Notas de Aula - AED2 – Grafos: conceitos básicos
Prof. Jefferson T. Oliva

A partir de agora, a maior parte das aulas serão relacionadas aos grafos, cuja área de estudo tem a finalidade de estudar a conectividade entre objetos.

Antes de vermos as suas definições, vamos algumas aplicações.

Muitas aplicações necessitam considerar conexões entre pares objetos, tais como redes sociais, páginas web, empresas, organizações, mapas, etc.

As conexões podem ser representadas por grafos, que é um sistema G = (V, E) composto por um conjunto de vértices V e um conjunto de arestas E.

Vértices representam objetos e geralmente possuem valores entre 0 e n – 1, onde n é a quantidade de vértices.

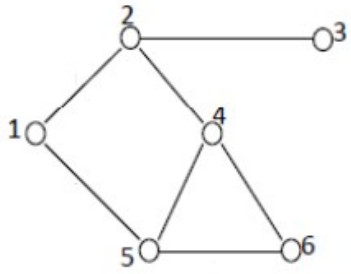
Uma aresta conecta dois vértices u e v, ou seja, é definida como par (u, v).

A partir de agora, iremos ver uma série de terminologias básicas a respeito de arestas.

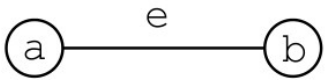
Terminologia

Exemplo de grafo

V = {1, 2, 3, 4, 5, 6}
E = {(1, 2); (1, 5); (2, 3); (2, 4); (4, 5); (4, 6); (5, 6)}



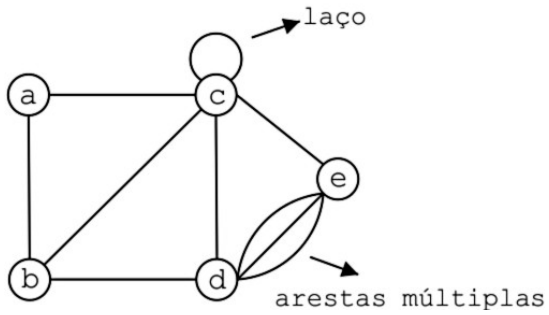
Dada uma aresta e = (a, b), dizemos que os vértices a e b são os **extremos** da aresta e se a e b são **adjacentes**



Seguindo a notação anterior, aresta e é incidente aos vértices a e b.

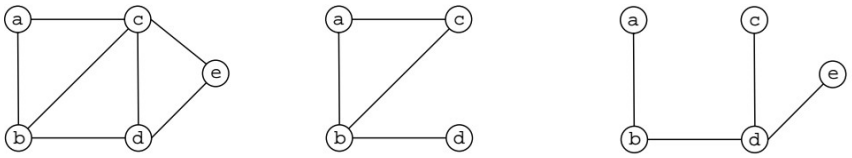
Dizemos que um grafo é **simples** quando não possui laços ou arestas múltiplas. Um laço é uma aresta cujo vértice de origem e destino são os mesmos

Um grafo pode possuir arestas múltiplas, as quais são duas ou mais arestas com o mesmo par de vértices.



Denotamos por |V| (quantidade de vértices) e |E| (quantidade de arestas) a cardinalidade dos conjuntos de vértices e arestas de um grafo G, respectivamente. Assim, o tamanho do grafo G é dado por |V| + |E|.

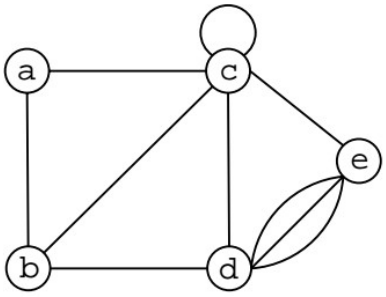
Um subgrafo H = (V', E') de um grafo G = (V, E) é um grafo tal que V' ⊆ V e E' ⊆ E. Um subgrafo gerador possui todos os vértices do grafo original, mas não os vértices



Existem 2 tipos de grafos:
- direcionados: as arestas (a, b) e (b, a) são iguais, ou seja, são consideradas a mesma (única).
- direcionado: arestas são representadas por setas. Pode conter arestas de um vértice para si mesmo (self-loop)

Adjacência: dois vértices são considerados adjacentes se há uma aresta conectando ambos, ou seja em uma aresta (a, b), a é adjacente a b (e vice-versa) para um grafo não direcionado (adjacência simétrica). Em um grafo direcionado, e (a, b) é uma aresta, então **b** é adjacente a **a,** mas **a** pode não ser adjacente a **b**.

Grau de um vértice (d(v)): determinado pela quantidade de arestas que incidem o vértice v.
- Grafo direcionado: quantidade de arestas que saem (out-degree) + quantidade de arestas que entram (in-degree)



d(a)=2

d(b)=3

d(c)=6

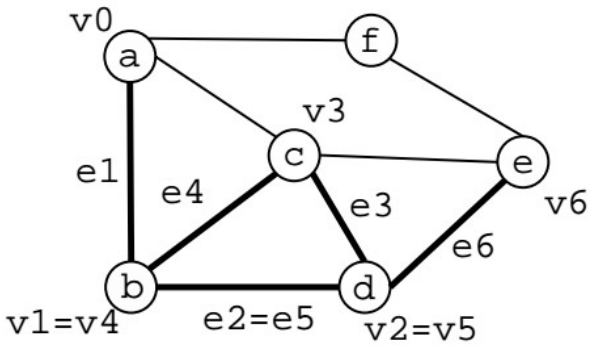
d(d)=5

d(e)=4

Vértice isolado: não há conexões por arestas

Grafo **k-regular**: todos os vértices tem grau k.

Um **passeio** P de v_0 a v_n no grafo G é uma sequência finita e não vazia (v_1, e_1, v_2, …, e_n, v_n), tal que, para todo $1 \leq i \leq n$, v_{i-1} e vi são extremos de e_i. O comprimento do passeio P é dado pelo seu número de arestas (n). Imagine um passeio simples e sem compromisso, onde não importa quantas vezes um mesmo destino é passado.



Caminho: passeio em que todos os vértices são distintos. (problema do caixeiro viajante)

Trilha: passeio em que todos as arestas são distintas. (problema chinês do correio)

Passeio simples (também denominado caminho simples): não há repetição de vértices e nem de arestas na sequência.

Um **ciclo** ou **caminho fechado** é um caminho onde v_0 = v_n

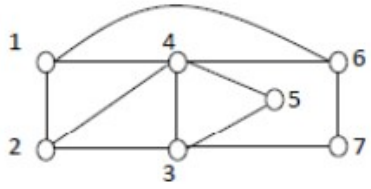
Um ciclo é dito ser simples se $v_0, \ldots, v_{n-1}$ são distintos

Um grafo que não possui ciclos é dito ser acíclico.

Caminho Hamiltoniano: caminho passa por todos os vértices.

Ciclo Hamiltoniano: ciclo que passa por todos os vértices.

Grafo Hamiltoniano: contém ciclo Hamiltoniano. O problema do caixeiro viajante pode ser solucionado ao verificar se o grafo é Hamiltoniano.



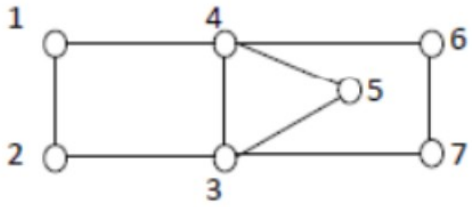
G_2

Caminho Euleriano: caminho passa por todas as arestas.

Ciclo Euleriano: ciclo que passa por todos os vértices.

Grafo Euleriano: contém ciclo Euleriano. O problema do correio chinês.

Teorema: um grafo G é euleriano e somente se todos os vértices de G têm grau par

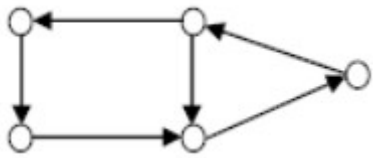


Grafo conexo: cada par de vértices está conectado por uma aresta

Grafo fortemente conexo: cada par de vértices são alcançáveis a partir de um outro (grafo não direcionado)

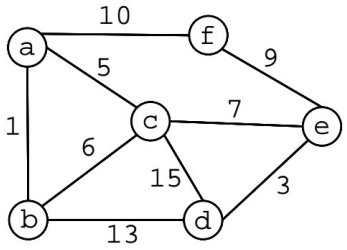
Grafo não conexo: contém vértices não acessíveis por arestas

Grafo orientado fortemente conexo: caso exista caminho orientado entre quaisquer par de vértices



Um grafo G é uma **árvore** se é conexo e acíclico. Esse tipo de grafo possui exatamente |V| - 1 vértices. (árvore de escoamento em redes). A remoção de qualquer aresta desconecta o grafo (minimal conexo). Para todo par de vértices u, v de G, existe um único caminho de u a v em G

Grafo ponderado: cada aresta está associado a um valor c(e), o qual denominamos custo (ou peso) da aresta.



Algoritmos em Grafos

Caminho mínimo: dado um conjunto de cidades, as distâncias entre elas e duas cidades A e B, determinar um caminho (trajeto) mais curto de A até B.

Árvore geradora de peso mínimo: dado um conjunto de computadores, onde cada par de computadores pode ser ligado usando uma quantidade de fibra ótica, encontrar uma rede interconectando-os que use a menor quantidade de fibra ótica possível

Emparelhamento máximo: dado um conjunto de pessoas e um conjunto de vagas para diferentes empregos, onde cada pessoa é qualificada para certos empregos e cada vaga pode ser ocupada por uma pessoa, encontrar um modo de empregar o maior número possível de pessoas

Problema do caixeiro viajante: dado um conjunto de cidades, encontrar um passeio que sai de uma cidade, passa por todas as cidades e volta para a cidade inicial tal que a distância total a ser percorrida seja menor possível.

Problema chines do correio: dado o conjunto das ruas de um bairro, encontrar um passeio que passa por todas as ruas voltando ao ponto inicial tal que a distância total a ser percorrida seja menor possível.

Representação de Grafos

A complexidade dos algoritmos para solução de problemas modelados por grafos depende fortemente da sua representação interna.

Duas abordagens são comumente utilizadas: matrizes de adjacência e lista de adjacência

O uso dessas representações depende da aplicação.

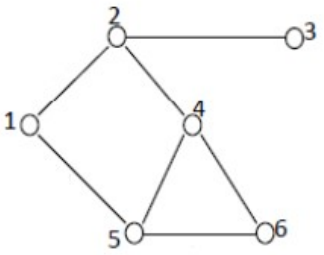
Matrizes de adjacência são adequadas para grafos densos e é útil para algoritmos que necessitam rapidez para verificar se há aresta ligando dois vetores.

Listas de adjacência são apropriadas em grafos esparsos, em que a quantidade de arestas é muito menor que a quantidade de vértices elevada ao quadrado. Esse tipo de representação é compacta e é utilizada na maioria das aplicações.

Matriz de adjacência

É uma matriz A |V| x |V|, onde, se a[i, j] = 1, então há uma aresta que liga os vértices i e j.
Caso a[i, j] = 0, então não há uma aresta que que torna os vértices i e j adjacentes.

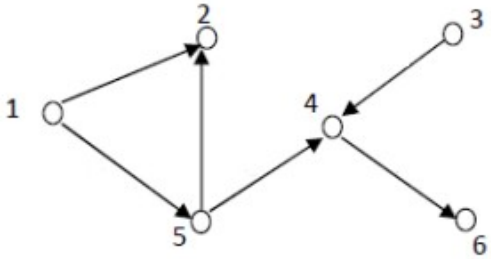
Esse tipo de matriz é simétrico para grafos não orientados, ou seja, a[i, j] = a[j, i]



G_1

$$A_{G_1} = \begin{bmatrix} 0 & 1 & 0 & 0 & 1 & 0 \\ 1 & 0 & 1 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 1 & 1 \\ 1 & 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 1 & 1 & 0 \end{bmatrix}$$

Em um grafo direcionado, a[i, j] significa que "j → i"



G_2

$$A_{G_2} = \begin{bmatrix} 0 & 1 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

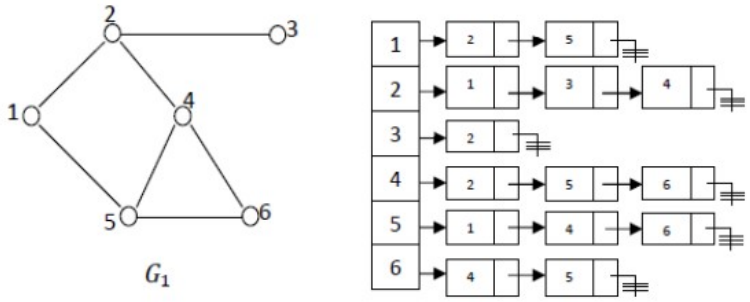
Em um grafo ponderado, em vez de 0's e 1's, a[i,j] contém o peso (distância) da aresta (i, j)

Lista de adjacência

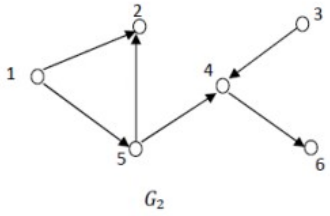
Em uma lista de adjacência, para cada vértice v há uma lista encadeada (Adj[v]) dos vértices adjacentes a v.

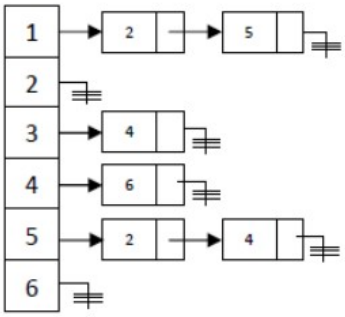
O vértice u aparece em Adj[v] se há aresta (v; u) no grafo G.

Os vértices podem estar em qualquer ordem em uma lista de adjacência.



G_1





G_2

Matriz × Lista de adjacência

Matriz de adjacência
- Fácil verificar se (i, j) é uma aresta de G
- Complexidade de espaço: $\Theta(|V|^2)$
- Adequada para grafos densos ($|E| = \Theta(|V|^2)$)

Lista de adjacência
- Fácil descobrir os vértices adjacentes a um dado vértice v (ou seja, listar Adj[v])
- Complexidade de espaço: $\Theta(|V| + |E|)$
- Adequada para grafos esparsos ($|E| = \Theta(|V|)$)

Para determinados problemas é essencial ter estruturas de dados adicionais para melhorar a eficiência dos algoritmos

Referências

Cormen, T. H., Leiserson, C. E., Rivest, R. L., Stein, C. Introduction to Algorithms. Third edition, The MIT Press, 2009.

Tenenbaum, A.; Langsam, Y. Estruturas de Dados usando C. Pearson, 1995.

Ziviani, N. Projeto de Algoritmos - com implementações em Java e C++. Thomson, 2007.