Problemas de caminhos em grafos e Programação Dinâmica

Claudia M. Justel

Instituto Militar de Engenharia, Rio de Janeiro, Brasil

Brazilian ICPC Summer School 2023 - Brazilian Final class

AGENDA

03/02/2023 - AULA 1: Problemas de caminhos em grafos

04/02/2023 - AULA 2: Programação Dinâmica

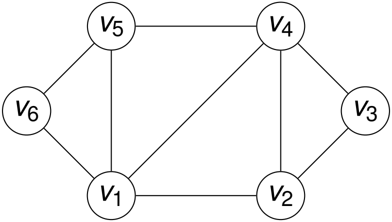
AULA 1 - Objetivo

Mostrar a construção dos caminhos de diferentes formas:

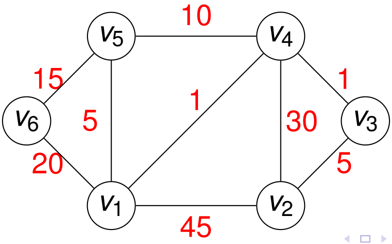
- percursos em grafos, BFS
- Dijkstra
- Floyd-Warshall

AULA 1 - Preliminares

$G = (V, E)$ $|V| = n$, $|E| = m$ grafo não direcionado

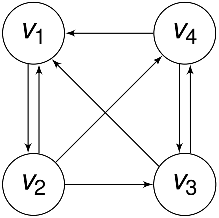


$G = (V, E)$ $|E| = n$, $|E| = m$, $w : E \to \mathbb{R}$ grafo não direcionado ponderado.

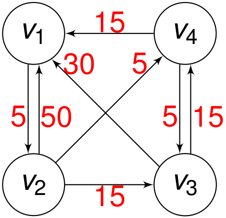


AULA 1 - Preliminares

$G = (V, E)$ $|V| = n$, $|E| = m$ grafo direcionado



$G = (V, E)$ $|V| = n$, $|E| = m$, $w : E \to \mathbb{R}$ grafo direcionado ponderado



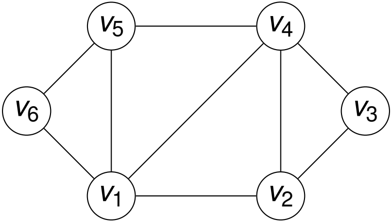
AULA 1 - Preliminares

aramazenamento de um grafo na memória do computador:

- matriz de adjacência,
- listas de adjacências,

$$A(G) = \begin{bmatrix} 0 & 1 & 0 & 1 & 1 & 1 \\ 1 & 0 & 1 & 1 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 & 0 \\ 1 & 1 & 1 & 0 & 1 & 0 \\ 1 & 0 & 0 & 1 & 0 & 1 \\ 1 & 0 & 0 & 0 & 1 & 0 \end{bmatrix}$$

$$\begin{array}{lllllll} v_1 & \rightarrow & \boxed{v_2} & \boxed{v_4} & \boxed{v_5} & \boxed{v_6} \\ v_2 & \rightarrow & \boxed{v_4} & \boxed{v_1} & \boxed{v_3} & \\ v_3 & \rightarrow & \boxed{v_2} & \boxed{v_4} & & \\ v_4 & \rightarrow & \boxed{v_3} & \boxed{v_5} & \boxed{v_1} & \boxed{v_2} \\ v_5 & \rightarrow & \boxed{v_1} & \boxed{v_4} & \boxed{v_6} & \\ v_6 & \rightarrow & \boxed{v_1} & \boxed{v_5} & & \end{array}$$



AULA 1 - Preliminares

aramazenamento de um grafo na memória do computador:

- matriz de adjacência,
- listas de adjacências,

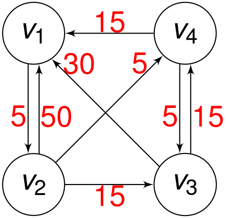
$$L = \begin{bmatrix} 0 & 5 & \infty & \infty \\ 50 & 0 & 15 & 5 \\ 30 & \infty & 0 & 15 \\ 15 & \infty & 5 & 0 \end{bmatrix}$$

$v_1 \rightarrow \boxed{v_2, 5}$

$v_2 \rightarrow \boxed{v_1, 50} \quad \boxed{v_3, 15} \quad \boxed{v_4, 5}$

$v_3 \rightarrow \boxed{v_1, 30} \quad \boxed{v_4, 15}$

$v_4 \rightarrow \boxed{v_1, 15} \quad \boxed{v_3, 5}$



AULA 1 - Preliminares

passeio em $G = (V, E)$: sequência $w_1, \ldots, w_k$ de vértices tais que $(w_i, w_{i+1}) \in E$ para $1 \leq i \leq k-1$, $k > 1$.

caminho em um grafo $\rightarrow$ comprimento do caminho é $k-1$.

distância entre dois vértices $\rightarrow$ comprimento do menor caminho entre eles.

grafo não ponderado $\rightarrow$ comprimento do caminho com menor número de arestas.

grafo ponderado $\rightarrow$ comprimento caminho com menor valor da soma dos pesos ou custos das arestas que o compõem.

outros conceitos: ciclo, grafo conexo, árvore, árvore geradora.

AULA 1 - Percursos em Grafos

Na busca ou percurso geral as escolhas de vértice e arestas são arbitrárias.

procedimento Busca-Geral(G)
escoher vertice inicial e marca-lo
enquanto existir vertice v marcado e incidente a aresta (v, w) nao visitada **faça**
    escolher o vertice v e visitar a aresta (v, w)
    se w nao marcado **então** marcar w

AULA 1 - Percursos em Grafos

A busca em profundidade DFS (Depth First Search) e a busca em largura BFS (Breadth First Search) são dois tipos especiais de busca, onde a escolha do vértice marcado obedece a algum critério especial.

DFS (regra 1): na busca-geral, o critério de escolha do vértice marcado v é o seguinte: ”dentre os vértices marcados e incidentes a uma aresta ainda não visitada, escolher aquele **mais** recentemente alcançado pela busca ” (alcançado=marcado).

BFS (regra 2): na busca-geral, o critério de escolha do vértice marcado v é o seguinte: ”dentre os vértices marcados e incidentes a alguma aresta ainda não visitada, escolher aquele **menos** recentemente alcançado pela busca”.

AULA 1 - Percursos em Grafos - DFS

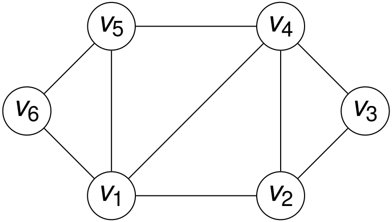
A escolha do vértice marcado torna-se única e sem ambigüidade pela regra 1 descrita para a DFS.
Seja $G = (V, E)$ um grafo conexo.

procedimento DFS(G, s)
procedimeto $P(v)$
    marcar v
    inserir v na pilha Q
    para $w \in Adj(v)$ *faça*
        se w nao marcado *então* visitar (v, w) (I)
            $P(w)$
        senão se $w \in Q$ e v, w sao nao consecutivos em Q *então* visitar (v, w) (II)
    remover v de Q
desmarcar todos os vertices e arestas
definir a pilha vazia Q
$P(s)$

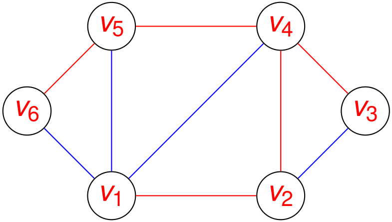
A complexidade do pior caso do procedimento DFS é $O(n + m)$.

AULA 1 - Percursos em Grafos -DFS

$G = (V, E)$ grafo conexo, $|V| = 6$, $|E| = 9$.



$s = v_1$



AULA 1 - Percursos em Grafos -DFS

No algoritmo **DFS** são arbitrárias as escolhas da raiz da busca (s) bem como da aresta (v, w) a ser visitada a partir do vértice marcado v.

O algoritmo **DFS** divide o conjunto de arestas do grafo E em dois conjuntos disjuntos:

- arestas visitadas em (I) denominadas **arestas de árvore** (vermelho)
- arestas visitadas em (II), chamadas de **arestas de retorno ou fronde** (azul).

AULA 1 - Percursos em Grafos - BFS

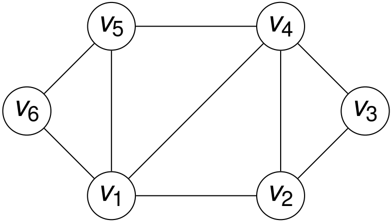
A escolha do vértice marcado torna-se única e sem ambigüidade pela regra 2 descrita para a BFS.
Seja $G = (V, E)$ um grafo conexo.

procedimento BFS(G, s)
desmarcar todos os vertices e arestas
definir a fila vazia Q
marcar s
inserir s em Q
enquanto $Q \neq \emptyset$ *faça*
    seja v o primeiro elemento em Q
    para $w \in Adj(v)$ *faça*
        se w nao marcado *então*
            visitar (v, w) (I)
            marcar w
            inserir w em Q
        senão se $w \in Q$ *então* visitar (v, w) (II)
    remover v de Q

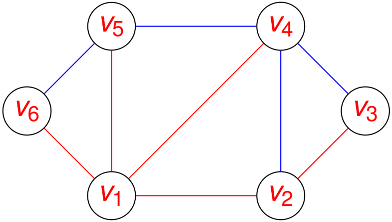
A complexidade do pior caso do procedimento BFS é $O(n + m)$.

AULA 1 - Percursos em Grafos - BFS

$G = (V, E)$ grafo conexo, $|V| = 6$, $|E| = 9$.



$s = v_1$



AULA 1 - Percursos em Grafos - BFS

Seja E_T o conjunto das arestas visitadas pelo procedimento BFS em (I).

TEOREMA: O grafo $T = (V, E_T)$ é uma árvore geradora do grafo conexo $G = (V, E)$.

Notaremos **árvore de largura** à árvore $T = (V, E_T)$ obtida a partir de uma BFS num grafo conexo G a partir do vértice s.

Define-se para cada $v \in V$ **nível do vértice** v na árvore de largura T ($nivel(v)$).

A árvore de largura $T = (V, E_T)$ foi construída a partir do vértice s, e portanto a raiz de T é s.

Para cada vértice $v \in V$, $nivel(v)$ determina o comprimento do caminho desde raiz até v ($nivel(s) = 0$).

AULA 1 - Percursos em Grafos - BFS

OBSERVAÇÕES:

BFS pode ser usado para achar caminhos mínimos em grafo ponderado com todos os pesos iguais.

AULA 1 - Caminhos mínimos de fonte simples - Dijkstra

$G = (V, E)$ grafo (ou digrafo), $|V| = n$, $|E| = m$ e $\forall e \in E$ $w(e) \geq 0$.

Problema de caminhos mínimos de fonte simples: consiste em achar o comprimento do menor caminho (ou caminho mais curto) desde um vértice especial ou **fonte**, s, até qualquer outro vértice do grafo.

O algoritmo guloso que resolve o problema é conhecido na literatura como algoritmo de DIJKSTRA.

AULA 1 - Caminhos mínimos de fonte simples - Dijkstra

$G = (V, E), \forall e \in E \; w(e) \geq 0, \; V = S \cup C$.

Inicialmente $S = \{s\}$ (s é a fonte).

No fin da execução do algoritmo, $S = V$.

A cada iteração:
S contém os vértices já escolhidos pelo algoritmo
C contém os vértices de $V - S$
escolhe-se o vértice no conjunto C cuja distância à s é mínima,
remove-se o mesmo de C e acrescenta-se a S,
atualizam-se os caminhos desde s até os vértices em C.

AULA 1 - Caminhos mínimos de fonte simples - Dijkstra

Caminho Especial desde a fonte até um outro vértice: caminho tal que todos os vértices intermediarios do mesmo pertencem ao conjunto S.

A cada iteração do algoritmo, um vetor D contém os comprimentos do menor caminho especial para cada vértice do grafo.

Pode-se provar (ver teorema a seguir) que a cada iteração, quando é acrescentado o vértice v no conjunto S, o menor caminho especial da fonte até v é também o menor de todos os caminhos da fonte até v.

AULA 1 - Caminhos mínimos de fonte simples - Dijkstra

Supor que os vértices do grafo G estão numerados de 1 até n, ou seja $V = \{1, 2, \ldots, n\}$. Supor que o vértice $s = 1$ é a fonte.

Seja L a matriz $n \times n$:
$L[i, j] = 0$, se $i = j$;
$L[i, j] = w(i, j)$, se $i \neq j$ e existe aresta (i, j) em G;
$L[i, j] = \infty$, se $i \neq j$ e não existe aresta (i, j) em G.

AULA 1 - Caminhos mínimos de fonte simples - Dijkstra

Seja D um vetor de dimensão n onde serão armazenados os comprimentos dos caminhos mínimos achados pelo algoritmo.

procedimento DIJKSTRA(L, D)
$C = \{2, 3, \ldots, n\}$ % $S = V - C$
para $i = 1, \ldots, n$ *faça* $D[i] = L[1, i]$
repetir $n - 2$ *vezes*
  escolher $v \in C$ que minimize $D[v]$
  $C = C - \{v\}$ % $S = V - C$
  para $u \in C$ *faça*
    se $D[u] > D[v] + L[v, u]$ *então*
      $D[u] = D[v] + L[v, u]$
retornar D

AULA 1 - Caminhos mínimos de fonte simples - Dijkstra

Para obter a seqüência de vértices que compõem o caminho mínimo desde 1 até i, utilizar um vetor P de dimensão n, $P[i] = j$ se vértice j precede ao vértice i no caminho mínimo desde 1 até i.

procedimento DIJKSTRA1(L, D, P)
$C = \{2, 3, \ldots, n\}$ % $S = V - C$
para $i = 1, \ldots, n$ *faça* $D[i] = L[1, i]$
    se $L[1, i] < \infty$ *então* $P[i] = 1$
repetir $n - 2$ *vezes*
    escolher $v \in C$ que minimize $D[v]$
    $C = C - \{v\}$ % $S = V - C$
    para $u \in C$ *faça*
        se $D[u] > D[v] + L[v, u]$ *então*
            $D[u] = D[v] + L[v, u]$
            $P[u] = v$
retornar D, P

AULA 1 - Caminhos mínimos de fonte simples - Dijkstra

TEOREMA: Seja G grafo conexo $w : E \rightarrow \Re_{\geq 0}$. O algoritmo de Dijkstra acha os comprimentos dos caminhos mínimos desde a fonte até qualquer outro vértice do grafo.

AULA 1 - Caminhos mínimos de fonte simples - Dijkstra

Complexidade do pior caso de DIJKSTRA: $O(n^2)$.

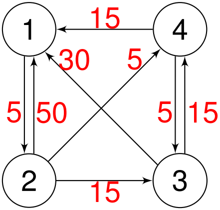
Usando heap (DIJKSTRAheap) pode ser obtida complexidade do pior caso $O(mlog(n))$.

AULA 1 - Caminhos mínimos de fonte simples - Dijkstra

procedimento DIJKSTRAheap(G,w,s)
$Q = \emptyset$ (Q heap)
$S = \emptyset$
para $v \in V$ *faça*
    $d[v] = \infty$
    $\pi[v] = \lambda$
    inserir v em Q
alterar prioridade $d[s]$ de ∞ para 0
enquanto $Q \neq \emptyset$ *faça*
        $min(Q, d[v])$
        $S = S \cup \{v\}$
        para $u \in Adj[v] \cap V - S$ *faça*
            se $d[u] > d[v] + w(v, u)$ *então*
                $\pi[u] = v$
                alterar prioridade de $d[u]$ para $d[v] + w(v, u)$

AULA 1 - Caminhos mínimos de fonte simples - Dijkstra

$G = (V, E)$ grafo (ou digrafo), $\forall e \in E$ $w(e) \geq 0$.

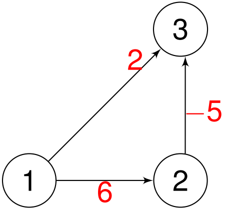


$$L = \begin{bmatrix} 0 & 5 & \infty & \infty \\ 50 & 0 & 15 & 5 \\ 30 & \infty & 0 & 15 \\ 15 & \infty & 5 & 0 \end{bmatrix}$$

passo	v	C	D	P
Inic.	–	{2,3,4}	[0, 5, ∞, ∞]	[-,1,-,-]
1	2	{3,4}	[0, 5, 20, 10]	[-,1,2,2]
2	4	{3}	[0, 5, 15, 10]	[-,1,4,2]

AULA 1 - Caminhos mínimos de fonte simples - Dijkstra

Exemplo no qual DIJKSTRA não funciona ($L(2,3) = -5$).



$$L = \begin{bmatrix} 0 & 6 & 2 \\ \infty & 0 & -5 \\ \infty & \infty & 0 \end{bmatrix}$$

passo	v	C	D	P
Inic.	–	{2,3}	[0, 6, 2]	[-,1,1]
1	3	{2}	[0, 6, 2]	[-,1,1]

AULA 1 - Caminhos mínimos de fonte simples - Dijkstra

OBSERVAÇÕES:

DIJKSTRA pode ser usado para determinar:

caminhos mínimos com peso limitado

caminhos mínimos usando número par de arestas

AULA 1 - Caminhos mínimos entre todos os pares - Floyd-Warshall

Seja $G = (V, E), \forall\, e \in E, w(e) \in \mathbb{R},$.

Problema de caminhos mínimos entre todos os pares de vértices: consiste em achar o comprimento do menor caminho (ou caminho mais curto) entre todos os pares de vértices do grafo.

O algoritmo baseado em Programação Dinâmica que resolve o problema é conhecido como algoritmo FLOYD-WARSHALL.

AULA 1 - Caminhos mínimos entre todos os pares - Floyd-Warshall

Seja $G = (V, E)$ um grafo direcionado, $w(e) \geq 0, \forall\, e \in E$.
Supor os vértices do grafo numerados de 1 até $n = |V|$.

Seja L uma matriz de dimensão $n \times n$ definida por:
$L[i, j] = 0$, se $i = j$
$L[i, j] = w(i, j)$, se $(i, j) \in E$
$L[i, j] = \infty$, se $(i, j) \notin E$

AULA 1 - Caminhos mínimos entre todos os pares - Floyd-Warshall

Seja D uma matriz gerada pelo algoritmo. Inicialmente $D^0 = L$. Na iteração k temos:

$$D^k[i,j] = min\{D^{k-1}[i,j]; D^{k-1}[i,k] + D^{k-1}[k,j]\}$$

procedimento FLOYD1(L, D, n)
para $i = 1, \ldots, n$ *faça*
    para $j = 1$ ate n *faça* $D^0[i,j] = L[i,j]$
para $k = 1, \ldots, n$ *faça*
    para $i = 1, \ldots, n$ *faça*
        para $j = 1, \ldots, n$ *faça*
            se $D^{k-1}[i,k] + D^{k-1}[k,j] < D^{k-1}[i,j]$ *então*
                $D^k[i,j] = D^{k-1}[i,k] + D^{k-1}[k,j]$
            senão $D^k[i,j] = D^{k-1}[i,j]$
retornar D^n

A cada iteração, a linha j e a coluna j da matriz D^k não mudam ($D^k[j,j] = 0$). Isto permite implementar o algoritmo utilizando únicamente duas matrizes (L e D).

AULA 1 - Caminhos mínimos entre todos os pares - Floyd-Warshall

Para obter a sequência de vértices que compõem o caminho mínimo entre cada par de vértices, utilizar a matriz P de dimensão $n \times n$, $P[i, j] = k$ se k precede o vértice j no caminho de i a j.

procedimento FLOYD2(L, D, P, n)
para $i = 1, \ldots, n$ *faça*
  para $j = 1, \ldots, n$ *faça* $D^0[i, j] = L[i, j]$; $P[i, j] = 0$
para $k = 1, \ldots, n$ *faça*
  para $i = 1, \ldots, n$ *faça*
    para $j = 1, \ldots, n$ *faça*
      se $D^{k-1}[i, k] + D^{k-1}[k, j] < D^{k-1}[i, j]$ *então*
        $D^k[i, j] = D^{k-1}[i, k] + D^{k-1}[k, j]$
        $P[i, j] = k$
      senão $D^k[i, j] = D^{k-1}[i, j]$
retornar D^n, P

AULA 1 - Caminhos mínimos entre todos os pares - Floyd-Warshall

A complexidade do algoritmo de Floyd é $O(n^3)$.

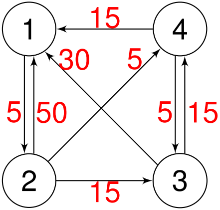
O algoritmo de Dijkstra pode ser usado para resolver o problema com complexidade menor no caso do grafo ser esparso ($m = |E|$ é $O(n)$).

O algoritmo de Floyd-Warshall pode ser usado se existem arestas com peso negativo.

Quando existir ciclo de comprimento negativo num grafo, o algoritmo de Floyd-Warshall não determina corretamente a solução do problema de caminhos mínimos entre todos os pares de vértices.

AULA 1 - Caminhos mínimos entre todos os pares - Floyd-Warshall

$G = (V, E)$, $|V| = n$, $|E| = m$ e $w : E \rightarrow R$, $|V| = n = 4$, L matriz 4×4.



$$L = \begin{bmatrix} 0 & 5 & \infty & \infty \\ 50 & 0 & 15 & 5 \\ 30 & \infty & 0 & 15 \\ 15 & \infty & 5 & 0 \end{bmatrix}.$$

AULA 1 - Caminhos mínimos entre todos os pares - Floyd-Warshall

k	i	j	$D^{k-1}(i,k) + D^{k-1}(k,j)$	$D^{k-1}(k,j)$	$D^k(i,j)$
1	2	3	$50 + \infty$	15	15
	2	4	$50 + \infty$	5	5
	3	2	30 + 5	∞	35
	3	4	$30 + \infty$	15	15
	4	2	15 + 5	∞	20
	4	3	$15 + \infty$	5	5
2	1	3	5 + 15	∞	20
	1	4	5 + 5	∞	10
	3	1	35 + 50	30	30
	3	4	35 + 5	15	15
	4	1	20 + 50	15	15
	4	3	20 + 15	5	5
3	1	2	20 + 35	5	5
	1	4	20 + 15	10	10
	2	1	15 + 30	50	45
	2	4	15 + 15	5	5
	4	1	5 + 30	15	15
	4	2	45 + 35	20	20
4	1	3	10 + 5	20	15
	1	2	10 + 20	5	5
	2	1	5 + 15	45	20
	2	3	5 + 15	15	10
	3	1	15 + 15	30	30
	3	3	15 + 20	35	35

AULA 1 - Caminhos mínimos entre todos os pares - Floyd-Warshall

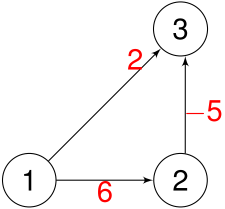
$$D^1 = \begin{bmatrix} 0 & 5 & \infty & \infty \\ 50 & 0 & 15 & 5 \\ 30 & 35 & 0 & 15 \\ 15 & 20 & 5 & 0 \end{bmatrix} \qquad D^2 = \begin{bmatrix} 0 & 5 & 20 & 10 \\ 50 & 0 & 15 & 5 \\ 30 & 35 & 0 & 15 \\ 15 & 20 & 5 & 0 \end{bmatrix}$$

$$D^3 = \begin{bmatrix} 0 & 5 & 20 & 10 \\ 45 & 0 & 15 & 5 \\ 30 & 35 & 0 & 15 \\ 15 & 20 & 5 & 0 \end{bmatrix} \qquad D^4 = \begin{bmatrix} 0 & 5 & 15 & 10 \\ 20 & 0 & 10 & 5 \\ 30 & 35 & 0 & 15 \\ 15 & 20 & 5 & 0 \end{bmatrix}$$

$$P = \begin{bmatrix} - & - & \cancel{2}4 & 2 \\ \cancel{3}4 & - & 4 & - \\ - & 1 & - & - \\ - & 1 & - & - \end{bmatrix}.$$

AULA 1 - Caminhos mínimos entre todos os pares - Floyd-Warshall

$G = (V, E)$, $|V| = n$, $|E| = m$ e $w : E \to R$, $|V| = n = 3$, L matriz 3×3.



$$D^0 = L = \begin{bmatrix} 0 & 6 & 2 \\ \infty & 0 & -5 \\ \infty & \infty & 0 \end{bmatrix}$$

AULA 1 - Caminhos mínimos entre todos os pares - Floyd-Warshall

k	i	j	$D^{k-1}(i,k) + D^{k-1}(k,j)$	$D^{k-1}(k,j)$	$D^{k}(i,j)$
1	2	3	-5 + ∞	-5	-5
	3	2	∞ +6	∞	∞
2	1	3	6 -5	2	1
	3	1	-5 + ∞	∞	∞
3	1	2	2 + ∞	∞	∞
	2	1	-5 + ∞	∞	∞

AULA 1 - Caminhos mínimos entre todos os pares - Floyd-Warshall

$$D^1 = \begin{bmatrix} 0 & 6 & 2 \\ \infty & 0 & -5 \\ \infty & \infty & 0 \end{bmatrix}.$$

$$D^2 = \begin{bmatrix} 0 & 6 & 1 \\ \infty & 0 & -5 \\ \infty & \infty & 0 \end{bmatrix}.$$

$$D^3 = \begin{bmatrix} 0 & 6 & 1 \\ \infty & 0 & -5 \\ \infty & \infty & 0 \end{bmatrix}.$$

$$P = \begin{bmatrix} - & 1 & \cancel{1}2 \\ - & - & 2 \\ - & - & - \end{bmatrix}.$$

AULA 1 - Caminhos mínimos entre todos os pares - Floyd-Warshall

OBSERVAÇÕES:

FLOYD pode ser usado quando for necessário usar alguma ordenação dos vértices no caminho mínimo.