Governo Federal

Ministério da Educação

Universidade Federal do Maranhão

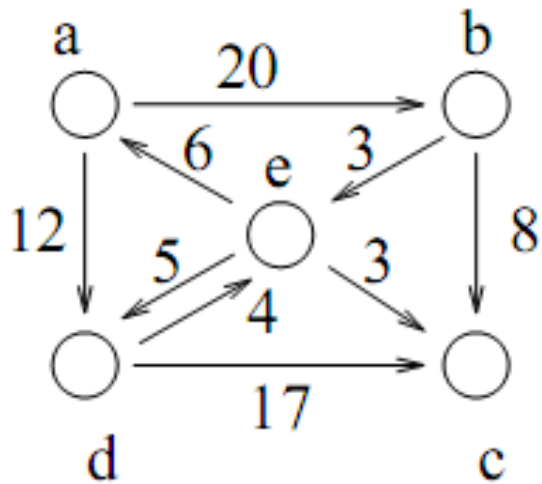
A Universidade que Cresce com Inovação e Inclusão Social

Caminhos mais curtos de múltiplas origens

Estrutura de Dados II

Problema

- **Dado um grafo ponderado G = (V,E) com uma função de peso w:**
  - Determinar o caminho mais curto entre todos os vértices de G
  - Assumindo que não exista ciclo com custo negativo ou zero



Aplicação

- **Exemplo caminho mínimo para todos os pares:**
  - Tabela em atlas e guias rodoviários que dão as distâncias entre várias cidades
  - Determinar o diâmetro de uma rede:
    - O diâmetro dá o tempo de trânsito mais longo possível para uma mensagem na rede.

- O nome deve-se a Robert **Floyd e** Stephen **Warshall**

Soluções

- **Dijkstra**
  - Rodar o algoritmo de dijkstra pelo menos uma vez para cada vértice do grafo
  - Custo: $O(n^3 lgn)$
- **Usando programação dinâmica para naturalmente decompor o problema em soluções menores recursivas**
  - Custo: $O(n^4)$

Floyd-Warshall

- **Complexidade $O(n^3)$**
- **Representação**
  - Programação dinâmica
    - recursão com apoio de uma tabela
  - Assumindo que o grafo é representado por uma matriz de adjacências com a seguinte regra:

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

Floyd-Warshall

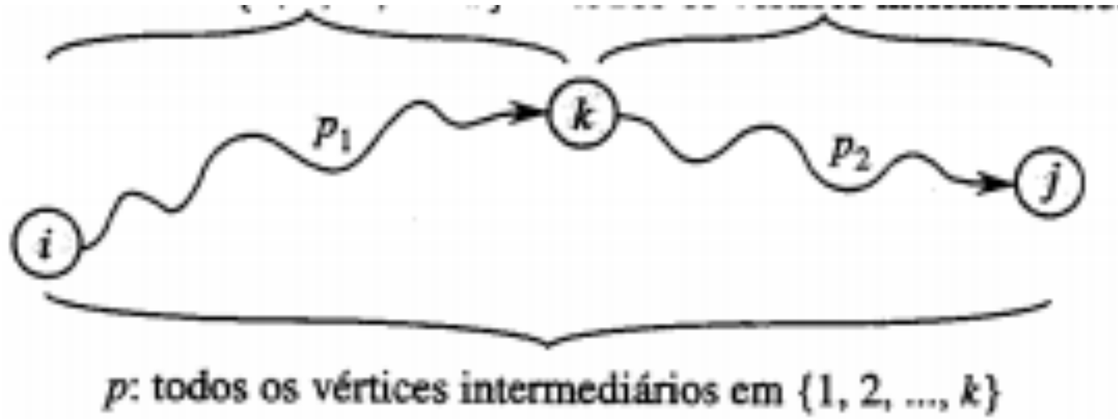
- **Resultado:**
  - Tipicamente a matriz D com as distâncias de custo mínimo
  - Normalmente também se retorna a matriz de antecessores para se construir o caminho mais curto

Decomposição

- **Dado um caminho simples entre dois vértices e também mais curto representado por p={v1,v2,v3,v4 .... vn}:**
  - Propõe que os vértices v2,v3,v4...v(n-1) são vértices intermediários
  - E que, sendo o caminho p o caminho mais curto então a distância em caminho simples entre o vértices intermediários também são caminhos mais curtos

Vértice intermediário

- **Se chamarmos de k um vértice intermediário do caminho p, então:**
  – Podemos desmembrar p em i->k->j, sendo i o ponto de partida, e j o ponto de chegada, k vezes.



Lema

- **Procura-se o caminho mais curto para ligar os vértices i e j:**
  - O caminho da soma por k(P1 + P2)
  - Ou o caminho de i a j sem considerar k como vértice intermediário

Solução Recursiva

- **Seja $d_{ij}(k)$ o peso de um caminho mais curto desde o vértice i até o vértice j para qual todos os vértices intermediários estão em {1,2,3...k}.**
- **Quando k=0, não existe vértices intermediários, logo a seguinte formulação:**

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

Antecessor

- **Construção interativa da matriz predecessora:**
- **Formulação recursiva da matriz pi. Quando k=0 um caminho mais curto de i até j não tem absolutamente nenhum vértice intermediário, logo:**

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

Antecessor

- **Para k > 1**
- **O antecessor pode ser qualquer vértice intermediário**

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

Algoritmo

FLOYD-WARSHALL(W, n)

$D^{(0)} \leftarrow W$

for $k \leftarrow 1$ **to** n

do for $i \leftarrow 1$ **to** n

do for $j \leftarrow 1$ **to** n

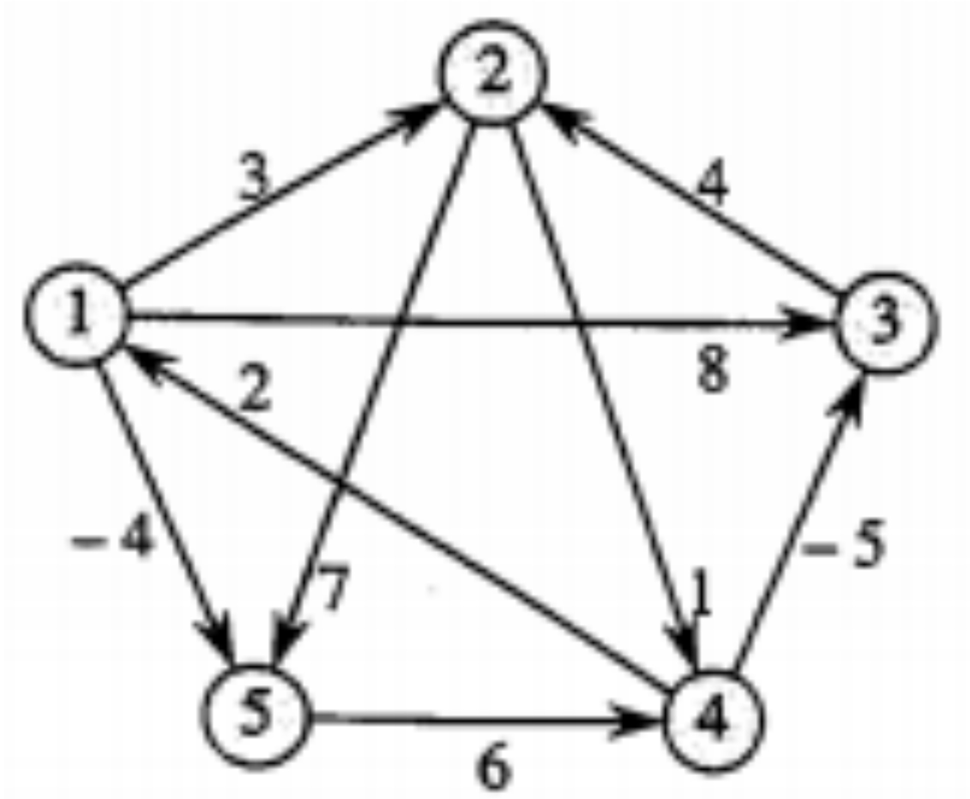
do $d_{ij}^{(k)} \leftarrow \min\left(d_{ij}^{(k-1)}, d_{ik}^{(k-1)} + d_{kj}^{(k-1)}\right)$

return $D^{(n)}$

Algoritmo

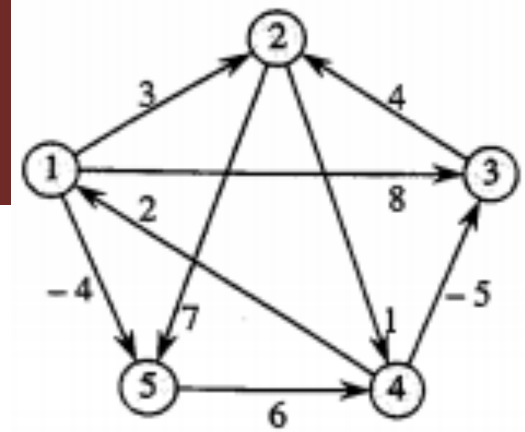
```
let dist be a |V| × |V| array of minimum distances initialized to ∞ (infinity)
for each vertex v
   dist[v][v] ← 0
for each edge (u,v)
   dist[u][v] ← w(u,v)  // the weight of the edge (u,v)
for k from 1 to |V|
   for i from 1 to |V|
      for j from 1 to |V|
         if dist[i][j] > dist[i][k] + dist[k][j]
             dist[i][j] ← dist[i][k] + dist[k][j]
         end if
```

Exemplo: aplicação Floyd e Warshall



Exemplo

K = 1
4 – 1 – 2
4 – 1 – 3
4 – 1 – 5

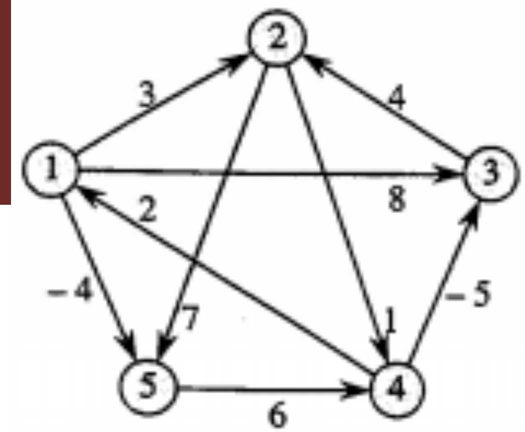


$$D^{(0)} = \begin{pmatrix} 0 & 3 & 8 & \infty & -4 \\ \infty & 0 & \infty & 1 & 7 \\ \infty & 4 & 0 & \infty & \infty \\ 2 & \infty & -5 & 0 & \infty \\ \infty & \infty & \infty & 6 & 0 \end{pmatrix} \quad \Pi^{(0)} = \begin{pmatrix} \text{NIL} & 1 & 1 & \text{NIL} & 1 \\ \text{NIL} & \text{NIL} & \text{NIL}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

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

Exemplo

K = 2
1 - 2 – 5
1 – 2 – 4
3 – 2 – 5
3 – 2 – 4

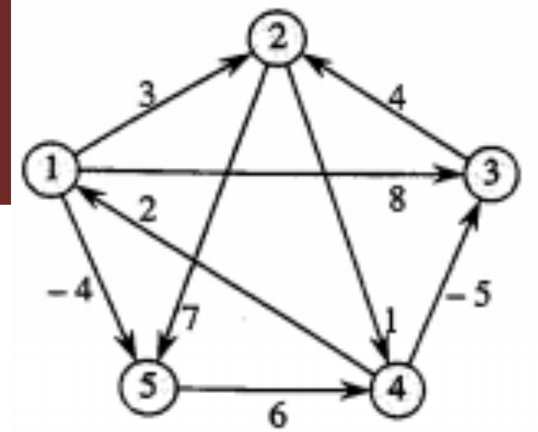


$$D^{(1)} = \begin{pmatrix} 0 & 3 & 8 & \infty & -4 \\ \infty & 0 & \infty & 1 & 7 \\ \infty & 4 & 0 & \infty & \infty \\ 2 & 5 & -5 & 0 & -2 \\ \infty & \infty & \infty & 6 & 0 \end{pmatrix} \quad \Pi^{(1)} = \begin{pmatrix} \mathrm{NIL} & 1 & 1 & \mathrm{NIL} & 1 \\ \mathrm{NIL} & \mathrm{NIL} & \mathrm{NIL}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

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

Exemplo

K = 3
4 – 3 – 2
1 – 3 – 2

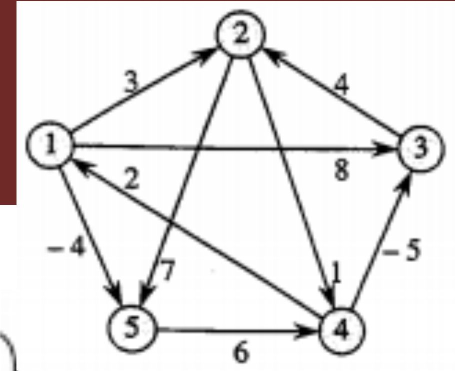


$$D^{(2)} = \begin{pmatrix} 0 & 3 & 8 & 4 & -4 \\ \infty & 0 & \infty & 1 & 7 \\ \infty & 4 & 0 & 5 & 11 \\ 2 & 5 & -5 & 0 & -2 \\ \infty & \infty & \infty & 6 & 0 \end{pmatrix} \quad \Pi^{(2)} = \begin{pmatrix} \text{NIL} & 1 & 1 & 2 & 1 \\ \text{NIL} & \text{NIL} & \text{NIL} & 2 & 2 \\ \text{NIL}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

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

Exemplo

K = 4
2 – 4 – 3
2 – 4 – 1
5 – 4 – 2
5 – 4 – 3



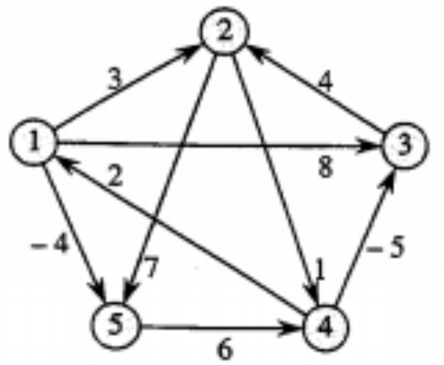
$$D^{(3)} = \begin{pmatrix} 0 & 3 & 8 & 4 & -4 \\ \infty & 0 & \infty & 1 & 7 \\ \infty & 4 & 0 & 5 & 11 \\ 2 & -1 & -5 & 0 & -2 \\ \infty & \infty & \infty & 6 & 0 \end{pmatrix} \quad \Pi^{(3)} = \begin{pmatrix} \text{NIL} & 1 & 1 & 2 & 1 \\ \text{NIL} & \text{NIL} & \text{NIL} & 2 & 2 \\ \text{NIL}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

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

Para K = 4, temos D(u,v,x) = D(1,3,4)
D(1,3) > D(1,4) + D(4,3)
OBS: analisar a matriz de precedencia (PI)
D(1,4) = D(1,2) + D(2,4)
Então. verificamos se:
D(1,3) > D(1,2) + D(2,4) + D(4,3)

Exemplo

K = 5



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

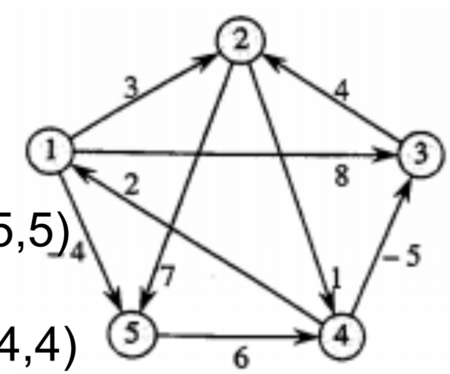
$$D^{(5)} = \begin{pmatrix} 0 & \boxed{1} & \boxed{-3} & \boxed{2} & -4 \\ 3 & 0 & -4 & 1 & -1 \\ 7 & 4 & 0 & 5 & 3 \\ 2 & -1 & -5 & 0 & -2 \\ 8 & 5 & 1 & 6 & 0 \end{pmatrix} \quad \Pi^{(5)} = \begin{pmatrix} \text{NIL} & \boxed{5} & \boxed{5} & \boxed{5} & 1 \\ 4 & \text{NIL} & 4 & 2 & 4 \\ 4 & 3 &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

Algoritmo para obter o caminho

```
Path (i, j)
  if (pred[u,v] == NULL) {
      print(u,v);
  } else {
      Path(i, pred[i,j]);
      Path(pred[i,j], j);
  }
```

Algoritmo para obter o caminho

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



Menor caminho entre 5 e 2 ?

5..2 Path(5,2). Pred[5,2] = 4

5..4..2 Path(5,4). Pred[5,4] = 5. Pred [5,5] = NULL print(5,5)

Path(4,2). Pred[4,2] = 3.

5.4.3.2 Path(4,3). Pred[4,3] = 4. Pred[4,4]. = NULL. Print(4,4)

5.4.3.2 Path(3,2). Pred[3,2] = 3. Pred[3,3] =NULL. Print (3,3)

Caminho: 5 – 4 – 3 - 2

Fecho Transitivo de um grafo Orientado

- **Para um grafo G = (V,E), com o conjunto de vértices V = {1,2,3,4...n}**
  - Deseja-se descobrir se existe um caminho em G, de i até j, para todos os pares de vértices i,j de G
  - O **fecho transitivo** de um vértice *v* é o conjunto de todos os vértices que podem ser atingidos por algum caminho iniciando em *v*.
- **O fecho transitivo de G é definido como o grafo G*=(V,E*) onde:**
  - E* {(i,j) : existe um caminho do vértice i até o vértice j em G}

Primeira Solução

- **Atribuir o peso 1 em todas as arestas do grafo, e rodar o floyd-warshall.**
- **Se o caminho de i até j existir, obtêm dij<n senão dij = infinito**
  - sendo n o número de vértices
- **Outra maneira**
  - Substituir as operações aritméticas min e + por OU (lógico) e E (lógico)
    - Pode reduzir tempo e espaço

Definição Recursiva

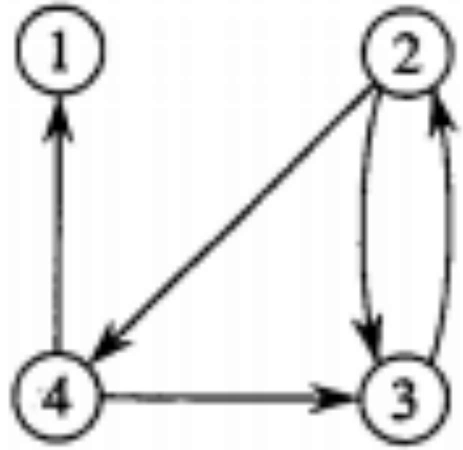
$$t_{ij}^{(0)} = \begin{cases} 0 & \text{if } i \neq j \text{ and } (i, j) \notin E , \\ 1 & \text{if } i = j \text{ or } (i, j) \in E . \end{cases}$$

$$t_{ij}^{(k)} = t_{ij}^{(k-1)} \vee \left(t_{ik}^{(k-1)} \wedge t_{kj}^{(k-1)} \right).$$

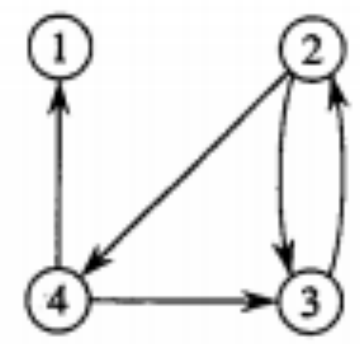
Algoritmo

$$
\begin{array}{l}
\text{TRANSITIVE-CLOSURE}(E, n) \\
\textbf{for } i \leftarrow 1 \textbf{ to } n \\
\quad \textbf{do for } j \leftarrow 1 \textbf{ to } n \\
\quad\quad \textbf{do if } i = j \text{ or } (i, j) \in E[G] \\
\quad\quad\quad \textbf{then } t_{ij}^{(0)} \leftarrow 1 \\
\quad\quad\quad \textbf{else } t_{ij}^{(0)} \leftarrow 0 \\
\textbf{for } k \leftarrow 1 \textbf{ to } n \\
\quad \textbf{do for } i \leftarrow 1 \textbf{ to } n \\
\quad\quad \textbf{do for } j \leftarrow 1 \textbf{ to } n \\
\quad\quad\quad \textbf{do } t_{ij}^{(k)} \leftarrow t_{ij}^{(k-1)} \vee \left(t_{ik}^{(k-1)} \wedge t_{kj}^{(k-1)}\right) \\
\textbf{return } T^{(n)}
\end{array}
$$

Exemplo



Exemplo



$$T^{(0)} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 \\ 0 & 1 & 1 & 0 \\ 1 & 0 & 1 & 1 \end{pmatrix} \quad T^{(1)} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 \\ 0 & 1 & 1 & 0 \\ 1 & 0 & 1 & 1 \end{pmatrix} \quad T^{(2)} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 \\ 0 & 1 & 1 & 1 \\ 1 & 0 & 1 & 1 \end{pmatrix}$$

$$T^{(3)} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 \\ 0 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \end{pmatrix} \quad T^{(4)} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \end{pmatrix}$$

Exercício

- **Utilize o algoritmo de Floyd Warshall para encontrar a distâncias entre todos os vértices do grafo abaixo:**