ESCUELA TÉCNICA SUPERIOR DE INGENIERÍA
DEPARTAMENTO DE ORGANIZACIÓN INDUSTRIAL

Investigación Operativa

Teoría de Grafos o Redes

Andrés Ramos
Pedro Linares
Pedro Sánchez
Angel Sarabia
Begoña Vitoriano

ESCUELA TÉCNICA SUPERIOR DE INGENIERÍA
DEPARTAMENTO DE ORGANIZACIÓN INDUSTRIAL

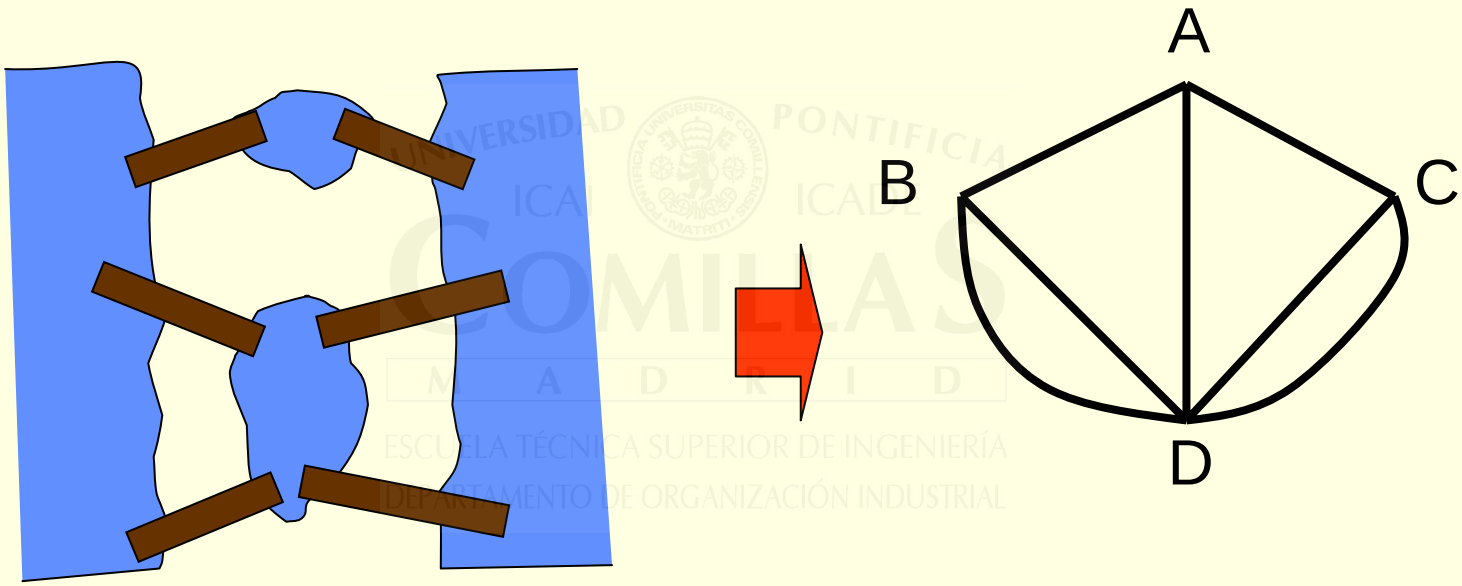
Teoría de Grafos o Redes

Parte I

Conceptos y problemas básicos

Introducción

- El problema de los puentes de Königsberg, s.XVIII



- Euler resolvió este problema mediante la teoría de grafos: sólo puede haber un ciclo euleriano cuando todos los nodos tienen un número par de aristas incidentes

Interés de la teoría de grafos (1)

- Sirve para resolver problemas representables mediante una red o un grafo
  - ✓ Modelos de optimización sobre redes de comunicaciones
    - Caminos de valor óptimo
      - Redes de actividades (Gestión de proyectos)
      - Redes de transporte y/o canalización
      - Encaminamiento óptimo en redes de comunicaciones (Internet, telefonia, etc) y distribución
    - Flujos óptimos sobre una red
    - Cálculo de puntos de control en una red de comunicaciones
  - ✓ Inventarios
  - ✓ Programación dinámica
  - ✓ Cadenas de Markov
  - ✓ Modelado de fenómenos de esperas (colas)

Interés de la teoría de grafos (2)

- Proporciona algoritmos muy eficientes, ya sean clásicos o metaheurísticos.

Algoritmos eficientes

- La eficiencia se puede medir en función de:
  - ✓ la capacidad de almacenamiento requerida
  - ✓ el tiempo de ejecución: complejidad
- La complejidad depende del número de operaciones elementales f(n) y de la dimensión del problema n

$$\theta(g(n)) \text{ si } \lim_{n\to\infty} \frac{f(n)}{g(n)} = cte$$

- Criterio del tiempo máximo o peor de los casos
- Tipos de algoritmo
  - ✓ polinomial: $\theta(n^p)$
  - ✓ no polinomial: $\theta(m^n)$

Algoritmos eficientes

- La eficiencia se puede medir en función de:
  - ✓ la capacidad de almacenamiento requerida
  - ✓ el tiempo de ejecución: complejidad
- La complejidad depende del número de operaciones elementales f(n) y de la dimensión del problema n

$$\theta(g(n)) \text{ si } \lim_{n\to\infty} \frac{f(n)}{g(n)} = cte$$

- Criterio del tiempo máximo o peor de los casos
- Tipos de algoritmo
  - ✓ polinomial: $\theta(n^p)$
  - ✓ no polinomial: $\theta(m^n)$

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

Teoría de Grafos o Redes

REDES ORIENTADAS

Teoría de Redes: Definición

Grafo o Red dirigido (u orientado) simple:

Es un objeto matemático G(V, U) donde:

- V es un conjunto cuyos elementos reciben el nombre de nodos ó vértices

- U es una relación binaria definida sobre V

Si *i* y *j* son dos nodos de V tales que *i*U*j* se dice que el par *(i,j)* es un arco de la red. El vértice *i* es el origen del arco y el vértice *j* su extremo

Dos arcos *(i, j)* y *(h, k)* se dice que son adyacentes si *j = h*, es decir cuando el extremo de uno de ellos es el origen del otro.

Representación de una red orientada simple (I)

a) Representación sagital:

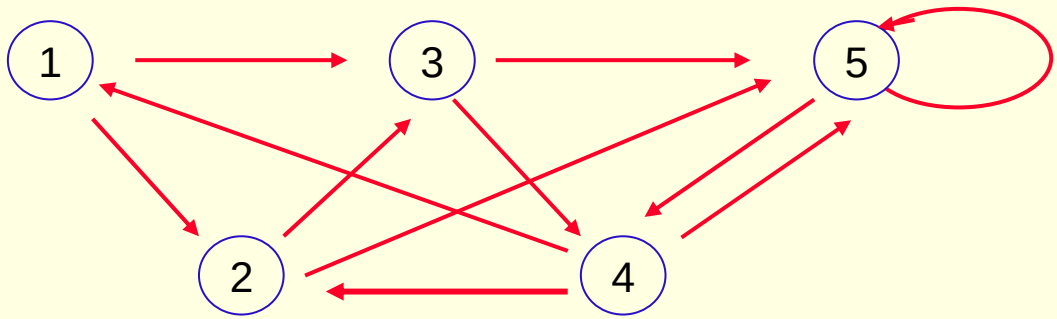
Cada vértice es representado por un punto y cada arco por una flecha cuyo origen es el primer elemento del arco y cuyo extremo es el segundo elemento.

Así, si V y R vienen definidos por

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

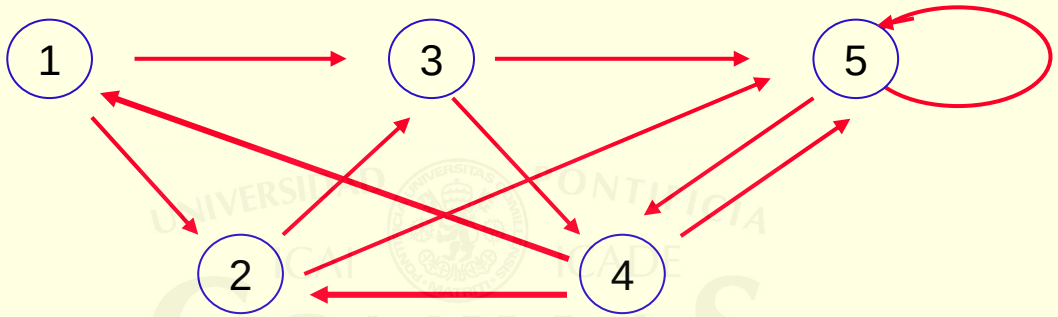
$$U = \{(1,3),(1,2),(2,3),(2,5),(3,4),(3,5),(4,1),(4,2),(4,5),(5,4),(5,5)\}$$

su representación sagital sería



Representación de una red orientada simple (II)

b) Por medio de una matriz, llamada de incidencia:

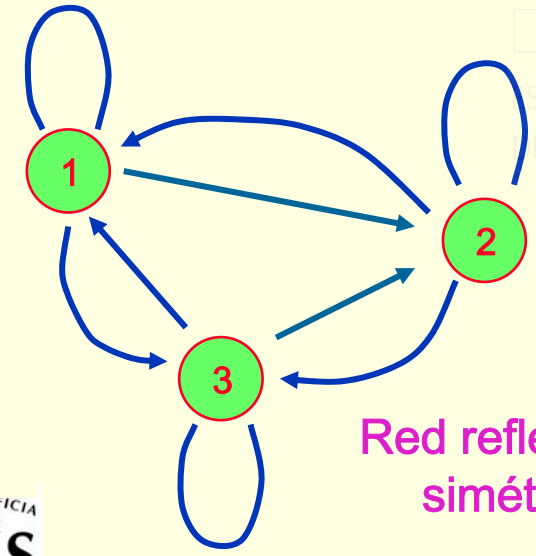


Una matriz cuadrada, con tantas filas y columnas como vértices tenga la red. En la posición *(i, j)* se coloca un 1 si hay un arco con origen en *i* y extremo en *j*; en otro caso, se pone un 0.

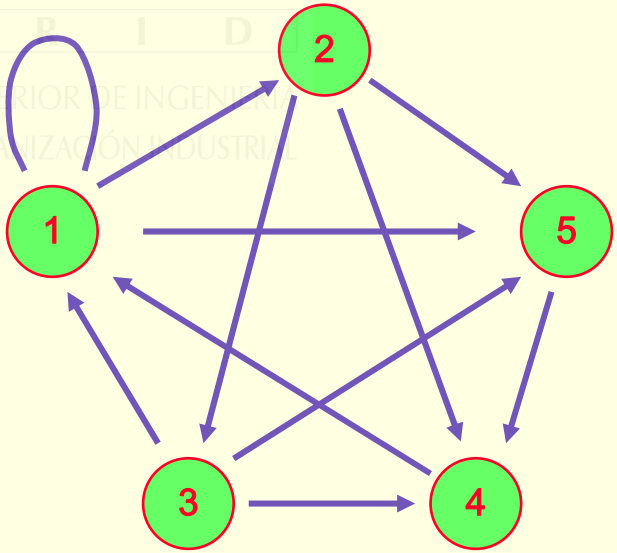
		Extremo				
		1	2	3	4	5
Origen	1	0	1	1	0	0
	2	0	0	1	0	1
	3	0	0	0	1	1
	4	1	1	0	0	1
	5	0	0	0	1	1

Redes orientadas: Tipos de redes

- Una red se dice que es *reflexiva* si a cada vértice de la red hay asociado un bucle. Su matriz de incidencia tiene su diagonal principal formada por 1´s.
- Una red es *simétrica* cuando, en caso de existir el arco (*i, j*) existe también el arco *(j, i)*. Su matriz de incidencia es simétrica. Si, caso de existir el arco (*i, j*), no existe el arco *(j, i). I* la red se dice que es *antisimétrica* . También lo es su matriz de incidencia.
- Una red se dice que es *transitiva* si la existencia de los arcos (*i,j*) y (*j,k*) supone también la del arco (*i,k*). Una red es *completa* cuando entre dos vértices cualesquiera existe al menos un arco que los une en uno de los dos sentidos.



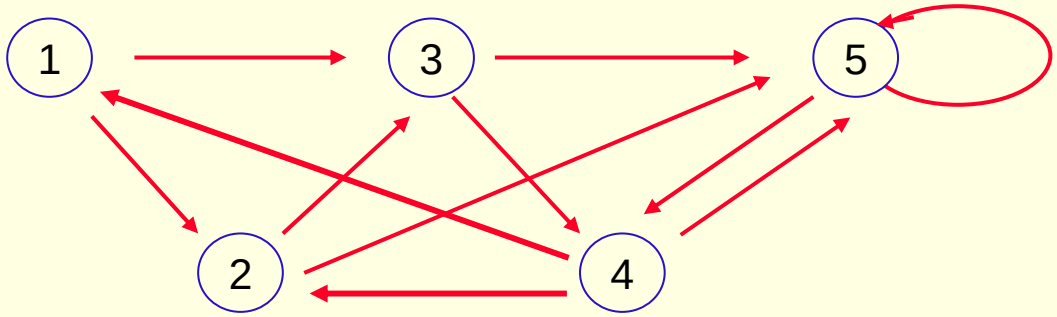
Red reflexiva y simétrica



Red completa

Redes orientadas: Definiciones básicas (I)

- Dado un vértice *i*, el conjunto de arcos que lo tienen como extremo recibe el nombre de arcos incidentes interiormente sobre *i*, lo notaremos por ***U(i-)*** y su cardinal recibe el nombre de ***semigrado interior*** de *i*. El conjunto de arcos de los que es origen recibe el nombre de arcos incidentes exteriormente sobre *i*, lo notaremos por ***U(i+)*** y su cardinal recibe el nombre de ***semigrado exterior*** de *i*.
- El número de arcos que inciden sobre un vértice es su **grado**.



En la red de la figura

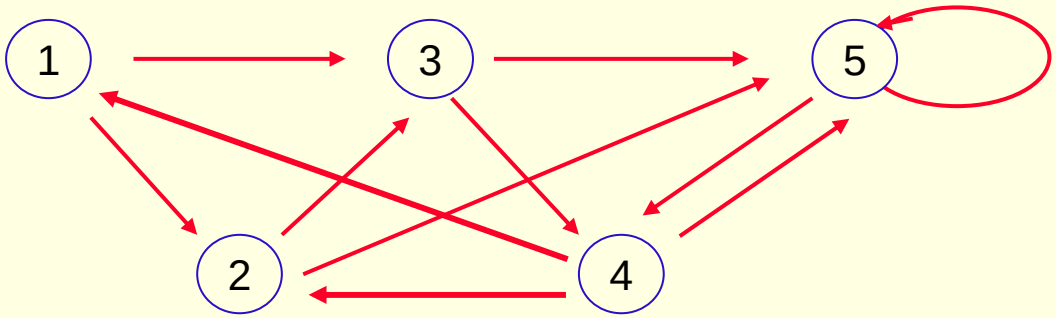
U(4 -) = 2

U(4 +) = 3

Redes orientadas: Definiciones básicas (II)

- **Dos vértices que están unidos al menos por un arco se dice que son adyacentes.**

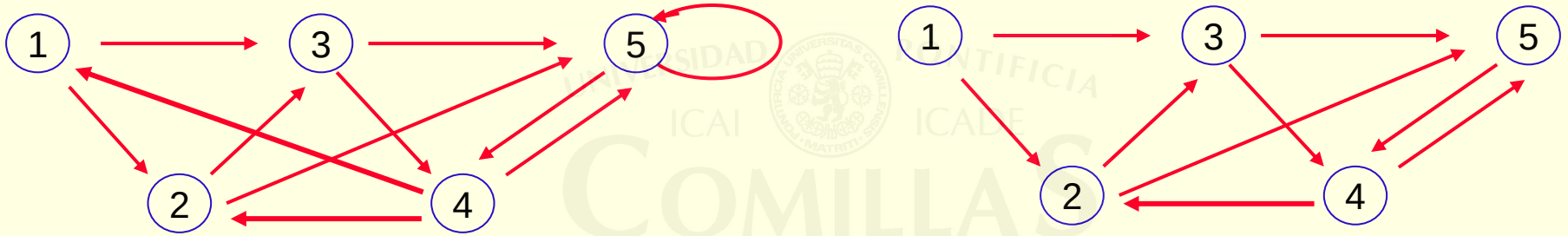
- **Dos arcos *(i, j)* y *(h, k)* se dice que son adyacentes si *j* = *h*, es decir cuando el extremo de uno de ellos es el origen del otro.**



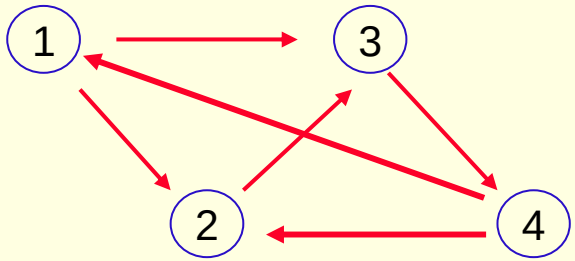
En la red de la figura los vértices 1 y 3 y los arcos (1 – 3) y (3 – 4) son adyacentes

Redes orientadas: Definiciones básicas (III)

- **Una red parcial es la obtenida de la inicial por supresión de al menos un arco de la misma.**

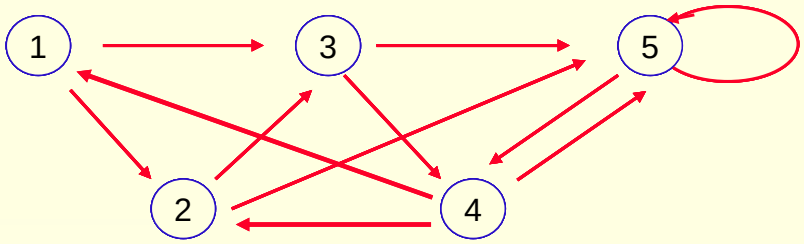


- **Una subred es la obtenida por eliminación de al menos un vértice y de todos los arcos que inciden sobre él.**



Redes orientadas: Definiciones básicas (IV)

❑ Camino



Una secuencia de arcos adyacentes.

Ejemplo: La secuencia de arcos

(1, 3) – (3, 4) – (4, 5)

El número de arcos que forman el camino es su **longitud**.

El camino anterior tiene longitud 3

Un camino es **simple** o **sencillo** si no utiliza dos veces el mismo arco. En otro caso se dice que es **compuesto**.

El camino anterior es simple. El camino (1, 3) – (3, 4) – (4, 1) – (1, 3) – (3, 5) es compuesto.

Un camino es **elemental** si no pasa dos veces por un mismo vértice.

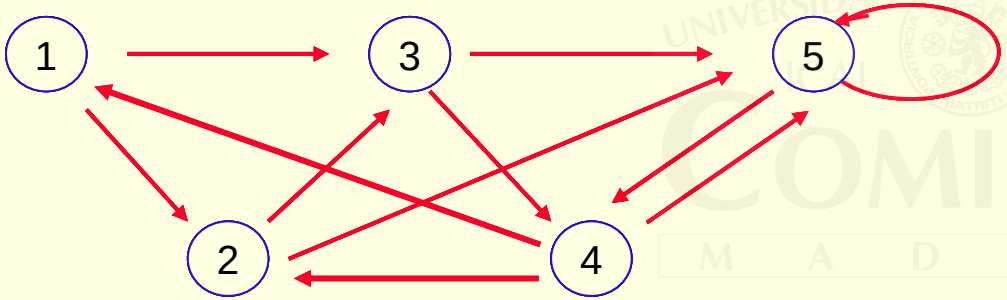
No es el caso del camino (1, 2) – (2, 3) – (3, 4) – (4, 2)

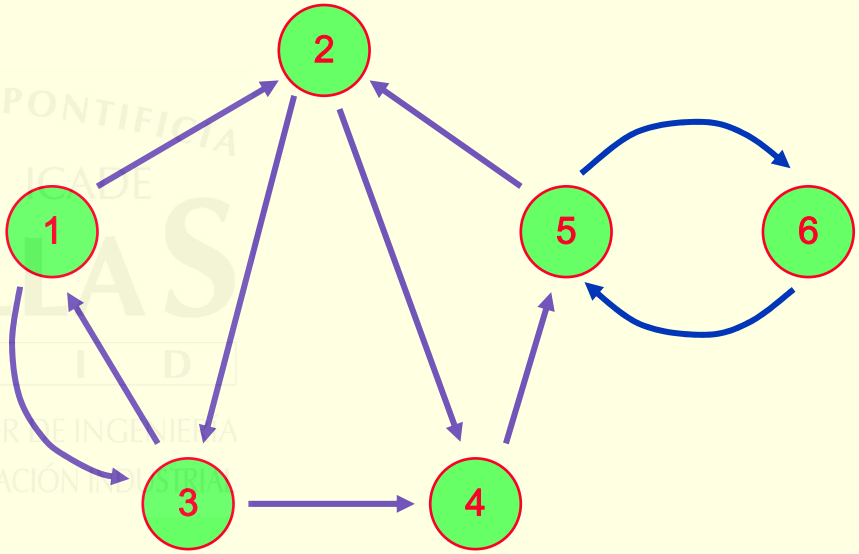
Dos vértices están **conectados** si al menos hay un camino entre ellos.

Convendremos en representar un camino por la secuencia de vértices que recorre.

Redes orientadas: Definiciones básicas (V)

- **Una red está *fuertemente conectada* o es *fuertemente conexa* si dos vértices cualesquiera están conectados. Una propiedad que debe satisfacer toda red de comunicaciones bien organizada.**

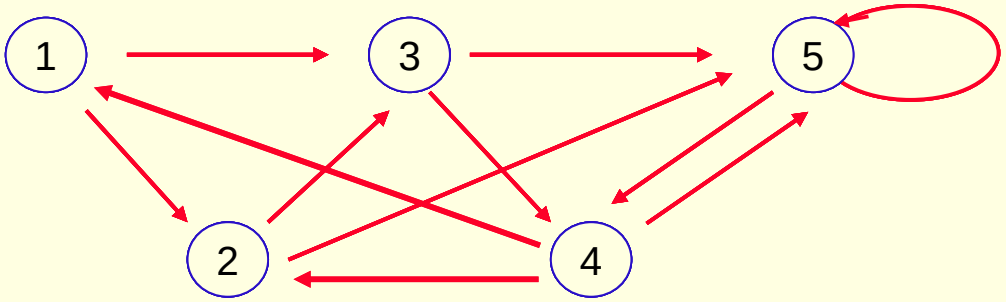




La redes de las figuras son fuertemente conectadas

Una componente fuertemente conexa es una subred fuertemente conexa tal que no existe otra subred que la contenga y que esté fuertemente conectada

Redes orientadas: Definiciones básicas (VI)



- **Un circuito es un camino de longitud finita cuyos vértices inicial y final coinciden.**
- **Un circuito elemental es el que, salvo u vértice inicial-final, no pasa dos veces por ningún otro.**
- **Un bucle o lazo es un circuito de longitud 1.**
- **Un camino o circuito es hamiltoniano si pasa una sola vez por todos los vértices de la red. Un camino es euleriano si pasa una sola vez por cada arco de la red.**

La secuencia 1-3-5-4-1 es un circuito

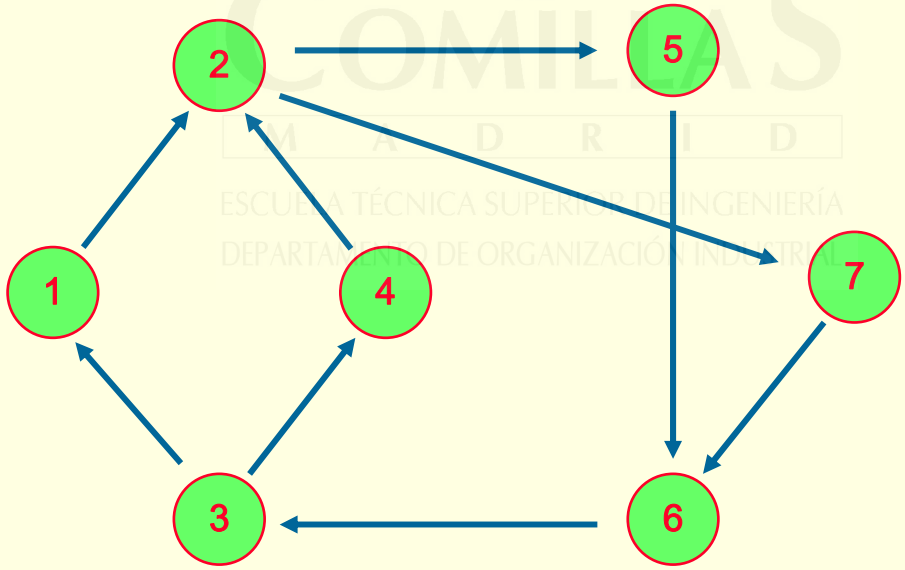
El circuito anterior es elemental

El arco 5-5 es un bucle

El circuito 1-2-3-5-4-1 es hamiltoniano. El camino 1-3-5-4-2 es hamiltoniano.

Redes orientadas: Definiciones básicas (VII)

- Una red *periódica* es una red fuertemente conectada en la que la longitud de todos sus circuitos es múltiplo de un número mayor que 1. Dicho número recibe el nombre de *periodo* de la red.
- Una red que tenga algún lazo es *aperiódica*.



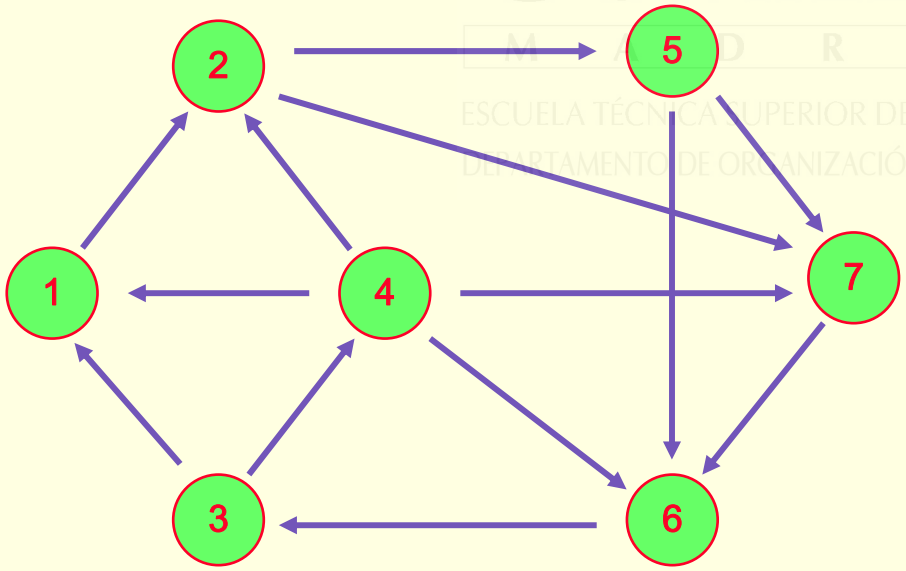
Una red periódica con periodo 5

Redes orientadas: Definiciones básicas (VIII)

- Dados *i y j* llamamos *desviación* entre ellos y designamos por *d(i,j)* a la menor de las longitudes de los caminos que los unen. Es obvio que puede no cumplirse que *d(i,j) = d(j,i).*
- La *separación* de un vértice *i* es

$$s(i) = \underset{j \in A}{Max}\left\{d(i,j)\right\}$$

conviniendo en que $d(i,i) = 0$ y $d(i,j) = \infty$ si *j* no es sucesor mediato o inmediato de *i*.



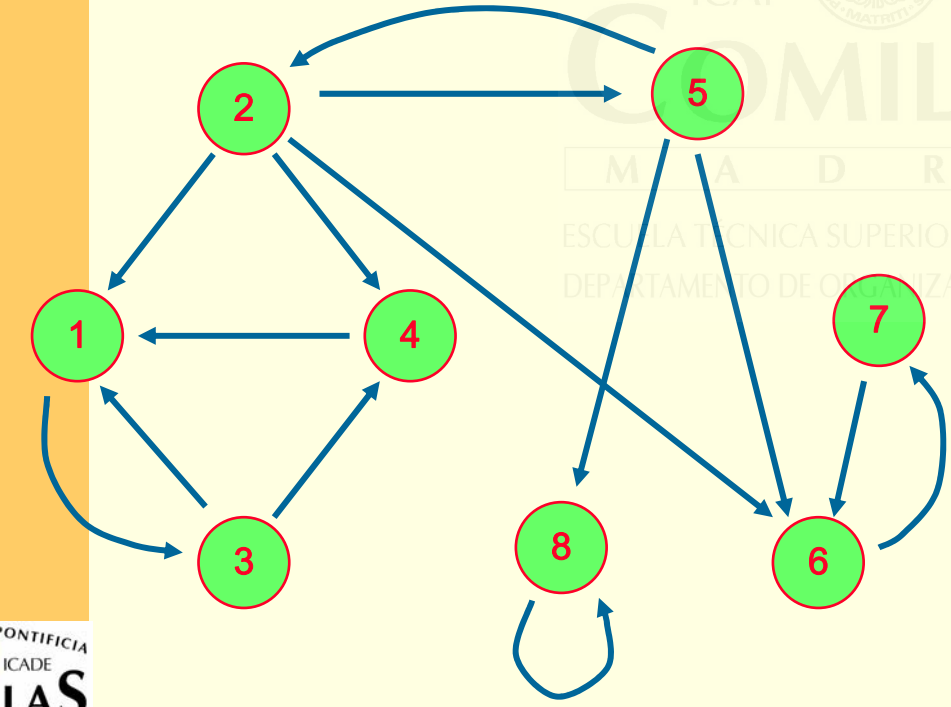
Centro de una red es un vértice al que corresponde una separación mínima. El vértice 4 en la red de la figura. Este valor mínimo es el *radio* de la red. Toda red completa tiene un centro.

Un vértice cuya separación sea máxima recibe el nombre de *punto periférico*. En la figura, el vértice 7. Esta máxima separación es el diámetro de la red: 5 en el ejemplo.

Redes orientadas: Definiciones básicas (IX)

Una subred *B* de una red *A* es una clase final si:

- ✓ **Existe al menos un vértice i no perteneciente a B que tiene al menos un sucesor directo en B**
- ✓ **Ningún vértice de B tiene sucesores que no pertenezcan a B.**
- ✓ **B es fuertemente conexa.**



Las subredes {1,3,4}, {6, 7} y {8} son clases finales de la red

Los nodos de B que son sucesores directos de nodos no pertenecientes a B reciben el nombre de sumideros: los vértices 1, 4, 6 y 8 son sumideros.

Una clase final constituida por un solo vértice recibe el nombre de vórtice. El nodo 8 es un vórtice.

Clasificación en niveles de los vértices de una red orientada sin circuitos (I)

En relación con muchos problemas de optimización que pueden ser modelados sobre una red orientada sin circuitos, es importante clasificarse sus vértices en clases o niveles de la siguiente forma:

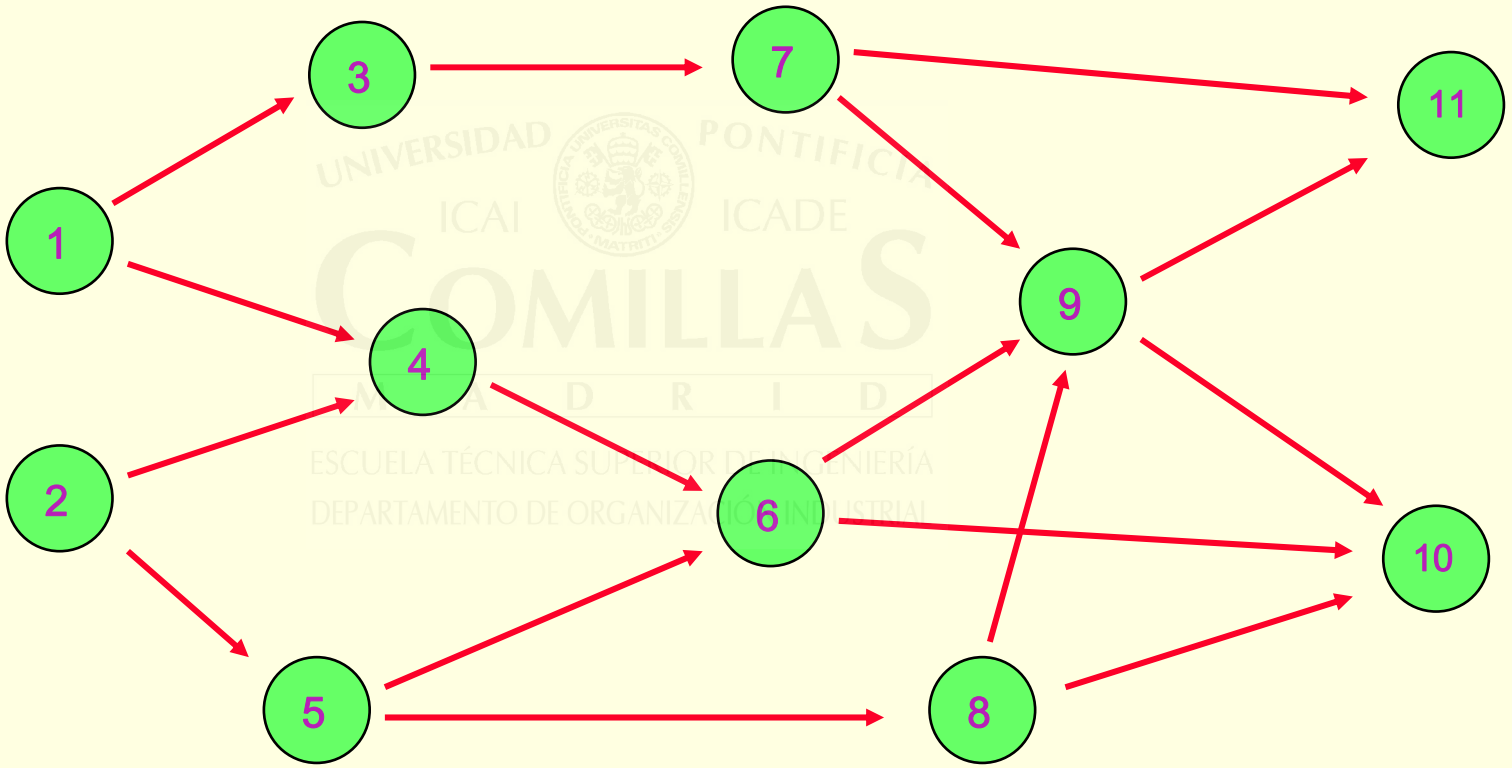
- En el nivel 1 se incluyen aquellos vértices que no tienen ascendientes directos.

- En el nivel k están aquellos vértices cuyos ascendientes directos están en niveles precedentes.

- El último nivel lo ocupan aquellos vértices que carecen de descendientes.

Clasificación en niveles de los vértices de una red orientada sin circuitos (II)

La siguiente red ilustra el mecanismo de clasificación



Clasificación en niveles de los vértices de una red orientada sin circuitos (III)

Matriz de adyacencia de la red

	1	2	3	4	5	6	7	8	9	10	11
1			1	1							
2				1	1						
3				1			1				
4						1					
5						1		1			
6									1	1	
7									1		1
8									1	1	
9										1	1
10											
11											

Clasificación en niveles de los vértices de una red orientada sin circuitos (IV)

	1	2	3	4	5	6	7	8	9	10	11
1			1	1							
2				1	1						
3				1			1				
4						1					
5						1		1			
6									1	1	
7									1		1
8									1	1	
9										1	1
10											
11											
S1	0	0	1	3	1	2	1	1	3	3	2

La nueva fila S0 se obtiene sumando todas las filas de la matriz de adyacencia.

Los ceros que aparecen en dicha fila corresponden a los nodos que carecen de ascendientes. Esos dos nodos, el 1 y el 2, integran el primer nivel.

$$N1 = \{1, 2\}$$

Clasificación en niveles de los vértices de una red orientada sin circuitos (V)

	1	2	3	4	5	6	7	8	9	10	11
1			1	1							
2				1	1						
3				1			1				
4						1					
5						1		1			
6									1	1	
7									1		1
8									1	1	
9										1	1
10											
11											
S1	0	0	1	3	1	2	1	1	3	3	2
S2	0	0	0	1	0	2	1	1	3	3	2

$$N2 = \{3, 5\}$$

Se añade una nueva fila S2, obtenida restando a S1 la suma de las filas asociadas a los vértices que constituyen el primer nivel.

$$S2 = S1 - C1 - C2$$

Los nuevos ceros que aparecen en S1 respecto a los que aparecían en S0 corresponden a los nodos que solo tienen ascendientes en los vértices de N1. Esos dos nodos, el 3 y el 5, integran el segundo nivel.

Clasificación en niveles de los vértices de una red orientada sin circuitos (VI)

	1	2	3	4	5	6	7	8	9	10	11
1			1	1							
2				1	1						
3				1			1				
4						1					
5						1		1			
6									1	1	
7									1		1
8									1	1	
9										1	1
10											
11											
S1	0	0	1	3	1	2	1	1	3	3	2
S2	0	0	0	1	0	2	1	1	3	3	2
S3	0	0	0	0	0	1	0	0	3	3	2

$$N3 = \{4, 7, 8\}$$

Una nueva fila, S3, es obtenida restando a S2 la suma de las filas asociadas a los vértices que constituyen el segundo nivel.

$$S3 = S2 - C3 - C5$$

Los nuevos ceros que aparecen en S3 respecto a los que aparecían en S2 corresponden a los nodos que solo tienen ascendientes en los vértices de N1 y N2. Hay tres nodos, los 4, 7 y 8, que integra el tercer nivel.

Clasificación en niveles de los vértices de una red orientada sin circuitos (VII)

	1	2	3	4	5	6	7	8	9	10	11
1			1	1							
2				1	1						
3				1			1				
4						1					
5						1		1			
6									1	1	
7									1		1
8									1	1	
9										1	1
10											
11											
S1	0	0	1	3	1	2	1	1	3	3	2
S2	0	0	0	1	0	2	1	1	3	3	2
S3	0	0	0	0	0	1	0	0	3	3	2
S4	0	0	0	0	0	0	0	0	1	2	1

Una nueva fila, S4, es obtenida restando a S3 la suma de las filas asociadas a los vértices que constituyen el tercer nivel.

$$S4 = S3 - C4 - C7 - C8$$

El nuevo cero que aparecen en S4 respecto a los que aparecían en S3 corresponden a los nodos que solo tienen ascendientes en los vértices de los niveles precedentes. Hay un sólo nodo, el 6, que integra el cuarto nivel.

$$N4 = \{6\}$$

Clasificación en niveles de los vértices de una red orientada sin circuitos (VIII)

	1	2	3	4	5	6	7	8	9	10	11
1			1	1							
2				1	1						
3				1			1				
4						1					
5						1		1			
6									1	1	
7									1		1
8									1	1	
9										1	1
10											
11											
S1	0	0	1	3	1	2	1	1	3	3	2
S2	0	0	0	1	0	2	1	1	3	3	2
S3	0	0	0	0	0	1	0	0	3	3	2
S4	0	0	0	0	0	0	0	0	1	2	1
S5	0	0	0	0	0	0	0	0	0	1	1

Una nueva fila, S5, es obtenida restando a S4 la suma de las filas asociadas a los vértices que constituyen el cuarto nivel.

$$S5 = S4 - C6$$

El nuevo cero que aparecen en S5 respecto a los que aparecían en S4 corresponden a los nodos que solo tienen ascendientes en los vértices de los niveles precedentes. Hay un sólo nodo, el 9, que integra el quinto nivel.

$$N5 = \{9\}$$

Clasificación en niveles de los vértices de una red orientada sin circuitos (IX)

	1	2	3	4	5	6	7	8	9	10	11
1			1	1							
2				1	1						
3				1			1				
4						1					
5						1		1			
6									1	1	
7									1		1
8									1	1	
9										1	1
10											
11											
S1	0	0	1	3	1	2	1	1	3	3	2
S2	0	0	0	1	0	2	1	1	3	3	2
S3	0	0	0	0	0	1	0	0	3	3	2
S4	0	0	0	0	0	0	0	0	1	2	1
S5	0	0	0	0	0	0	0	0	0	1	1
S6	0	0	0	0	0	0	0	0	0	0	0

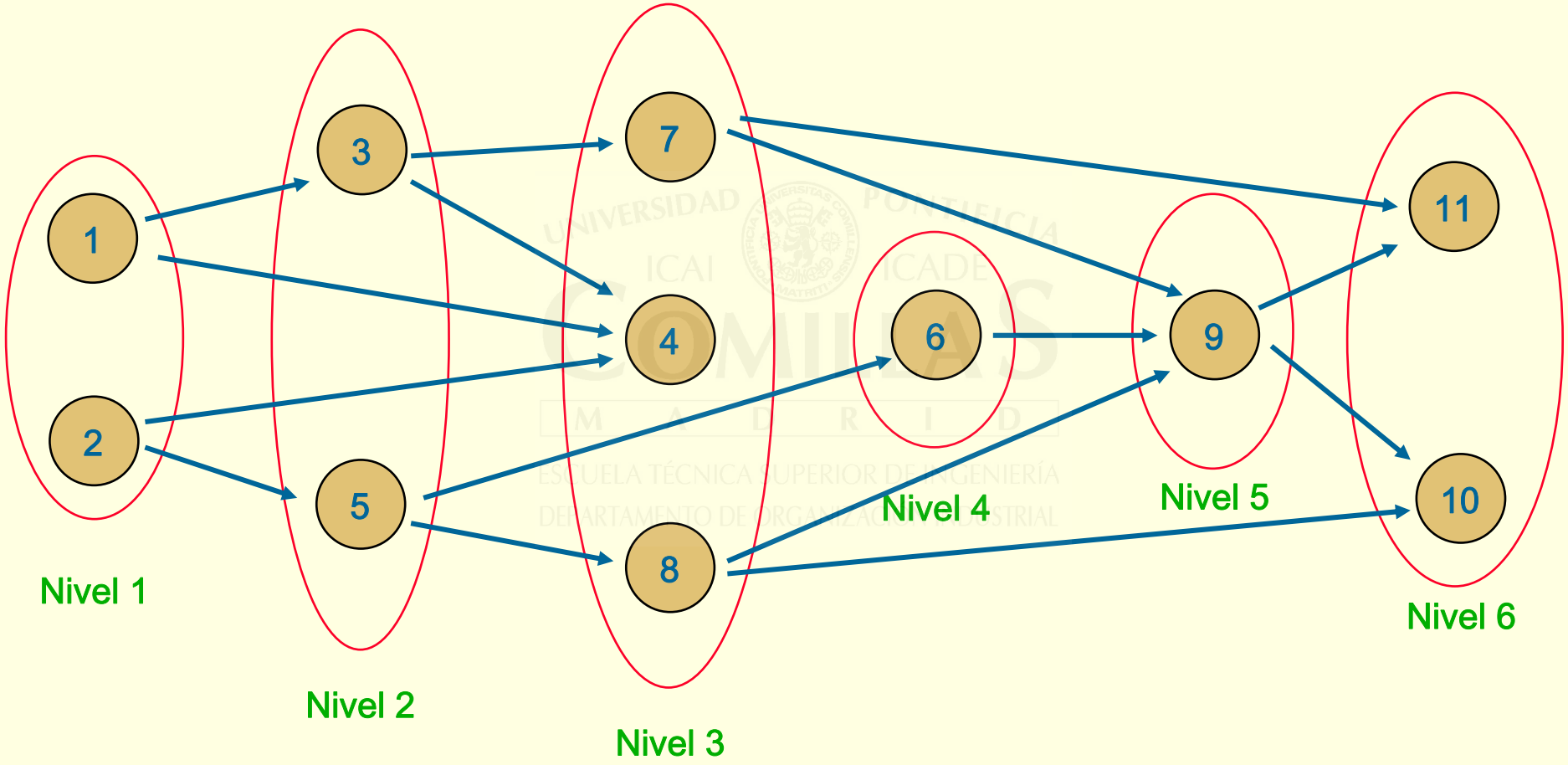
Una nueva fila, S6, es obtenida restando a S5 la suma de las filas asociadas a los vértices que constituyen el quinto nivel.

$$S6 = S5 - C9$$

Los nuevos ceros que aparecen en S6 respecto a los que aparecían en S5 corresponden a los nodos que solo tienen ascendientes en los vértices de los niveles precedentes. Hay dos nodos, el 10 y el 11, que integran el sexto y último nivel, formado por los vértices que carecen de descendientes.

$$N6 = \{10, 11\}$$

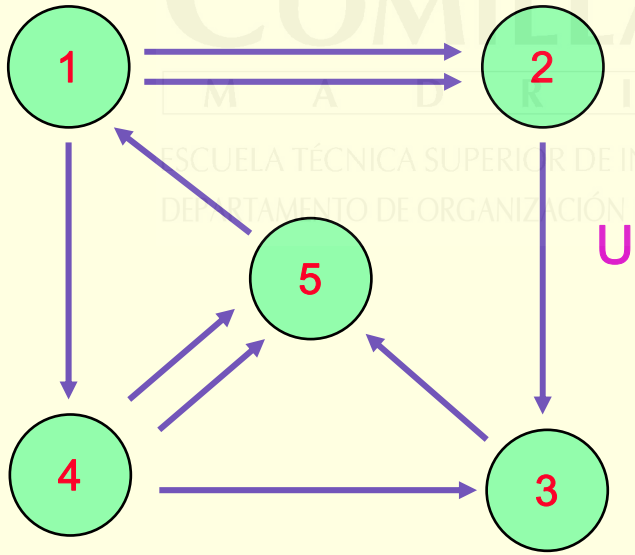
Clasificación en niveles de los vértices de una red orientada sin circuitos (X)



Multirredes ó redes múltiples

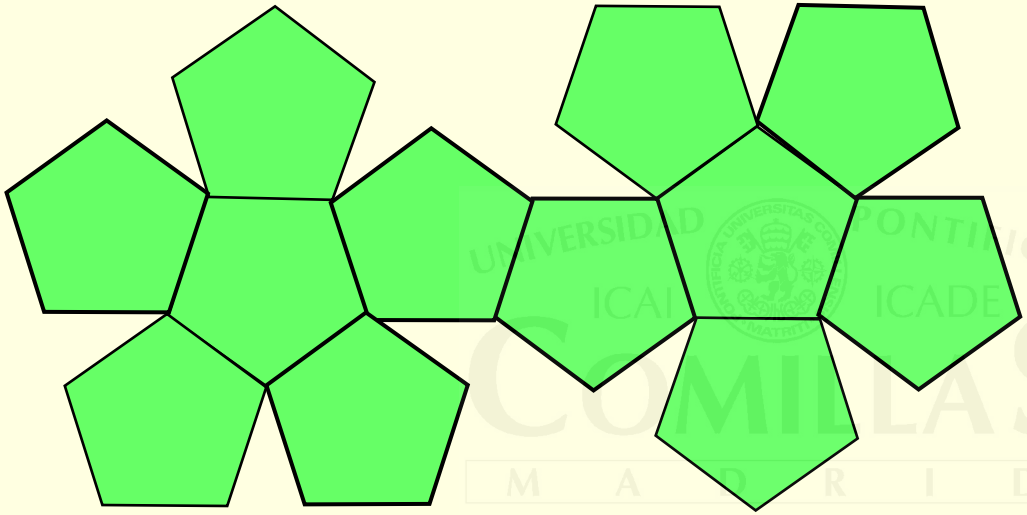
- **Son aquellas en las que puede haber más de un arco con el mismo origen y extremo.**

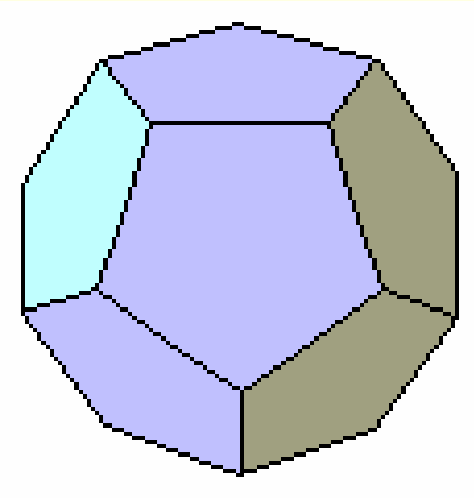
- **Un digrafo es una multirred en la que el número máximo de arcos con el mismo origen y extremo dados es 2.**



Un digrafo

Dodecadro regular





Dodecaedro:

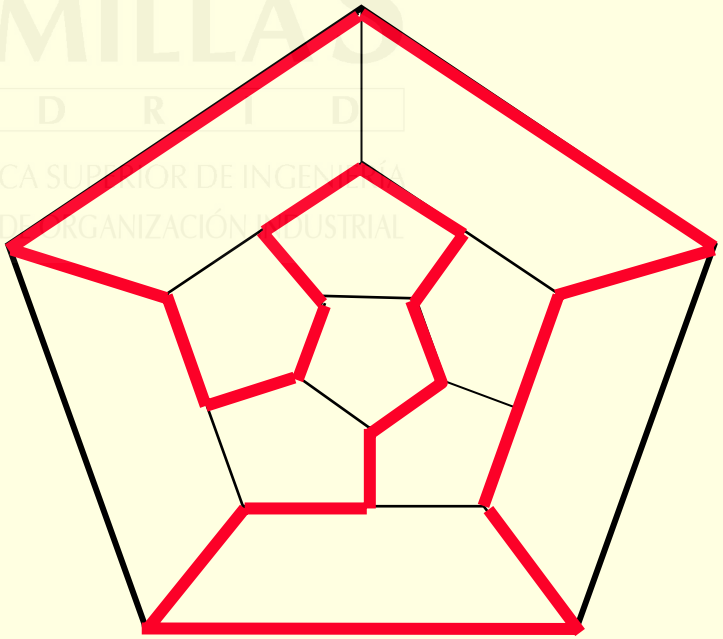
- 12 caras pentágonos regulares
- 30 aristas
- 20 vértices

Recorridos hamiltonianos (I)

- Un camino o circuito es hamiltoniano si pasa una sola vez por todos los vértices de la red. Un camino es euleriano si pasa una sola vez por cada arco de la red.

William Hamilton, 1859:

Recorrer los vértices de un dodecaedro pasando por ellos una y sólo una vez.



ESCUELA TÉCNICA SUPERIOR DE INGENIERÍA
DEPARTAMENTO DE ORGANIZACIÓN INDUSTRIAL

Teoría de Grafos o Redes

Determinación de circuitos, cadenas y componentes fuertemente conexas

ESCUELA TÉCNICA SUPERIOR DE INGENIERÍA
DEPARTAMENTO DE ORGANIZACIÓN INDUSTRIAL

Teoría de Grafos o Redes

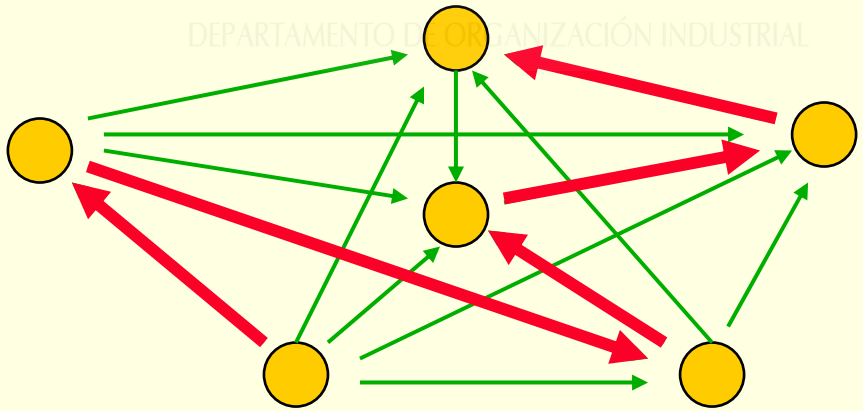
Determinación de circuitos hamiltonianos

Circuitos hamiltonianos (I)

No hay teoremas que que proporcionen condiciones necesarias y suficientes para la existencia de caminos y circuitos hamiltonianos.

Teorema de Köening:

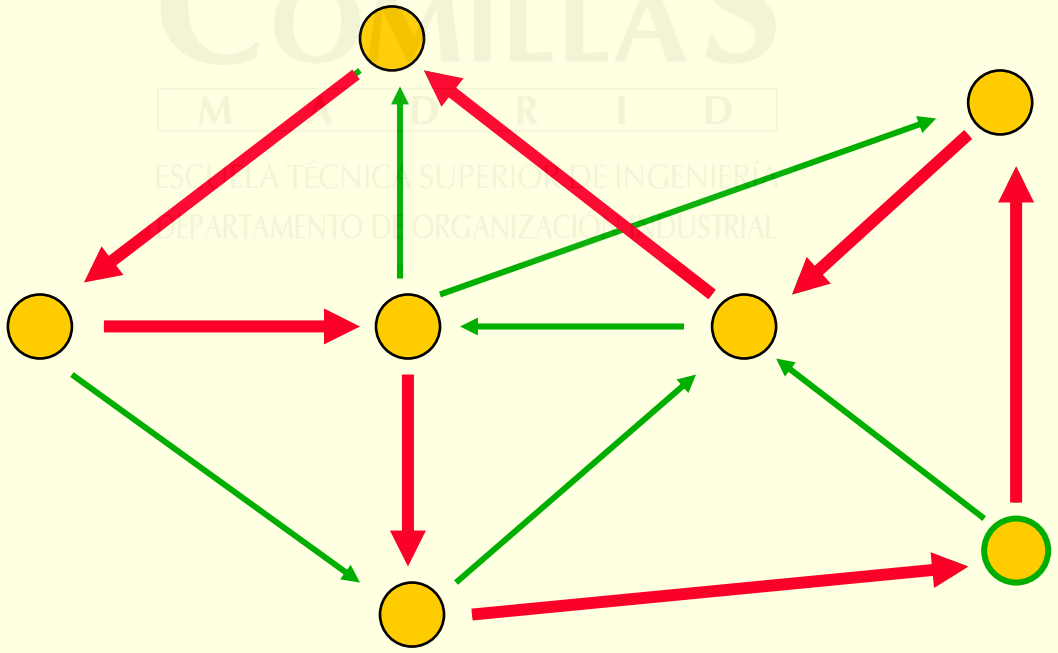
Si una red es completa existe al menos un camino hamiltoniano



Circuitos hamiltonianos (II)

Teorema de Dirac:

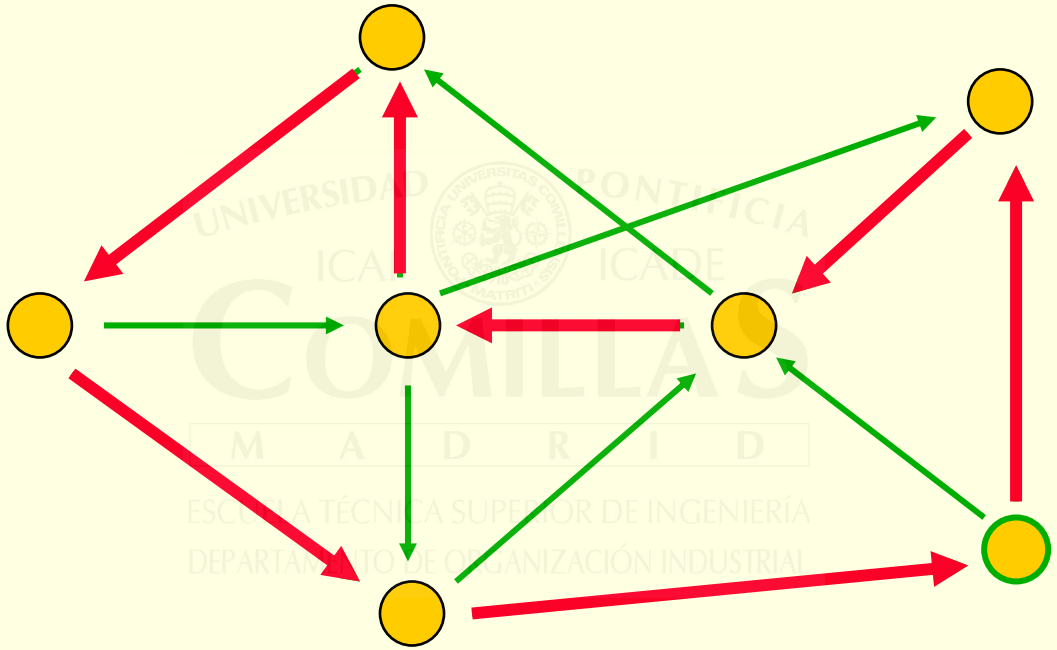
Si una red es simétrica, fuertemente conexa y sin bucles y para todo nodo se cumple que el número de sus descendientes es mayor o igual que la mitad del número de nodos tiene al menos un circuito hamiltoniano.



Esta red no cumple el teorema pero tiene más de un circuito hamiltoniano

Circuitos hamiltonianos (III)

Otro circuito en la red anterior



La multiplicación latina: un algoritmo para obtención de circuitos hamiltonianos (I)

- Llamado de multiplicación latina, debido a Kaufmann y Malgrange.
- Sirve para grafos y digrafos.
- Fácilmente adaptable para buscar el circuito hamiltoniano más corto.
- Bastante ineficiente.
- Idea básica: Si A es la matriz de adyacencia, el elemento (i,j) de A^k es el número de caminos de longitud k que van de i a j
  - ✓ k=n-1 : caminos hamiltonianos
  - ✓ k=n : circuitos hamiltonianos

La multiplicación latina: un algoritmo para obtención de circuitos hamiltonianos (II)

PASO 1: Construir una matriz M_1 obtenida de la matriz de adyacencia reemplazando cada elemento distinto de 0 por la hilera ij, salvo en la diagonal.

PASO 2: Construir M eliminando la primera letra de cada hilera.

PASO 3: Calcular M_j, para todo j entre 1 y n

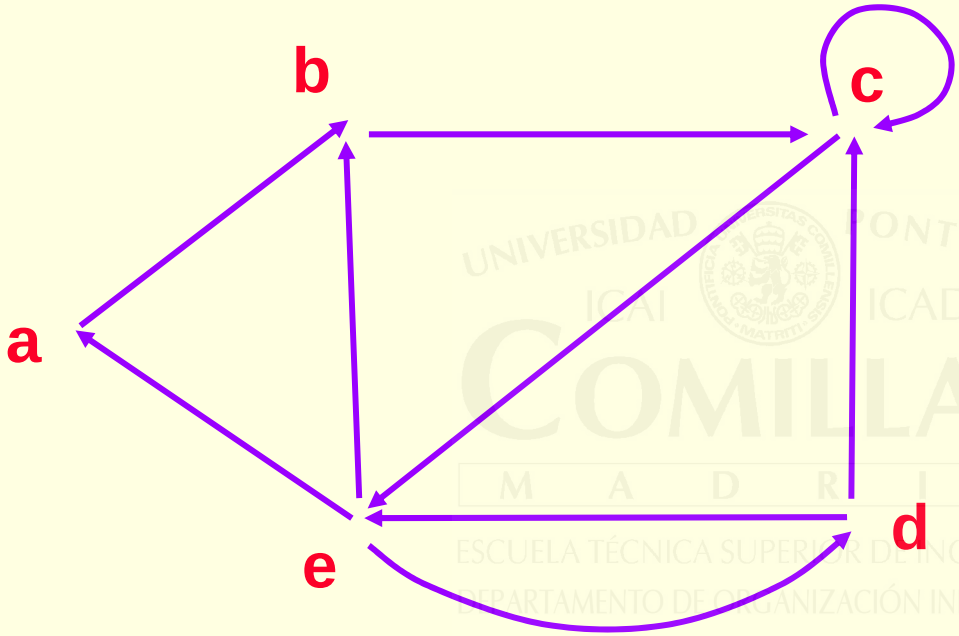
$$M_j(r,s) = M_{j-1}(r,t)M(t,s)$$

donde ningún elemento es cero ni tiene vértices comunes (en ese caso, se hace cero)

(es el conjunto de caminos de r a s usando j aristas)

PASO 4: Para encontrar los circuitos hamiltonianos basta ver en cuales caminos de los caminos obtenidos pueden conectarse los vértices inicial y final.

Ejemplo (I)



$$A \equiv \begin{array}{c} \\ a \\ b \\ c \\ d \\ e \end{array} \begin{array}{c} \begin{array}{ccccc} a & b & c & d & e \end{array} \\ \begin{bmatrix} 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & 0 & 1 \\ 1 & 1 & 0 & 1 & 0 \end{bmatrix} \end{array}$$

Ejemplo (II)

$$A \equiv \begin{array}{c} \\ a \\ b \\ c \\ d \\ e \end{array} \begin{array}{c} \begin{array}{ccccc} a & b & c & d & e \end{array} \\ \begin{bmatrix} 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & 0 & 1 \\ 1 & 1 & 0 & 1 & 0 \end{bmatrix} \end{array} \Rightarrow M_1 \equiv \begin{array}{c} \\ a \\ b \\ c \\ d \\ e \end{array} \begin{array}{c} \begin{array}{ccccc} a & b & c & d & e \end{array} \\ \begin{bmatrix} 0 & ab & 0 & 0 & 0 \\ 0 & 0 & bc & 0 & 0 \\ 0 & 0 & 0 & cd & ce \\ 0 & 0 & dc & 0 & de \\ ea & eb & 0 & ed & 0 \end{bmatrix} \end{array}$$

$$M \equiv \begin{array}{c} \\ a \\ b \\ c \\ d \\ e \end{array} \begin{array}{c} \begin{array}{ccccc} a & b & c & d & e \end{array} \\ \begin{bmatrix} 0 & b & 0 & 0 & 0 \\ 0 & 0 & c & 0 & 0 \\ 0 & 0 & 0 & d & e \\ 0 & 0 & c & 0 & e \\ a & b & 0 & d & 0 \end{bmatrix} \end{array}$$

Ejemplo (III)

$$M_2 = M_1 \times M \equiv \begin{bmatrix} 0 & ab & 0 & 0 & 0 \\ 0 & 0 & bc & 0 & 0 \\ 0 & 0 & 0 & cd & ce \\ 0 & 0 & dc & 0 & de \\ ea & eb & 0 & ed & 0 \end{bmatrix} \times \begin{bmatrix} 0 & b & 0 & 0 & 0 \\ 0 & 0 & c & 0 & 0 \\ 0 & 0 & 0 & d & e \\ 0 & 0 & c & 0 & e \\ a & b & 0 & d & 0 \end{bmatrix} =$$

$$= \begin{bmatrix} 0 & 0 & abc & 0 & 0 \\ 0 & 0 & 0 & bcd & bce \\ cea & ceb & 0 & ced & cde \\ dea & deb & 0 & 0 & dce \\ 0 & eab & ebc & 0 & 0 \end{bmatrix}$$

Ejemplo (IV)

$$M_3 = M_2 \times M \equiv \begin{bmatrix} 0 & 0 & abc & 0 & 0 \\ 0 & 0 & 0 & bcd & bce \\ cea & ceb & 0 & ced & cde \\ dea & deb & 0 & 0 & dce \\ 0 & eab & 0 & ebd & 0 \end{bmatrix} \times \begin{bmatrix} 0 & b & 0 & 0 & 0 \\ 0 & 0 & c & 0 & 0 \\ 0 & 0 & 0 & d & e \\ 0 & 0 & c & 0 & e \\ a & b & 0 & d & 0 \end{bmatrix} =$$

$$= \begin{bmatrix} 0 & 0 & 0 & abcd & abce \\ bcea & 0 & 0 & bced & bcde \\ cdea & \begin{matrix} ceab \\ cdeb \end{matrix} & 0 & 0 & 0 \\ dcea & \begin{matrix} deab \\ dceb \end{matrix} & deac & 0 & 0 \\ 0 & 0 & \begin{matrix} eabc \\ ebdc \end{matrix} & 0 & 0 \end{bmatrix}$$

Ejemplo (V)

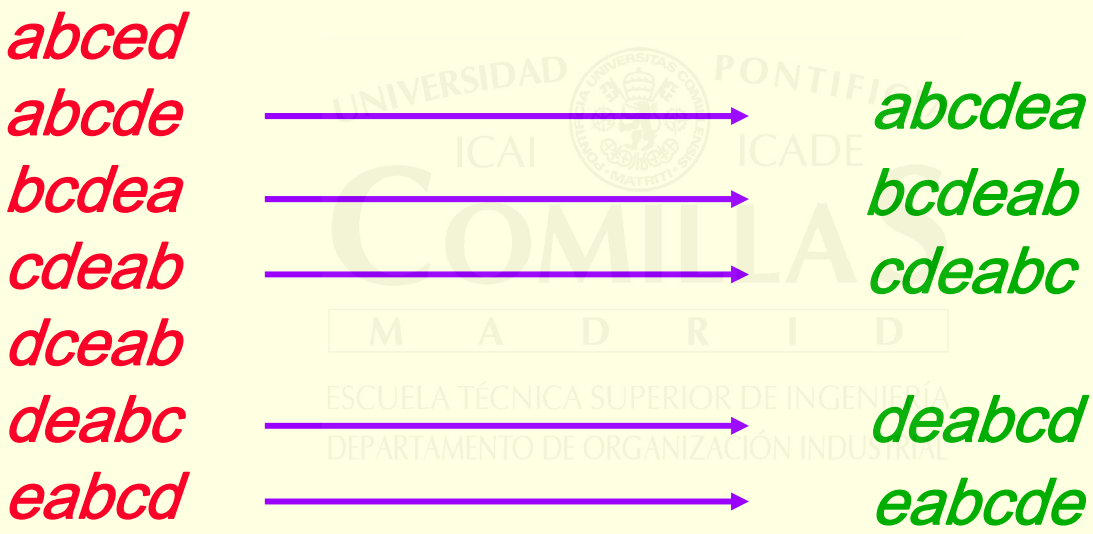
$$M_4 = M_3 \times M \equiv \begin{bmatrix} 0 & 0 & 0 & abcd & abce \\ bcea & 0 & 0 & bced & bcde \\ cdea & ceab & 0 & 0 & 0 \\ & cdeb & & & \\ dcea & deab & deac & 0 & 0 \\ & dceb & & & \\ 0 & 0 & eabc & 0 & 0 \\ & & ebdc & & \end{bmatrix} \times \begin{bmatrix} 0 & b & 0 & 0 & 0 \\ 0 & 0 & c & 0 & 0 \\ 0 & 0 & 0 & d & e \\ 0 & 0 & c & 0 & e \\ a & b & 0 & d & 0 \end{bmatrix} =$$

$$= \begin{bmatrix} 0 & 0 & 0 & abced & abcde \\ bcdea & 0 & 0 & 0 & 0 \\ 0 & cdeab & 0 & 0 & 0 \\ 0 & dceab & deabc & 0 & 0 \\ 0 & 0 & 0 & eabcd & 0 \end{bmatrix}$$

Esta matriz muestra todos los caminos hamiltonianos de la red

Ejemplo (VI)

Algunos caminos
pueden convertirse en
circuitos hamiltonianos



Obtenido un circuito hamiltoniano, por una permutación circular de sus elementos, se obtienen otros n-1 circuitos.

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

Teoría de Grafos o Redes

Obtención de circuitos eulerianos

Circuitos eulerianos

- Un camino es euleriano si pasa una sola vez por cada arco de la red.
- Si el camino comienza y acaba en el mismo vértice tenemos un circuito euleriano.

- Teorema

  La condición necesaria y suficiente para que una multirred tenga un circuito euleriano es que sea conexa y que de cada vértice salgan tantos arcos como llegan a él.

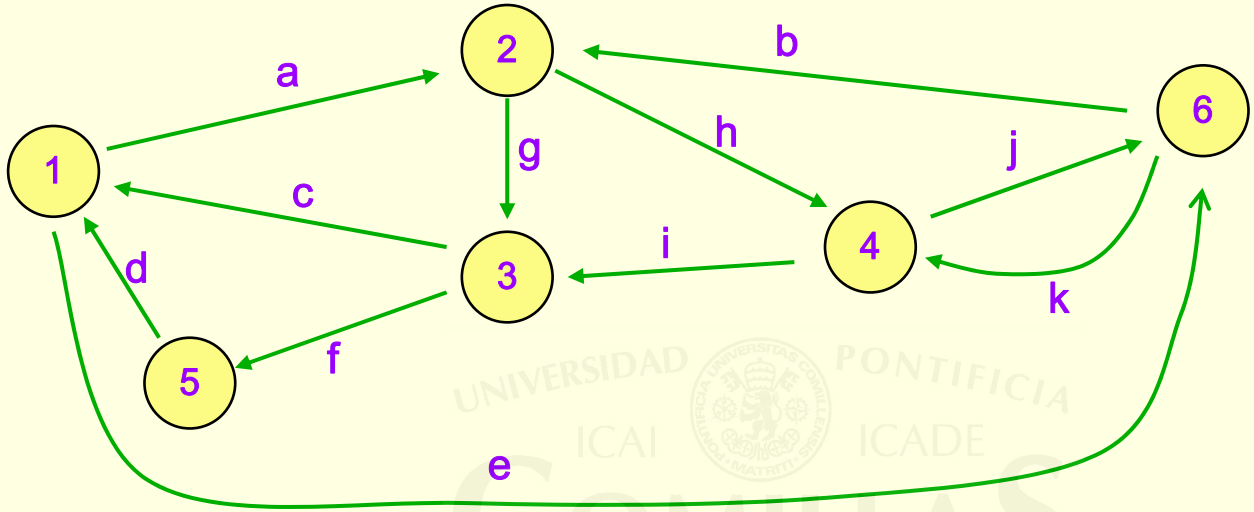
Obtención de circuitos eulerianos

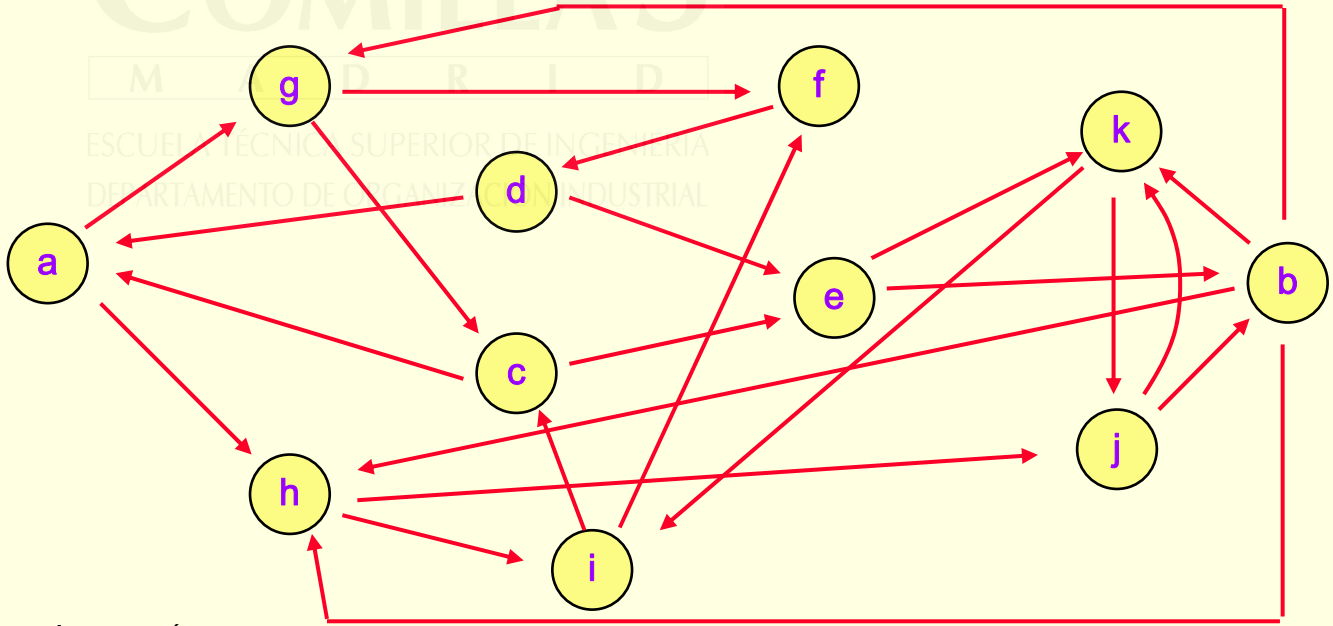
- Se substituye la red dada por otra en la que cada vértice represente un arco de la red original.

- Si en la red original el arco *u* precede al arco *v*, en la segunda red trazamos un arco desde el vértice asociado a *u* con el vértice asociado a *v*.

- Una vez establecida la segunda red obtenemos sus caminos hamiltonianos por alguno de los métodos descritos al efecto.

Obtención de circuitos eulerianos





ESCUELA TÉCNICA SUPERIOR DE INGENIERÍA
DEPARTAMENTO DE ORGANIZACIÓN INDUSTRIAL

Teoría de Grafos o Redes

Obtención de componentes fuertemente conexas

Obtención de componentes fuertemente conexas de una red

❑ Algoritmo del + y del -

Paso 1. Elegir un vértice cualquiera y marcarlo con + y –

Paso 2. Marcar con signo – a los predecesores inmediatos y con signo + a los sucesores inmediatos del vértice seleccionado.

Paso 3. Marcar con signo – a los predecesores inmediatos de los vértices marcados con signo – y con signo + a los sucesores inmediatos de los vértices marcados con signo +.

Paso 4. Reiterar el paso 3 mientras sea posible una nueva marca. Hecho esto, los nodos marcados simultáneamente con signo + y – forman una componente fuertemente conexa.

Paso 5. Eliminamos del grafo la componente obtenida y se reitera el proceso desde el paso 1 con el resto de la red.

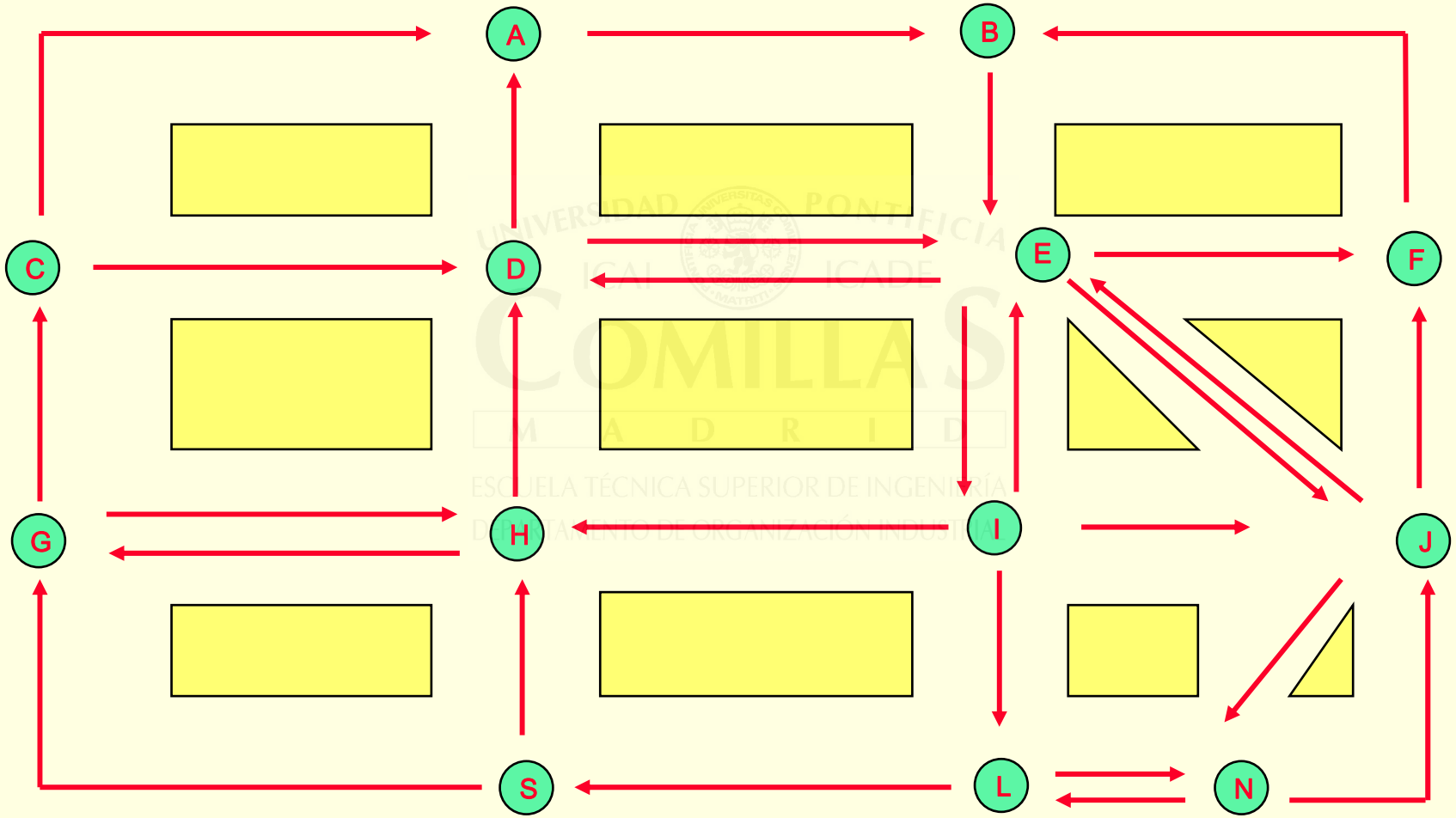
Obtención de componentes fuertemente conexas de una red

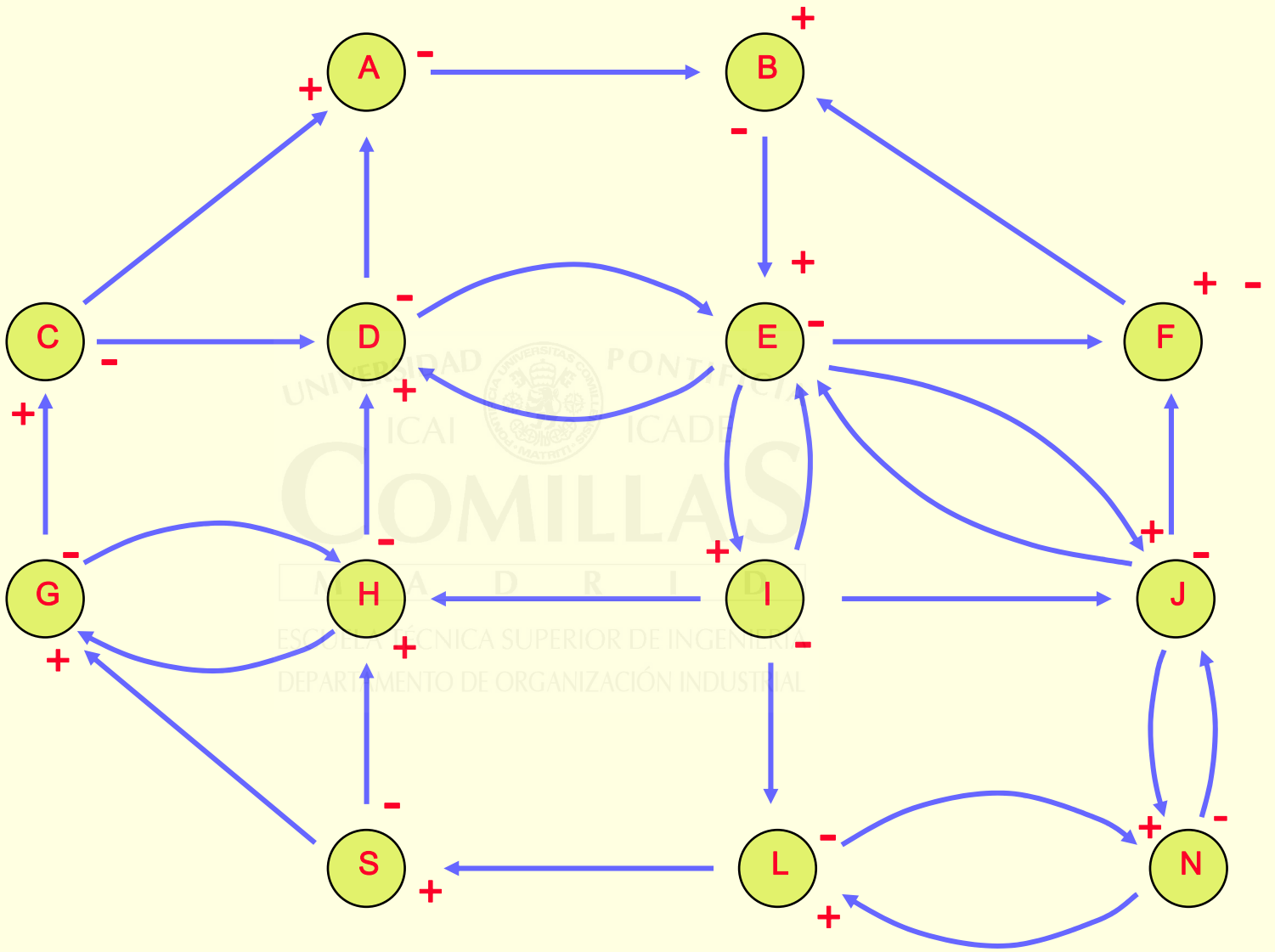
- Es un problema importante desde el punto de vista de muchos problemas reales, tales como el correcto direccionamiento de las calles de una ciudad a fin de que todos los puntos de la misma sean accesibles al tráfico.

- El siguiente ejemplo muestra una de estas situaciones.

Obtención de componentes fuertemente conexas de una red.

¿Es correcta la distribución de direcciones y sentidos de tráfico de la figura?




A
B
C
D
E
F
G
H
I
J
S
L
N

ESCUELA TÉCNICA SUPERIOR DE INGENIERÍA
DEPARTAMENTO DE ORGANIZACIÓN INDUSTRIAL

Teoría de Grafos o Redes

REDES NO ORIENTADAS

Redes no orientadas: Definiciones básicas (I)

- Dos vértice i y j se dice que están ligados por una **arista** si

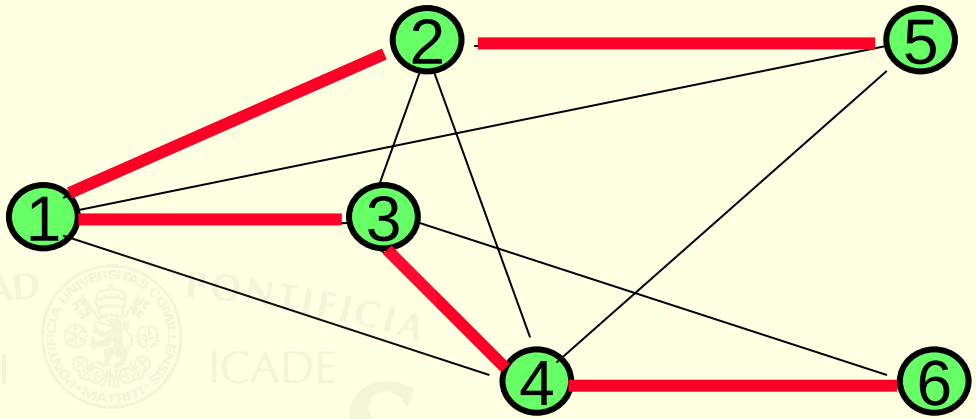
$$(i, j) \in R \text{ o } (j, i) \in R$$

o ambas condiciones a la vez

- **Matriz de adyacencia $A(a_{ij})$ de una red no orientada (es simétrica)**
  - ✓ $a_{ij} = 1$ si existe arista entre *i* y *j*
  - ✓ $a_{ij} = 0$ en otro caso

Redes no orientadas: Definiciones básicas (II)

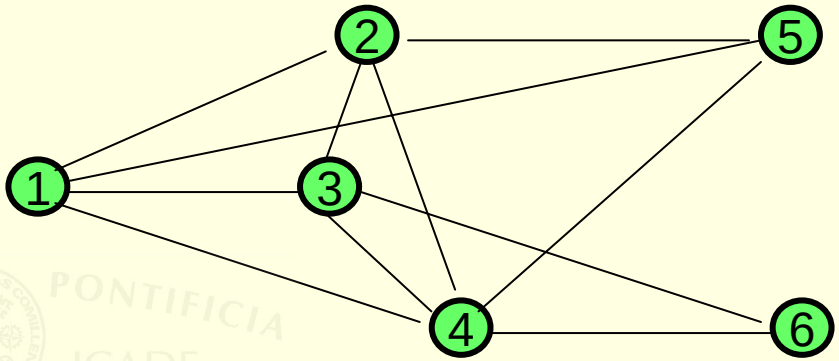
- Una **cadena** es una sucesión de aristas cada una de ellas unida a la anterior y a la siguiente por cada una de sus extremidades, excepto las aristas "comienzo" y "final" de la misma



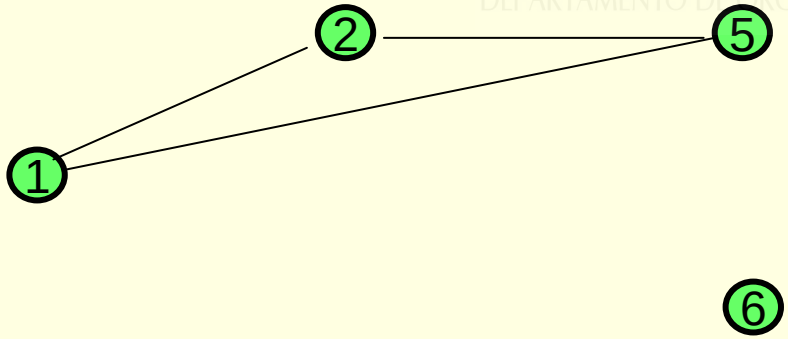
- Si todas las aristas de la cadena son distintas se dice que la cadena es **sencilla** (el ejemplo mostrado) y en caso contrario **compuesta**. El conjunto de vértices que pueden conectarse entre sí por medio de una cadena se dice que es **una componente conexa** de la red. El número de aristas que conforman una cadena es su **longitud**.

Redes no orientadas: Definiciones básicas (III)

- Una red es **conexa** si dos vértices cualesquiera están unidos por una cadena. Un subconjunto B de vértices se dice que es un **subconjunto de articulación** si la subred que resulta de eliminar los vértices de B no es conexa. Si B se reduce a un solo vértice, dicho vértice recibe el nombre de **punto de articulación** o **punto de corte**. Una red conexa tiene una sola componente conexa.



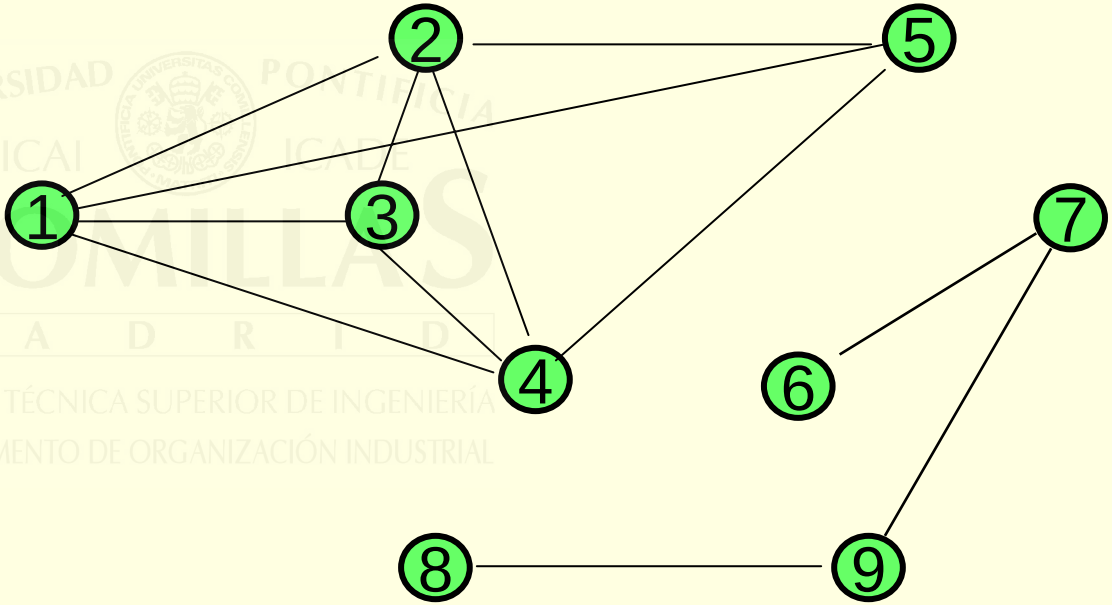
Una red conexa



Los vértices 3 y 4 son un subconjunto de articulación de esta red. Su supresión genera una red con dos componentes conexas: la formada por los vértices 1,2 y 5 y la constituida por el vértice 6

Redes no orientadas: Definiciones básicas (IV)

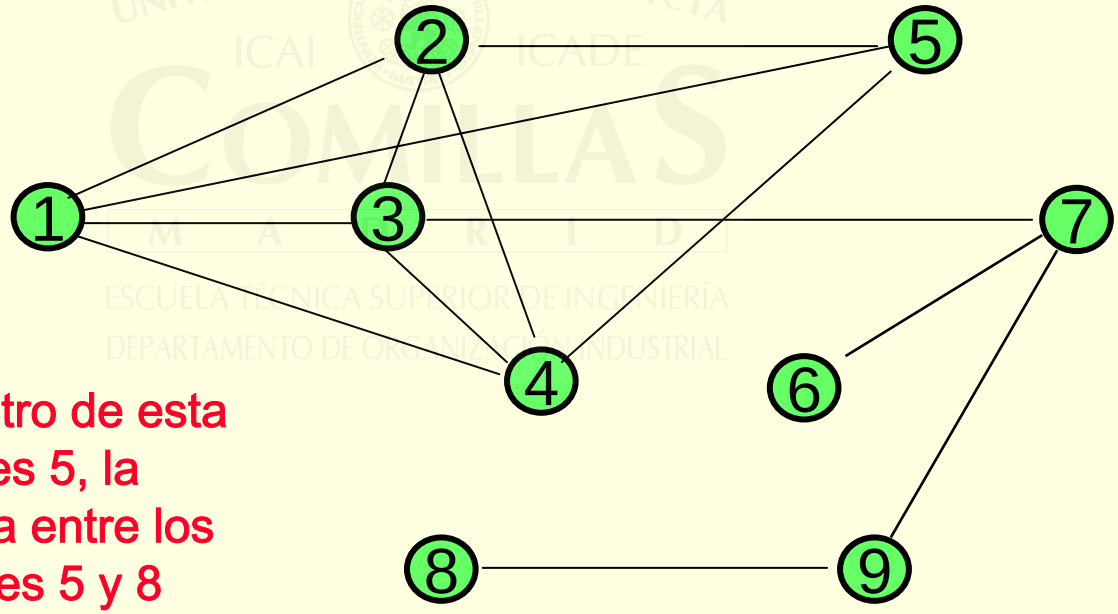
- Un **istmo o puente** es una arista cuya supresión incrementa el número de componentes conexas



- La arista 3-7 de la figura es un puente. Su supresión da lugar a dos componentes conexas.

Redes no orientadas: Definiciones básicas (V)

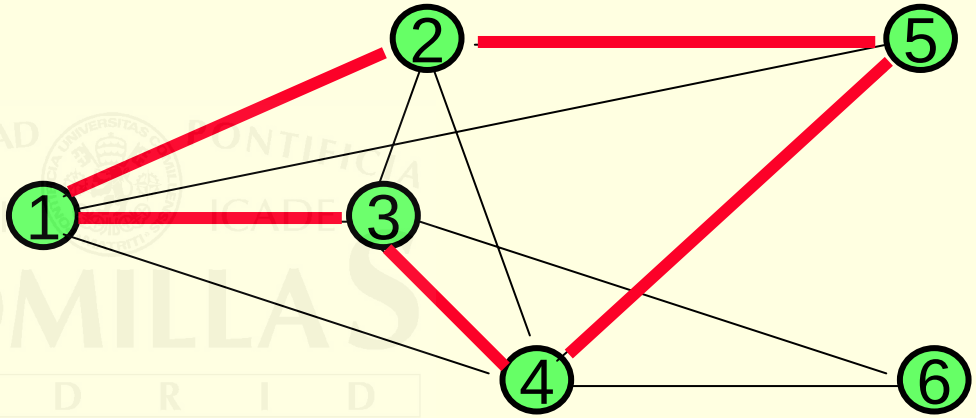
En una red conexa se da el nombre de *distancia* entre los vértices i y j a la menor de las longitudes de las cadenas que los unen y diámetro de la red al máximo de las distancias entre sus vértices.



El diámetro de esta red es 5, la distancia entre los vértices 5 y 8

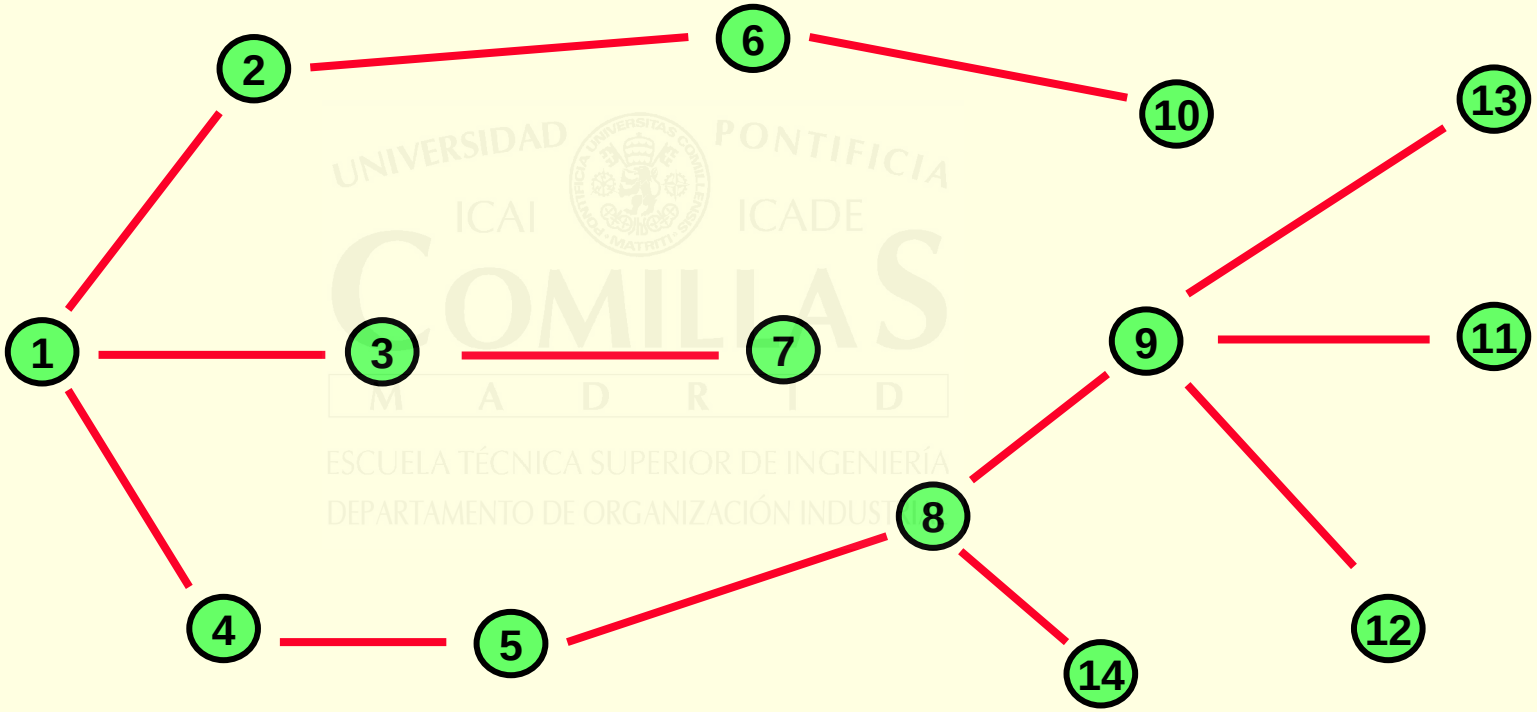
Redes no orientadas: Definiciones básicas (VI)

- Un ciclo es una cadena finita que une un nodo con el mismo.
- Un ciclo es elemental si, salvo el vértice inicial-final, el ciclo pasa una sola vez por cada vértice que lo constituye (el caso del ejemplo mostrado).
- Una cadena es euleriana si recorre, sin repetirla ninguna, toda las aristas de la red.
- Un ciclo es euleriano si recorre, sin repetirlas, todas las aristas de la red, acabando en el vértice de partida



Redes no orientadas: Definiciones básicas (VII)

- **Un árbol es un grafo finito, conexo, sin ciclos y con al menos dos nodos.**

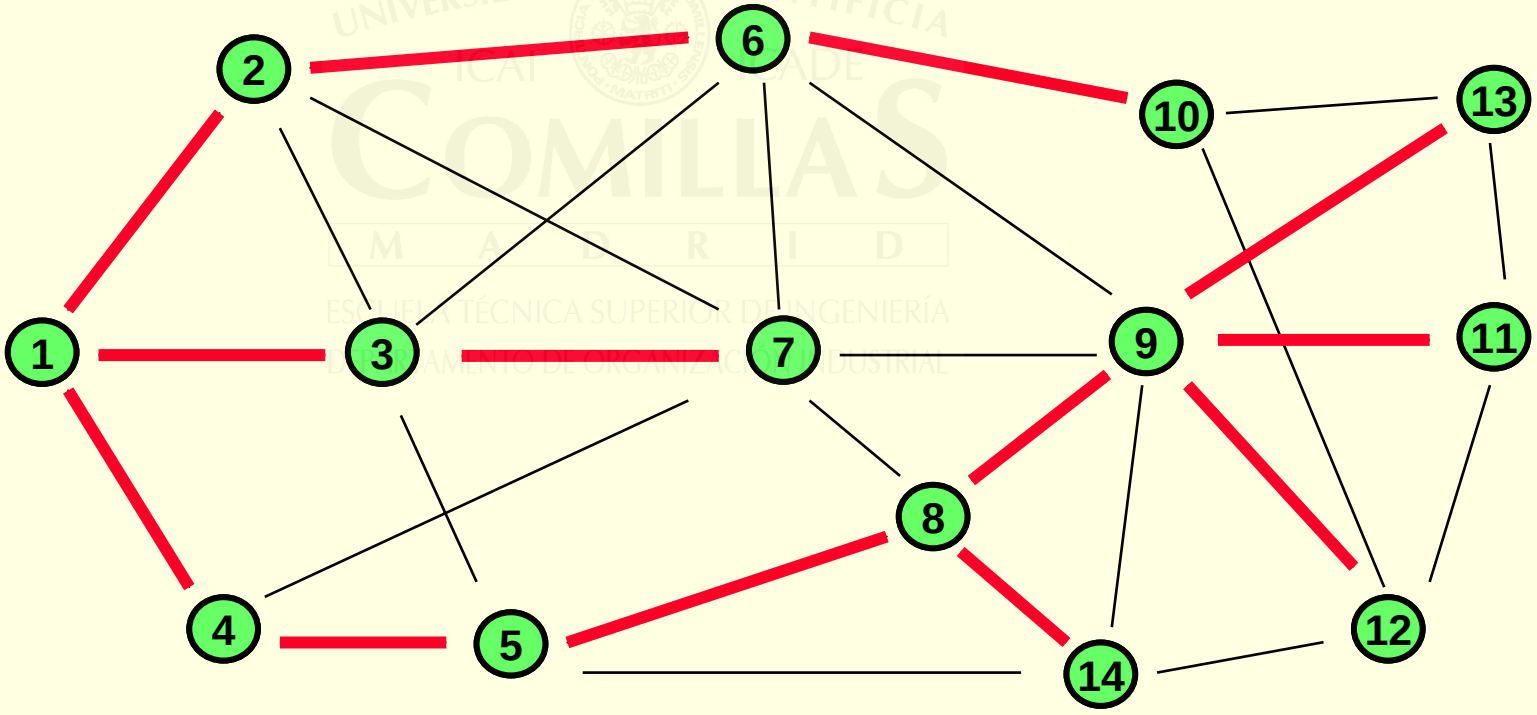


Si un árbol tiene n vértices el número de aristas es n-1

Todo par de vértices está unido por una única cadena

Redes no orientadas: Definiciones básicas (VIII)

- **Arbol generador (o de extensión)** (spanning tree) de un grafo es un árbol que incluye a todos los vértices del mismo



Obtención de cadenas y ciclos eulerianos

- **Cadena euleriana:**
  - ✓ **Una cadena que pasa por todos los vértices y atraviesa exactamente una vez cada arista**
- **Ciclo euleriano:**
  - ✓ **Una cadena euleriana que pasa por todos los vértices, volviendo al de partida.**

- **Teorema (Euler, 1766)**

La condición necesaria y suficiente para que una red, simple o múltiple, tenga una cadena euleriana es que sea conexa y que el número de vértices de grado impar sea 0 ó 2.

Si dicho número es 0 hay un ciclo eulearíano.

Si ese número es 2 hay una cadena euleriana que une a los dos vértices de grado impar.

Algoritmo ciclo euleriano

Se parte de la matriz de adyacencia $A=(a_{ij})$

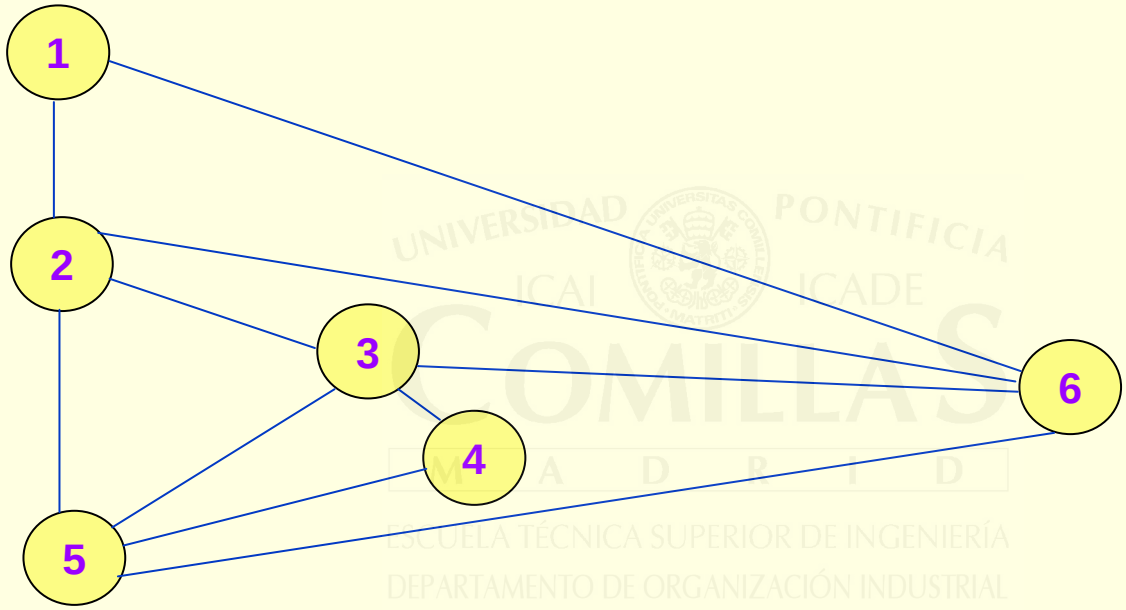
PASO 1: **Si existe i tal que $\sum a_{ij}$ no es par: NO existe ciclo**
En otro caso, ir a PASO 2. p=0

PASO 2: **Elegir fila k no nula de la matriz A. p=p+1**
a) Definir n=k y $C=\{v_k\}$
b) Buscar m tal que $a_{nm}>0$
c) Incluir v_m en C. Hacer $a_{nm}=a_{mn}=0$
d) Hacer n=m. Si $n\neq k$ ir a b). Si no, almacenar C_p e ir a PASO 3

PASO 3: **Si existe $a_{ij}\neq 0$ ir a PASO 2. Si no, PASO 4**

PASO 4: **Buscar 2 ciclos C_p con un vértice común. Sustituir el vértice por el ciclo. Repetir hasta que sólo quede un ciclo.**

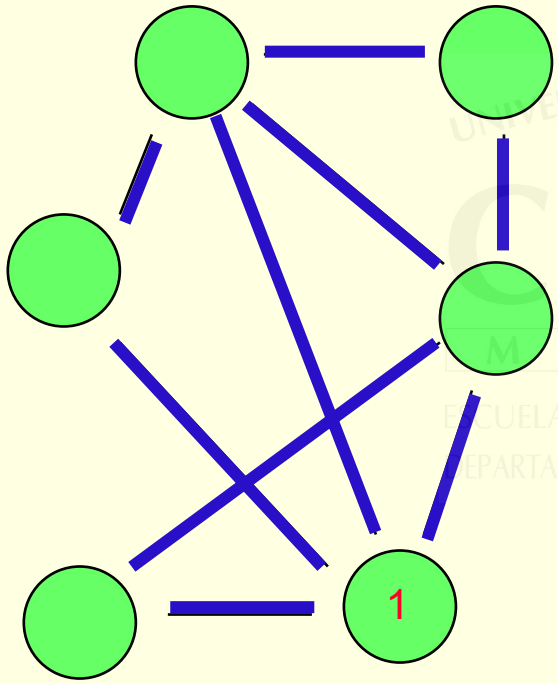
Ejemplo nº5

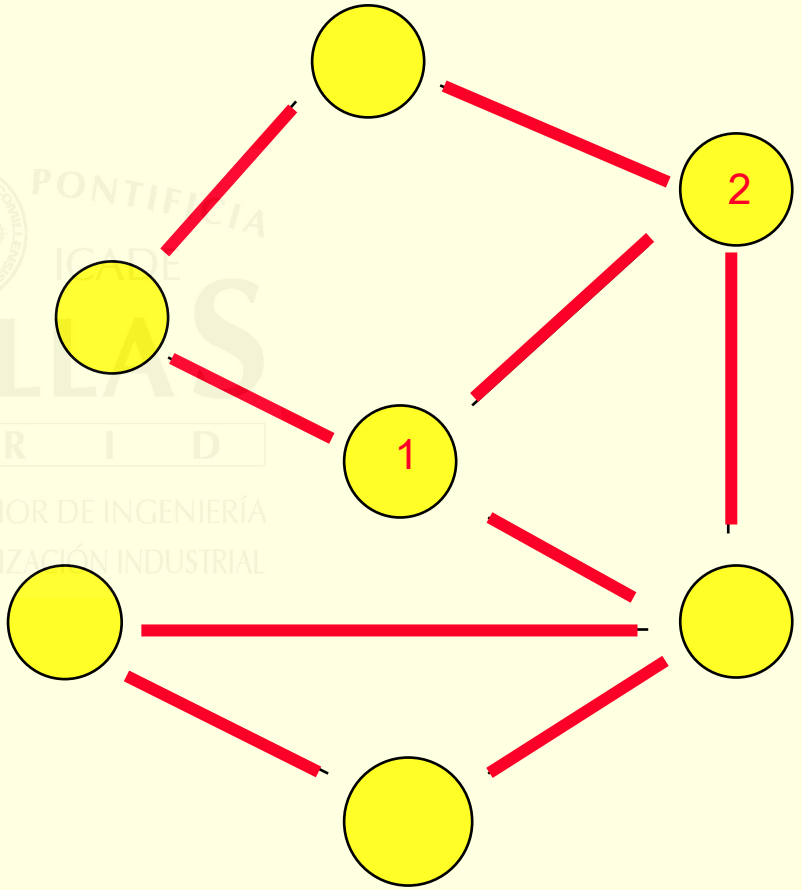


	1	2	3	4	5	6
1	0	1	0	0	0	1
2	1	0	1	0	1	1
3	0	1	0	1	1	1
4	0	0	1	0	1	0
5	0	1	1	1	0	1
6	1	1	1	0	1	0

3-2-1-6-2-5-3-4-5-6-3

Cadenas y ciclos eulerianos (II)



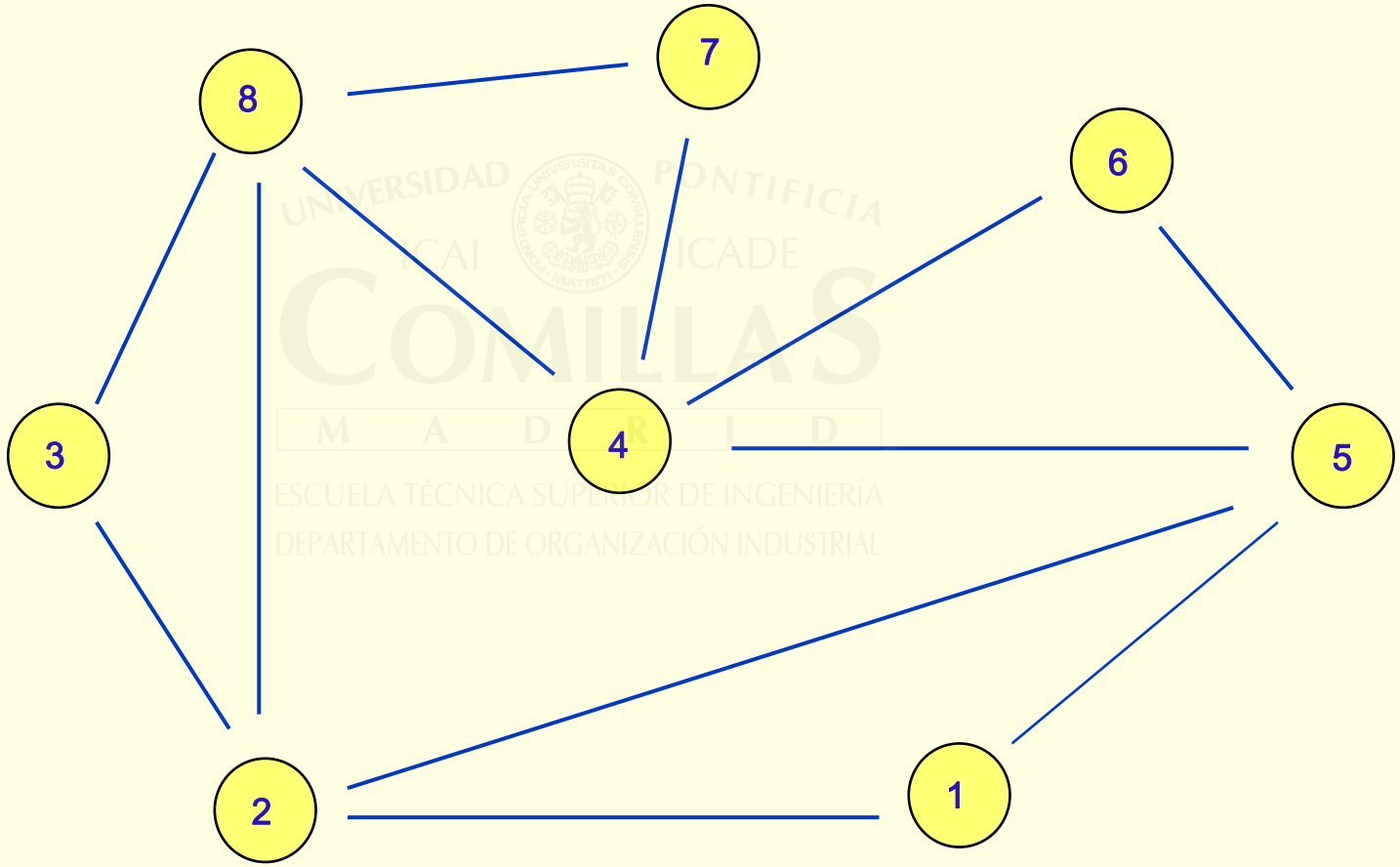


Algoritmo de Fleury

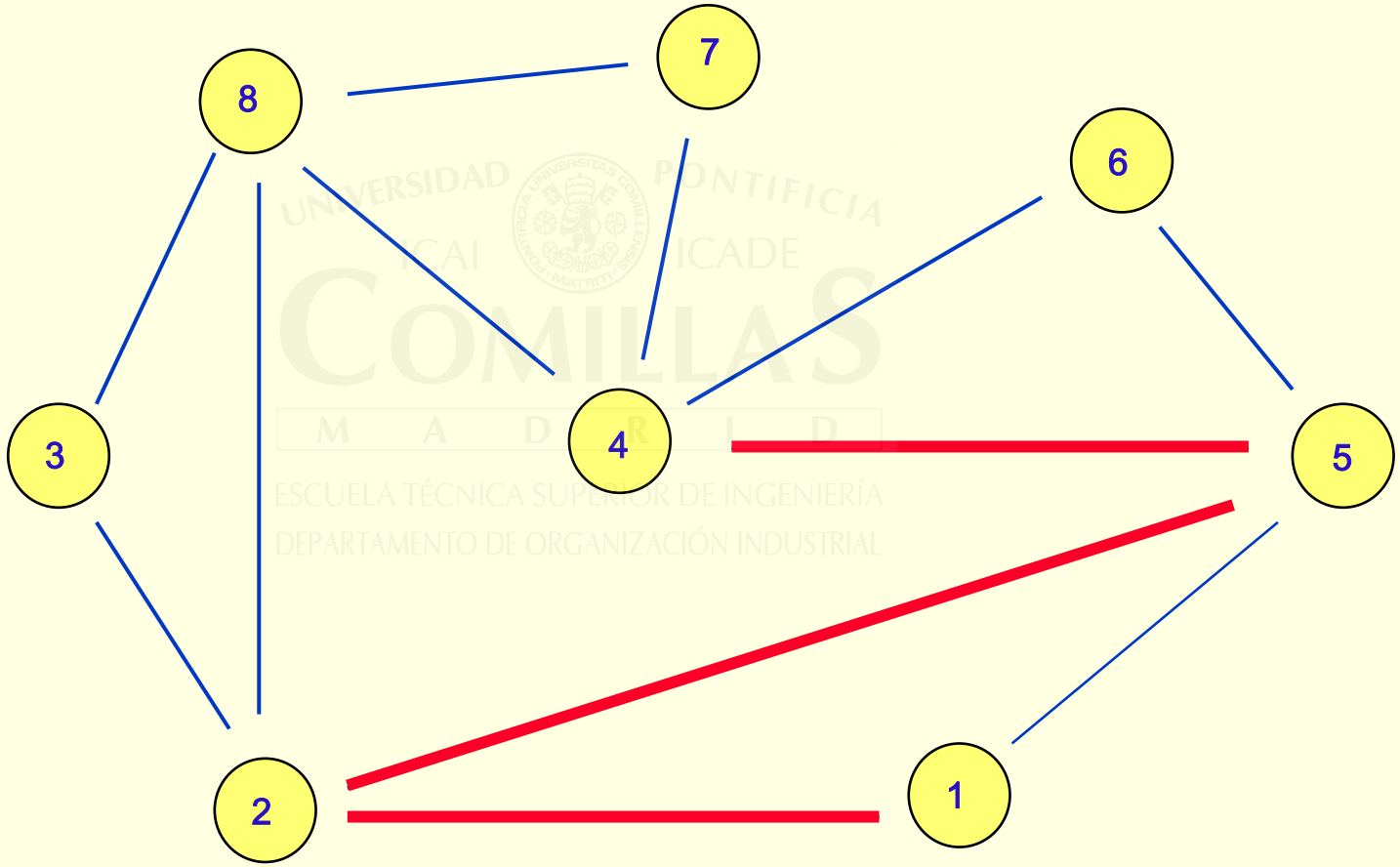
Si una red es conexa y tal que todos sus vértices son de grado par es posible recorrer todas sus aristas de un solo trazo sin necesidad de corregir el trayecto según el siguiente esquema:

- ✓ **Salir de un vértice cualquiera.**
- ✓ **Cada vez que recorramos una arista procedemos a tacharla.**
- ✓ **Cuando todas las aristas que inciden en un vértice han sido "tachadas", "tachamos" dicho vértice.**
- ✓ **No utilizar nunca una arista que, en el momento considerado, sea un itsmo.**

Algoritmo de Fleury. Ejemplo (1)

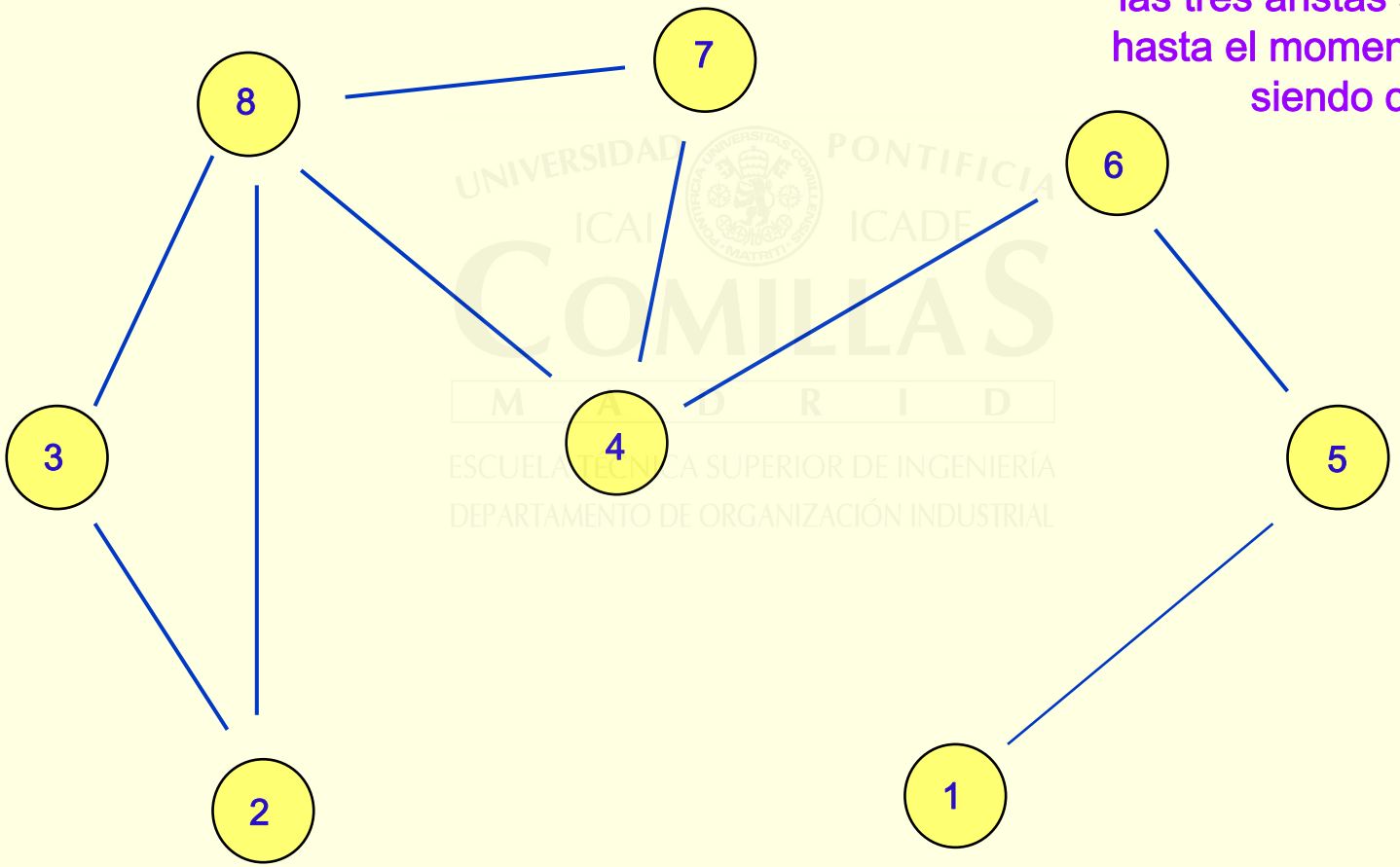


Algoritmo de Fleury. Ejemplo (2)



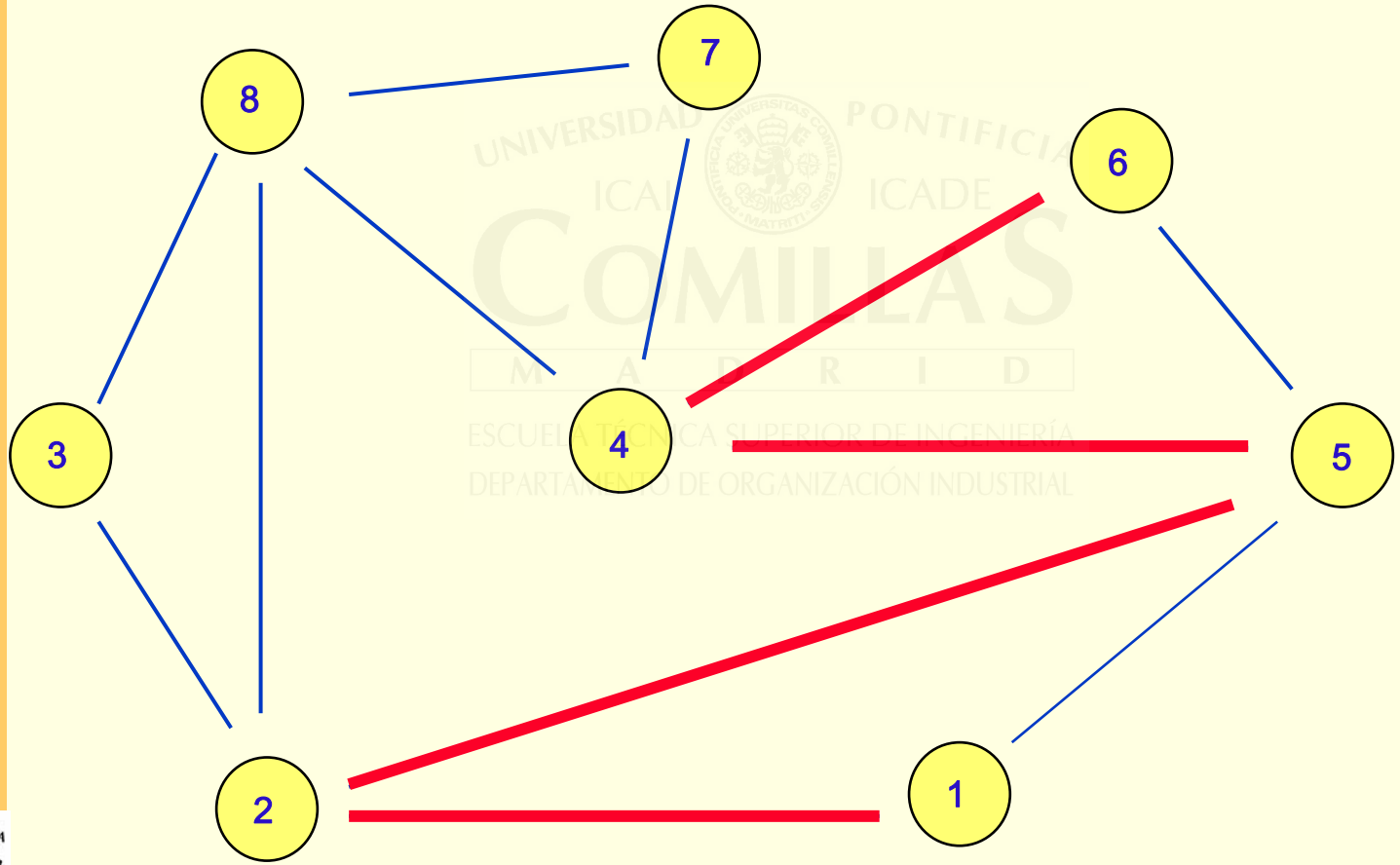
Algoritmo de Fleury. Ejemplo (3)

La arista 5-4 ha podido ser seleccionada porque al eliminar las tres aristas seleccionadas hasta el momento la red sigue siendo conexa.



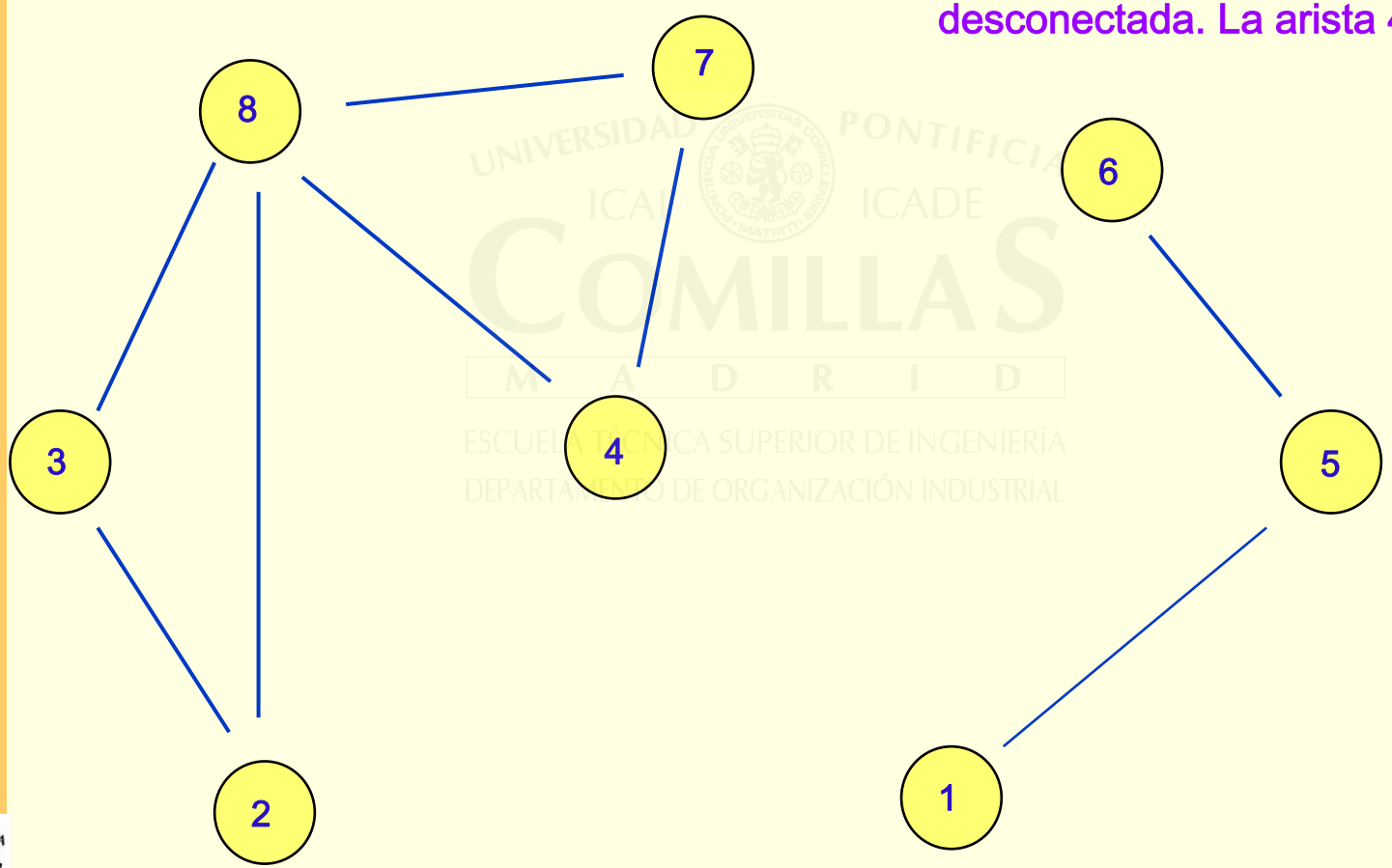
Algoritmo de Fleury. Ejemplo (4)

Si seleccionamos la arista 4-6

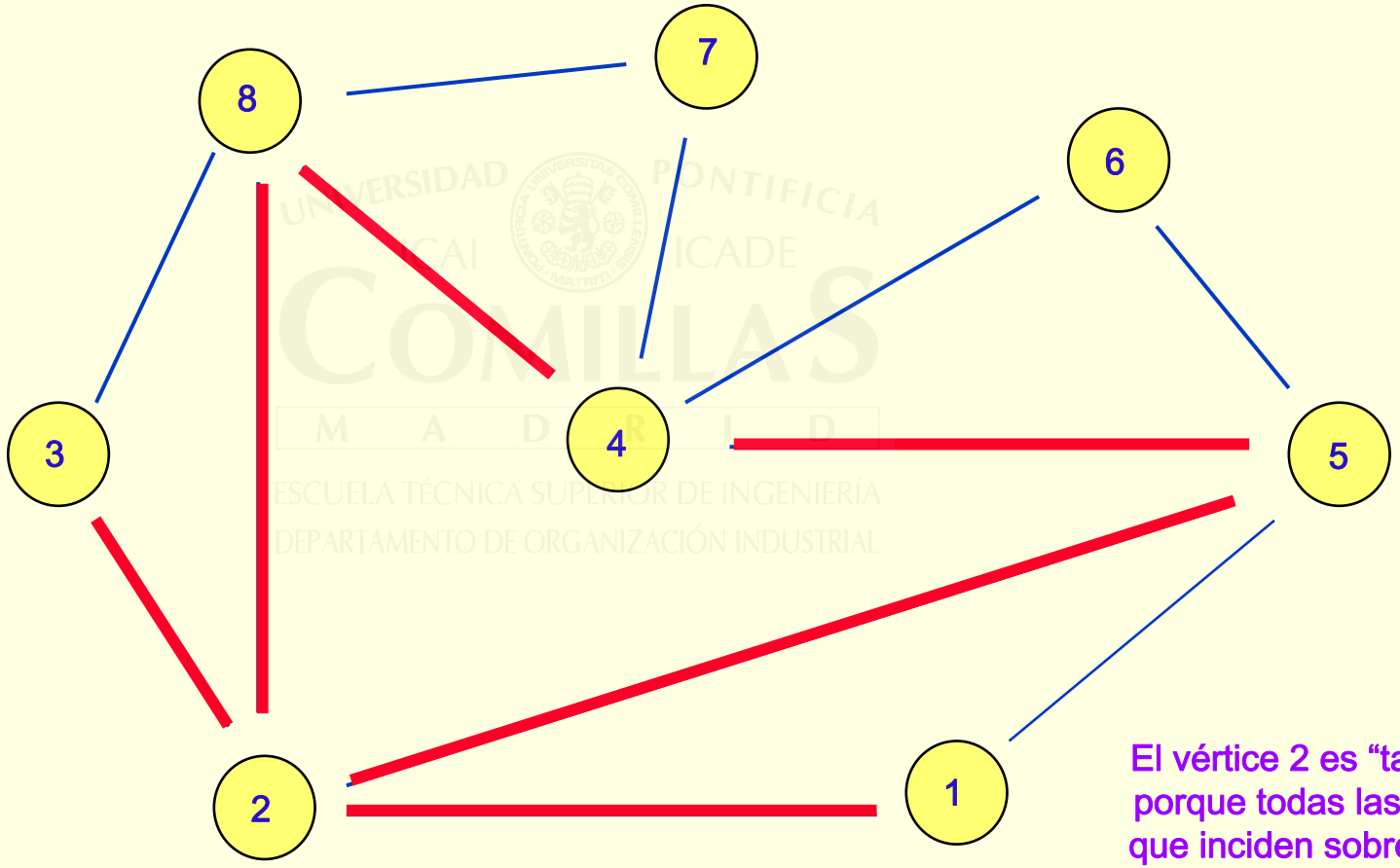


Algoritmo de Fleury. Ejemplo (5)

al eliminar las 4 aristas seleccionadas nos encontramos que la red queda desconectada. La arista 4-6 era un itsmo.

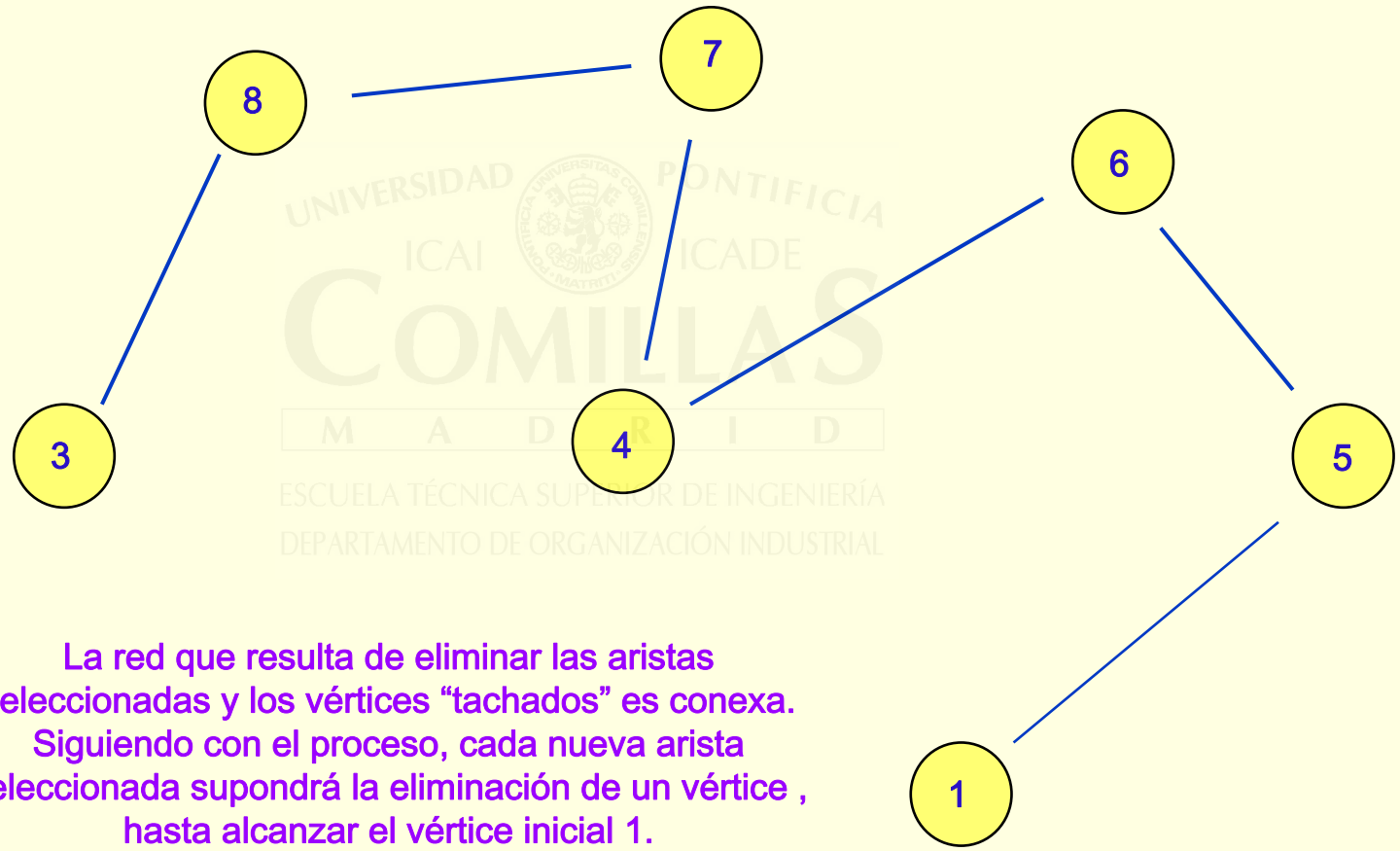


Algoritmo de Fleury. Ejemplo (6)



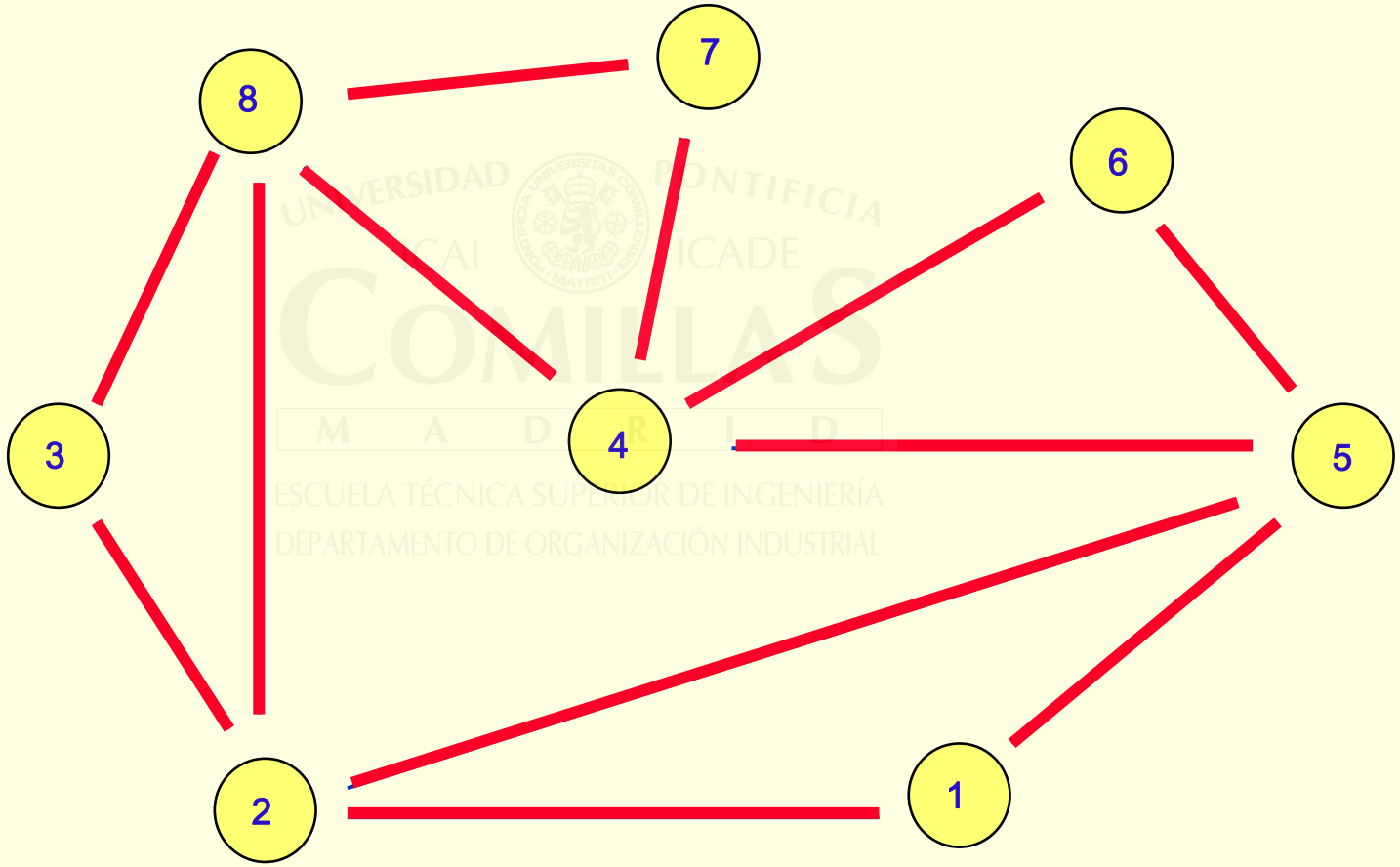
El vértice 2 es “tachado” porque todas las aristas que inciden sobre él han sido ya seleccionadas.

Algoritmo de Fleury. Ejemplo (7)



La red que resulta de eliminar las aristas seleccionadas y los vértices “tachados” es conexa. Siguiendo con el proceso, cada nueva arista seleccionada supondrá la eliminación de un vértice , hasta alcanzar el vértice inicial 1.

Algoritmo de Fleury. Ejemplo (8)



ESCUELA TÉCNICA SUPERIOR DE INGENIERÍA
DEPARTAMENTO DE ORGANIZACIÓN INDUSTRIAL

Teoría de Grafos o Redes

Parte II

Modelado y optimización sobre redes

Redes valoradas

- Una red se dice que está valorada si a cada arco o arista de la misma hay asociado uno o más valores de ciertas magnitudes.

- Estos valores pueden ser probabilidades, distancias, tiempos, costes o beneficios, etc., relacionados con el problema que está siendo modelado por el grafo. Pueden tomar valores negativos.

Modelado y optimización sobre redes

Problemas clásicos

- **Obtención de caminos de valor mínimo.**
- **Flujos óptimos sobre una red**
- **Arboles de expansión mínima**
- **Circuitos hamiltonianos de valor mínimo**
  - ✓ **El problema del viajante**
  - ✓ **El problema del cartero chino**

ESCUELA TÉCNICA SUPERIOR DE INGENIERÍA
DEPARTAMENTO DE ORGANIZACIÓN INDUSTRIAL

Teoría de Grafos o Redes

Modelado y optimización sobre redes

Obtención de caminos de valor mínimo

Problemas de camino mínimo (I)

Se trata de encontrar el camino entre dos vértices de una red orientada cuya suma de valores asociados a los arcos del mismo sea mínima (ó máxima, según los casos).

Son problemas muy importantes:

- **en sí mismos**
- **subrutinas**

No deben existir circuitos de valor total negativo

Los algoritmos más utilizados son los de

- Dijkstra
- Bellman-Ford

Problemas de camino mínimo (II)
Ecuaciones generales de Bellman

Sea u_1 el vértice de partida y v_{ij} el valor asociado al arco (i, j). El modelo de optimización consiste en las ecuaciones de recurrencia, conocidas como ecuaciones generales de Bellman

$$u_1 = 0$$

$$u_j = \underset{k \neq j}{Min} \left\{ u_k + v_{kj} \right\}, \; j = 2..n$$

donde

- $v_{kj} = \infty$ si no existe el arco (k, j), y

- u_j es un conjunto de etiquetas que, una vez hechas permanentes, representan la longitud del camino mínimo de 1 a j.

Problemas de camino mínimo: Algoritmo de Dijkstra

Válido cuando las longitudes de los arcos son positivas

P: nodos etiquetados de forma permanente **T: nodos etiquetados de forma transitoria**

u_j: longitud camino mínimo de 1 a i **pred(j): predecesor del vértice j**

Complejidad: $\theta(n^2)$

PASO 0

$u_1=0$, $u_j=d_{1j}$ (∞ si no existe arco), **P={1}, T={2,...,n}, pred(j)= 1**

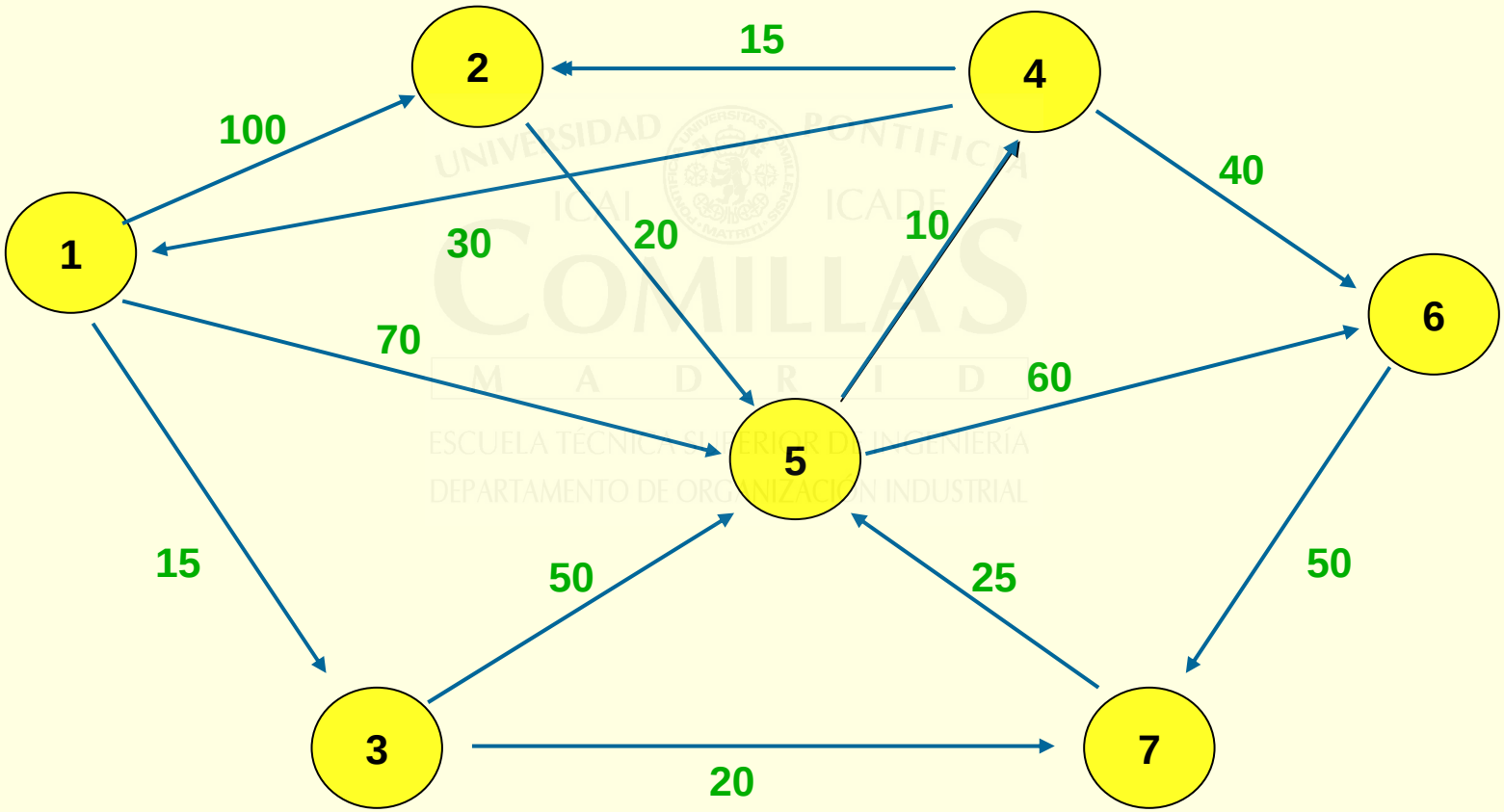
PASO 1: Designar nueva etiqueta permanente: vértice con menor valor de etiqueta transitoria

- Buscar $k \in T$ / $u_k=\min \{ u_j\}$, $T=T-\{k\}$, $P=P \cup \{k\}$
- **Si T=Ø**, PARAR. **Si no, ir al** PASO 2

PASO 2: Revisar etiquetas transitorias

- Poner para todo $j \in T$, $u_k = \min \{u_j, u_k + d_{kj}\}$.
- Si se modifica la etiqueta, pred(j) = k. Ir al PASO 1

Ejemplo: Encontrar el camino de valor mínimo entre los vértices 1 y 2 (I)



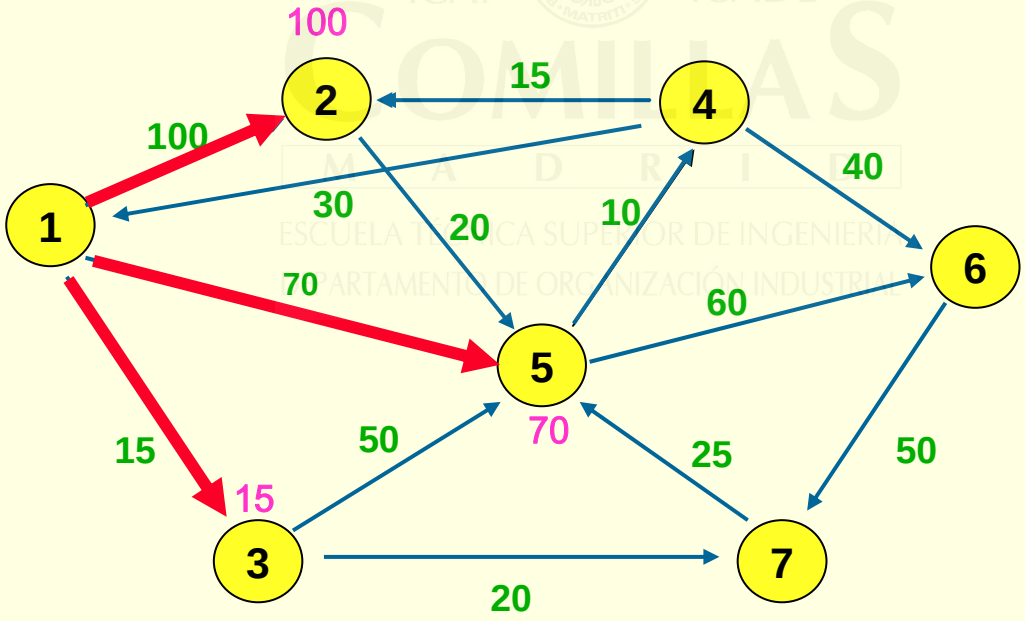
Ejemplo: Encontrar el camino de valor mínimo entre los vértices 1 y 2 (II)

Paso 0:

$$u_1 = 0; u_2 = 100; u_3 = 15; u_5 = 70; u_4 = u_6 = u_7 = \infty$$

$$P = \{1\}; T = \{2,3,4,5,6,7\}$$

$$pred(j) = 1,\ j = 2,3,4,5,6,7$$



Caminos de valor mínimo desde el vértice 1 a los demás vértices del grafo de longitud 1

Ejemplo: Encontrar el camino de valor mínimo entre los vértices 1 y 2 (II)

Paso 1: $u_k = 15, k = 3$

$P = \{1,3\}; T = \{2,4,5,6,7\} \neq \phi$

Paso 2:

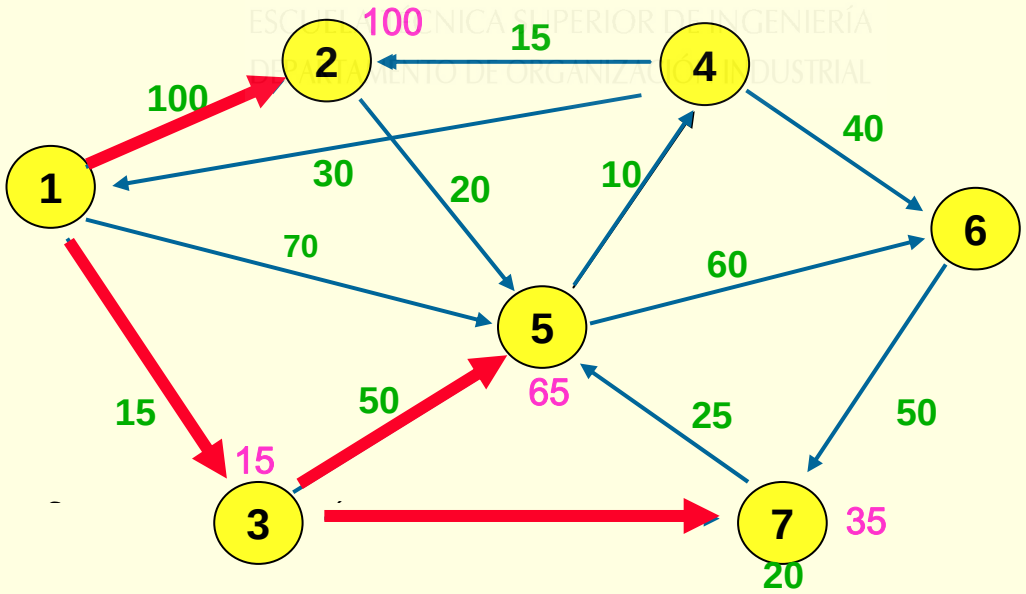
$u_2 = \min\{u_2, u_3 + v_{3,2}\} = \min\{100, 15+\infty\} = 100$

$u_4 = \min\{u_4, u_3 + v_{3,4}\} = \min\{\infty, 15+\infty\} = \infty$

$u_5 = \min\{u_5, u_3 + v_{3,5}\} = \min\{70, 15+50\} = 65 \Rightarrow pred(5) = 3$

$u_6 = \min\{u_6, u_3 + v_{3,6}\} = \min\{\infty, 15+\infty\} = \infty$

$u_7 = \min\{u_7, u_3 + v_{3,7}\} = \min\{\infty, 15+20\} = 35 \Rightarrow pred(7) = 3$



Caminos de valor mínimo desde el vértice 1 a los demás vértices del grafo de longitud menor o igual que 2.

Ejemplo: Encontrar el camino de valor mínimo entre los vértices 1 y 2 (III)

Paso 1:

$$u_k = 35, k = 7$$

$$P = \{1,3,7\};\ T = \{2,4,5,6\} \neq \phi$$

Paso 2:

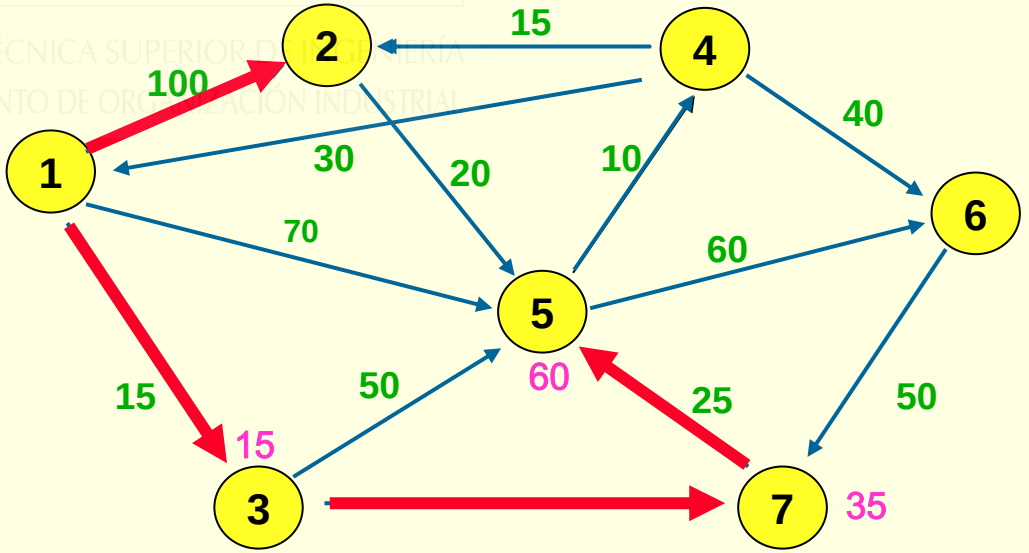
$$u_2 = \min\{u_2, u_7 + v_{7,2}\} = \min\{100, 35 + \infty\} = 100$$

$$u_4 = \min\{u_4, u_7 + v_{7,4}\} = \min\{\infty, 35 + \infty\} = \infty$$

$$u_5 = \min\{u_5, u_7 + v_{7,5}\} = \min\{65, 35 + 25\} = 60 \Rightarrow pred(5) = 7$$

$$u_6 = \min\{u_6, u_7 + v_{7,6}\} = \min\{\infty, 35 + \infty\} = \infty$$

Caminos de valor mínimo desde el vértice 1 a los demás vértices del grafo de longitud menor o igual que 3.



Ejemplo: Encontrar el camino de valor mínimo entre los vértices 1 y 2 (IV)

Paso 1:

$$u_k = 60, k = 5$$

$$P = \{1,3,7,5\}; T = \{2,4,6\} \neq \phi$$

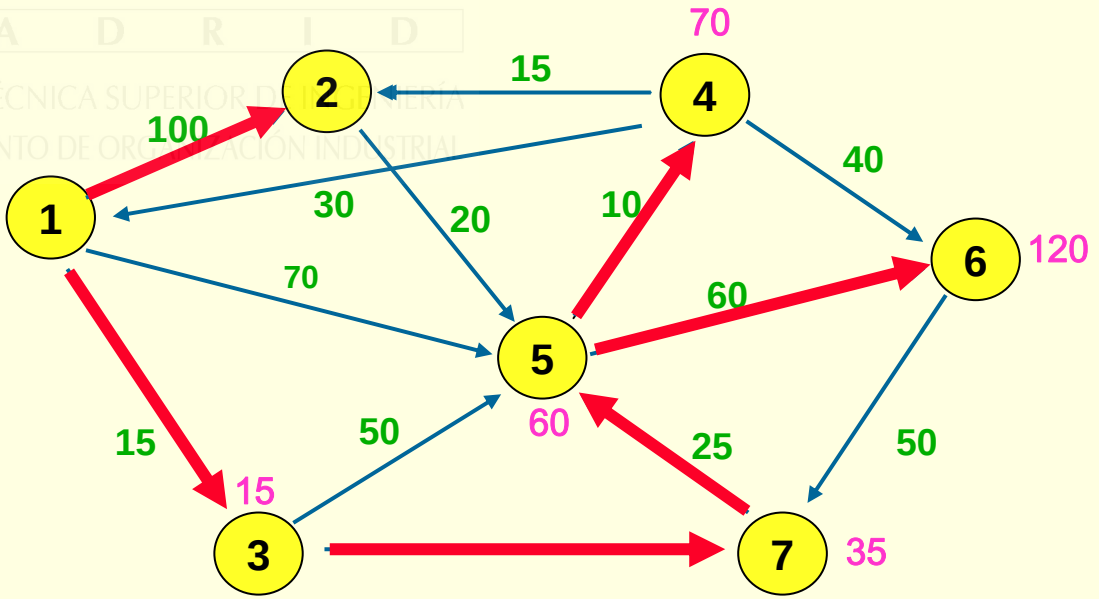
Paso 2:

$$u_2 = \min\{u_2, u_5 + v_{5,2}\} = \min\{100, 60 + \infty\} = 100$$

$$u_4 = \min\{u_4, u_5 + v_{5,4}\} = \min\{\infty, 60 + 10\} = 70 \Rightarrow pred(6) = 5$$

$$u_6 = \min\{u_6, u_5 + v_{5,6}\} = \min\{\infty, 60 + 60\} = 120 \Rightarrow pred(6) = 5$$

Caminos de valor mínimo desde el vértice 1 a los demás vértices del grafo de longitud menor o igual que 4.



Ejemplo: Encontrar el camino de valor mínimo entre los vértices 1 y 2 (V)

Paso 1:

$$u_k = 70, k = 4$$

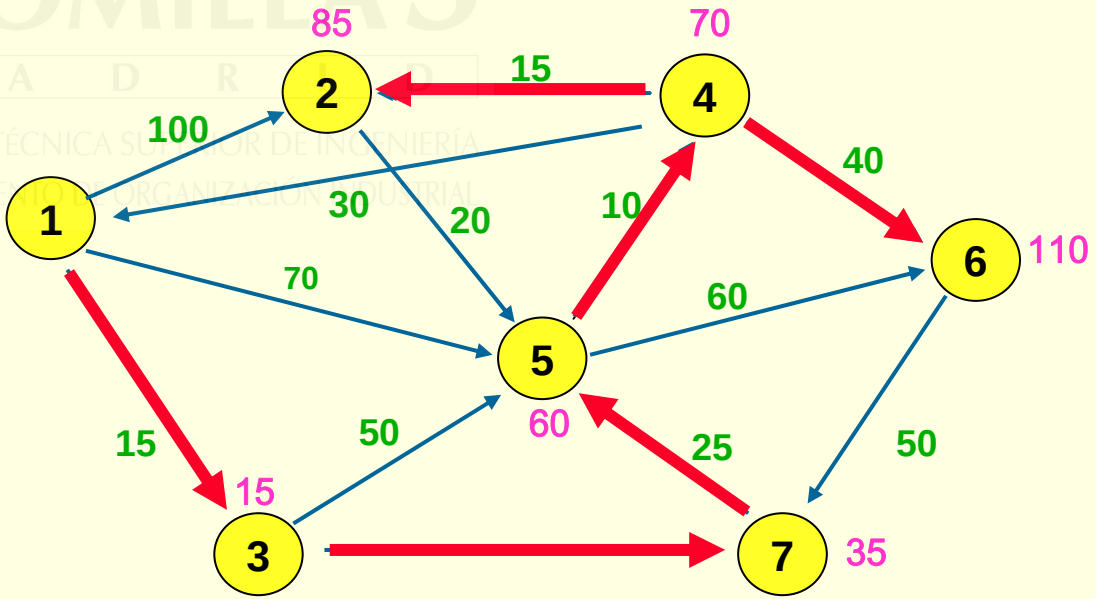
$$P = \{1,3,7,5,4\};\ T = \{2,6\} \neq \phi$$

Paso 2:

$$u_2 = \min\{u_2, u_4 + v_{4,2}\} = \min\{100, 70+15\} = 85 \Rightarrow pred(2) = 4$$

$$u_6 = \min\{u_6, u_4 + v_{4,6}\} = \min\{120, 70+40\} = 110 \Rightarrow pred(6) = 4$$

Caminos de valor mínimo desde el vértice 1 a los demás vértices del grafo de longitud menor o igual que 5.



Algoritmo de Bellman-Ford

- **Es un algoritmo más general que el anterior**
- **No recomendable si no hay arcos de valor negativo**
- **Complejidad: $\theta(n^3)$**
- **No válido si hay circuitos de valor negativo**

u_j: longitud camino mínimo de 1 a j
u_j^m: long. camino mínimo de 1 a j usando **m** arcos o menos

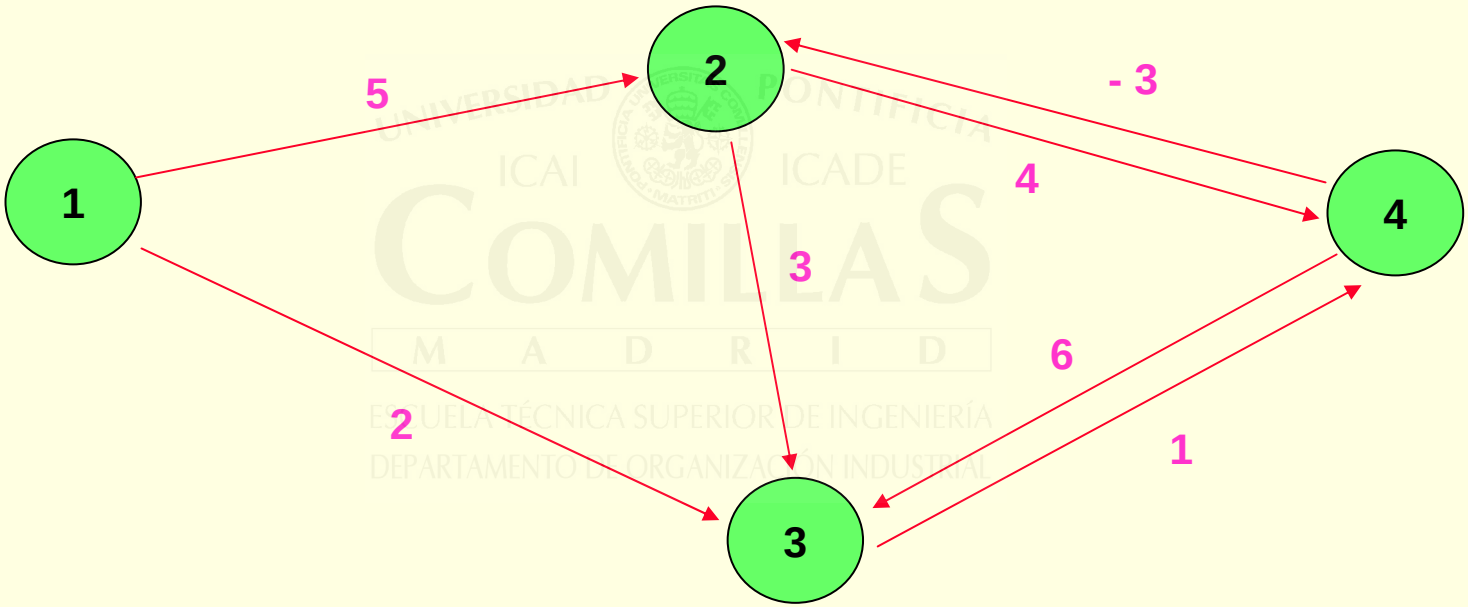
INICIO: $u_1^1=0$, $u_j^1 = d_{1j}$

ITERACIÓN m:

$$u_j^{m+1} = Min\left\{ u_j^m , Min_{k\neq j}\left\{ u_k^m + v_{kj} \right\} \right\}$$

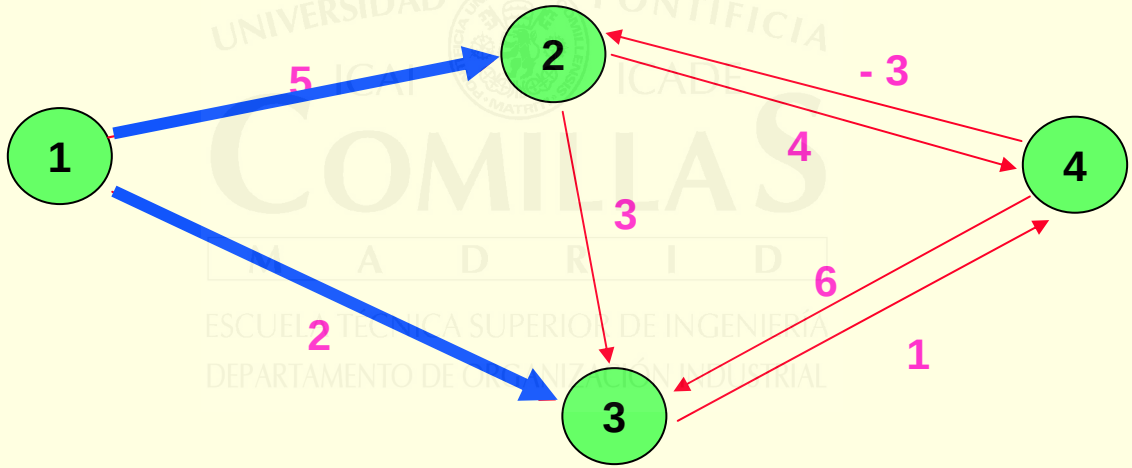
PARAR: Si $u_j^m = u_j^{m+1}$, para todo j. En otro caso, hacer **m=m+1** y volver a la iteración.

Ejemplo (I): Encontrar los caminos de valor mínimo que unen el vértice 1 con los demás



Ejemplo (II)

Inicio $m = 1;\ u_1^1 = 0,\ u_2^1 = 5,\ u_3^1 = 2,\ u_4^1 = \infty$



Ejemplo (III)

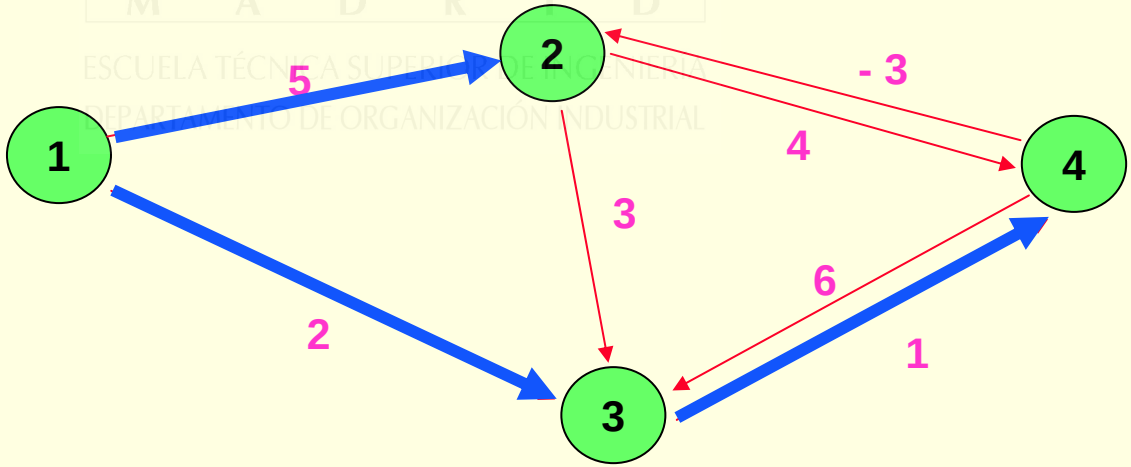
Iteración

$$m = 2$$

$$u_2^2 = \min\left\{u_2^1, \min\left\{u_3^1 + v_{3,2}, u_4^1 + v_{4,2}\right\}\right\} = \min\left\{5, \min\left\{2+\infty, \infty-3\right\}\right\} = 5$$

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

$$u_4^2 = \min\left\{u_4^1, \min\left\{u_2^1 + v_{2,4}, u_3^1 + v_{3,4}\right\}\right\} = \min\left\{\infty, \min\left\{5+4, 2+1\right\}\right\} = 3$$



Como $u_{4,2} \neq u_{4,1}$ se vuelve a iterar

Ejemplo (IV)

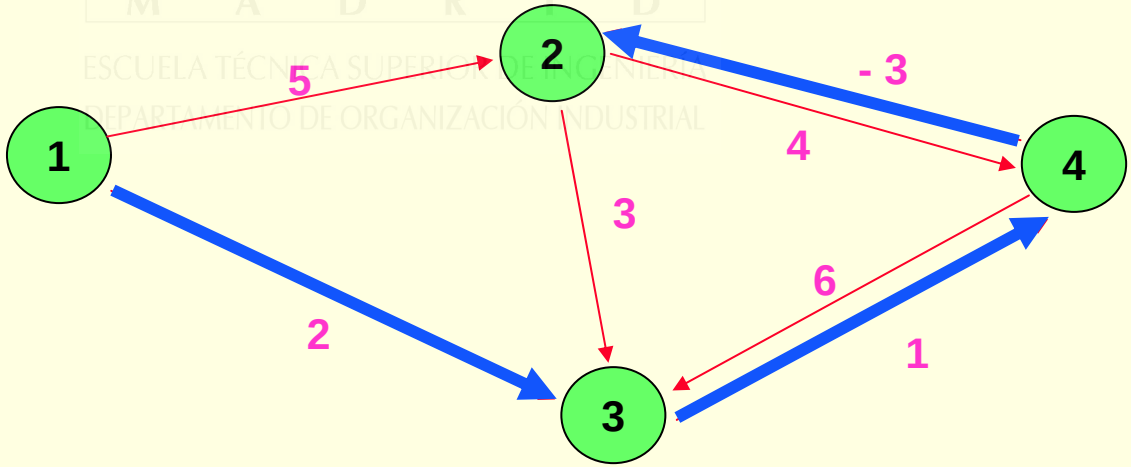
Iteración

$$m = 3$$

$$u_2^3 = \min\left\{u_2^2, \min\left\{u_3^2 + v_{3,2}, u_4^2 + v_{4,2}\right\}\right\} = \min\left\{5, \min\left\{2+\infty, 3-3\right\}\right\} = 0$$

$$u_3^3 = \min\left\{u_3^2, \min\left\{u_2^2 + v_{2,3}, u_4^2 + v_{4,3}\right\}\right\} = \min\left\{2, \min\left\{5+3, 3+6\right\}\right\} = 2$$

$$u_4^3 = \min\left\{u_4^2, \min\left\{u_2^2 + v_{2,4}, u_3^2 + v_{3,4}\right\}\right\} = \min\left\{3, \min\left\{5+4, 2+1\right\}\right\} = 3$$



Como $u_2^3 \neq u_2^2$ *se vuelve a iterar*

Ejemplo (V)

Iteración

$$m = 4$$

$$u_2^4 = \min\left\{u_2^3, \min\left\{u_3^3 + v_{3,2}, u_4^3 + v_{4,2}\right\}\right\} = \min\left\{0, \min\left\{2+\infty, 3-3\right\}\right\} = 0$$

$$u_3^4 = \min\left\{u_3^3, \min\left\{u_2^3 + v_{2,3}, u_4^3 + v_{4,3}\right\}\right\} = \min\left\{2, \min\left\{0+3, 3+6\right\}\right\} = 2$$

$$u_4^4 = \min\left\{u_4^3, \min\left\{u_2^3 + v_{2,4}, u_3^3 + v_{3,4}\right\}\right\} = \min\left\{3, \min\left\{0+4, 2+1\right\}\right\} = 3$$

Como $u_j^4 \neq u_j^3, \forall j$, se cumple el criterio de parada

Lo que por otra parte era de esperar ya que habíamos llegado a la tercera iteración, que coincide con el número de vértices a los que se quiere llegar desde el vértice 1.

Observaciones

Como habrá podido observarse la evolución del algoritmo es la característica de los algoritmos de programación dinámica:

1º: un proceso de optimización de carácter secuencial, y

2º: a la vez que se obtiene el óptimo buscado se hayan también los caminos de valor mínimo desde el vértice inicial 1 al resto de los vértices de la red.

En el caso de que todos los arcos tengan valor 1, la aplicación de cualquiera de los algoritmos anteriores conduce a la obtención de caminos de longitud mínima entre dos vértices..

ESCUELA TÉCNICA SUPERIOR DE INGENIERÍA
DEPARTAMENTO DE ORGANIZACIÓN INDUSTRIAL

Teoría de Grafos o Redes

Modelado y optimización sobre redes

Flujos óptimos sobre una red

Problemas de flujo máximo en red

- **Un grafo G(V,U) dirigido, sin bucles y fuertemente conexo es una red de transporte si:**
  - ✓ **existe una única fuente *f***
  - ✓ **existe un único sumidero *s***
  - ✓ **G está valorado. El valor asociado a un arco es una cantidad no negativa, su *capacidad*.**

- **Se trata de asignar (*función de flujo*) a cada arco un valor, su *flujo*, conforme a las condiciones que a continuación se especifican, de forma que el objetivo es "transportar" la mayor cantidad posible de la fuente al sumidero, es decir, que la suma de los flujos asociados a los arcos que inciden en el sumidero sea máxima.**

Conceptos relacionados con este problema (I)

❑ Flujo compatible, posible o realizable:

- ✓ El flujo asociado a un arco no supera su capacidad
- ✓ Verifica la ley de conservación de flujo: el flujo que "entra" en un vértice es igual al que "sale" de él.
- ✓ El flujo que sale de *f* es el que llega a *s*

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

❑ Corte

- ✓ Conjunto de aristas cuya eliminación desconecta a G en dos componentes tal que s y t no están en la misma
- ✓ Capacidad de un corte: suma de las capacidades de los arcos que lo forman

Conceptos relacionados con este problema (II)

Arco saturado

Es aquel por el que circula un flujo igual a su capacidad.

Flujo completo

Es una función de flujo en la que todo camino ligando la entrada con la salida contiene al menos un arco saturado.

Corte de una red de transporte

Todo subconjunto de arcos cuya supresión aisla la entrada de la salida.

Capacidad de un corte

Es la suma de las capacidades de sus arcos.

Conceptos relacionados con este problema (III)

Si φ es un flujo posible cuyo valor es $V(\varphi)$ y (P,P') es un corte, cuya capacidad total es $C(P, P')$, se tiene

$$V(\varphi) \leq C(P, P')$$

Además,

$$\max V(\varphi) = \min C(P, P')$$

Teorema de Ford -Fulkerson

El máximo flujo que se puede enviar coincide con la mínima capacidad del conjunto de los cortes de la red

Algoritmo de Ford-Fulkerson (I)

Un algoritmo de etiquetado

PASO 1: **Obtener un flujo compatible.**

PASO 2: **Obtener un flujo completo**

Si el flujo obtenido en el paso 1 no es completo existirá entre la entrada y la salida al menos un camino carente de arcos saturados. Aumentamos los flujos en los arcos del mismo de unidad en unidad hasta saturar al menos un arco. Si hay otros caminos en las mismas circunstancias se repite la operación

PASO 3: **Marcaje de vértices**

3.1 Comenzamos marcando la entrada con un signo +.

3.2 Una vez marcado un vértice i:

- Marcamos con + *i* todo vértice *k* aún no marcado y tal que exista el arco (*i, k*) y no esté saturado
- Marcamos con – *i* todo vértice *k* aún no marcado y tal que exista el arco (*k, i*) por el que circule un flujo no nulo.

Algoritmo de Ford-Fulkerson (II)

PASO 4: **Si por este proceso llegamos a marcar la salida, el flujo no es máximo. Consideramos entonces una secuencia de vértices marcados que una la entrada con la salida. A todo arco de la misma orientado en el sentido de la secuencia de vértices le aumentamos su flujo en una unidad y se lo restamos si está orientado en sentido opuesto. El proceso se reitera hasta saturar uno de los arcos orientados en el sentido de la secuencia o anular el flujo que circula por uno de los orientados en sentido opuesto.**

PASO 5: **Se reitera el paso 4 hasta que sea imposible marcar la salida.**

PASO 6: Cuando esto sucede el flujo es máximo. El corte de mínima capacidad son los vértices etiquetados.

Otra versión del Algoritmo de Ford-Fulkerson

PASO 1: **obtener un flujo compatible asignando a todos los arcos un flujo 0. Etiquetar f con (+, ∞)**

PASO 2: Si no se pueden etiquetar más vértices: **PASO 6**

Si se pueden etiquetar: **PASO 3**

PASO 3: Sea ***i*** el último etiquetado y ***j*** no etiquetado unido a ***i***

Si *(i,j)*$\in U$ y $\varphi_{ij} < C_{ij}$, $\delta_j = \min\{\delta_i, C_{ij} - \varphi_{ij}\}$, etiqueta j: $(i+, \delta_j)$

Si ***(j,i)***$\in U$ y $\varphi_{ij} > 0$, $\delta_j = \min\{\delta_i, \varphi_{ij}\}$, **etiqueta j: $(i-, \delta_j)$**

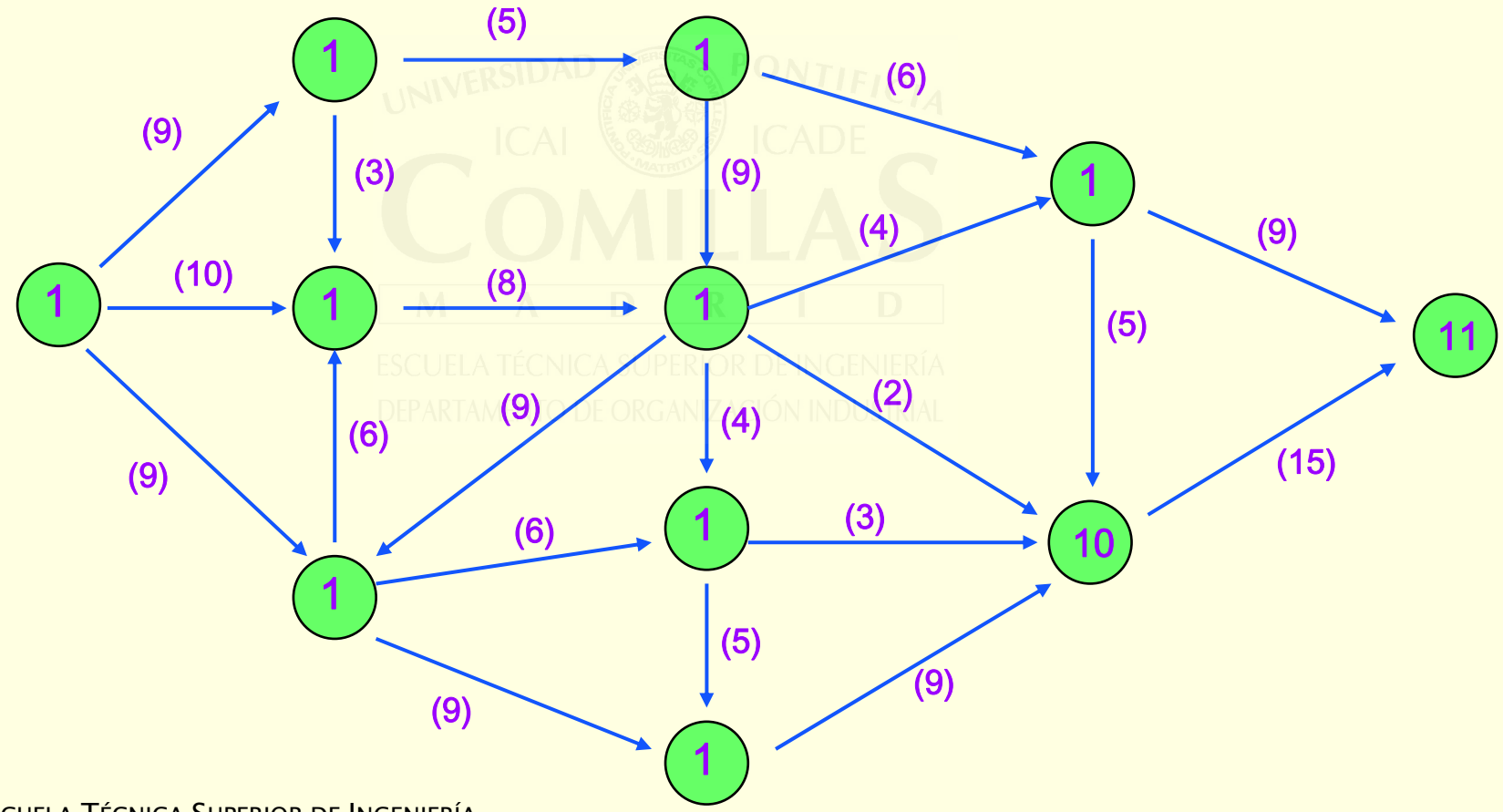
PASO 4: Si el sumidero está etiquetado **ir a PASO 5**. Si no, **PASO 2**

PASO 5: Aumentar el flujo de unidad en unidad en los vértices etiquetados en δ_t (disminuyendolo de igual forma en las etiquetas negativas). Borrar etiquetas.

PASO 6: **El flujo es máximo. El corte de mínima capacidad son los vértices etiquetados**.

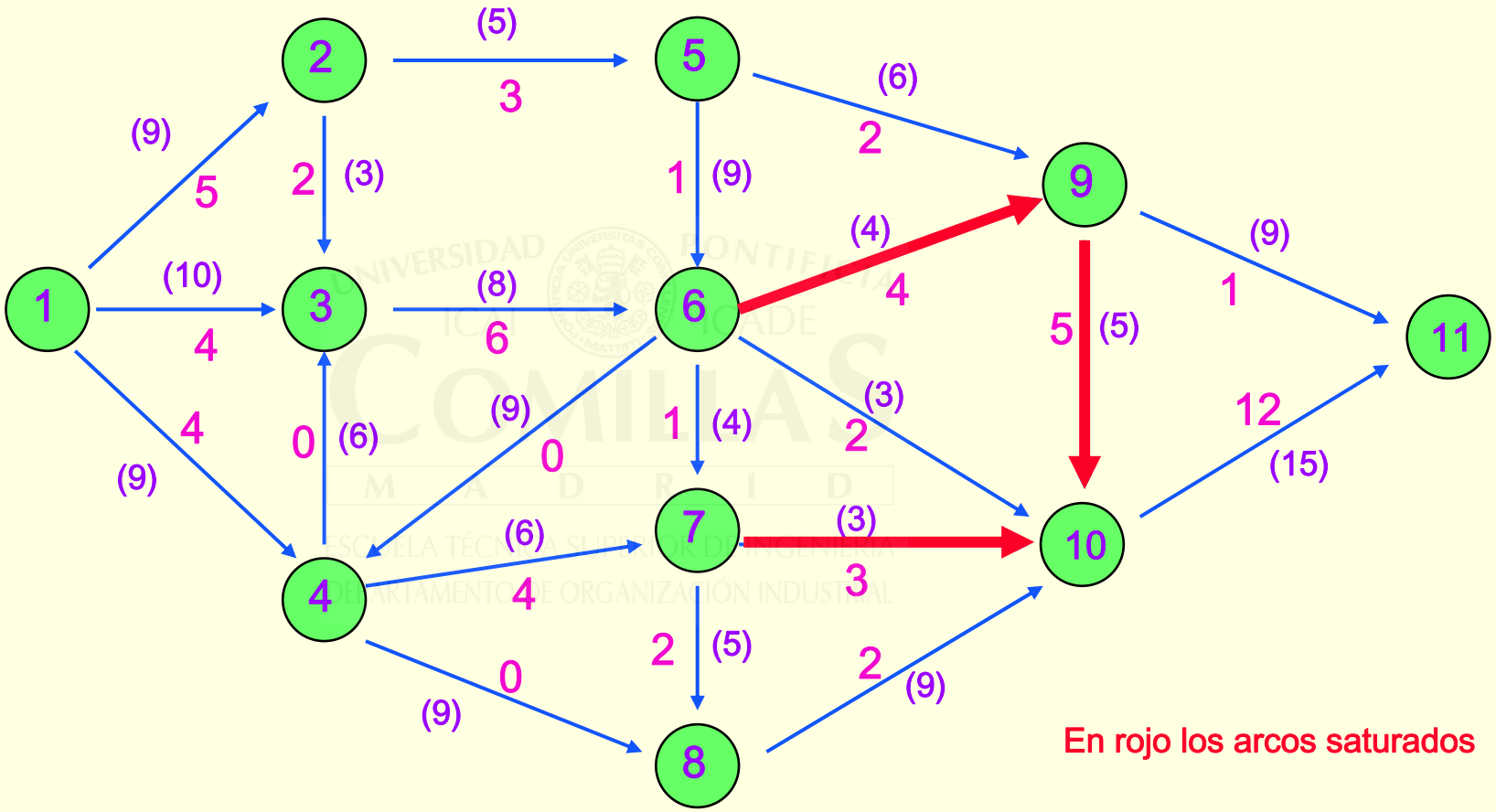
Ejemplo.

El número entre paréntesis representa la capacidad del arco



Ejemplo.

Obtención de un flujo posible



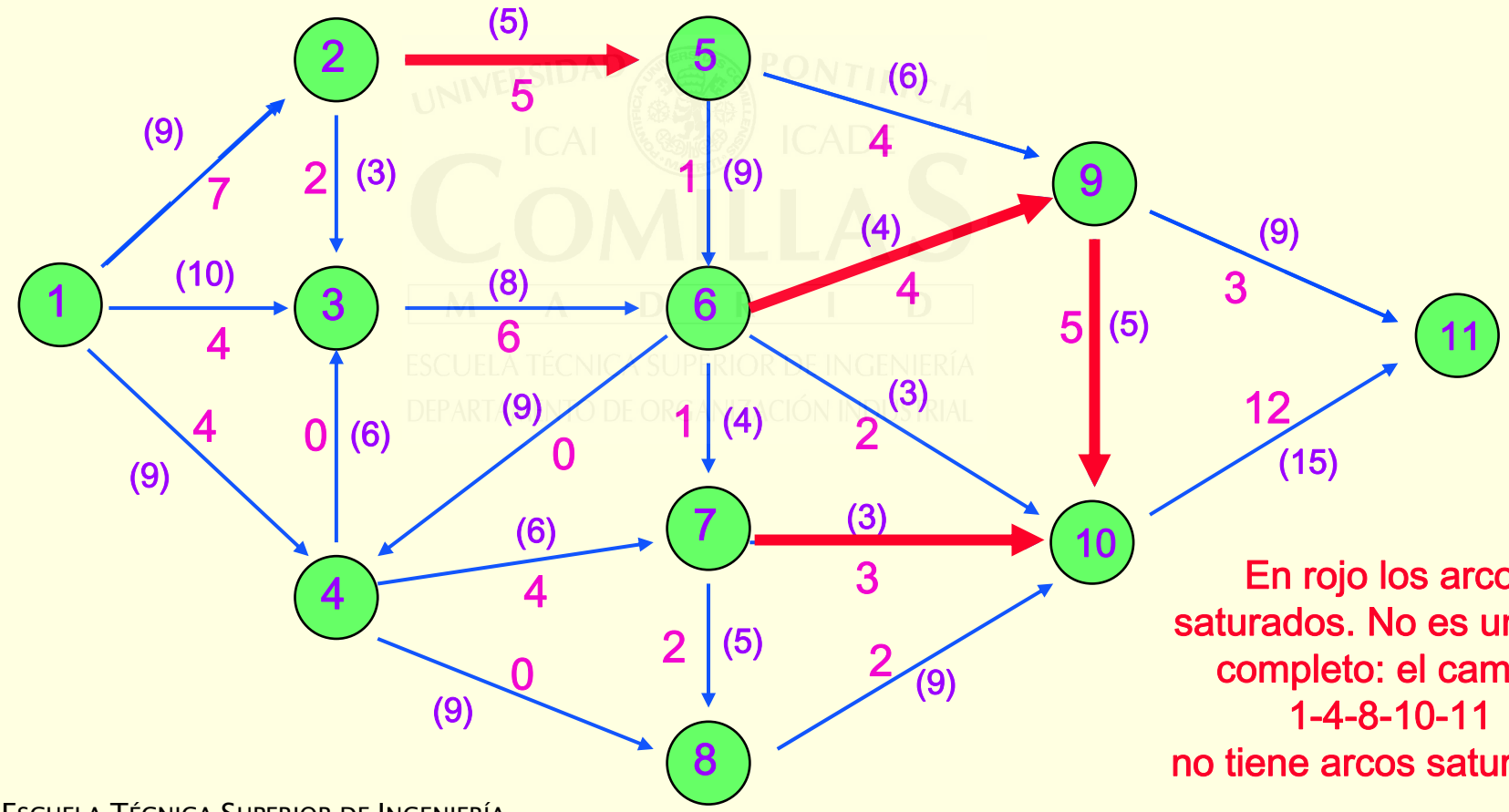
En rojo los arcos saturados

Es un flujo posible pero no es completo

Ejemplo.

Obtención de un flujo completo (I)

El camino 1-2-5-9-11 no tiene ningún arco saturado. Incrementamos su flujo en dos unidades hasta saturar el arco (2-5)

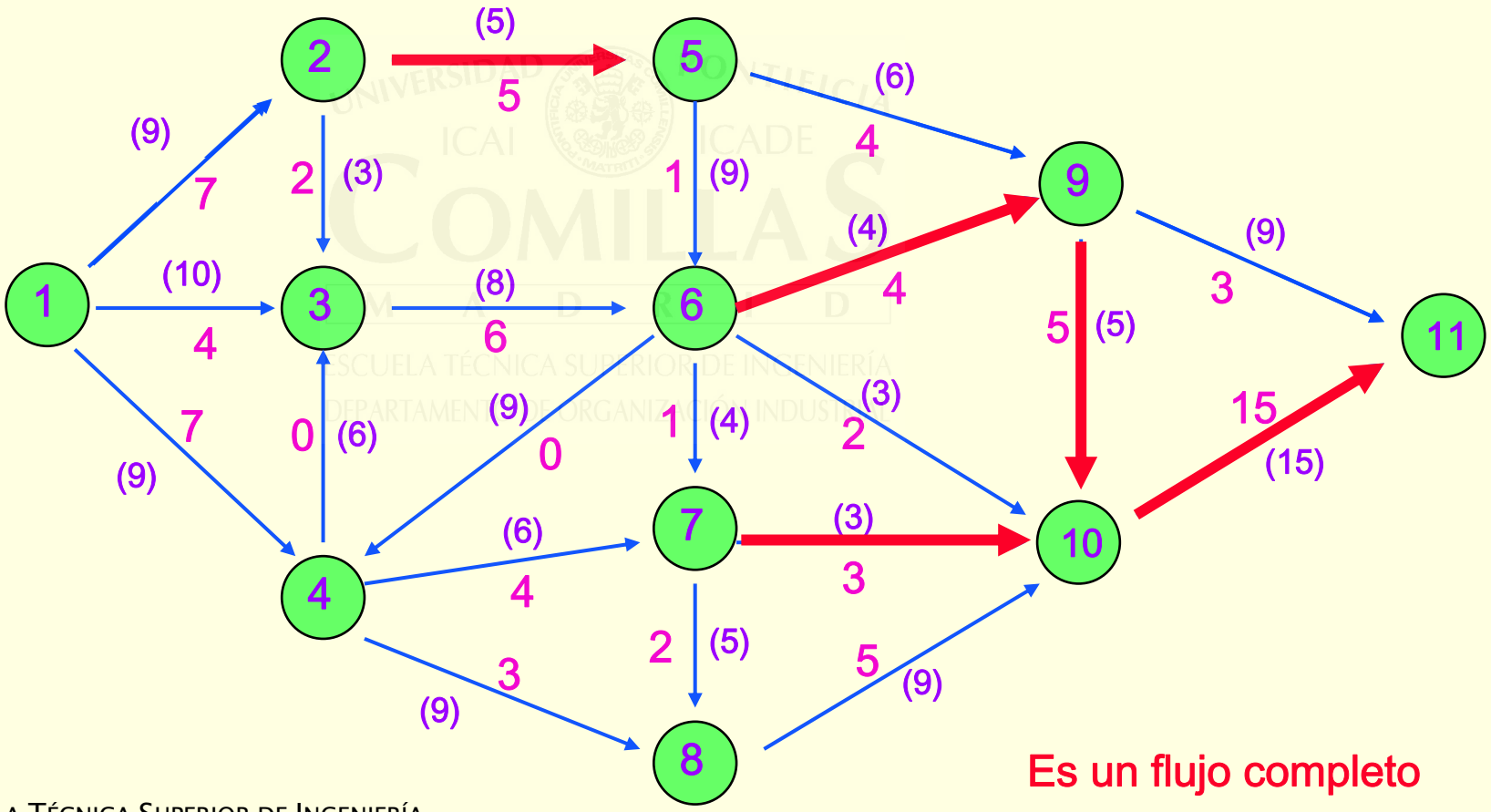


En rojo los arcos saturados. No es un flujo completo: el camino 1-4-8-10-11 no tiene arcos saturados.

Ejemplo.

Obtención de un flujo completo (II)

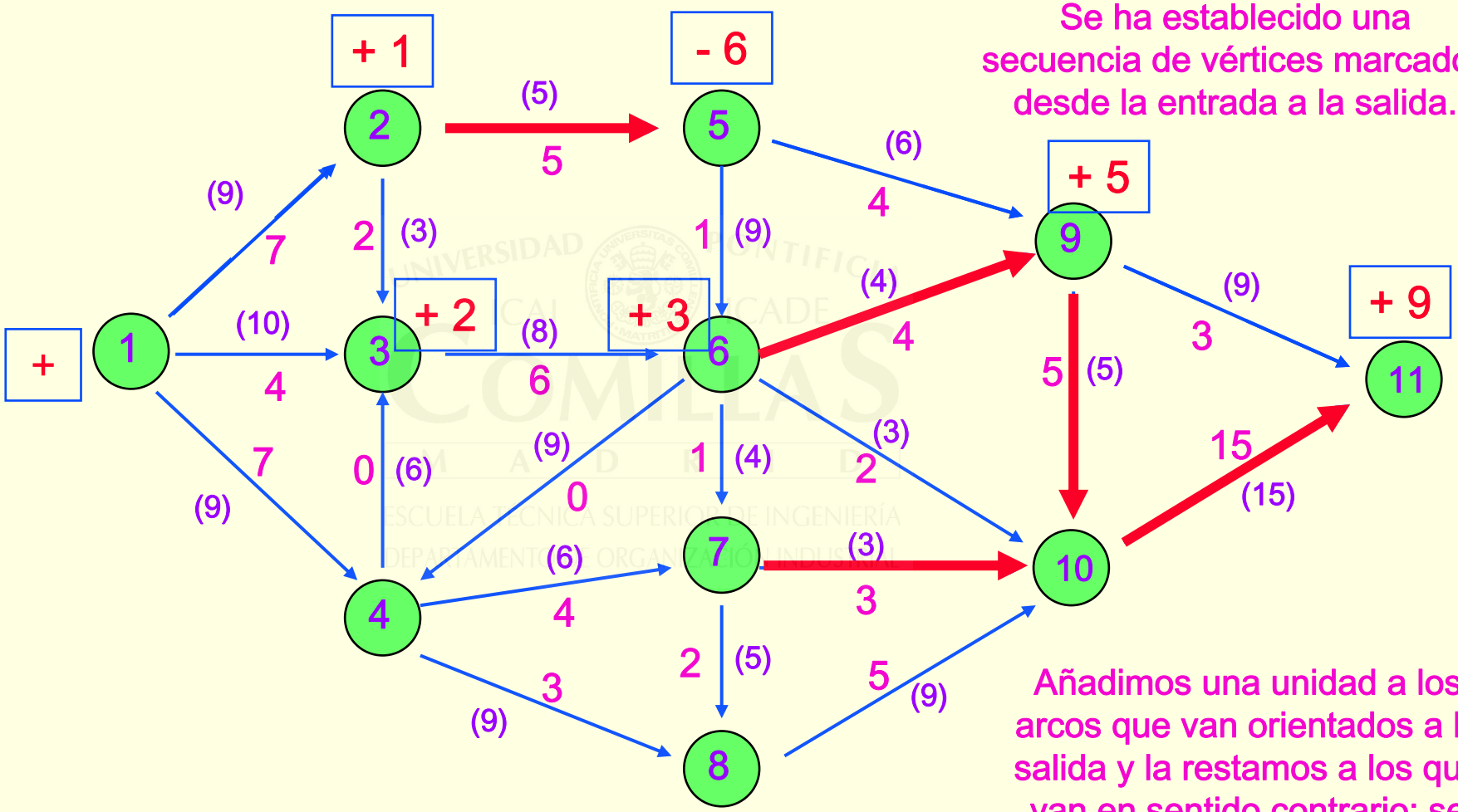
El camino 1-4-8-10-11 no tiene ningún arco saturado. Incrementamos su flujo en tres unidades hasta saturar el arco (10-11)



Es un flujo completo

Ejemplo.

Obtención de un flujo máximo: Fase de marcado de vértices (I)

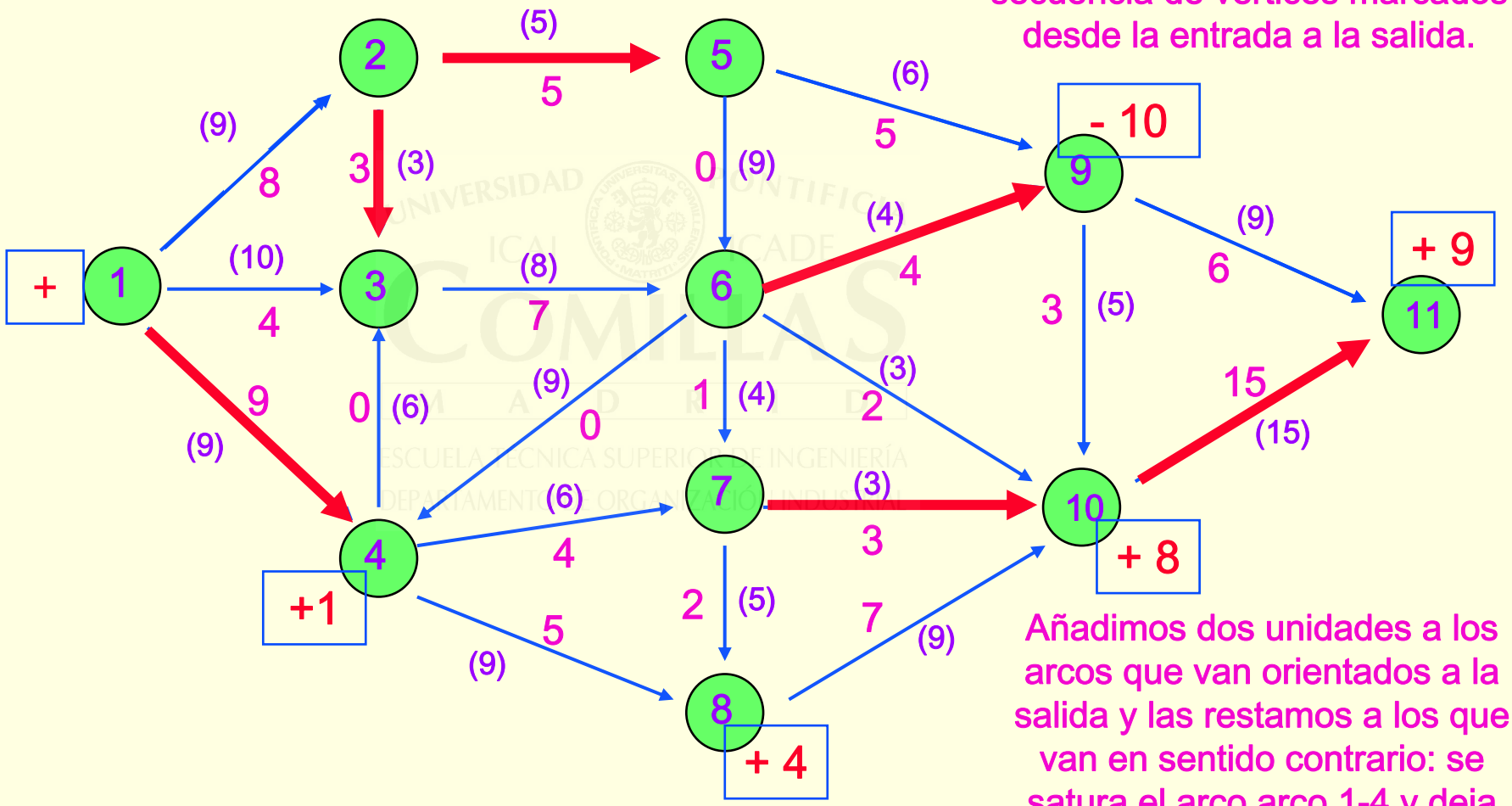


Se ha establecido una secuencia de vértices marcados desde la entrada a la salida.

Añadimos una unidad a los arcos que van orientados a la salida y la restamos a los que van en sentido contrario: se anula el flujo del arco 5-6 y se satura el arco 2-3.

Ejemplo.

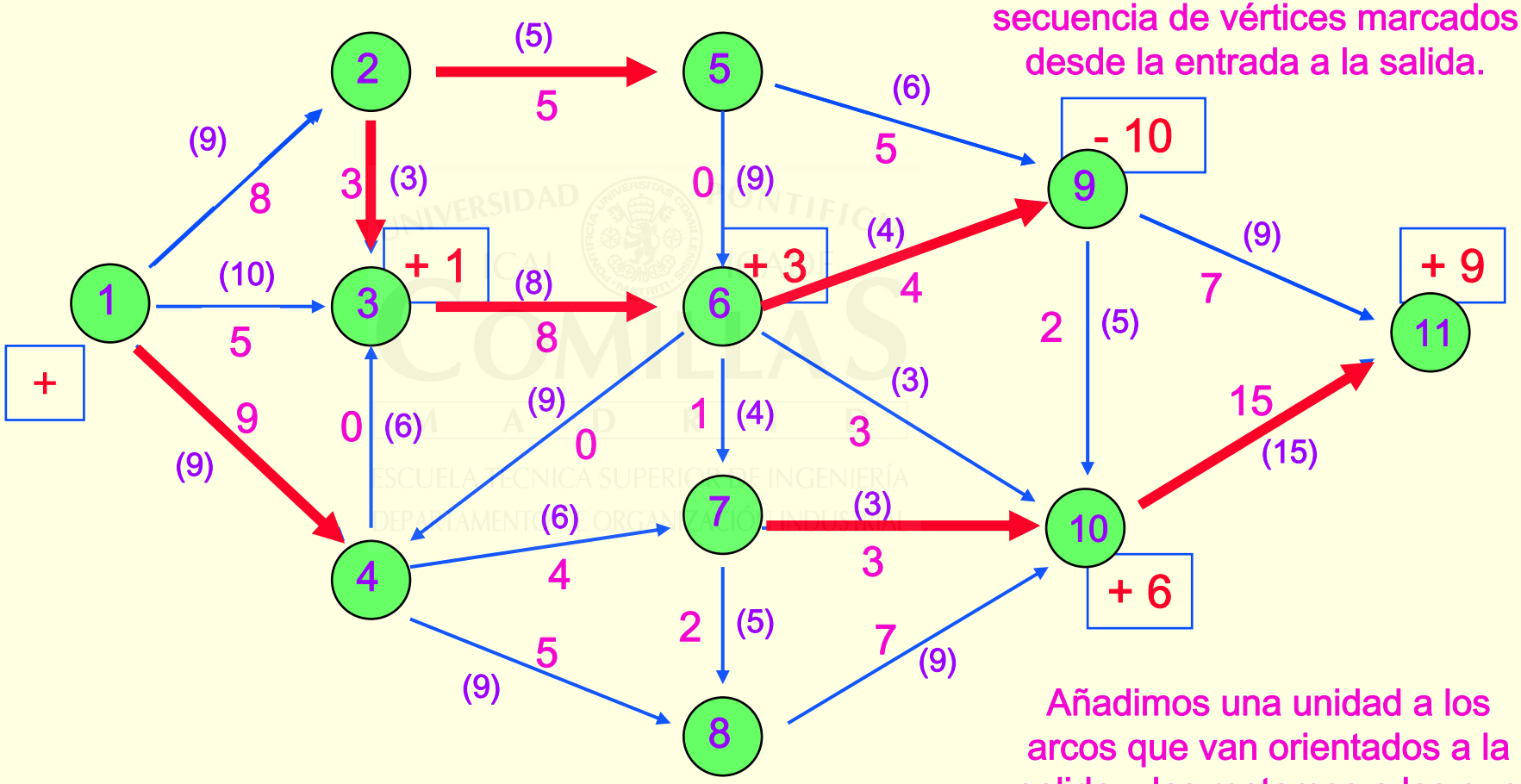
Obtención de un flujo máximo: Fase de marcado de vértices (II)



Añadimos dos unidades a los arcos que van orientados a la salida y las restamos a los que van en sentido contrario: se satura el arco arco 1-4 y deja de serlo el 9-10.

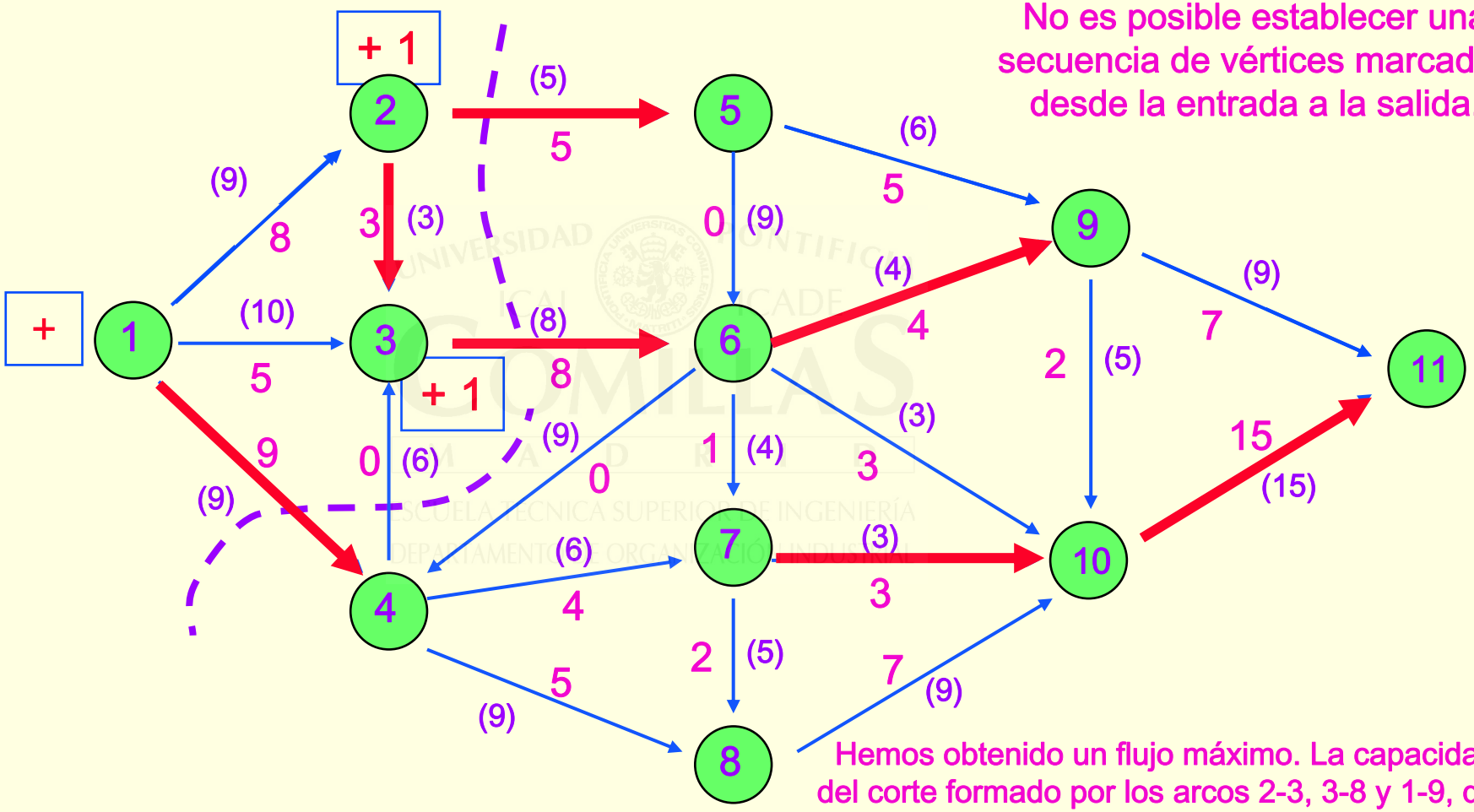
Ejemplo.

Obtención de un flujo máximo: Fase de marcado de vértices (III)



Ejemplo.

Obtención de un flujo máximo: Fase de marcado de vértices (IV)



No es posible establecer una secuencia de vértices marcados desde la entrada a la salida.

Hemos obtenido un flujo máximo. La capacidad del corte formado por los arcos 2-3, 3-8 y 1-9, que unen los vértice marcados con los que no lo están, coincide con el flujo máximo y la mínima capacidad que tiene cualquier corte

Formulación en programación lineal

$$Max\ V$$

$$sujeto\ a:$$

- Limitación de capacidad de los arcos

$$0 \le \varphi_{ij} \le C_{ij}$$

- Continuidad del flujo

$$\sum_{j/(j,i)\in U} \varphi_{ji} - \sum_{j/(i,j)\in U} \varphi_{ij} = 0, i, j = 2, 3, ..., n-1$$

- Cantidad V de flujo que alcanza la salida

$$V = \sum_{j/(j,Salida)\in U} \varphi_{j,salida}$$

Problemas de flujo con coste mínimo

- El objetivo es enviar una cantidad fija de flujo de una fuente s a un sumidero t con coste total mínimo.

- Se resuelve mediante el programa lineal

$$Min \sum_{(i,j)\in U} c_{ij}\varphi_{ij}$$

$$\sum_{i/(f,i)\in U} \varphi_{si} = \theta$$

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

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

ESCUELA TÉCNICA SUPERIOR DE INGENIERÍA
DEPARTAMENTO DE ORGANIZACIÓN INDUSTRIAL

Teoría de Grafos o Redes

Modelado y optimización sobre redes

Arboles de extensión (o expansión) mínima

Árbol de extensión mínima (ó máxima)

Dado un grafo G=(V, E) con pesos en las aristas, se pretende obtener un árbol que conecte a todos los vértices de forma que la suma de los pesos de sus aristas sea mínima.

Algoritmos ávidos, avaros (greedy):

- en cada etapa escogen la elección óptima para esa etapa
- en general son heurísticos
- en el caso de los árboles la solución es óptima

Algoritmo de Prim

C_k: conjunto de nodos conectados en la iteración k

$\overline{C}_k$: nodos no conectados aún

n: número de nodos

T: árbol construido

PASO 1:

$C_0=\emptyset$, $\overline{C}_0=V$. Se toma un vértice cualquiera $i \in \overline{C}_0$

$C_1=\{i\}$, $\overline{C}_1=V - \{i\}$, $k=2$, $T=\emptyset$

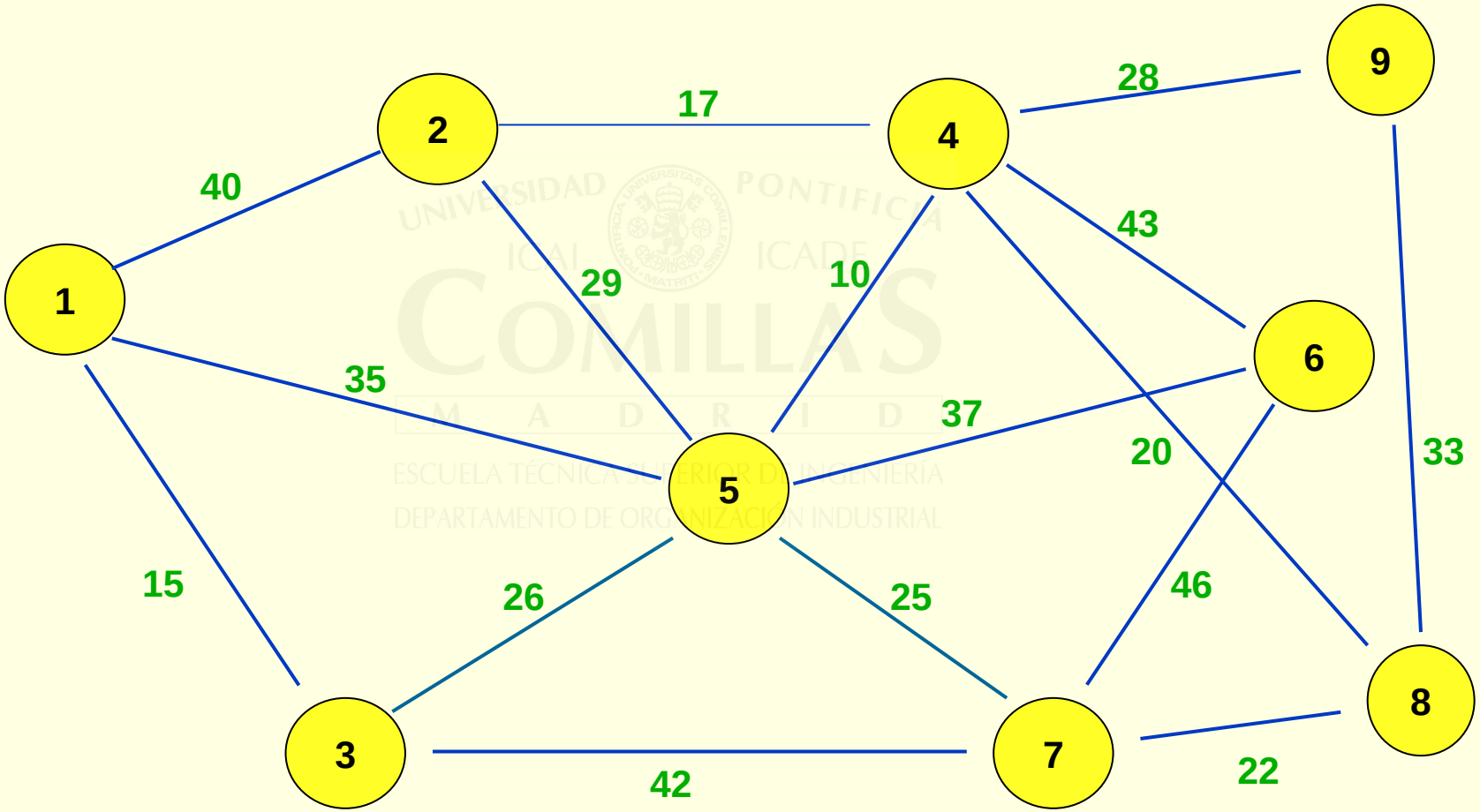
PASO 2:

Seleccionar $j \in \overline{C}_{k-1}$ que sea el que se une a algún vértice de C_{k-1} con la arista de menor peso. Sea este vértice e_{k-1}

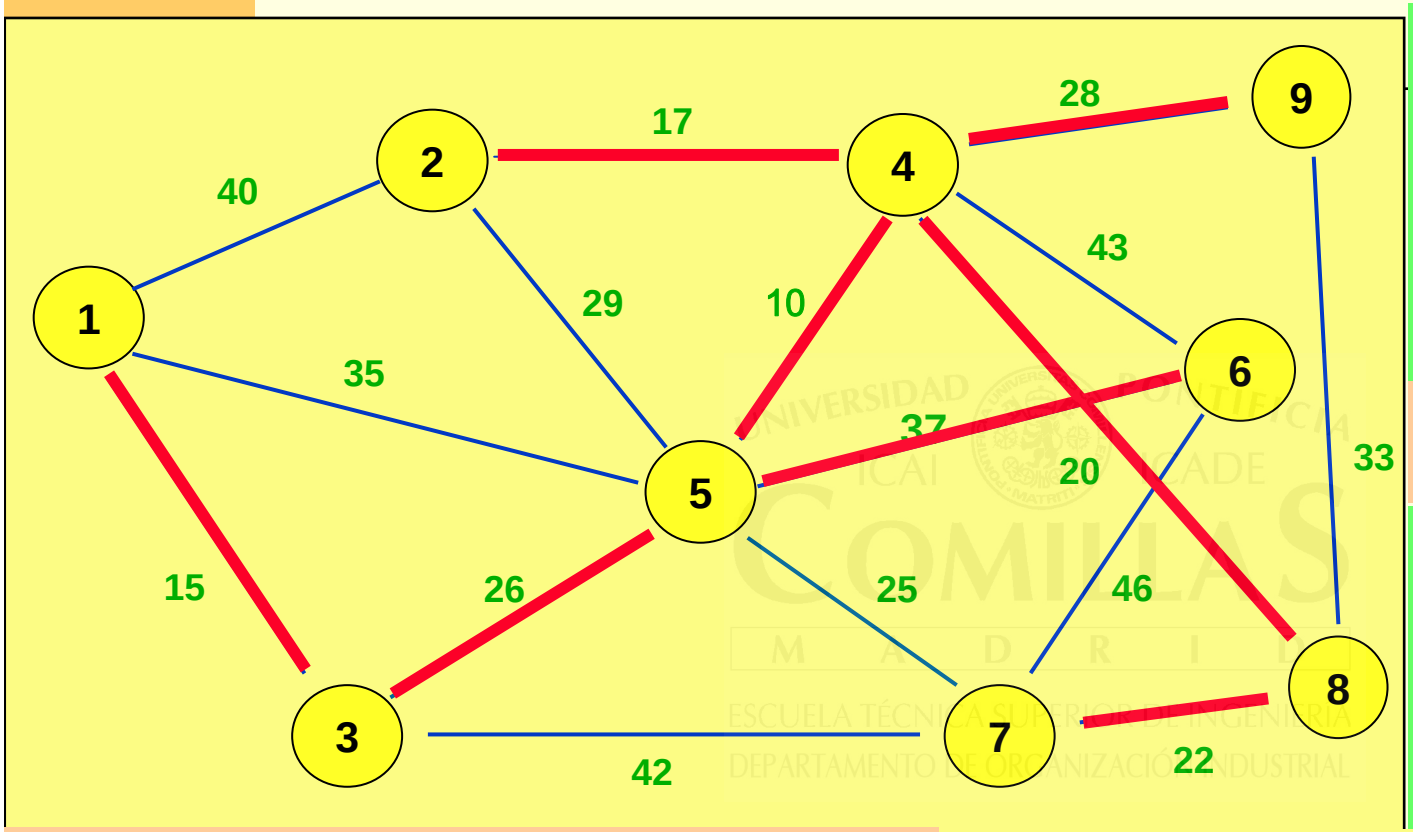
$C_k=C_{k-1} \cup \{j\}$, $\overline{C}_k= \overline{C}_{k-1} - \{j\}$, $T=T \cup \{e_{k-1}\}$

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

Ejemplo: Encontrar el árbol de expansión mínima en la red de la figura.



Ejemplo: Encontrar el árbol de expansión mínima



Seleccionamos un vértice cualquiera; el 5, por ejemplo, y de las aristas que lo tienen por uno de sus extremos la que tiene menor valor asociado: en nuestro caso la arista 5-4, de valor 10.

Esa arista formará parte del árbol buscado

Seleccionamos la arista que, con extremo en 4 ó en 5 tiene el valor menor. Es la arista 2-4, con valor 17. Pasa a integrar el árbol, en el que ya están conectados los vértices 2, 4 y 5.

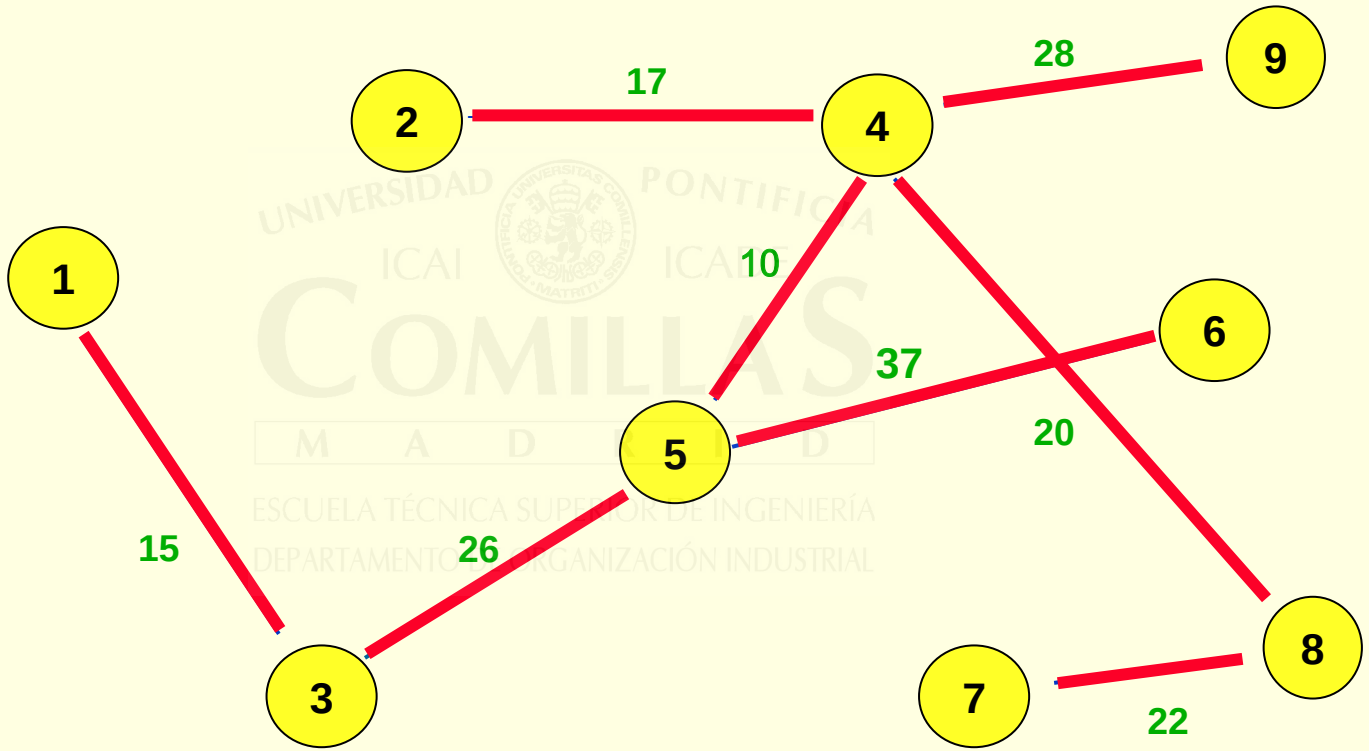
Seguimos buscando, entre las aristas que inciden en 2, 4 ó 5 desde los otros vértices la de menor valor. Es la arista 4-8, de valor 20. La arista es incorporada al árbol y el vértice 8 al conjunto de vértices conectados por el árbol

Ese mecanismo sigue incorporando la arista 7-8 y el vértice 7.

Luego la arista 3-5 y el vértice 3.

Y siguen sucesivamente la arista 1-3 y el vértice 1, la arista 9-4 y el vértice 9 y la arista 6-5 y el vértice 6, con lo cual el árbol queda terminado

Ejemplo: El árbol de expansión mínima es



Algoritmo de Solin (para búsqueda de un mínimo)

- **Paso 1: Seleccionar el elemento de valor mínimo de la matriz de incidencia valorada y marcarlo. Caso de existir varios, se selecciona uno cualquiera.**
- **Paso 2: Eliminar la columna del elemento marcado y marcar la fila cuya número coincide con el de la columna recién eliminada.**
- **Paso 3: De entre los elementos que pertenecen a filas marcadas y no pertenecen a columnas eliminadas se toma el de valor mínimo y se le marca.**
- **Paso 4: Se repiten los pasos 2 y 3 hasta que no se puedan seleccionar mas elementos. Si por este método se consigue marcar todas las filas el problema tiene solución y queda resuelto.**
- **Nota: El algoritmo funciona de forma similar para el caso de máximo, sin más que seleccionar valores máximos, en lugar de mínimos**

Algoritmo de Solin. Ejemplo (1)

Se trata de obtener el árbol de expansión mínima de una red cuya matriz de aristas valoradas es

		1	2	3	4	5	6	7	8
	1	–	20	18	14	19	15	16	13
	2	20	–	16	9	12	11	10	16
1 *	3	18	16	–	8	22	12	13	14
2 *	4	14	9	8	–	13	16	21	17
	5	19	12	22	13	–	10	17	24
	6	15	11	12	16	10	–	13	18
	7	16	10	13	21	17	13	–	14
	8	13	16	14	17	24	18	14	–

Paso 1. Seleccionamos el mínimo elemento de la tabla: el 8, en la casilla (3, 4). Eliminamos las columnas 3 y 4 y marcamos las filas 3 y 4

Como no están marcadas todas las filas, vamos al paso 2.

Algoritmo de Solin. Ejemplo (2)

		1	2	3	4	5	6	7	8
	1	–	20			19	15	16	13
3 *	2	20	–			12	11	10	16
1 *	3	18	16			22	12	13	14
2 *	4	14	9			13	16	21	17
	5	19	12			–	10	17	24
	6	15	11			10	–	13	18
	7	16	10			17	13	–	14
	8	13	16			24	18	14	–

Paso 2. Seleccionamos el mínimo elemento de la subtabla de fondo amarillo : el 9, en la casilla (4, 2). Eliminamos la columna 2 y marcamos la fila 2.

Como no están marcadas todas las filas, vamos al paso 2.

Algoritmo de Solin. Ejemplo (3)

		1	2	3	4	5	6	7	8
	1	–				19	15	16	13
3 *	2	20				12	11	10	16
1 *	3	18				22	12	13	14
2 *	4	14				13	16	21	17
	5	19				–	10	17	24
	6	15				10	–	13	18
4 *	7	16				17	13	–	14
	8	13				24	18	14	–

Paso 2. Seleccionamos el mínimo elemento de la subtabla de fondo amarillo : el 10, en la casilla (2, 7). Eliminamos la columna 7 y marcamos la fila 7.

Como no están marcadas todas las filas, volvemos al paso 2.

Algoritmo de Solin. Ejemplo (4)

		1	2	3	4	5	6	7	8
	1	–				19	15		13
3 *	2	20				12	11		16
1 *	3	18				22	12		14
2 *	4	14				13	16		17
	5	19				–	10		24
5 *	6	15				10	–		18
4 *	7	16				17	13		14
	8	13				24	18		–

Paso 2. Seleccionamos el mínimo elemento de la subtabla de fondo amarillo : el 11, en la casilla (2, 6). Eliminamos la columna 6 y marcamos la fila 6.

Como no están marcadas todas las filas, volvemos al paso 2.

Algoritmo de Solin. Ejemplo (5)

		1	2	3	4	5	6	7	8
	1	–				19			13
3 *	2	20				12			16
1 *	3	18				22			14
2 *	4	14				13			17
6 *	5	19				–			24
5 *	6	15				10			18
4 *	7	16				17			14
	8	13				24			–

Paso 2. Seleccionamos el mínimo elemento de la subtabla de fondo amarillo : el 12, en la casilla (2, 5). Eliminamos la columna 5 y marcamos la fila 5.

Como no están marcadas todas las filas, volvemos al paso 2.

Algoritmo de Solin. Ejemplo (6)

		1	2	3	4	5	6	7	8
	1	–							13
3 *	2	20							16
1 *	3	18							14
2 *	4	14							17
6 *	5	19							24
5 *	6	15							18
4 *	7	16							14
7 *	8	13							–

Paso 2. Seleccionamos el mínimo elemento de la subtabla de fondo amarillo : el 14, de la casilla (7, 8) (aunque hay otros mínimos iguales). Eliminamos la columna 8 y marcamos la fila 8.

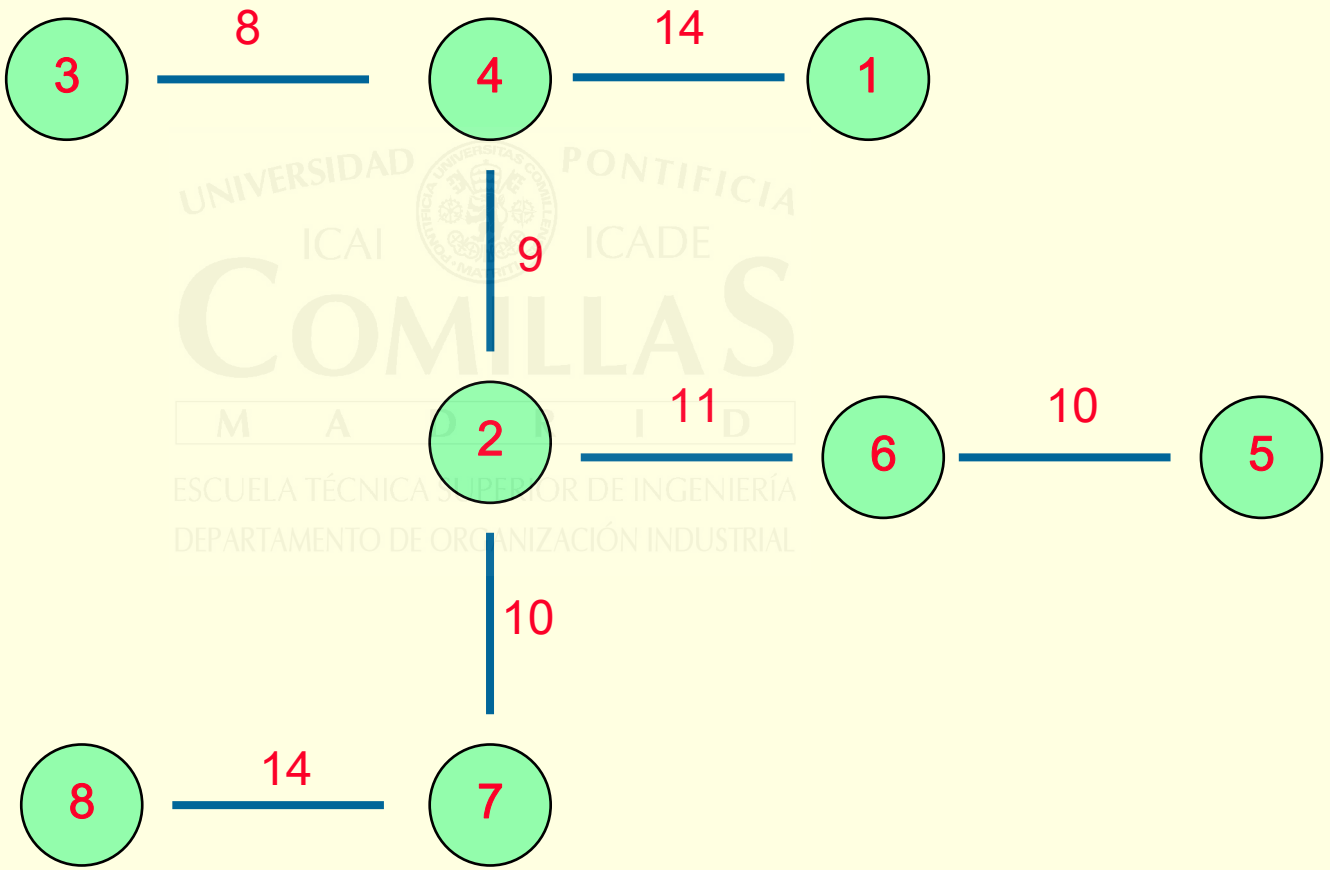
Como no están marcadas todas las filas, volvemos al paso 2.

Algoritmo de Solin. Ejemplo (7)

Paso 2. Seleccionamos el mínimo elemento de la subtabla de fondo amarillo : el 14, de la casilla (4, 1)
Eliminamos la columna 1 y marcamos la fila 1.

Al estar marcadas todas las filas, se ha construido el árbol

Arbol obtenido mediante el algoritmo de Solin



Algoritmo de Kruskal (para búsqueda de un máximo)

- **Paso 1: Seleccionar y marcar la arista de mayor valor. Caso de existir varias, se selecciona una cualquiera.**
- **Paso 2: Eliminar una de las aristas que no estén marcadas y tengan un extremo en una cualquiera de las marcadas, de forma que si se marca la arista seleccionada no se genere un ciclo. De estas aristas se selecciona y marca la asociada al mayor valor.**
- **Paso 3: Reiterar el paso 2 hasta que las aristas seleccionadas conecten a todos los vértices de la red.**
- **Nota: El algoritmo para la obtención de un árbol de expansión mínima es en todo semejante, excepto que se seleccionan aristas de valor mínimo.**

Algoritmo de Kruskal. Ejemplo (1)

Se trata de obtener el árbol de expansión máxima de una red cuya matriz de aristas valoradas es

	1	2	3	4	5	6	7	8
1	–	20	18	14	19	15	16	13
2	20	–	16	9	12	11	10	16
3	18	16	–	8	22	12	13	14
4	14	9	8	–	13	16	21	17
5	19	12	22	13	–	10	17	
6	15	11	12	16	10	–	13	18
7	16	10	13	21	17	13	–	14
8	13	16	14	17		18	14	–
					1 *			2 *

Paso 1.

a) Seleccionamos el mayor elemento de la tabla: el 24, en la casilla (5, 8).

b) Tachamos las casillas (5,8) y (8,5) donde se encuentra ese máximo.

c) Marcamos las columnas 5 y 8 de las que se han eliminado las casillas (5,8) y (8,5)

Como no están marcadas todas las columnas, vamos al paso 2.

Algoritmo de Kruskal. Ejemplo (2)

	1	2	3	4	5	6	7	8
1	–	20	18	14	19	15	16	13
2	20	–	16	9	12	11	10	16
3	18	16	–	8		12	13	
4	14	9	8	–	13	16	21	17
5	19	12		13	–	10	17	
6	15	11	12	16	10	–	13	18
7	16	10	13	21	17	13	–	14
8	13	16		17		18	14	–
			3 *		1 *			2 *

Paso 2.

a) Seleccionamos el mayor elemento de la subtabla de fondo verde : el 22, en la casilla (3, 5).

b) Tachamos las casillas (3,5) y (5,3) donde se encuentra ese máximo y las (3,8) y (8,3) para evitar ciclos.

c) Marcamos la columna 3 de la que se han eliminado las casillas (5,8) y (8,5)

Como no están marcadas todas las columnas, vamos al paso 2.

Algoritmo de Kruskal. Ejemplo (3)

	1	2	3	4	5	6	7	8
1	–	20		14		15	16	
2	20	–	16	9	12	11	10	16
3		16	–	8		12	13	
4	14	9	8	–	13	16	21	17
5		12		13	–	10	17	
6	15	11	12	16	10	–	13	18
7	16	10	13	21	17	13	–	14
8		16		17		18	14	–
	4*		3 *		1 *			2 *

Paso 2.

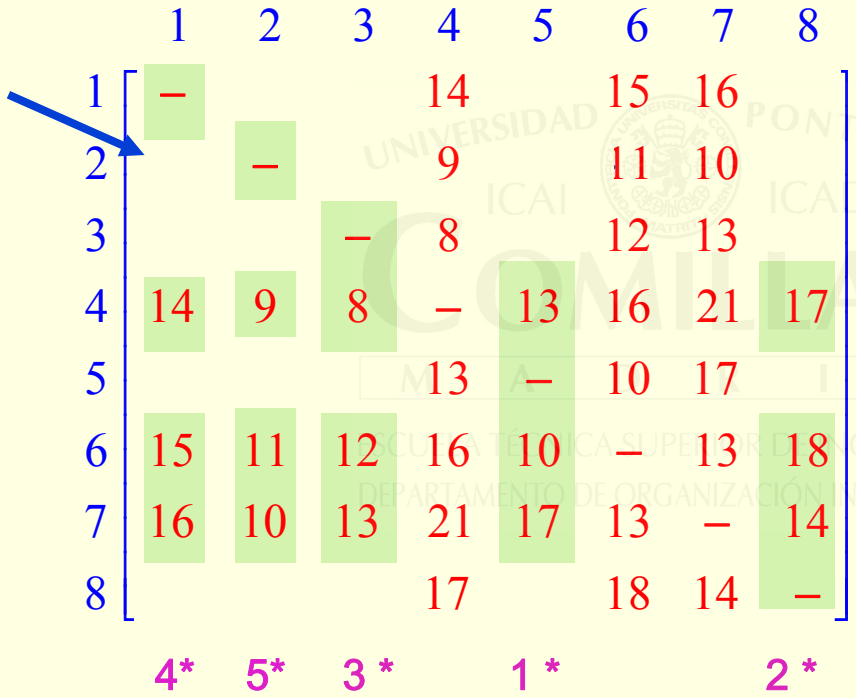
a) Seleccionamos el mayor elemento de la subtabla de fondo verde : el 19, en la casilla (1, 5).

b) Tachamos las casillas (1,5), (1,3) y (1,8), así como las (3,1), (5,1) y (8,1) para evitar ciclos.

c) Marcamos lo que queda de la columna 1

Como no están marcadas todas las columnas, vamos al paso 2.

Algoritmo de Kruskal. Ejemplo (4)



Paso 2.

a) Seleccionamos el mayor elemento de la subtabla de fondo verde : el 20, en la casilla (1, 2).

b) Tachamos las casillas (2,1), (2,3), (2,5) y (2,8) y sus simétricas para evitar ciclos.

c) Marcamos lo que queda de la columna 2

Como no están marcadas todas las columnas, vamos al paso 2.

Algoritmo de Kruskal. Ejemplo (5)

	1	2	3	4	5	6	7	8
1	–			14			16	
2		–		9			10	
3			–	8			13	
4	14	9	8	–	13	16	21	17
5				13	–		17	
6				16		–	13	
7	16	10	13	21	17	13	–	14
8				17			14	–
	4*	5*	3 *		1 *	6*		2 *

Paso 2.

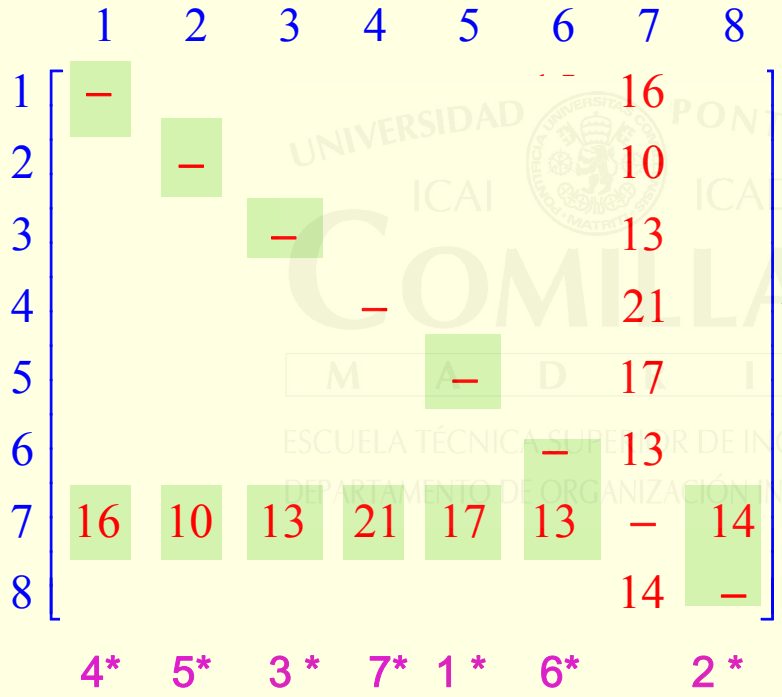
a) Seleccionamos el mayor elemento de la subtabla de fondo verde : el 18, en la casilla (6, 8).

b) Tachamos las casillas (6,1), (6,2), (6,3), (6,5) y (6,8) y sus simétricas para evitar ciclos.

c) Marcamos lo que queda de la columna 6

Como no están marcadas todas las columnas, vamos al paso 2.

Algoritmo de Kruskal. Ejemplo (6)



Paso 2.

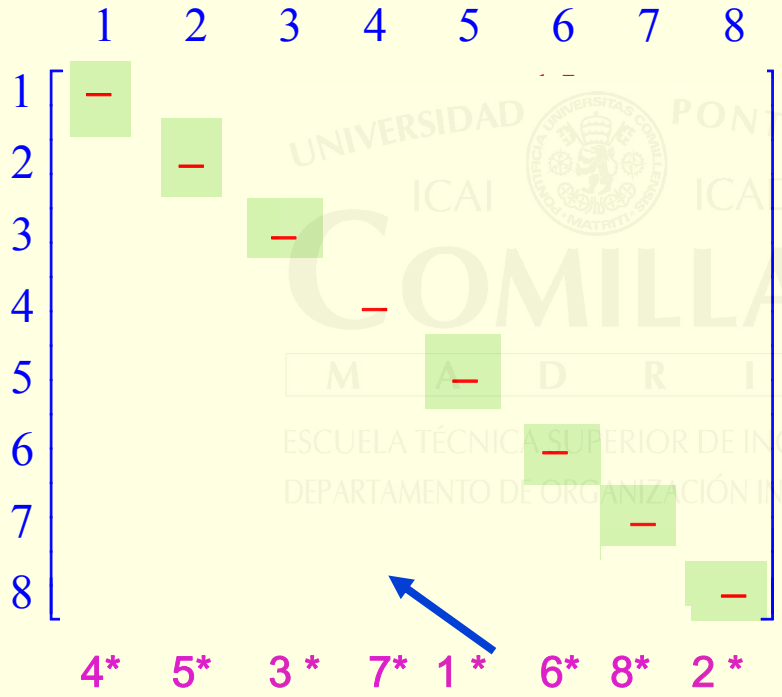
a) Seleccionamos el mayor elemento de la subtabla de fondo verde : el 17, en la casilla (4, 8).

b) Tachamos las casillas (4,1), (4,2), (4,3), (4,5), (4,6) y (4,8) y sus simétricas para evitar ciclos.

c) Marcamos lo que queda de la columna 4

Como no están marcadas todas las columnas, vamos al paso 2.

Algoritmo de Kruskal. Ejemplo (7)



Paso 2.

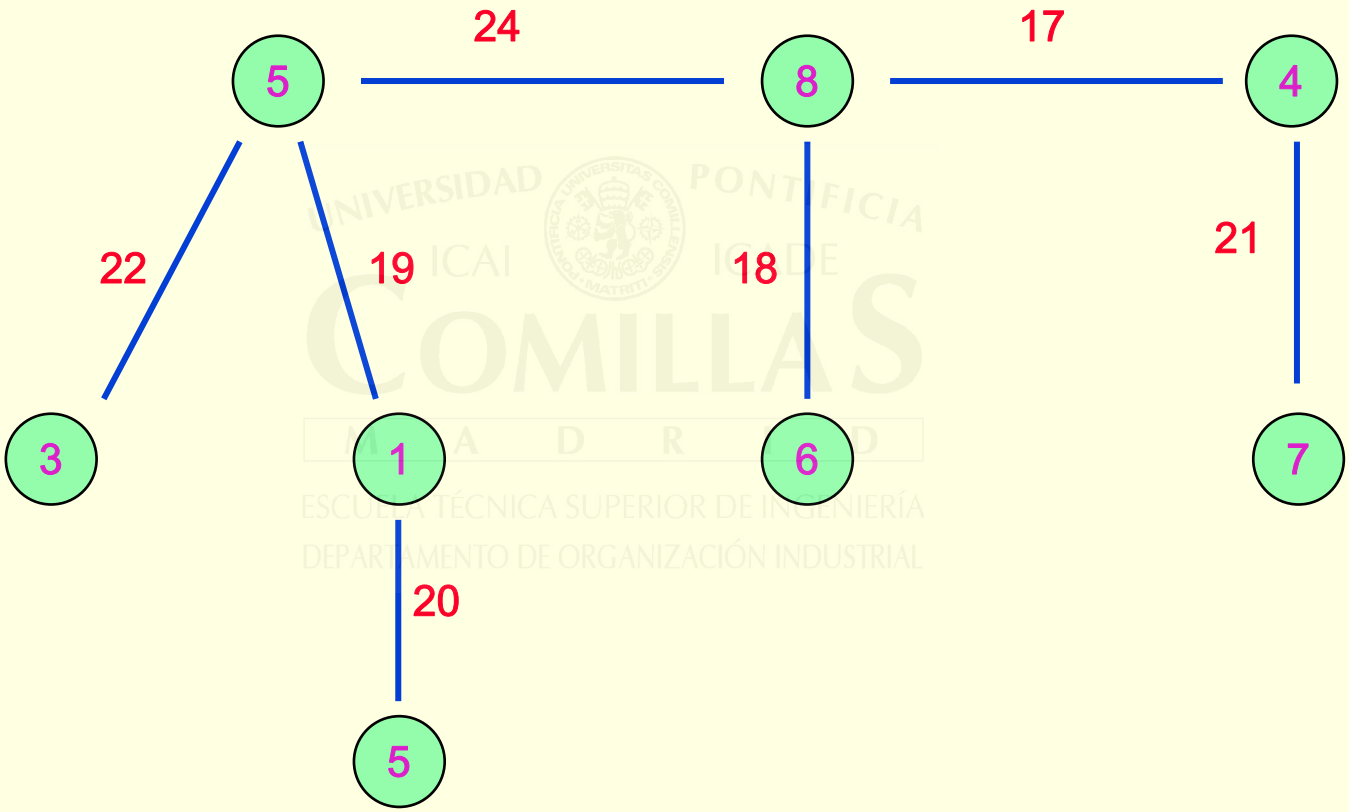
a) Seleccionamos el mayor elemento de la subtabla de fondo verde : el 21, en la casilla (4, 7).

b) Tachamos todas las casillas de la línea 7 y sus simétricas.

c) Marcamos lo que queda de la columna 4

Como están marcadas todas las columnas, hemos obtenido el árbol.

El árbol de expansión máxima



ESCUELA TÉCNICA SUPERIOR DE INGENIERÍA
DEPARTAMENTO DE ORGANIZACIÓN INDUSTRIAL

Teoría de Grafos o Redes

Modelado y optimización sobre redes

Los problemas del viajante de comercio y del cartero chino

Problema del viajante (TSP)

- **Consiste en encontrar el circuito de valor mínimo en un grafo valorado que:**
  - ✓ **visite cada vértice exactamente una vez**
    - **el problema es encontrar un circuito o un ciclo hamiltoniano de longitud mínima**
  - ✓ **visite cada vértice al menos una vez**
    - **en este caso la solución no tiene por que ser un circuito o ciclo simple.**
- **Si el grafo es orientado o dirigido el problema se conoce como TSP asimétrico.**
- **No se ha encontrado ningún algoritmo exacto de complejidad polinomial que lo resuelva.**

Formulación en programación lineal

$$Min \sum_{i,j} c_{i,j} x_{i,j}$$

$$sujeto\ a:$$

$$x_{i,j} = \begin{cases} 1, \text{si en la ruta seleccionada se va de } i \text{ a } j \\ 0, \text{en otro caso} \end{cases}$$

$$0 \le c_{ij} < \infty$$

$$\sum_{j=1}^{n} x_{i,j} = 1, \forall i \; ; \; \sum_{i=1}^{n} x_{i,j} = 1, \forall j$$

Estas últimas condiciones establecen que a cada vértice llega un solo arco y que de cada vértice sale también un solo arco. Para evitar caer en ciclos parciales se asocia a cada vértice una variable v sobre las que se consideran las condiciones adicionales:

$$v_i - v_j + 1 \le n\left(1 - x_{i,j}\right),\ i \ne j,\ i = 2,3,....,n;\ j = 2,3,....,n$$

$$\text{con } v_1 = 1$$

Problema del cartero chino

- Es una variante del programa anterior.
- Se permite pasar más de una vez por cada arista.
- Se busca el recorrido de longitud mínima.
- Si existe un ciclo euleriano, esa es la solución.
- Si no, hay que pasar por alguna arista más de una vez.
- Variantes
  - ✓ Cartero motorizado (grafo dirigido)
  - ✓ Parte del recorrido prefijado
  - ✓ Vuelta a la base cada cierto tiempo

Algoritmo bioptimal para el TSP simétrico

PASO 1: Seleccionar origen s. Elegir un vértice t cuya distancia a s sea mínima. Hacer l=t

PASO 2: Seleccionar t entre los no visitados t.q. d(l,t) sea mínima. Añadir t al recorrido y hacer l=t

Si hay nudos que no estén en el recorrido: PASO 1

Si no, añadir s al recorrido: PASO 2

PASO 3: El recorrido será $x_1, x_2, ..., x_n, x_1$. Longitud L, i=1

PASO 4: j=i+2

PASO 5: Considerar recorrido $x_1, .., x_i, x_j, x_{j-1}, .., x_{i+1}, x_{j+1}, .., x_n$

Si L' < L : Se considera nuevo recorrido: PASO 3

Si L' > L: PASO 6

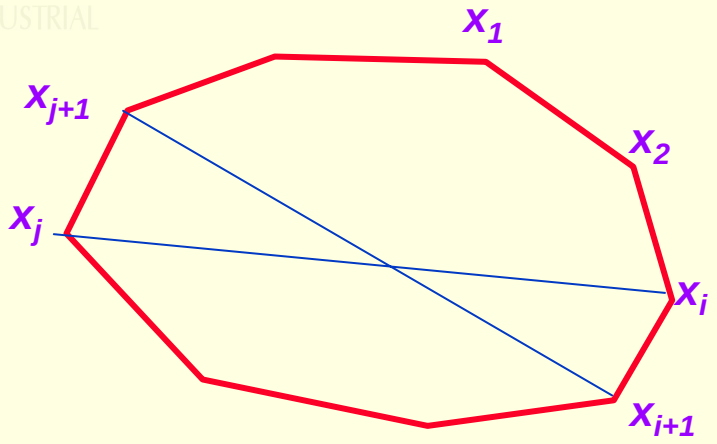
PASO 6: j=j+1

Si j≤n : PASO 5

Si j>n, i=i+1

Si i ≤n-2 : PASO 4

Si i >n-2 : PARAR



Ejemplo (I)

∞	13	12	18	7	14
13	∞	21	26	15	25
12	21	∞	11	6	4
18	26	11	∞	12	14
7	15	6	12	∞	9
14	25	4	14	9	∞

Paso 1: s = 1; t = 1 ; recorrido parcial 1–5

Paso 2: l = 5 ; t = 3 ; recorrido parcial 1-5-3
Paso 2: l = 3 ; t = 6 ; recorrido parcial 1-5-3-6
Paso 2: l = 6 ; t = 4 ; recorrido parcial 1-5-3-6-4
Paso 2: l = 4 ; t = 2 ; recorrido parcial 1-5-3-6-4-2 Todos los vértices visitados

Paso 3: Circuito hamiltoniano 1-5-3-6-4-2-1 . Longitud = 70 ; i = 1

Paso 4: j = 3

Ejemplo (II)

Paso 5: Considerar 1-3-5-6-4-2-1. Longitud = 80. Ir a paso 6

Paso 6: j = 4 < = 6; entonces ir a paso 5

Paso 5: Considerar 1-6-3-5-4-2-1. Longitud = 77. Ir a paso 6

Paso 6: j = 5 < = 6; entonces ir a paso 5

Paso 5: Considerar 1-4-6-3-5-2-1. Longitud = 70. Ir a paso 6

Paso 6: j = 6 < = 6; entonces ir a paso 5

Paso 5: Considerar 1-2-4-6-3-5-1. Longitud = 70. Ir a paso 6

Paso 6: j = 7 > 6; entonces poner i =1+1=2 <= 6-2=4; ir a paso 4

Paso 4: j = 4

Paso 5: Considerar 1-5-6-3-4-2-1. Longitud = 70. Ir a paso 6

Paso 6: j = 5 < = 6; entonces ir a paso 5

Paso 5: Considerar 1-5-4-6-3-2-1. Longitud = 71. Ir a paso 6

Paso 6: j = 6 < = 6; entonces ir a paso 5

Ejemplo (III)

Paso 5: Considerar 1-5-2-4-6-3-1. Longitud = 78. Ir a paso 6

Paso 6: j = 7 > 6; entonces poner i =2+1=3 <= 6-2=4; ir a paso 4

Paso 4: j = 5

Paso 5: Considerar 1-5-3-4-6-2-1. Longitud = 76. Ir a paso 6

Paso 6: j = 6 < = 6; entonces ir a paso 5

Paso 5: Considerar 1-2-4-6-3-5-1. Longitud = 86. Ir a paso 6

Paso 6: j = 7 > 6; entonces poner i =3+1=4 <= 6-2=4; ir a paso 4

Paso 4: j = 6

Paso 5: Considerar 1-5-3-6-2-4-1. Longitud = 86. Ir a paso 6

Paso 6: j = 7 > 6; entonces poner i =4+1=5 > 6-2=4; Parar

El recorrido inicial

1-5-3-6-4-2-1

es un recorrido bioptimal que, al ser el método heurístico, no es seguro sea el circuito hamiltoniano óptimo.

Problema del viajante.
Algoritmo heurístico de Foulkes (I)

Supongamos la siguiente matriz de tiempos de trayecto entre vértices

	A	B	C	D	E	F	G	H
A	–	8	5	3	6	4	3	8
B	4	–	6	7	4	6	5	3
C	4	3	–	2	7	8	3	6
D	13	5	3	–	9	7	5	12
E	5	3	5	2	–	6	3	10
F	7	2	9	6	8	–	8	7
G	8	10	6	7	5	9	–	6
H	2	1	9	3	8	7	5	–

$$a_{i,j} = \begin{cases} 1, \text{si } c_{i,j} \le c_{j,i} \text{ o } i = j \\ 0, \text{en otro caso} \end{cases}$$

	A	B	C	D	E	F	G	H
A	1	0	0	1	0	1	1	0
B	1	1	0	0	0	0	1	0
C	1	1	1	1	0	1	1	1
D	0	1	0	1	0	0	1	0
E	1	1	1	1	1	1	1	0
F	0	1	0	1	0	1	1	0
G	0	0	0	0	0	0	1	0
H	1	1	0	1	1	1	0	1

Asociamos la siguiente matriz

Problema del viajante. Algoritmo de Foulkes (II)

Calculamos sucesivamente las potencias booleanas

$$A^2, A^4, A^8, \ldots.$$

hasta llegar a dos iteraciones sucesivas que proporcionen la misma potencia.
En nuestro caso, eso ocurre en la octava potencia

$$A^8 = \begin{array}{c} \\ A \\ B \\ C \\ D \\ E \\ F \\ G \\ H \end{array} \begin{array}{c} \begin{array}{cccccccc} A & B & C & D & E & F & G & H \end{array} \\ \begin{bmatrix} 1 & 1 & 0 & 1 & 0 & 1 & 1 & 0 \\ 1 & 1 & 0 & 1 & 0 & 1 & 1 & 0 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 0 & 1 & 0 & 1 & 1 & 0 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 0 & 1 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \end{bmatrix} \end{array}$$

Buscamos, si las hay, aquellas líneas que contienen solamente 1´s. Son las de los vértices C, E y H.

Significa que esos vértices son equivalentes en cuanto a preferencia y deben ser anteriores a cualesquiera otros. Constituirán la primera subred fuertemente conexa y de ellos partirán todos los caminos a considerar.

Problema del viajante. Algoritmo de Foulkes (III)

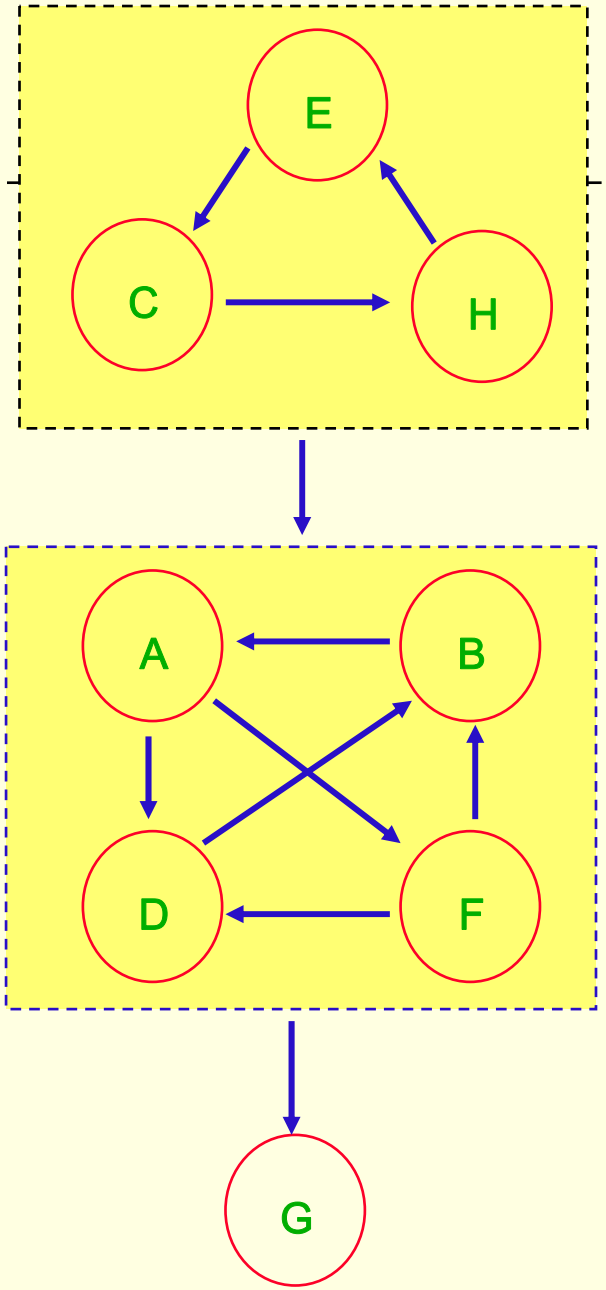
Eliminamos las filas y columnas de esos vértices

	A	*B*	*D*	*F*	*G*
A	1	1	1	1	1
B	1	1	1	1	1
D	1	1	1	1	1
F	1	1	1	1	1
G	0	0	0	0	0

y en la matriz reducida volvemos a seleccionar aquellas filas constituidas por 1´s: todas, excepto la del vértice G. *A*, *B*, *D* y *F* formarán la segunda subred y *G* la tercera.

El problema del viajante. Algoritmo de Foulkes (IV)

- Para obtener todos los caminos hamiltonianos de la red, candidatos a dar el mínimo tiempo, obtenemos los de cada subred, encadenándolos luego si ello es posible.
- En la primera subred obtenemos CHE, HEC y ECH
- En la segunda AFDB, FBAD, BAFD, FDBA y DBAF.
- En la tercera solo G.



El problema del viajante. Algoritmo de Foulkes (V)

- **Como todos los encadenamientos entre caminos de las subredes sucesivas son posibles obtenemos 15 caminos hamiltonianos candidatos a dar el tiempo de trayecto mínimo, tal como se refleja en la tabla.**

- Como ocurre con todo heurístico, el método no garantiza que el camino proporcionado sea el mejor posible.

Camino	Coste
CHEFBADG	6+8+6+2+4+3+5= 34
CHEFDBAG	6+8+6+6+5+4+3= 38
CHEDBAFG	6+8+2+5+4+4+8= 37
CHEAFDBG	6+8+5+4+6+5+5= 39
CHEBAFDG	6+8+5+4+4+6+5= 36
HECFBADG	8+5+8+2+4+3+5= 35
HECFDBAG	8+5+8+6+5+4+3= 39
HECDBAFG	8+5+2+5+4+4+8= 36
HECAFDBG	8+5+4+4+6+5+5= 37
HECBAFDG	8+5+3+4+4+6+5= 35
ECHFBADG	5+6+7+2+4+3+5= 32
ECHFDBAG	5+6+7+6+5+4+3= 36
ECHBDAFG	5+6+3+5+4+4+8= 35
ECHAFDBG	5+6+2+4+6+5+5= 33
ECHBAFDG	5+6+1+4+4+6+5= **31**