

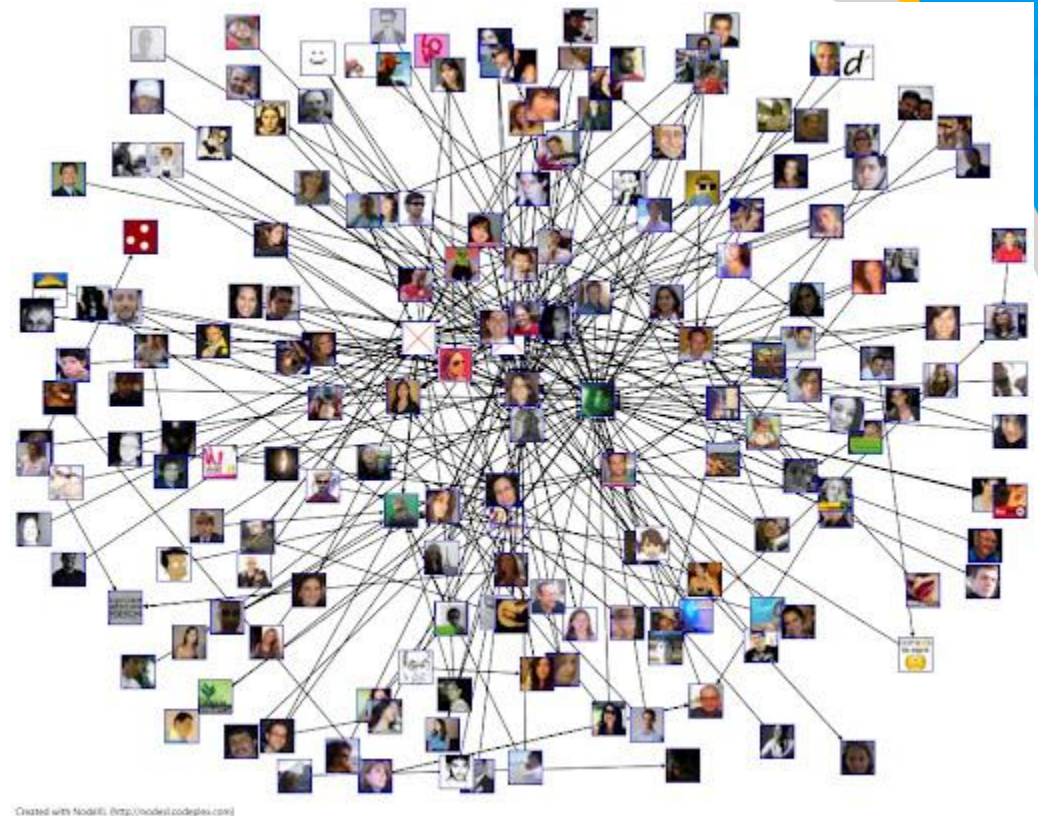
Representação Computacional de Grafos

Prof Edson Orivaldo Lessa Junior



Estruturas de Dados para Grafos

- Por que precisamos de estruturas específicas?
- Volume de dados e relacionamentos complexos
- Eficiência na consulta e manipulação
- Exemplo: redes sociais, mapas, logística

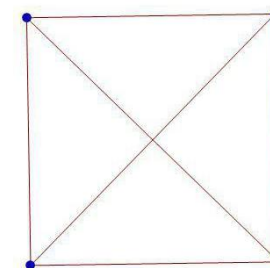
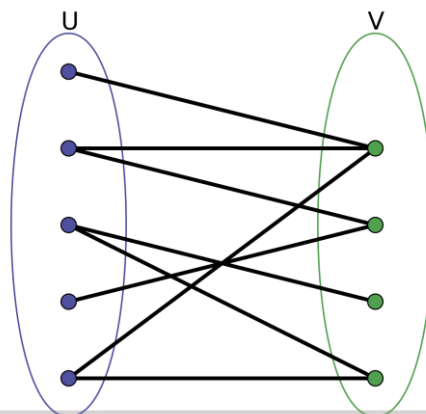
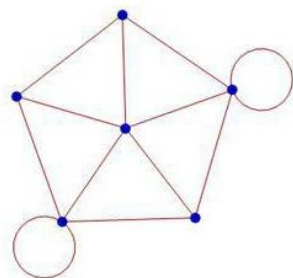
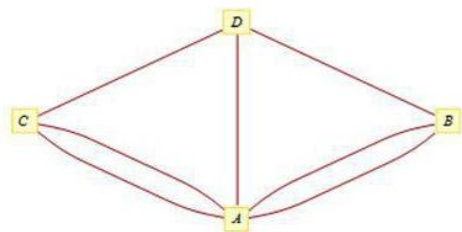
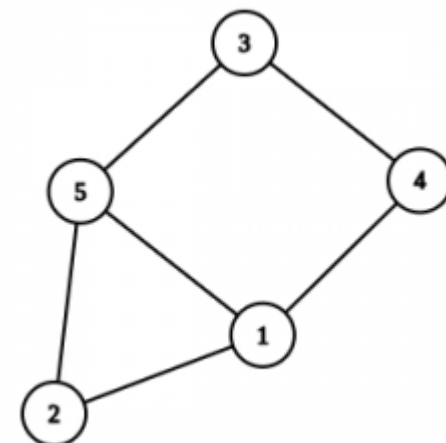
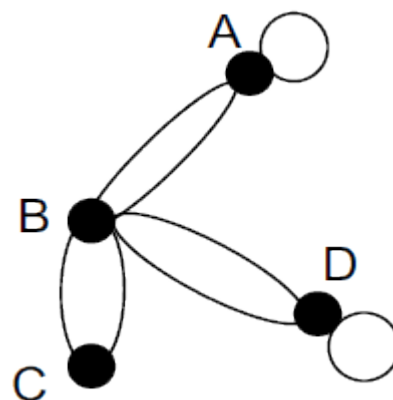


Elementos Fundamentais de um Grafo

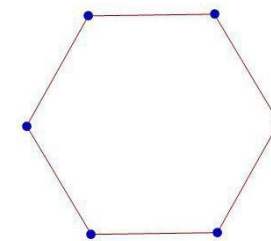
- Conjunto de vértices (nós)
- Conjunto de arestas (ligações)
- Grafo direcionado vs não direcionado
- Peso das arestas e grafos ponderados

Tipos de Grafos e Suas Representações

- Grafos simples
- Grafos completos
- Grafos bipartidos
- Grafos com laços e múltiplas arestas



Completo

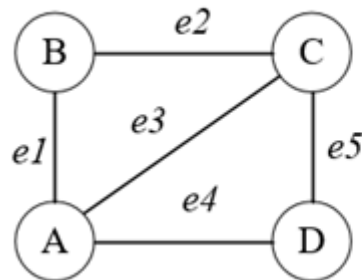


Não completo

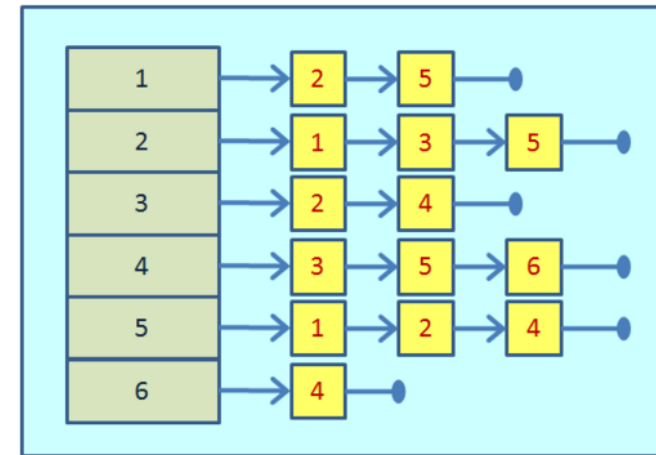
Principais Formas de Representação

- Matriz de adjacência
- Lista de adjacência
- Matriz de incidência
- Outras variações menos comuns

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



	<i>e1</i>	<i>e2</i>	<i>e3</i>	<i>e4</i>	<i>e5</i>
A	1	0	1	1	0
B	1	1	0	0	0
C	0	1	1	0	1
D	0	0	0	1	1



Matriz de Adjacência: Conceito

- Estrutura em formato de matriz $N \times N$
- Cada célula indica conexão entre vértices
- Uso para grafos pequenos ou densos
- Exemplo visual de matriz

Matriz de Adjacência: Implementação em C

- Definição da matriz como vetor bidimensional
- Atribuição de valores (0 ou 1, ou peso)
- Exemplo de código em C
- Pontos de atenção na declaração
- <https://github.com/edson-lessa-jr/unicesumar-aula8-estrutura-de-dados>
 - 01_matriz_adjacencia

Vantagens da Matriz de Adjacência

- Acesso rápido a arestas ($O(1)$)
- Simplicidade na implementação
- Facilidade para operações de verificação
- Ideal para grafos densos

Desvantagens da Matriz de Adjacência

- Uso elevado de memória ($N \times N$)
- Pouco eficiente para grafos esparsos
- Limitação no tratamento de grafos muito grandes
- Atualização pode ser custosa em grandes estruturas

Lista de Adjacência: Conceito

- Estrutura baseada em listas ligadas ou vetores de listas
- Cada vértice armazena lista de vizinhos
- Eficiência em grafos esparsos
- Exemplo visual

Lista de Adjacência: Implementação em C

- Estrutura usando ponteiros e listas ligadas
- Definição de struct para vértices e arestas
- Exemplo de código do repositório
- Dicas de organização do código
- <https://github.com/edson-lessa-jr/unicesumar-aula8-estrutura-de-dados>
 - 02_lista_adjacencia

Vantagens da Lista de Adjacência

- Uso eficiente de memória para grafos esparsos
- Facilidade para percorrer vizinhos
- Crescimento dinâmico das listas
- Flexibilidade para grafos grandes

Desvantagens da Lista de Adjacência

- Acesso menos eficiente entre vértices distantes ($O(k)$)
- Mais complexidade para buscar existência de aresta
- Cuidado com manipulação de ponteiros em C
- Pode dificultar operações de remoção em massa

Comparando Matriz e Lista de Adjacência

- Eficiência em diferentes cenários
- Critérios para escolha da estrutura
- Exemplos de quando usar cada uma
- Relevância para algoritmos clássicos

Matriz de Incidência: Conceito

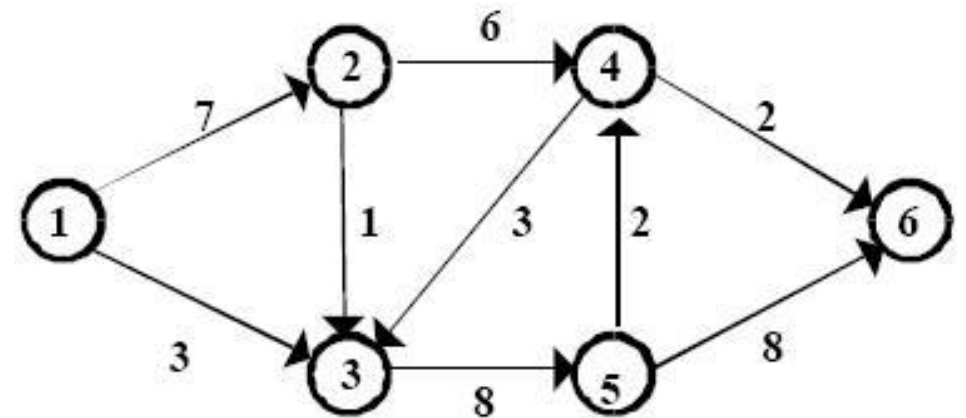
- Estrutura menos comum, mas útil em casos específicos
- Matrizes que relacionam vértices e arestas
- Usada em aplicações de redes elétricas e fluxos
- Exemplo visual
- <https://github.com/edson-lessa-jr/unicesumar-aula8-estrutura-de-dados>
 - 03_matriz_adj_nao_direcionado

Implementação de Matriz de Incidência em C

- Definição da matriz ($V \times E$)
- Atribuição de valores: +1, -1, 0
- Exemplo básico de código
- Situações práticas de uso
- <https://github.com/edson-lessa-jr/unicesumar-aula8-estrutura-de-dados>
 - 04_matriz_adj_direcionado

Casos Especiais: Grafos Ponderados

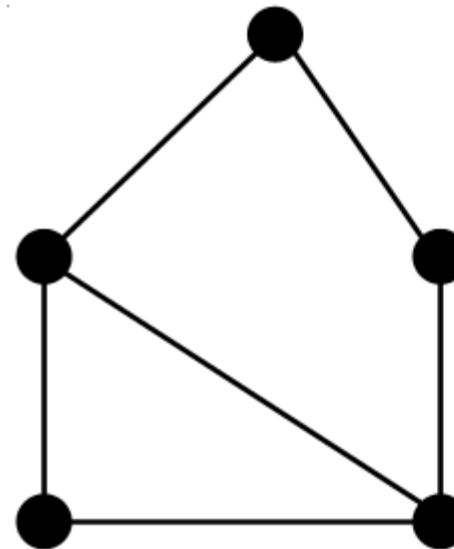
- Inclusão de pesos nas arestas
- Modificações na matriz/lista para armazenar valores
- Exemplo de grafo com pesos
- Aplicações em caminhos mínimos e logística



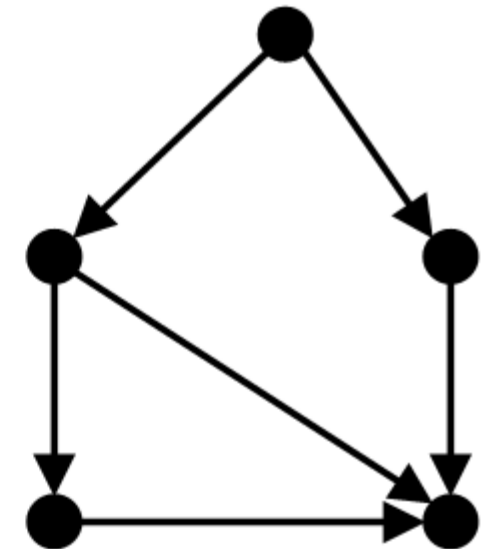
Grafos Direcionados vs Não Direcionados

- Representação de direcionamento nas estruturas
- Diferença de preenchimento na matriz/lista
- Exemplo comparativo de código
- Impactos nos algoritmos de busca

Grafo não-direcionado



Grafo direcionado



Exemplo Prático: Criação de Grafo por Matriz

- Exemplo de declaração e inicialização
- Inserção de arestas manualmente
- Código disponível no repositório
- Demonstração de consulta de adjacência

Exemplo Prático: Criação de Grafo por Lista

- Estrutura do código em C
- Inserção de vértices e arestas
- Uso de funções auxiliares
- Demonstração de busca por vizinhos
- <https://github.com/edson-lessa-jr/unicesumar-aula8-estrutura-de-dados>
 - 05_grafo

Visualizando a Estrutura do Grafo

- Impressão da matriz de adjacência
- Impressão das listas de vizinhos
- Saídas esperadas dos códigos exemplos
- Dicas para depuração

Manipulação de Arestas e Vértices

- Adição e remoção de arestas
- Inclusão e exclusão de vértices
- Atualização de estruturas (matriz/lista)
- Cuidados na liberação de memória

Conversão entre Representações

- Matriz para lista e vice-versa
- Razões para converter entre formatos
- Exemplos práticos no repositório
- Ganhos e perdas de eficiência

Desafios na Implementação em C

- Gerenciamento manual de memória
- Cuidados com ponteiros nulos
- Prevenção de vazamentos (free)
- Erros comuns e dicas de depuração



Aplicações Reais de Grafos

- Redes sociais e conexões de pessoas
- Mapas e rotas de GPS
- Redes de computadores
- Estruturas de organização empresarial

Dicas de Boas Práticas em C

- Modularização do código
- Uso de comentários explicativos
- Nomes sugestivos para variáveis
- Testes progressivos das funções

BONS ESTUDOS