generador mínimo (Algoritmo de Prim)

Notación

n : número de nodos del grafo G (n =card (V))
T : árbol construido
C_k : conjunto de nodos conectados hasta la iteración k
$\overline{C}_k$: conjunto de nodos todavía no conectados hasta la iteración k

Algoritmo

Paso 1: Selección de un vértice inicial cualquiera

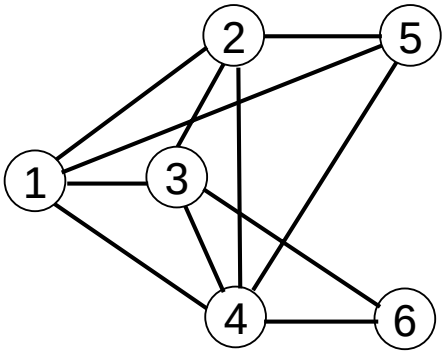
Hacer $C_0 = \varnothing$, $\overline{C}_0 = V$. Elegir un vértice cualquiera $i \in V$ y hacer $C_1 = \{i\}$, $\overline{C}_1 = V - \{i\}$. Sea $k = 2$ y $T = \varnothing$.

Paso 2: Selección del vértice más cercano a los conectados y que todavía no haya sido conectado, añadir la arista al árbol generador

Seleccionar $j^* \in \overline{C}_{k-1}$, un nodo no conectado tal que se une a algún vértice ya conectado (de C_{k-1}) con la arista de menor peso e_{k-1} .
Hacer $C_k = C_{k-1} \cup \{j^*\}$, $\overline{C}_k = \overline{C}_{k-1} - \{j^*\}$ y $T = T \cup \{e_{k-1}\}$

Si $k = n$, parar. Si no, poner $k = k+1$ y repetir el paso 2.

Árbol generador mínimo (Ejemplo)



$$\begin{pmatrix} . & 1 & 5 & 7 & 9 & \infty \\ 1 & . & 6 & 4 & 3 & \infty \\ 5 & 6 & . & 5 & \infty & 10 \\ 7 & 4 & 5 & . & 8 & 3 \\ 9 & 3 & \infty & 8 & . & \infty \\ \infty & \infty & 10 & 3 & \infty & . \end{pmatrix}$$

Paso 1:

$C_0 = \varnothing$, $\overline{C}_0 = V = \{1,2,3,4,5,6\}$.
Sea por ejemplo $i = 1$: $C_1 = \{1\}$, $\overline{C}_1 = \{2,3,4,5,6\}$. Sea $k = 2$ y $T = \varnothing$

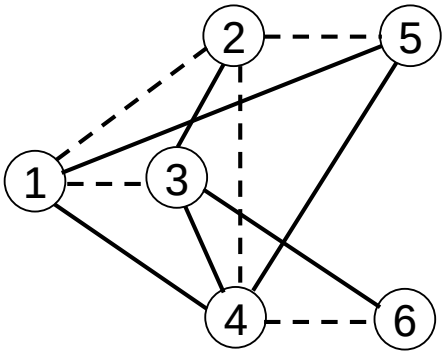
Paso 2:

$j^* = 2$ $e_1 = \{1,2\}$ $C_2 = \{1,2\}$ $\overline{C}_2 = \{3,4,5,6\}$ $T = \{e_1\}$ $k < 6$ $k = 3$

Paso 3:

$j^* = 5$ $e_2 = \{2,5\}$ $C_3 = \{1,2,5\}$ $\overline{C}_3 = \{3,4,6\}$ $T = \{e_1, e_2\}$ $k < 6$ $k = 4$

Árbol generador mínimo (Ejemplo cont.)



$$\begin{pmatrix} . & 1 & 5 & 7 & 9 & \infty \\ 1 & . & 6 & 4 & 3 & \infty \\ 5 & 6 & . & 5 & \infty & 10 \\ 7 & 4 & 5 & . & 8 & 3 \\ 9 & 3 & \infty & 8 & . & \infty \\ \infty & \infty & 10 & 3 & \infty & . \end{pmatrix}$$

Paso 4:

$j^* = 4 \quad e_3 = \{2,4\} \quad C_4 = \{1,2,5,4\} \quad \overline{C}_4 = \{3,6\} \quad T = \{e_1, e_2, e_3\} \quad k < 6 \quad k = 5$

Paso 5:

$j^* = 6 \quad e_4 = \{4,6\} \quad C_5 = \{1,2,5,4,6\} \quad \overline{C}_5 = \{3\} \quad T = \{e_1, e_2, e_3, e_4\} \quad k < 6 \quad k = 6$

Paso 6:

$j^* = 3 \quad e_5 = \{1,3\} \quad C_6 = \{1,2,5,4,6,3\} \quad \overline{C}_6 = \varnothing \quad T = \{e_1, e_2, e_3, e_4, e_5\} \quad k = 6$ parar

Camino mínimo

Hipótesis:

- Red conexa dirigida o no dirigida
- Dos nodos especiales: *origen y destino*
- Cada arco lleva asociado una distancia:
  - No negativa: Algoritmo Djikstra
  - Positiva o negativa: Algoritmo Bellman-Ford

Otras posibilidades:

- Restricciones en los caminos

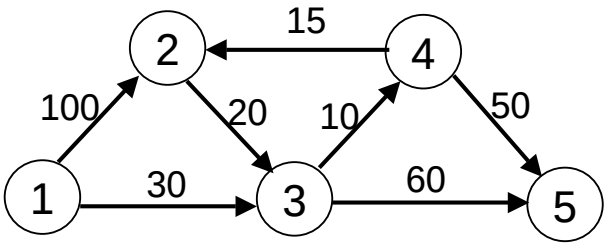
Objetivo:

- Encontrar la distancia (coste, tiempo,...) mínima entre el origen y el destino. Al mismo tiempo se consigue el menor camino entre el origen y el resto de los nodos (arborescencia)

Camino mínimo (Algoritmo Dijkstra)

- Es más eficiente que el algoritmo Bellman-Ford si no hay distancias negativas
- Se basa en el principio de Bellman de que el óptimo está formado por subóptimos

- Se etiquetan los nodos cuando se determina su camino mínimo al origen
- En cada paso se obtiene el camino mínimo al nodo origen de los nodos no etiquetados
- La evaluación del camino mínimo de los nodos no etiquetados se hace comparando la distancia previa al último nodo etiquetado y la distancia considerando el último nodo etiquetado
- El nodo con camino mínimo de los no etiquetados se etiqueta



Este algoritmo no es válido para arcos de longitud negativa

Camino mínimo (Algoritmo Dijkstra)

Paso 0: Inicio, se etiquetan los nodos con la longitud del arco que los une con el origen

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

Paso 1: Designar la etiqueta permanente de la iteración: vértice con menor valor de la etiqueta transitoria

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

Si $T = \varnothing$, parar.

Paso 2: Revisar las etiquetas transitorias

$\forall j \in T$ calcular $u_j = \min\{u_j, u_k + d_{kj}\}$, si se modifica poner $pred(j) = k$.

Volver al Paso 1.

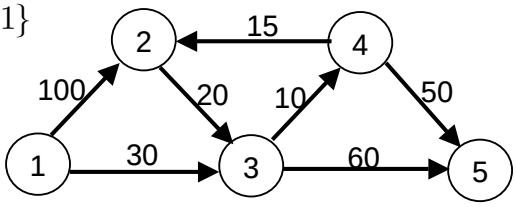
P : Conjunto de nodos etiquetados
T : Conjunto de nodos NO etiquetados

Camino mínimo (Ejemplo)

Paso 0:

$u_1 = 0 \;\; u_2 = 100 \;\; u_3 = 30 \;\; u_4 = \infty \;\; u_5 = \infty \;\; P = \{1\}$

$T = \{2,3,4,5\} \quad pred(j) = 1,\ j = 2,3,4,5$



Paso 1:

$u_k = 30 \;\; k = 3 \;\; P = \{1,3\} \;\; T = \{2,4,5\} \neq \varnothing$

Paso 2:

$u_2 = \min\{u_2, u_3 + d_{32}\} = \min\{100, 30 + \infty\} = 100$

$u_4 = \min\{u_4, u_3 + d_{34}\} = \min\{\infty, 30 + 10\} = 40 \quad pred(4) = 3$

$u_5 = \min\{u_5, u_3 + d_{35}\} = \min\{\infty, 30 + 60\} = 90 \quad pred(5) = 3$

Paso 1:

$u_k = 40 \;\; k = 4 \;\; P = \{1,3,4\} \;\; T = \{2,5\} \neq \varnothing$

Paso 2:

$u_2 = \min\{u_2, u_4 + d_{42}\} = \min\{100, 40 + 15\} = 55 \quad pred(2) = 4$

$u_5 = \min\{u_5, u_4 + d_{45}\} = \min\{90, 40 + 50\} = 90 \quad pred(5) = 4$

Camino mínimo (Ejemplo cont.)

Paso 1:

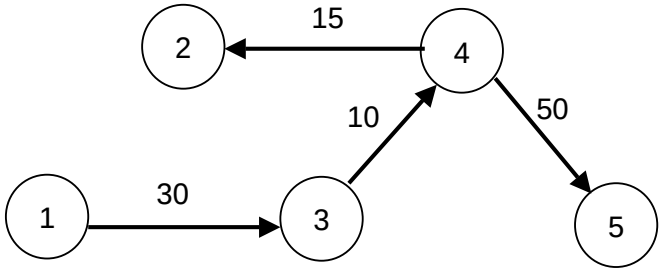
$u_k = 55 \quad k = 2 \quad P = \{1,3,4,2\} \quad T = \{5\} \neq \varnothing$

Paso 2:

$u_5 = \min\{u_5, u_2 + d_{25}\} = \min\{90, 55 + \infty\} = 90$

Paso 1:

$u_k = 90 \quad k = 5 \quad P = \{1,3,4,2,5\} \;\; T = \varnothing$ parar



Camino mínimo (Algoritmo de Bellman-Ford)

- Se utiliza sólo para el ***caso de arcos con distancia negativa***
- No se aplica si existe un circuito con recorrido negativo, es decir, la suma de pesos de los arcos del circuito suma menos de cero

u_j^m :longitud del camino mínimo del vértice 1 al vértice j usando a lo sumo m arcos

Algoritmo:

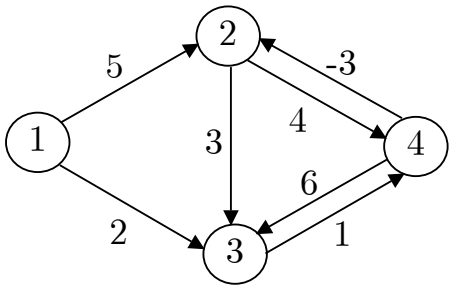
Paso 1: Iniciar con longitudes de los arcos del nodo 1 a los demás nodos
$u_1^1 = 0 \quad u_j^1 = d_{1j} \quad j = 2,...,n$ Poner $m = 1$

Paso 2: Calcular las distancias mínimas usando a lo sumo $m+1$ arcos

Calcular $u_j^{m+1} = \min\left\{u_j^m, \min_{k \neq j}\left\{u_k^m + d_{kj}\right\}\right\}$.

Si $u_j^{m+1} = u_j^m \quad \forall j$, parar. Si no, poner $m = m+1$ y volver al paso 2.

Camino mínimo (Ejemplo de Bellman-Ford)



Paso 1:
$u_1^1 = 0 \;\; u_2^1 = 5 \;\; u_3^1 = 2 \;\; u_4^1 = \infty$ pred(j)=1 j=2,3,4. $m = 1$

Paso 2:

$$u_2^2 = \min\left\{u_2^1, \min\left\{u_3^1 + d_{32}, u_4^1 + d_{42}\right\}\right\} = \min\left\{5, \min\left\{2+\infty, \infty - 3\right\}\right\} = 5$$
$$u_3^2 = \min\left\{u_3^1, \min\left\{u_2^1 + d_{23}, u_4^1 + d_{43}\right\}\right\} = \min\left\{2, \min\left\{5+3, \infty + 6\right\}\right\} = 2$$
$$u_4^2 = \min\left\{u_4^1, \min\left\{u_2^1 + d_{24}, u_3^1 + d_{34}\right\}\right\} = \min\left\{\infty, \min\left\{5+4, 2+1\right\}\right\} = 3$$

Como $u_4^2 \neq u_4^1$, pred(4)=3, poner $m = 2$ y volver al paso 2.

Camino mínimo (Ejemplo de Bellman-Ford)

Paso 2:

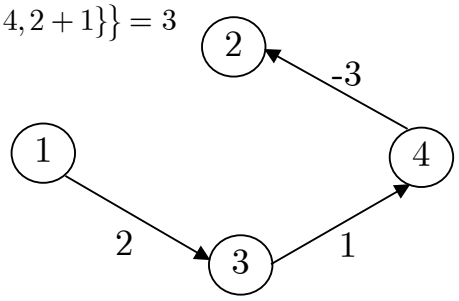
$$u_2^3 = \min\left\{u_2^2, \min\left\{u_3^2 + d_{32}, u_4^2 + d_{42}\right\}\right\} = \min\left\{5, \min\left\{2+\infty, 3-3\right\}\right\} = 0$$
$$u_3^3 = \min\left\{u_3^2, \min\left\{u_2^2 + d_{23}, u_4^2 + d_{43}\right\}\right\} = \min\left\{2, \min\left\{5+3, 3+6\right\}\right\} = 2$$
$$u_4^3 = \min\left\{u_4^2, \min\left\{u_2^2 + d_{24}, u_3^2 + d_{34}\right\}\right\} = \min\left\{3, \min\left\{5+4, 2+1\right\}\right\} = 3$$

Como $u_2^3 \neq u_2^2$, pred(2)=4, poner $m = 3$ y volver al paso 2 (aunque ya se sabe que se ha acabado por ser $m = n - 1 = 4 - 1 = 3$.

Paso 2:

$$u_2^4 = \min\left\{u_2^3, \min\left\{u_3^3 + d_{32}, u_4^3 + d_{42}\right\}\right\} = \min\left\{0, \min\left\{2+\infty, 3-3\right\}\right\} = 0$$
$$u_3^4 = \min\left\{u_3^3, \min\left\{u_2^3 + d_{23}, u_4^3 + d_{43}\right\}\right\} = \min\left\{2, \min\left\{0+3, 3+6\right\}\right\} = 2$$
$$u_4^4 = \min\left\{u_4^3, \min\left\{u_2^3 + d_{24}, u_3^3 + d_{34}\right\}\right\} = \min\left\{3, \min\left\{0+4, 2+1\right\}\right\} = 3$$

Como $u_j^4 = u_j^3 \quad \forall j$, parar.



Problema del flujo máximo

Hipótesis:

- Red conexa dirigida sin bucles
- Nodos *origen* y *destino*, el resto son de transbordo
- Cada arco tiene una capacidad asociada

Objetivo:

- Maximizar el flujo total desde el origen al destino

Métodos de resolución:

- Programación lineal
- Algoritmo del camino de aumento (Ford-Fulkerson)

Conceptos del problema de flujo máximo

Red de transporte:

- Una única fuente s
- Un único sumidero t
- Cada arco tiene una capacidad asociada
- Cada vértice es alcanzable desde la fuente y el sumidero es alcanzable desde todos los vértices

Flujo compatible:

- Es un vector φ de dimensión igual al número de arcos con dos propiedades:
  1. El flujo no supera la capacidad de ningún arco ($\varphi_{ij} \leq c_{ij} \quad \forall (i,j)$)
  2. Verifica ley de conservación de flujo

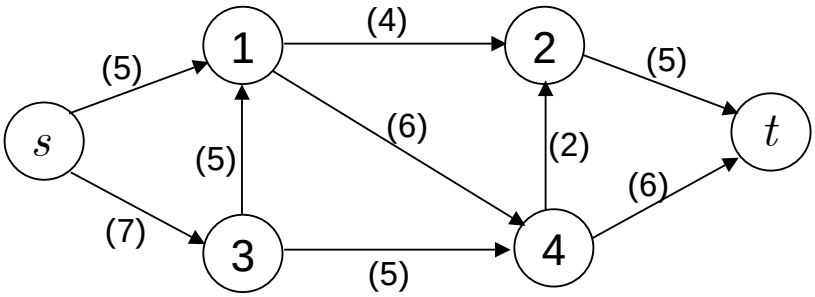
$$\forall i \neq s,t \qquad \sum_{j/(j,i)\in U} \varphi_{ji} - \sum_{j/(i,j)\in U} \varphi_{ij} = 0$$

El flujo que sale de la fuente coincide con el que llega al sumidero. Dicho valor se denomina **valor** del flujo

Conceptos del problema de flujo máximo

Corte:

- Conjunto de arcos cuya eliminación desconecta la fuente del sumidero
- Da lugar a dos componentes: uno incluye la fuente y el otro el sumidero



Los arcos (1,2), (1,4) y (3,4) forman un corte obteniendo los componentes {s,1,3} y {2,4,t}

Capacidad de corte:

- Suma de las capacidades de los arcos que forman el corte

En el ejemplo la capacidad de corte es de 15

Conceptos del problema de flujo máximo

Corte de mínima capacidad:

- Consiste en encontrar el corte cuya capacidad sea mínima

Dado un flujo compatible φ , su capacidad es siempre menor que cualquier capacidad de corte. Por ello se puede asegurar que:

"El máximo flujo que se puede enviar de la fuente al sumidero coincide con la mínima capacidad de un corte"

La aplicación de esta afirmación da lugar al **Algoritmo de Ford-Fulkerson o también conocido como del camino de aumento**

Algoritmo de Ford-Fulkerson

Capacidad residual de un arco:

· Es la diferencia entre la capacidad de un arco y el flujo asignado a éste

Camino de aumento:

· Es un camino de la fuente al sumidero tal que el mínimo de las capacidades residuales de los arcos que lo forman es positivo, denominando a este valor **capacidad residual del camino de aumento**

Algoritmo:

Paso 1:Comenzar con un flujo compatible y obtener las capacidades residuales de los arcos

Poner flujo 0 $\varphi_{ij} = 0 \; \forall (i,j) \in U$ y capacidades residuales $c^*_{ij} = c_{ij} \; \forall (i,j) \in U$.
Etiquetar $s \rightarrow (-, \infty)$

Algoritmo de Ford-Fulkerson (cont.)

Algoritmo (continuación):

Paso 2:Obtener un camino de aumento, incluso redireccionando flujo anteriormente asignado

Sea $i \in V$ un nodo ya etiquetado y sea $j \in V$ un nodo sin etiquetar tal que existe el arco (i, j) o el arco (j,i) . Hacer según el caso:

a) Si $(i, j) \in V$ y la capacidad residual no es 0 ($c_{ij}^{*} > 0$) puede aumentar flujo, calcular el mínimo de la capacidad residual de los arcos anteriores y este arco y etiquetar el nodo, es decir, calcular $\delta_j = \min\left\{\delta_i, c_{ij}^{*}\right\}$ y etiquetar el nodo j con $j \to (i+, \delta_j)$

b) Si $(j,i) \in V$ y se está enviando flujo por ese arco ($\varphi_{ji} > 0$) se puede redireccionar parte de ese flujo (se puede disminuir en ese arco), entonces calcular $\delta_j = \min\left\{\delta_i, \varphi_{ji}\right\}$ y etiquetar el nodo j con $j \to (i-, \delta_j)$

Repetir hasta que el sumidero esté etiquetado e ir al paso 3 o hasta que no sea posible etiquetar más nodos e ir al paso 4

Algoritmo de Ford-Fulkerson (cont.)

Algoritmo (continuación):

Paso 3:Aumentar el flujo en el camino obtenido y recalcular las capacidades residuales

Recorrer el camino de nodos etiquetados aumentando el flujo en los arcos etiquetados positivamente y disminuyéndolo en los etiquetados negativamente en la cantidad con que se etiquetó el sumidero δ_t (capacidad residual del camino). Recalcular las capacidades residuales (disminuir en los arcos positivos y aumentar en los negativos). Borrar las etiquetas, excepto la de la fuente, y volver al paso 2.

Paso 4: Parar, no es posible aumentar el flujo

El flujo obtenido es máximo y el corte de capacidad mínima viene dado por los nodos etiquetados en una componente y los no etiquetados en otra (es decir, el corte son los arcos que vayan de una componente a otra).

Flujo compatible con coste mínimo

- En este problema se incorpora un nuevo elemento en la red, el *coste* d_{ij} .
- Se supone asociado a cada arco un coste unitario de envío de flujo
- El problema consiste en enviar una cantidad de flujo conocida, θ, a través de la red desde la fuente (origen) al sumidero (destino), con coste total mínimo.

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

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

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

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

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