Algoritmos e Estruturas de Dados II

Busca em Grafos

Prof. Tiago Eugenio de Melo
tmelo@uea.edu.br

www.tiagodemelo.info

Observações

- As palavras com a fonte `Courier` indicam as palavras-reservadas da linguagem de programação.

Referências

- **Projetos de Algoritmos – com implementações em Pascal e C**. Nivio Ziviani. 2ª edição. Thomson, 2005.
- Material baseado nas notas de aula do professor Marco Antônio Moreira de Carvalho. Acessado em 14/10/2019: http://www.decom.ufop.br/marco/ensino/bcc204/material-das-aulas

Introdução
Busca em Grafos

Busca em Grafos

Busca em Grafos

- Definição

Busca em Grafos

- Definição
  - A busca em grafos (ou percurso em grafos) é a examinação de vértices e arestas de um grafo.

Busca em Grafos

- Definição
  - A busca em grafos (ou percurso em grafos) é a examinação de vértices e arestas de um grafo.
  - O projeto de bons algoritmos para determinação de estruturas ou propriedades de grafos depende fortemente do domínio dessas técnicas.

Busca em Grafos

Busca em Grafos

- Terminologia

Busca em Grafos

- Terminologia
  - Uma aresta ou um vértice ainda não examinados são marcados como não explorados ou não visitados.

Busca em Grafos

- Terminologia
  - Uma aresta ou um vértice ainda não examinados são marcados como não explorados ou não visitados.
  - Inicialmente, todos os vértices e arestas são marcados como não explorados.

Busca em Grafos

- Terminologia
  - Uma aresta ou um vértice ainda não examinados são marcados como não explorados ou não visitados.
  - Inicialmente, todos os vértices e arestas são marcados como não explorados.
  - Após terem sido examinados, os mesmos são marcados como explorados ou visitados.

Busca em Grafos

- Terminologia
  - Uma aresta ou um vértice ainda não examinados são marcados como não explorados ou não visitados.
  - Inicialmente, todos os vértices e arestas são marcados como não explorados.
  - Após terem sido examinados, os mesmos são marcados como explorados ou visitados.
  - Ao final, todos os vértices e arestas são marcados como explorados (no caso de uma busca completa).

Busca Genérica

```
   Entrada: Grafo G=(V, A)
1  enquanto existir j ∈ V com uma aresta {j,k} não visitada faça
2  |   Escolha o vértice j e visite a aresta {j, k};
3  |   se j não é marcado então
4  |   |   marque j como visitado;
5  |   fim
6  |   se k não é marcado então
7  |   |   marque k como visitado;
8  |   fim
9  fim
```

Atravessando Labirintos

Como escapar?

Como escapar?

- Algoritmo de Trémaux (Século XIX)

Como escapar?

- Algoritmo de Trémaux (Século XIX)
  - Pierre Trémaux

Como escapar?

- Algoritmo de Trémaux (Século XIX)
  - Pierre Trémaux
  - Arquiteto francês, fotógrafo e orientalista.

Como escapar?

- Algoritmo de Trémaux (Século XIX)
  - Pierre Trémaux
  - Arquiteto francês, fotógrafo e orientalista.
  - Autor de várias publicações científicas e etnografias.

Como escapar?

Como escapar?

- Algoritmo

Como escapar?

- Algoritmo
  - Inicialmente, uma direção aleatória é escolhida.

Como escapar?

- Algoritmo
  - Inicialmente, uma direção aleatória é escolhida.
  - Ao chegar em uma junção não visitada (ou seja, nenhuma linha), escolha a direção aleatória e risque o caminho.

Como escapar?

- Algoritmo
  - Inicialmente, uma direção aleatória é escolhida.
  - Ao chegar em uma junção não visitada (ou seja, nenhuma linha), escolha a direção aleatória e risque o caminho.
  - Ao chegar em uma junção já marcada, vire-se e caminhe de volta, marcando o caminho pela segunda vez.

Como escapar?

- Algoritmo
  - Inicialmente, uma direção aleatória é escolhida.
  - Ao chegar em uma junção não visitada (ou seja, nenhuma linha), escolha a direção aleatória e risque o caminho.
  - Ao chegar em uma junção já marcada, vire-se e caminhe de volta, marcando o caminho pela segunda vez.
  - Se este não for o caso, escolha o caminho com menos linhas e marque-o novamente.

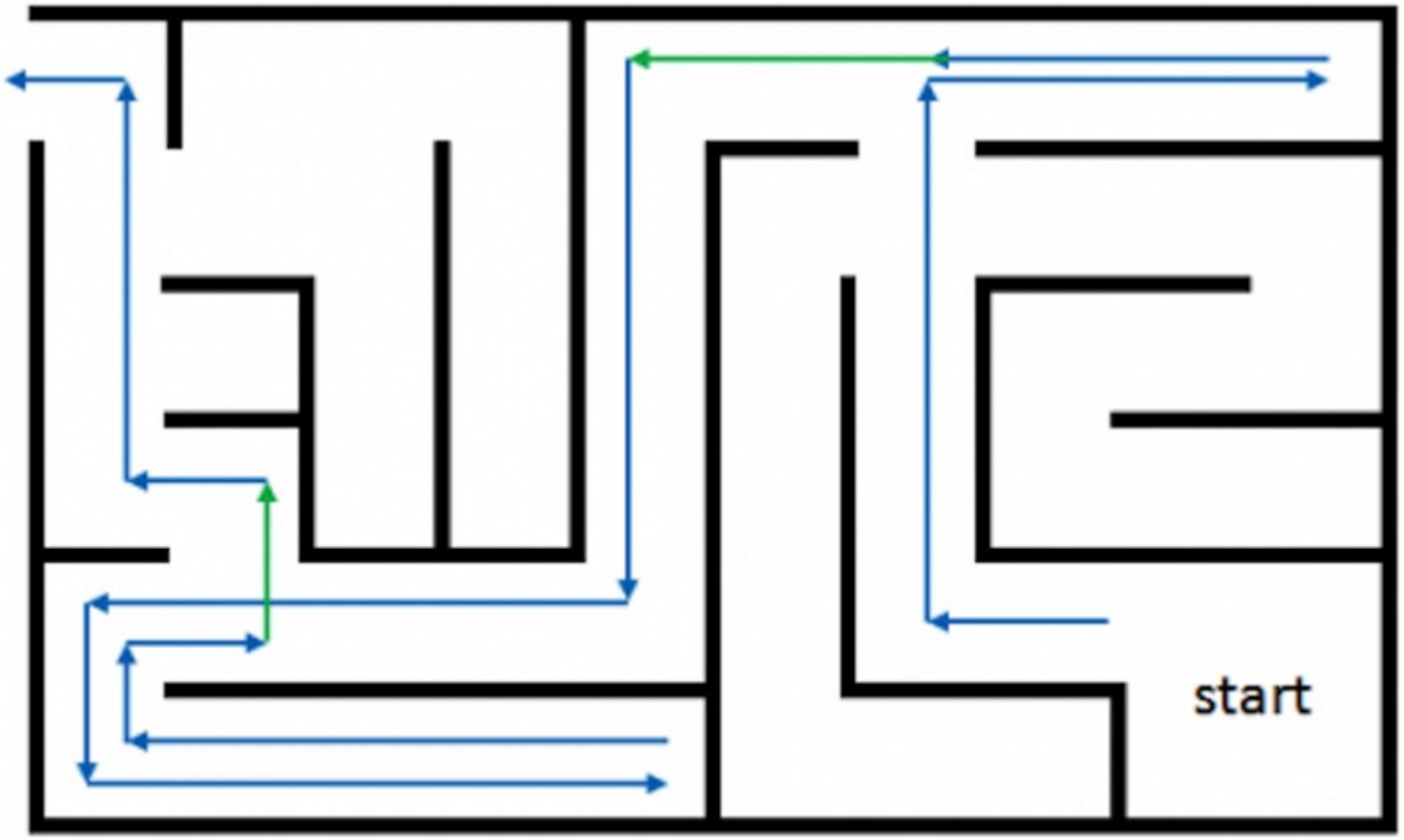
Como escapar?

- Algoritmo
  - Inicialmente, uma direção aleatória é escolhida.
  - Ao chegar em uma junção não visitada (ou seja, nenhuma linha), escolha a direção aleatória e risque o caminho.
  - Ao chegar em uma junção já marcada, vire-se e caminhe de volta, marcando o caminho pela segunda vez.
  - Se este não for o caso, escolha o caminho com menos linhas e marque-o novamente.
  - Quando finalmente chegar à saída do labirinto, os caminhos marcados com apenas uma linha indicarão o caminho direto até o ponto inicial.

Como escapar?

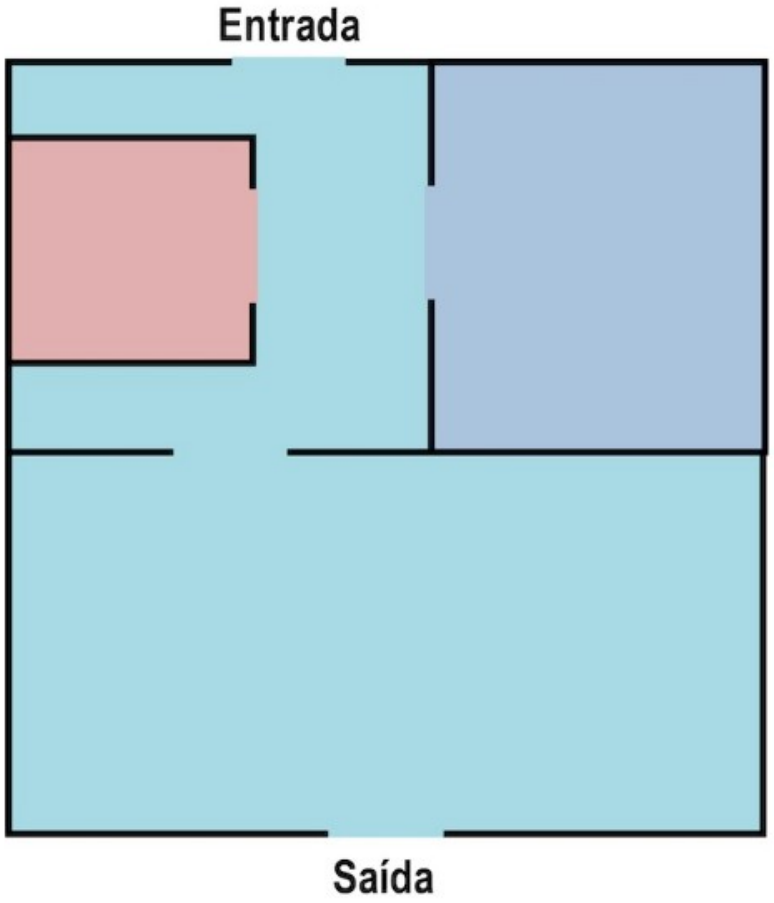
- Algoritmo
  - Inicialmente, uma direção aleatória é escolhida.
  - Ao chegar em uma junção não visitada (ou seja, nenhuma linha), escolha a direção aleatória e risque o caminho.
  - Ao chegar em uma junção já marcada, vire-se e caminhe de volta, marcando o caminho pela segunda vez.
  - Se este não for o caso, escolha o caminho com menos linhas e marque-o novamente.
  - Quando finalmente chegar à saída do labirinto, os caminhos marcados com apenas uma linha indicarão o caminho direto até o ponto inicial.
  - Se não houver saída, você voltará ao ponto inicial, no qual todos os caminhos possuem duas linhas.

Atravessando Labirintos



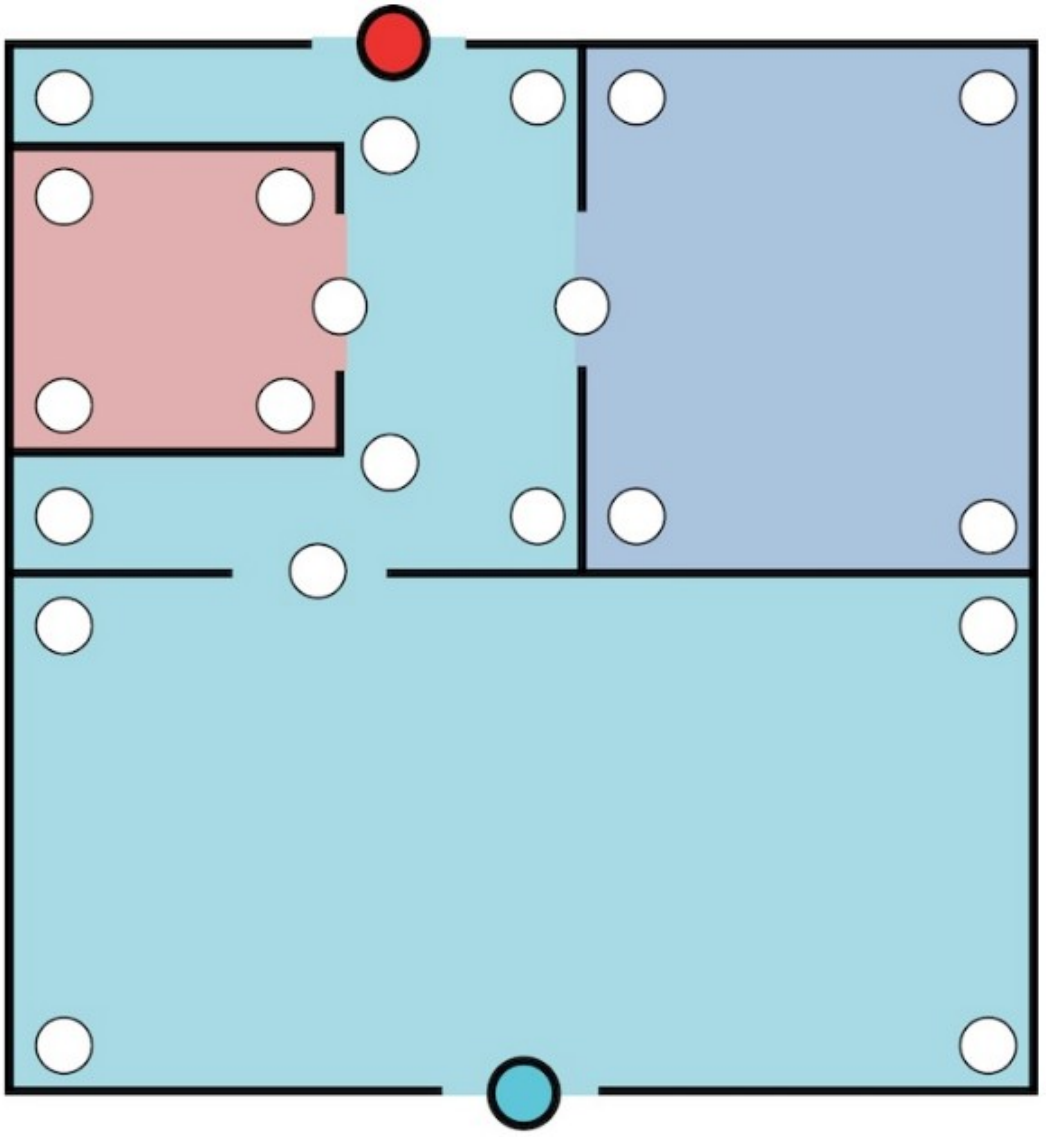
Exemplo de execução do Algoritmo de *Trémaux*.

Atravessando Labirintos



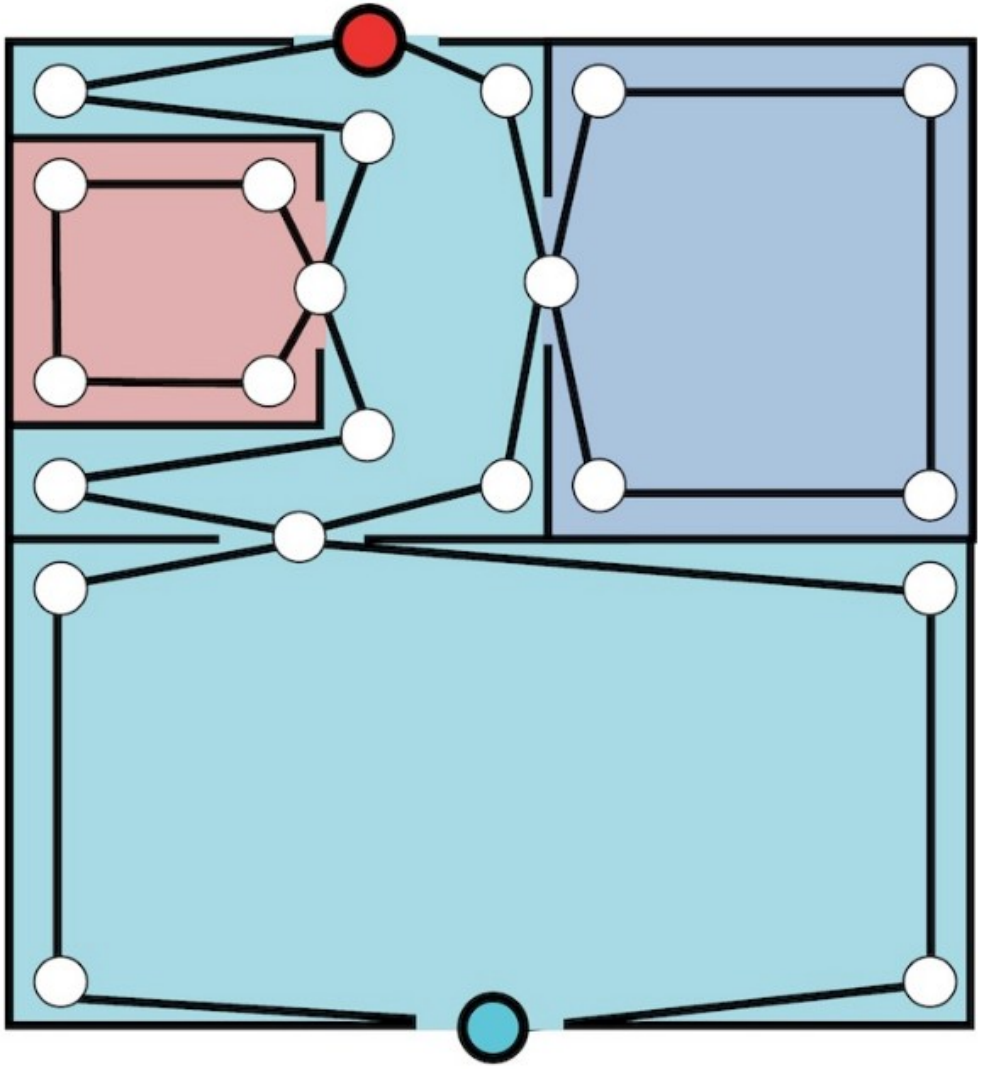
Um labirinto. Considera-se que a circulação neste labirinto é feita margeando alguma parede.

Atravessando Labirintos



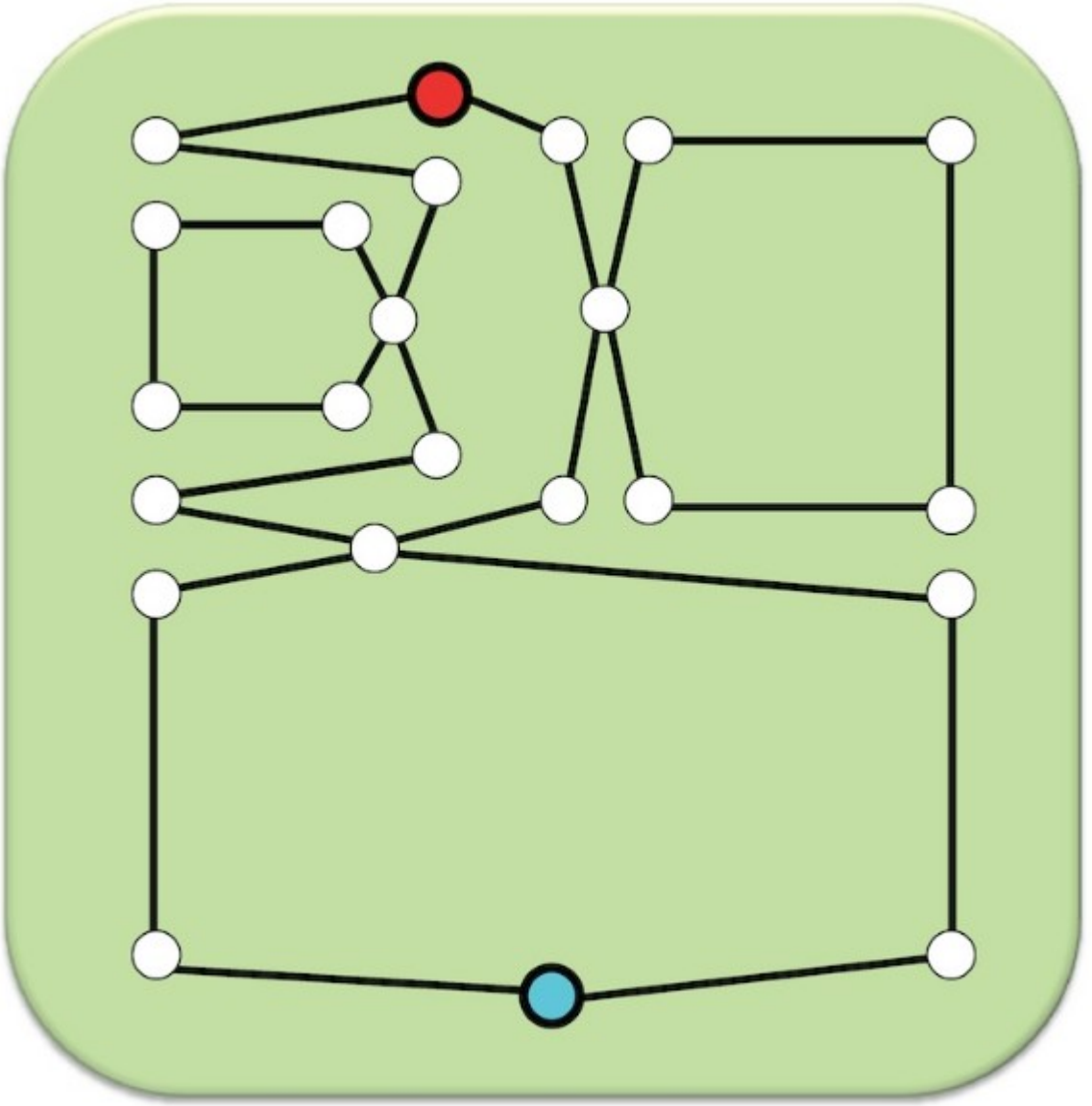
Pontos de entrada, saída e mudança de direção do labirinto.

Atravessando Labirintos



Grafo no labirinto.

Atravessando Labirintos



Grafo do labirinto.

Atravessando Labirintos

Atravessando Labirintos

- Ideia

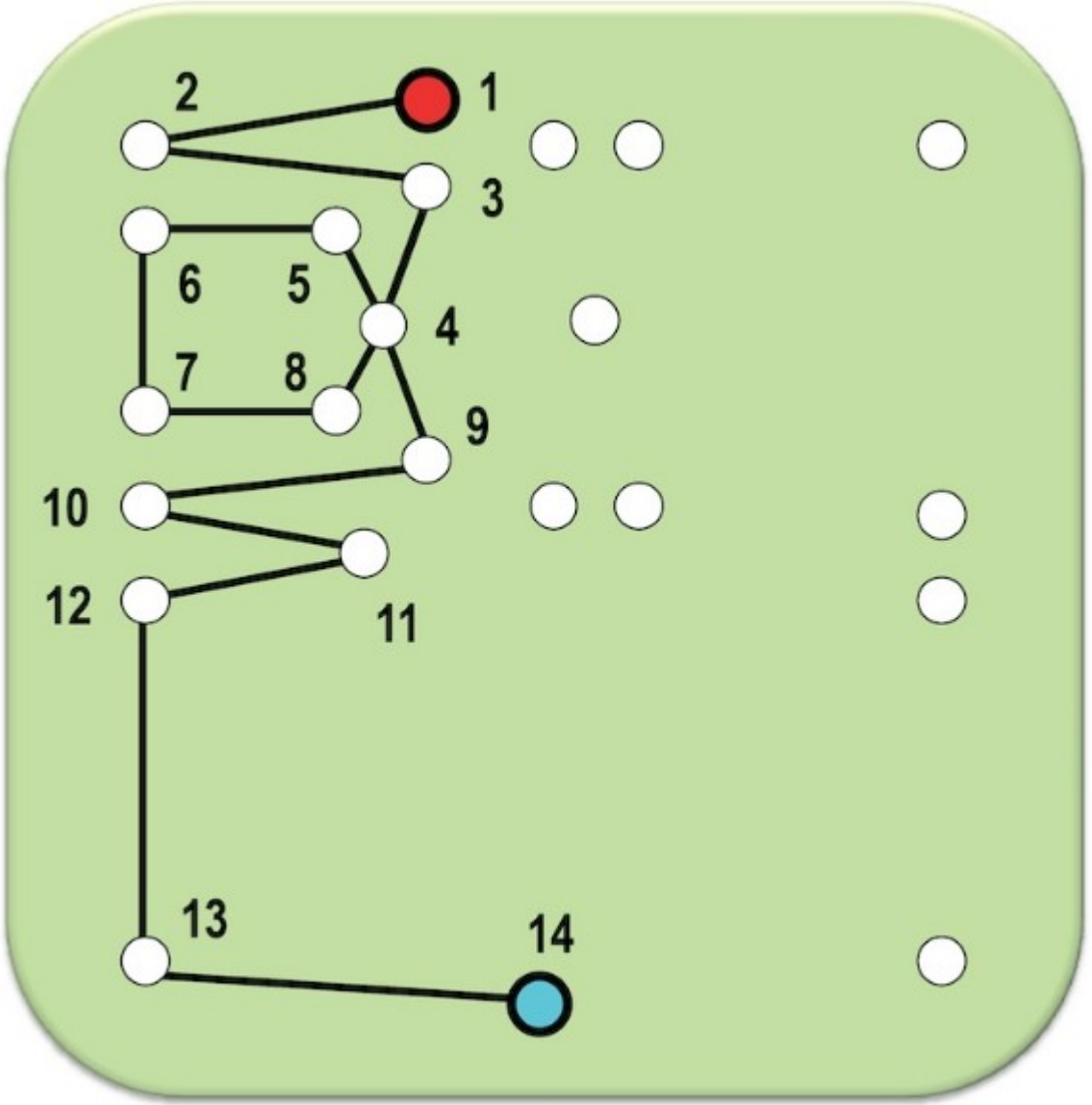
Atravessando Labirintos

- Ideia
  - Uma forma de encontrar a saída do labirinto sem nunca percorrer mais de uma vez o mesmo caminho consiste em realizar uma busca no grafo de modo a nunca repetir arestas, marcando-as.

Atravessando Labirintos

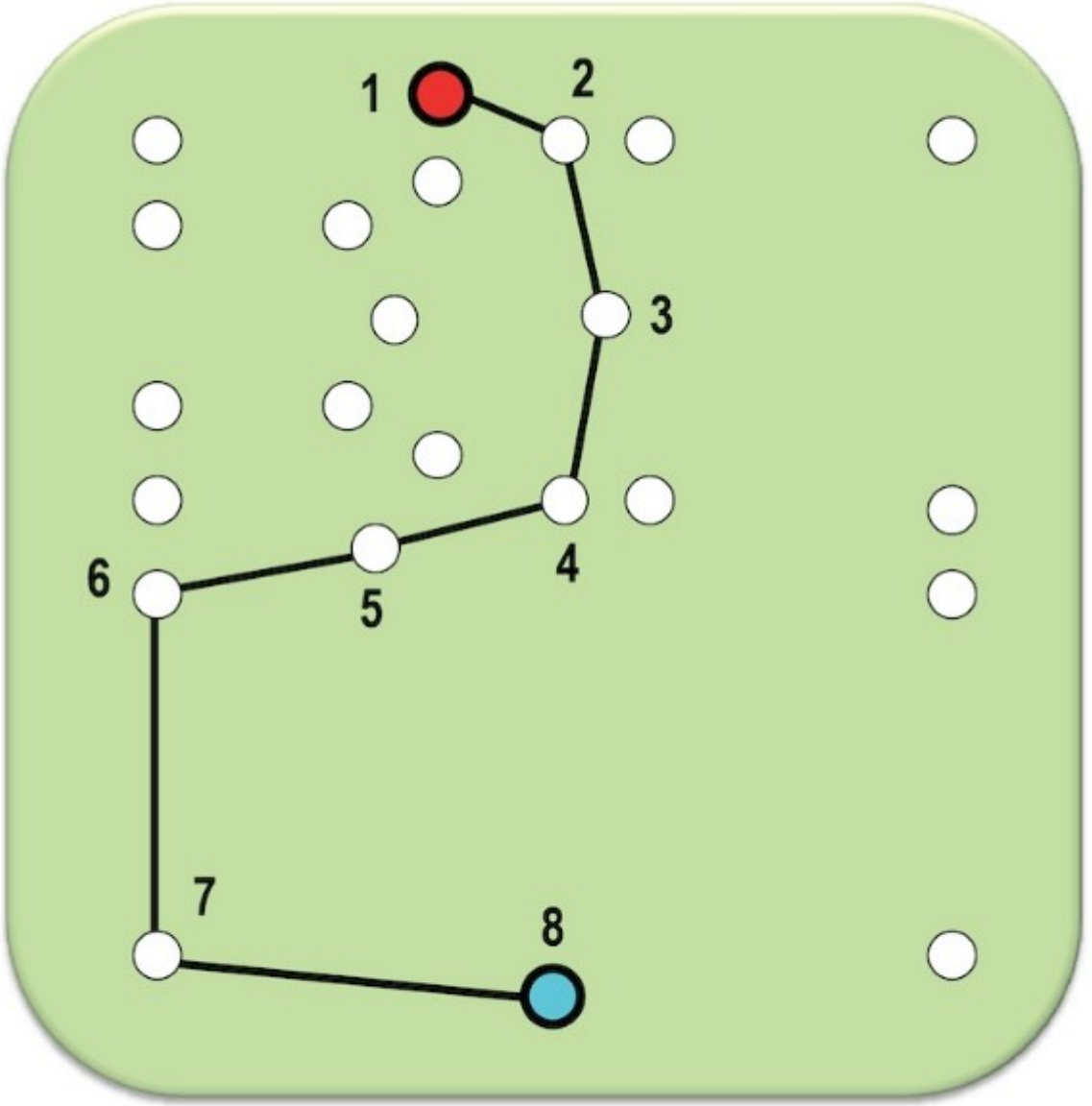
- Ideia
  - Uma forma de encontrar a saída do labirinto sem nunca percorrer mais de uma vez o mesmo caminho consiste em realizar uma busca no grafo de modo a nunca repetir arestas, marcando-as.
  - A busca se inicia no vértice que representa a entrada e termina no vértice que representa a saída.

Atravessando Labirintos



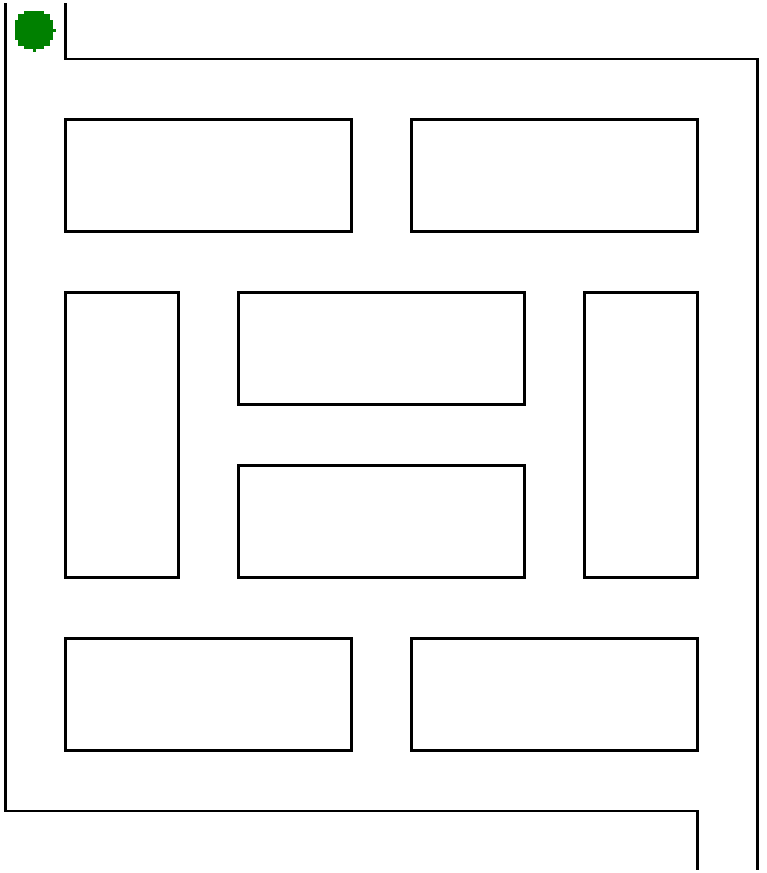
Busca Genérica no grafo de exemplo (1).

Atravessando Labirintos



Busca Genérica no grafo de exemplo (2).

Atravessando Labirintos



Busca em Grafos

Busca em Grafos

Busca em Grafos

- Dependendo do critério utilizado para a escolha dos vértices e arestas a serem visitados, diferentes tipos de buscas são desenvolvidos a partir da busca genérica.

Busca em Grafos

- Dependendo do critério utilizado para a escolha dos vértices e arestas a serem visitados, diferentes tipos de buscas são desenvolvidos a partir da busca genérica.
- Basicamente, duas buscas completas em grafos são essenciais:

Busca em Grafos

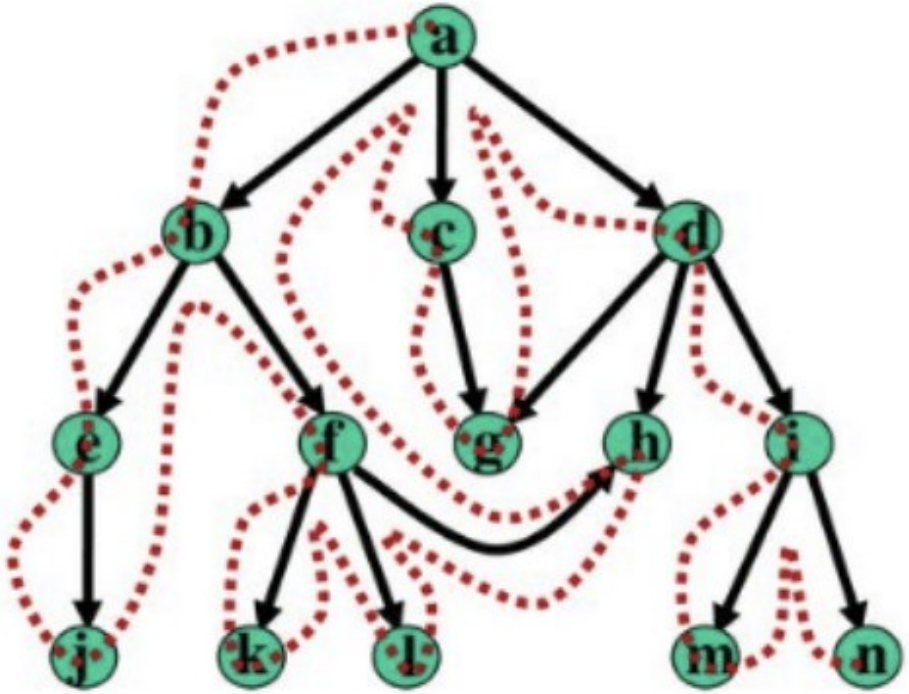
- Dependendo do critério utilizado para a escolha dos vértices e arestas a serem visitados, diferentes tipos de buscas são desenvolvidos a partir da busca genérica.
- Basicamente, duas buscas completas em grafos são essenciais:
  - Busca em **profundidade** (ou **DFS** – *Depth-First Search*);

Busca em Grafos

- Dependendo do critério utilizado para a escolha dos vértices e arestas a serem visitados, diferentes tipos de buscas são desenvolvidos a partir da busca genérica.
- Basicamente, duas buscas completas em grafos são essenciais:
  - Busca em **profundidade** (ou **DFS** – *Depth-First Search*);
  - Busca em **largura** (ou **BFS** – *Breadth-First Search*).

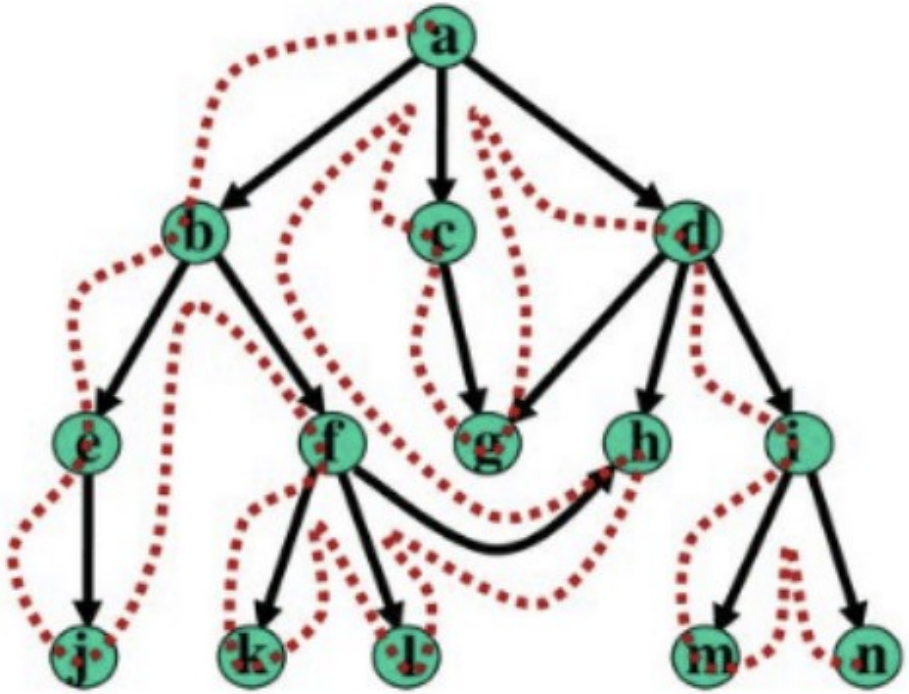
DFS versus BFS

DFS versus BFS

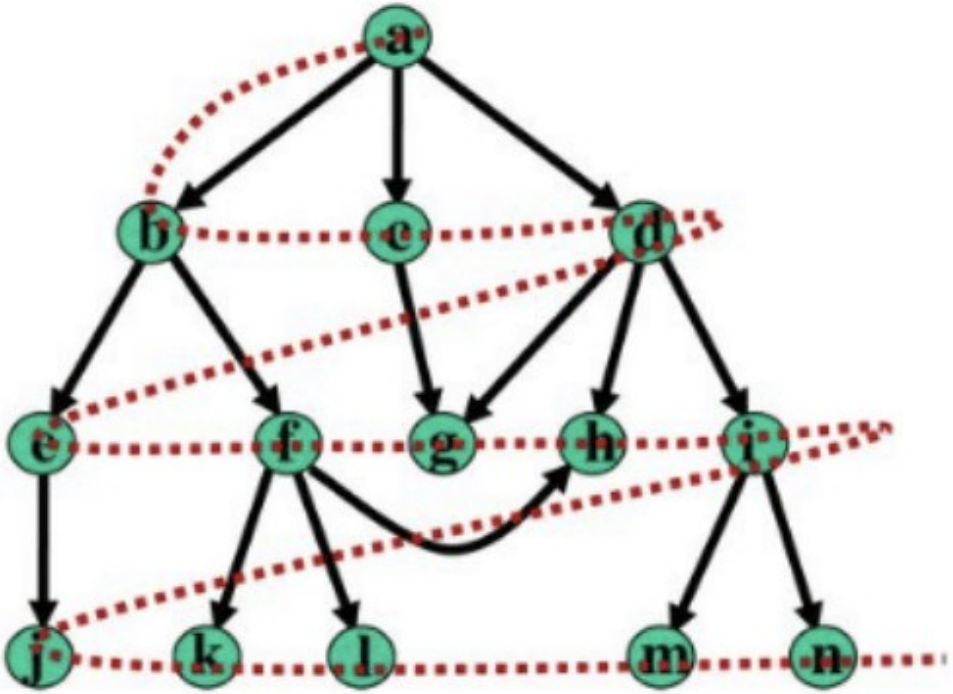


DFS

DFS versus BFS



DFS



BFS

Busca em Profundidade (DFS)

Busca em Profundidade (DFS)

- Características

Busca em Profundidade (DFS)

- Características
  - A busca em profundidade visita todos os vértices de um grafo, usando como critério **os vizinhos do vértice visitado mais recentemente**.

Busca em Profundidade (DFS)

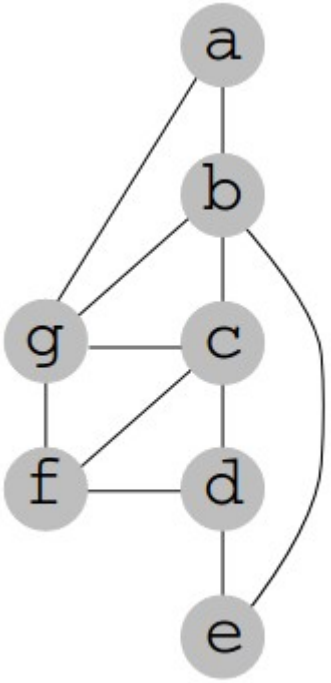
- Características
  - A busca em profundidade visita todos os vértices de um grafo, usando como critério **os vizinhos do vértice visitado mais recentemente**.
  - Principal característica:

Busca em Profundidade (DFS)

- Características
  - A busca em profundidade visita todos os vértices de um grafo, usando como critério **os vizinhos do vértice visitado mais recentemente**.
  - Principal característica:
    - Utiliza uma **pilha explícita** ou **recursividade** para guiar a busca.

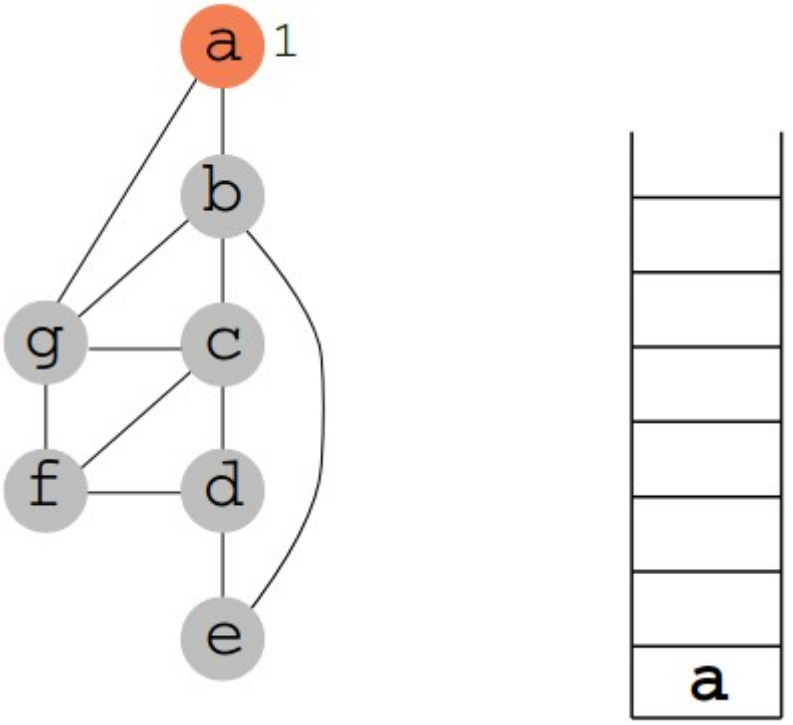
Busca em Profundidade (DFS)

- Exemplo de uso de pilha:



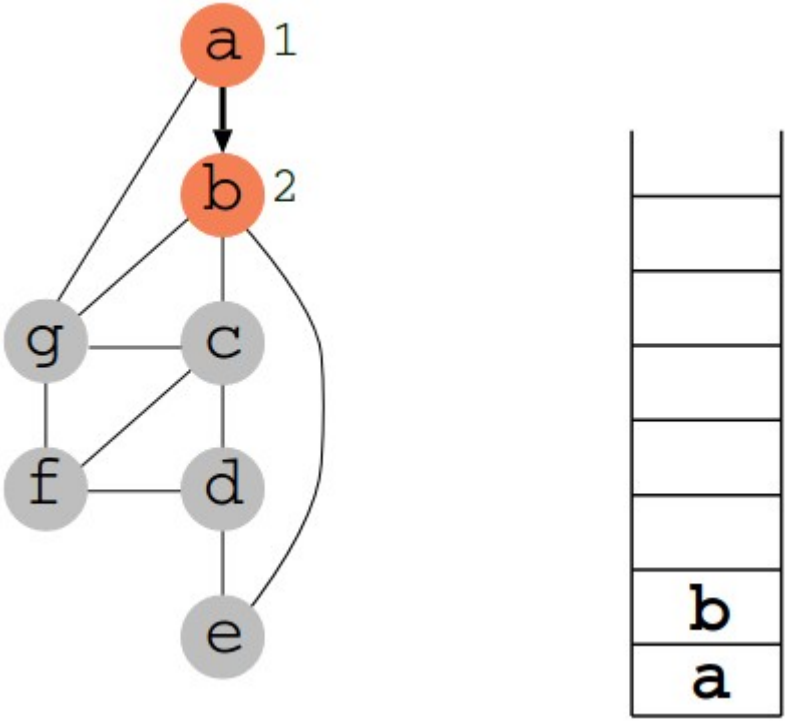
Busca em Profundidade (DFS)

- Exemplo de uso de pilha:



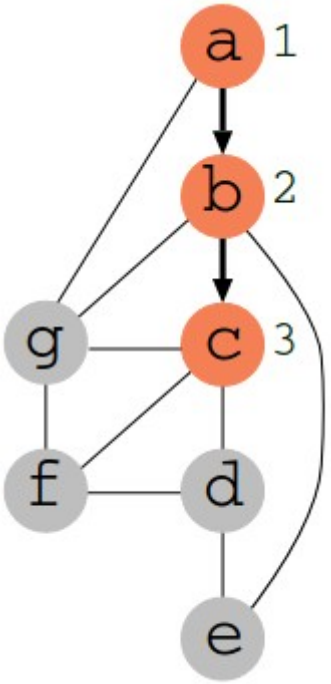
Busca em Profundidade (DFS)

- Exemplo de uso de pilha:



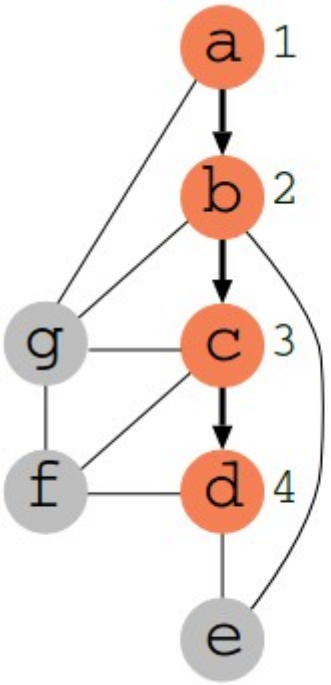
Busca em Profundidade (DFS)

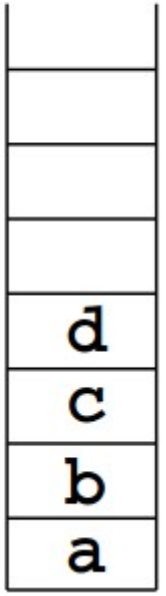
- Exemplo de uso de pilha:



Busca em Profundidade (DFS)

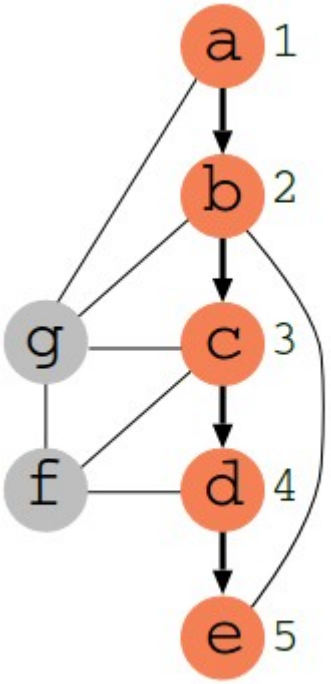
- Exemplo de uso de pilha:





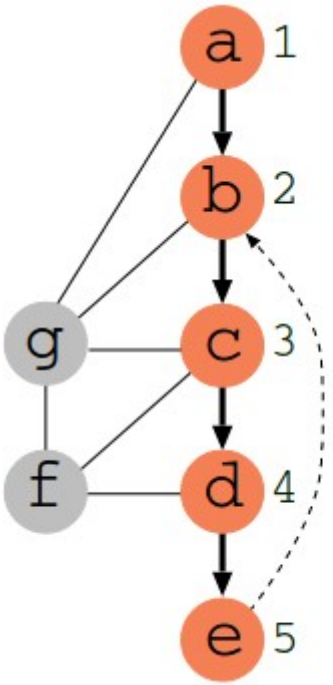
Busca em Profundidade (DFS)

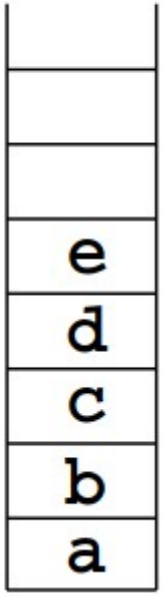
- Exemplo de uso de pilha:



Busca em Profundidade (DFS)

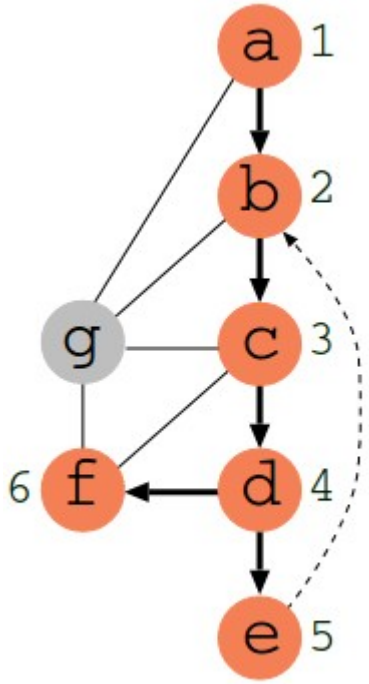
- Exemplo de uso de pilha:

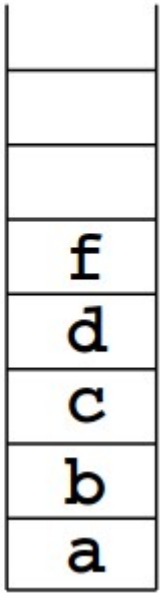




Busca em Profundidade (DFS)

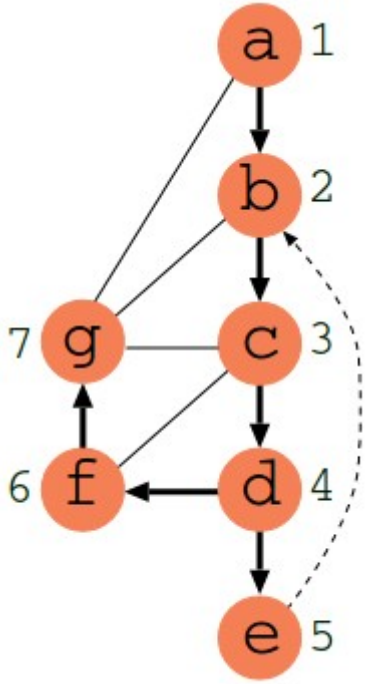
- Exemplo de uso de pilha:

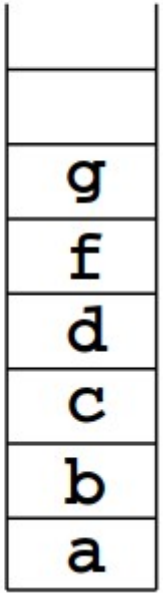




Busca em Profundidade (DFS)

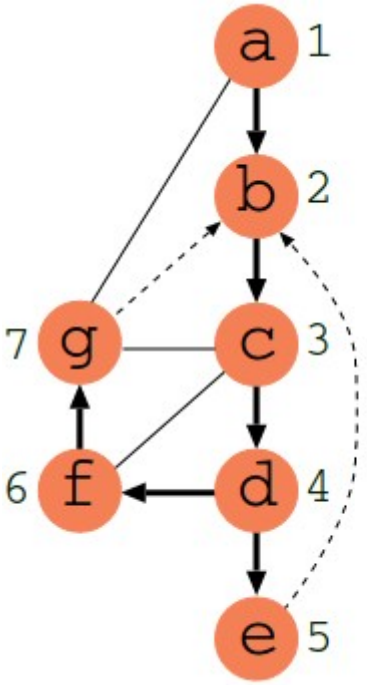
- Exemplo de uso de pilha:

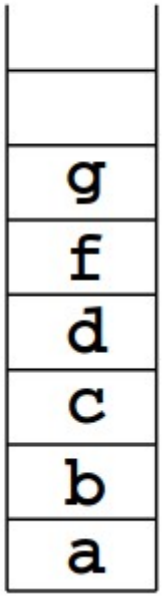




Busca em Profundidade (DFS)

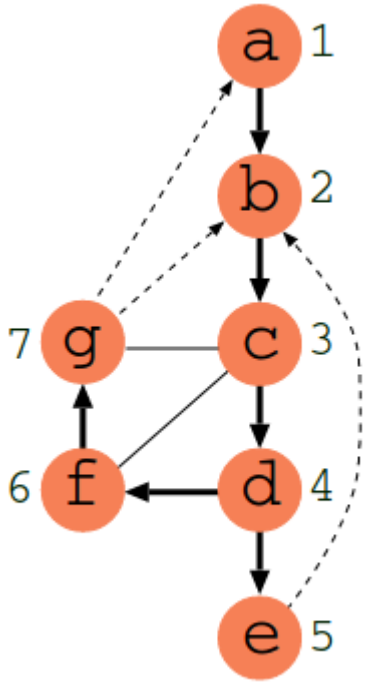
- Exemplo de uso de pilha:

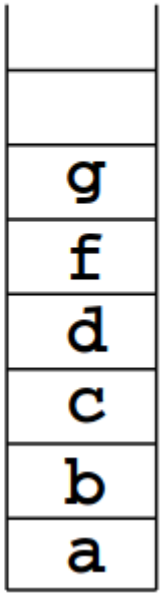




Busca em Profundidade (DFS)

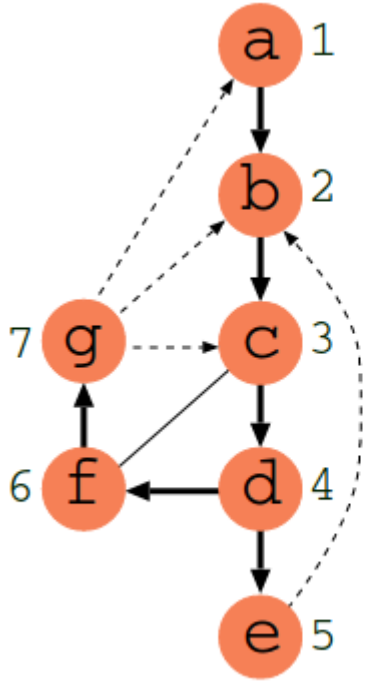
- Exemplo de uso de pilha:

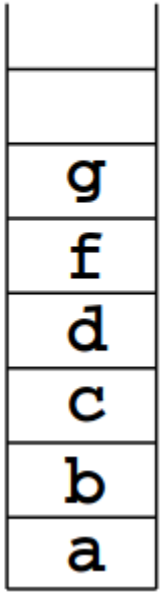




Busca em Profundidade (DFS)

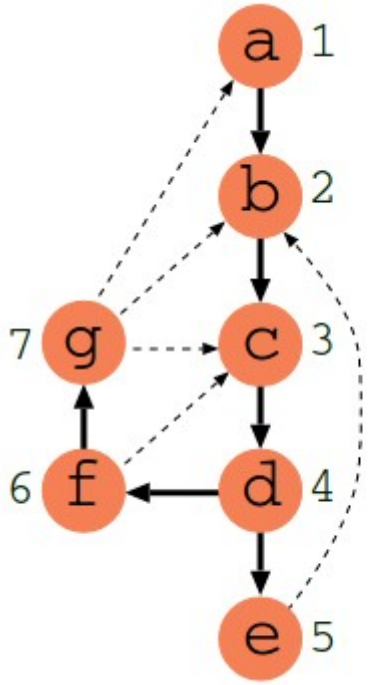
- Exemplo de uso de pilha:

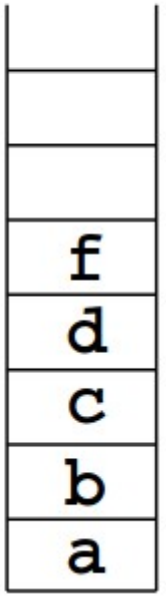




Busca em Profundidade (DFS)

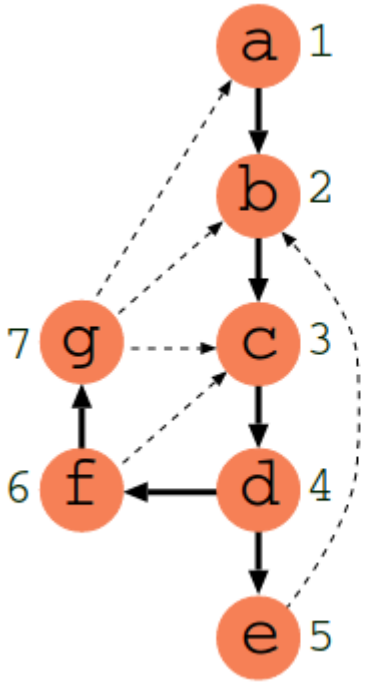
- Exemplo de uso de pilha:

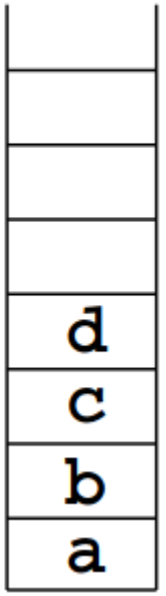




Busca em Profundidade (DFS)

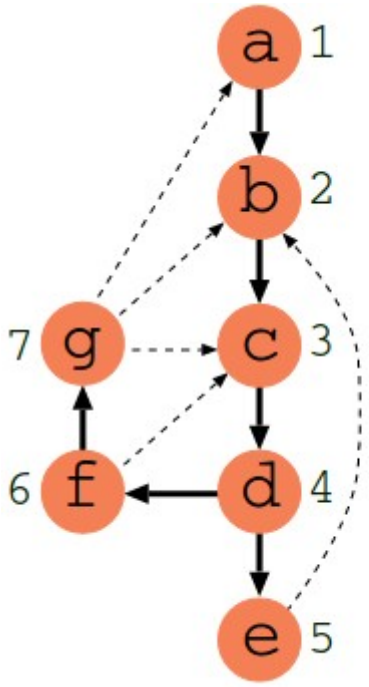
- Exemplo de uso de pilha:





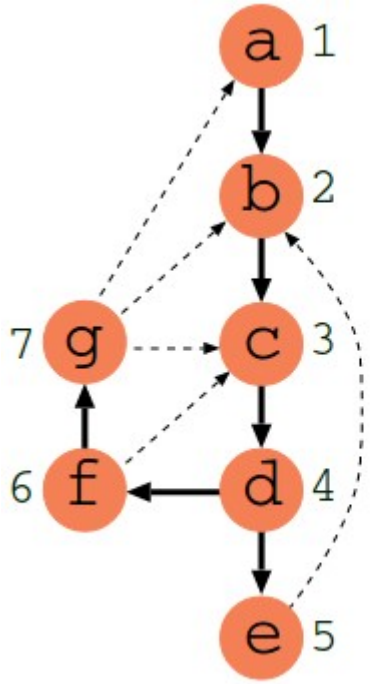
Busca em Profundidade (DFS)

- Exemplo de uso de pilha:



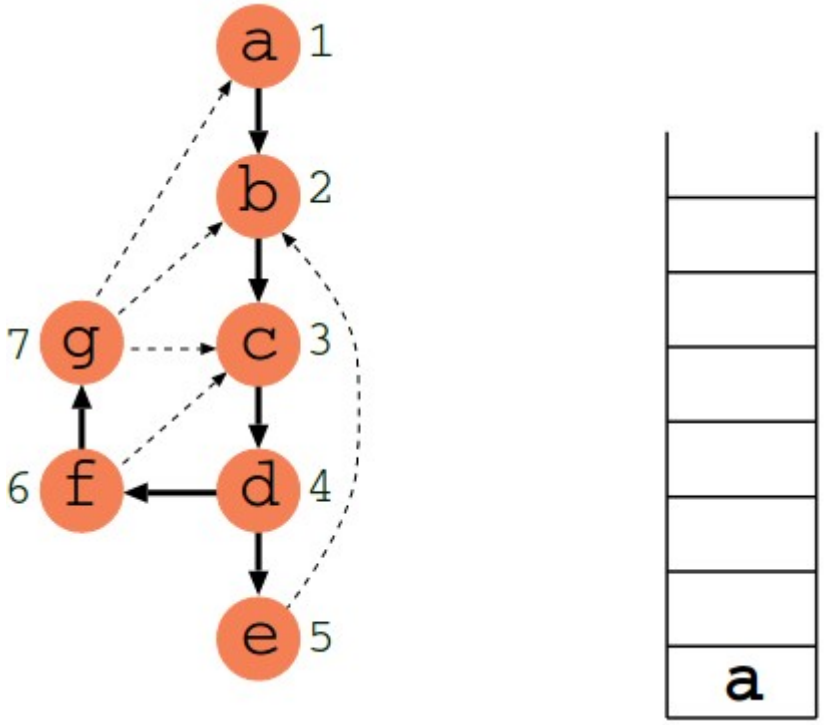
Busca em Profundidade (DFS)

- Exemplo de uso de pilha:



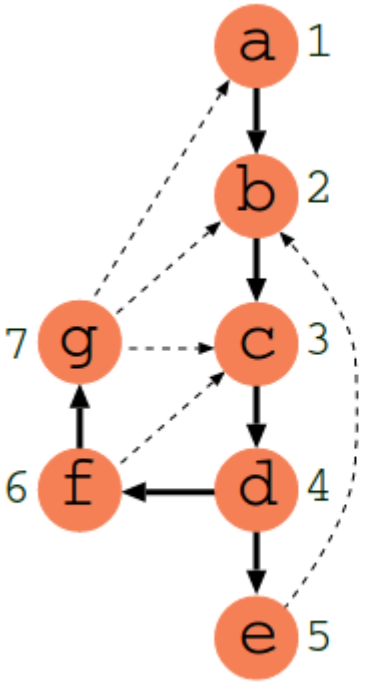
Busca em Profundidade (DFS)

- Exemplo de uso de pilha:



Busca em Profundidade (DFS)

- Exemplo de uso de pilha:



Busca em Profundidade (DFS)

```
   Entrada: Grafo G=(V, A), vértice inicial v
 1 Marque o vértice v como visitado;
 2 enquanto existir w vizinho de v faça
 3 |   se w é marcado como não visitado então
 4 |   |   Visite a aresta {v, w};
 5 |   |   Marque w como visitado;
 6 |   |   BP(G, w);//chamada recursiva da função
 7 |   fim
 8 |   senão
 9 |   |   se {v, w} não foi visitada ainda então
10 |   |   |   Visite {v, w};
11 |   |   fim
12 |   fim
13 fim
```

Busca em Profundidade (DFS)

Busca em Profundidade (DFS)

- Classificação de arestas

Busca em Profundidade (DFS)

- Classificação de arestas
  - Ao explorar um grafo G conexo usando a DFS, podemos categorizar as arestas:

Busca em Profundidade (DFS)

- Classificação de arestas
  - Ao explorar um grafo G conexo usando a DFS, podemos categorizar as arestas:
    - Arestas de árvore: satisfazem ao primeiro **se** do algoritmo (linha 3), ou seja, levam à visitação de vértices ainda não visitados.

Busca em Profundidade (DFS)

- Classificação de arestas
  - Ao explorar um grafo G conexo usando a DFS, podemos categorizar as arestas:
    - Arestas de árvore: satisfazem ao primeiro **se** do algoritmo (linha 3), ou seja, levam à visitação de vértices ainda não visitados.
    - Arestas de retorno: demais arestas. Formam ciclos, pois levam a vértices já visitados.

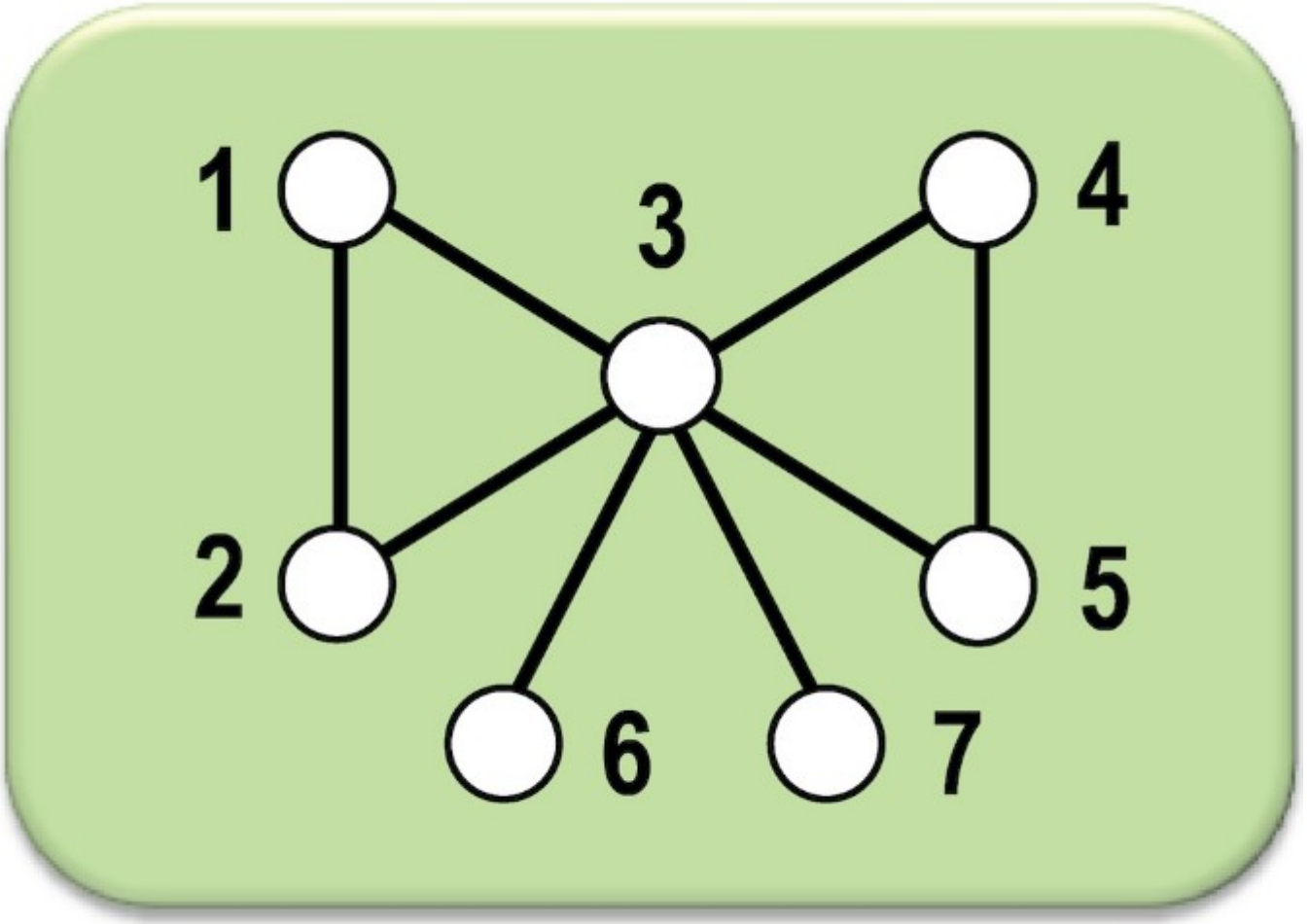
Busca em Profundidade (DFS)

- Classificação de arestas
  - Ao explorar um grafo G conexo usando a DFS, podemos categorizar as arestas:
    - Arestas de árvore: satisfazem ao primeiro **se** do algoritmo (linha 3), ou seja, levam à visitação de vértices ainda não visitados.
    - Arestas de retorno: demais arestas. Formam ciclos, pois levam a vértices já visitados.
  - Árvore de profundidade

Busca em Profundidade (DFS)

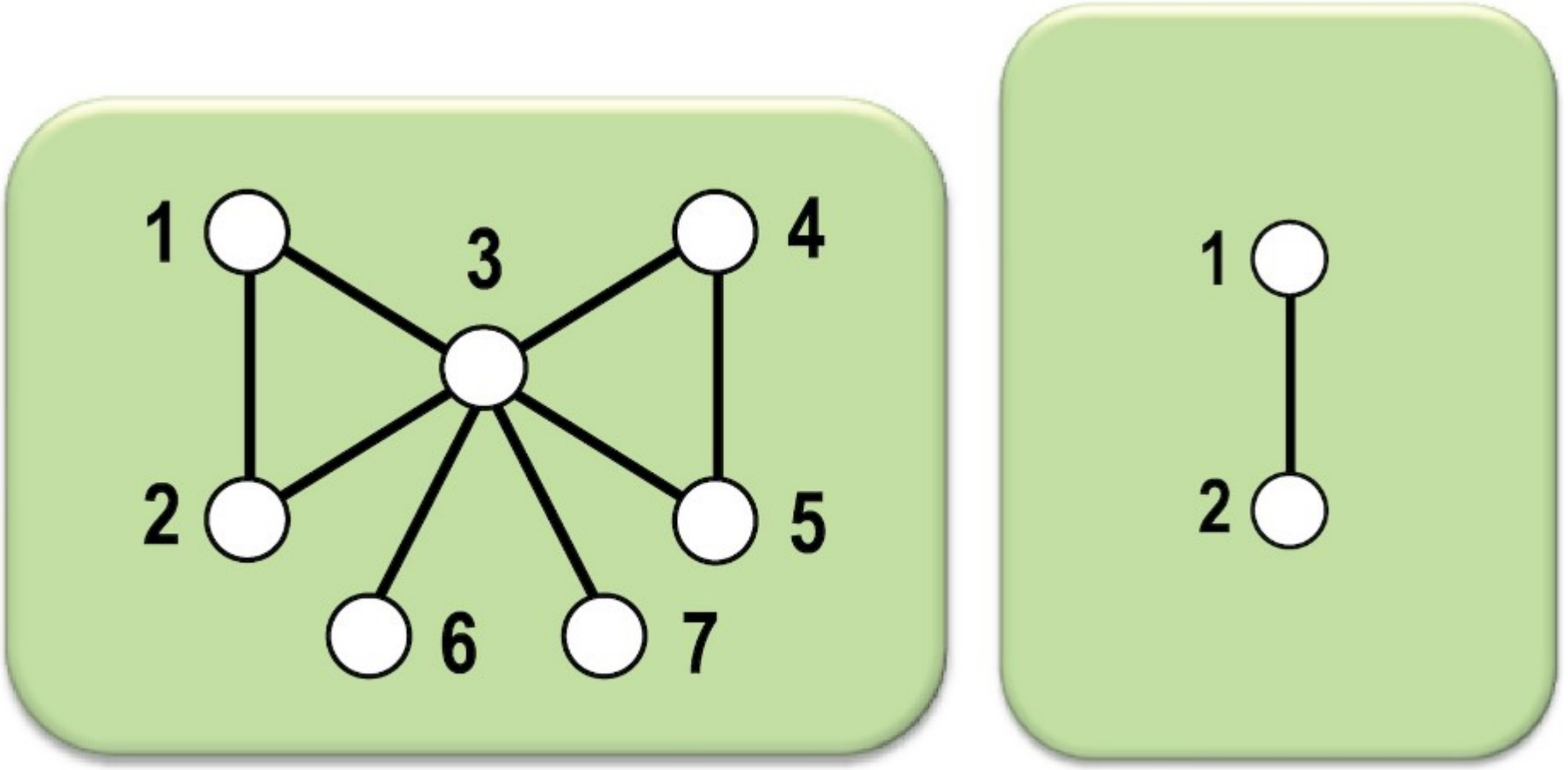
- Classificação de arestas
  - Ao explorar um grafo G conexo usando a DFS, podemos categorizar as arestas:
    - Arestas de árvore: satisfazem ao primeiro **se** do algoritmo (linha 3), ou seja, levam à visitação de vértices ainda não visitados.
    - Arestas de retorno: demais arestas. Formam ciclos, pois levam a vértices já visitados.
  - Árvore de profundidade
    - A subárvore G formada pelas arestas de árvore é chamada de **Árvore de Profundidade G**.

DFS - Exemplo



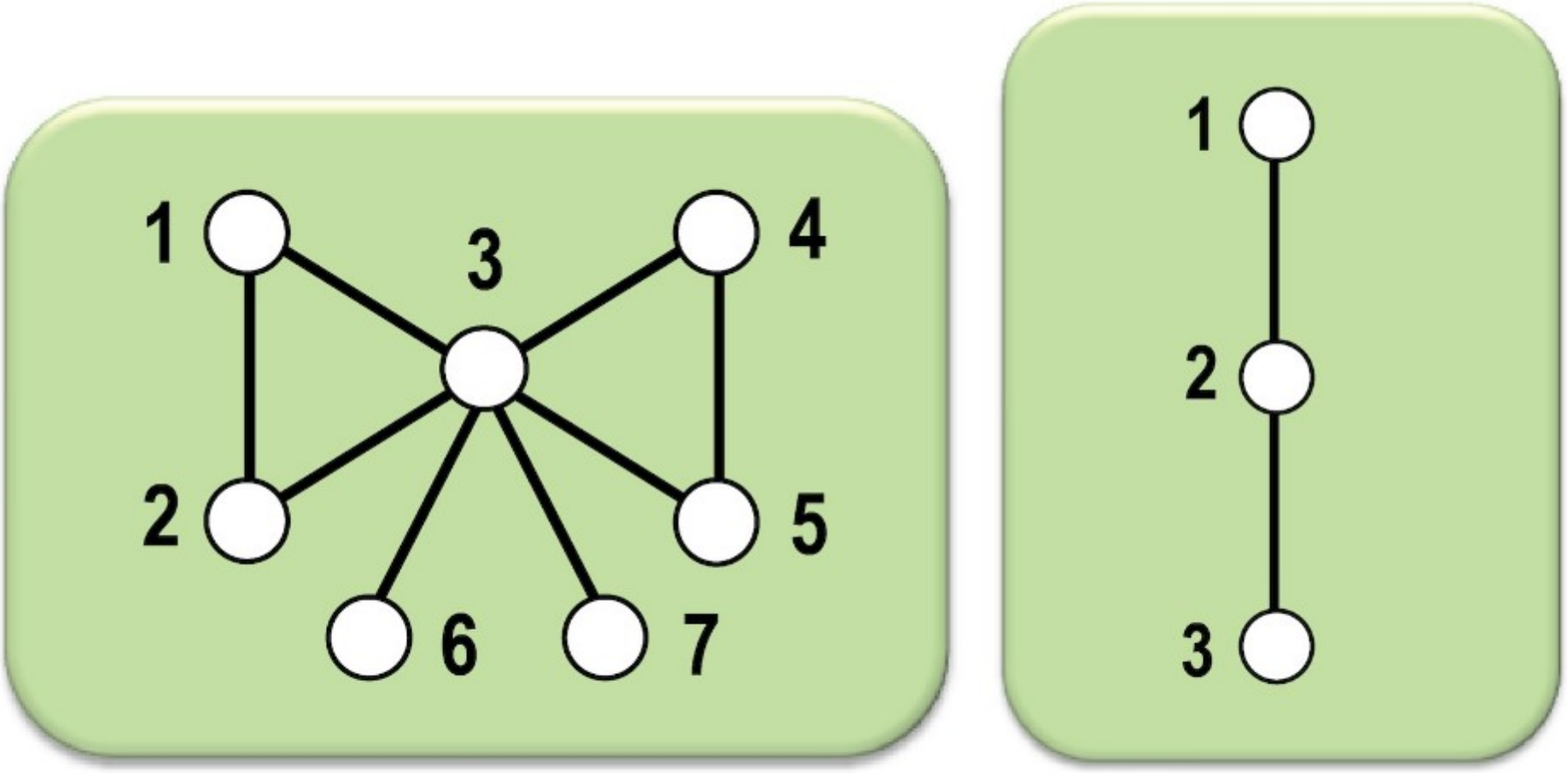
Grafo de exemplo.

DFS - Exemplo



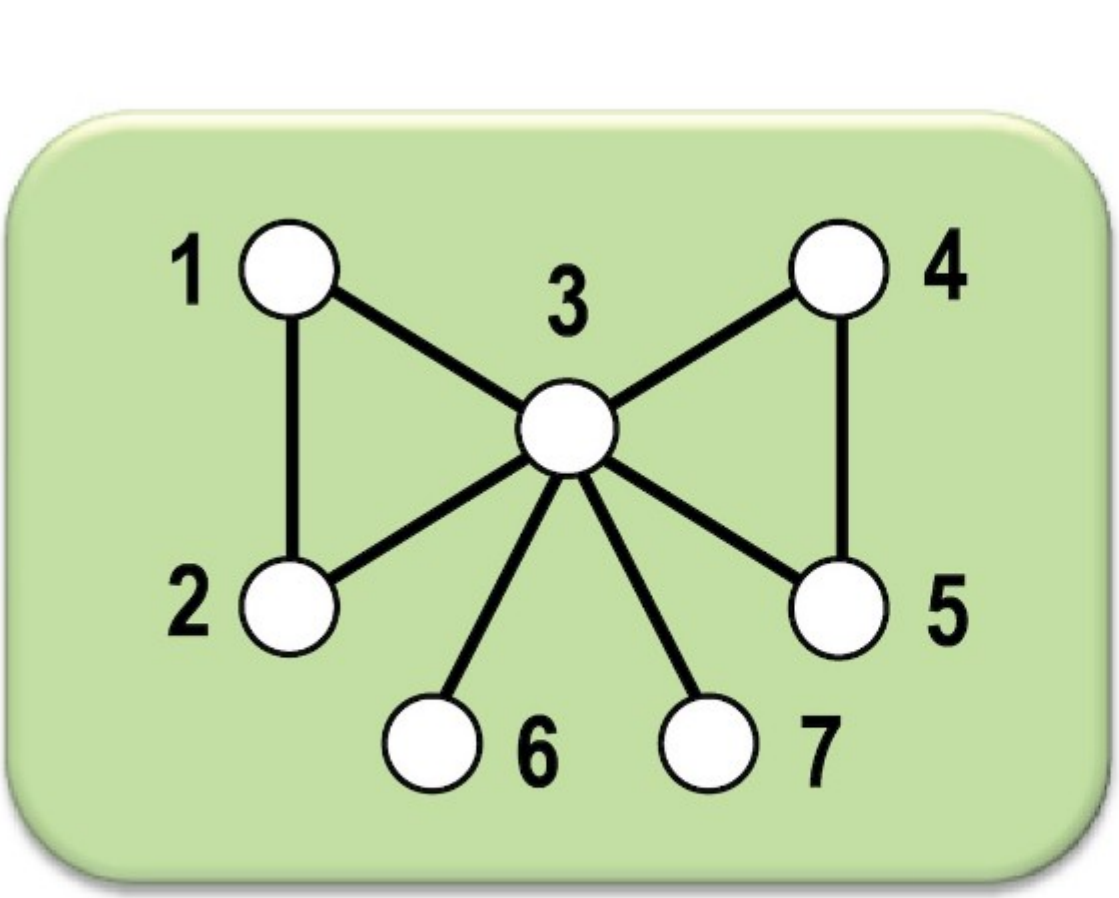
(1) Aresta {1,2}.

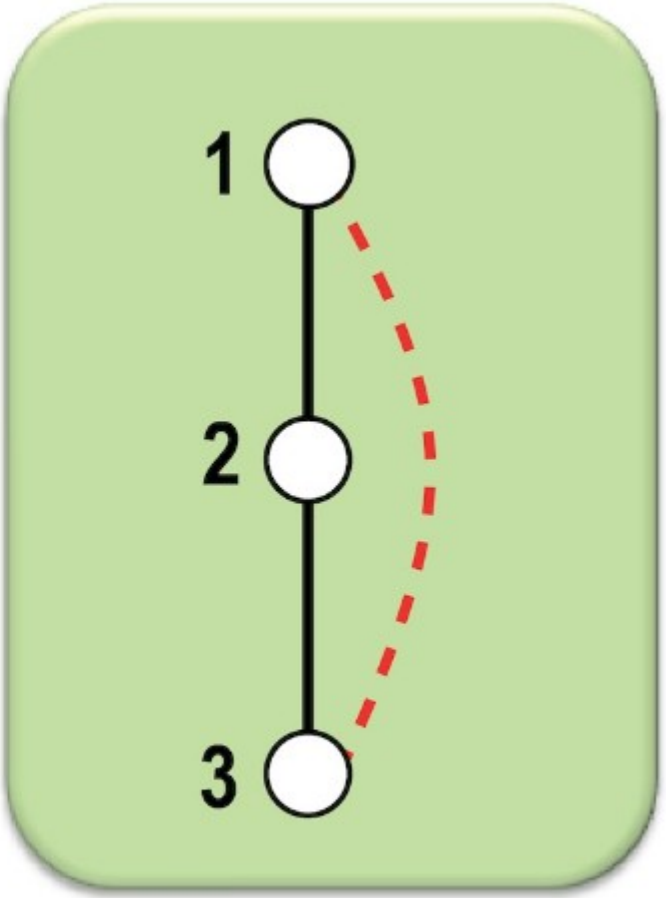
DFS - Exemplo



(2) Aresta {2,3}.

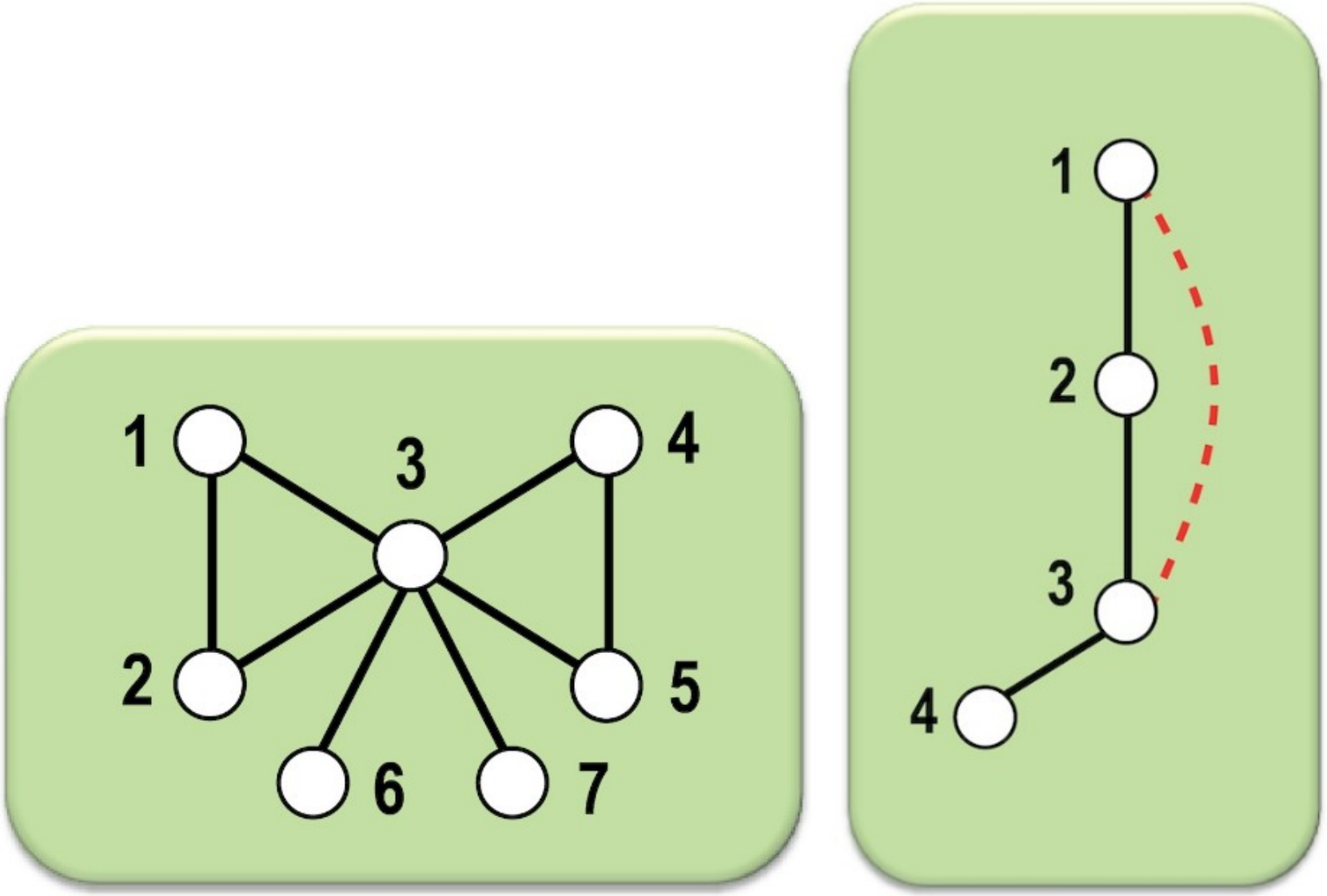
DFS - Exemplo





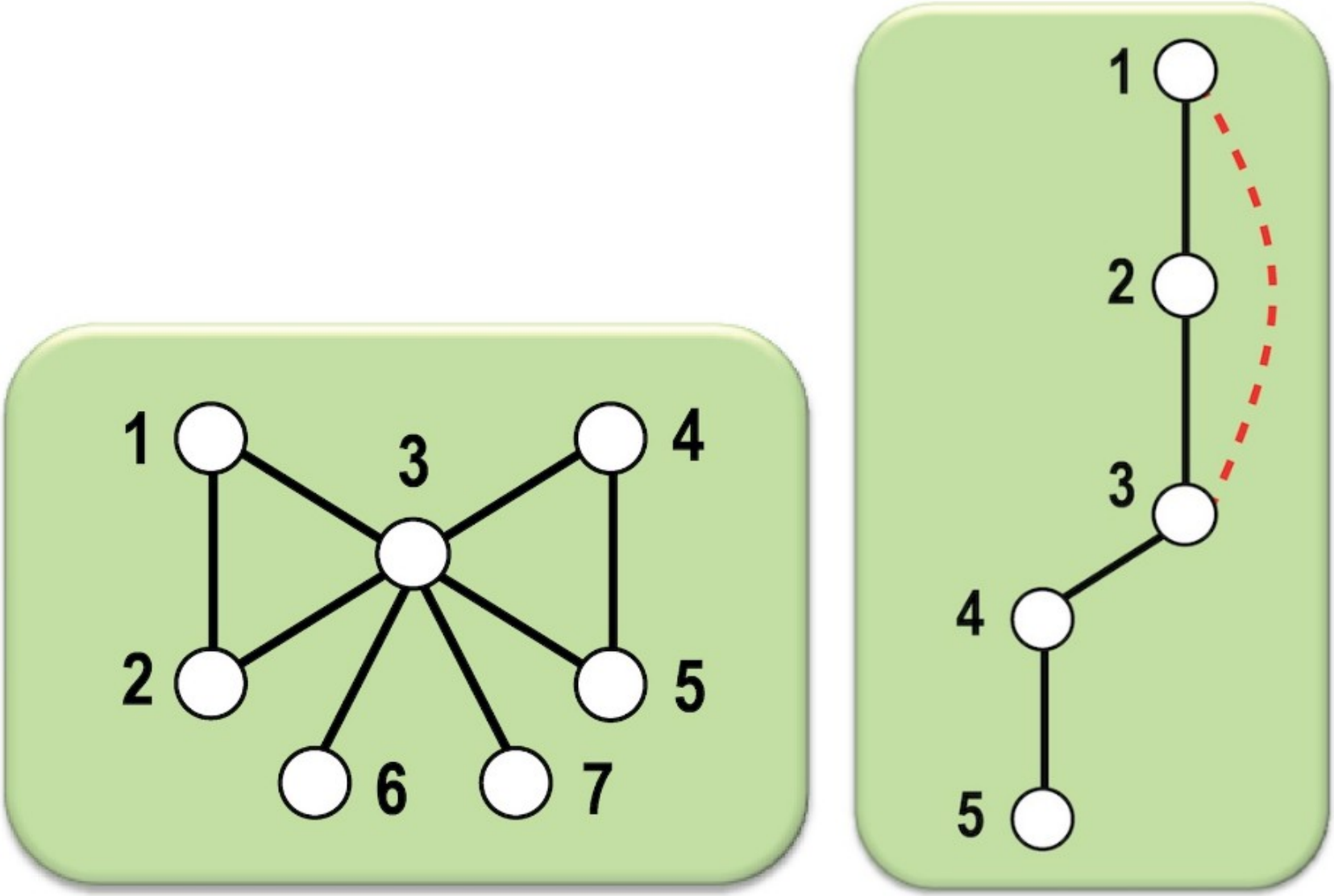
(3) Aresta $\{3, 1\}$.

DFS - Exemplo



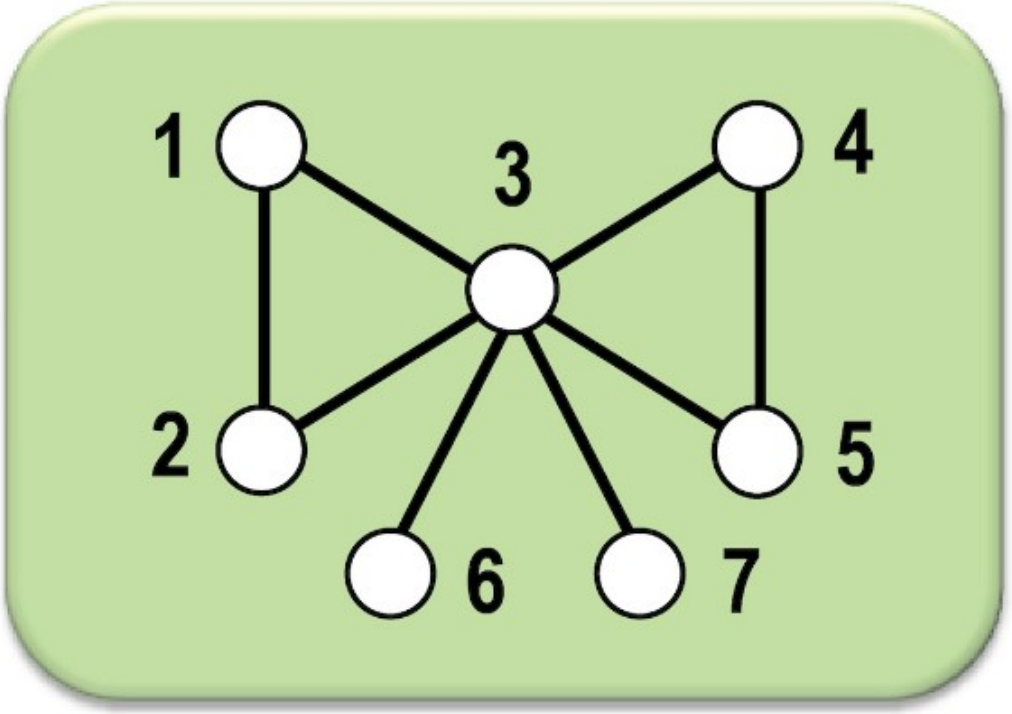
(4) Aresta {3, 4}.

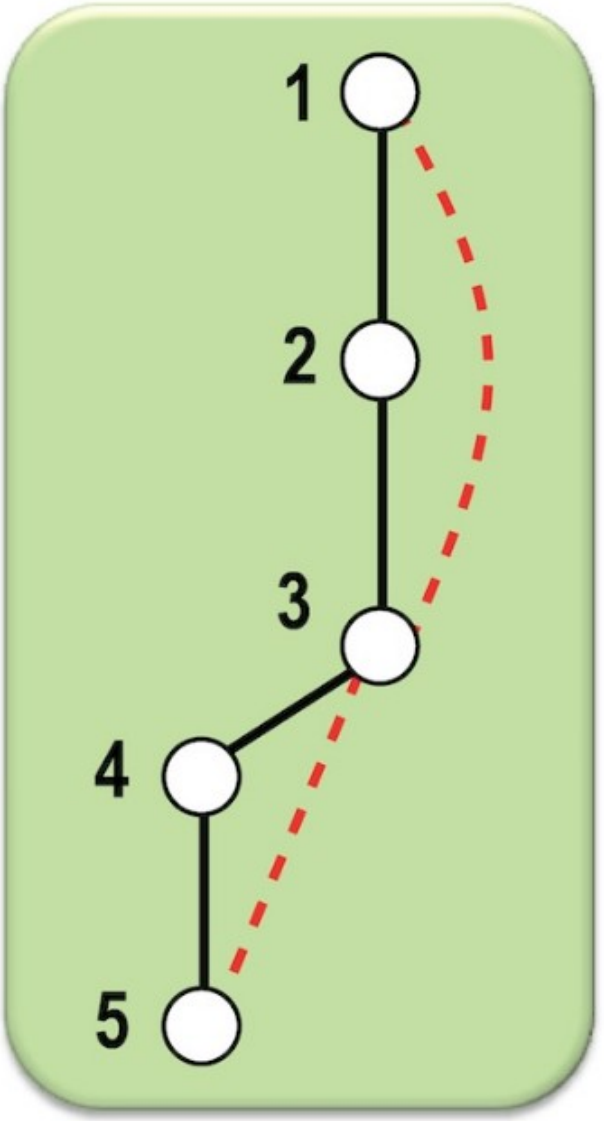
DFS - Exemplo



(5) Aresta $\{4, 5\}$.

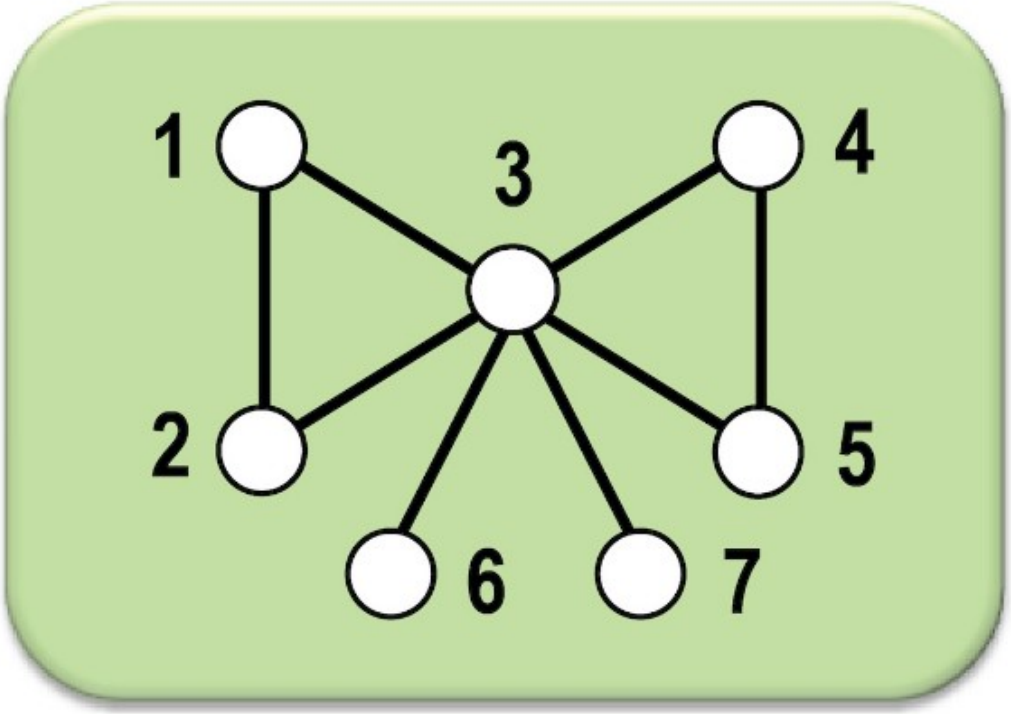
DFS - Exemplo

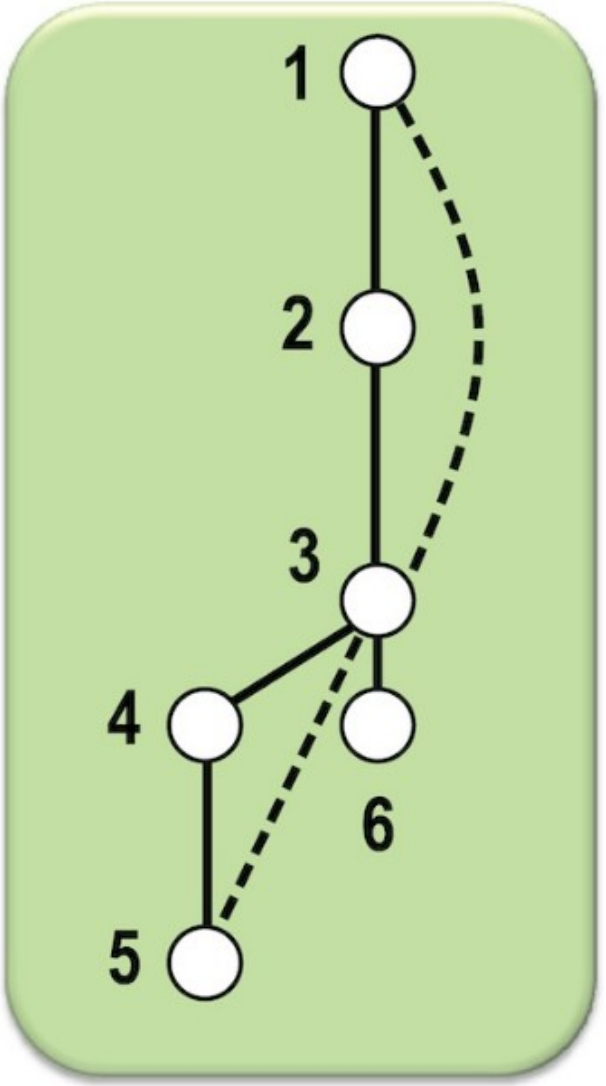




(6) Aresta $\{5, 3\}$.

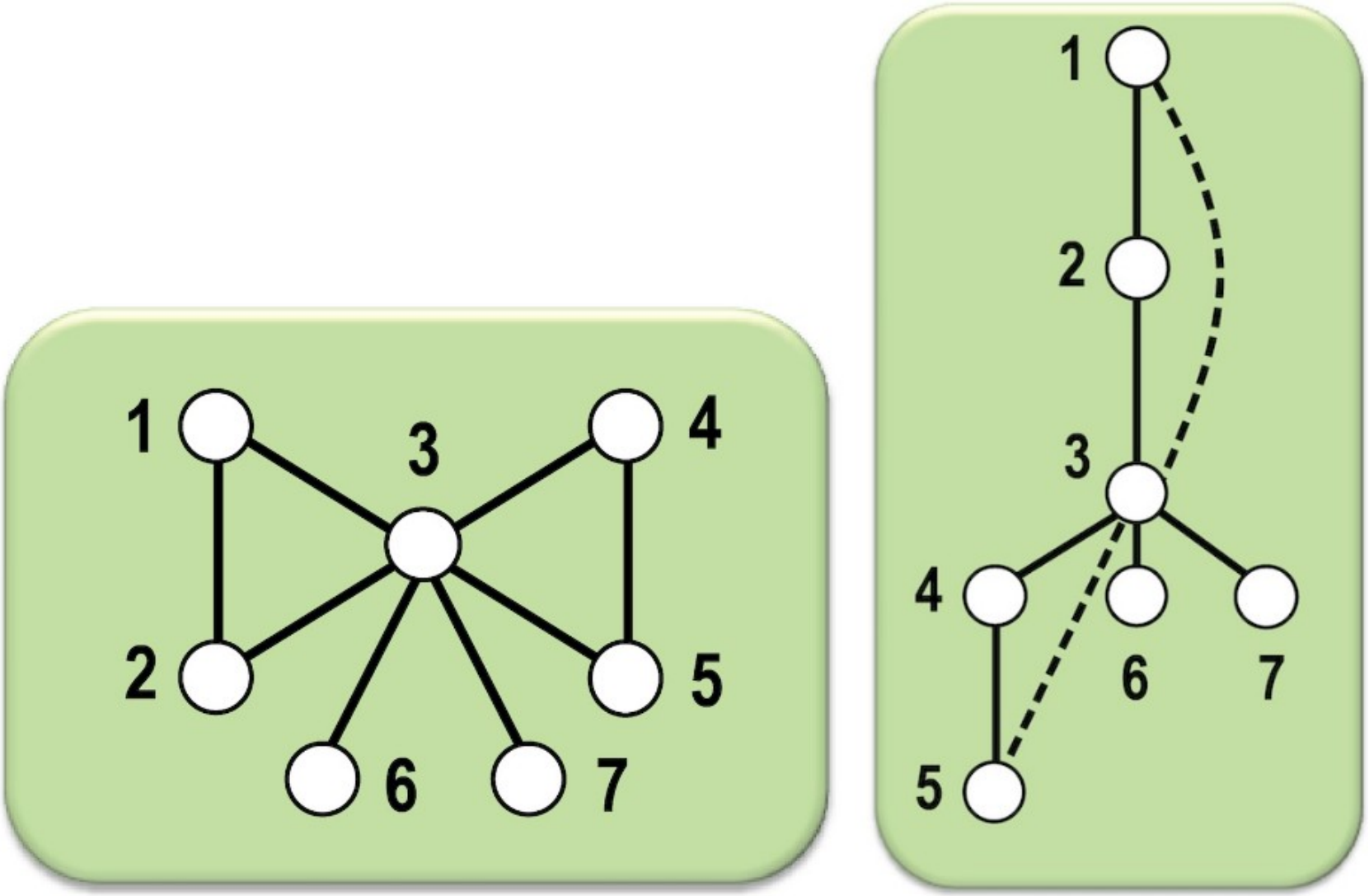
DFS - Exemplo





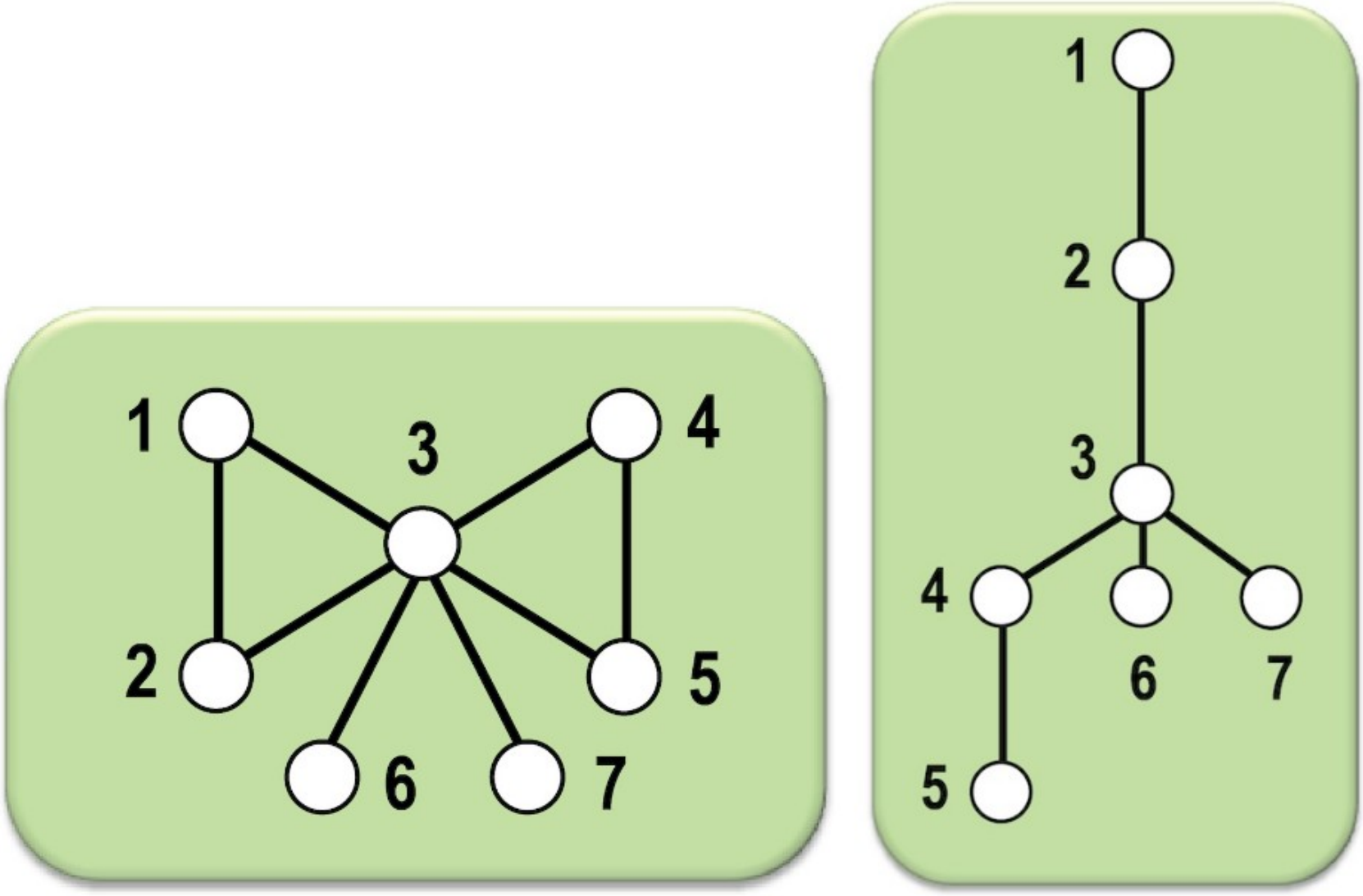
(7) Aresta $\{3, 6\}$.

DFS - Exemplo



(8) Aresta {3, 7}.

DFS - Exemplo



Grafo original e correspondente árvore de profundidade.

DFS

DFS

- Complexidade

DFS

- Complexidade
  - Para cada vértice do grafo, a DFS percorre todos os seus vizinhos.

DFS

- Complexidade
  - Para cada vértice do grafo, a DFS percorre todos os seus vizinhos.
  - Cada aresta é visitada duas vezes.

DFS

- Complexidade
  - Para cada vértice do grafo, a DFS percorre todos os seus vizinhos.
  - Cada aresta é visitada duas vezes.
  - Se apresentarmos o grafo por uma lista de adjacências, a DFS tem complexidade **O(|V| + |E|)**.

DFS – Grafos Direcionados

DFS – Grafos Direcionados

- A aplicação da DFS em grafos direcionados é essencialmente igual à aplicação em grafos não direcionados.

DFS – Grafos Direcionados

- A aplicação da DFS em grafos direcionados é essencialmente igual à aplicação em grafos não direcionados.
- No entanto, mesmo o grafo direcionado sendo conexo, a DFS pode precisar ser chamada repetidas vezes enquanto houver vértices não visitados, retornando uma floresta.

DFS – Grafos Direcionados

- A aplicação da DFS em grafos direcionados é essencialmente igual à aplicação em grafos não direcionados.
- No entanto, mesmo o grafo direcionado sendo conexo, a DFS pode precisar ser chamada repetidas vezes enquanto houver vértices não visitados, retornando uma floresta.
- Este é o mesmo caso quando a DFS é aplicada a um GND desconexo.

Busca em Profundidade - Reinício

Entrada: Grafo $G=(V, A)$
1 **enquanto** *existir* $v \in V$ *não visitado* **faça**
2 | BP(G, v);
3 **fim**

Busca em Profundidade

Busca em Profundidade

- Classificação de arestas

Busca em Profundidade

- Classificação de arestas
  - Ao explorar um grafo G direcionado usando a DFS, podemos categorizar as arestas.

Busca em Profundidade

- Classificação de arestas
  - Ao explorar um grafo G direcionado usando a DFS, podemos categorizar as arestas.
  - Sejam o vértice *v* a origem da aresta e o vértice *w* o destino da mesma:

Busca em Profundidade

- Classificação de arestas
  - Ao explorar um grafo G direcionado usando a DFS, podemos categorizar as arestas.
  - Sejam o vértice *v* a origem da aresta e o vértice *w* o destino da mesma:
    - Arcos de avanço

Busca em Profundidade

- Classificação de arestas
  - Ao explorar um grafo G direcionado usando a DFS, podemos categorizar as arestas.
  - Sejam o vértice *v* a origem da aresta e o vértice *w* o destino da mesma:
    - Arcos de avanço
    - Arcos de retorno

Busca em Profundidade

- Classificação de arestas
  - Ao explorar um grafo G direcionado usando a DFS, podemos categorizar as arestas.
  - Sejam o vértice *v* a origem da aresta e o vértice *w* o destino da mesma:
    - Arcos de avanço
    - Arcos de retorno
    - Arcos de cruzamento

Busca em Profundidade

Busca em Profundidade

- Classificação de arestas

Busca em Profundidade

- Classificação de arestas
  - Arcos de avanço

Busca em Profundidade

- Classificação de arestas
  - Arcos de avanço
    - Caso *w* seja descendente de *v* na floresta.

Busca em Profundidade

- Classificação de arestas
  - Arcos de avanço
    - Caso *w* seja descendente de *v* na floresta.
  - Arcos de retorno

Busca em Profundidade

- Classificação de arestas
  - Arcos de avanço
    - Caso *w* seja descendente de *v* na floresta.
  - Arcos de retorno
    - Caso *v* seja descendente de *w* na floresta.

Busca em Profundidade

- Classificação de arestas
  - Arcos de avanço
    - Caso *w* seja descendente de *v* na floresta.
  - Arcos de retorno
    - Caso *v* seja descendente de *w* na floresta.
    - Ou, *w* é ancestral de *v*.

Busca em Profundidade

- Classificação de arestas
  - Arcos de avanço
    - Caso *w* seja descendente de *v* na floresta.
  - Arcos de retorno
    - Caso *v* seja descendente de *w* na floresta.
    - Ou, *w* é ancestral de *v*.
  - Arcos de cruzamento (cruzado)

Busca em Profundidade

- Classificação de arestas
  - Arcos de avanço
    - Caso *w* seja descendente de *v* na floresta.
  - Arcos de retorno
    - Caso *v* seja descendente de *w* na floresta.
    - Ou, *w* é ancestral de *v*.
  - Arcos de cruzamento (cruzado)
    - Caso *w* não seja descendente de *v* e *v* não seja descendente de *w*.

Busca em Profundidade

- Classificação de arestas
  - Arcos de avanço
    - Caso *w* seja descendente de *v* na floresta.
  - Arcos de retorno
    - Caso *v* seja descendente de *w* na floresta.
    - Ou, *w* é ancestral de *v*.
  - Arcos de cruzamento (cruzado)
    - Caso *w* não seja descendente de *v* e *v* não seja descendente de *w*.
    - Ou, *w* é primo de *v*.

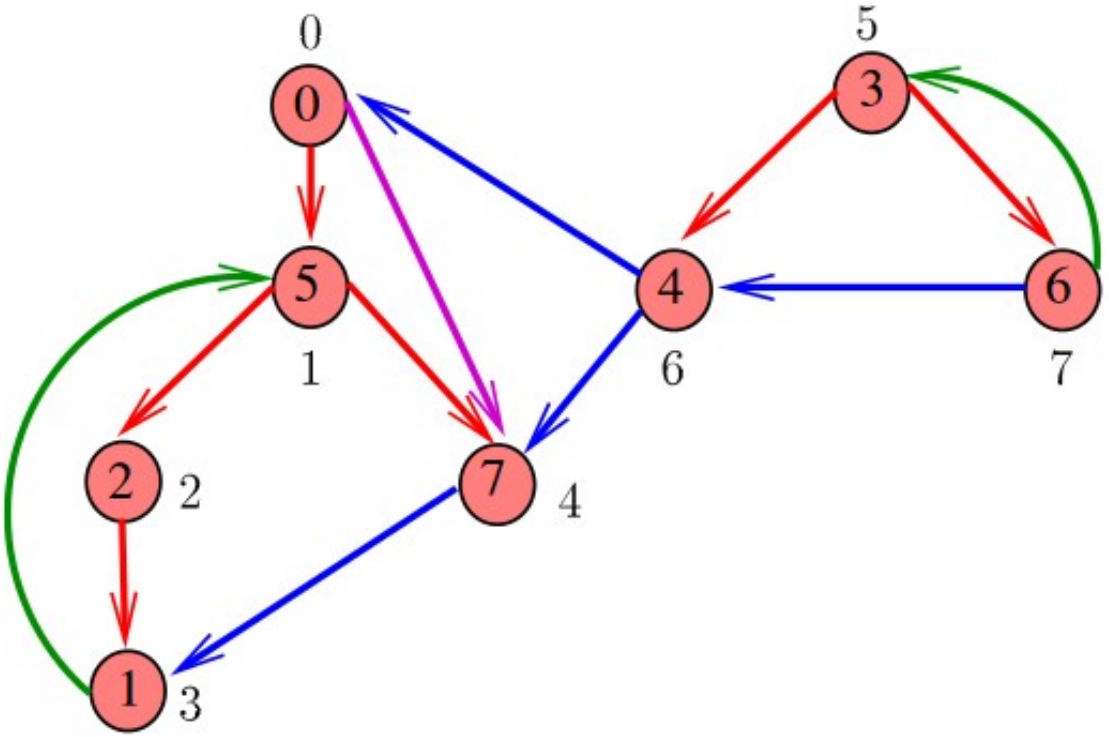
Busca em Profundidade

Busca em Profundidade

- Exemplo

Busca em Profundidade

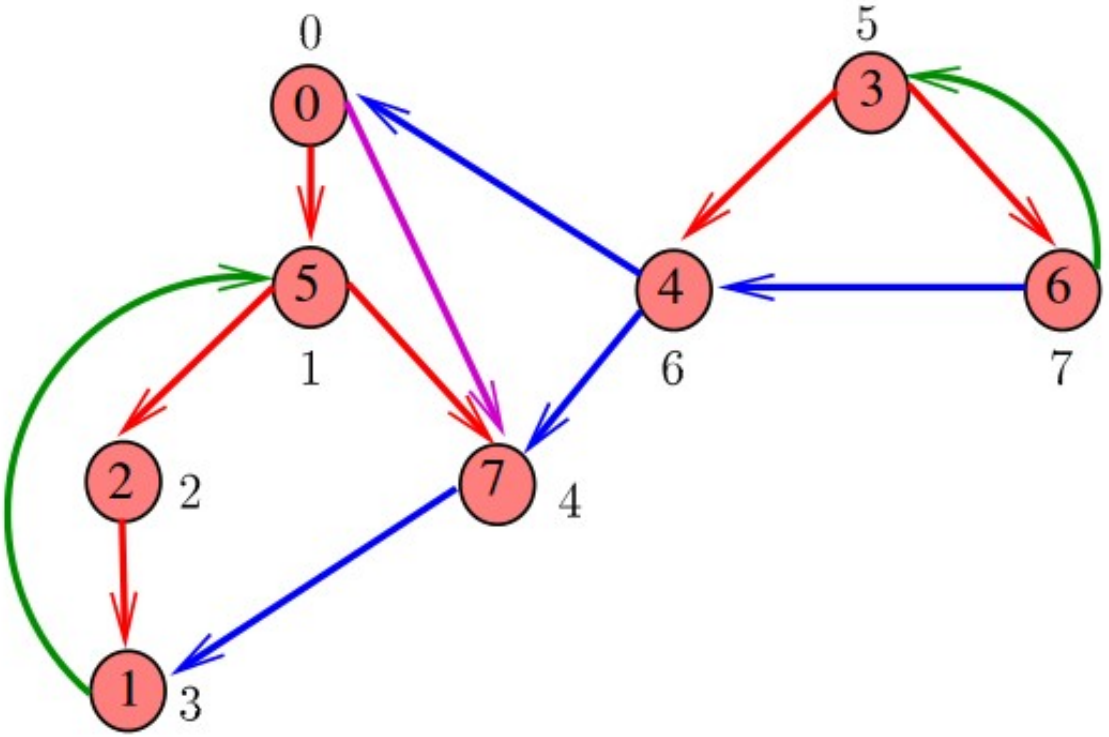
- Exemplo



Busca em Profundidade

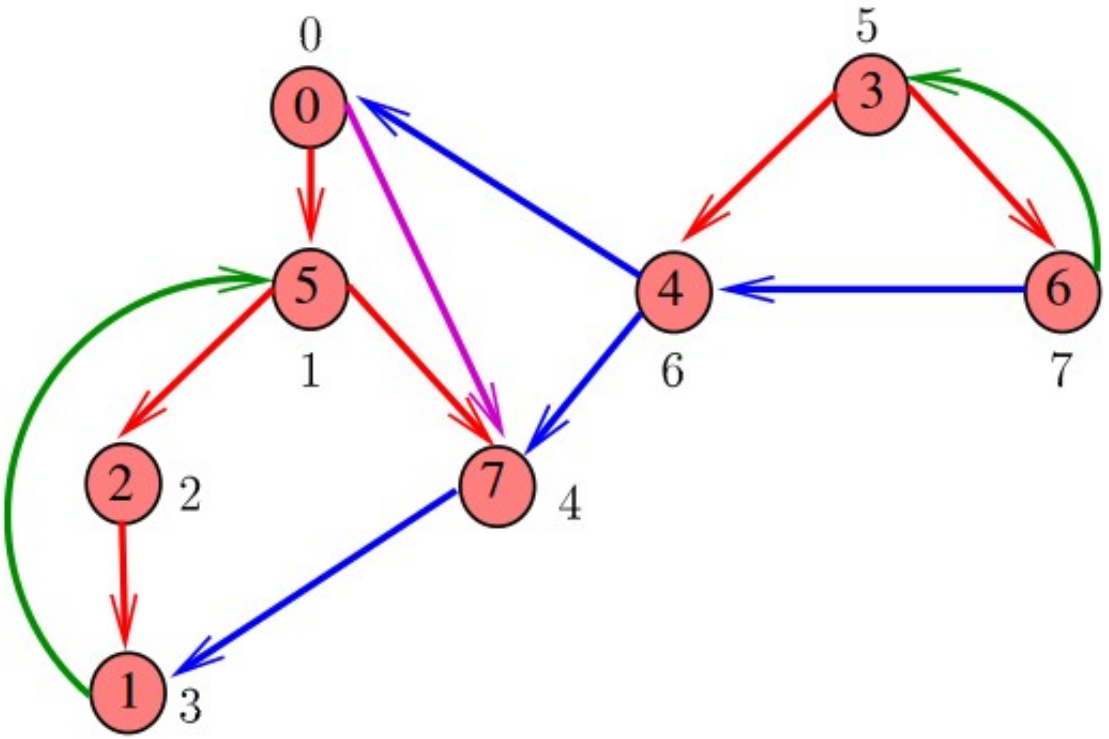
- Exemplo

Arcos da floresta são vermelhos.



Busca em Profundidade

- Exemplo

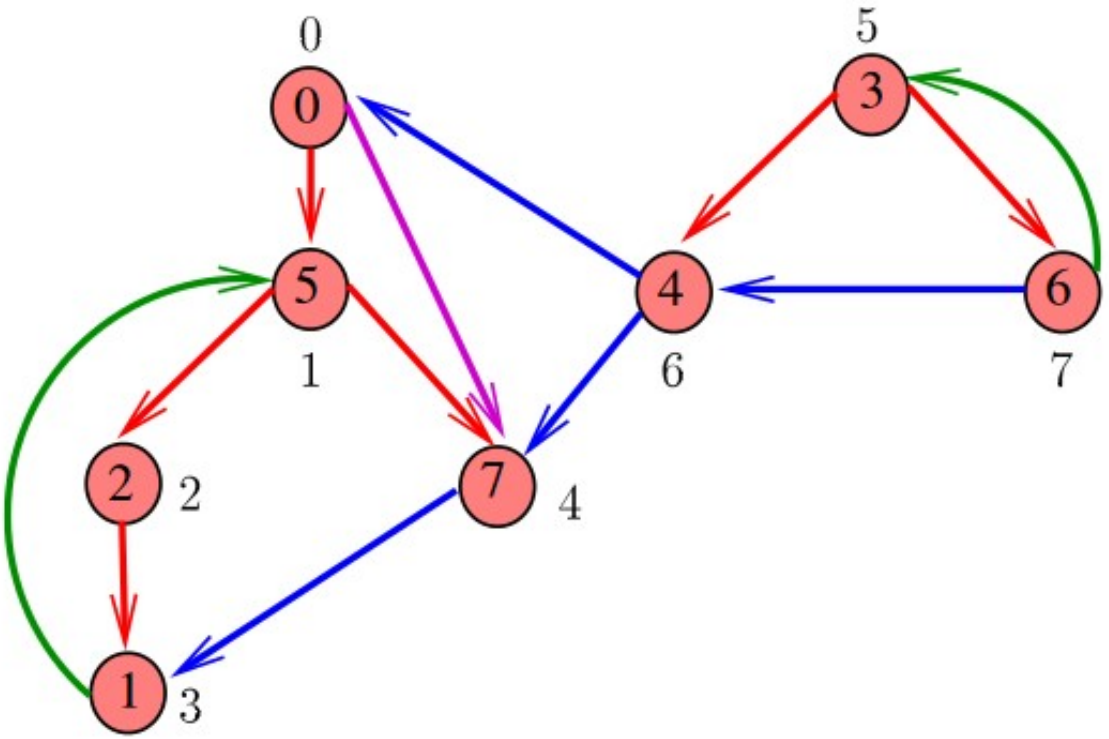


Arcos da floresta são vermelhos.

Arcos de retorno são verdes.

Busca em Profundidade

- Exemplo



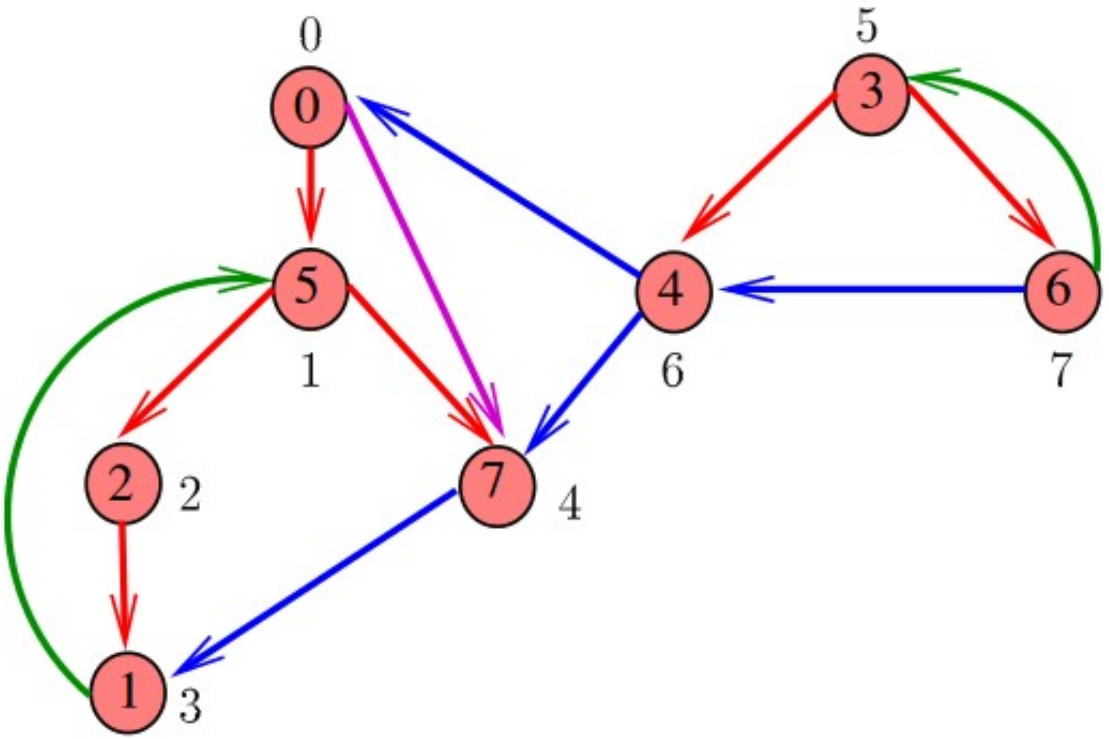
Arcos da floresta são vermelhos.

Arcos de retorno são verdes.

Arcos de avanço são roxos.

Busca em Profundidade

- Exemplo



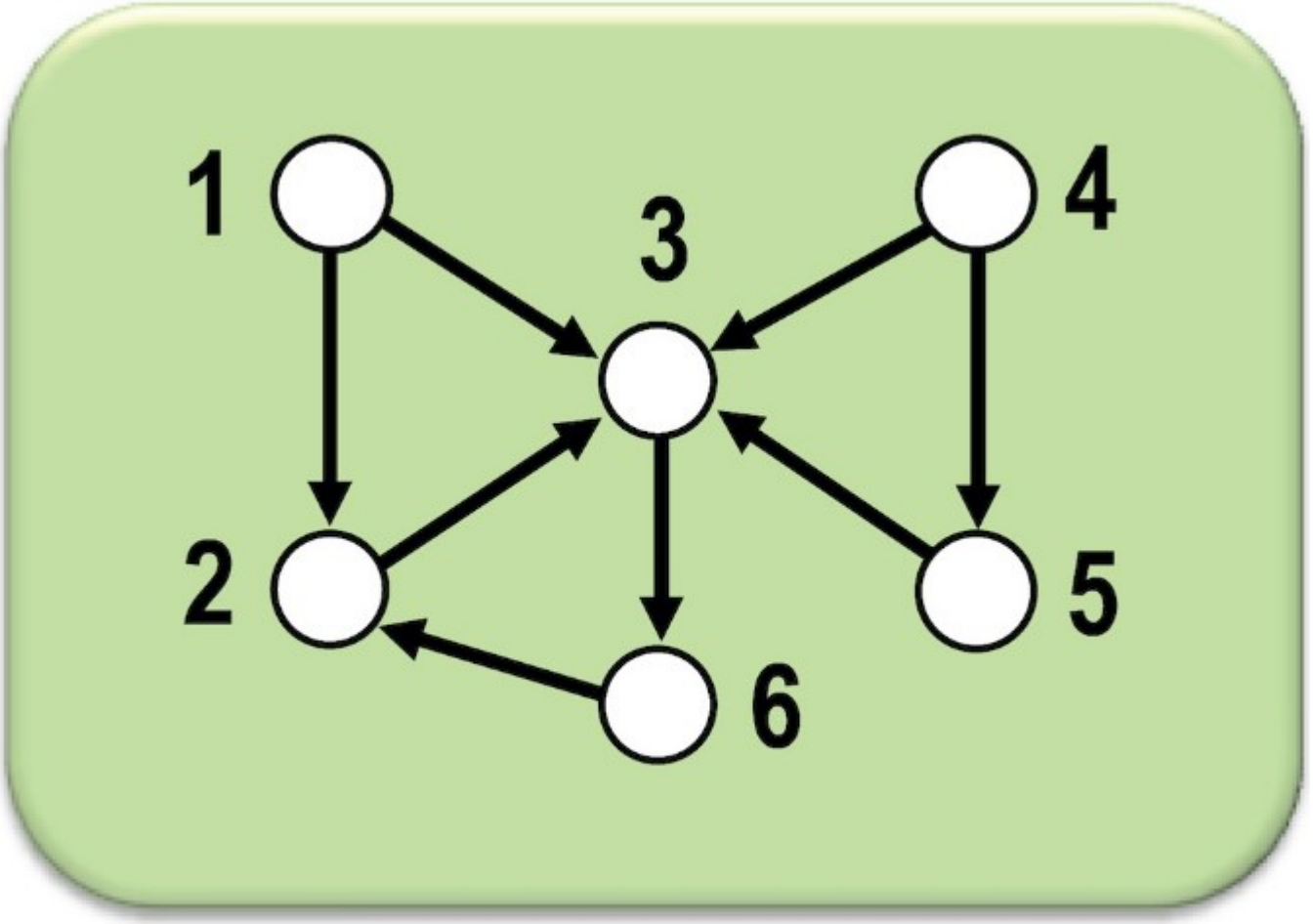
Arcos da floresta são vermelhos.

Arcos de retorno são verdes.

Arcos de avanço são roxos.

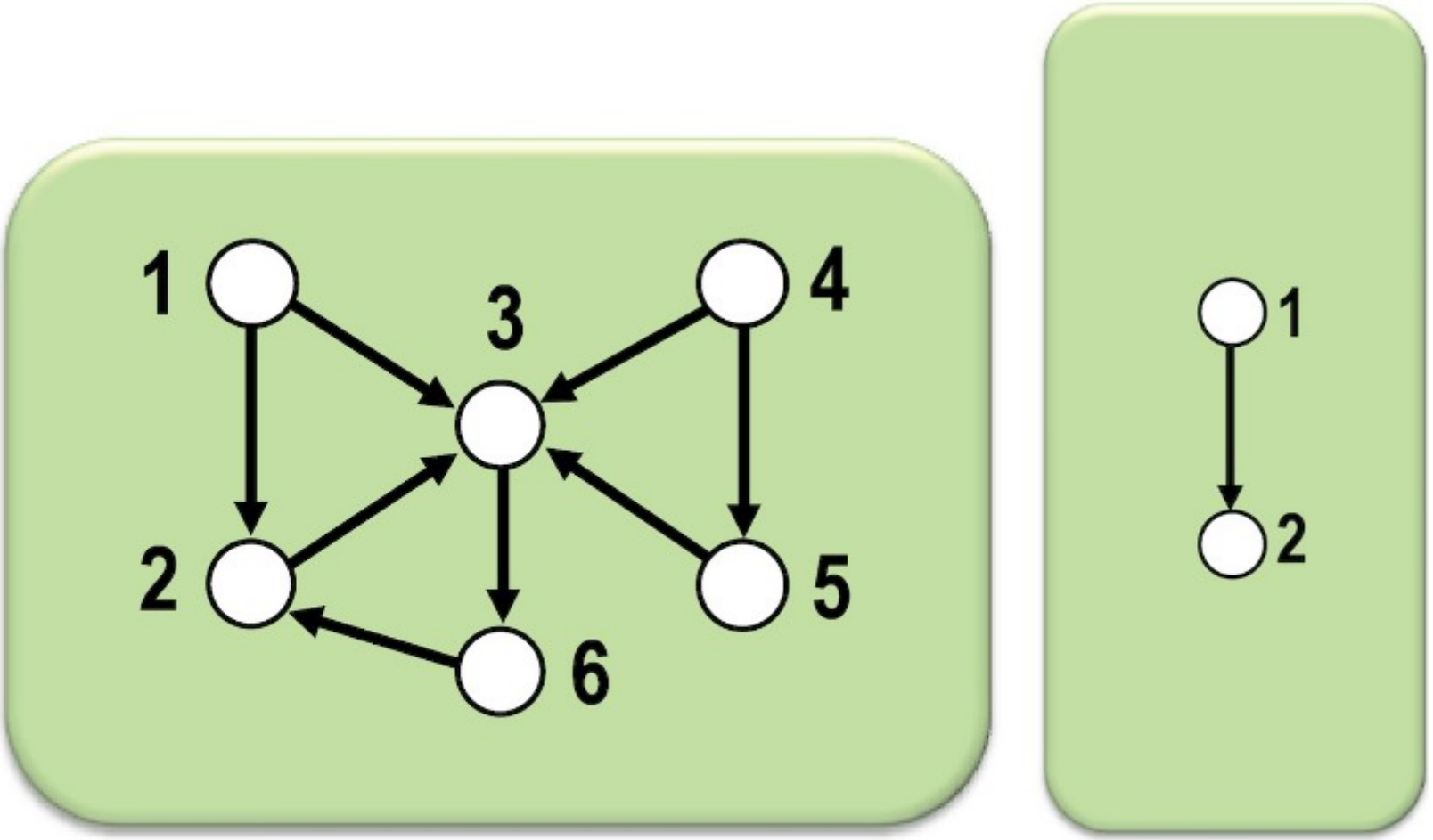
Arcos cruzados são azuis.

DFS - Exemplo



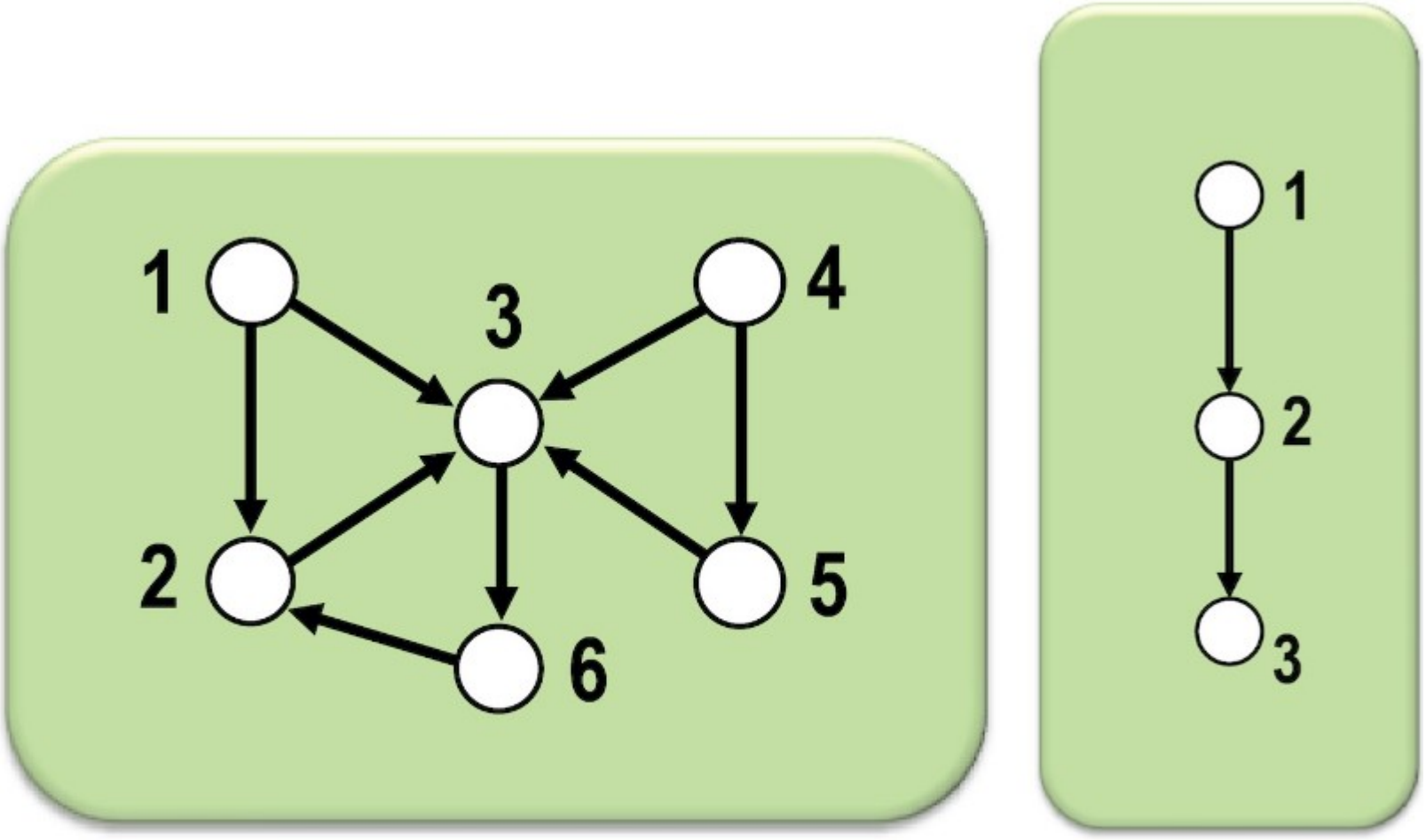
Grafo de exemplo.

DFS - Exemplo



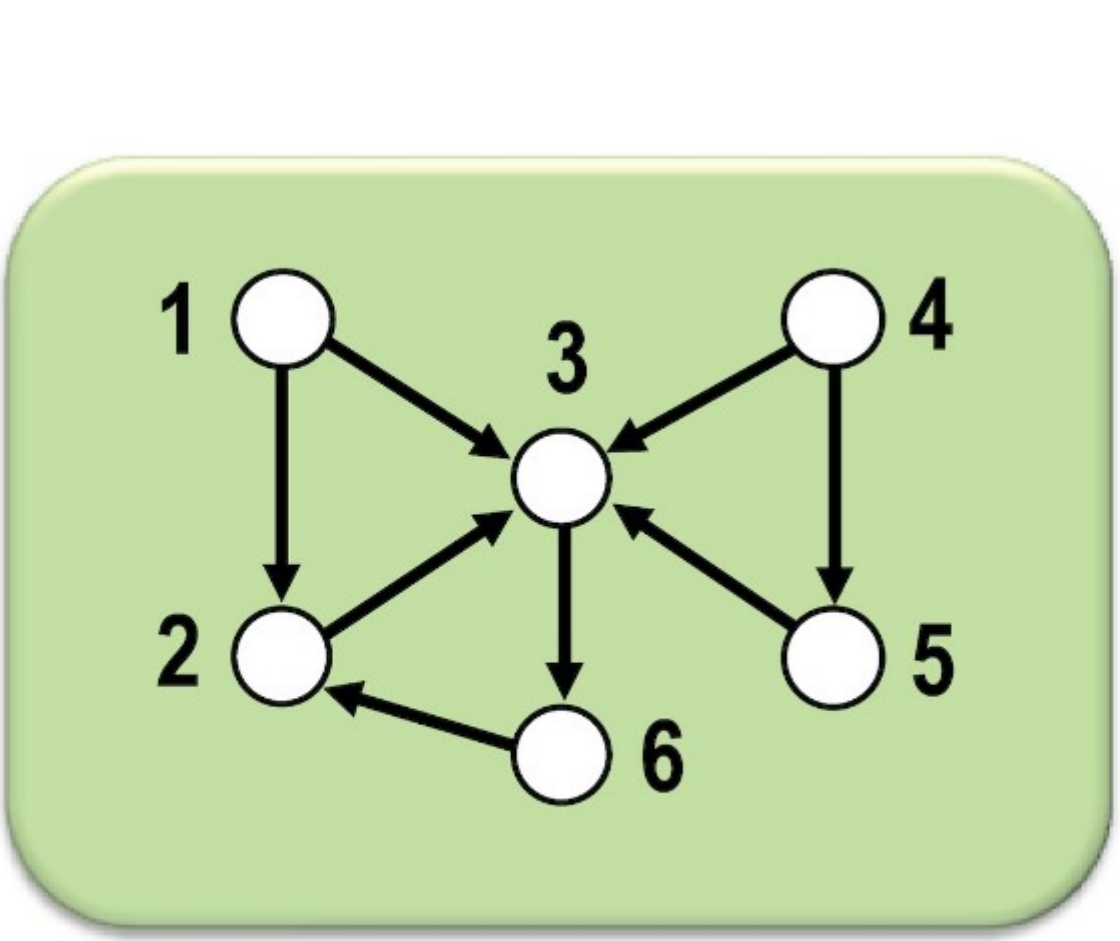
(1) Arco (1,2).

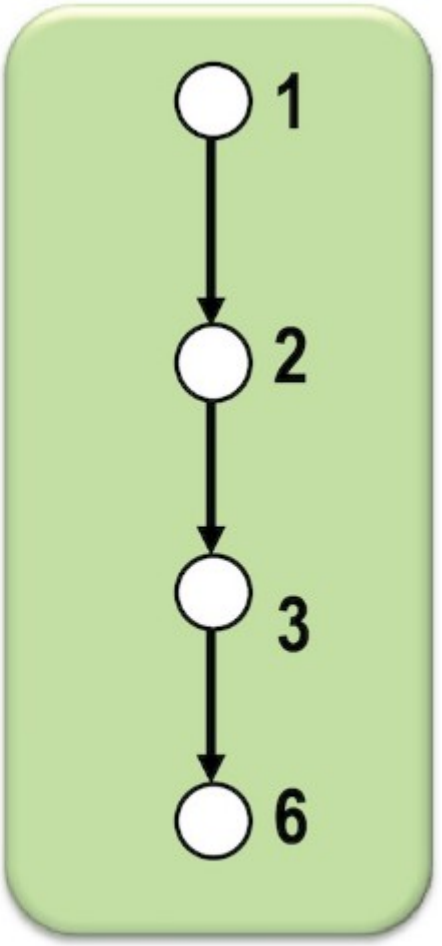
DFS - Exemplo



(2) Arco (2,3).

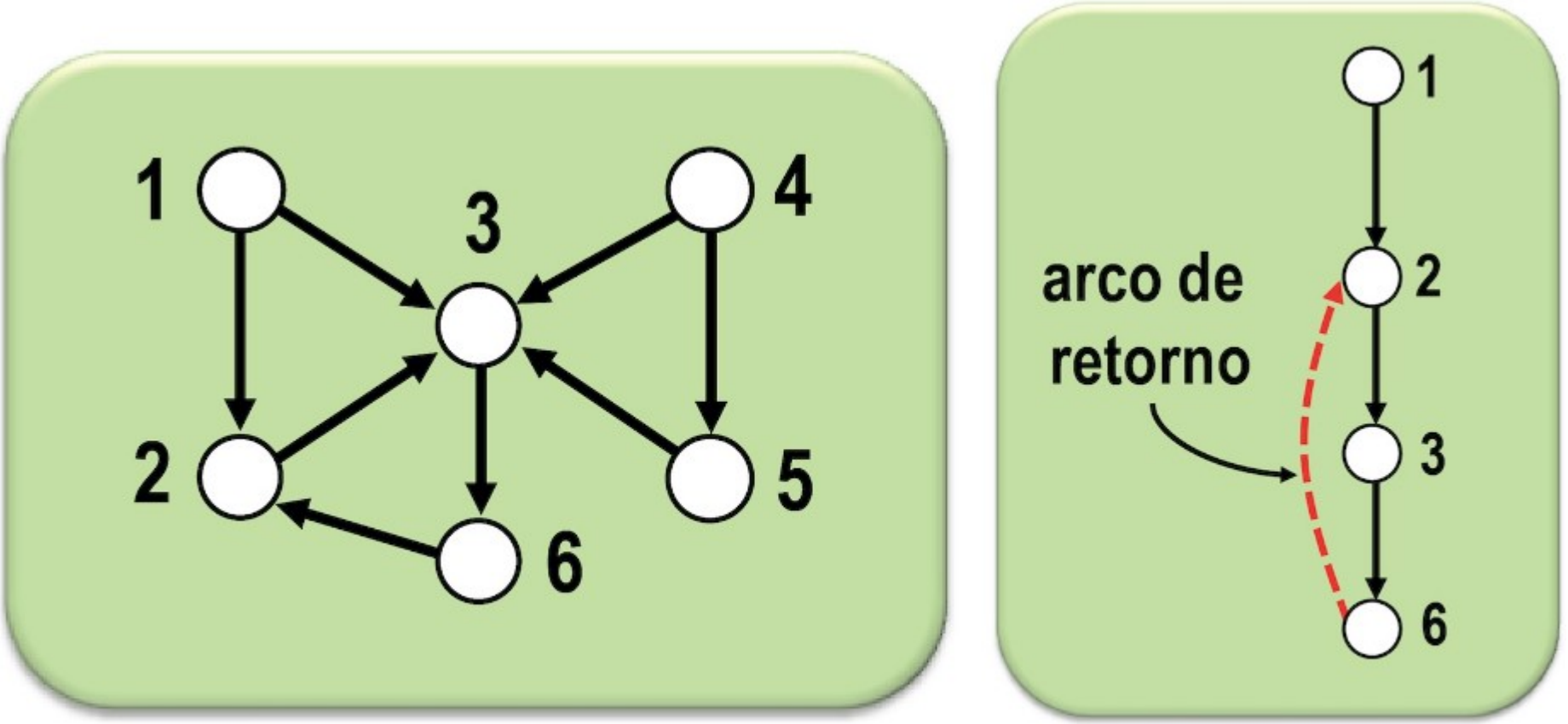
DFS - Exemplo





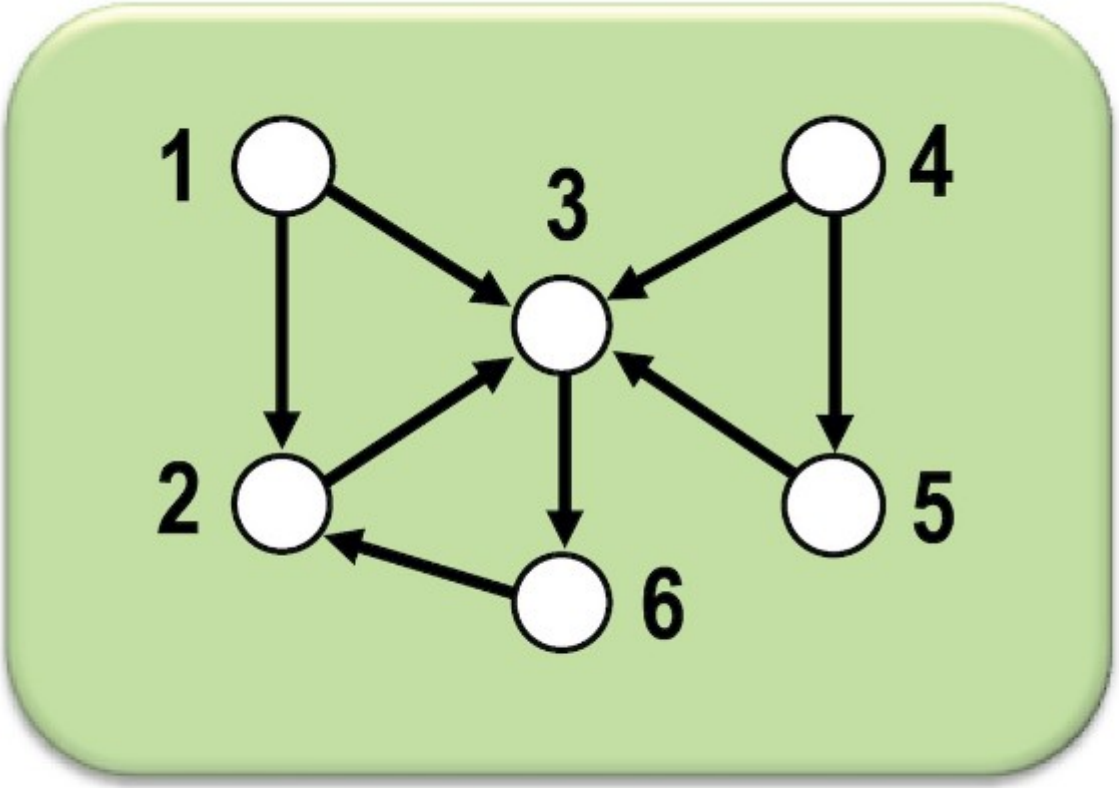
(3) Arco (3, 6).

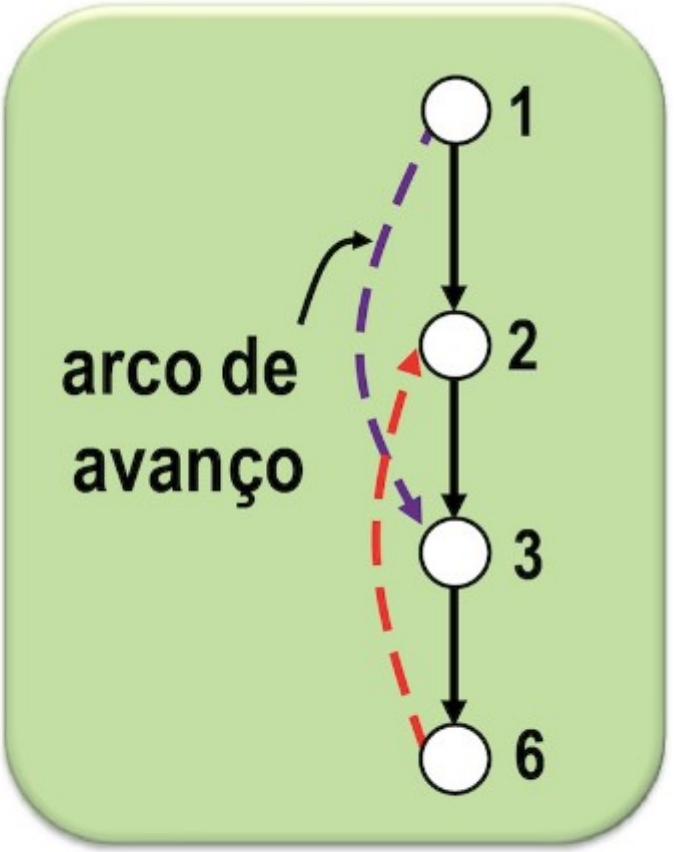
DFS - Exemplo



(4) Arco (6, 2).

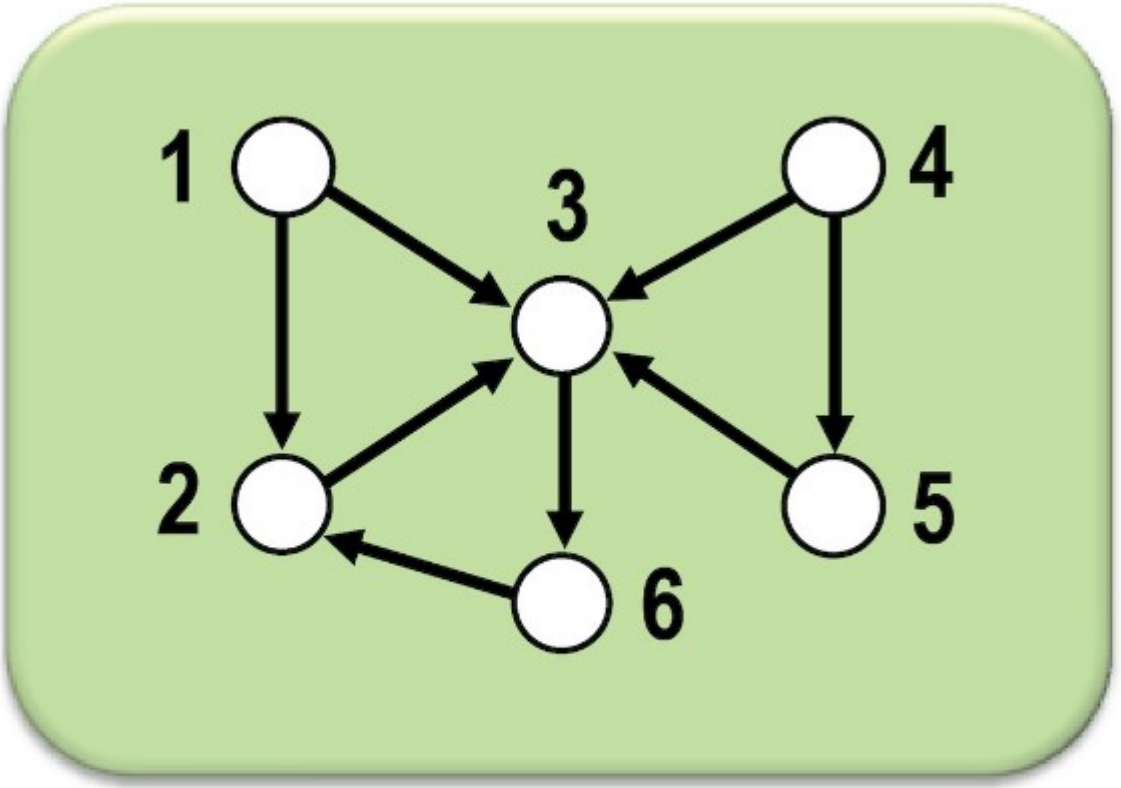
DFS - Exemplo

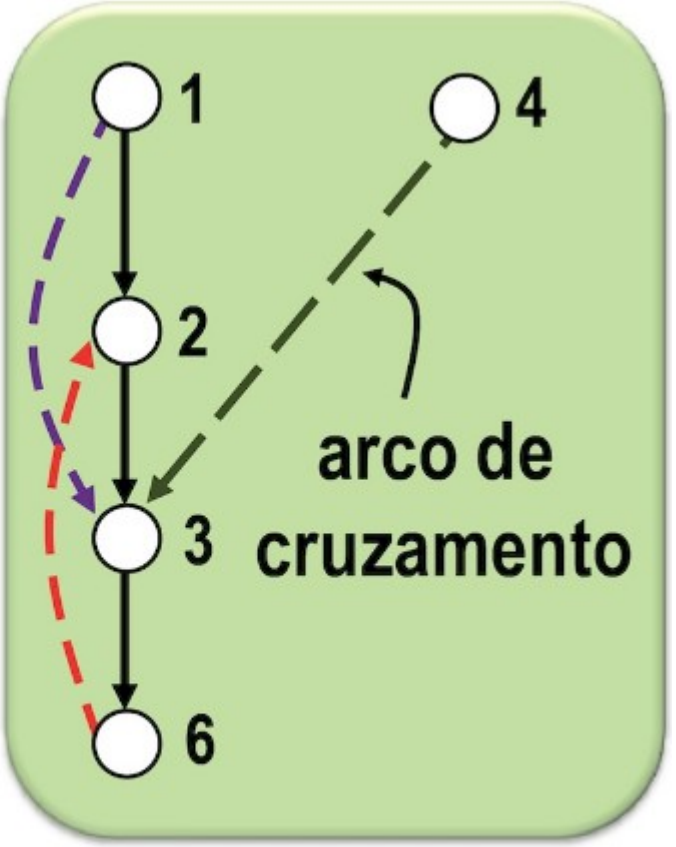




(5) Arco (1, 3).

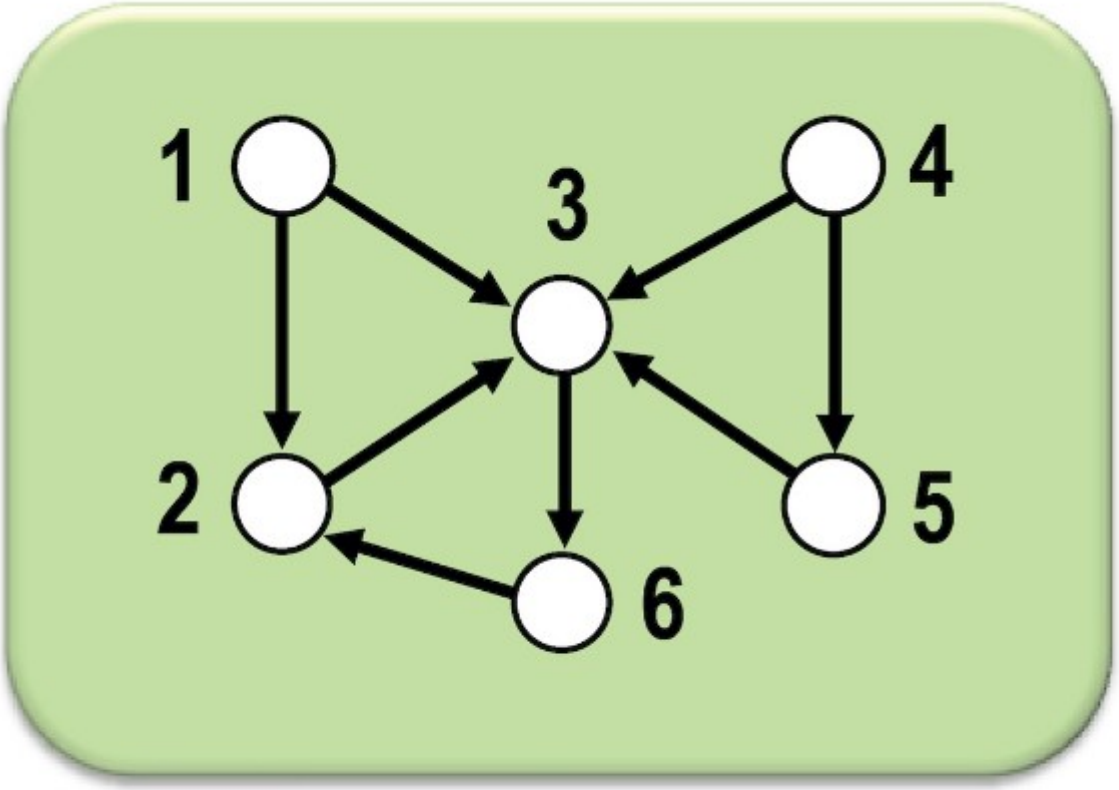
DFS - Exemplo

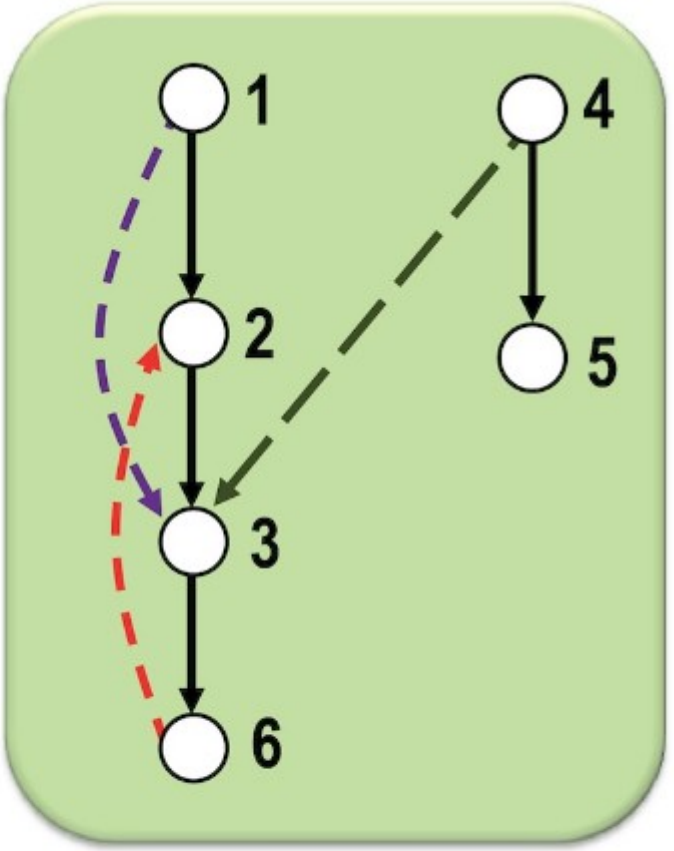




(6) Arco (4, 3).

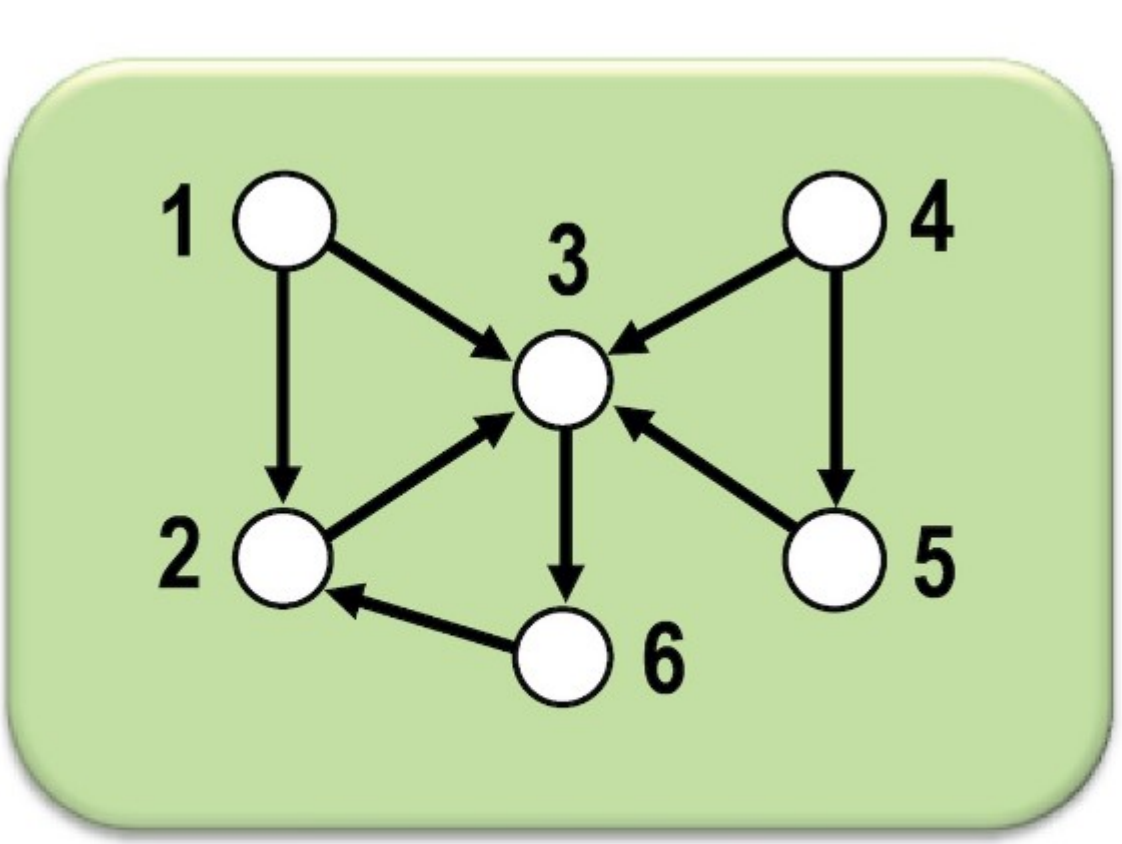
DFS - Exemplo

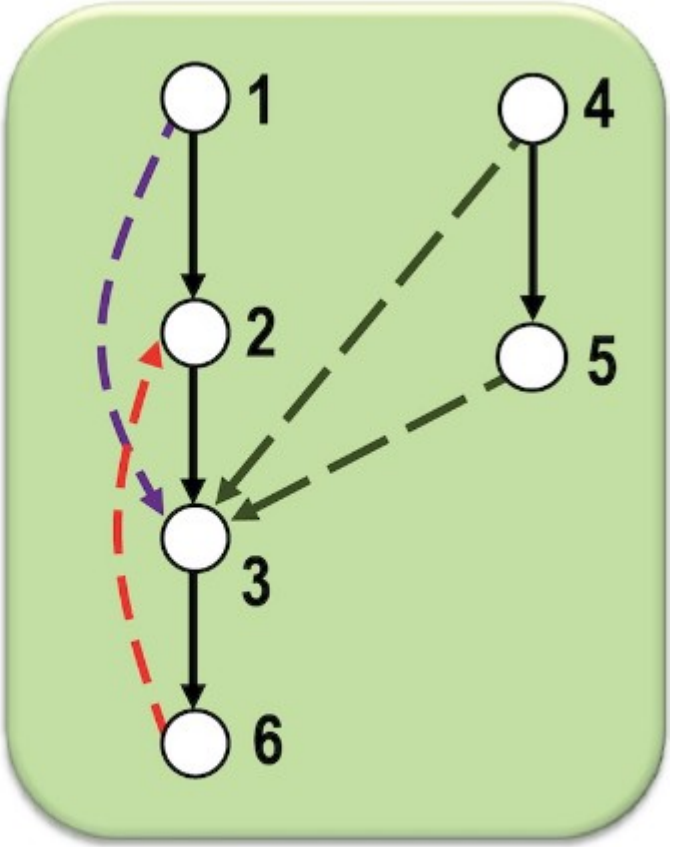




(7) Arco (4, 5).

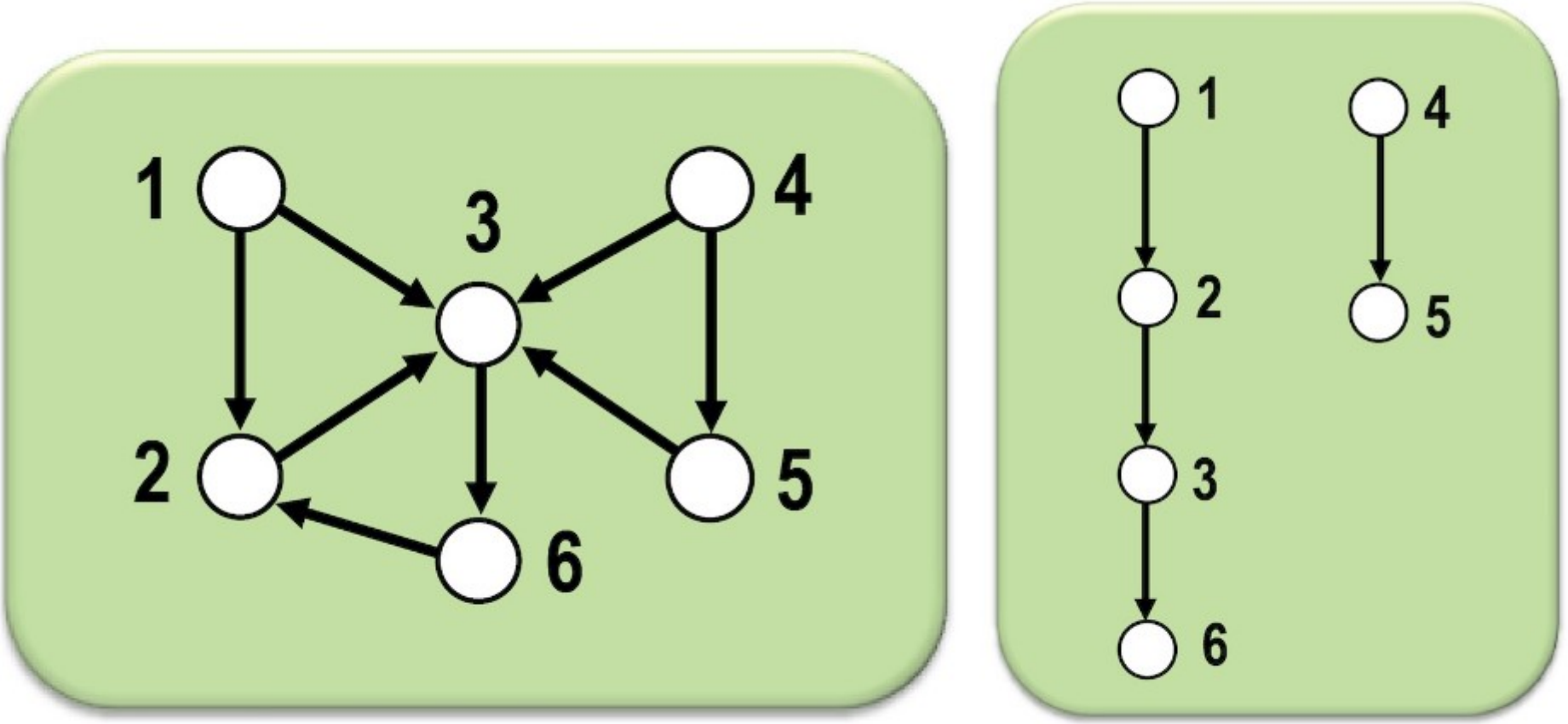
DFS - Exemplo





(8) Arco (5, 3).

DFS - Exemplo



Grafo original e respectiva floresta de profundidade.

Exemplo

- https://www.cs.usfca.edu/~galles/visualization/DFS.html

Busca em Largura

Busca em Largura

Busca em Largura

- Características

Busca em Largura

- Características
  - A busca em largura visita todos os vértices de um grafo, usando como critério o vértice visitado menos recentemente e cuja vizinhança ainda não foi explorada.

Busca em Largura

- Características
  - A busca em largura visita todos os vértices de um grafo, usando como critério o vértice visitado menos recentemente e cuja vizinhança ainda não foi explorada.
  - Característica principal: utiliza uma fila para guiar a busca.

Busca em Largura

Busca em Largura

- Atuação em camadas

Busca em Largura

- Atuação em camadas
  - Inicialmente são considerados os vértices com distância 0 (zero) do vértice inicial.

Busca em Largura

- Atuação em camadas
  - Inicialmente são considerados os vértices com distância 0 (zero) do vértice inicial.
  - Na iteração 1 são visitados os vértices com distância 1; prosseguindo, de modo genérico, na iteração d será adicionada uma camada com todos os vértices com distância d do vértice inicial.

Busca em Largura

- Atuação em camadas
  - Inicialmente são considerados os vértices com distância 0 (zero) do vértice inicial.
  - Na iteração 1 são visitados os vértices com distância 1; prosseguindo, de modo genérico, na iteração d será adicionada uma camada com todos os vértices com distância d do vértice inicial.
  - Cada novo vértice visitado é adicionado no final de uma fila Q.

Busca em Largura

- Atuação em camadas
  - Inicialmente são considerados os vértices com distância 0 (zero) do vértice inicial.
  - Na iteração 1 são visitados os vértices com distância 1; prosseguindo, de modo genérico, na iteração d será adicionada uma camada com todos os vértices com distância d do vértice inicial.
  - Cada novo vértice visitado é adicionado no final de uma fila Q.
  - Cada vértice da fila é removido depois que toda a vizinhança for visitada.

Busca em Largura

- Atuação em camadas
  - Inicialmente são considerados os vértices com distância 0 (zero) do vértice inicial.
  - Na iteração 1 são visitados os vértices com distância 1; prosseguindo, de modo genérico, na iteração d será adicionada uma camada com todos os vértices com distância d do vértice inicial.
  - Cada novo vértice visitado é adicionado no final de uma fila Q.
  - Cada vértice da fila é removido depois que toda a vizinhança for visitada.
  - Cada vértice da fila é removido depois que toda a vizinhança for visitada.

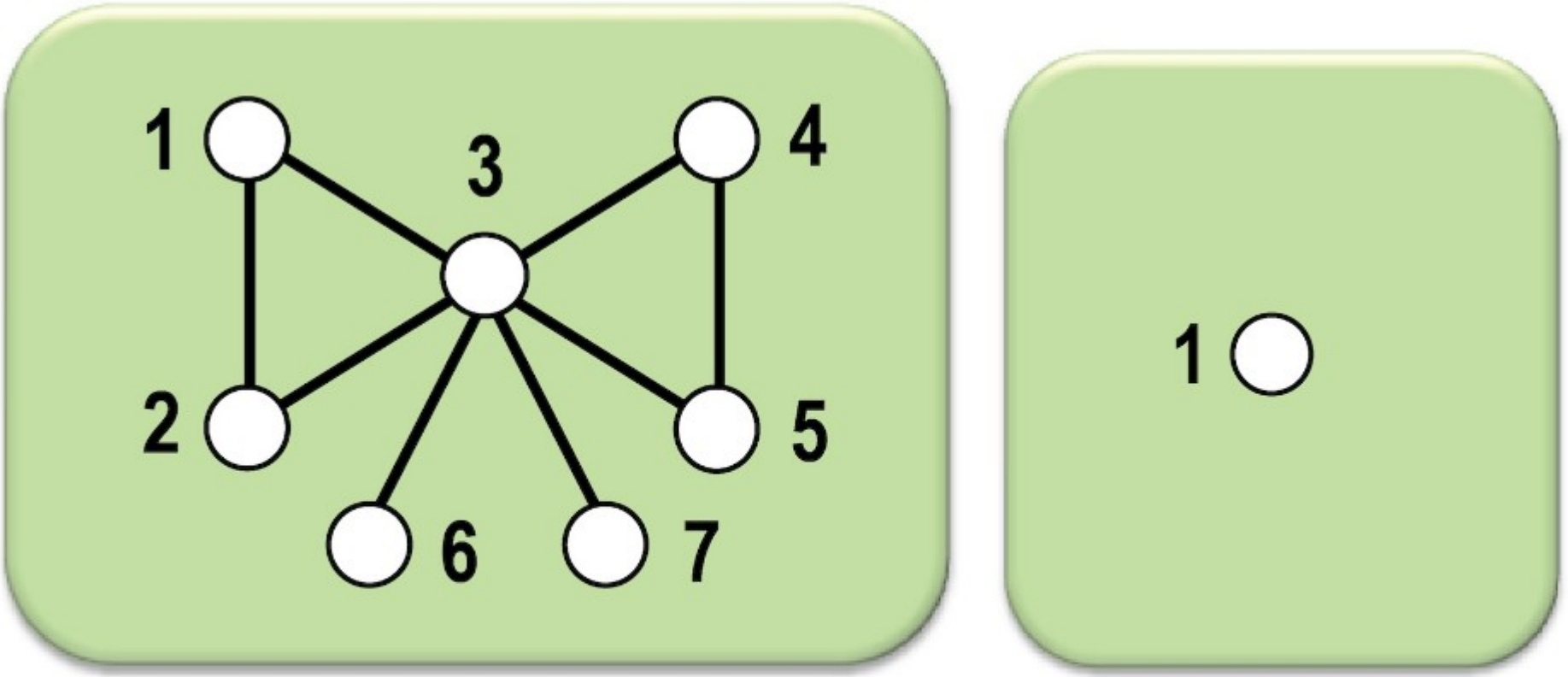
Busca em Largura

- Atuação em camadas
  - Inicialmente são considerados os vértices com distância 0 (zero) do vértice inicial.
  - Na iteração 1 são visitados os vértices com distância 1; prosseguindo, de modo genérico, na iteração d será adicionada uma camada com todos os vértices com distância d do vértice inicial.
  - Cada novo vértice visitado é adicionado no final de uma fila Q.
  - Cada vértice da fila é removido depois que toda a vizinhança for visitada.
  - Cada vértice da fila é removido depois que toda a vizinhança for visitada.
  - A busca termina quando a fila se torna vazia.

Busca em Largura - BFS

```
   Entrada: Grafo G=(V, A), vértice inicial v
1  Crie uma fila Q vazia;
2  Marque v como visitado;
3  Insira v em Q;
4  enquanto Q ≠ ∅ faça
5      v ← remove elemento de Q;
6      para todo vértice w vizinho de v faça
7          se w é marcado como não visitado então
8              Visite a aresta {v, w};
9              Insira w em Q;
10             Marque w como visitado;
11         fim
12         senão
13             se {v, w} não foi visitada ainda então
14                 Visite {v, w};
15             fim
16         fim
17     fim
18 fim
```

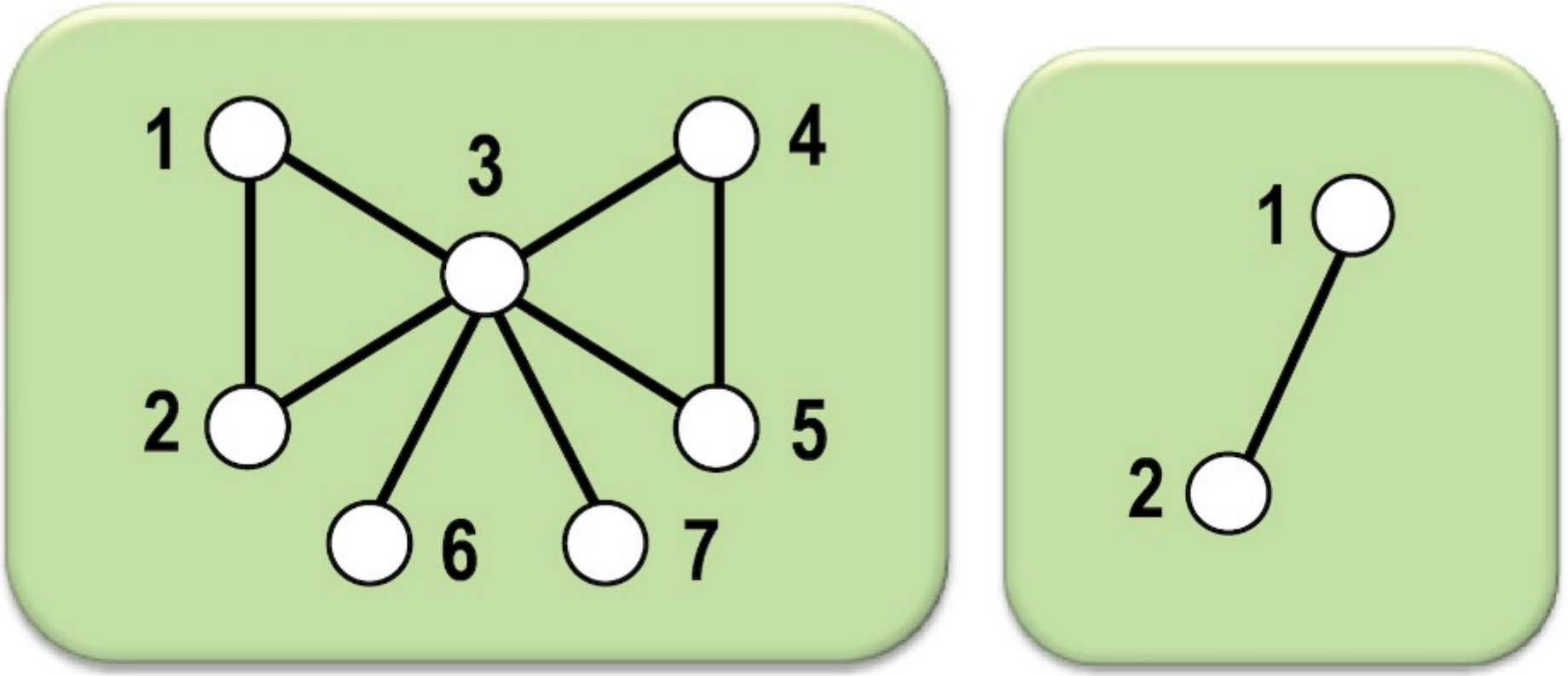
Busca em Largura - BFS



(1) Inclusão de 1

$Q = \{1\}$

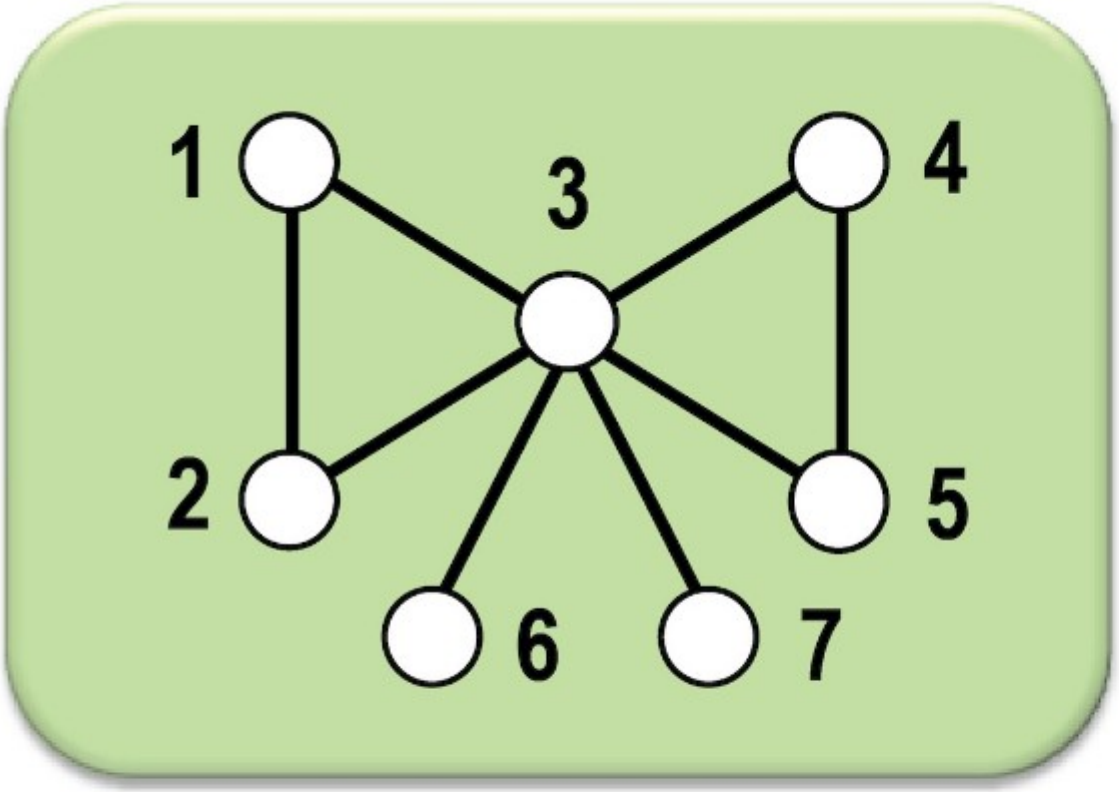
Busca em Largura - BFS

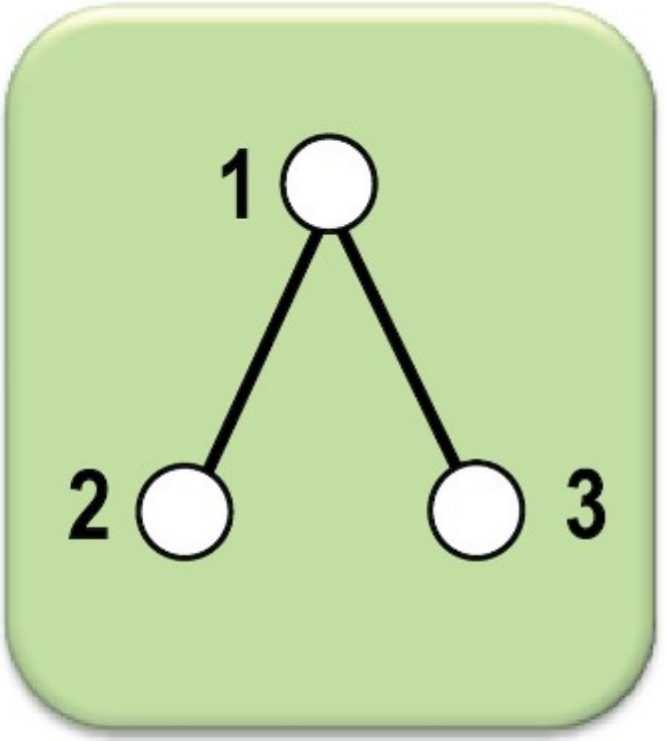


(2) ArestaAresta{1, 2}

$Q = \{2\}$

Busca em Largura - BFS

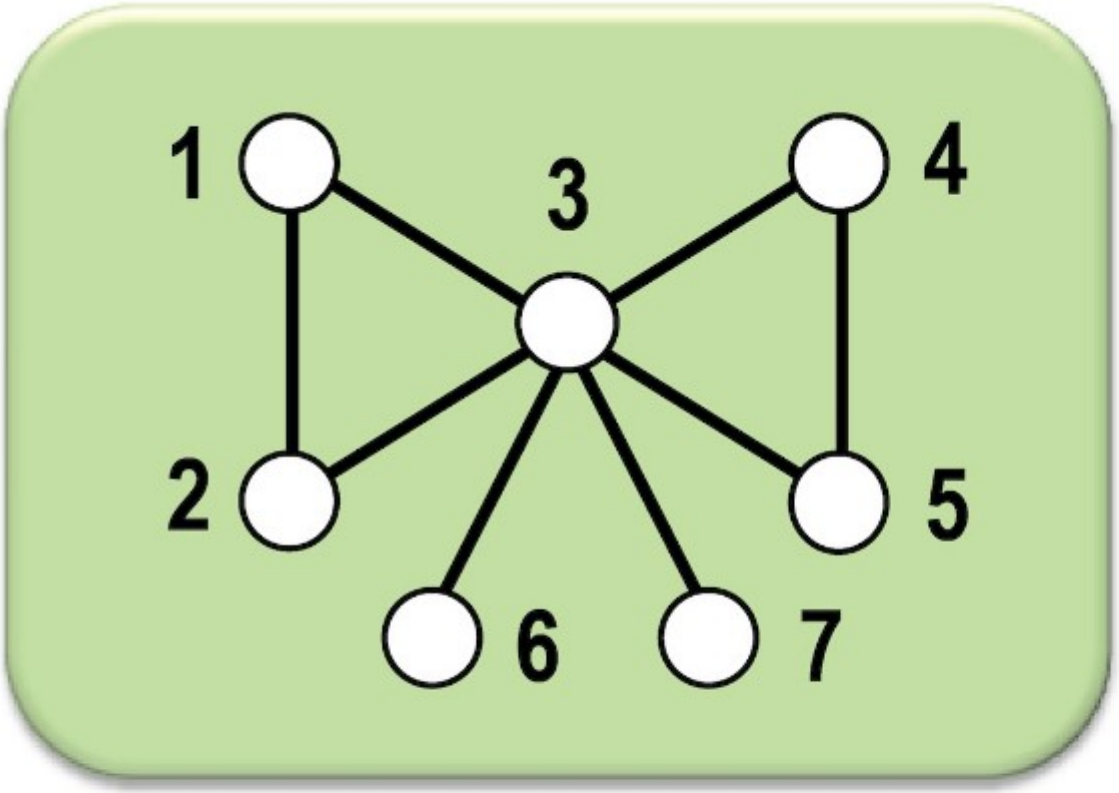


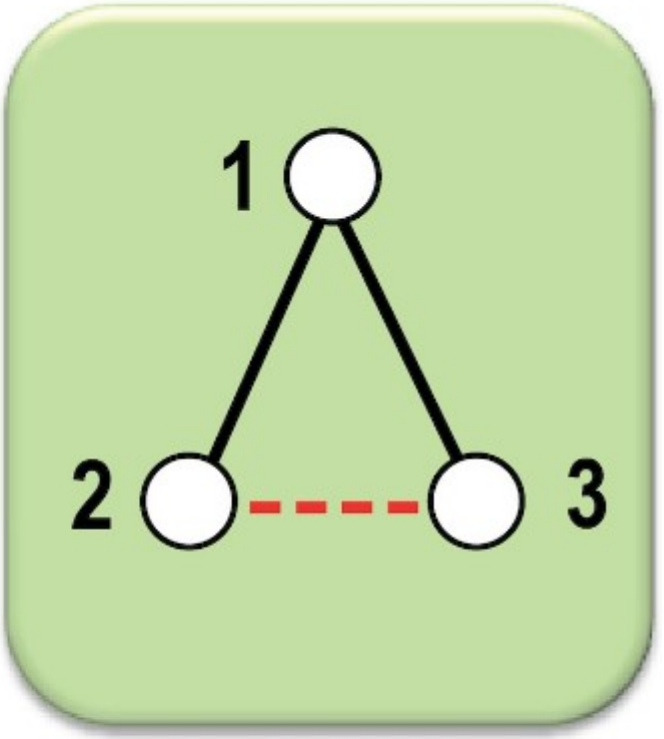


(3) Aresta$\{1, 3\}$

$Q = \{2, 3\}$

Busca em Largura - BFS

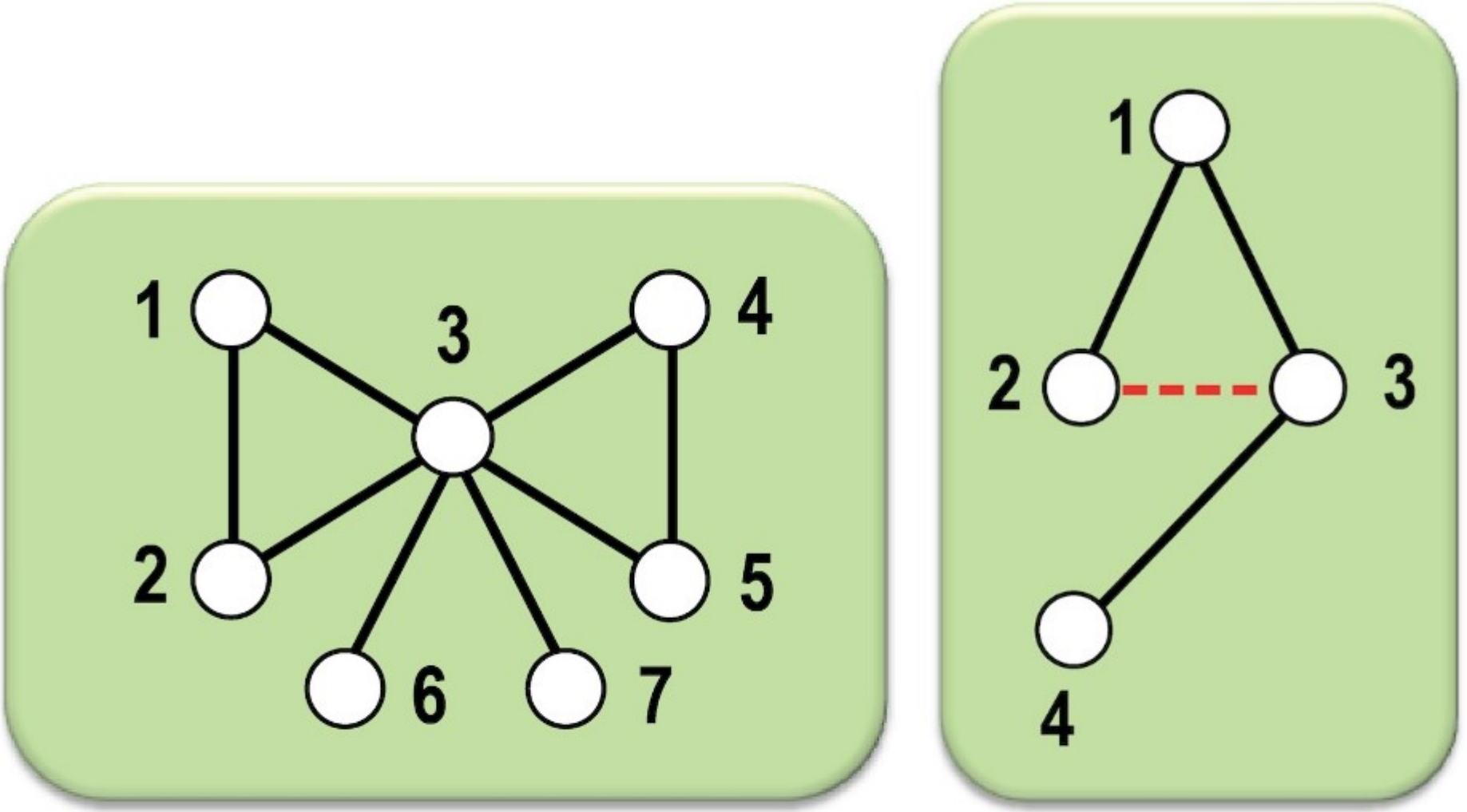




(4) Aresta$\{2, 3\}$

$Q = \{3\}$

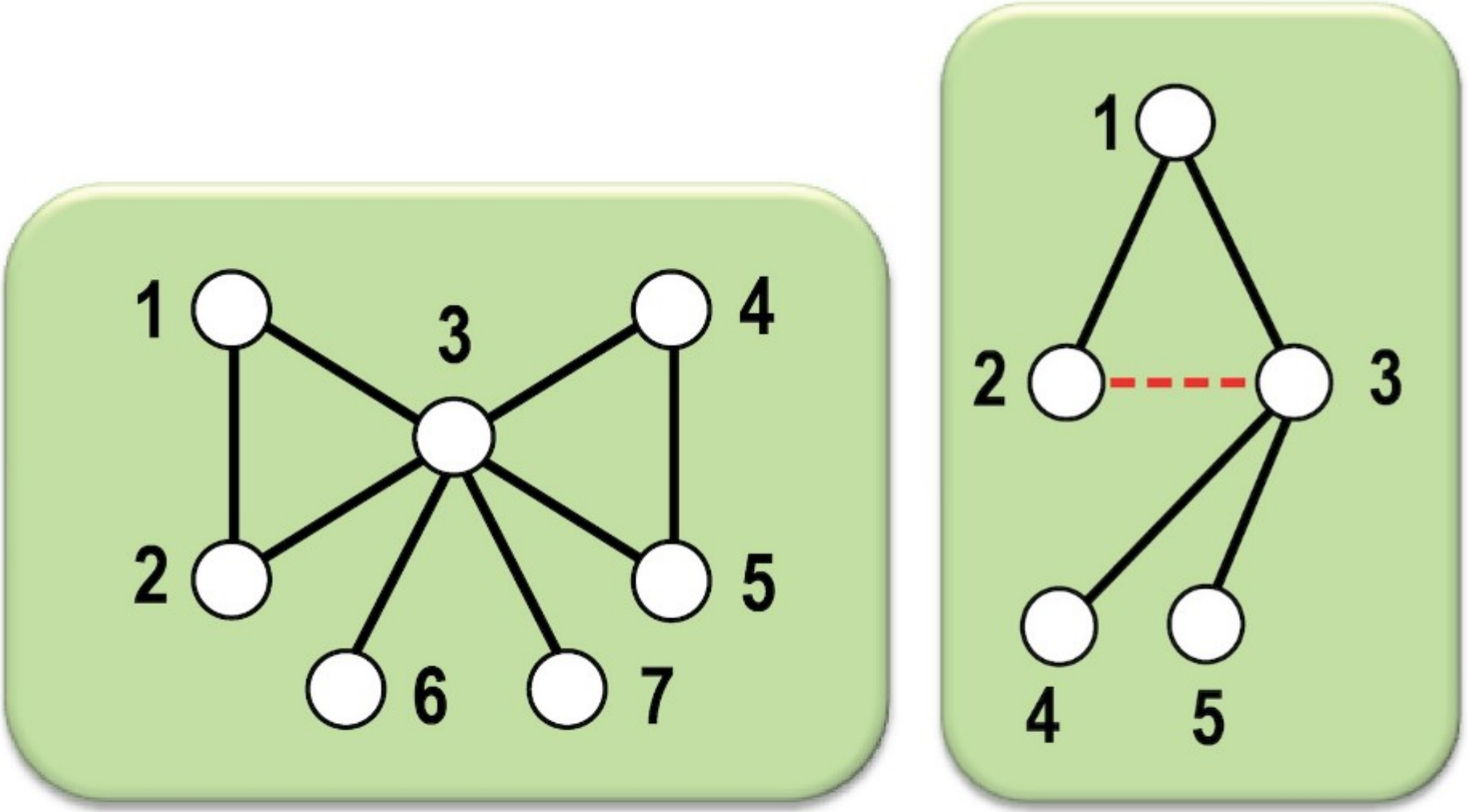
Busca em Largura - BFS



(5) Aresta$\{3, 4\}$

$Q = \{4\}$

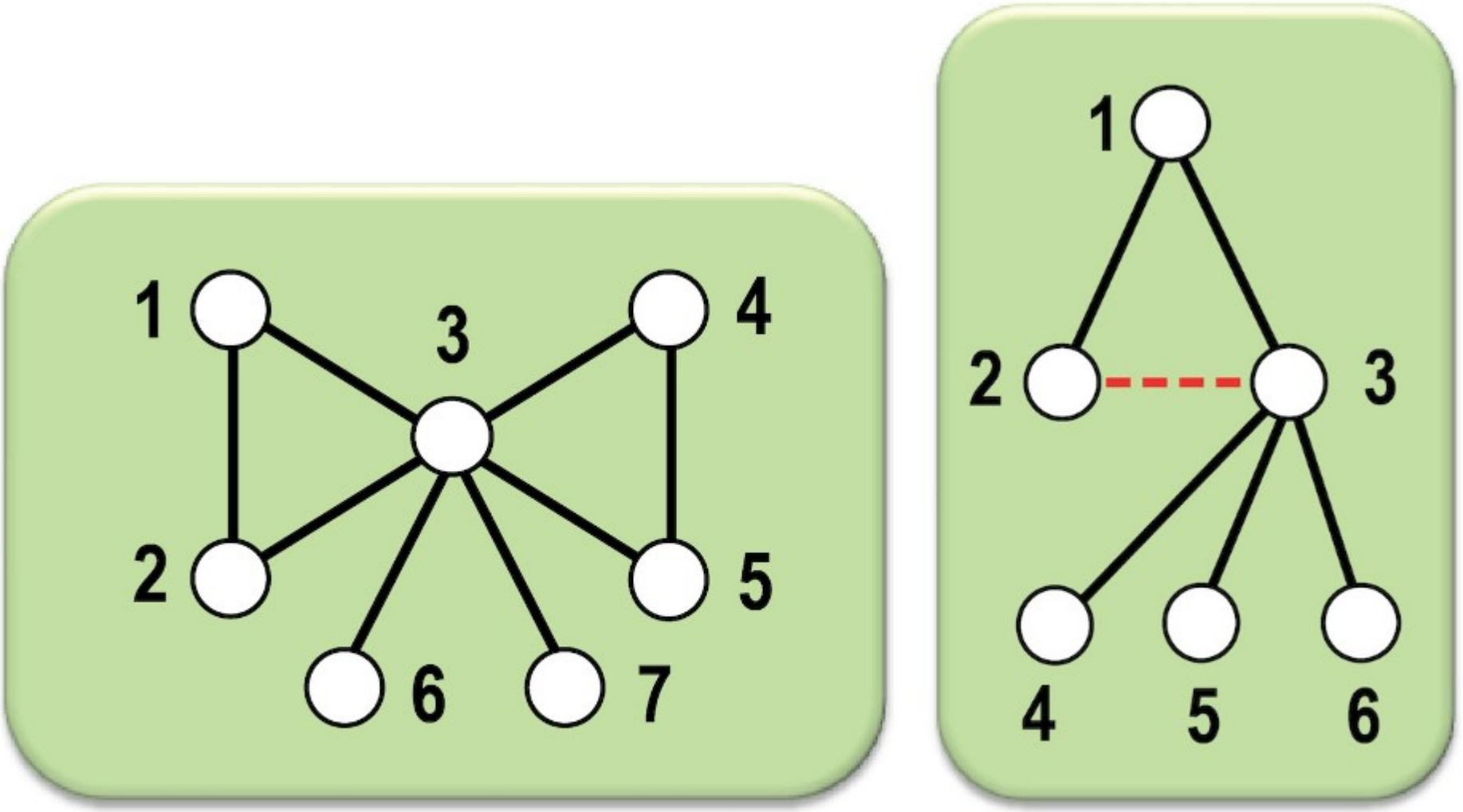
Busca em Largura - BFS



(6) Aresta$\{3, 5\}$

$Q = \{4, 5\}$

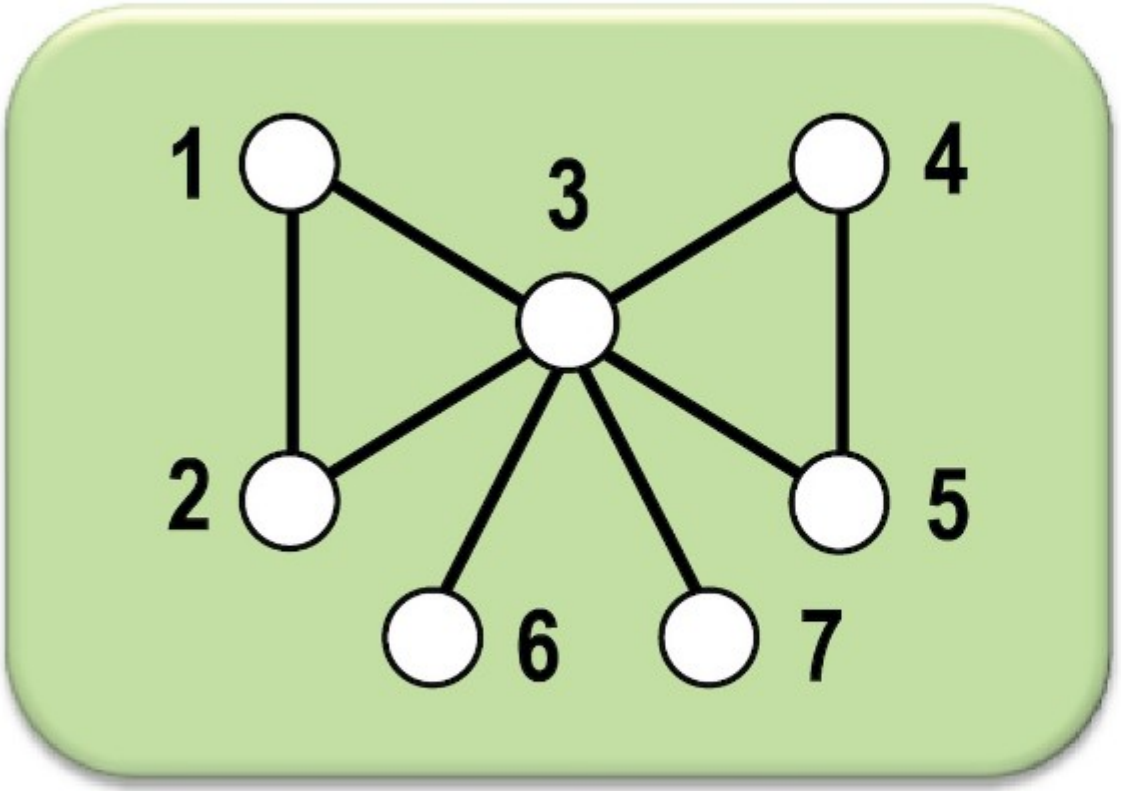
Busca em Largura - BFS

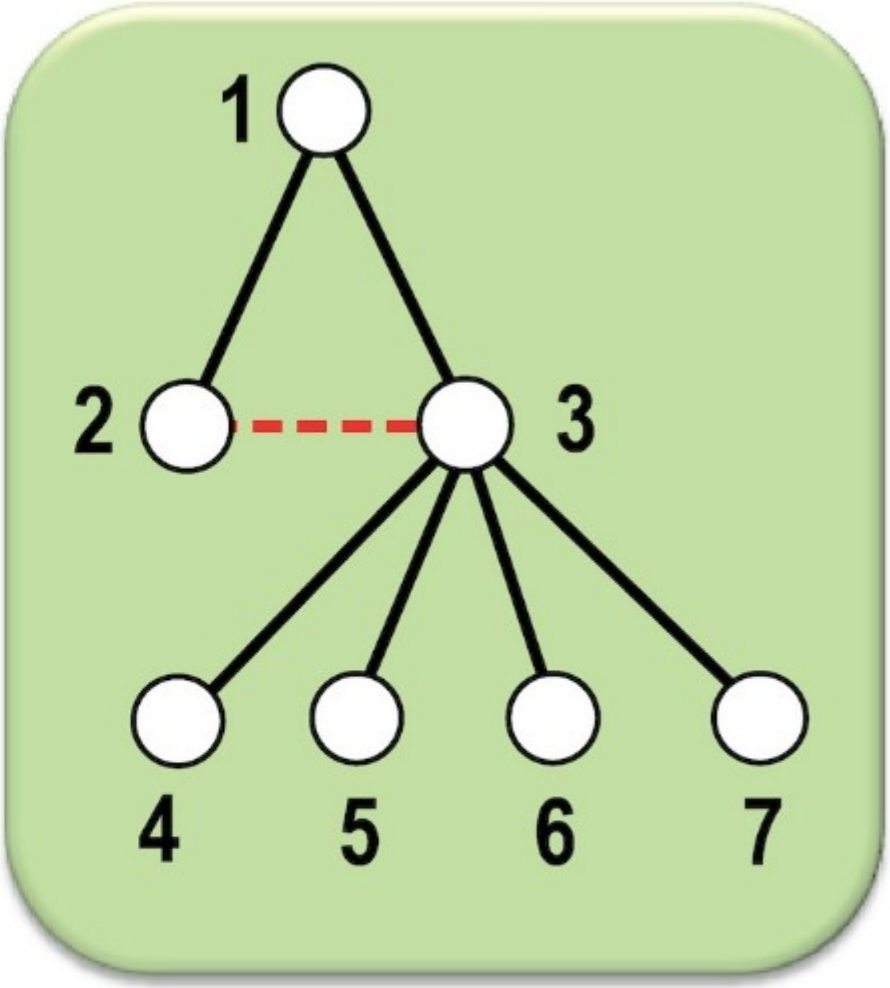


(7) Aresta$\{3, 6\}$

$Q = \{4, 5, 6\}$

Busca em Largura - BFS

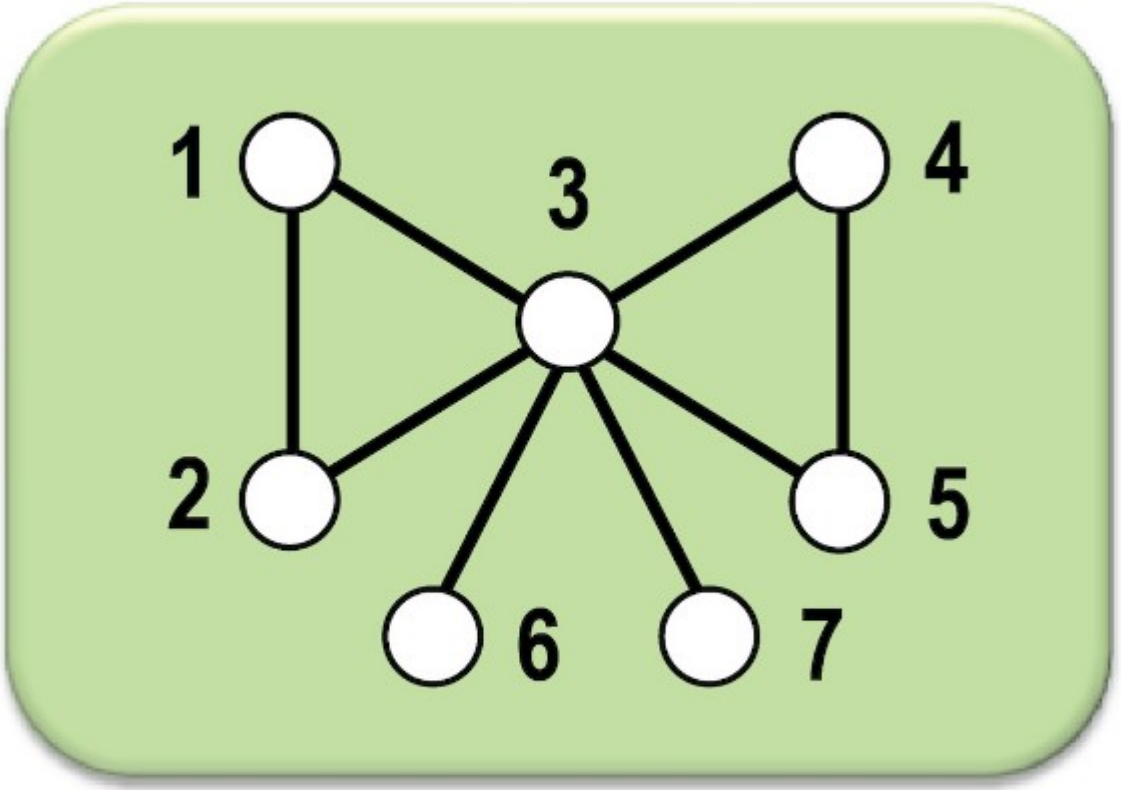


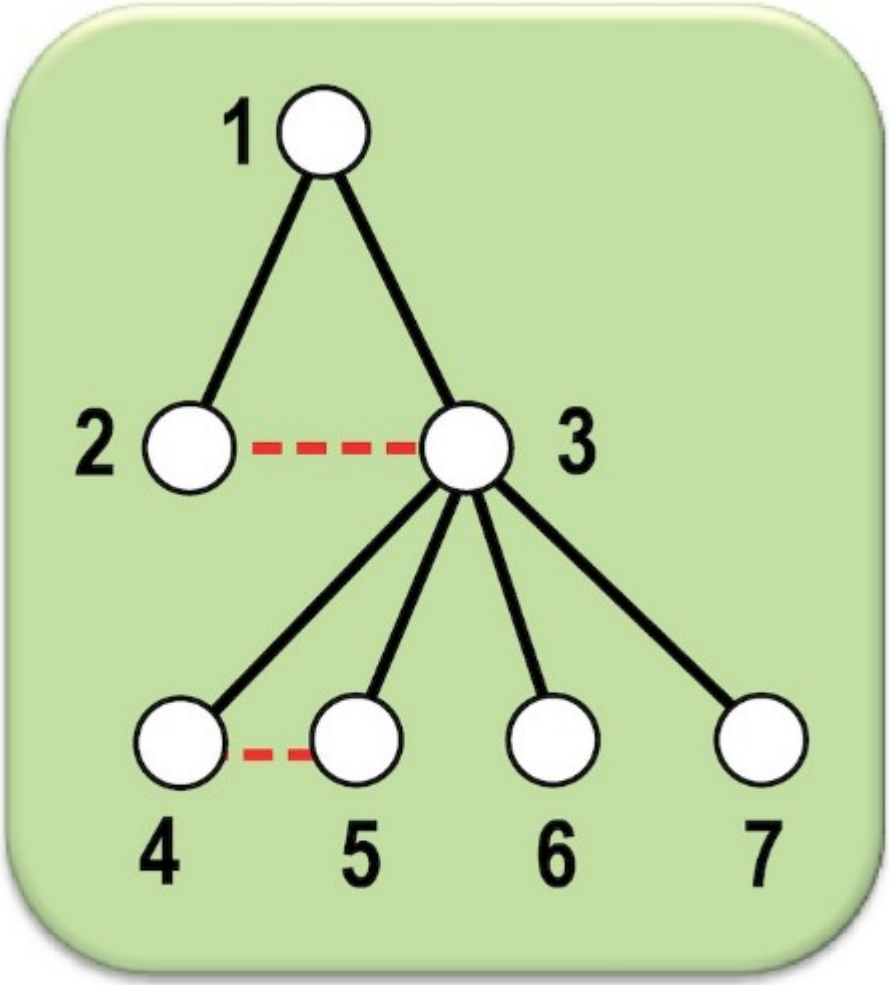


(8) Aresta$\{3, 7\}$

$Q = \{4, 5, 6, 7\}$

Busca em Largura - BFS

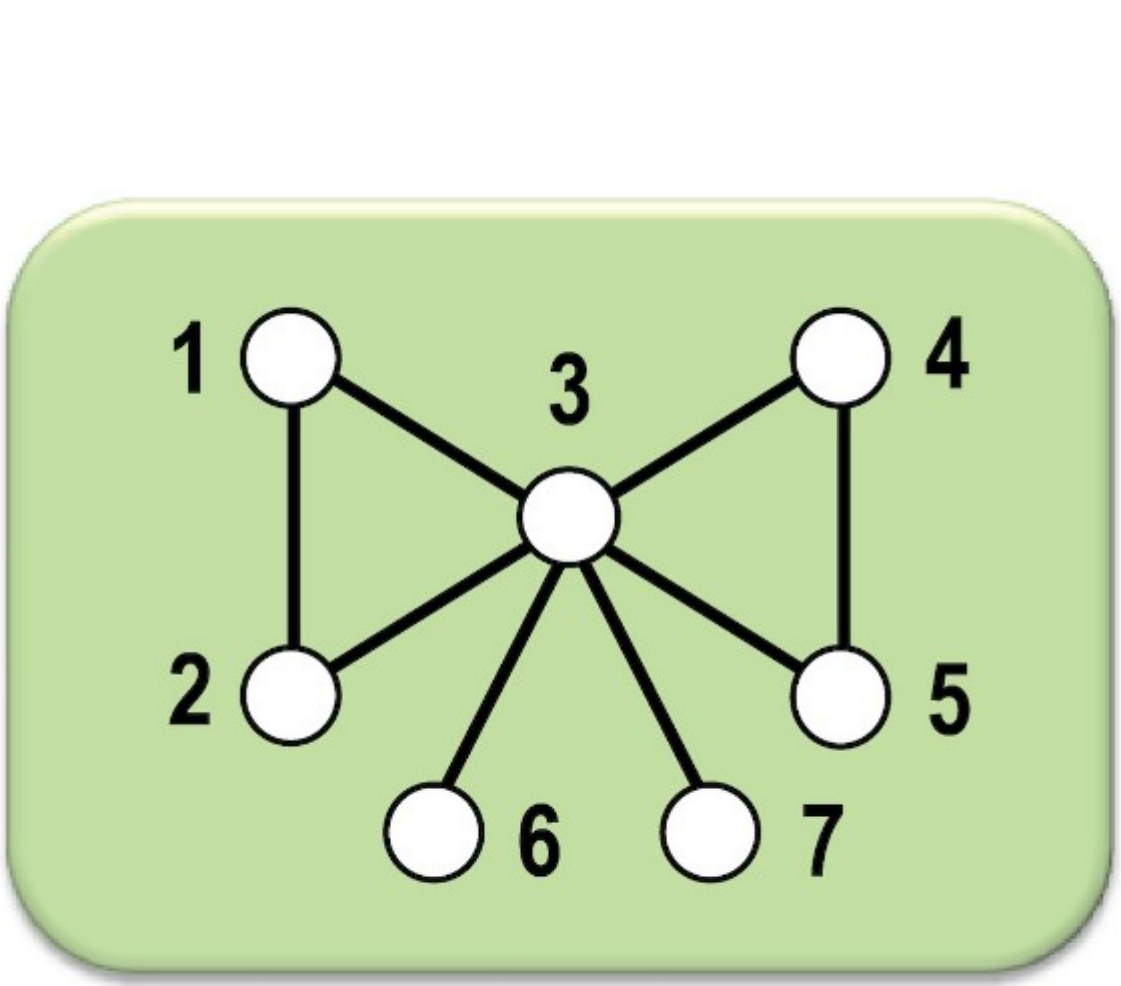


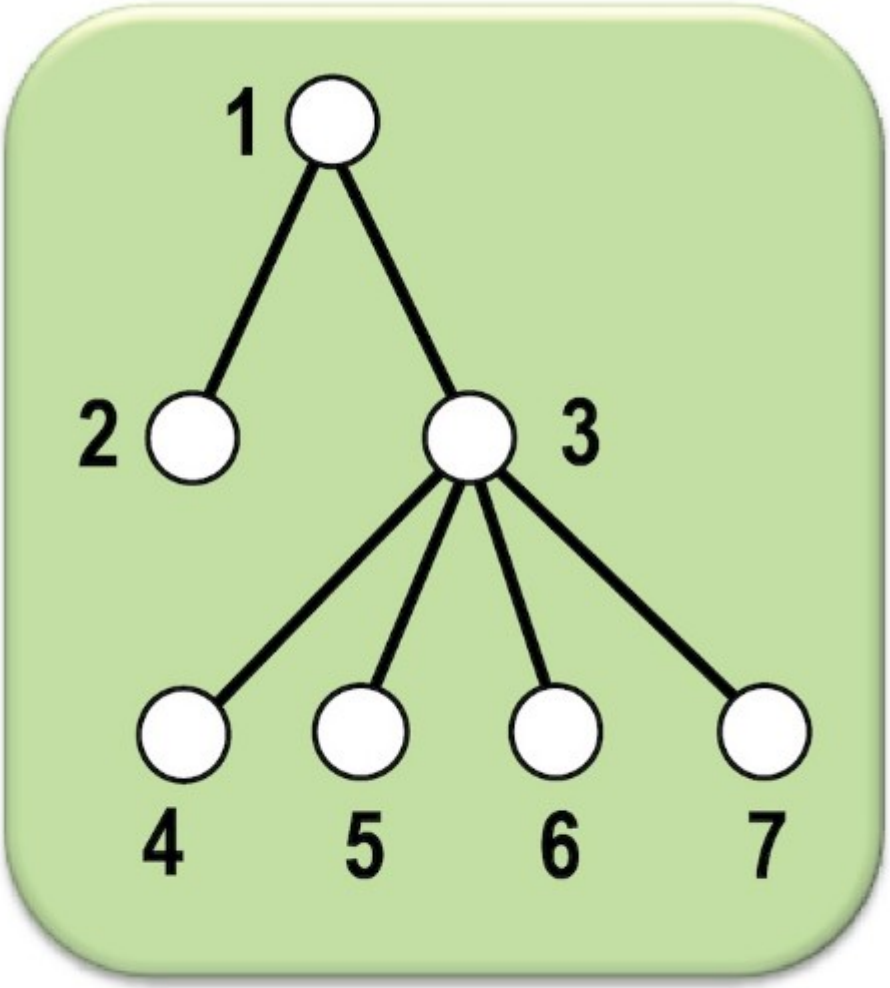


(9) Aresta$\{4, 5\}$

$Q = \{5, 6, 7\}$

Busca em Largura - BFS





Grafo original e respectiva árvore.

Busca em Largura - BFS

Busca em Largura - BFS

- Complexidade

Busca em Largura - BFS

- Complexidade
  - Cada vértice só entra na fila uma vez.

Busca em Largura - BFS

- Complexidade
  - Cada vértice só entra na fila uma vez.
  - Inserir e remover na fila possuem complexidade constante, realizadas |V| vezes cada.

Busca em Largura - BFS

- Complexidade
  - Cada vértice só entra na fila uma vez.
  - Inserir e remover na fila possuem complexidade constante, realizadas |V| vezes cada.
  - A lista de adjacências de cada vértice é examinada apenas uma vez, e a soma dos comprimentos de todas as listas é Θ(m).

Busca em Largura - BFS

- Complexidade
  - Cada vértice só entra na fila uma vez.
  - Inserir e remover na fila possuem complexidade constante, realizadas |V| vezes cada.
  - A lista de adjacências de cada vértice é examinada apenas uma vez, e a soma dos comprimentos de todas as listas é Θ(m).
  - Logo, se representarmos o grafo por uma lista de adjacências, a busca em largura (BFS) tem complexidade O(n+m).

Exemplo

https://www.cs.usfca.edu/~galles/visualization/BFS.html

DFS x BFS

DFS x BFS

- DFS – Busca em Profundidade

DFS x BFS

- DFS – Busca em Profundidade
  - Incursão profundas no grafo, voltando somente quando não existem mais vértices desconhecidos pela frente.

DFS x BFS

- DFS – Busca em Profundidade
  - Incursão profundas no grafo, voltando somente quando não existem mais vértices desconhecidos pela frente.
  - Marca o vértice antes de visitar toda sua vizinhança.

DFS x BFS

- DFS – Busca em Profundidade
  - Incursão profundas no grafo, voltando somente quando não existem mais vértices desconhecidos pela frente.
  - Marca o vértice antes de visitar toda sua vizinhança.
  - Uso de pilha.

DFS x BFS

- DFS – Busca em Profundidade
  - Incursão profundas no grafo, voltando somente quando não existem mais vértices desconhecidos pela frente.
  - Marca o vértice antes de visitar toda sua vizinhança.
  - Uso de pilha.
- BFS – Busca em Largura

DFS x BFS

- DFS – Busca em Profundidade
  - Incursão profundas no grafo, voltando somente quando não existem mais vértices desconhecidos pela frente.
  - Marca o vértice antes de visitar toda sua vizinhança.
  - Uso de pilha.
- BFS – Busca em Largura
  - Busca progride em largura: certifica-se de que vizinhos próximos sejam visitados primeiramente.

DFS x BFS

- DFS – Busca em Profundidade
  - Incursão profundas no grafo, voltando somente quando não existem mais vértices desconhecidos pela frente.
  - Marca o vértice antes de visitar toda sua vizinhança.
  - Uso de pilha.
- BFS – Busca em Largura
  - Busca progride em largura: certifica-se de que vizinhos próximos sejam visitados primeiramente.
  - Marca o vértice depois de visitar toda a sua vizinhança.

DFS x BFS

- DFS – Busca em Profundidade
  - Incursão profundas no grafo, voltando somente quando não existem mais vértices desconhecidos pela frente.
  - Marca o vértice antes de visitar toda sua vizinhança.
  - Uso de pilha.
- BFS – Busca em Largura
  - Busca progride em largura: certifica-se de que vizinhos próximos sejam visitados primeiramente.
  - Marca o vértice depois de visitar toda a sua vizinhança.
  - Uso de fila.

Aplicações

Aplicações

- Detectar grafos desconectados.

Aplicações

- Detectar grafos desconectados.
- Detectar se o grafo possui ciclos.

Aplicações

- Detectar grafos desconectados.
- Detectar se o grafo possui ciclos.
- Encontrar componentes biconexas.

Aplicações

- Detectar grafos desconectados.
- Detectar se o grafo possui ciclos.
- Encontrar componentes biconexas.
- Classificar arestas.

Aplicações

- Detectar grafos desconectados.
- Detectar se o grafo possui ciclos.
- Encontrar componentes biconexas.
- Classificar arestas.
- Encontrar componentes fortemente conexas.

Aplicações

- Detectar grafos desconectados.
- Detectar se o grafo possui ciclos.
- Encontrar componentes biconexas.
- Classificar arestas.
- Encontrar componentes fortemente conexas.
- Determinar o menor caminho em grafos não ponderados.