ESTRUTURAS DE DADOS II

MSC. DANIELE CARVALHO OLIVEIRA

DOUTORANDA EM CIÊNCIA DA COMPUTAÇÃO - USP

MESTRE EM CIÊNCIA DA COMPUTAÇÃO – UFU

BACHAREL EM CIÊNCIA DA COMPUTAÇÃO - UFJF

2

REPRESENTAÇÃO DE GRAFOS

3 REPRESENTAÇÃO DOS GRAFOS

- Como representar grafos em nossos algoritmos?

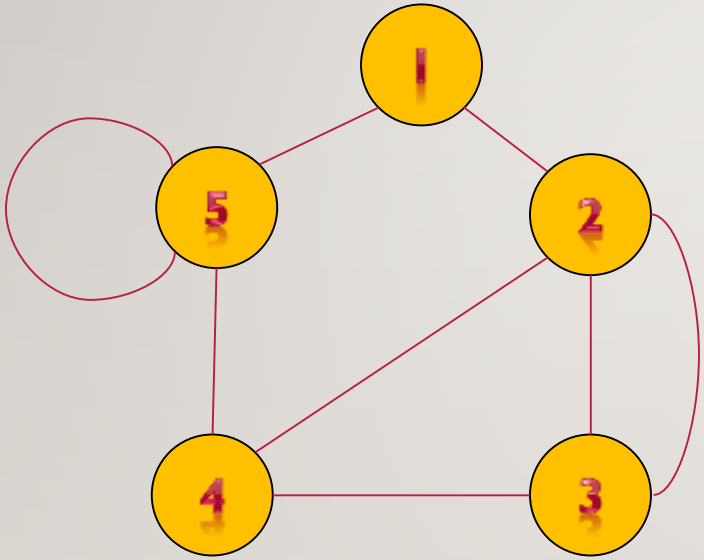
- Estruturas de dados!
  - Matrizes
    - Matriz de Adjacência
    - Matriz de Incidência
  - Listas
    - Listas de Adjacência

4 MATRIZ DE ADJACÊNCIA

- Dado um grafo $G(V, E)$
- Uma Matriz de Adjacência M é formada por n linhas e n colunas
  - n = número de vértices do grafo

$$M_{ij} = \begin{cases} 1, se\ (i,j) \in E(G) \\ 0, se\ (i,j) \notin E(G) \end{cases}$$

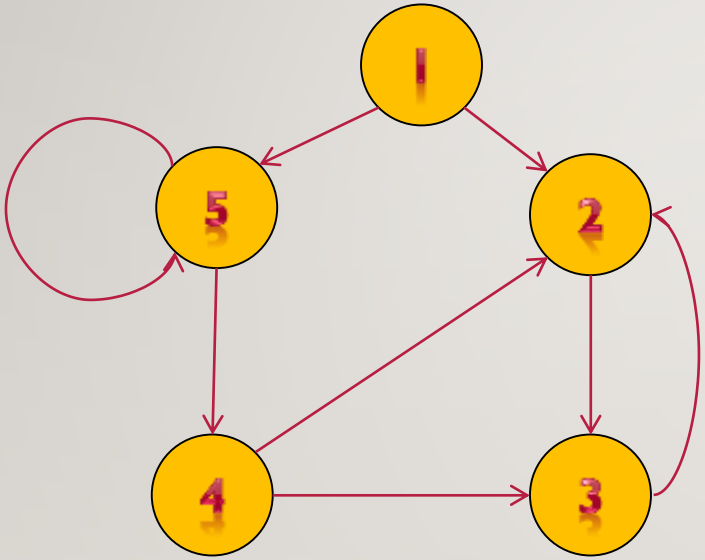
5



	1	2	3	4	5
1	0	1	0	0	1
2	1	0	1	1	0
3	0	1	0	1	0
4	0	1	1	0	1
5	1	0	0	1	1

Simétrica

6



	1	2	3	4	5
1	0	1	0	0	1
2	0	0	1	0	0
3	0	1	0	0	0
4	0	1	1	0	0
5	0	0	0	1	1

7 MATRIZ DE ADJACÊNCIA

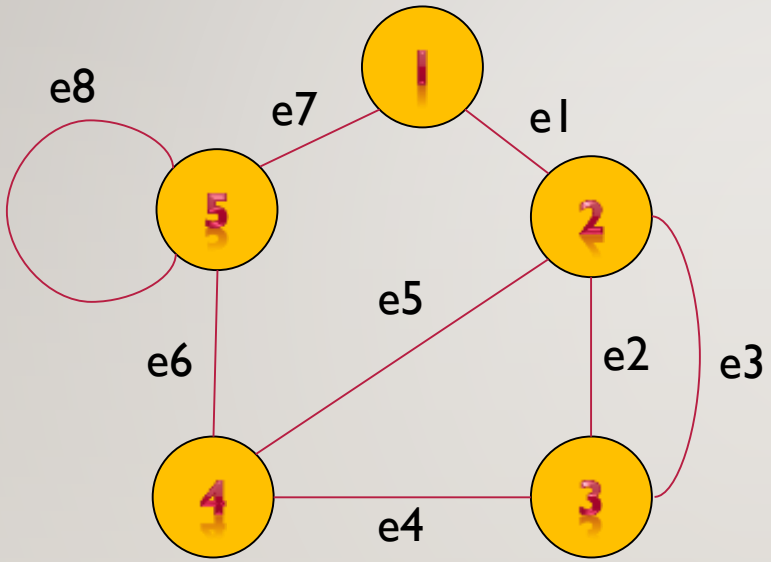
- O espaço reservado para armazenar as informações da matriz é da ordem $O(|V|^2)$
- Para buscar uma aresta: $O(1)$

8 MATRIZ DE INCIDÊNCIA

- Dado um grafo $G(V, E)$ de n vértices e m arestas

$$M_{ij} = \begin{cases} 1, \forall v_i, se\ (v_i, v_j) \in E(G) \\ 0, se\ (v_i, v_j) \notin E(G) \\ 0, se\ (v_i, v_i) \in E(G) \end{cases}$$

9

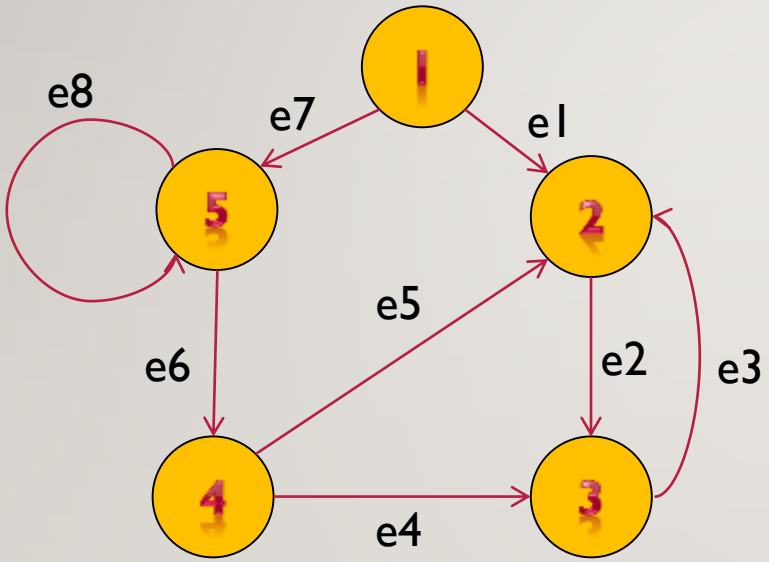


	e1	e2	e3	e4	e5	e6	e7	e8
1	1	0	0	0	0	0	1	0
2	1	1	1	0	1	0	0	0
3	0	1	1	1	0	0	0	0
4	0	0	0	1	1	1	0	0
5	0	0	0	0	0	1	1	0

MATRIZ DE INCIDÊNCIA NO DIGRAFO

$$M_{ij} = \begin{cases} 1, \forall v_i, se\ (v_i, v_j) \in E(G) \\ -1, \forall v_j, se\ (v_i, v_j) \in E(G) \\ 0, se\ (v_i, v_j) \notin E(G) \\ 0, se\ (v_i, v_i) \in E(G) \end{cases}$$

11



	e1	e2	e3	e4	e5	e6	e7	e8
1	1	0	0	0	0	0	1	0
2	-1	1	-1	0	-1	0	0	0
3	0	-1	1	-1	0	0	0	0
4	0	0	0	1	1	-1	0	0
5	0	0	0	0	0	1	-1	0

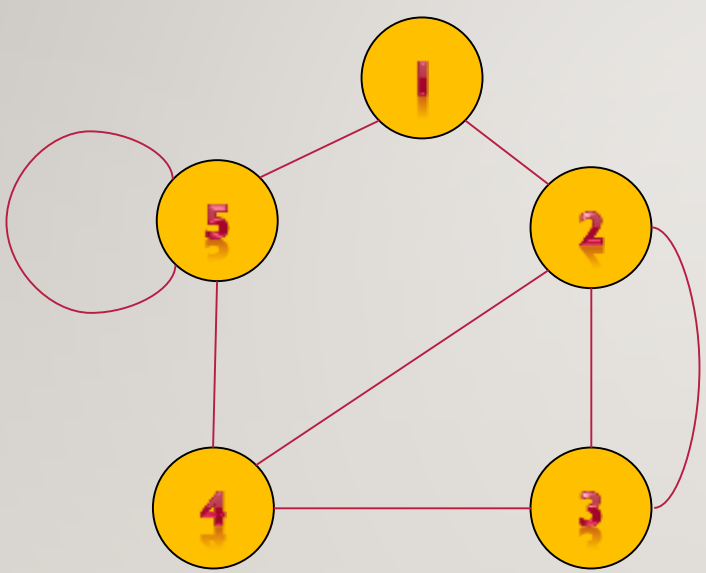
MATRIZ DE INCIDÊNCIA

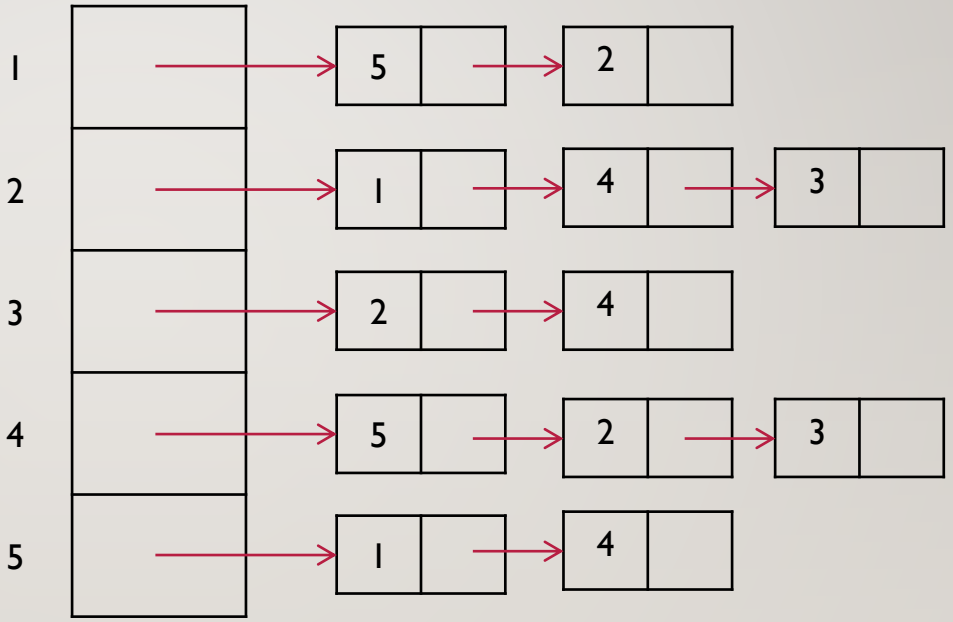
- O espaço reservado para armazenar as informações da matriz é da ordem $O(|V| \times |E|)$
- Para buscar uma aresta: $O(1)$

13 LISTA DE ADJACÊNCIA

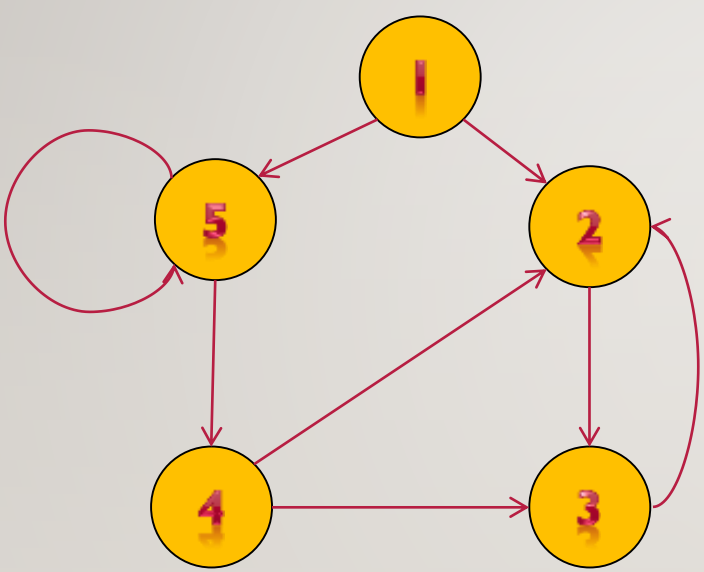
- Consiste em um vetor Adj com $n = |V|$ entradas
- Cada entrada $Adj[v]$ possui uma lista encadeada de vértices adjacentes à v.

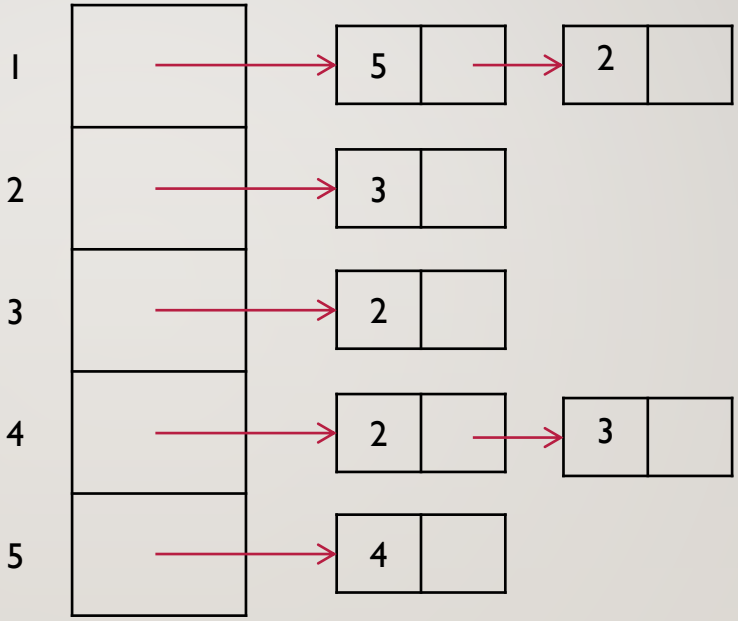
14





15





16 LISTA DE ADJACÊNCIA

- O espaço reservado para armazenar as informações da lista é da ordem $O(|V| + |E|)$
- Para buscar uma aresta: $O(n)$
  - Para descobrir se existe a aresta (i, j) deve-se percorrer a lista do nó i até encontrar (ou não) j

BUSCA EM GRAFOS

18 ALGORITMOS DE BUSCA

- Operação mais comum em Grafos
  - Visita sistemática a seus nós (uma única vez!)
- Similar às buscas em árvore

- Para passear ou caminhar pelos vértices e arestas, utiliza-se marcar um vértice quando ele já foi visitado.

19 ALGORITMOS DE BUSCA

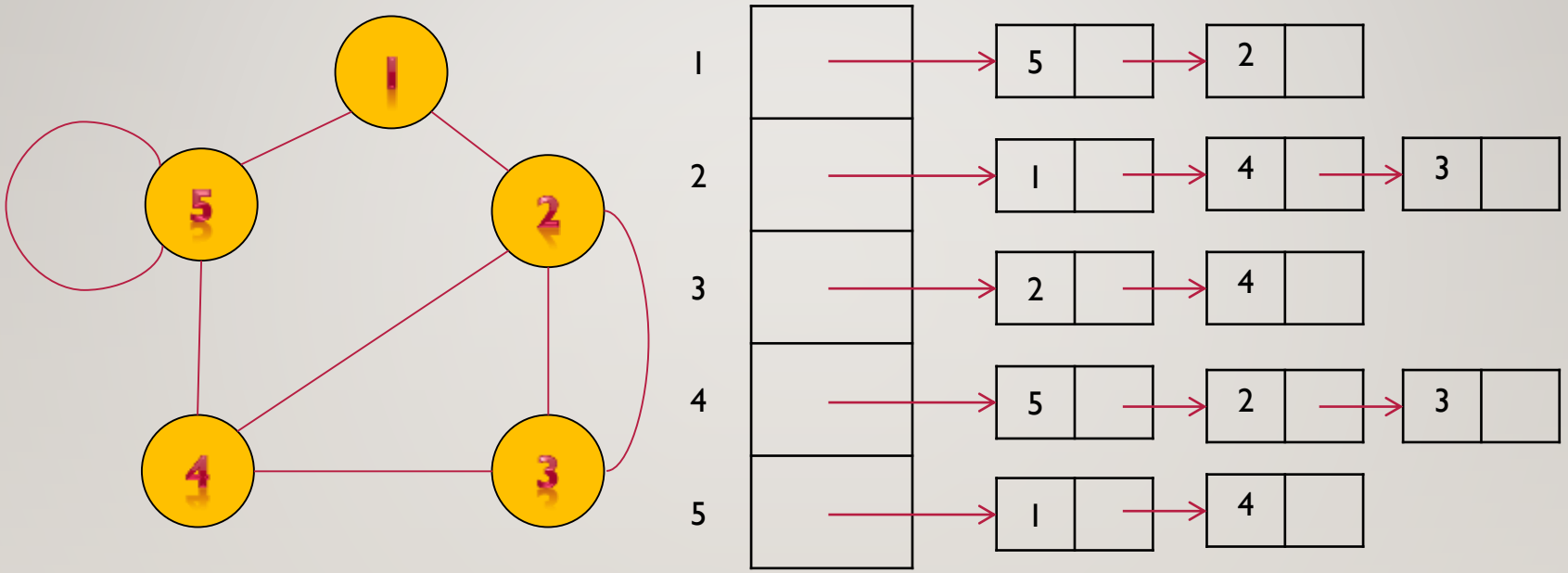
- Dois tipos básicos de busca
  - Busca em Profundidade
  - Busca em Largura

```
procedimento Busca(G: Grafo)

    Para Cada vértice v de G:

        Marque v como não visitado

    Para Cada vértice v de G:

        Se v não foi visitado:

            Busca-Prof(v)


procedimento Busca-Prof(v: vértice)

    Marque v como visitado
    Para Cada vértice w adjacente a v:

        Se w não foi visitado:

            Busca-Prof(w)
```

BUSCA EM PROFUNDIDADE

DFS

21



22 BUSCA EM PROFUNDIDADE

- Para acessar todos os vértices do grafo, a busca em profundidade varre a lista de adjacência de cada vértice do grafo
  - Assim, o tempo gasto para a operação é

$$O(|V| + |E|)$$

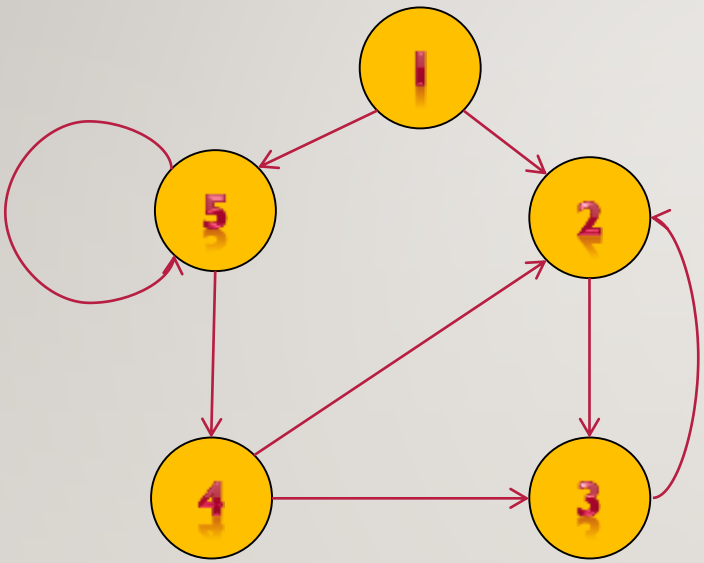
```
procedimento Busca-Largura(v: vértice)

    Inicializar F
    Marcar v como visitado
    Colocar v no final de F
    Enquanto F não vazio:

        u := primeiro elemento de F
        Retirar u de F
        Para cada vértice w adjacente a u:

            Se w não foi visitado:

                Marcar w como visitado
                Colocar w no final de F
```

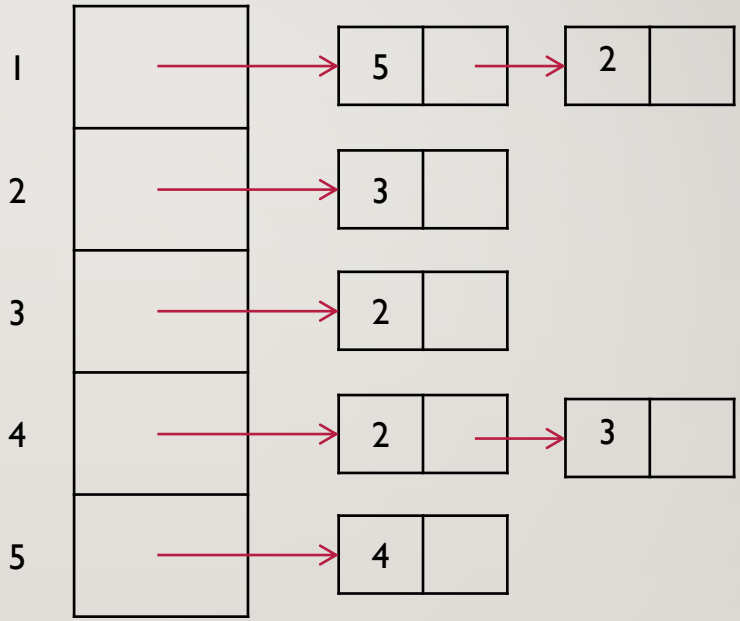
BUSCA EM LARGURA

BFS

Distâncias

1	2	3	4	5
∞	∞	∞	∞	∞



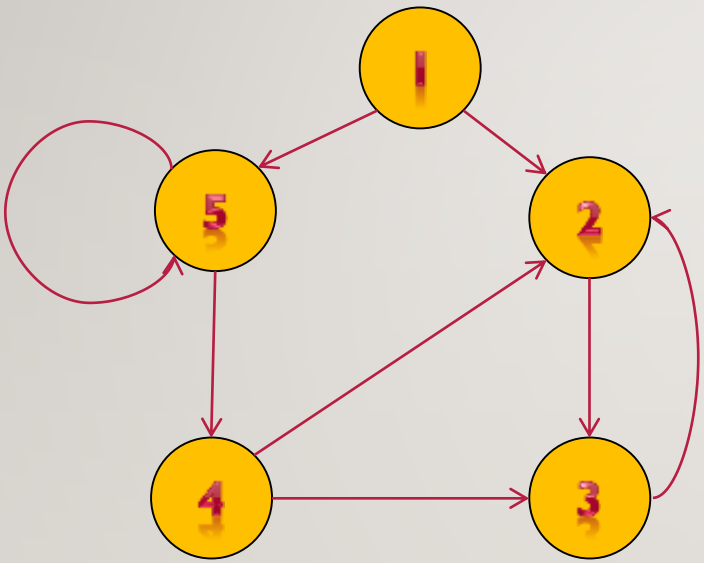


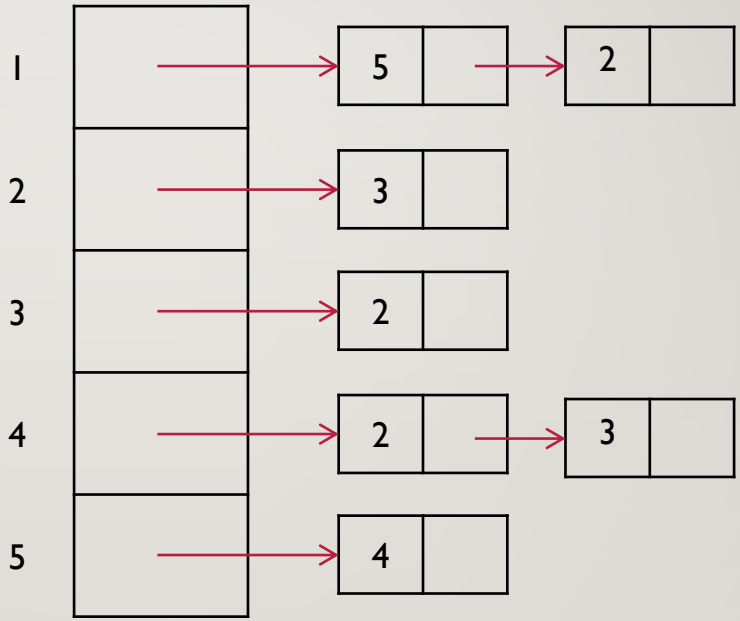
Fila

1									

Distâncias

1	2	3	4	5
0	∞	∞	∞	∞

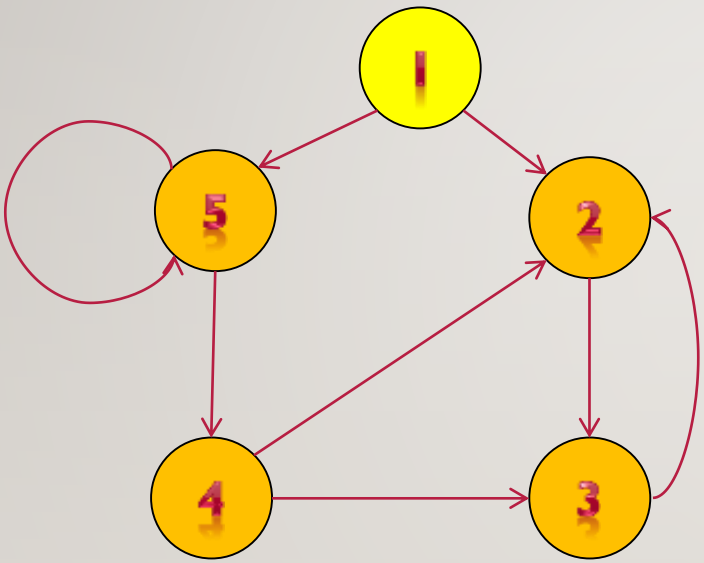


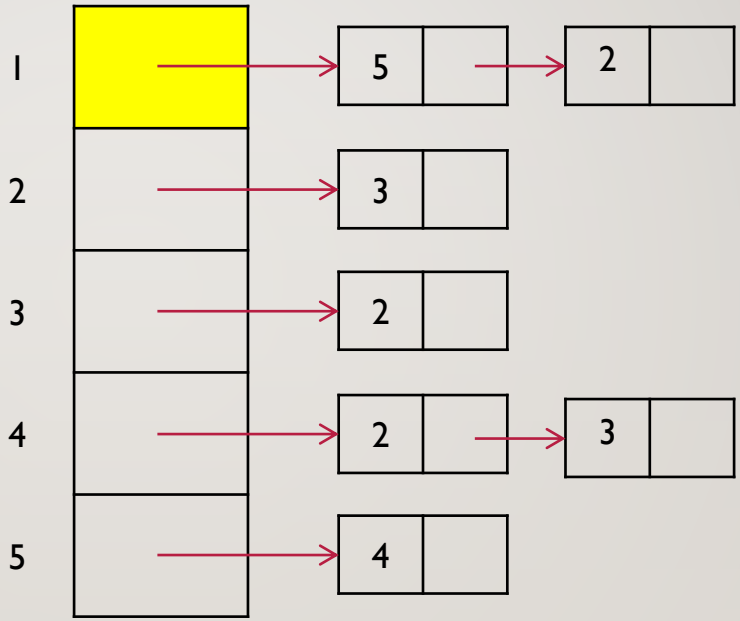


Fila

Distâncias

1	2	3	4	5
0	1	∞	∞	1



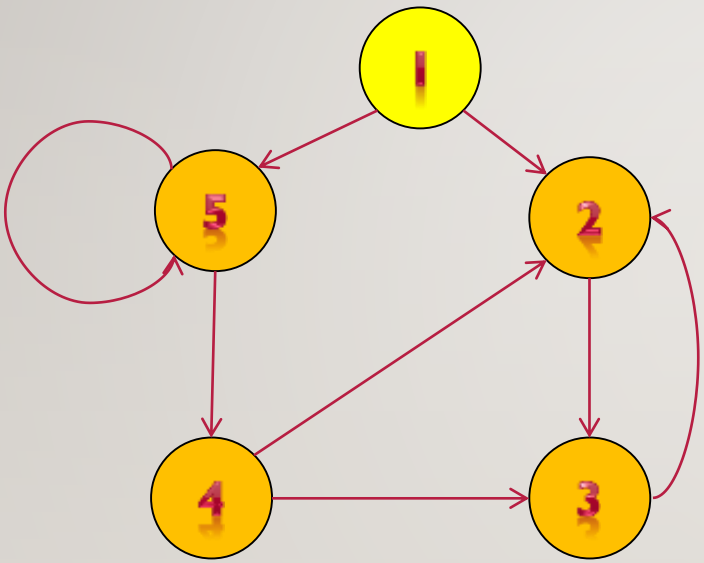


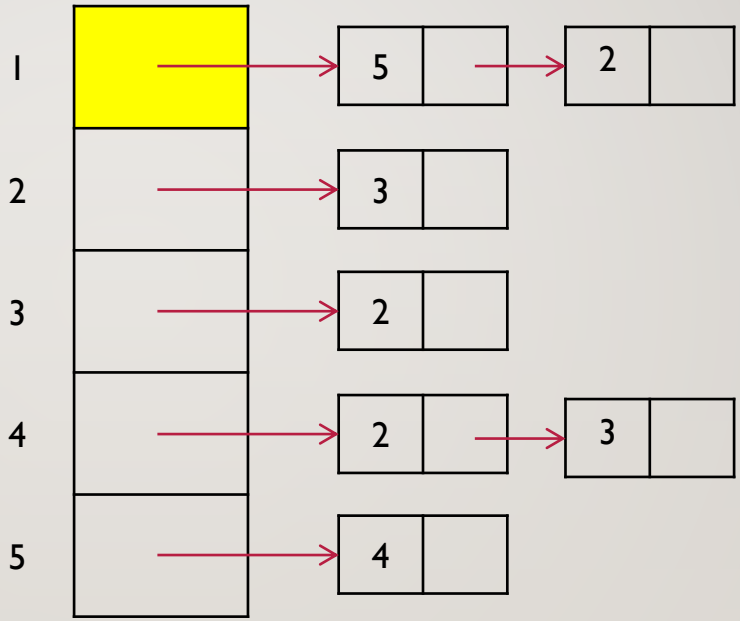
Fila

5	2								

Distâncias

1	2	3	4	5
0	1	∞	∞	1



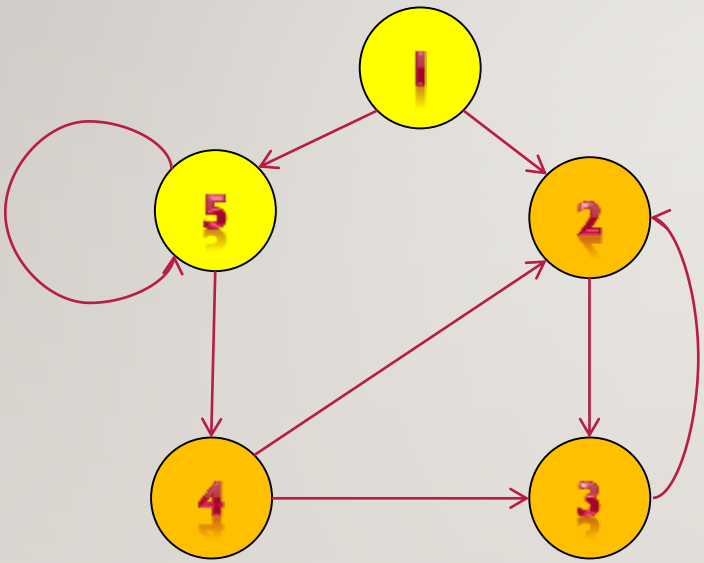


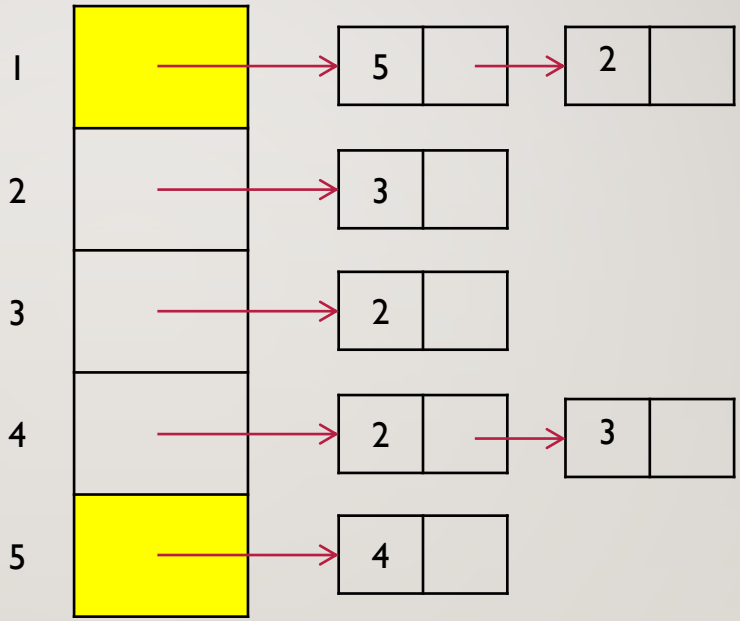
Fila

2									

Distâncias

1	2	3	4	5
0	1	∞	∞	1



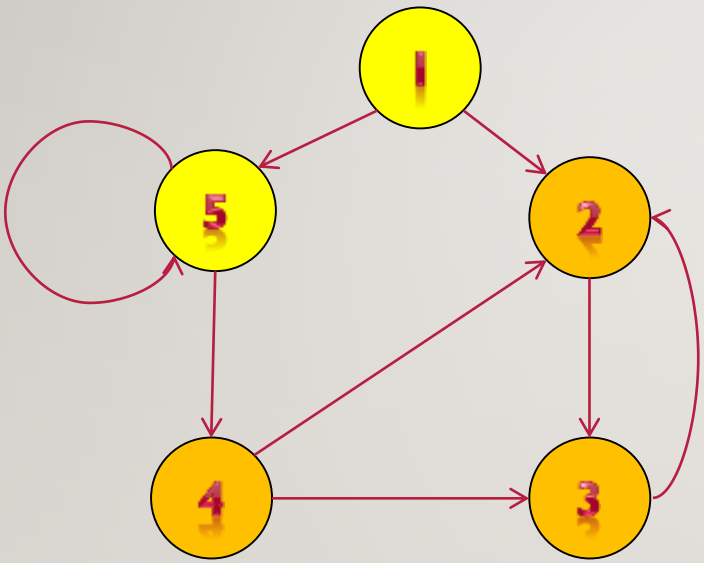


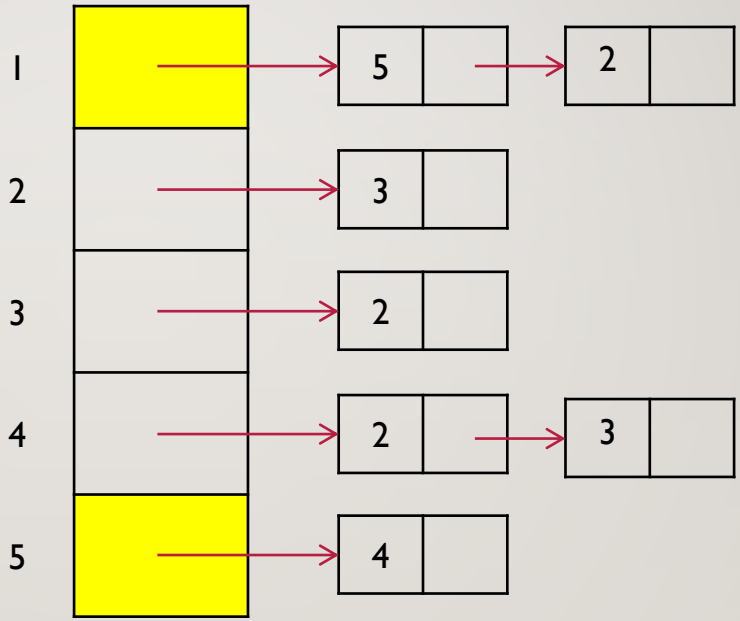
Fila

2									

Distâncias

1	2	3	4	5
0	1	∞	2	1



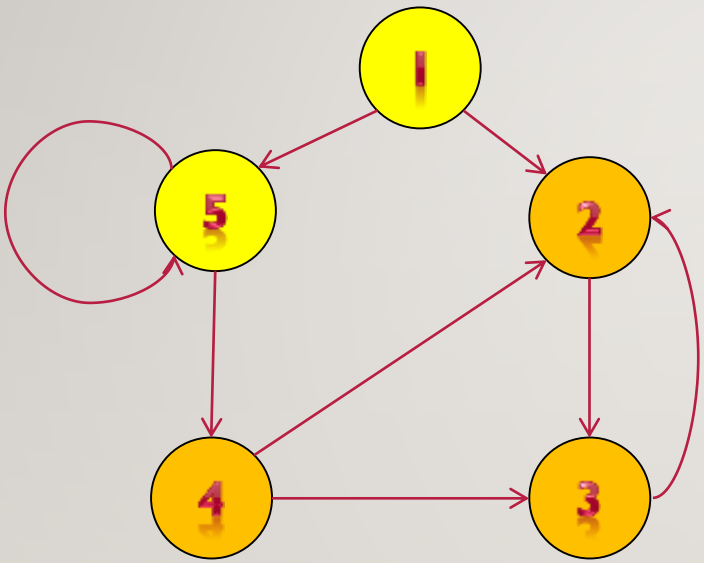


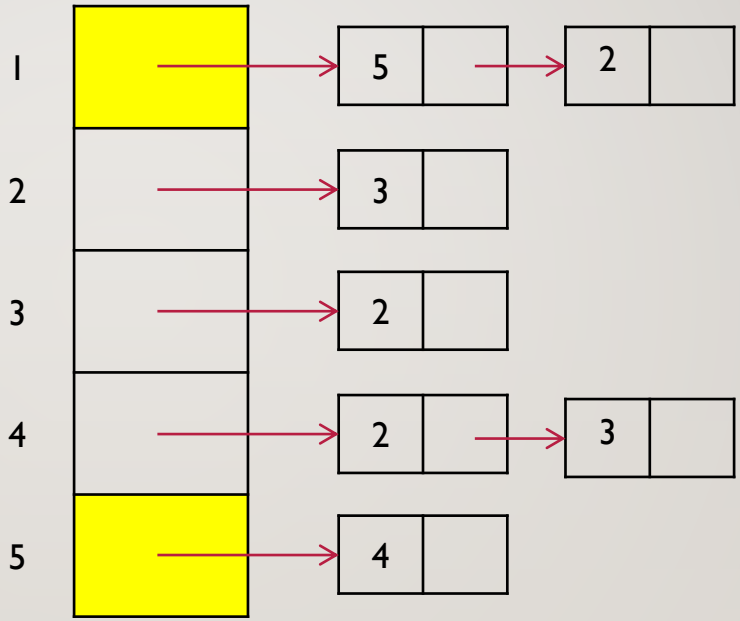
Fila

2	4								

Distâncias

1	2	3	4	5
0	1	∞	2	1



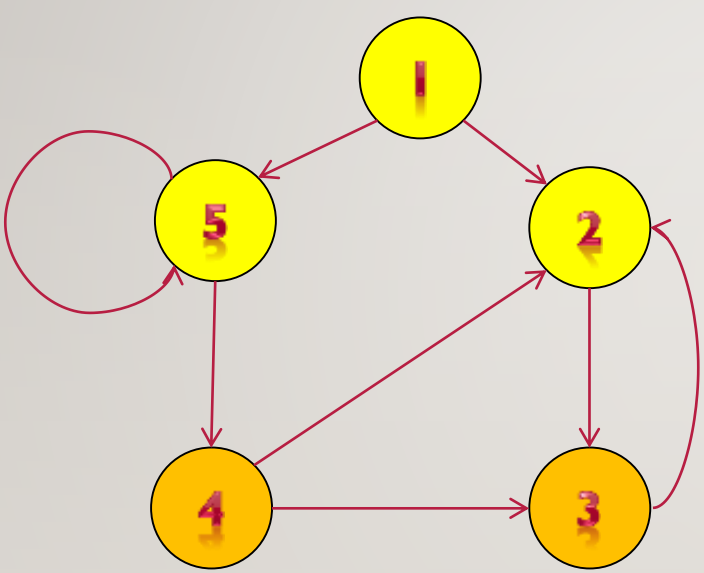


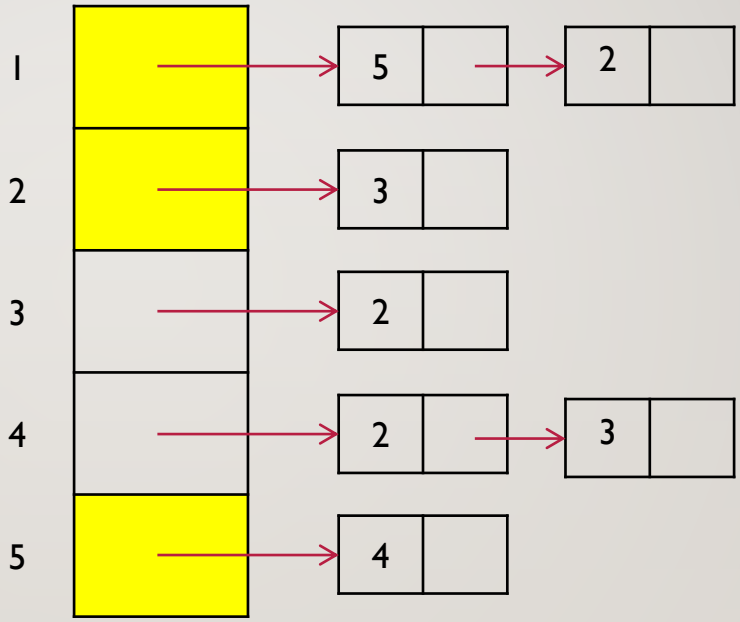
Fila

4									

Distâncias

1	2	3	4	5
0	1	∞	2	1



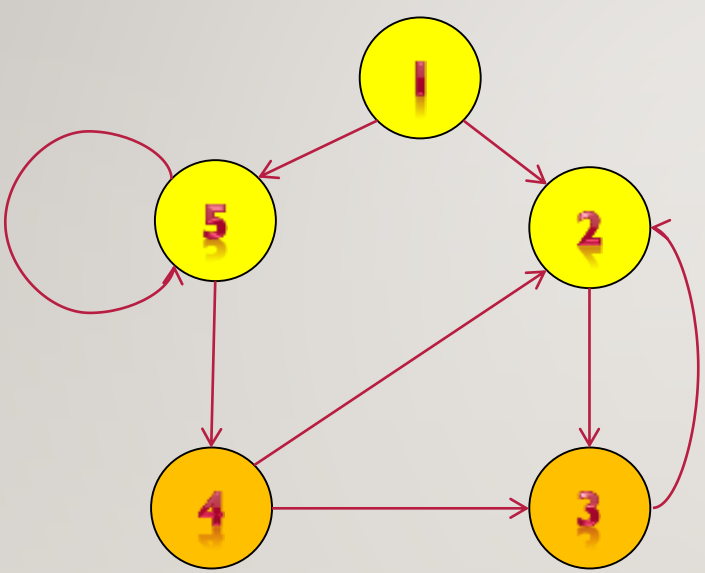


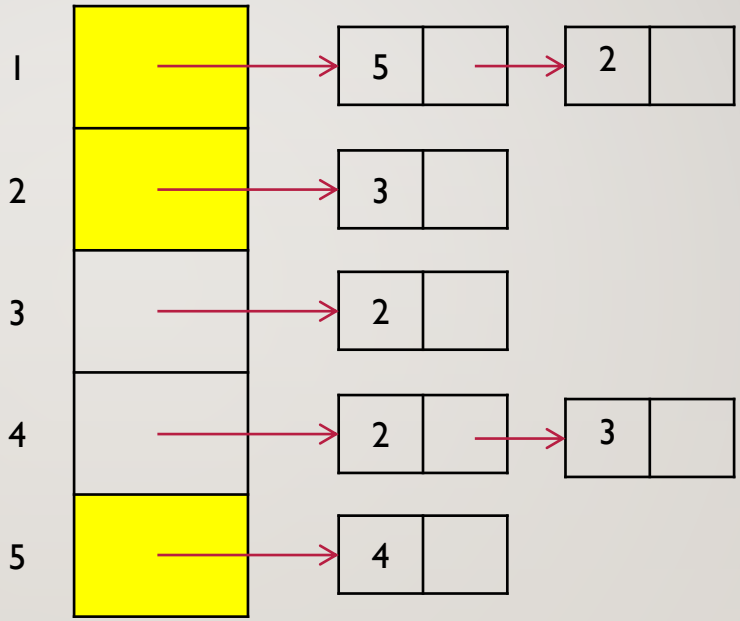
Fila

4									

Distâncias

1	2	3	4	5
0	1	2	2	1



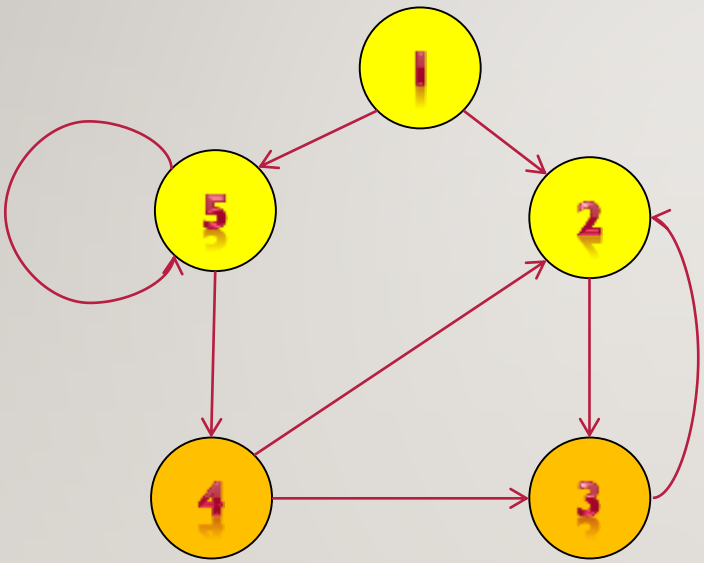


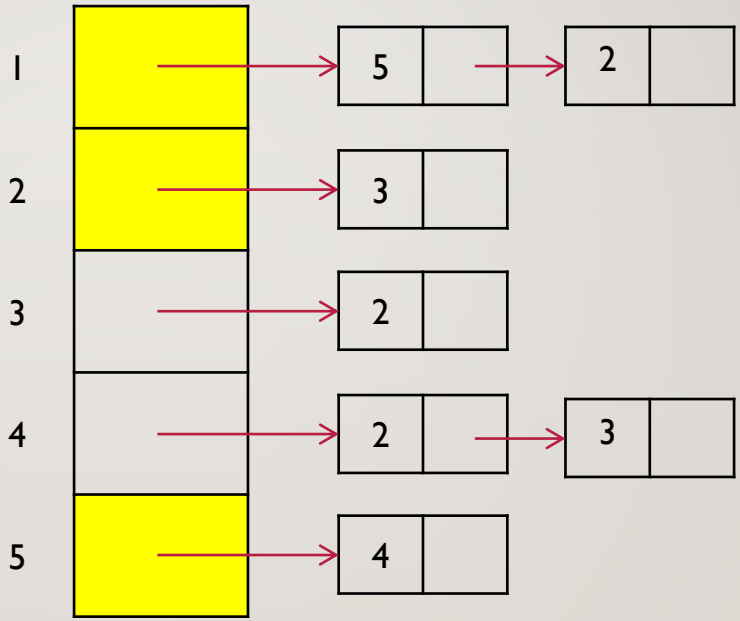
Fila

4	3								

Distâncias

1	2	3	4	5
0	1	2	2	1



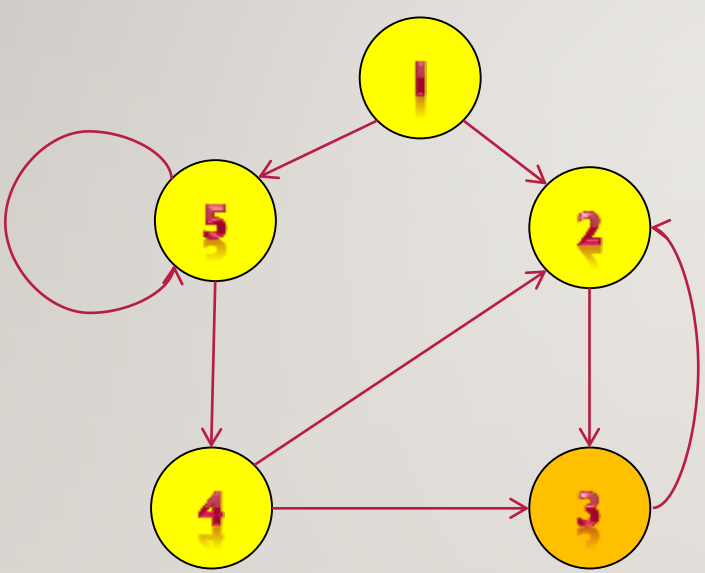


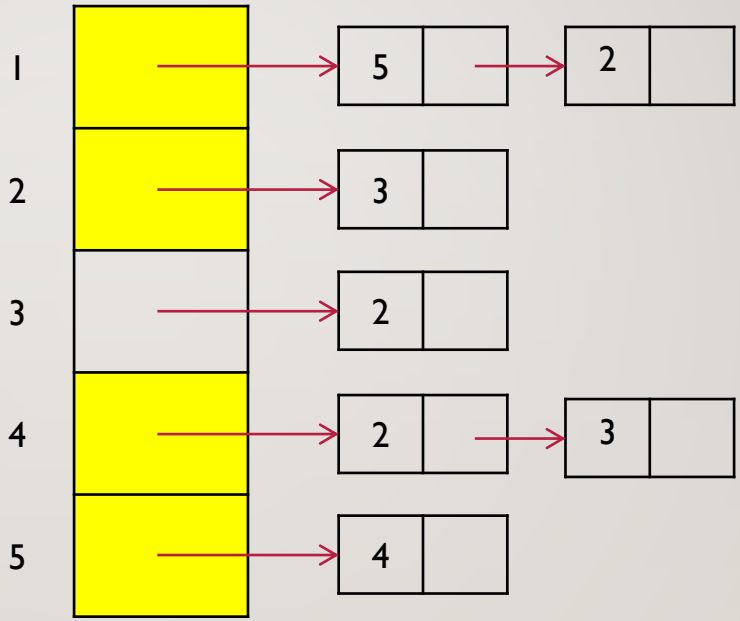
Fila

3									

Distâncias

1	2	3	4	5
0	1	2	2	1



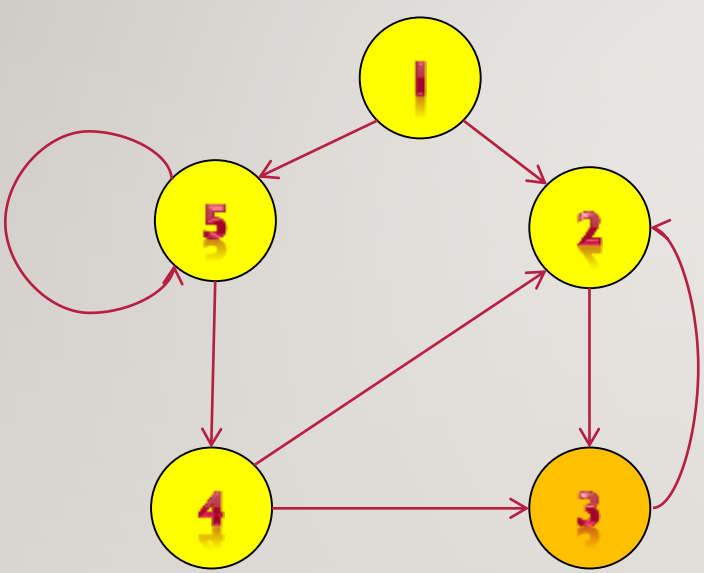


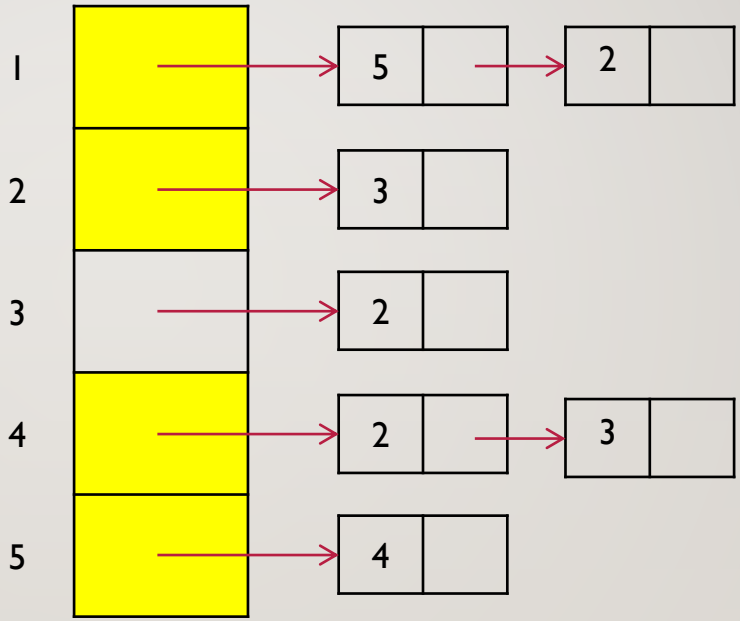
Fila

3									

Distâncias

1	2	3	4	5
0	1	2	2	1

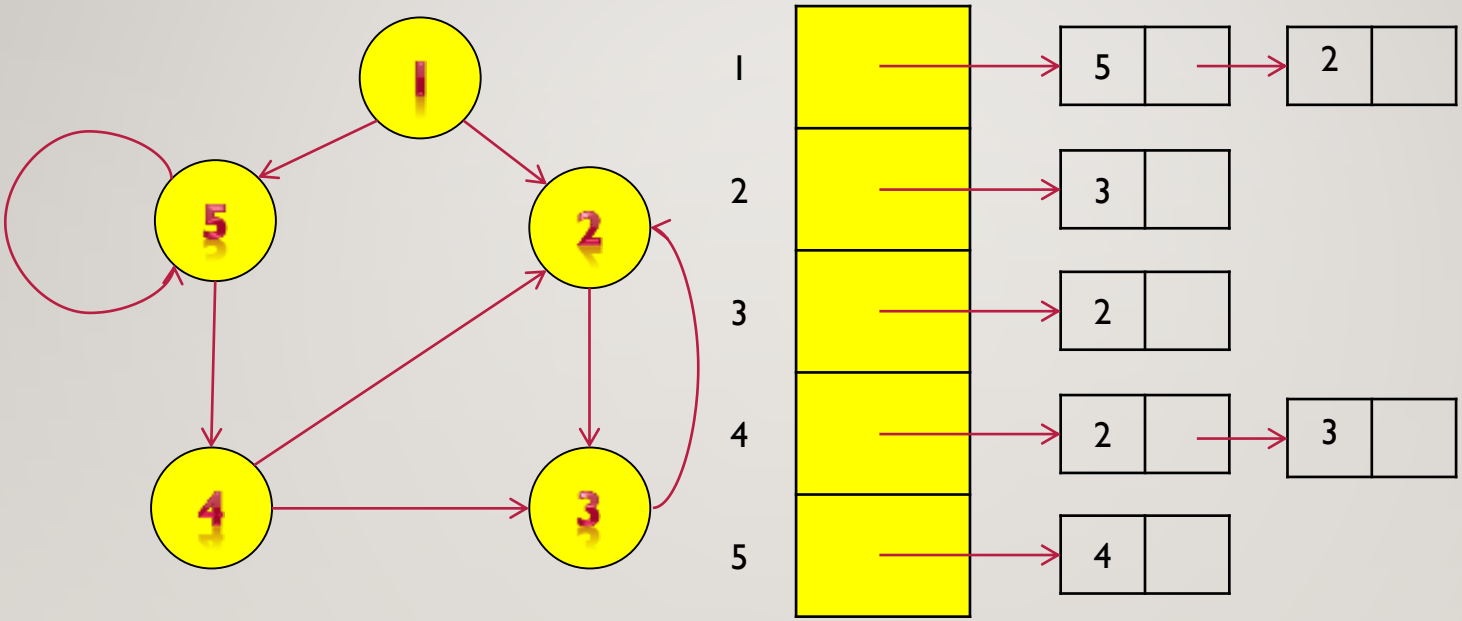




Fila

Distâncias

1	2	3	4	5
0	1	2	2	1



Fila

BUSCA EM LARGURA

- Primeiro os vértices são inicializados, ou seja, marcados como não visitados.
- O vetor de distâncias é inicializado com ∞ em todas as posições
  - Este processo de inicialização gasta tempo $O(|V|)$

38 BUSCA EM LARGURA

- A medida que os vértices são encontrados, são colocados em uma fila
  - As operações de inserção e remoção da fila gastam tempo $O(1)$
  - Como todos os vértices são colocados e retirados da fila, o tempo total é $O(|V|)$

BUSCA EM LARGURA

- Como cada vértice é colocado na fila apenas 1 vez, a lista de adjacência de cada um só é analisado 1 vez
  - Tempo $O(|E|)$

- Portanto o tempo de execução da busca em largura é $O(|V| + |E|)$

Trabalho

- Implemente o DFS e o BFS

Extra

- URI
  - 1076; 1081; 1317; 1469; 1862; 1082; 1269; 1550; 1692; 1910; 1928; 1487

FIM DA AULA 11

Próxima aula:

Grafos: Djikstra, Conjuntos Estáveis, Cliques