

INSTITUTO MAUÁ DE TECNOLOGIA



16

Grafos

Busca em Grafos



Buscas em Grafos/Dígrafos

Buscas em Grafos / Dígrafos

Explorar um grafo;

Gera uma árvore de buscas;

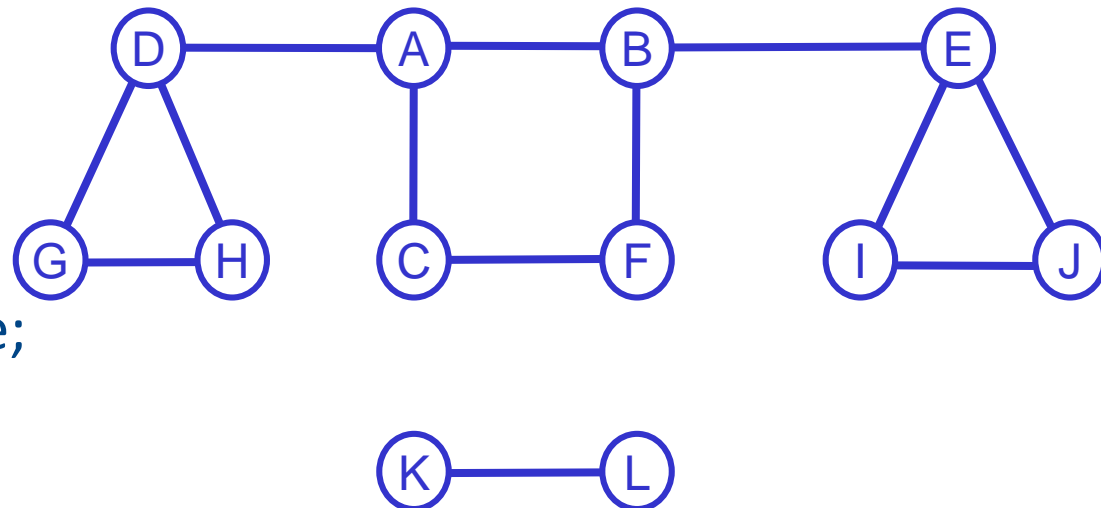
Localizar uma situação (vértice) desejada;

Base para a resolução de vários problemas modelados com grafos;

Tipo de percurso:

Busca em profundidade;

Busca em largura.



Buscas em Profundida (DFS)

Depth First Search (DFS);

Permite determinar quais partes de um grafo são alcançáveis a partir de um vértice;

A partir do vértice de partida, explorar um dos vértices adjacentes **não visitados**. Essa repetição ocorre até que não haja mais vértices não visitados, obrigando o algoritmo a **retornar** o caminho e explorar um outro vértice não visitado.

Cada vértice visitado é inserido em uma **pilha**. Quando o **retorno** for necessário, a pilha é desfeita.

Para automatizar a pilha, utilizaremos a abordagem **recursiva**.

Busca em Profundidade

```
Procedimento DFS (Grafo g, int origem, int destino,  
                    ListaDeVertices *caminho) {
```

```
    achei ← falso // se for procurar algo
```

```
    caminho->nVertices ← 0
```

```
    Para cada vértice v de g faça
```

```
        visitado[v] ← falso
```

Procura a partir da origem

```
    Explorar(g, origem, destino, visitado, caminho, &achei);
```

```
    Para cada vértice v de g faça
```

**O laço garante que todos os vértices
sejam visitados, mesmo se o grafo
for desconexo**

```
        Se (!visitado[v] && !achei) então
```

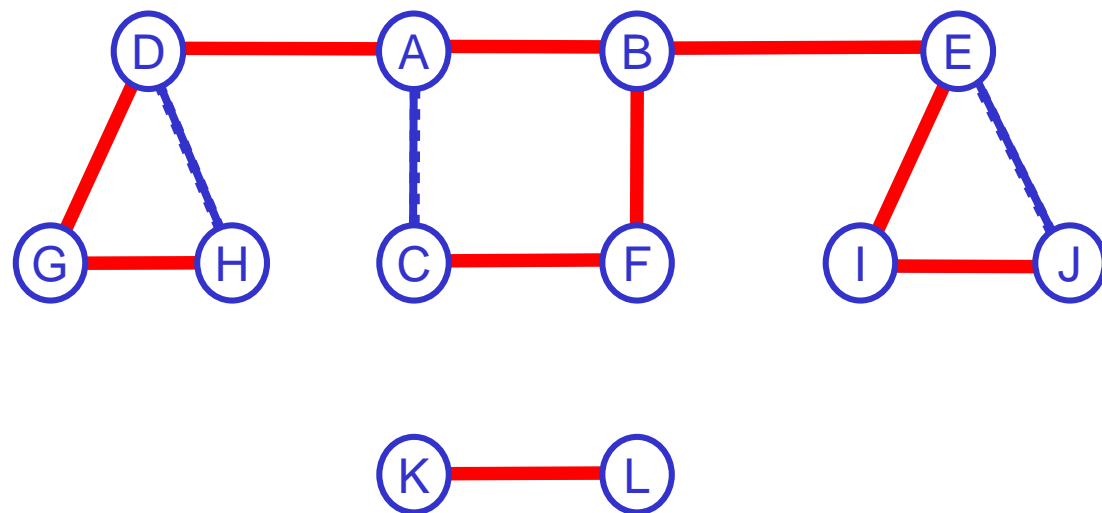
```
            Explorar(g, v, destino, visitado, caminho, &achei)
```

```
}
```

Busca em Profundidade

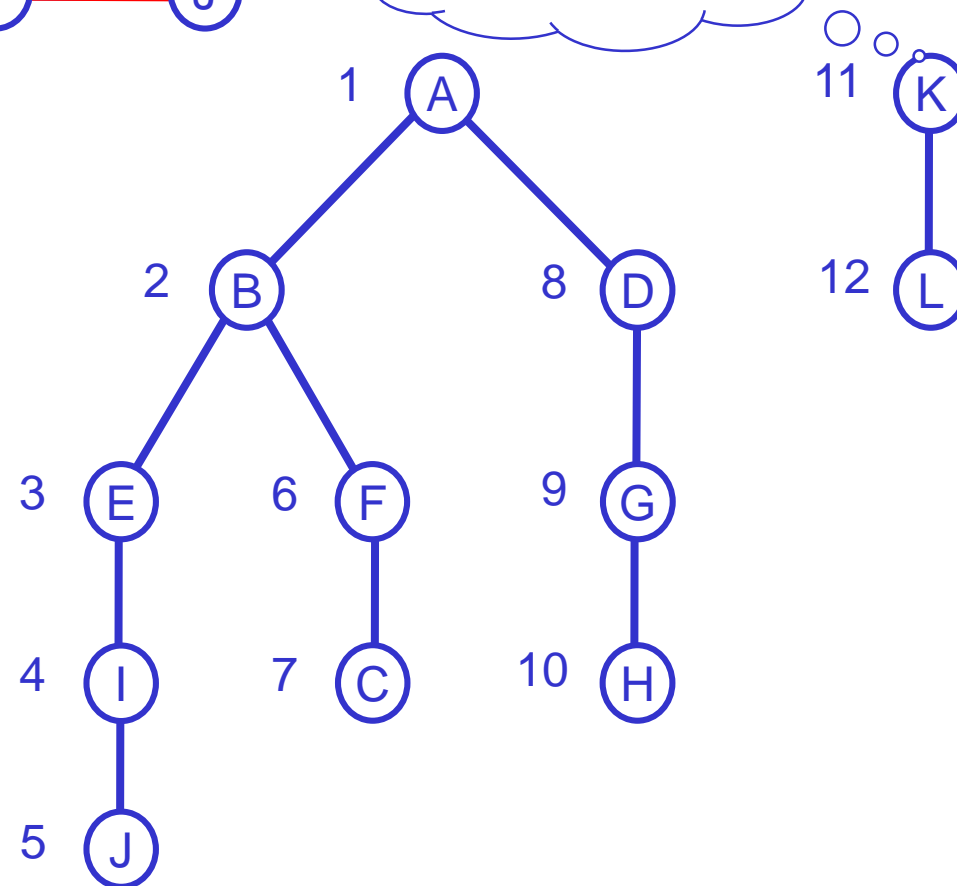
```
Procedimento Explorar (Grafo g, int v, int destino,  
int *visitado, ListaDeVertices *caminho, int *achei) {  
    visitado[v] ← verdadeiro  
  
    Os procedimentos Pre_visita e  
    Pos_visita contém os algoritmos específicos  
    para a resolução de algum problema  
  
    Pre_visita_DFS (parâmetros) //opcional  
  
    Para cada vértice u de g faça  
        Se (!visitado[u])  
            && (u adjacente v)  
                && (!(*achei)) então  
                    Explorar(g, u, destino, visitado, caminho, achei)  
  
    Pos_visita_DFS (parâmetros) // opcional  
}
```

Busca em Profundidade



Nesse exemplo, a escolha dos vértices segue a ordem alfabética

Lembra do laço do procedimento DFS?

J
⊕
G
B
K

Pilha

Buscas em Largura (BFS)

Breadth First Search (BFS);

Normalmente utilizada para determinar caminhos mais curtos entre vértices;

A partir do vértice de partida, expandir todos os seus vértices adjacentes em um **novo nível**. Nesse novo nível, repetir a operação para todos os vértices que possuam adjacentes não visitados.

Cada vértice visitado insere os vértices adjacentes a ele em uma **fila**. Quando todos os adjacentes a ele forem inseridos, o vértice analisado é retirado da **fila**.

Se o grafo for desconexo, alguns vértices não serão visitados.

Busca em Largura

```
procedimento BFS (Grafo g, int origem, int destino,  
                  float *dist, ListaDeVertices *caminho) {
```

```
    CriarFila(&fila)    Temos que incluir a biblioteca "fila.h"
```

```
    achei ← falso
```

```
    caminho->nVertices ← 0
```

```
    Para cada vértice v de g faça
```

```
        dist[v] ←  $\infty$ 
```

```
    Pre_visita_BFD(parâmetros) //opcional
```

```
    dist[origem] ← 0
```

```
    dados.valor ← origem
```

```
    InserirNaFila(&fila, dados)
```

```
    Pos_visita_BFD(parâmetros) //opcional
```

} Procura a partir
da origem

Busca em Largura

Enquanto (!FilaVazia(fila) && (!achei)) **faça**

Pre_visita_BFD(parâmetros) //opcional

RemoverDaFila(&fila, &dados)

$v \leftarrow \text{dados.valor}$

Para cada vértice u de g **faça**

Se (dist[u] = ∞)

&& (u adjacente v)

&& (!achei) **então**

$\text{dados.valor} \leftarrow u$

InserirNaFila(&fila, dados)

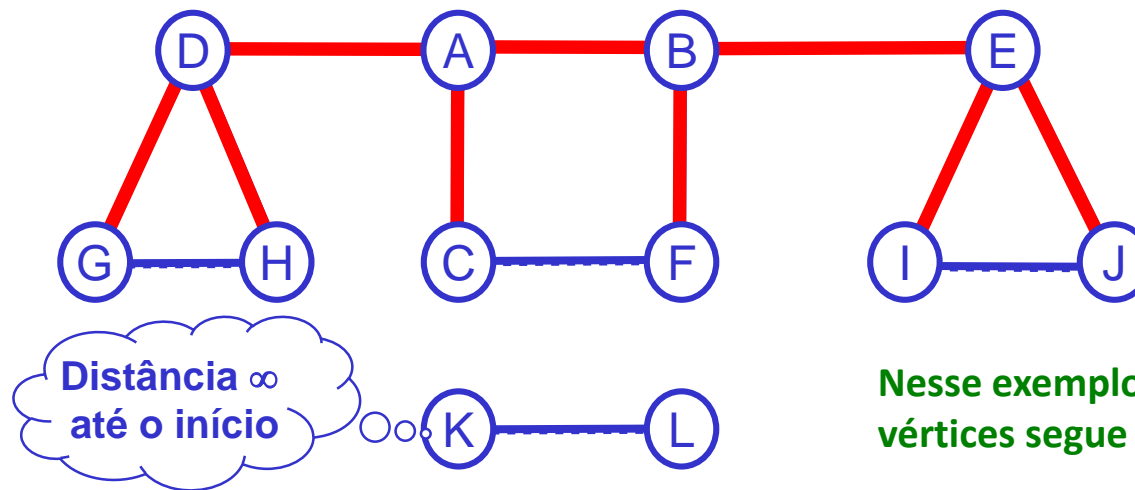
Pos_visita_BFD(parâmetros) //opcional

dist[u] \leftarrow dist[v] + 1 // ou

// dist[u] \leftarrow dist[v] + PesoDaAresta(g, v, u)

}

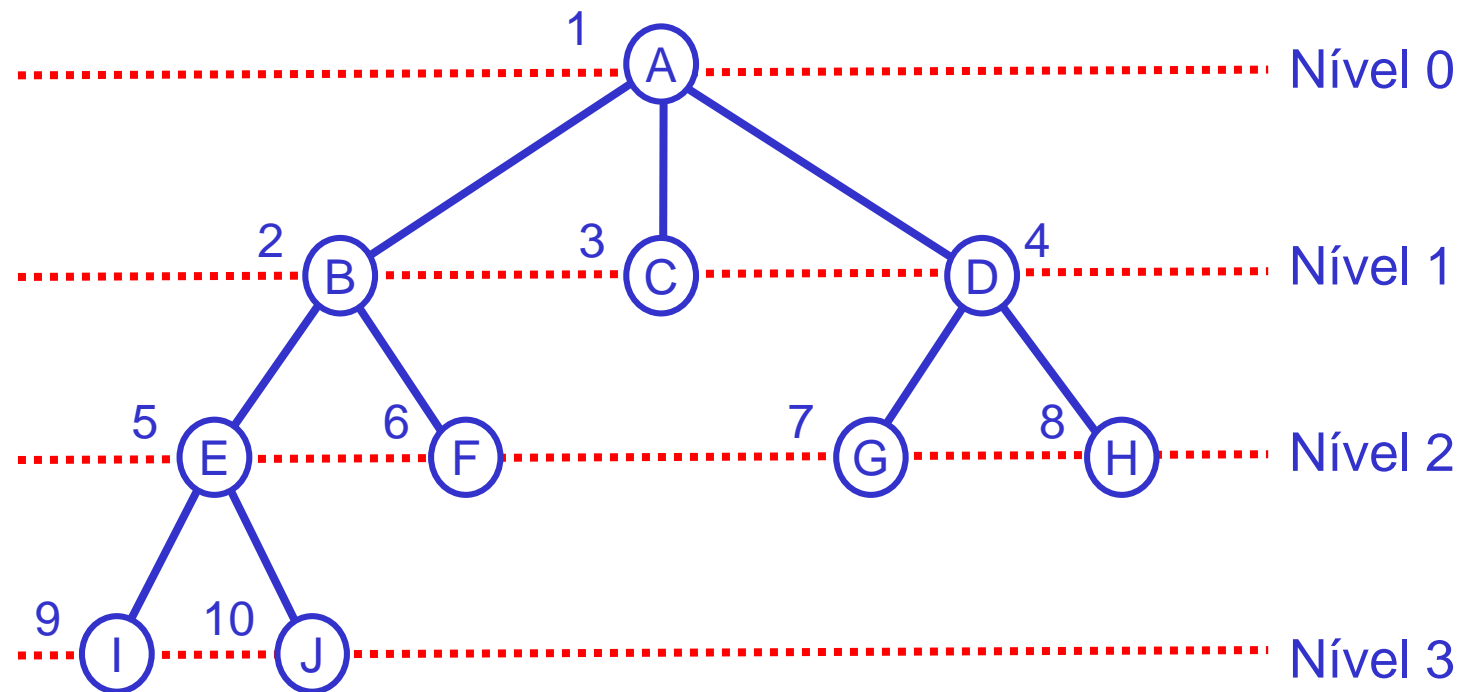
Busca em Largura



Nesse exemplo, a escolha dos vértices segue a ordem alfabética

J
I
H
G
F
E
D
C
B
A

Fila



Exemplo: armazenando os vértices descobertos

```
void Pre_visita_DFS(Grafo g, int v,  
                   ListaDeVertices *caminho)  
  
void Pos_visita_BFD(Grafo g, int v,  
                   ListaDeVertices *caminho)  
  
{  
    caminho->vertices[caminho->nVertices] ← v;  
    caminho->nVertices++;  
}
```

Mesmo código para os dois!!