Algoritmo de Floyd-Warshall

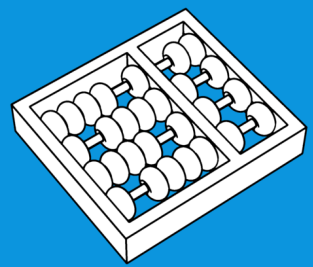
MC558 - Projeto e Análise de Algoritmos II

09/24

14

Santiago Valdés Ravelo
https://ic.unicamp.br/~santiago/
ravelo@unicamp.br

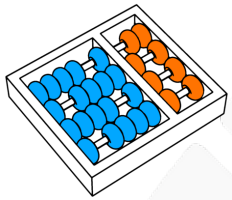
UNICAMP



“O caminho mais curto nem sempre é uma linha reta.”

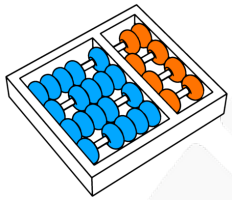
Anônimo

Caminhos mínimos entre todos os pares de vértices



Novo problema

Dado grafo ponderado (G, w) sem ciclos negativos, queremos encontrar um caminho mínimo de **u** a **v** para **TODO** par de vértices **u**, **v**.



Algoritmos para grafos esparsos

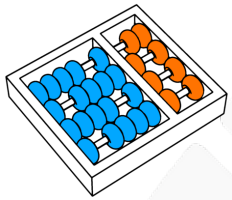
Podemos executar $|V|$ vezes um algoritmo de Caminhos Mínimos com Mesma Origem:

- Se (G, w) não possui arestas negativas, usamos Dijkstra:

Tipo de fila	Uma vez	$\|V\|$ vezes
Heap	$O(E \log V)$	$O(VE \log V)$
Fibonacci	$O(V \log V + E)$	$O(V^2 \log V + VE)$

- Se (G, w) possui arestas negativas, usamos Bellman-Ford:

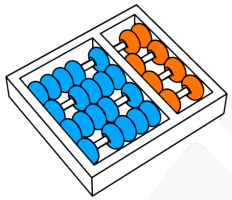
Uma vez	$\|V\|$ vezes
$O(VE)$	$O(V^2 E)$



O algoritmo de Floyd-Warshall

Floyd-Warshall é um algoritmo que resolve o problema diretamente e é melhor se G for **DENSO**.

- Um grafo é denso se $|E| = \omega(V^2)$.
- É basado em programação dinâmica.
- Resolve o problema em tempo $O(V^3)$.
- Supomos que G é completo:
  $\Rightarrow$ Se (i, j) não é aresta, definimos $w(i, j) = \infty$



Subproblema

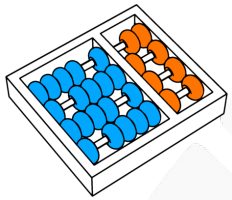
Para simplificar a discussão, suponha que $\mathbf{V = \{1, 2, \ldots, n\}}$.

Considere um caminho $P = (\mathbf{v_1, v_2, \ldots, v_{l-1}, v_l})$:

- Os **VÉRTICES INTERNOS** de P são $\{\mathbf{v_2, \ldots, v_{l-1}}\}$.
- P é chamado k-**INTERNO** se $\{\mathbf{v_2, \ldots, v_{l-1}}\} \subseteq \{\mathbf{1, \ldots, k}\}$.

Problema (Subproblema ótimo)

Sejam $\mathbf{i}$ *e* $\mathbf{j}$ *vértices de* G *e* k *um inteiro com* $k \geq 0$. *Dentre todos os caminhos* k*-internos de* $\mathbf{i}$ *até* $\mathbf{j}$, *encontre algum que tenha custo mínimo.*



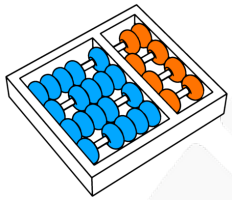
Estrutura de um caminho mínimo

O algoritmo de Floyd-Warshall explora a relação entre:

- Um caminho k-interno P de **i** até **j** que tem custo mínimo
- e outros caminhos $(k-1)$-internos.

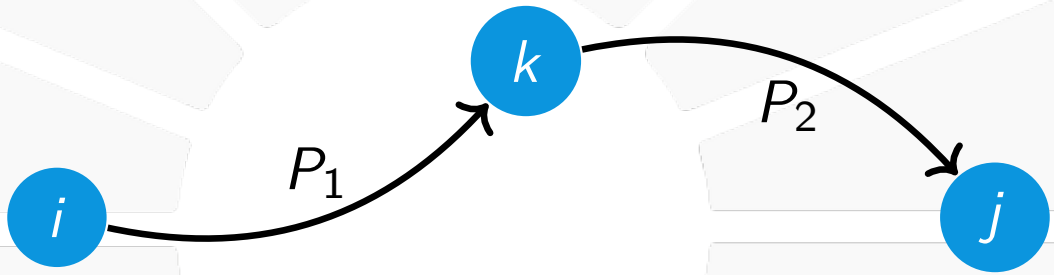
Caso 1: Se **k** não é um vértice interno de P:

- Todos os vértices internos de P estão em $\{\mathbf{1}, \ldots, \mathbf{k-1}\}$.
- Então, P é um caminho $(k-1)$-interno de custo mínimo.

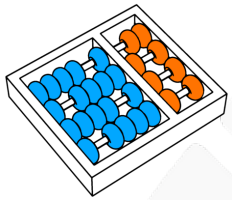


Estrutura de um caminho mínimo

Caso 2: Se **k** é um vértice interno de P, então P pode ser dividido em dois caminhos P_1 (com início em **i** e fim em **k**) e P_2 (com início em **k** e fim em **j**).



- P_1 é um caminho mínimo de **i** a **k** com vértices internos em $\{\mathbf{1}, \ldots, \mathbf{k-1}\}$.
- P_2 é um caminho mínimo de **k** a **j** com vértices internos em $\{\mathbf{1}, \ldots, \mathbf{k-1}\}$.



Recorrência para caminhos mínimos

Seja $d_{ij}^{(k)}$ o peso de um caminho k-interno mínimo de **i** a **j**.

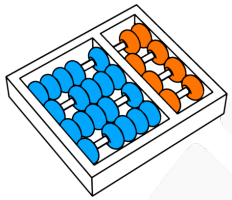
- Se $k = 0$ então $d_{ij}^{(0)} = w(i,j)$.
- Senão, caímos em algum dos dois casos anteriores.

Obtemos a seguinte recorrência:

$$d_{ij}^{(k)} = \begin{cases} w(i,j) & \text{se } k = 0, \\ \min(d_{ij}^{(k-1)}, d_{ik}^{(k-1)} + d_{kj}^{(k-1)}) & \text{se } k \geq 1. \end{cases}$$

Note que, $d_{ij}^{(n)} = \text{dist}(i,j)$. Por quê?

- Calculamos as matrizes $D^{(k)} = (d_{ij}^{(k)})$ para $k = 1, 2 \ldots, n$.
- A resposta do problema é $D^{(n)}$.



Algoritmo de Floyd-Warshall

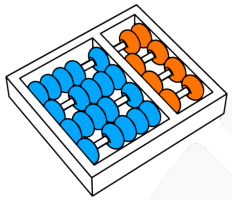
Problema

Entrada: *Matriz* $W = (w(i,j))$ *com dimensões* $n \times n$.
Saída: *Matriz* $D^{(n)}$.

Algoritmo: FLOYD-WARSHALL(W, n)

```
1 D^(0) ← W
2 para k ← 1 até n
3     para i ← 1 até n
4         para j ← 1 até n
5             d_ij^(k) ← min(d_ij^(k−1), d_ik^(k−1) + d_kj^(k−1))
6 devolva D^(n)
```

Complexidade: $O(V^3)$.



Como encontrar os caminhos?

Devolvemos matriz de predecessores $\Pi = (\pi_{ij})$

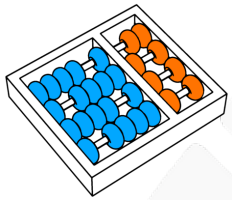
(a) $\pi_{ij} = \text{NIL}$ se **i** $=$ **j** ou se não existe caminho de **i** a **j**,

(b) π_{ij} é o **PREDECESSOR** de **j** em algum caminho mínimo a partir de **i**, caso contrário.

- Calculamos do mesmo modo que $D^{(k)}$.
- Obtemos uma sequência de matrizes $\Pi^{(0)}, \Pi^{(1)}, \ldots, \Pi^{(n)}$

Quando $k = 0$:

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



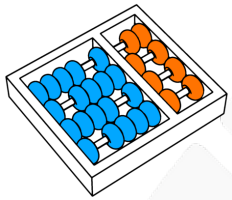
Como encontrar os caminhos?

Se $k \geq 1$:

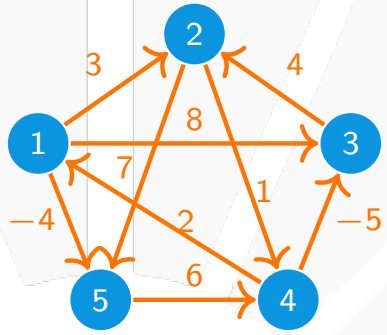
- Considere um caminho k-interno mínimo P de **i** a **j**.
- Obtemos o **PREDECESSOR DE j**:
  - Se k não aparece em P, então usamos o predecessor de um caminho $(k-1)$-interno de **i** a **j**.
  - Se k aparece em P, então usamos o predecessor de um caminho $(k-1)$-interno de **k** a **j**.

Formalmente,

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

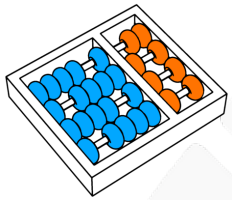


Exemplo

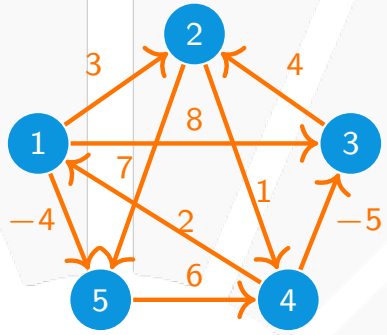


$$D^{(0)} = \begin{pmatrix} 0 & 3 & 8 & \infty & -4 \\ \infty & 0 & \infty & 1 & 7 \\ \infty & 4 & 0 & \infty & \infty \\ 2 & \infty & -5 & 0 & \infty \\ \infty & \infty & \infty & 6 & 0 \end{pmatrix} \qquad \Pi^{(0)} = \begin{pmatrix} N & 1 & 1 & N & 1 \\ N & N & N & 2 & 2 \\ N & 3 & N & N & N \\ 4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

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

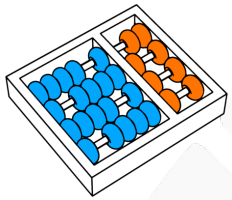


Exemplo

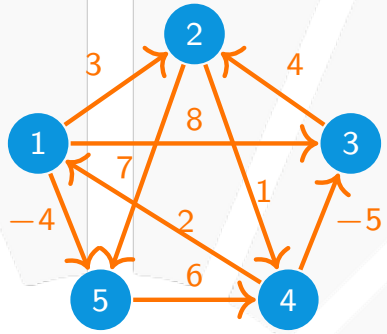


$$D^{(1)} = \begin{pmatrix} 0 & 3 & 8 & \infty & -4 \\ \infty & 0 & \infty & 1 & 7 \\ \infty & 4 & 0 & \infty & \infty \\ 2 & 5 & -5 & 0 & -2 \\ \infty & \infty & \infty & 6 & 0 \end{pmatrix} \qquad \Pi^{(1)} = \begin{pmatrix} N & 1 & 1 & N & 1 \\ N & N & N & 2 & 2 \\ N & 3 & N & N & N \\ 4 & 1 & 4 &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

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

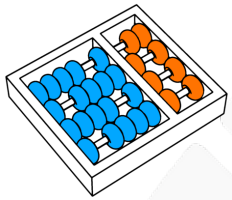


Exemplo

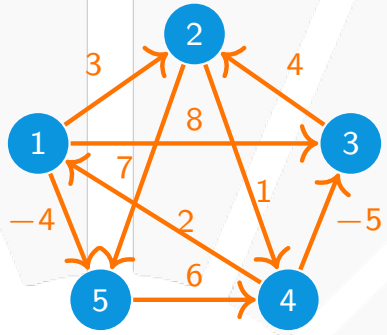


$$D^{(2)} = \begin{pmatrix} 0 & 3 & 8 & 4 & -4 \\ \infty & 0 & \infty & 1 & 7 \\ \infty & 4 & 0 & 5 & 11 \\ 2 & 5 & -5 & 0 & -2 \\ \infty & \infty & \infty & 6 & 0 \end{pmatrix} \qquad \Pi^{(2)} = \begin{pmatrix} N & 1 & 1 & 4 & 1 \\ N & N & N & 2 & 2 \\ N & 3 & N & 2 & 2 \\ 4 & 1 & 4 & N & 1 \\ N &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

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

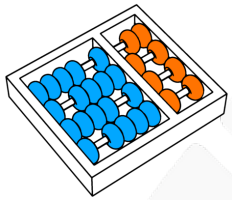


Exemplo

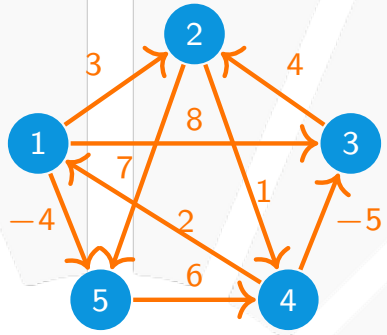


$$D^{(3)} = \begin{pmatrix} 0 & 3 & 8 & 4 & -4 \\ \infty & 0 & \infty & 1 & 7 \\ \infty & 4 & 0 & 5 & 11 \\ 2 & -1 & -5 & 0 & -2 \\ \infty & \infty & \infty & 6 & 0 \end{pmatrix} \qquad \Pi^{(3)} = \begin{pmatrix} N & 1 & 1 & 2 & 1 \\ N & N & N & 2 & 2 \\ N & 3 & N & 2 & 2 \\ 4 & 3 & 4 & N & 1 \\ N &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

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



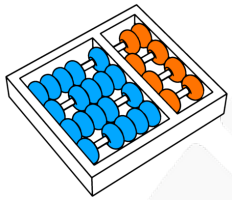
Exemplo



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

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

Fecho transitivo de grafos direcionados

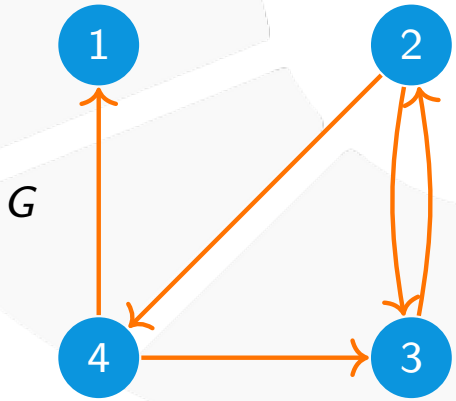


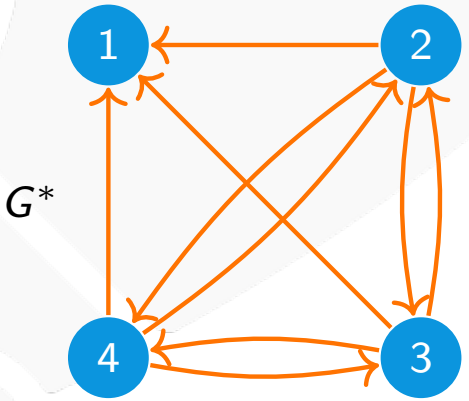
Fecho transitivo de grafos direcionados

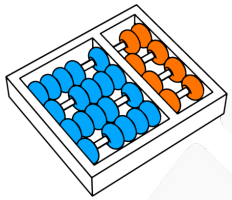
Seja $G = (\mathbf{V}, \mathbf{E})$ um grafo direcionado com $\mathbf{V} = \{\mathbf{1}, \mathbf{2}, \ldots, \mathbf{n}\}$.

O **FECHO TRANSITIVO** de $G = (\mathbf{V}, \mathbf{E})$ é o grafo $G^* = (\mathbf{V}, \mathbf{E}^*)$ onde:

$$\mathbf{E}^* = \{(\mathbf{i}, \mathbf{j}) : \text{ existe um caminho de } i \text{ a } j \text{ em } G\}.$$







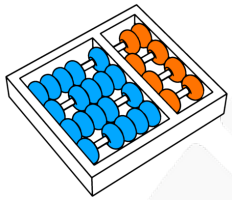
Um detalhe de implementação

Determinando o fecho transitivo de $G = (\mathbf{V}, \mathbf{E})$:

1. Atribuímos peso 1 a cada aresta.
2. Executamos FLOYD-WARSHALL em tempo $\Theta(V^3)$.
3. Existe um caminho de **i** a **j** se e somente se $d_{ij} < |V|$.

Na prática:

- Executamos FLOYD-WARSHALL substituindo:
  - min por $\vee$ (OU lógico).
  - $+$ por $\wedge$ (E lógico).
- Tem a mesma complexidade assintótica.
- Economiza tempo e espaço.



Fecho transitivo de grafos direcionados

Defina $t_{i,j}^{(k)}$ o valor booleano:

- TRUE se existe caminho k-interno de **i** a **j**.
- FALSE se **NÃO** existe caminho k-interno de **i** a **j**.

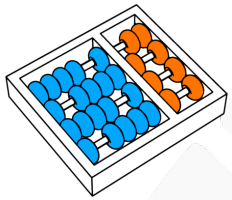
Para $k = 0$

$$t_{ij}^{(0)} = \begin{cases} 0 & \text{se } i \neq j \text{ e } (i,j) \notin E, \\ 1 & \text{se } i = j \text{ ou } (i,j) \in E, \end{cases}$$

e para $k \geq 1$,

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

Como em FLOYD-WARSHALL, calculamos matrizes $T^{(k)} = (t_{ij}^{(k)})$.



Algoritmo de Transitive-Closure

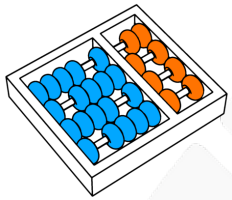
Problema

Entrada: *Matriz de adjacência A de G.*
Saída: *Matriz de adjacência $T^{(n)}$ de G^*.*

Algoritmo: TRANSITIVE-CLOSURE(A, n)

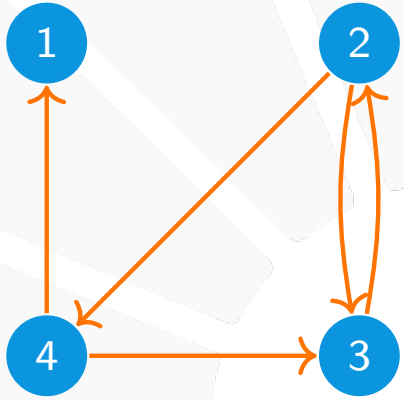
```
1 T^(0) ← A + I_n
2 para k ← 1 até n
3     para i ← 1 até n
4         para j ← 1 até n
5             t_ij^(k) ← t_ij^(k−1) ∨ (t_ik^(k−1) + t_kj^(k−1))
6 devolva T^(n)
```

Complexidade: $O(V^3)$.

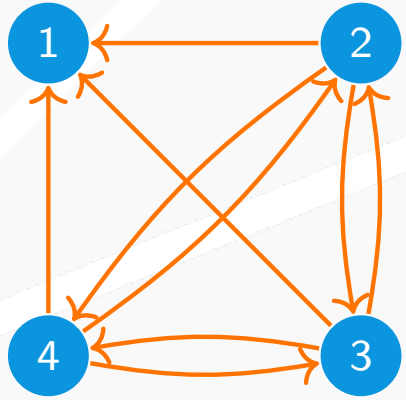


Exemplo

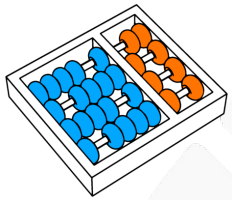
G



G^*

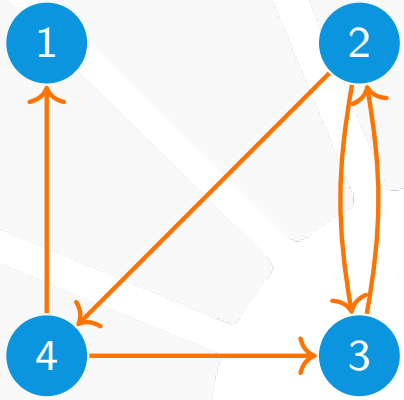


$$T^{(0)} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 \\ 0 & 1 & 1 & 0 \\ 1 & 0 & 1 & 1 \end{pmatrix} \qquad T^{(1)} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 \\ 0 & 1 & 1 & 0 \\ 1 & 0 & 1 & 1 \end{pmatrix}$$

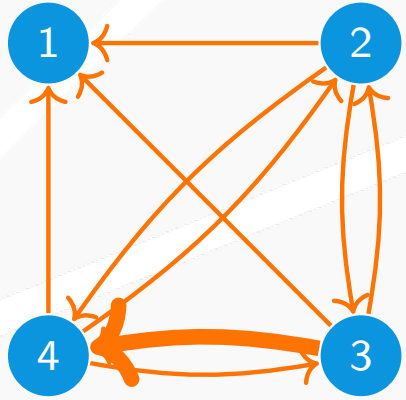


Exemplo

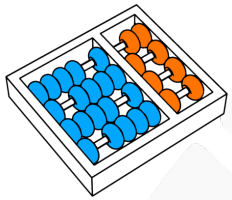
G



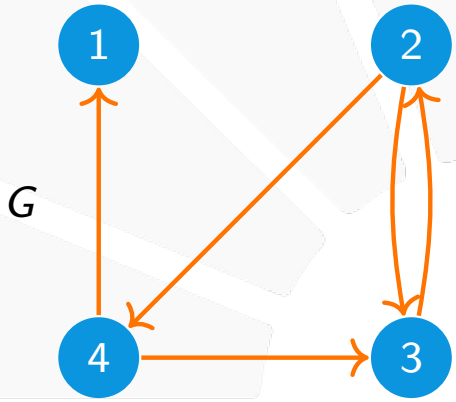
G^*

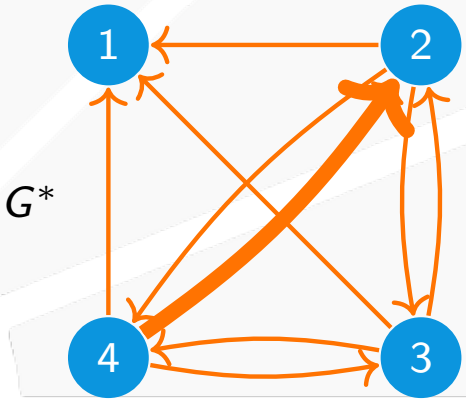


$$T^{(1)} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 \\ 0 & 1 & 1 & 0 \\ 1 & 0 & 1 & 1 \end{pmatrix} \qquad T^{(2)} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 \\ 0 & 1 & 1 & \mathbf{1} \\ 1 & 0 & 1 & 1 \end{pmatrix}$$

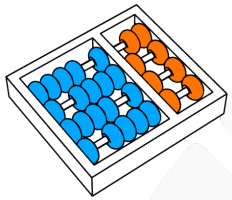


Exemplo



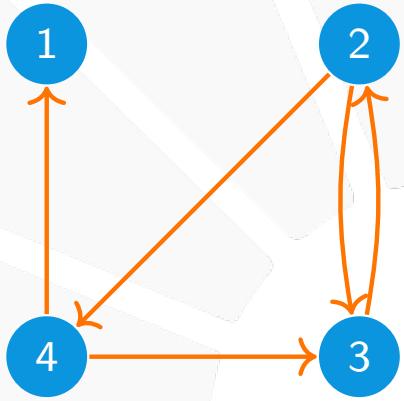


$$T^{(2)} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 \\ 0 & 1 & 1 & 1 \\ 1 & 0 & 1 & 1 \end{pmatrix} \qquad T^{(3)} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 \\ 0 & 1 & 1 & 1 \\ 1 & \mathbf{1} & 1 & 1 \end{pmatrix}$$

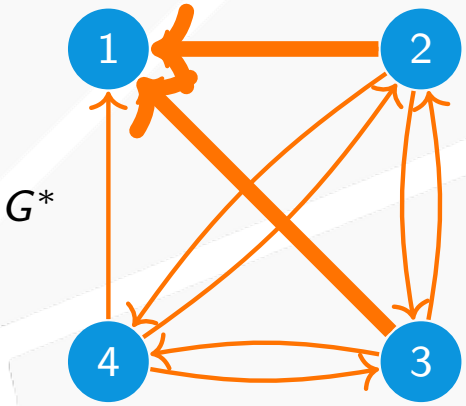


Exemplo

G



G^*



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

Algoritmo de Floyd-Warshall

MC558 - Projeto e Análise de Algoritmos II

09/24

14

Santiago Valdés Ravelo
https://ic.unicamp.br/~santiago/
ravelo@unicamp.br

UNICAMP