# Grafos: Conceitos Preliminares

## Tópicos Especiais em Algoritmos - Ciência da Computação

Prof. Daniel Saad Nogueira Nunes

IFB – Instituto Federal de Brasília, Campus Taguatinga

# Sumário

# Sumário

# Introdução

- Muitos problemas em Ciência da Computação são modelados em formas de relacionamento entre objetos, no sentido amplo da palavra.
- Precisamos de formalismo que consegue modelar relações presentes desde problemas envolvendo interações entre pessoas à problemas envolvendo redes gigantescas de computadores.
- A chave para resolução de muitos problemas computacionais pode residir em um único formalismo, os **grafos**.

# Introdução

- A Teoria dos Grafos provê uma linguagem para falar de propriedades e relacionamentos dos objetos mencionados.
- Projetar um algoritmo novo usando grafos é extremamente complicado, contudo muitas das vezes só precisamos apenas utilizar um algoritmo já conhecido.
- Às vezes o mais difícil é modelar o problema em termos de grafos!

# Introdução

**Definição (Grafo)**

Um grafo é uma dupla $G = (V, E)$, em que $V$ é o conjunto de vértices e $E \subseteq V \times V$ é o conjunto de arestas.

# Introdução

- Repare que as arestas formam uma relação sobre o conjunto dos pares de vértices.
- Por exemplo, os vértices $v \in V$ poderiam representar cidades, enquanto uma aresta $(u, v)$ informaria que estive uma rodovia entre a cidade $u$ e $v$.
- As arestas representam relacionamentos entre os objetos!

# Sumário

# Introdução

## Tipos de Grafos

- Existem diversas especialidades de grafos.
- Cada qual com suas propriedades distintas, o que faz o seu uso mais adequado em determinados problemas:
  1. Simples $\times$ Não-simples.
  2. Dirigido $\times$ Não-dirigido.
  3. Com peso $\times$ Sem peso.
  4. Esparso $\times$ Denso.
  5. Cíclico $\times$ Acíclico.
  6. Incorporado $\times$ Topológico.
  7. Implícito $\times$ Explícito.
  8. Rotulado $\times$ Não-rotulado.

# Tipos de Grafos

## Simples $\times$ Não-simples

- Grafos simples não possuem estruturas complexas, tais como:
  - Loops: arestas que ligam o vértice nele mesmo.
  - Multiarestas: podemos ter várias arestas ligando dois vértices.

# Tipos de Grafos

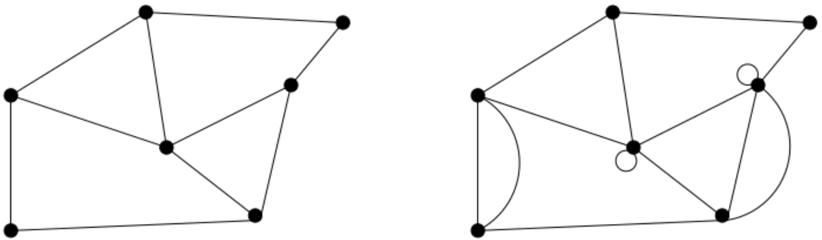


Figura: Simples × Não-simples.

# Tipos de Grafos

## Dirigido $\times$ Não-dirigido

- Um grafo é não-dirigido se a direção das arestas não importa.
- Um grafo é dirigido se podemos ter arestas em uma única direção.
  - Muito útil para modelas problemas específicos.
  - Exemplo: uma via de mão única.

# Tipos de Grafos

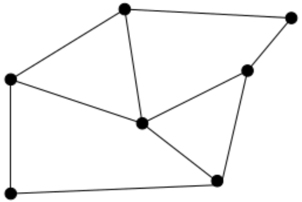
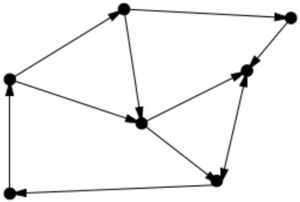



Figura: Dirigido × Não-dirigido.

# Tipos de Grafos

## Com Peso × Sem Peso

- Em um grafo com peso nas areas, para cada aresta $(u, v)$, temos um peso relacionado a ela. que pode ser por exemplo números inteiros ou reais.
  - Muito utilizado em problemas de otimização, como o problema do menor caminho.

# Tipos de Grafos

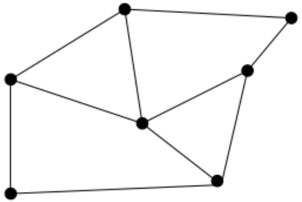


Figura: Dirigido × Não-dirigido.

# Tipos de Grafos

## Esparso × Denso

- Em grafos simples, podemos ter $\binom{n}{2}$ pares de vértices.
- 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.
- Não há uma regra geral, geralmente dizemos que um grafo é denso se $|E| \in \Theta(n^2)$.

# Tipos de Grafos

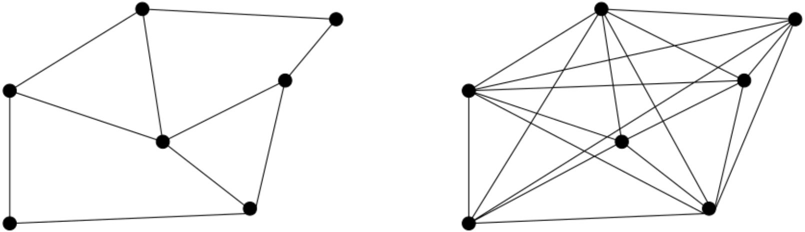


Figura: Esparso × Denso.

# Tipos de Grafos

## Cíclicos $\times$ Acíclicos

- Um grafo acíclico são grafos que não possuem ciclos.

# Tipos de Grafos

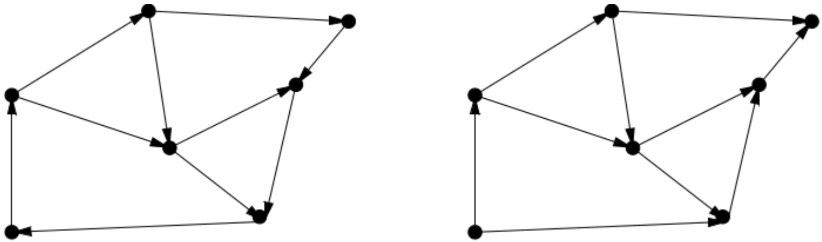


Figura: Cíclico $\times$ Acíclico.

# Tipos de Grafos

## Incorporado × Topológico

- Um grafo é incorporado se seus vértices estão mapeados em posições geométricas, como em um grid.
- Isso pode ter relevância em alguns problemas.
- Em problemas que isto não é importante, nos importamos apenas com a topologia do grafo (o esqueleto).

# Tipos de Grafos

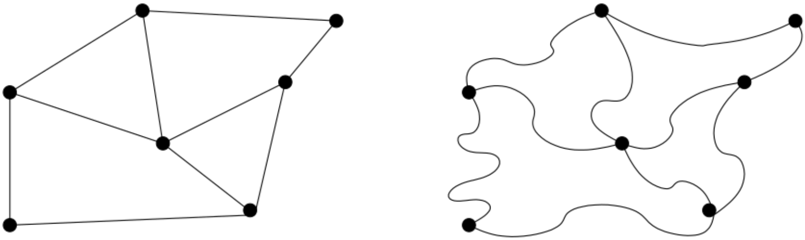


Figura: Incorporado × Topológico.

# Tipos de Grafos

## Implícitos × Explícitos

- Grafos ímplicitos vão sendo construídos conforme vamos utilizando eles.
  - Backtracking, simulação...
- Em outros casos, precisamos do grafo já construído para resolver certos problemas.

# Tipos de Grafos

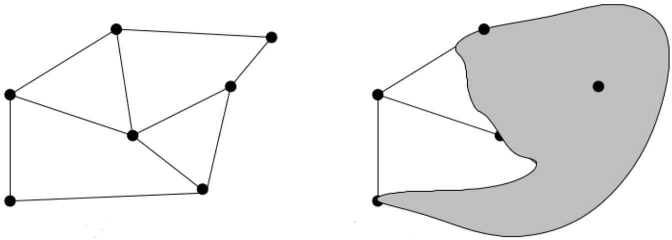


Figura: Implícito $\times$ Explícito.

# Tipos de Grafos

## Rotulado $\times$ Não rotulados

- Se o grafo é rotulado, a cada vértice é atribuído um rótulo que o identifica unicamente.
- Em grafos sem rótulo, não temos essa distinção.

# Tipos de Grafos

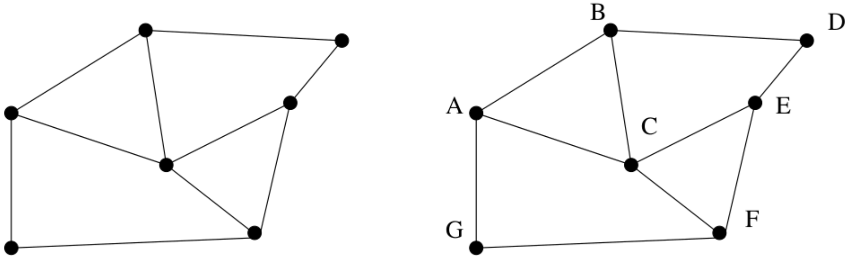



Figura: Rotulado × Não-rotulado.

# Sumário

# Aplicações

- Usando esse formalismo, podemos resolver problemas reais!
- Desde problemas biológicos como problemas em rede de computadores!

# Aplicações

Figura: Navegação.

# Aplicações

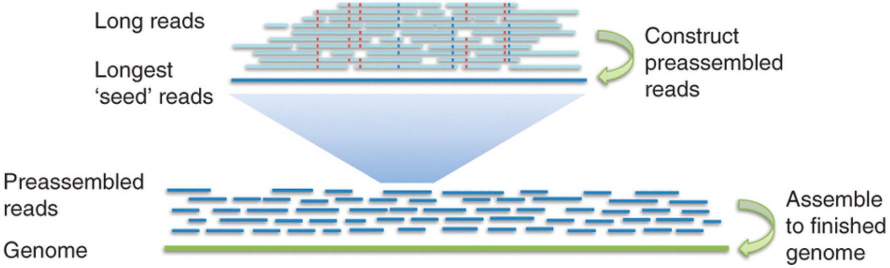



Figura: Montagem de Genomas.

# Aplicações

Figura: Análise de Tráfego.

# Aplicações

Figura: Redes de Computadores.

# Exemplos

## Exemplo

- Vamos começar nosso estudo desse incrível formalismo com uma modelagem simples.
- O grafo de relacionamento de pessoas!
  - Nosso Facebook.

# Aplicações

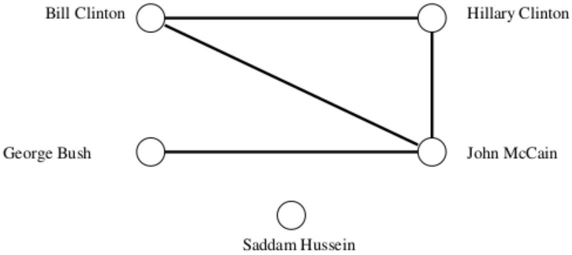



Figura: Grafo de relacionamento de pessoas.

# Aplicações

## Exemplo

- Através do grafo de relacionamento, podemos responder várias perguntas interessantes:
  - Meu amigo também me considera como amigo?
  - Qual é o nível da nossa amizade?
  - Eu sou amigo de mim mesmo?
  - Quem tem mais amigos?
  - Meus amigos moram perto de mim?
  - Você conhece esta pessoa?
  - Você é um indivíduo ou apenas um rosto?

# Aplicações

## Exemplo

- Meu amigo também me considera como amigo?

# Aplicações

### Exemplo

- Meu amigo também me considera como amigo?
- Existe uma aresta do seu amigo pra você?

# Aplicações

## Exemplo

- Qual é o nível da nossa amizade?

# Aplicações

## Exemplo

- Qual é o nível da nossa amizade?
- Quanto é o peso sobre a aresta que nos liga?

# Aplicações

## Exemplo

- Eu sou amigo de mim mesmo?

# Aplicações

## Exemplo

- Eu sou amigo de mim mesmo?
- O grafo é simples? Possuo um loop pra mim mesmo?

# Aplicações

### Exemplo

- Quem tem mais amigos?

# Aplicações

## Exemplo

- Quem tem mais amigos?
- Qual é o vértice que tem mais arestas saindo dele?

# Aplicações

## Exemplo

- Meus amigos moram perto de mim?

# Aplicações

## Exemplo

- Meus amigos moram perto de mim?
- Dado que o grafo é incorporado, qual a distância do seu vértice ao dos seus amigos?

# Aplicações

## Exemplo

- Você é um indivíduo ou apenas um rosto?

# Aplicações

## Exemplo

- Você é um indivíduo ou apenas um rosto?
- O grafo é rotulado?

# Sumário

# Conceitos Fundamentais

## Definição (Grau de Entrada)

- Definido sobre um nó $v$.
- Representa o número de arestas que chegam em um nó $v$.

# Conceitos Fundamentais

## Definição (Grau de Saída)

- Definido sobre um nó $v$.
- Representa o número de arestas que saem de um nó $v$.
- **OBS:** em um grafo não direcionado, o grau de entrada de cada vértice é igual ao grau de saída.

# Conceitos Fundamentais

## Definição (Caminho)

- Um caminho é uma sequência de arestas que conecta vértices distintos.

# Conceitos Fundamentais

## Definição (Conectividade)

- Um grafo não-dirigido é dito conexo se existe um caminho para qualquer dois pares de vértices.
- Um grafo com apenas um vértice também é considerado conexo.

# Conceitos Fundamentais

## Definição (Componente Conexa)

- Uma componente conexa de um grafo não dirigido é um subgrafo maximal conexo do grafo original.

# Conceitos Fundamentais

## Definição (Conectividade Fraca)

- Um grafo **dirigido** é dito fracamente conexo se ao trocarmos suas arestas pela versão não dirigida, obtemos um grafo conexo.

# Conceitos Fundamentais

## Definição (Conectividade Forte)

- Um grafo **dirigido** é dito fortemente conexo se para quaisquer par $u$ e $v$ de vértices, existe um caminho de $u$ para $v$ e um de $v$ para $u$.

# Conceitos Fundamentais

## Definição (Corte)

- Um corte é um conjunto de vértices que separa o grafo, isto é, que o deixa com mais de uma componente conexa.

# Conceitos Fundamentais

## Definição ($k$-conectividade)

- Um grafo não-dirigido é dito $k$-conexo se não existe um conjunto de $k - 1$ vértices, que desconecta o grafo.

# Sumário

# Representação de Grafos

- Como representar grafos computacionalmente?
- Temos que escolher uma representação eficiente.
- Escolhas mais comuns:
  1. Listas de Adjacências.
  2. Matrizes de Adjacências.

# Representação de Grafos

## Listas de Adjacência

- As listas de adjacências consistem de um vetor de tamanho $|V|$ de listas encadeada.
- Cada elemento do vetor, aponta para uma lista encadeada.
- Suponha o $i$-ésimo elemento deste vetor. Ele apontará para uma lista encadeada que contém as arestas que saem do nó $i$.

# Listas de Adjacência

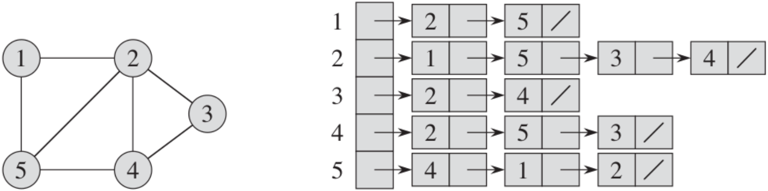



Figura: Lista de adjacências.

# Listas de Adjacência

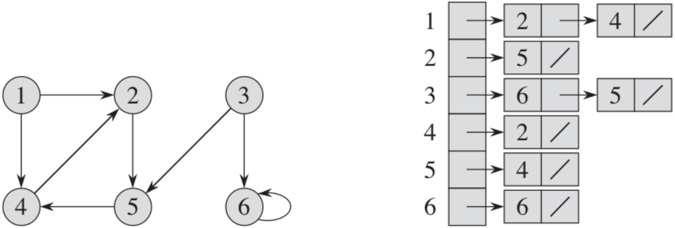


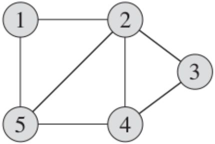

Figura: Lista de adjacências.

# Representação de Grafos

## Matrizes de Adjacências

- As matrizes de adjacências, como um nome diz, é uma matriz.
- O elemento $M[i][j]$, indica se existe uma aresta entre os nós $i$ e $j$.

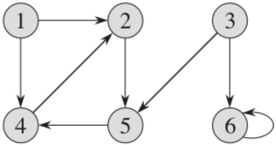
# Matrizes de Adjacências




$$
\begin{array}{c|ccccc|}
 & 1 & 2 & 3 & 4 & 5 \\
\hline
1 & 0 & 1 & 0 & 0 & 1 \\
2 & 1 & 0 & 1 & 1 & 1 \\
3 & 0 & 1 & 0 & 1 & 0 \\
4 & 0 & 1 & 1 & 0 & 1 \\
5 & 1 & 1 & 0 & 1 & 0 \\
\hline
\end{array}
$$

Figura: Matriz de Adjacências.

# Matrizes de Adjacências




| | 1 | 2 | 3 | 4 | 5 | 6 |
|---|---|---|---|---|---|---|
| 1 | 0 | 1 | 0 | 1 | 0 | 0 |
| 2 | 0 | 0 | 0 | 0 | 1 | 0 |
| 3 | 0 | 0 | 0 | 0 | 1 | 1 |
| 4 | 0 | 1 | 0 | 0 | 0 | 0 |
| 5 | 0 | 0 | 0 | 1 | 0 | 0 |
| 6 | 0 | 0 | 0 | 0 | 0 | 1 |

Figura: Matriz de Adjacências.

# Listas vs Matrizes de Adjacências

- Cada abordagem tem seus pontos fortes e fracos.
- Listas de adjacência são mais econômicas em espaço quando o grafo é esparso.
- Matrizes de adjacência permitem acesso em tempo constante a qualquer aresta.
- Qual utilizar?

# Listas vs Matrizes de Adjacências

Tabela: Comparação entre listas e matrizes de adjacências.

| Critério | Ganhador |
|---|---|
| Tempo de acesso em arestas | Matriz |
| Verificar o grau do vértice | Lista |
| Consumo de memória em grafos esparsos | Lista |
| Consumo de memória em grafos densos | Matriz |
| Inserção/remoção de arestas | Matriz |
| Percurso do grafo | Listas |

# Sumário

# Considerações

- Grafos: poderoso formalismo para modelar uma série de problemas em Ciência da Computação.
- Representam um relacionamento entre elementos de um conjunto.
- Através de seus algoritmos, podemos resolver esses problemas eficientemente.
- Muitas das vezes só precisamos utilizar um algoritmo padrão de grafos com uma leve modificação para resolver o problema.