**Marisangila Alves, MSc**
marisangila.alves@catolicasc.org.br
marisangila.com.br

*Católica de Santa Catarina*

2025/2

# Estrutura de Dados

*Grafo*

# Sumário

# Definição

# História I

- A **teoria dos grafos** teve início com um artigo do matemático **Leonhard Euler**, publicado em **1736**.
- O artigo tratava de um desafio popular na cidade de **Königsberg**, na Prússia (atual Kaliningrado, Rússia).
- A cidade era cortada por um rio e ligada por **sete pontes**, conectando duas ilhas e as margens do rio.

# História II

- O problema era: **seria possível caminhar pela cidade cruzando todas as pontes uma única vez, sem repetir nenhuma?**
- Euler representou o problema usando **pontos (vértices)** para as terras e **linhas (arestas)** para as pontes, criando um modelo abstrato.
- Ele provou que **não era possível** fazer tal caminhada, e com isso, estabeleceu os fundamentos da **teoria dos grafos**.

# História III

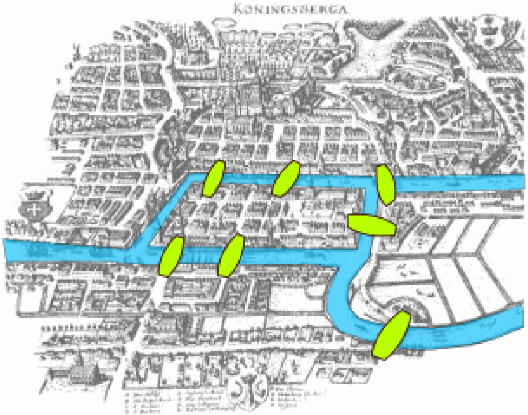


Figura 1: Mapa das pontes de Königsberg

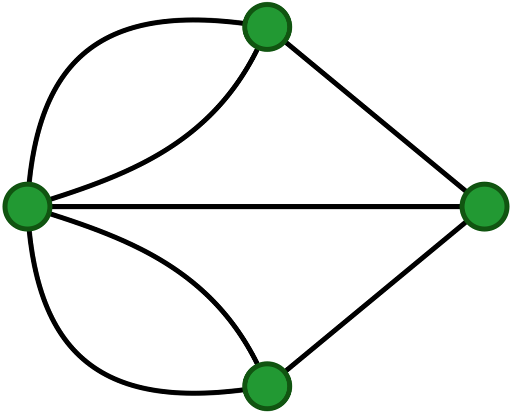


Figura 2: Representação em grafo

## Grafo

> *Um grafo é uma estrutura de dados que representa uma* ***relação de conectividade*** *entre nós.*
> *Cada nó representa um vértice e cada conexão representa uma aresta do grafo.*
>
> (CELES; CERQUEIRA; RANGEL, 2004)

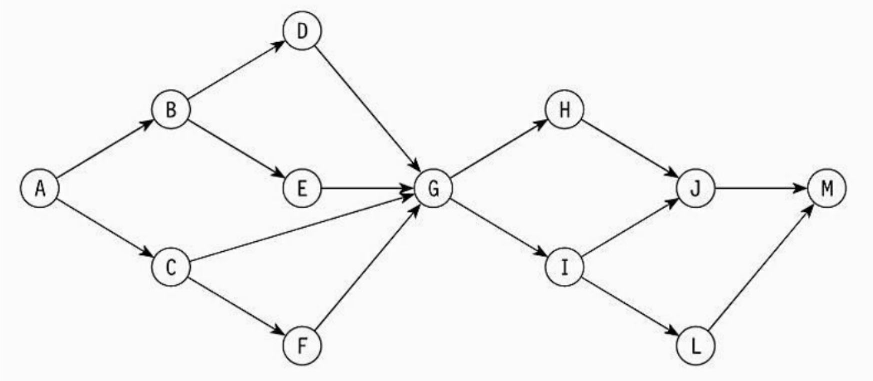
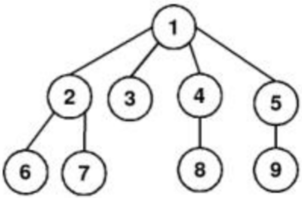


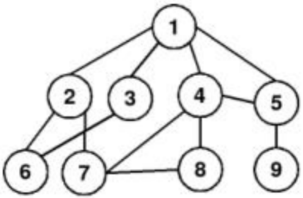

Figura 3: Exemplo de grafo (FORBELLONE; EBERSPÄCHER, 2010).

**Nota:**

Árvore são grafos em que não existem ciclos!



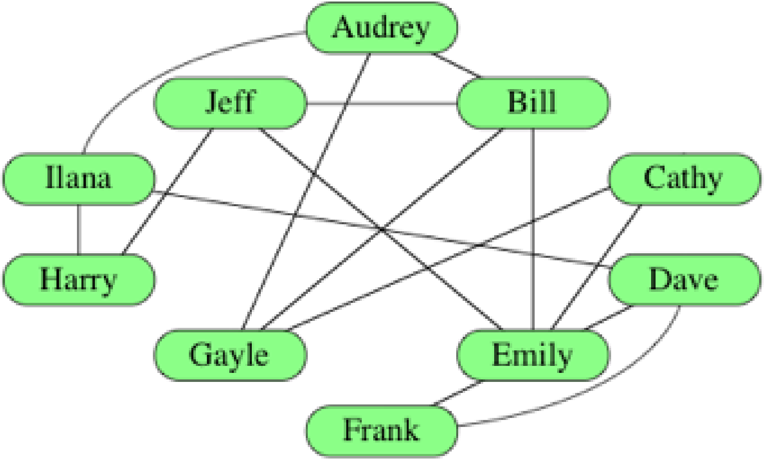

Uma árvore




Um grafo comum com ciclos

- Um grafo é uma estrutura matemática que representa um conjunto de objetos chamados vértices (ou nós) e as conexões entre eles chamadas arestas.
- Grafos são um **poderoso formalismo** para modelar (abstrair) uma série de problemas em **Ciência da Computação**.
- Representam o **relacionamento entre elementos** de um conjunto.

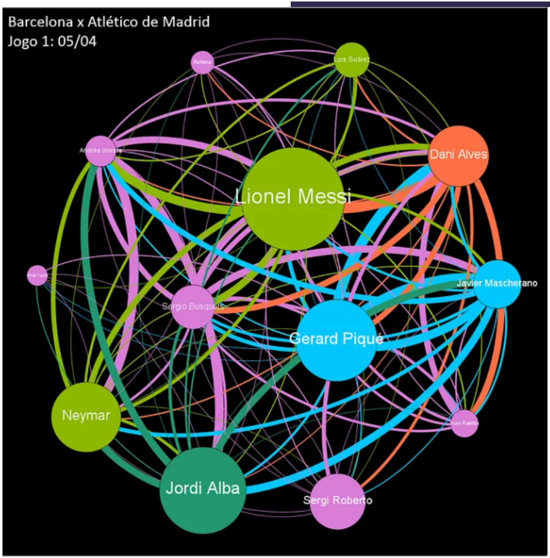
- Algoritmos de Redes de Computadores;
- Representação de um mapa - Transporte e Logística;
- Planejamento de Tarefas;
- Modelagem de redes sociais;
- Representação de redes de fluxo;
- Algortimos de Machine Learning - Redes Neurais;
- Sistemas de Recomendação;
- Banco de Dados.

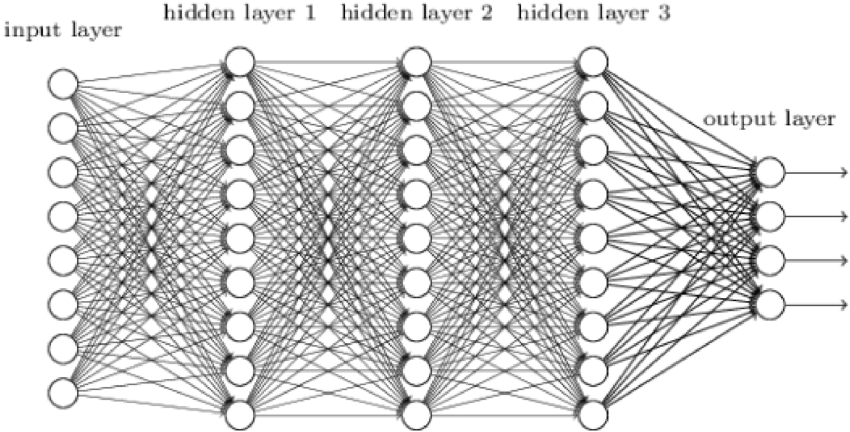

Audrey
Jeff
Bill
Ilana
Cathy
Harry
Dave
Gayle
Emily
Frank

# Aplicações III

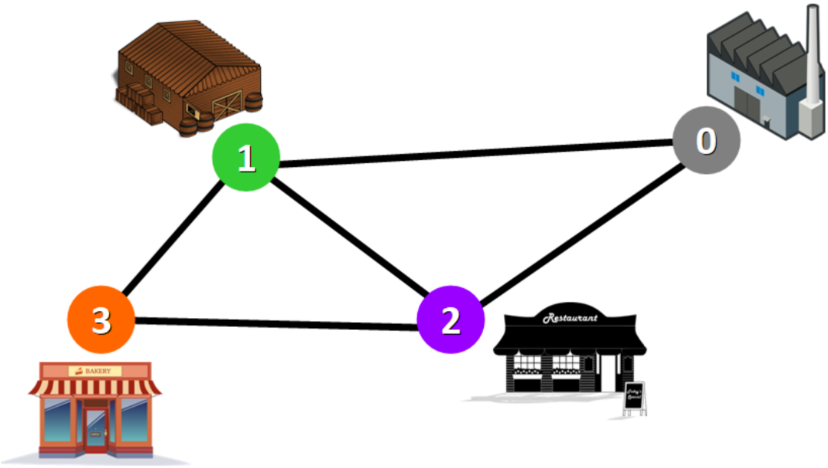

RR
AP
AM
PA
MA
CE
RN
PI
PB
PE
AC
AL
TO
RO
BA
SE
DF
MT
GO
MG
ES
MS
SP
RJ
PR
SC
RS

# Aplicações V






*Machine Learning* - Rede Neural Profunda.


1
0
3
2
BAKERY
Restaurant

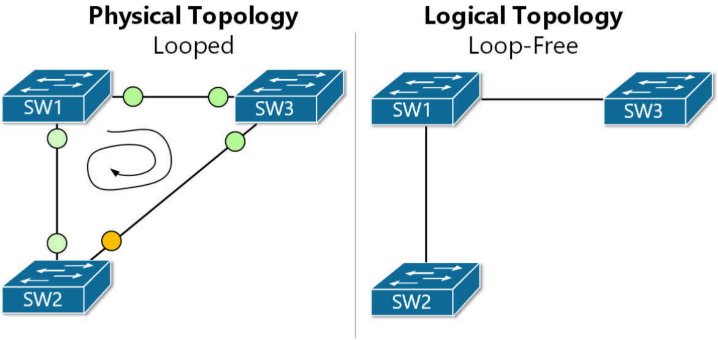
# Exemplo: Spanning Tree Protocol (STP) I

## Árvore Geradora Mínima

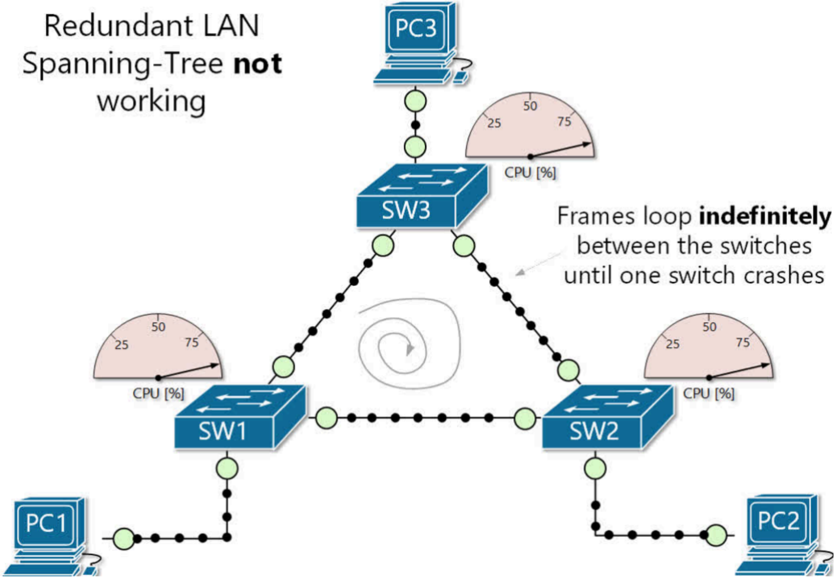
- Dado um grafo conectado e ponderado, uma **Árvore Geradora** é um subconjunto de arestas que conecta todos os vértices **sem formar ciclos**.
- A **Árvore Geradora Mínima** é aquela cuja soma dos pesos das arestas é a **menor possível**.

- Em redes Ethernet, *loops* (ciclos) causam tempestades de *broadcast*.
- O **Spanning Tree Protocol (STP)** usa um algoritmo inspirado em grafos:
- Ele constrói uma **árvore geradora** das conexões da rede, **bloqueando algumas portas** para impedir ciclos.

# Exemplo: Spanning Tree Protocol (STP) II



# Exemplo: Spanning Tree Protocol (STP) III



# Terminologia

Um grafo é, portanto, representado por um conjunto de vértices e arestas: $G = (V, E)$.

- O número de vértices do grafo é $|V|$ e o número de arestas é $|E|$.

## Grafo Orientado (Dirigido) e Não Orientado (Não Dirigido)

- Um grafo pode ser **orientado** (também chamado de *dirigido* ou *direcionado*) ou **não orientado**.
- Em um **grafo não orientado**, cada **aresta** representa uma conexão entre os dois **vértices** ligados, **nos dois sentidos**.
- Assim, num grafo não orientado, se existe uma aresta entre os vértices $a$ e $b$, isso significa que existe uma conexão de $\boldsymbol{a}$ **para** $\boldsymbol{b}$ ($a \rightarrow b$) e de $\boldsymbol{b}$ **para** $\boldsymbol{a}$ ($b \rightarrow a$).
- Já num **grafo orientado**, cada **aresta possui uma direção**, ou seja, a conexão ocorre **em apenas um sentido**.

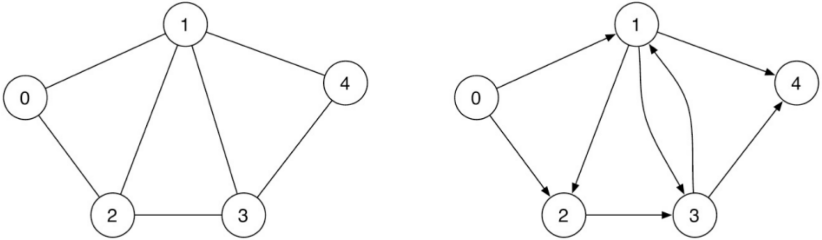
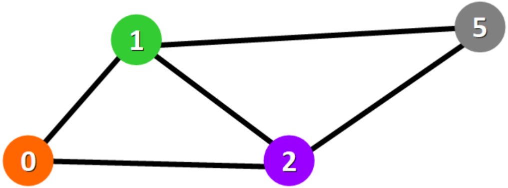



Figura 4: Exemplo de grafo não orientado e orientado (CELES; CERQUEIRA; RANGEL, 2004).

## Grafo Ponderado e Não Ponderado

- **Grafo ponderado**: cada aresta possui um **peso** ou **custo** associado, que pode representar distância, tempo, capacidade, entre outros.
- **Grafo não ponderado**: as arestas não possuem peso; elas indicam apenas a existência de uma conexão entre os vértices.



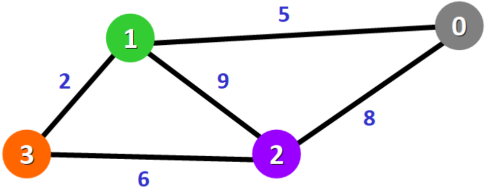
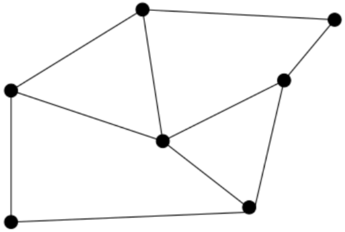
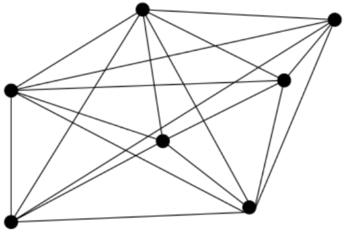




Figura 5: Representação em grafo

## Grafo Esparso e Denso

- Grafos são esparsos se temos apenas uma pequena fração de arestas sobre os possíveis pares de vértice.
- Grafos densos possuem uma grande porção de ligações entre os vértices.





# Representação

# Representação Computacional I

Podemos escolher entre dois modos padrões para representar um grafo G = (V, E): como uma coleção de listas de adjacências ou como uma matriz de adjacências. (Cormen *et al.*, 2012)

- Na estrutura por **lista de adjacência**, em geral, representamos um **vetor de vértices**.
- Cada **vértice** guarda uma **lista de arestas** que **partem deste vértice** (isto é, arestas cujo vértice em questão é o **vértice origem**).
- Na estrutura por **matriz de adjacência**, representamos uma **matriz** de dimensão $V \times V$.
- Cada elemento $m_{ij}$ da matriz representa uma **possível aresta** conectando o *vértice* $v_i$ ao *vértice* $v_j$.

## Representação Computacional II

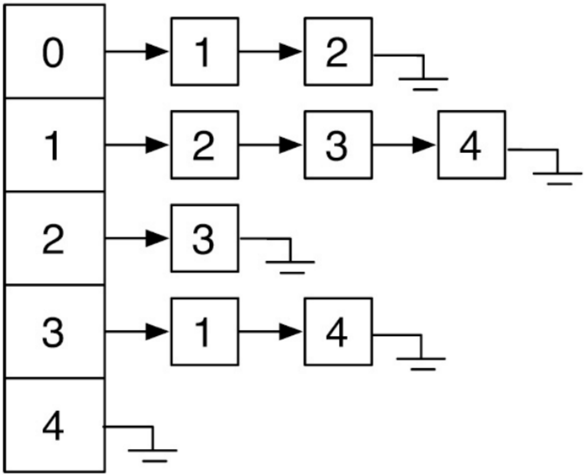



Figura 6: Lista de adjacência de um grafo orientado (CELES; CERQUEIRA; RANGEL, 2004)

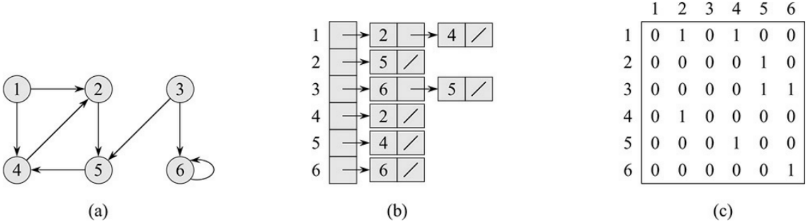



Figura 7: Lista de adjacência de um grafo orientado (CELES; CERQUEIRA; RANGEL, 2004)

# Representação Computacional IV

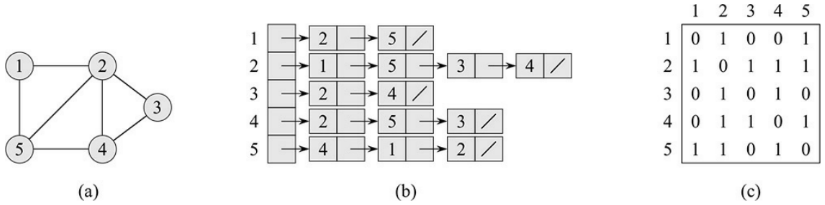



Figura 8: Lista de adjacência de um grafo não orientado (CELES; CERQUEIRA; RANGEL, 2004)

# Representação Computacional V

- O espaço de memória requerido para representar um grafo por **lista de adjacência** é $O(V + E)$.
- Já a representação por **matriz de adjacência** requer espaço $O(V^2)$.
- Assim, a **lista de adjacência** é mais eficiente para **grafos esparsos**.
- A **matriz de adjacência** é preferível para **grafos densos**, em que o número de arestas é **próximo ao máximo possível** $(E \approx V^2)$.

# Implementação I

# Busca em Profundidade

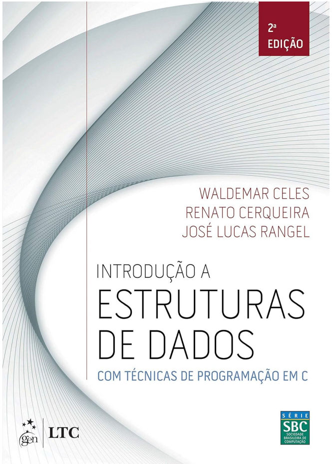
# Busca em Largura

# Caixeiro Viajante

# Caminho Mínimo

# Dijkstra

(CELES; CERQUEIRA; RANGEL, 2004) - Capítulo 22



# Referências I


CELES, Waldemar; CERQUEIRA, Renato; RANGEL, José Lucas. **Introdução a estruturas de dados: com técnicas de programação em C**. Rio de Janeiro: Elsevier, 2004.

CORMEN, T.H. *et al.* **Algoritmos: Teoria e Prática**. 3. ed. Rio de Janeiro: Elsevier, 2012.

FORBELLONE, André Luiz Villar; EBERSPÄCHER, Henri Frederico. **Lógica de programação: a construção de algoritmos e estrutura de dados**. 3. ed. São Paulo: Pearson Prentice Hall, 2010.

## Estes slides estão protegidos por uma licença Creative Commons

Este modelo foi adaptado de Maxime Chupin.

**Marisangila Alves, MSc**
marisangila.alves@catolicasc.org.br
marisangila.com.br
*Católica de Santa Catarina*

2025/2

# Estrutura de Dados

*Grafo*