

# Grafos - Representação Computacional

Prof. Andrei Braga



# Conteúdo

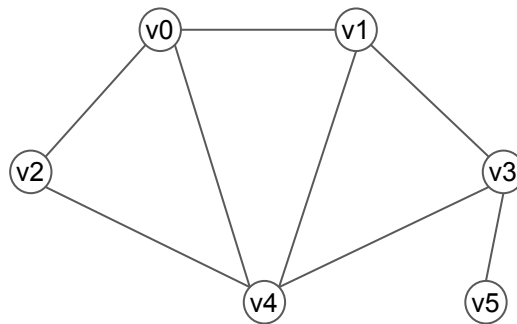
- Representação computacional
- Exercícios
- Referências

# Grafo (Revisão)

- Um **grafo**  $G$  é um par ordenado  $(V, E)$  composto por
  - um conjunto de **vértices**  $V$  e
  - um conjunto de **arestas**  $E$ , sendo cada aresta um conjunto  $\{v_i, v_j\}$  de dois vértices de  $G$ 
    - note que  $\{v_i, v_j\} = \{v_j, v_i\}$ , ou seja, não consideramos uma direção para a aresta

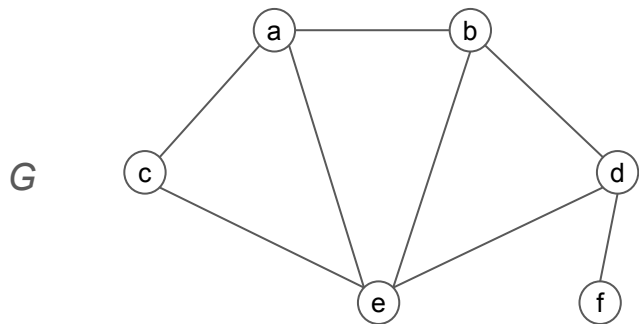
- Exemplo:

- $G = (V, E)$ , onde
  - $V = \{v_0, v_1, v_2, v_3, v_4, v_5\}$  e
  - $E = \{\{v_0, v_1\}, \{v_0, v_2\}, \{v_0, v_4\}, \{v_1, v_3\}, \{v_1, v_4\}, \{v_2, v_4\}, \{v_3, v_4\}, \{v_3, v_5\}\}$



# Representação computacional

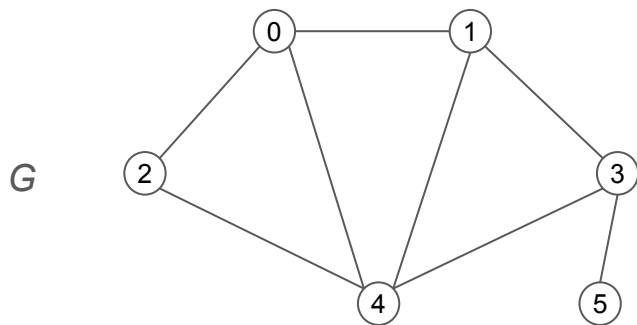
- A seguir, veremos duas formas comuns de representar um grafo  $G$
- Para isso, vamos considerar que fizemos uma associação dos índices  $0, 1, \dots, |V(G)| - 1$  aos vértices de  $G$



- $0 \rightarrow a$
- $1 \rightarrow b$
- $2 \rightarrow c$
- $3 \rightarrow d$
- $4 \rightarrow e$
- $5 \rightarrow f$

# Matriz de adjacências

- A representação de  $G$  como uma **matriz de adjacências** consiste em uma matriz de  $|V(G)|$  linhas, com índices  $0, 1, \dots, |V(G)| - 1$ , e de  $|V(G)|$  colunas, com índices  $0, 1, \dots, |V(G)| - 1$ , tal que a célula  $(i, j)$  da matriz é igual a
  - 1 se  $i, j$  é uma aresta de  $G$
  - 0 caso contrário

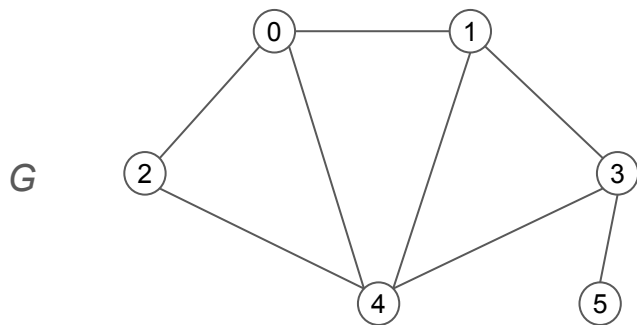


Matriz de adjacências de  $G$

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

# Matriz de adjacências

- Observações:
  - Não é possível representar arestas paralelas
  - Para grafos simples, todas as células da diagonal principal da matriz são iguais a 0
  - Para grafos onde não consideramos uma direção para as arestas, uma aresta  $ij$  é representada por duas células da matriz:  $(i, j)$  e  $(j, i)$

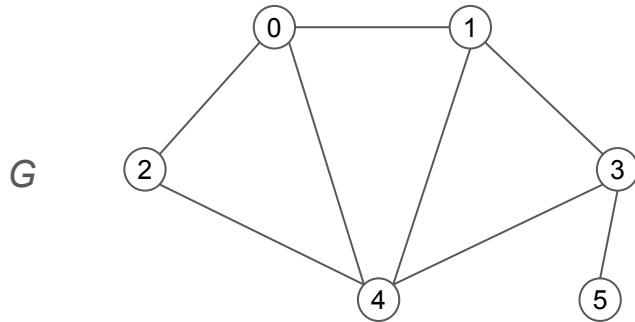


Matriz de adjacências de  $G$

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

# Matriz de adjacências

- Observações:
  - Para grafos onde não consideramos uma direção para as arestas, a matriz é simétrica em relação à diagonal principal

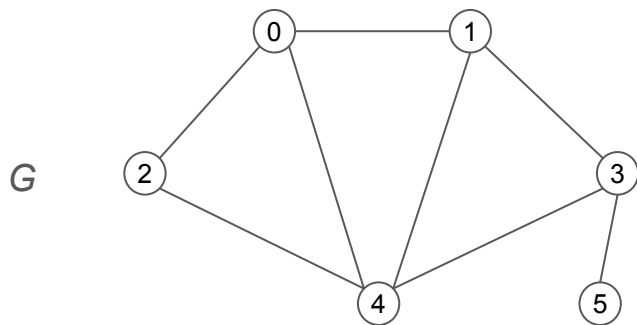


Matriz de adjacências de  $G$

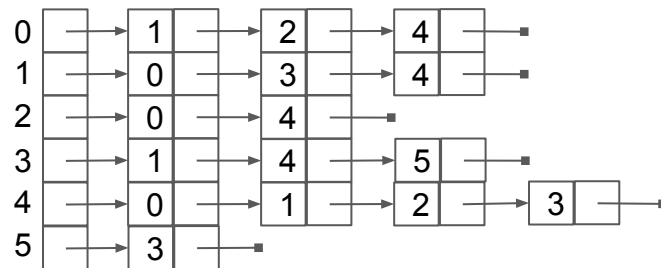
	0	1	2	3	4	5
0	0	1	1	0	1	0
1	1	0	0	1	1	0
2	1	0	0	0	1	0
3	0	1	0	0	1	1
4	1	1	1	1	0	0
5	0	0	0	1	0	0

# Listas de adjacência

- A representação de  $G$  como **listas de adjacência** consiste em um vetor de  $|V(G)|$  elementos, com índices  $0, 1, \dots, |V(G)| - 1$ , tal que o elemento  $i$  do vetor armazena uma lista com os vértices adjacentes ao vértice  $i$  em  $G$



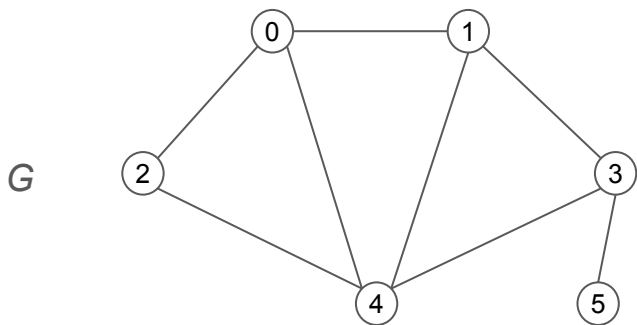
Listas de adjacência de  $G$



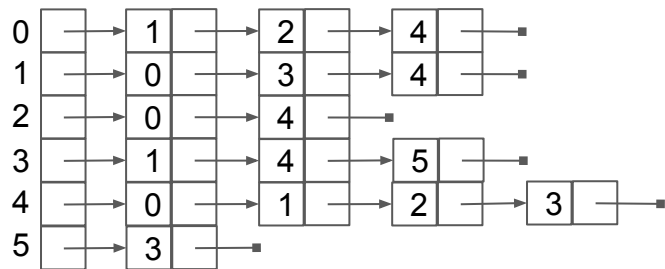


# Listas de adjacência

- Observações:
  - Para grafos onde não consideramos uma direção para as arestas, uma aresta  $ij$  é representada em duas listas de adjacência: o vértice  $i$  está na lista do vértice  $j$  e o vértice  $j$  está na lista do vértice  $i$



Listas de adjacência de  $G$



# Matriz de adjacências vs. listas de adjacência

- Dado um grafo  $G = (V, E)$ , a quantidade de memória utilizada para representar  $G$ 
  - como uma matriz de adjacências é proporcional a  $|V|^2$  e
  - como listas de adjacências é proporcional a  $|V| + |E|$
- Se  $G$  é um grafo **esparso**, isto é,  $|E|$  é bem menor que  $|V|^2$ , então é usualmente mais interessante representar  $G$  como listas de adjacência
- Se  $G$  é um grafo **denso**, isto é,  $|E|$  é um número próximo a  $|V|^2$ , então é usualmente mais interessante representar  $G$  como uma matriz de adjacências

# Matriz de adjacências vs. listas de adjacência

- Dado um grafo  $G = (V, E)$ :

	Matriz de Adjacências	Listas de Adjacência
Memória utilizada	$ V ^2$	$ V  +  E $
Tempo para inserir aresta	constante	constante
Tempo para verificar aresta	constante	pior caso: $ V $
Tempo para remover aresta	constante	pior caso: $ V $

Valores  
proporcionais a

# Exercícios

## 1. Implemente em C++ uma classe que

- a. represente um grafo como uma matriz de adjacências e
- b. permita a realização das seguintes operações no grafo:
  - construir o grafo com um dado número de vértices e sem arestas;
  - obter o número de vértices do grafo;
  - obter o número de arestas do grafo;
  - verificar se uma aresta existe no grafo;
  - inserir uma aresta no grafo;
  - remover uma aresta do grafo;
  - (se necessário) destruir o grafo (liberar a memória alocada para o grafo).

# Referências

- Esta apresentação é baseada nos seguintes materiais:
  - Capítulo 22 do livro  
Cormen, T. H., Leiserson, C. E., Rivest, R. L., Stein, C. Introduction to Algorithms. 3rd. ed. MIT Press, 2009.
  - Capítulo 17 do livro  
Sedgewick, R. Algorithms in C – Part 5. Graph Algorithms. 3rd. ed. Addison-Wesley, 2002.