# Projeto e Análise de Algoritmos

## Algoritmos elementares de grafos

Diane Castonguay
diane@inf.ufg.br

Instituto de Informática
Universidade Federal de Goiás

# Grafo orientado

Um ***grafo orientado*** G é um par (V, E)

onde

V é um conjunto finito, o conjunto das vértices

E é uma relação binária, o conjunto das arestas

(i.e. $E \subseteq V \times V$)

*Observação: Pode ter laço i.e. aresta de um vértice para ele mesmo, mas não pode ter duas flechas de uma vértice para outra.*

# Grafo não orientado

Em um ***grafo não orientado*** G=(V, E)

o conjunto das arestas E consiste em pares não ordenados, em lugar de pares ordenados

*Observação: Consideramos as arestas (u, v) e (v, u) como a mesma.*

# Incidência

Seja (u,v) uma aresta de G i.e. (u,v) $\in$ E.

Se G=(V, E) é um grafo orientado
dizemos que (u,v) é ***incidente do*** ou ***sai do*** vértice u
e é ***incidente no*** ou ***entra no*** vértice u.

Se G=(V, E) é um grafo não orientado
dizemos que (u,v) é ***incidentes nos*** vértices u e c.

# Adjacência

Seja (u,v) uma aresta de G i.e. $(u,v) \in E$.

Dizemos que o vértice v é adjacente ao vértice u.

Quando G é um grafo não orientado, a relação de adjacência é simétrica.

# Grau num grafo orientado

o ***grau de saída*** de um vértice é
o número de arestas que saem dele.

o ***grau de entrada*** de um vértice é
o número de arestas que entram dele.

O ***grau*** de um vértice é
a soma de seus graus de entrada e saída.

# Grau num grafo não orientado

O ***grau*** de um vértice é
o número de arestas incidentes nele.

# Caminho de comprimento k

Um ***caminho de comprimento k*** de um vértice u até um vértice v em um grafo G=(V, E) é uma seqüência $< u_0 , u_1 , u_2 , \ldots , u_{k-1} , u_k >$ de vértices tais que $u = u_0$ , $u_k = v$ e $(u_i , u_{i+1}) \in E$.

$$u = u_0 \rightarrow u_1 \rightarrow u_2 \rightarrow \ldots \rightarrow u_{k-1} \rightarrow u_k = v$$

Obs.: O commprimento do caminho é o número de arestas nele.

# Vértice Acessível, caminho simples e ciclo

Se existir um caminho p de u até v,
dizemos que v é ***acessível*** a partir de u via p.

Um caminho é ***simples*** se
todos os vértices do caminho são distintos.

Um ***ciclo*** é um caminho de u a u.

# Conexo

Um grafo não orientado é ***conexo*** se
todo par de vértices são acessíveis.

Os ***componentes conexos*** de um grafo
são as classes de quivalência de vértices sob
a relação "é acessível a partir de".

Um grafo orientado é ***conexo*** se tem exactamente
um componente conexo.

# Representação de listas de adjacências de um grafo

A ***Representação de listas de adjacências***
de um grafo G = (V, E)
consiste em um vetor Adj de |V| listas,
uma para cada vértice em V.

Para cada u ∈ V, a lista de adjacência Adj[u]
consite em todos os vértices adjacentes de u
i.e os vértices v tais que existe uma aresta (u,v) ∈ E.

# Representação de matriz de adjacências de um grafo

A ***Representação de matriz de adjacências***
de um grafo G = (V, E)
consiste em uma matriz |V| x |V|
$A=(a_{ij})$ tal que
$a_{ij} = 1$ se $(i, j) \in E$ e $a_{ij} = 0$ senão

*Observação: Supomos que os vértices são numerados 1, 2, ..., |V|.*

# Aplicação

Grafos ponderados
(arestas com peso)

Grafo transposta

# Busca em largura – 1/2

BFS(G, s)

1. para cada vértice $u \in V[G] - \{s\}$ faça
2. $cor[u] \leftarrow$ BRANCO
3. $d[u] \leftarrow \infty$
4. $[u] \leftarrow$ NIL
5. fim-para
6. $cor[s] \leftarrow$ CINZA
7. $d[s] \leftarrow 0$
8. $[s] \leftarrow$ NIL
9. $Q \leftarrow 0$
10. ENQUEUE($Q$, s)

# Busca em largura – 2/2

```
11. enquanto Q ≠ 0 faça
12.     u ← DEQUEUE(Q)
13.     para cada vértice v ∈ Adj[u] faça
14.             se cor[v] = BRANCO
15.                 então cor[v] ← CINZA
16.                         d[v] ← d[u] + 1
17.                          [v] ← u
18.                         ENQUEUE(Q, v)
19.             fim-se
20.     fim-para
21.     cor[u] ← PRETO
22. fim-enquanto
```