

Universidade Federal da Paraíba

Centro de Informática

Departamento de Informática

Estrutura de Dados

Grafos

Tiago Maritan

tiago@ci.ufpb.br

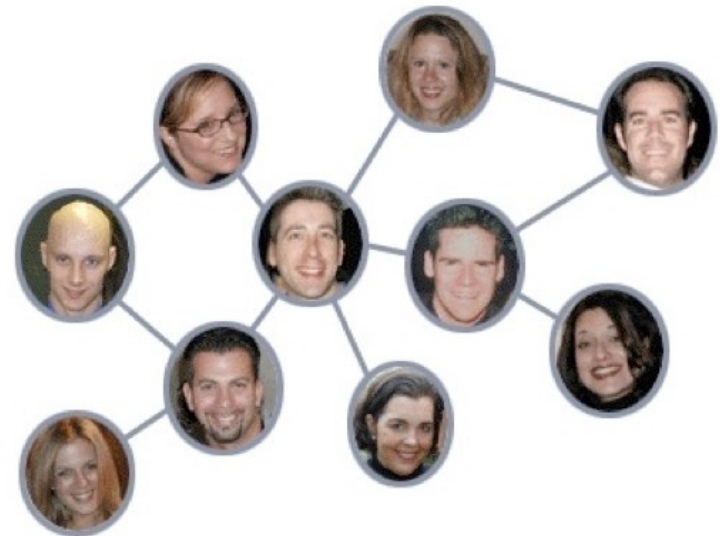
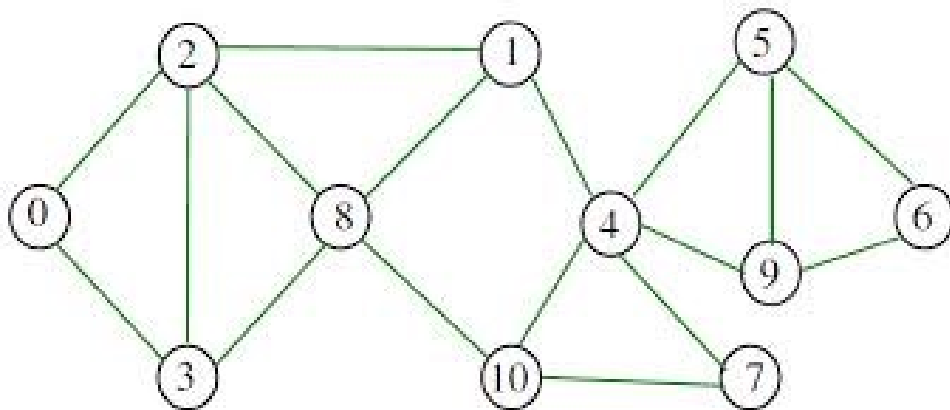
Modificado por Gilberto Farias

Grafos

Um grafo é um par (V, A) , onde

V é um conjunto de nós, chamados de **vértices**

A é uma coleção de pares de vértices, chamados de **arestas**

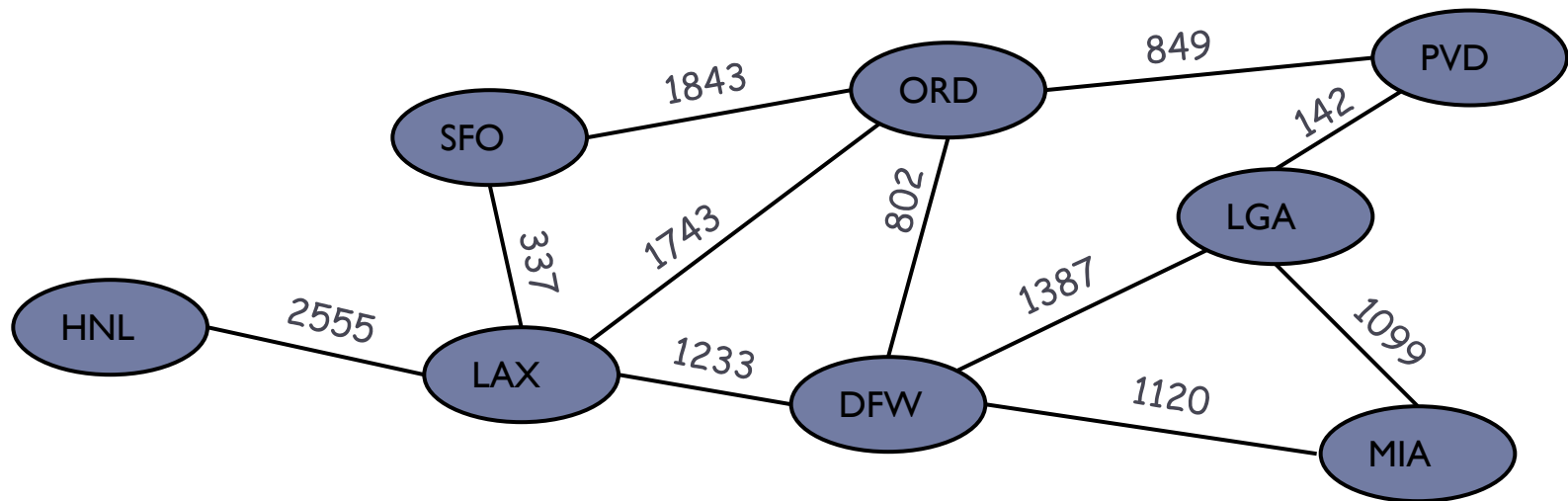


Grafos

Exemplo: Rede de Aeroportos

Um vértice representa um aeroporto e armazena o código do aeroporto composto de três letras

Uma aresta representa uma rota de voo entre dois aeroportos e armazena a milhagem da rota



Aplicações

Circuitos Eletrônicos

Redes de transporte

Rede de rodovias

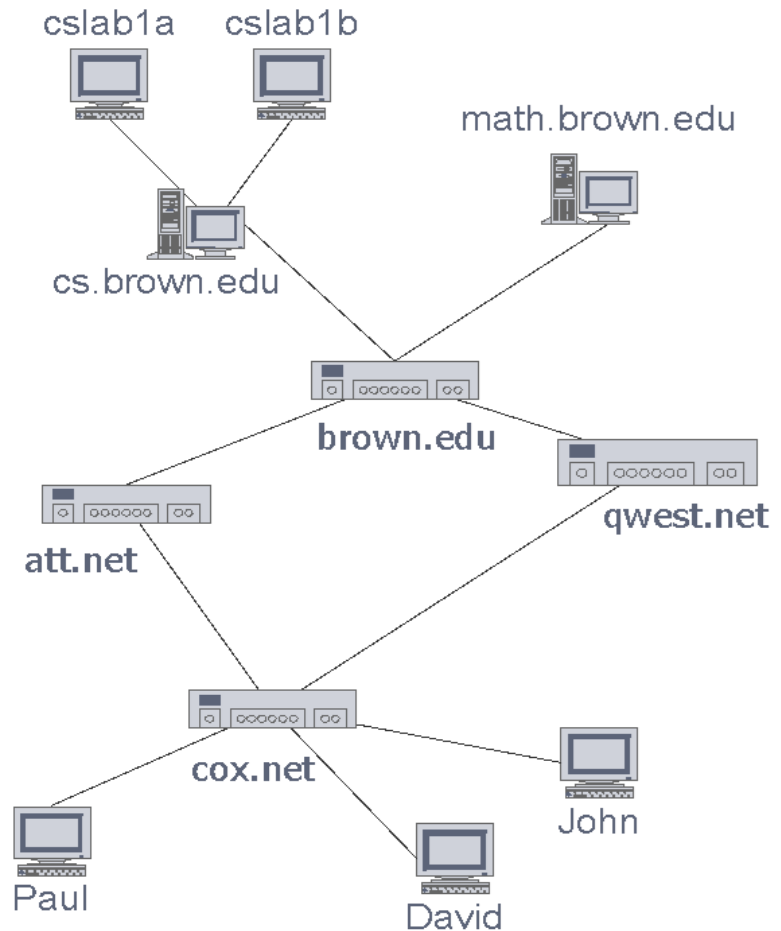
Rede de vôos

Redes de computadores

Rede local

Internet

Redes Sociais



Tipos de arestas

- ❑ **Aresta dirigida**

Par ordenado de vértices (u,v)

1º vértice **u** é a origem

2º vértice **v** é o destino

Ex. Um vôo

- ❑ **Aresta não-dirigida**

Pares não-ordenados de vértices (u,v)

Ex. Distância entre aeroportos

- ❑ **Grafo dirigido**

Todas as arestas são dirigidas

Ex. Rede de vôos

- ❑ **Grafo não-dirigido**

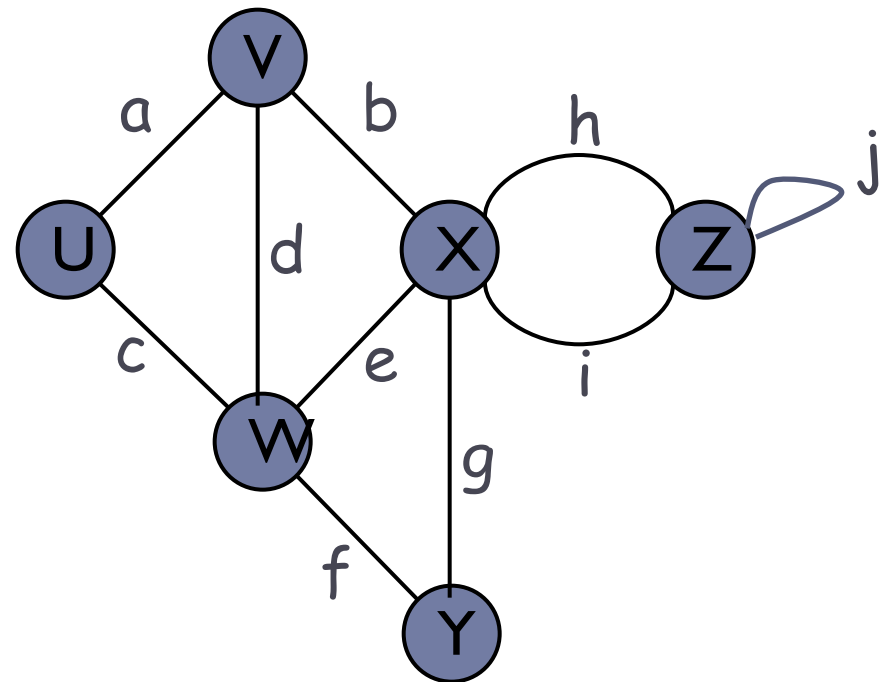
Todas as arestas são não-dirigidas

Ex. Rede de distância entre aeroportos



Terminologia

- ❑ **Vértices-fim de uma aresta**
U e V são os pontos-finais da aresta **a**
- ❑ **Arestas incidentes em um vértice**
a, d, e b são incidentes em V
- ❑ **Vértices Adjacentes**
U e V são adjacentes
- ❑ **Grau de um vértice**
Nº de arestas do vértice
Ex: X tem grau 5
- ❑ **Arestas paralelas**
h e i são arestas paralelas
- ❑ **Auto-loop**
j é um auto-loop



Terminologia (cont.)

❑ Caminho

Sequência de vértices e arestas

Começa com um vértice

Finaliza com um vértice

Cada aresta é precedida e seguida por seus pontos-finais

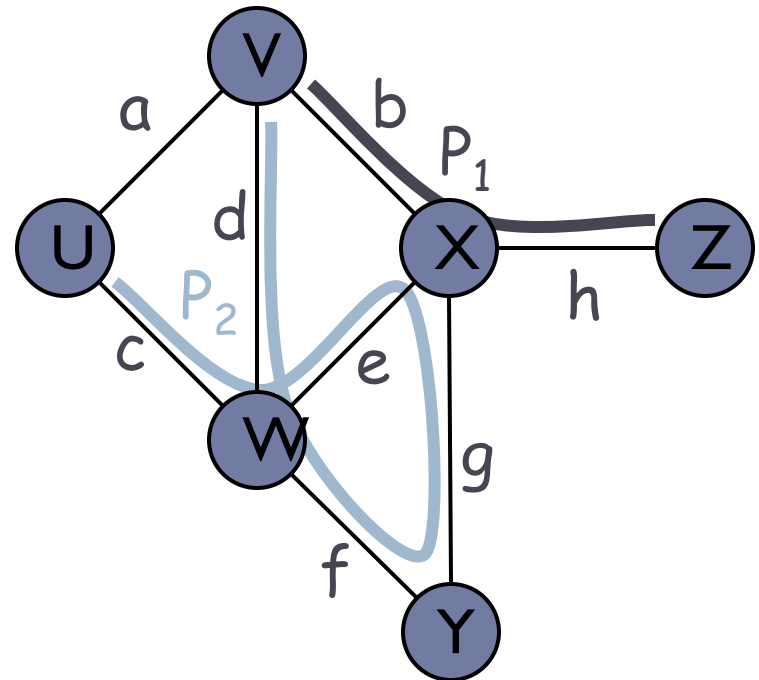
❑ Caminho Simples

Caminho de tal forma que todos os seus vértices e arestas são distintos

❑ Exemplos

$P_1 = (V, b, X, h, Z)$ é um caminho simples

$P_2 = (U, c, W, e, X, g, Y, f, W, d, V)$ é um caminho que não é simples



Terminologia (cont.)

❑ Ciclo

Sequência circular de vértices e arestas

Cada aresta é precedida e seguida pelos seus pontos-finais

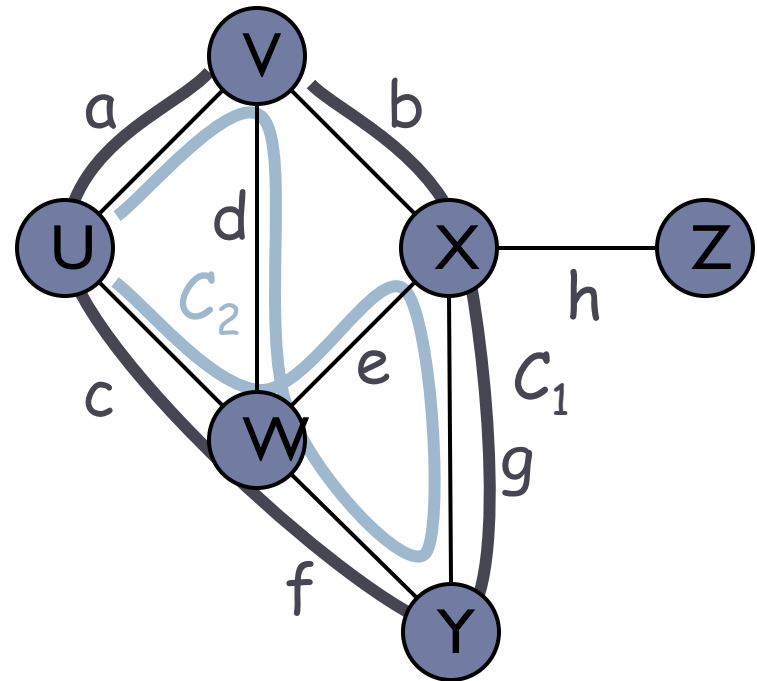
❑ Ciclo simples

Um ciclo de forma que todos os seus vértices e arestas sejam diferentes

❑ Exemplos

$C_1 = (V, b, X, g, Y, f, W, c, U, a, \hookleftarrow)$ é um ciclo simples

$C_2 = (U, c, W, e, X, g, Y, f, W, d, V, a, \hookleftarrow)$ é um ciclo que não é simples



Implementação de Grafos

Como implementar grafos?

Lista de adjacências

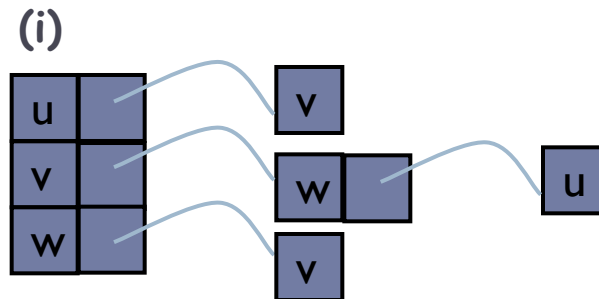
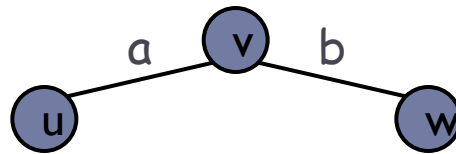
Matriz de adjacências



Lista de Adjacências

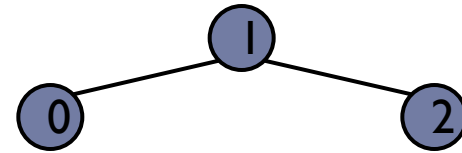
Array de listas encadeadas

Para descobrir se existe aresta (i,j) percorremos a lista do nó i até encontrarmos (ou não) j



Matriz de Adjacências

Matriz de tamanho $N \times N$, onde N é o número de vértices



A célula (i,j) indica se existe aresta entre i e j .

Valor 0 indica aresta inexistente.

	0	1	2
0	0	1	0
1	1	0	1
2	0	1	0

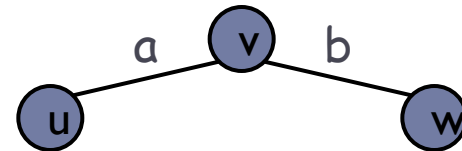


Matriz de Adjacências

Estrutura Vértice

Chave inteira (índice) associado com o vértice

Informação armazenada



Estrutura Aresta

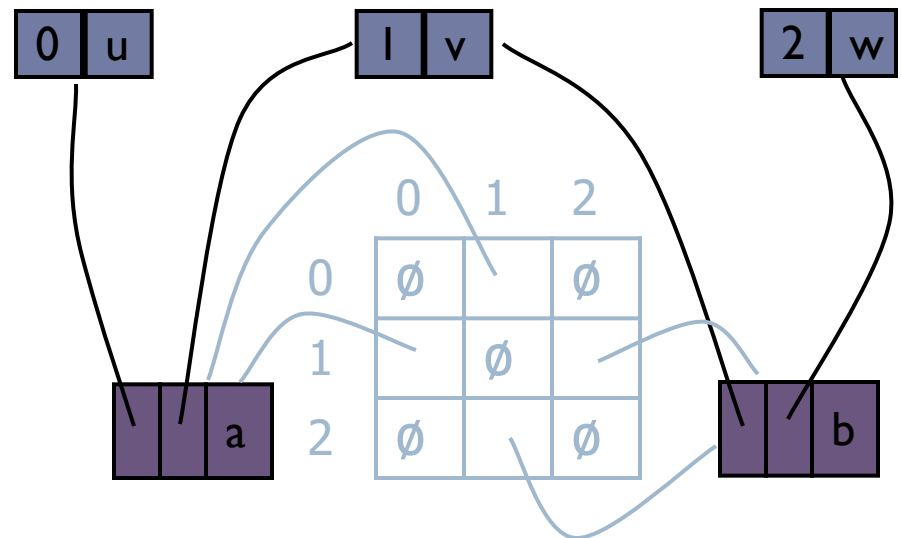
Informação armazenada

Ponteiros para os vértices

Matriz

Ponteiro para a aresta de vértices adjacentes

Null caso não haja aresta



Busca em Grafos

Operação mais comum em Grafos: visita sistemática a seus nós (uma única vez!)

Dois tipos básicos de busca:

- Busca em largura/extensão

- Busca em profundidade



Busca em Largura

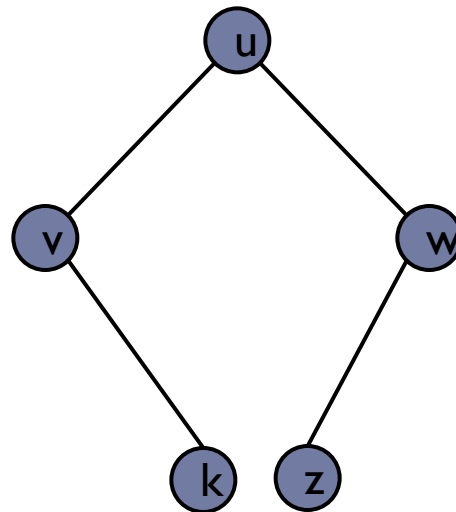
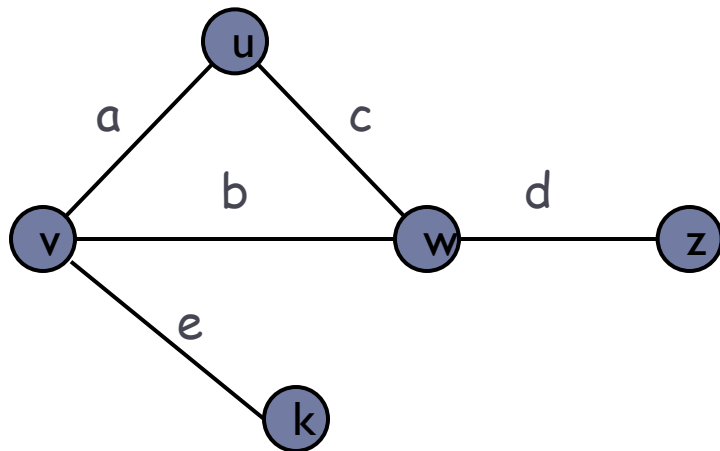
BFS (G, s)

```
1 for cada vértice de  $u \in V[G] - \{s\}$ 
2   do  $cor[u] \leftarrow BRANCO$ 
3      $d[u] \leftarrow \infty$ 
4      $\pi[u] \leftarrow NULL$ 
5  $cor[s] \leftarrow CINZA$ 
6  $d[s] \leftarrow 0$ 
7  $\pi[s] \leftarrow NULL$ 
8  $Q \leftarrow \emptyset$ 
9 ENQUEUE( $Q, s$ )
10 while  $Q \neq \emptyset$ 
11   do  $u \leftarrow DEQUEUE(Q)$ 
12     for cada vértice de  $v \in adj[u]$ 
13       do if  $cor[v] = BRANCO$ 
14         then  $cor[v] \leftarrow CINZA$ 
15            $d[v] \leftarrow d[u] + 1$ 
16            $\pi[v] \leftarrow u$ 
17           ENQUEUE( $Q, v$ )
18    $cor[u] \leftarrow PRETO$ 
```



Busca em Largura

Para cada nó, o mesmo é processado e colocamos seus adjacentes em uma FILA

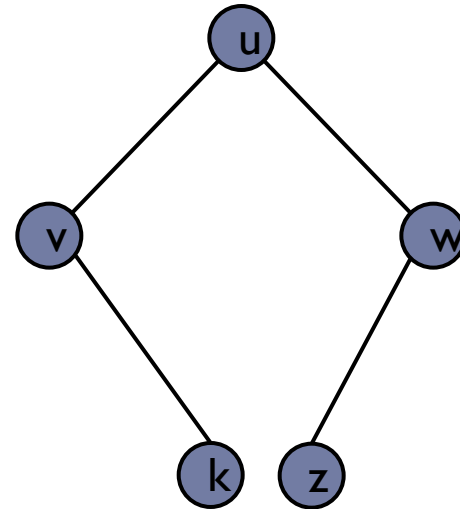
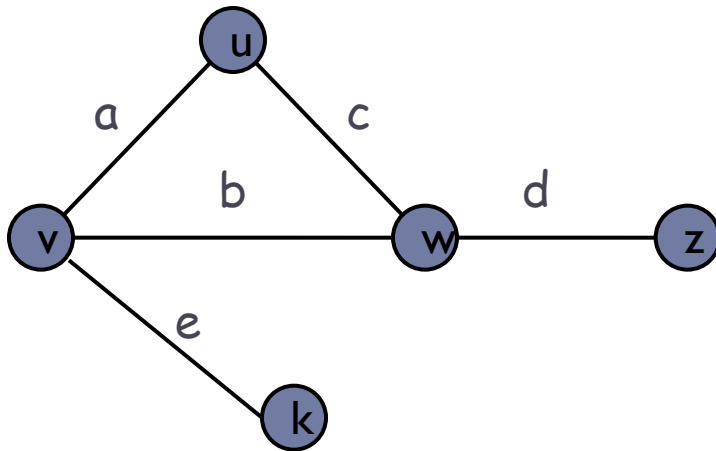


u, v, w, k, z



Busca em Profundidade

Semelhante a busca em “pré-ordem” em uma árvore



u, v, k, w,
z



Busca em Profundidade

DFS (G)

```
1 for cada vértice de  $u \in V[G]$ 
2   do  $cor[u] \leftarrow BRANCO$ 
3      $\pi[u] \leftarrow NULL$ 
4 for cada vértice de  $u \in V[G]$ 
5   do if  $cor[u] = BRANCO$ 
6     then  $DFS - VISIT(u)$ 
```

DFS - VISIT (u)

```
1  $cor[u] \leftarrow CINZA$ 
2 for cada vértice  $v \in adj[u]$ 
3   do if  $cor[v] = BRANCO$ 
4     then  $\pi[v] \leftarrow u$ 
5      $DFS - VISIT(v)$ 
6  $cor[u] \leftarrow PRETO$ 
```



Universidade Federal da Paraíba

Centro de Informática

Departamento de Informática

Estrutura de Dados

Grafos

Tiago Maritan

tiago@ci.ufpb.br

