# Grafos
# Algoritmo de Floyd-Warshall

Estructuras de Datos

Andrea Rueda

Pontificia Universidad Javeriana
Departamento de Ingeniería de Sistemas

# Algoritmo de Floyd-Warshall

# Algoritmo de Floyd-Warshall

- El algoritmo de Dijkstra, aunque muy eficiente, puede presentar problemas cuando los pesos o factores de conexión son negativos.
- Otra alternativa, más elegante y directa: Algoritmo de Floyd-Warshall.
  - Otros nombres: Floyd, Roy-Warshall, Roy-Floyd, WFI.
  - Búsqueda de rutas más cortas en un grafo.
  - Permite valores negativos para las conexiones.

# Algoritmo de Floyd-Warshall

- Respuesta al problema de encontrar las rutas más cortas entre <u>todos</u> los pares de vértices de un grafo.
- Dado un grafo G = ($V$, E) con una función de pesos $w : E \rightarrow \mathrm{R}$, determinar la longitud de la ruta más corta (distancia) entre todos los pares de vértices en G.

# Algoritmo de Floyd-Warshall

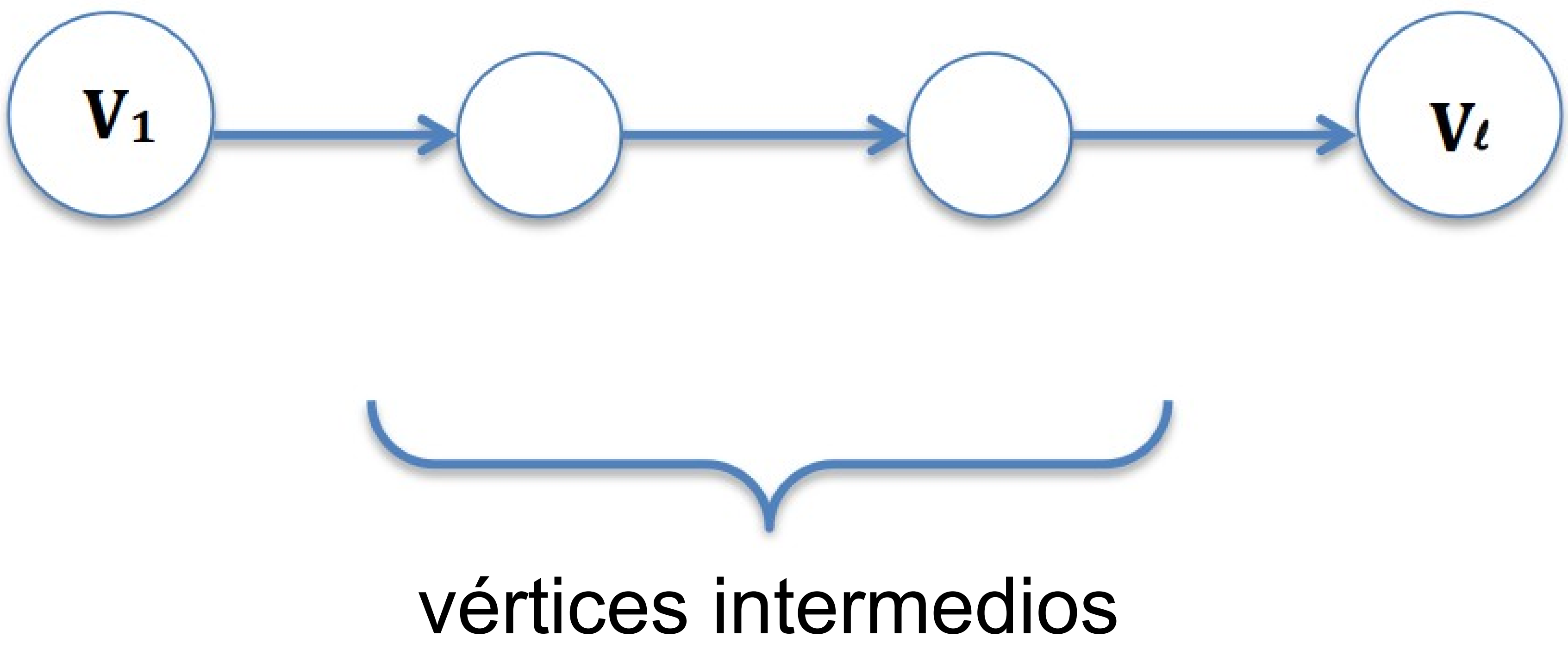
- Caracterización del grafo:

  Matriz de adyacencia, con los pesos de las conexiones en los elementos de la matriz.

$$w_{ij} = \begin{cases} 0 & \text{if } i = j, \\ w(i,j) & \text{if } i \neq j \text{ and } (i,j) \in E, \\ \infty & \text{if } i \neq j \text{ and } (i,j) \notin E. \end{cases}$$

# Algoritmo de Floyd-Warshall

- El algoritmo prueba todas las rutas que pasan entre dos nodos, utilizando un subconjunto de los vértices como vértices intermedios.



# Algoritmo de Floyd-Warshall

- Caracterización del resultado:

  El algoritmo genera una matriz de distancias (costos) del mismo tamaño, $D$ = [$d_{ij}$], donde $d_{ij}$ es la distancia del vértice $i$ al vértice $j$.

  **dist(k,i,j)** : longitud (suma de pesos) de la ruta más corta entre $i$ y $j$ que pasa por el nodo $k$.

# Algoritmo de Floyd-Warshall

La longitud de la ruta más corta se puede definir recursivamente.

$k = 0$ : caso base.

- **dist(0,i,j)** : peso de la arista entre i y j. Infinito si no existe arista.

$k > 0$ : caso general.

- **dist(k,i,j) = min(dist(k-1,i,k) + dist(k-1,k,j), dist(k-1,i,j))**

- "La ruta más corta entre $i$ y $j$ puede pasar por el nodo $k$ o puede no pasar por el nodo $k$".

# Algoritmo de Floyd-Warshall

- Caracterización del resultado: Estructuras (matrices) auxiliares.
  - Longitud de la ruta más corta entre dos nodos (matriz de distancias).

$$d_{ij}^{(k)} = \begin{cases} w_{ij} & \text{if } k = 0\,, \\ \min\left(d_{ij}^{(k-1)}, d_{ik}^{(k-1)} + d_{kj}^{(k-1)}\right) & \text{if } k \geq 1\,. \end{cases}$$

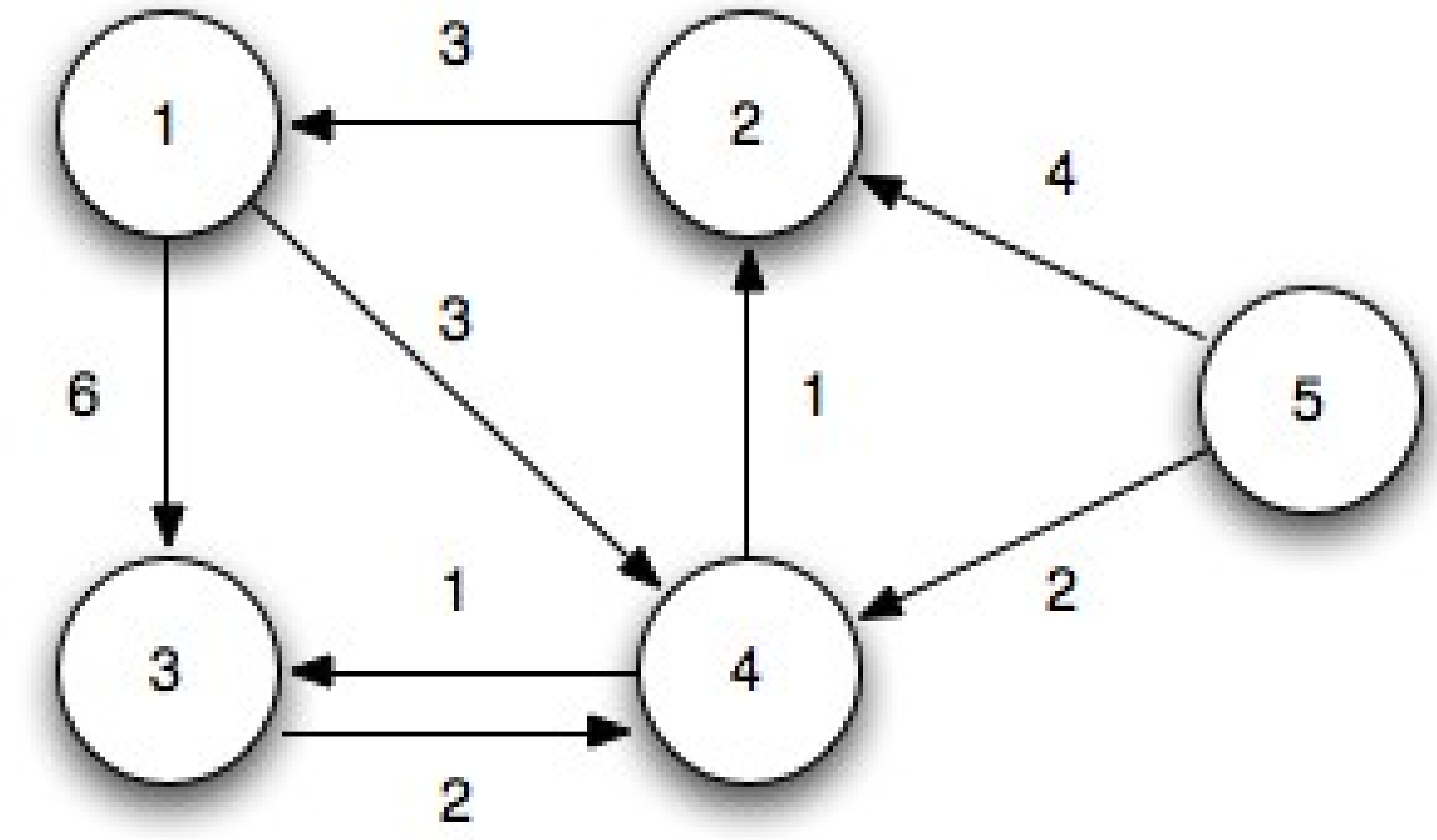
# Algoritmo de Floyd-Warshall

- Caracterización del resultado: Estructuras (matrices) auxiliares.
  - Nodo previo en cada ruta: (matriz de predecesores).

$$\pi_{ij}^{(0)} = \begin{cases} \text{NIL} & \text{if } i = j \text{ or } w_{ij} = \infty , \\ i & \text{if } i \neq j \text{ and } w_{ij} < \infty . \end{cases}$$

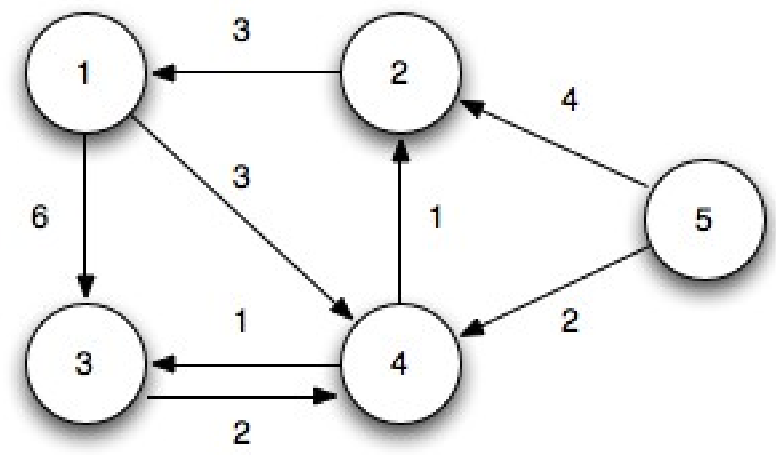
$$\pi_{ij}^{(k)} = \begin{cases} \pi_{ij}^{(k-1)} & \text{if } d_{ij}^{(k-1)} \leq d_{ik}^{(k-1)} + d_{kj}^{(k-1)} , \\ \pi_{kj}^{(k-1)} & \text{if } d_{ij}^{(k-1)} > d_{ik}^{(k-1)} + d_{kj}^{(k-1)} . \end{cases}$$

# Algoritmo de Floyd-Warshall




http://goose.ycp.edu/~dbabcock/PastCourses/cs360/lecture/lecture23.html

# Algoritmo de Floyd-Warshall




k=0

¿Cuáles son las conexiones del grafo?

**D**

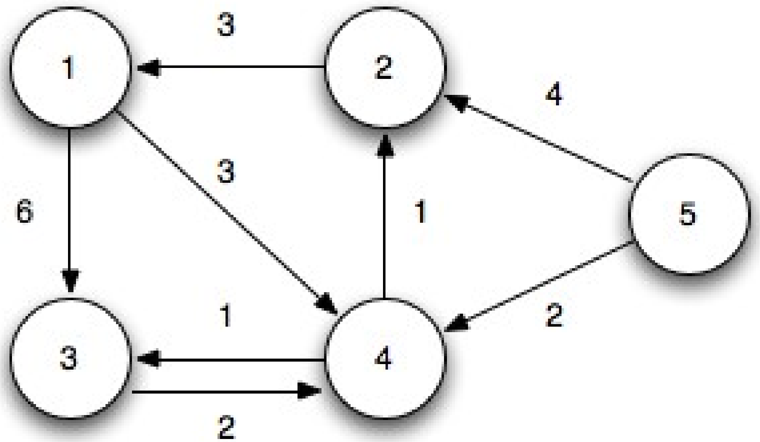
| | | | | |
|---|---|---|---|---|
| 0 | | 6 | 3 | |
| 3 | 0 | | | |
| | | 0 | | |
| | | | 0 | |
| | | | | 0 |

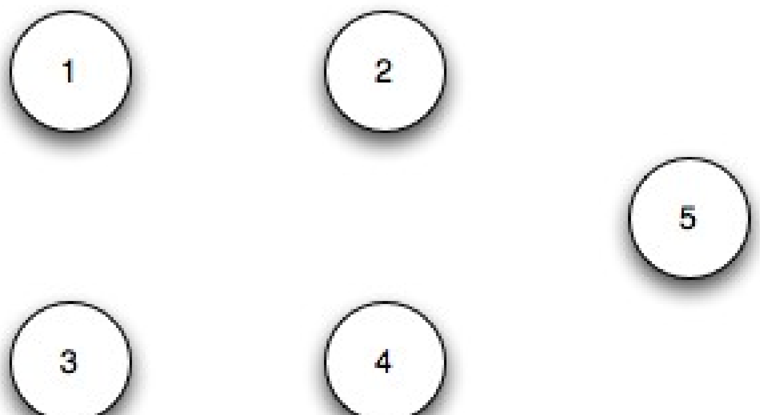
**Π**

| | | | | |
|---|---|---|---|---|
| / | | 1 | 1 | |
| 2 | / | | | |
| | | / | | |
| | | | / | |
| | | | | / |

http://goose.ycp.edu/~dbabcock/PastCourses/cs360/lecture/lecture23.html

# Algoritmo de Floyd-Warshall




**D**

| | 1 | 2 | 3 | 4 | 5 |
|---|---|---|---|---|---|
| 1 | 0 | ∞ | 6 | 3 | ∞ |
| 2 | 3 | 0 | ∞ | ∞ | ∞ |
| 3 | ∞ | ∞ | 0 | 2 | ∞ |
| 4 | ∞ | 1 | 1 | 0 | ∞ |
| 5 | ∞ | 4 | ∞ | 2 | 0 |

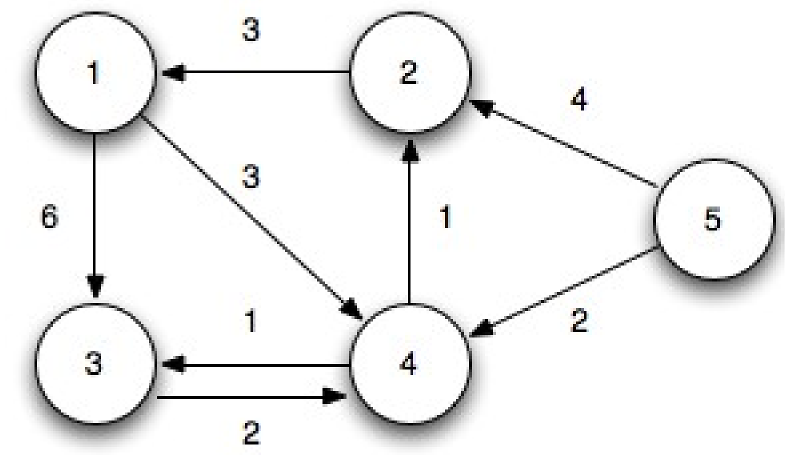
**Π**

| | 1 | 2 | 3 | 4 | 5 |
|---|---|---|---|---|---|
| 1 | / | / | 1 | 1 | / |
| 2 | 2 | / | / | / | / |
| 3 | / | / | / | 3 | / |
| 4 | / | 4 | 4 | / | / |
| 5 | / | 5 | / | 5 | / |

k=0. ¿Cuáles son las conexiones del grafo?

http://goose.ycp.edu/~dbabcock/PastCourses/cs360/lecture/lecture23.html

# Algoritmo de Floyd-Warshall




$$d_{ij}^{(k)} = \begin{cases} w_{ij} & \text{if } k = 0\,, \\ \min\left(d_{ij}^{(k-1)}, d_{ik}^{(k-1)} + d_{kj}^{(k-1)}\right) & \text{if } k \geq 1\,. \end{cases}$$

$$\pi_{ij}^{(k)} = \begin{cases} \pi_{ij}^{(k-1)} & \text{if } d_{ij}^{(k-1)} \leq d_{ik}^{(k-1)} + d_{kj}^{(k-1)}\,, \\ \pi_{kj}^{(k-1)} & \text{if } d_{ij}^{(k-1)} > d_{ik}^{(k-1)} + d_{kj}^{(k-1)}\,. \end{cases}$$

k=1

¿Qué rutas más cortas pasan sólo por el nodo 1?

**D**

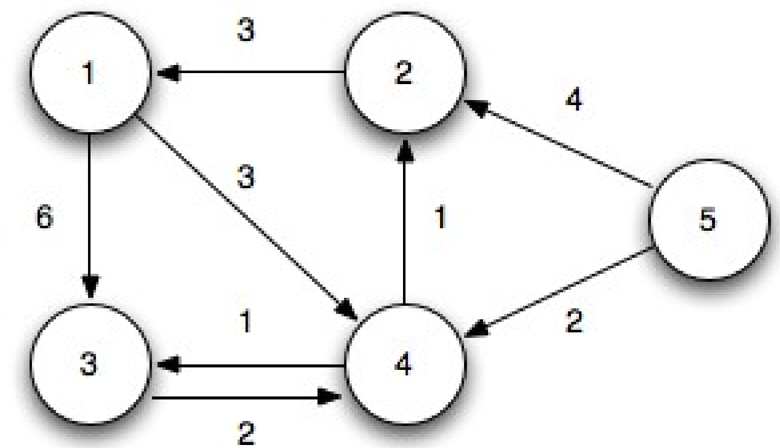
| | 1 | 2 | 3 | 4 | 5 |
|---|---|---|---|---|---|
| 1 | 0 | ∞ | 6 | 3 | ∞ |
| 2 | 3 | 0 | ∞ | ∞ | ∞ |
| 3 | ∞ | ∞ | 0 | 2 | ∞ |
| 4 | ∞ | 1 | 1 | 0 | ∞ |
| 5 | ∞ | 4 | ∞ | 2 | 0 |

**Π**

| | 1 | 2 | 3 | 4 | 5 |
|---|---|---|---|---|---|
| 1 | / | / | 1 | 1 | / |
| 2 | 2 | / | / | / | / |
| 3 | / | / | / | 3 | / |
| 4 | / | 4 | 4 | / | / |
| 5 | / | 5 | / | 5 | / |

http://goose.ycp.edu/~dbabcock/PastCourses/cs360/lecture/lecture23.html

# Algoritmo de Floyd-Warshall




**D**

| | 1 | 2 | 3 | 4 | 5 |
|---|---|---|---|---|---|
| 1 | 0 | ∞ | 6 | 3 | ∞ |
| 2 | 3 | 0 | ∞ | ∞ | ∞ |
| 3 | ∞ | ∞ | 0 | 2 | ∞ |
| 4 | ∞ | 1 | 1 | 0 | ∞ |
| 5 | ∞ | 4 | ∞ | 2 | 0 |

**Π**

| | 1 | 2 | 3 | 4 | 5 |
|---|---|---|---|---|---|
| 1 | / | / | 1 | 1 | / |
| 2 | 2 | / | / | / | / |
| 3 | / | / | / | 3 | / |
| 4 | / | 4 | 4 | / | / |
| 5 | / | 5 | / | 5 | / |

k=1

¿Qué rutas más cortas pasan sólo por el nodo 1?

**D**

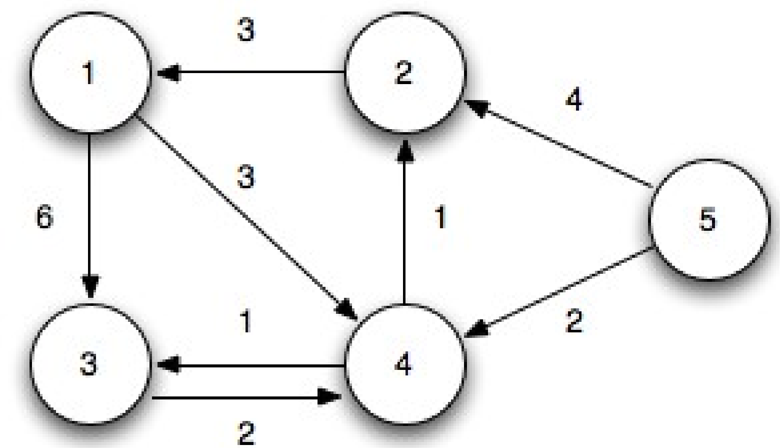
| 1 | 2 | 3 | 4 | 5 |
|---|---|---|---|---|
| 0 | in | 6 | 3 | in |
| 3 | 0 | 9 | 6 | in |
| in | | 0 | | |
| in | | | 0 | |
| in | | | | 0 |

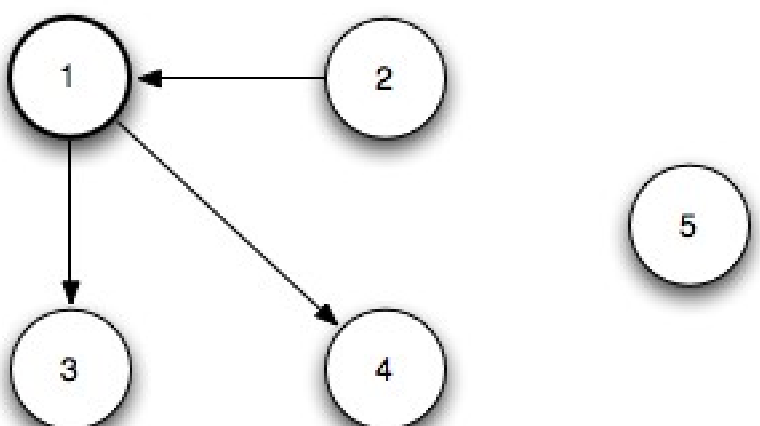
**Π**

| 1 | 2 | 3 | 4 | 5 |
|---|---|---|---|---|
| / | / | 1 | 1 | / |
| 2 | / | 1 | 1 | / |
| / | | / | | |
| / | | | / | |
| / | | | | / |

http://goose.ycp.edu/~dbabcock/PastCourses/cs360/lecture/lecture23.html

# Algoritmo de Floyd-Warshall




**D**

| | 1 | 2 | 3 | 4 | 5 |
|---|---|---|---|---|---|
| 1 | 0 | ∞ | 6 | 3 | ∞ |
| 2 | 3 | 0 | **9** | **6** | ∞ |
| 3 | ∞ | ∞ | 0 | 2 | ∞ |
| 4 | ∞ | 1 | 1 | 0 | ∞ |
| 5 | ∞ | 4 | ∞ | 2 | 0 |

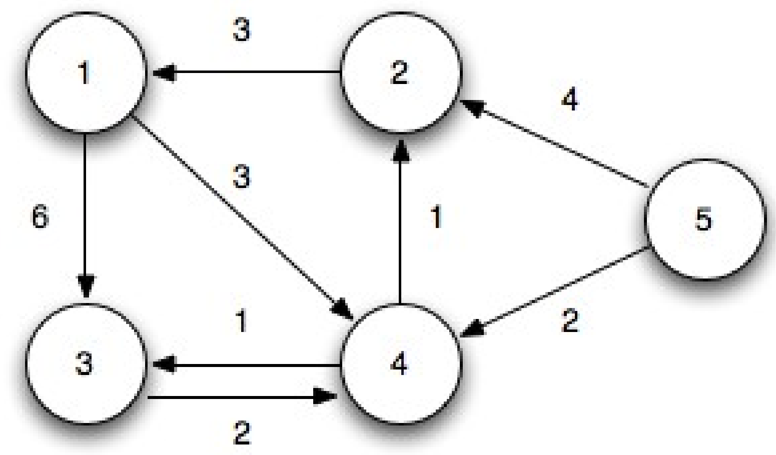
Π

| | 1 | 2 | 3 | 4 | 5 |
|---|---|---|---|---|---|
| 1 | / | / | 1 | 1 | / |
| 2 | 2 | / | **1** | **1** | / |
| 3 | / | / | / | 3 | / |
| 4 | / | 4 | 4 | / | / |
| 5 | / | 5 | / | 5 | / |

k=1. ¿Qué rutas más cortas pasan sólo por el nodo 1?

http://goose.ycp.edu/~dbabcock/PastCourses/cs360/lecture/lecture23.html

# Algoritmo de Floyd-Warshall




$$d_{ij}^{(k)} = \begin{cases} w_{ij} & \text{if } k = 0 , \\ \min\left(d_{ij}^{(k-1)}, d_{ik}^{(k-1)} + d_{kj}^{(k-1)}\right) & \text{if } k \geq 1 . \end{cases}$$

$$\pi_{ij}^{(k)} = \begin{cases} \pi_{ij}^{(k-1)} & \text{if } d_{ij}^{(k-1)} \leq d_{ik}^{(k-1)} + d_{kj}^{(k-1)} , \\ \pi_{kj}^{(k-1)} & \text{if } d_{ij}^{(k-1)} > d_{ik}^{(k-1)} + d_{kj}^{(k-1)} . \end{cases}$$

k=2

¿Qué rutas más cortas pasan por el nodo 2?

**D**

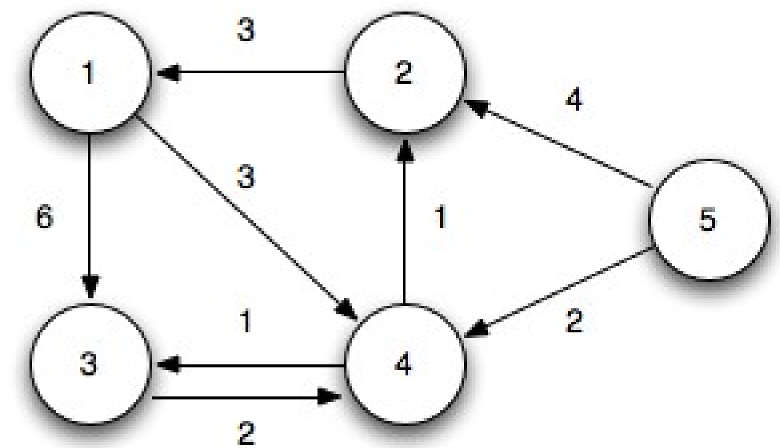
| | 1 | 2 | 3 | 4 | 5 |
|---|---|---|---|---|---|
| 1 | 0 | ∞ | 6 | 3 | ∞ |
| 2 | 3 | 0 | **9** | **6** | ∞ |
| 3 | ∞ | ∞ | 0 | 2 | ∞ |
| 4 | ∞ | 1 | 1 | 0 | ∞ |
| 5 | ∞ | 4 | ∞ | 2 | 0 |

Π

| | 1 | 2 | 3 | 4 | 5 |
|---|---|---|---|---|---|
| 1 | / | / | 1 | 1 | / |
| 2 | 2 | / | **1** | **1** | / |
| 3 | / | / | / | 3 | / |
| 4 | / | 4 | 4 | / | / |
| 5 | / | 5 | / | 5 | / |

http://goose.ycp.edu/~dbabcock/PastCourses/cs360/lecture/lecture23.html

# Algoritmo de Floyd-Warshall




**D**

| | 1 | 2 | 3 | 4 | 5 |
|---|---|---|---|---|---|
| 1 | 0 | ∞ | 6 | 3 | ∞ |
| 2 | 3 | 0 | **9** | **6** | ∞ |
| 3 | ∞ | ∞ | 0 | 2 | ∞ |
| 4 | ∞ | 1 | 1 | 0 | ∞ |
| 5 | ∞ | 4 | ∞ | 2 | 0 |

**Π**

| | 1 | 2 | 3 | 4 | 5 |
|---|---|---|---|---|---|
| 1 | / | / | 1 | 1 | / |
| 2 | 2 | / | **1** | **1** | / |
| 3 | / | / | / | 3 | / |
| 4 | / | 4 | 4 | / | / |
| 5 | / | 5 | / | 5 | / |

k=2

¿Qué rutas más cortas pasan por el nodo 2?

**D**

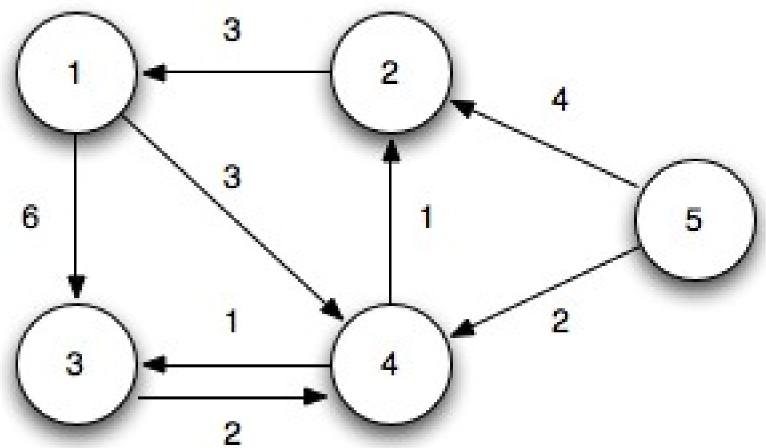
| 1 | 2 | 3 | 4 | 5 |
|---|---|---|---|---|
| 0 | in | 6 | | |
| 3 | 0 | 9 | 6 | in |
| | in | 0 | | |
| 4 | 1 | 1 | 0 | |
| 7 | 4 | | | 0 |

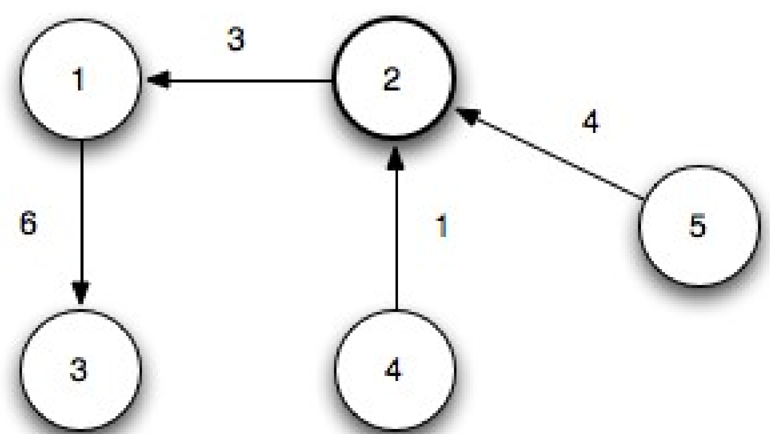
**Π**

| 1 | 2 | 3 | 4 | 5 |
|---|---|---|---|---|
| / | / | 1 | | |
| 2 | / | 1 | 1 | / |
| | / | / | | |
| 2 | 4 | 4 | / | |
| 2 | 5 | | | / |

http://goose.ycp.edu/~dbabcock/PastCourses/cs360/lecture/lecture23.html

# Algoritmo de Floyd-Warshall




**D**

| | 1 | 2 | 3 | 4 | 5 |
|---|---|---|---|---|---|
| 1 | 0 | ∞ | 6 | 3 | ∞ |
| 2 | 3 | 0 | 9 | 6 | ∞ |
| 3 | ∞ | ∞ | 0 | 2 | ∞ |
| 4 | **4** | 1 | 1 | 0 | ∞ |
| 5 | **7** | 4 | **13** | 2 | 0 |

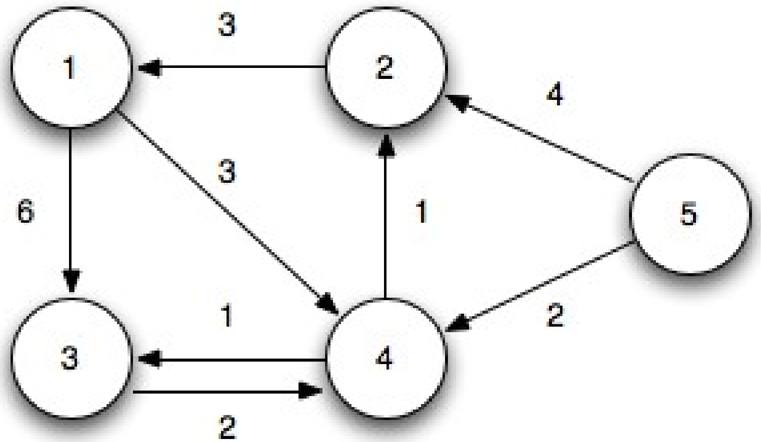
**Π**

| | 1 | 2 | 3 | 4 | 5 |
|---|---|---|---|---|---|
| 1 | / | / | 1 | 1 | / |
| 2 | 2 | / | 1 | 1 | / |
| 3 | / | / | / | 3 | / |
| 4 | **2** | 4 | 4 | / | / |
| 5 | **2** | 5 | **1** | 5 | / |

k=2. ¿Qué rutas más cortas pasan por el nodo 2?

http://goose.ycp.edu/~dbabcock/PastCourses/cs360/lecture/lecture23.html

# Algoritmo de Floyd-Warshall




$$d_{ij}^{(k)} = \begin{cases} w_{ij} & \text{if } k = 0\,, \\ \min\left(d_{ij}^{(k-1)}, d_{ik}^{(k-1)} + d_{kj}^{(k-1)}\right) & \text{if } k \geq 1\,. \end{cases}$$

$$\pi_{ij}^{(k)} = \begin{cases} \pi_{ij}^{(k-1)} & \text{if } d_{ij}^{(k-1)} \leq d_{ik}^{(k-1)} + d_{kj}^{(k-1)}\,, \\ \pi_{kj}^{(k-1)} & \text{if } d_{ij}^{(k-1)} > d_{ik}^{(k-1)} + d_{kj}^{(k-1)}\,. \end{cases}$$

k=3

¿Qué rutas más cortas pasan por el nodo 3?

**D**

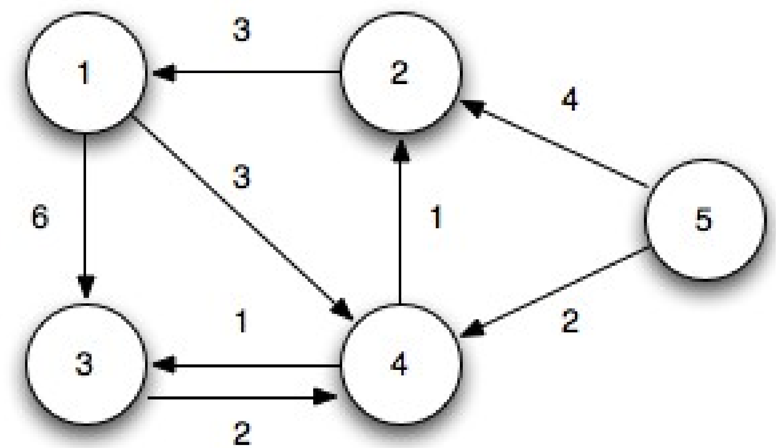
| | 1 | 2 | 3 | 4 | 5 |
|---|---|---|---|---|---|
| 1 | 0 | ∞ | 6 | 3 | ∞ |
| 2 | 3 | 0 | 9 | 6 | ∞ |
| 3 | ∞ | ∞ | 0 | 2 | ∞ |
| 4 | **4** | 1 | 1 | 0 | ∞ |
| 5 | **7** | 4 | **13** | 2 | 0 |

**Π**

| | 1 | 2 | 3 | 4 | 5 |
|---|---|---|---|---|---|
| 1 | / | / | 1 | 1 | / |
| 2 | 2 | / | 1 | 1 | / |
| 3 | / | / | / | 3 | / |
| 4 | **2** | 4 | 4 | / | / |
| 5 | **2** | 5 | 1 | 5 | / |

http://goose.ycp.edu/~dbabcock/PastCourses/cs360/lecture/lecture23.html

# Algoritmo de Floyd-Warshall




**D**

| | 1 | 2 | 3 | 4 | 5 |
|---|---|---|---|---|---|
| 1 | 0 | ∞ | 6 | 3 | ∞ |
| 2 | 3 | 0 | 9 | 6 | ∞ |
| 3 | ∞ | ∞ | 0 | 2 | ∞ |
| 4 | **4** | 1 | 1 | 0 | ∞ |
| 5 | **7** | 4 | **13** | 2 | 0 |

**Π**

| | 1 | 2 | 3 | 4 | 5 |
|---|---|---|---|---|---|
| 1 | / | / | 1 | 1 | / |
| 2 | 2 | / | 1 | 1 | / |
| 3 | / | / | / | 3 | / |
| 4 | **2** | 4 | 4 | / | / |
| 5 | **2** | 5 | **1** | 5 | / |

k=3

¿Qué rutas más cortas pasan por el nodo 3?

**D**

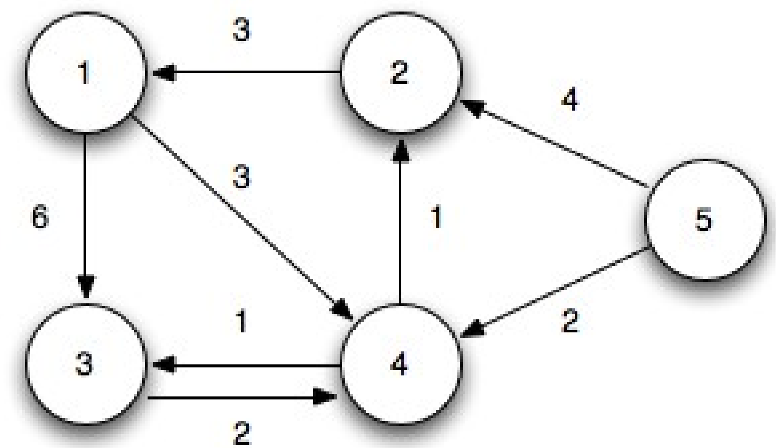
| 1 | 2 | 3 | 4 | 5 |
|---|---|---|---|---|
| 0 | in | 6 | 3 | |
| | 0 | 9 | 6 | |
| in | in | 0 | 2 | in |
| | | 1 | 0 | |
| 7 | | 13 | | 0 |

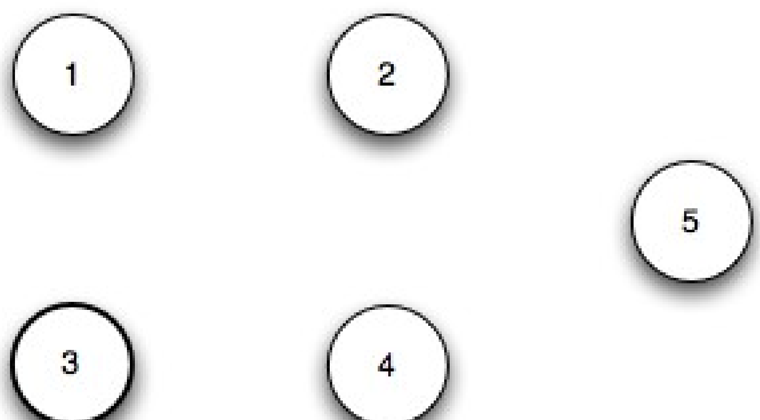
**Π**

| 1 | 2 | 3 | 4 | 5 |
|---|---|---|---|---|
| / | / | 1 | 1 | |
| | / | 1 | 1 | |
| / | / | / | 3 | / |
| | | 4 | / | |
| 2 | | 1 | | / |

http://goose.ycp.edu/~dbabcock/PastCourses/cs360/lecture/lecture23.html

# Algoritmo de Floyd-Warshall




**D**

| | 1 | 2 | 3 | 4 | 5 |
|---|---|---|---|---|---|
| 1 | 0 | ∞ | 6 | 3 | ∞ |
| 2 | 3 | 0 | 9 | 6 | ∞ |
| 3 | ∞ | ∞ | 0 | 2 | ∞ |
| 4 | 4 | 1 | 1 | 0 | ∞ |
| 5 | 7 | 4 | 13 | 2 | 0 |

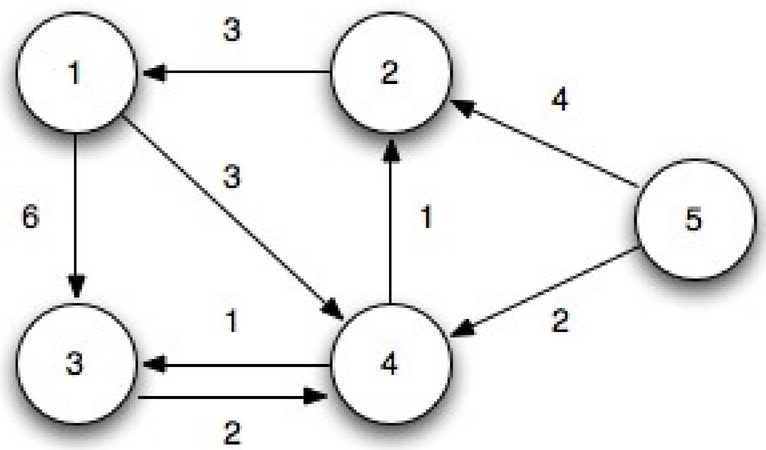
**Π**

| | 1 | 2 | 3 | 4 | 5 |
|---|---|---|---|---|---|
| 1 | / | / | 1 | 1 | / |
| 2 | 2 | / | 1 | 1 | / |
| 3 | / | / | / | 3 | / |
| 4 | 2 | 4 | 4 | / | / |
| 5 | 2 | 5 | 1 | 5 | / |

k=3. ¿Qué rutas más cortas pasan por el nodo 3?

http://goose.ycp.edu/~dbabcock/PastCourses/cs360/lecture/lecture23.html

# Algoritmo de Floyd-Warshall




$$d_{ij}^{(k)} = \begin{cases} w_{ij} & \text{if } k = 0\,, \\ \min\left(d_{ij}^{(k-1)}, d_{ik}^{(k-1)} + d_{kj}^{(k-1)}\right) & \text{if } k \geq 1\,. \end{cases}$$

$$\pi_{ij}^{(k)} = \begin{cases} \pi_{ij}^{(k-1)} & \text{if } d_{ij}^{(k-1)} \leq d_{ik}^{(k-1)} + d_{kj}^{(k-1)}\,, \\ \pi_{kj}^{(k-1)} & \text{if } d_{ij}^{(k-1)} > d_{ik}^{(k-1)} + d_{kj}^{(k-1)}\,. \end{cases}$$

k=4

¿Qué rutas más cortas pasan por el nodo 4?

**D**

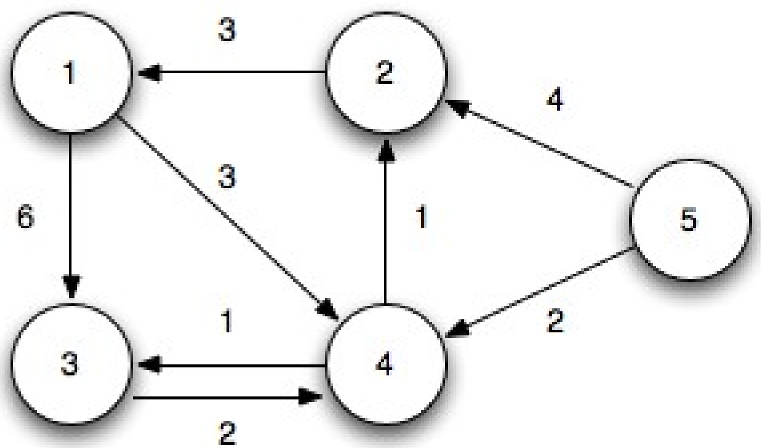
| | 1 | 2 | 3 | 4 | 5 |
|---|---|---|---|---|---|
| 1 | 0 | ∞ | 6 | 3 | ∞ |
| 2 | 3 | 0 | 9 | 6 | ∞ |
| 3 | ∞ | ∞ | 0 | 2 | ∞ |
| 4 | 4 | 1 | 1 | 0 | ∞ |
| 5 | 7 | 4 | 13 | 2 | 0 |

**Π**

| | 1 | 2 | 3 | 4 | 5 |
|---|---|---|---|---|---|
| 1 | / | / | 1 | 1 | / |
| 2 | 2 | / | 1 | 1 | / |
| 3 | / | / | / | 3 | / |
| 4 | 2 | 4 | 4 | / | / |
| 5 | 2 | 5 | 1 | 5 | / |

http://goose.ycp.edu/~dbabcock/PastCourses/cs360/lecture/lecture23.html

# Algoritmo de Floyd-Warshall




D

| | 1 | 2 | 3 | 4 | 5 |
|---|---|---|---|---|---|
| 1 | 0 | ∞ | 6 | 3 | ∞ |
| 2 | 3 | 0 | 9 | 6 | ∞ |
| 3 | ∞ | ∞ | 0 | 2 | ∞ |
| 4 | 4 | 1 | 1 | 0 | ∞ |
| 5 | 7 | 4 | 13 | 2 | 0 |

Π

| | 1 | 2 | 3 | 4 | 5 |
|---|---|---|---|---|---|
| 1 | / | / | 1 | 1 | / |
| 2 | 2 | / | 1 | 1 | / |
| 3 | / | / | / | 3 | / |
| 4 | 2 | 4 | 4 | / | / |
| 5 | 2 | 5 | 1 | 5 | / |

k=4

¿Qué rutas más cortas pasan por el nodo 4?

D

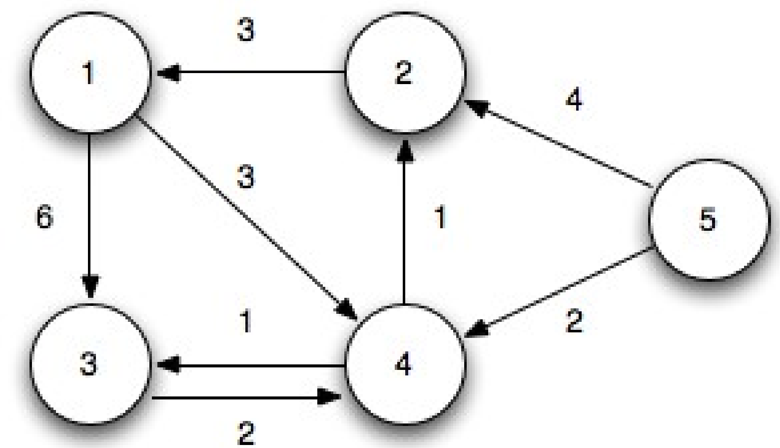
| 1 | 2 | 3 | 4 | 5 |
|---|---|---|---|---|
| 0 | 4 | 4 | 3 | |
| | 0 | 7 | 6 | |
| | | 0 | 2 | |
| 4 | 1 | 1 | 0 | in |
| | | | 2 | 0 |

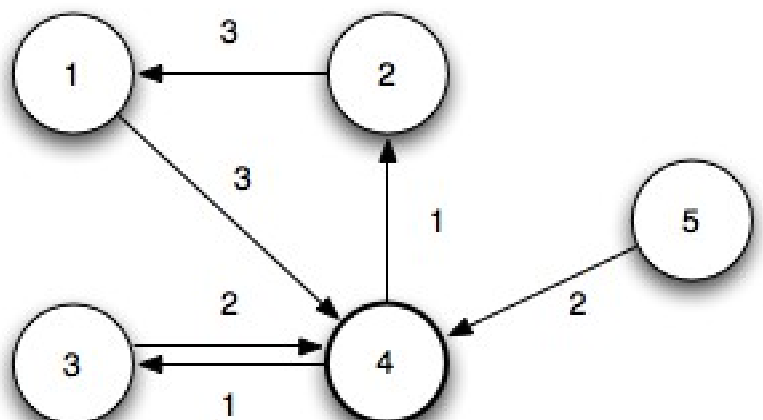
Π

| 1 | 2 | 3 | 4 | 5 |
|---|---|---|---|---|
| / | 4 | 4 | 1 | |
| | / | 4 | 1 | |
| | | / | 3 | |
| 2 | 4 | 4 | / | / |
| | | | 5 | / |

http://goose.ycp.edu/~dbabcock/PastCourses/cs360/lecture/lecture23.html

# Algoritmo de Floyd-Warshall





**D**

| | 1 | 2 | 3 | 4 | 5 |
|---|---|---|---|---|---|
| 1 | 0 | **4** | **4** | 3 | ∞ |
| 2 | 3 | 0 | **7** | 6 | ∞ |
| 3 | **6** | **3** | 0 | 2 | ∞ |
| 4 | 4 | 1 | 1 | 0 | ∞ |
| 5 | **6** | **3** | **3** | 2 | 0 |

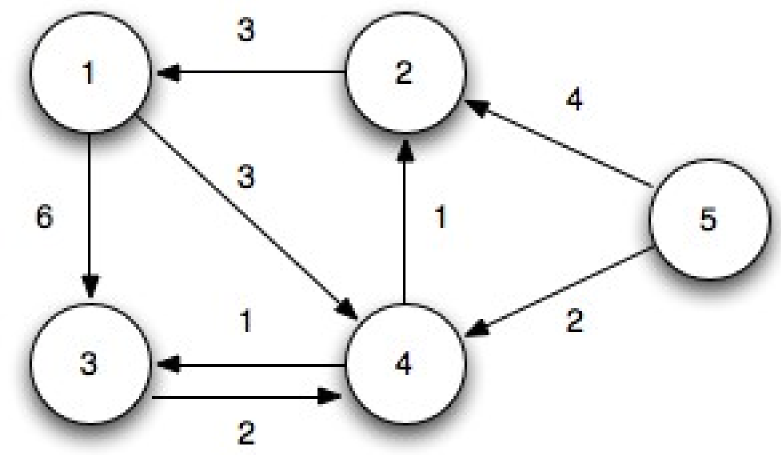
**Π**

| | 1 | 2 | 3 | 4 | 5 |
|---|---|---|---|---|---|
| 1 | / | **4** | **4** | 1 | / |
| 2 | 2 | / | **4** | 1 | / |
| 3 | **2** | **4** | / | 3 | / |
| 4 | 2 | 4 | 4 | / | / |
| 5 | 2 | **4** | **4** | 5 | / |

k=4. ¿Qué rutas más cortas pasan por el nodo 4?

http://goose.ycp.edu/~dbabcock/PastCourses/cs360/lecture/lecture23.html

# Algoritmo de Floyd-Warshall




$$d_{ij}^{(k)} = \begin{cases} w_{ij} & \text{if } k = 0\,, \\ \min\left(d_{ij}^{(k-1)}, d_{ik}^{(k-1)} + d_{kj}^{(k-1)}\right) & \text{if } k \geq 1\,. \end{cases}$$

$$\pi_{ij}^{(k)} = \begin{cases} \pi_{ij}^{(k-1)} & \text{if } d_{ij}^{(k-1)} \leq d_{ik}^{(k-1)} + d_{kj}^{(k-1)}\,, \\ \pi_{kj}^{(k-1)} & \text{if } d_{ij}^{(k-1)} > d_{ik}^{(k-1)} + d_{kj}^{(k-1)}\,. \end{cases}$$

k=5

¿Qué rutas más cortas pasan por el nodo 5?

**D**

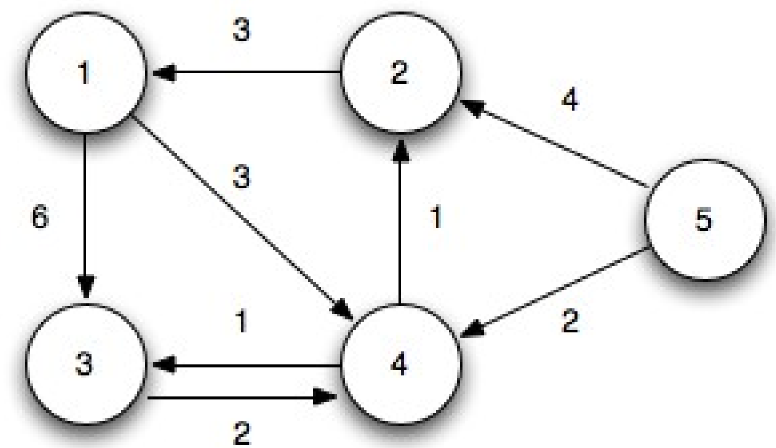
| | 1 | 2 | 3 | 4 | 5 |
|---|---|---|---|---|---|
| 1 | 0 | **4** | **4** | 3 | ∞ |
| 2 | 3 | 0 | **7** | 6 | ∞ |
| 3 | **6** | **3** | 0 | 2 | ∞ |
| 4 | 4 | 1 | 1 | 0 | ∞ |
| 5 | **6** | **3** | **3** | 2 | 0 |

Π

| | 1 | 2 | 3 | 4 | 5 |
|---|---|---|---|---|---|
| 1 | / | **4** | **4** | 1 | / |
| 2 | 2 | / | **4** | 1 | / |
| 3 | **2** | **4** | / | 3 | / |
| 4 | 2 | 4 | 4 | / | / |
| 5 | 2 | **4** | **4** | 5 | / |

http://goose.ycp.edu/~dbabcock/PastCourses/cs360/lecture/lecture23.html

# Algoritmo de Floyd-Warshall




**D**

| | 1 | 2 | 3 | 4 | 5 |
|---|---|---|---|---|---|
| 1 | 0 | **4** | **4** | 3 | ∞ |
| 2 | 3 | 0 | **7** | 6 | ∞ |
| 3 | **6** | **3** | 0 | 2 | ∞ |
| 4 | 4 | 1 | 1 | 0 | ∞ |
| 5 | **6** | **3** | **3** | 2 | 0 |

**Π**

| | 1 | 2 | 3 | 4 | 5 |
|---|---|---|---|---|---|
| 1 | / | **4** | **4** | 1 | / |
| 2 | 2 | / | **4** | 1 | / |
| 3 | **2** | **4** | / | 3 | / |
| 4 | 2 | 4 | 4 | / | / |
| 5 | 2 | **4** | **4** | 5 | / |

k=5

¿Qué rutas más cortas pasan por el nodo 5?

**D**

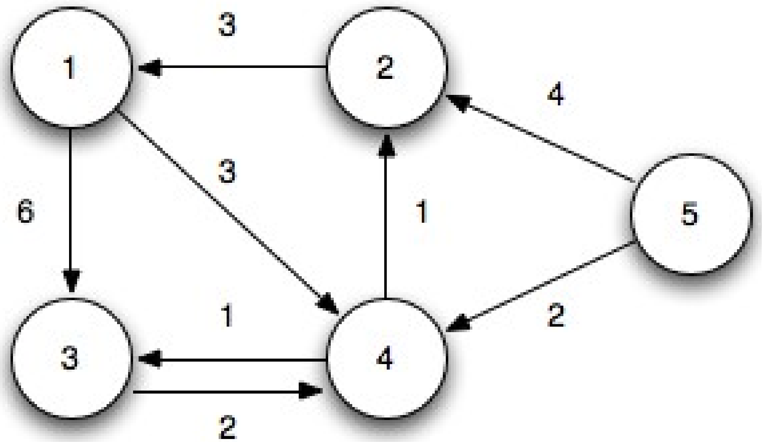
| 1 | 2 | 3 | 4 | 5 |
|---|---|---|---|---|
| 0 | | | | in |
| | 0 | | | in |
| | | 0 | | in |
| | | | 0 | in |
| 6 | 3 | 3 | 2 | 0 |

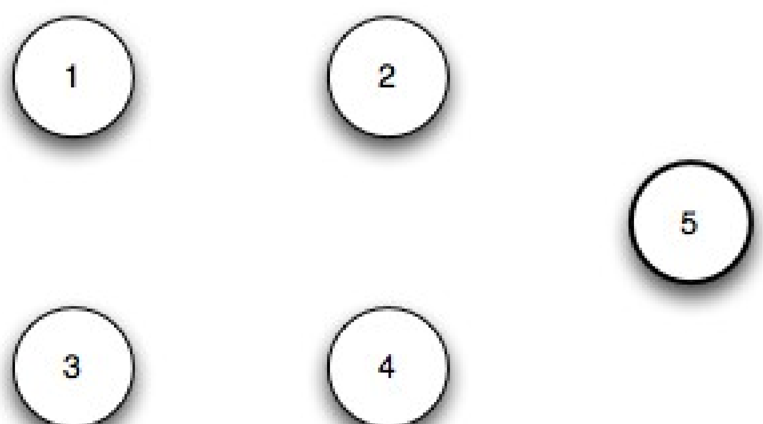
**Π**

| 1 | 2 | 3 | 4 | 5 |
|---|---|---|---|---|
| / | | | | / |
| | / | | | / |
| | | / | | / |
| | | | / | / |
| 2 | 4 | 4 | 5 | / |

http://goose.ycp.edu/~dbabcock/PastCourses/cs360/lecture/lecture23.html

# Algoritmo de Floyd-Warshall




**D**

| | 1 | 2 | 3 | 4 | 5 |
|---|---|---|---|---|---|
| 1 | 0 | 4 | 4 | 3 | ∞ |
| 2 | 3 | 0 | 7 | 6 | ∞ |
| 3 | 6 | 3 | 0 | 2 | ∞ |
| 4 | 4 | 1 | 1 | 0 | ∞ |
| 5 | 6 | 3 | 3 | 2 | 0 |

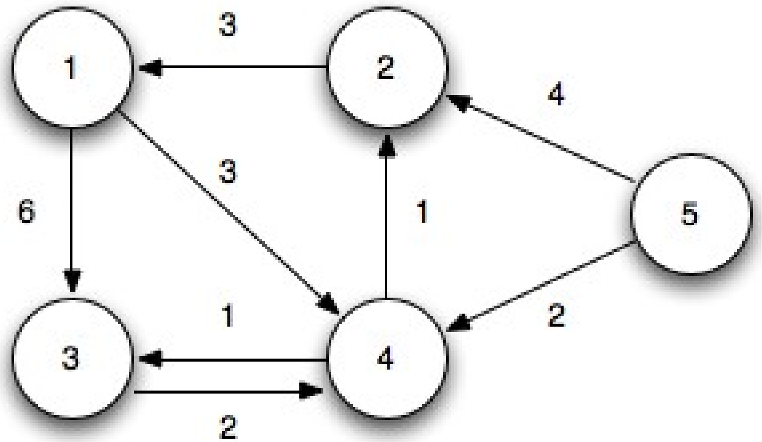
**Π**

| | 1 | 2 | 3 | 4 | 5 |
|---|---|---|---|---|---|
| 1 | / | 4 | 4 | 1 | / |
| 2 | 2 | / | 4 | 1 | / |
| 3 | 2 | 4 | / | 3 | / |
| 4 | 2 | 4 | 4 | / | / |
| 5 | 2 | 4 | 4 | 5 | / |

k=5. ¿Qué rutas más cortas pasan por el nodo 5?

http://goose.ycp.edu/~dbabcock/PastCourses/cs360/lecture/lecture23.html

# Algoritmo de Floyd-Warshall




**D**

| | 1 | 2 | 3 | 4 | 5 |
|---|---|---|---|---|---|
| 1 | 0 | 4 | 4 | 3 | ∞ |
| 2 | 3 | 0 | 7 | 6 | ∞ |
| 3 | 6 | 3 | 0 | 2 | ∞ |
| 4 | 4 | 1 | 1 | 0 | ∞ |
| 5 | 6 | 3 | 3 | 2 | 0 |

Π

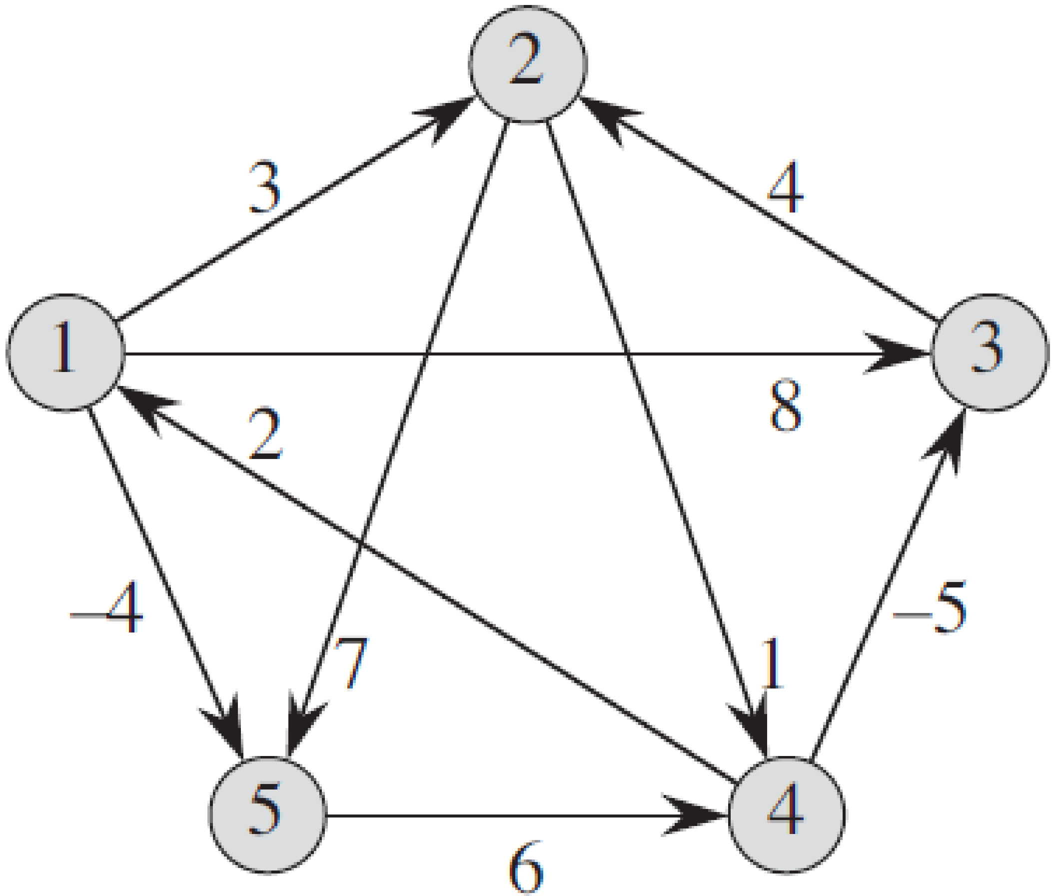
| | 1 | 2 | 3 | 4 | 5 |
|---|---|---|---|---|---|
| 1 | / | 4 | 4 | 1 | / |
| 2 | 2 | / | 4 | 1 | / |
| 3 | 2 | 4 | / | 3 | / |
| 4 | 2 | 4 | 4 | / | / |
| 5 | 2 | 4 | 4 | 5 | / |

Resultado final

http://goose.ycp.edu/~dbabcock/PastCourses/cs360/lecture/lecture23.html

# Algoritmo de Floyd-Warshall

- Ejercicio



# Algoritmo de Floyd-Warshall

k=0

$$D^{(0)} = \begin{pmatrix} 0 & 3 & 8 & \infty & -4 \\ \infty & 0 & \infty & 1 & 7 \\ \infty & 4 & 0 & \infty & \infty \\ 2 & \infty & -5 & 0 & \infty \\ \infty & \infty & \infty & 6 & 0 \end{pmatrix}$$

$$\Pi^{(0)} = \begin{pmatrix} \text{NIL} & 1 & 1 & \text{NIL} & 1 \\ \text{NIL} & \text{NIL} & \text{NIL} & 2 & 2 \\ \text{NIL} & 3 & \text{NIL} & \text{NIL} & \text{NIL} \\ 4 & \text{NIL} & 4 & \text{NIL} & \text{NIL} \\ \text{NIL} & \text{NIL} & \text{NIL} & 5 & \text{NIL} \end{pmatrix}$$

# Algoritmo de Floyd-Warshall

k=1

$$D^{(1)} = \begin{pmatrix} 0 & 3 & 8 & \infty & -4 \\ \infty & 0 & \infty & 1 & 7 \\ \infty & 4 & 0 & \infty & \infty \\ 2 & 5 & -5 & 0 & -2 \\ \infty & \infty & \infty & 6 & 0 \end{pmatrix}$$

$$\Pi^{(1)} = \begin{pmatrix} \text{NIL} & 1 & 1 & \text{NIL} & 1 \\ \text{NIL} & \text{NIL} & \text{NIL} & 2 & 2 \\ \text{NIL} & 3 & \text{NIL} & \text{NIL} & \text{NIL} \\ 4 & 1 & 4 & \text{NIL} & 1 \\ \text{NIL} & \text{NIL} & \text{NIL} & 5 & \text{NIL} \end{pmatrix}$$

# Algoritmo de Floyd-Warshall

k=2

$$D^{(2)} = \begin{pmatrix} 0 & 3 & 8 & 4 & -4 \\ \infty & 0 & \infty & 1 & 7 \\ \infty & 4 & 0 & 5 & 11 \\ 2 & 5 & -5 & 0 & -2 \\ \infty & \infty & \infty & 6 & 0 \end{pmatrix}$$

$$\Pi^{(2)} = \begin{pmatrix} \text{NIL} & 1 & 1 & 2 & 1 \\ \text{NIL} & \text{NIL} & \text{NIL} & 2 & 2 \\ \text{NIL} & 3 & \text{NIL} & 2 & 2 \\ 4 & 1 & 4 & \text{NIL} & 1 \\ \text{NIL} & \text{NIL} & \text{NIL} & 5 & \text{NIL} \end{pmatrix}$$

# Algoritmo de Floyd-Warshall

k=3

$$D^{(3)} = \begin{pmatrix} 0 & 3 & 8 & 4 & -4 \\ \infty & 0 & \infty & 1 & 7 \\ \infty & 4 & 0 & 5 & 11 \\ 2 & -1 & -5 & 0 & -2 \\ \infty & \infty & \infty & 6 & 0 \end{pmatrix}$$

$$\Pi^{(3)} = \begin{pmatrix} \text{NIL} & 1 & 1 & 2 & 1 \\ \text{NIL} & \text{NIL} & \text{NIL} & 2 & 2 \\ \text{NIL} & 3 & \text{NIL} & 2 & 2 \\ 4 & 3 & 4 & \text{NIL} & 1 \\ \text{NIL} & \text{NIL} & \text{NIL} & 5 & \text{NIL} \end{pmatrix}$$

# Algoritmo de Floyd-Warshall

k=4

$$D^{(4)} = \begin{pmatrix} 0 & 3 & -1 & 4 & -4 \\ 3 & 0 & -4 & 1 & -1 \\ 7 & 4 & 0 & 5 & 3 \\ 2 & -1 & -5 & 0 & -2 \\ 8 & 5 & 1 & 6 & 0 \end{pmatrix}$$

$$\Pi^{(4)} = \begin{pmatrix} \text{NIL} & 1 & 4 & 2 & 1 \\ 4 & \text{NIL} & 4 & 2 & 1 \\ 4 & 3 & \text{NIL} & 2 & 1 \\ 4 & 3 & 4 & \text{NIL} & 1 \\ 4 & 3 & 4 & 5 & \text{NIL} \end{pmatrix}$$

# Algoritmo de Floyd-Warshall

k=5

$$D^{(5)} = \begin{pmatrix} 0 & 1 & -3 & 2 & -4 \\ 3 & 0 & -4 & 1 & -1 \\ 7 & 4 & 0 & 5 & 3 \\ 2 & -1 & -5 & 0 & -2 \\ 8 & 5 & 1 & 6 & 0 \end{pmatrix}$$

$$\Pi^{(5)} = \begin{pmatrix} \text{NIL} & 3 & 4 & 5 & 1 \\ 4 & \text{NIL} & 4 & 2 & 1 \\ 4 & 3 & \text{NIL} & 2 & 1 \\ 4 & 3 & 4 & \text{NIL} & 1 \\ 4 & 3 & 4 & 5 & \text{NIL} \end{pmatrix}$$

# Algoritmo de Floyd-Warshall

- Reconstrucción de la ruta
  (a partir de la matriz de predecesores).

  Considere la siguiente matriz de predecesores:

| | | | | |
|---|---|---|---|---|
| **NIL** | **3** | **4** | **5** | **1** |
| **4** | **NIL** | **4** | **2** | **1** |
| **4** | **3** | **NIL** | **2** | **1** |
| **4** | **3** | **4** | **NIL** | **1** |
| **4** | **3** | **4** | **5** | **NIL** |

# Algoritmo de Floyd-Warshall

- Reconstrucción de la ruta
  (a partir de la matriz de predecesores):
  - Reconstruir la ruta del nodo 1 al nodo 2 requiere:

    1. revisar posición [1][2] = 3: significa que en la ruta de 1 a 2, 3 es el último visitado. Ruta: 1 … 3 → 2.

    2. revisar posición [1][3] = 4: en la ruta de 1 a 3, 4 es el último visitado. Ruta: 1 … 4 → 3 → 2.

    3. revisar posición [1][4] = 5: en la ruta de 1 a 4, 5 es el último visitado. Ruta: 1 … 5 → 4 → 3 → 2.

    4. revisar posición [1][5] = 1: 1 es el nodo de inicio. Ruta completa: 1 → 5 → 4 → 3 → 2.

# Algoritmo de Floyd-Warshall

- Applet de demostración:

  www.cs.usfca.edu/~galles/visualization/Floyd.html

# Algoritmo de Floyd-Warshall

- Implementación:

FLOYD-WARSHALL(W) // W: Matriz de adyacencia, wij representa el elemento en la posición I,j de la matriz

```
n = W.rows  // n es la cantidad de nodos del grafo == cantidad de filas de la matriz W
D(0) = W   // D es un arreglo de n matrices de nxn
Π(0) = π(0)ij = NULL if i=j or wij = ∞    // Π es un arreglo de n matrices de nxn
              = i   if i≠j and wij < ∞      // π(0)ij es el valor en la posición i,j de la matriz de
                                            // la posición 0 de Π
for k = 1 to n
   let D(k) = (d(k)ij) be a new nxn matrix
   for i = 1 to n
      for j = 1 to n
         d(k)ij = min(d(k-1)ij, d(k-1)ik + d(k-1)kj)
         if d(k-1)ij ≤ d(k-1)ik + d(k-1)kj
            π(k)ij = π(k-1)ij
         else
            π(k)ij = π(k-1)kj
return D(n)
```

http://goose.ycp.edu/~dbabcock/PastCourses/cs360/lecture/lecture23.html

# Referencias


- Cormen, T., Leiserson, C., Rivest, R., & Stein, C. (2009). Introduction to Algorithms (Third Ed.).
- www.cse.ust.hk/~dekai/271/notes/L07/L07.pdf
- goose.ycp.edu/~dbabcock/PastCourses/cs360/lecture/lecture23.html
- www.cse.ust.hk/faculty/golin/COMP271Sp03/Notes/MyL15.pdf
- www.cs.rit.edu/~zjb/courses/800/lec15-2.pdf
- en.wikipedia.org/wiki/Floyd-Warshall_algorithm