DCC
DEPARTAMENTO DE
CIÊNCIA DA COMPUTAÇÃO

UFMG
UNIVERSIDADE FEDERAL
DE MINAS GERAIS

Estruturas de Dados

Grafos

Professores: Anisio Lacerda
Wagner Meira Jr.
Washington Cunha

Grafos

Um grafo é uma representação abstrata de um conjunto de objetos e das relações existentes entre eles. É definido por um conjunto de **vértices** e por ligações que conectam pares de vértices, chamadas **arestas**. Um grafo *G* é denotado da forma:

$$G(V, E)$$

Onde **V** é o conjunto de vértices e **E** o conjunto de arestas.

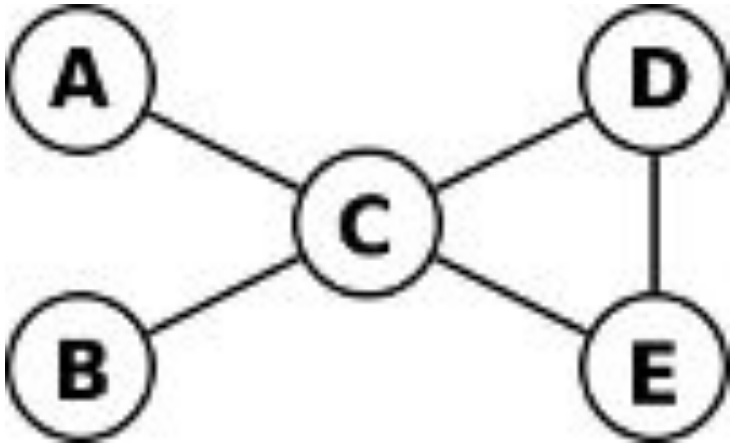
Grafos - representação

Um grafo pode ser representado exibindo a descrição de ambos conjuntos V e E, no entanto a representação mais comum é a gráfica.

Representação por conjuntos:

V = {A, B, C, D, E}
E = {(A,C), (B,C), (C,D), (D, E), (C, E)}

Representação gráfica:



Grafos - algumas definições

- Dois vértices *u* e *v* são ditos **adjacentes** ou **vizinhos** se existe a aresta *uv*.
- Um **caminho** é uma sequência de vértices $v_1 v_2 \ldots v_k$ de forma que $v_i v_{i+1}$ são vizinhos para todo 1 < i < k.
- O **grau** de um vértice *v*, denotado por *d(v)* é a quantidade de vizinhos que *v* possui. O grau máximo de um grafo é denotado $\Delta(G)$ e o grau mínimo por $\delta(G)$.

Grafos - algumas definições

- Um **laço** é uma aresta que conecta um vértice a ele mesmo.
- Uma aresta que conecta *u* a *v* é dita **paralela** se existe uma outra aresta que conecta *u* a *v*.
- Um grafo é **simples** se **não** possui laços nem arestas paralelas. Os grafos que veremos nesta disciplina serão todos simples.

Grafos - representação computacional

Mas como representar um grafo no computador? Basicamente temos uma lista de vértices e uma lista de arestas. Existem várias maneiras de representar um grafo como uma estrutura de dados, mas as duas principais são:

- Matriz de adjacência
- Lista de adjacência

Grafos - Operações

Outra preocupação são as operações estamos interessados em realizar. Vamos ver duas operações bem comuns, que são:

- Verificar a existência de uma aresta
- Dado um vértice *v*, consultar quais vértices são vizinhos de *v*.

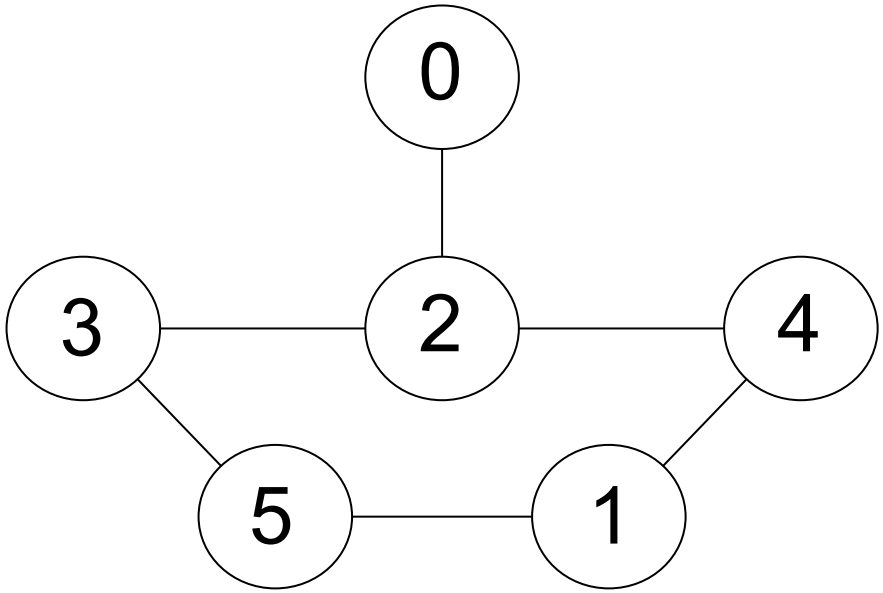
Grafos - Matriz de adjacência

Seja *G* um grafo com *n* vértices. Vamos construir uma matriz quadrada *A* com *n* dimensões que satisfaz a seguinte propriedade:

A[i][j] = 1 se e somente se existe uma aresta que conecta o vértice *i* com o vértice *j*.

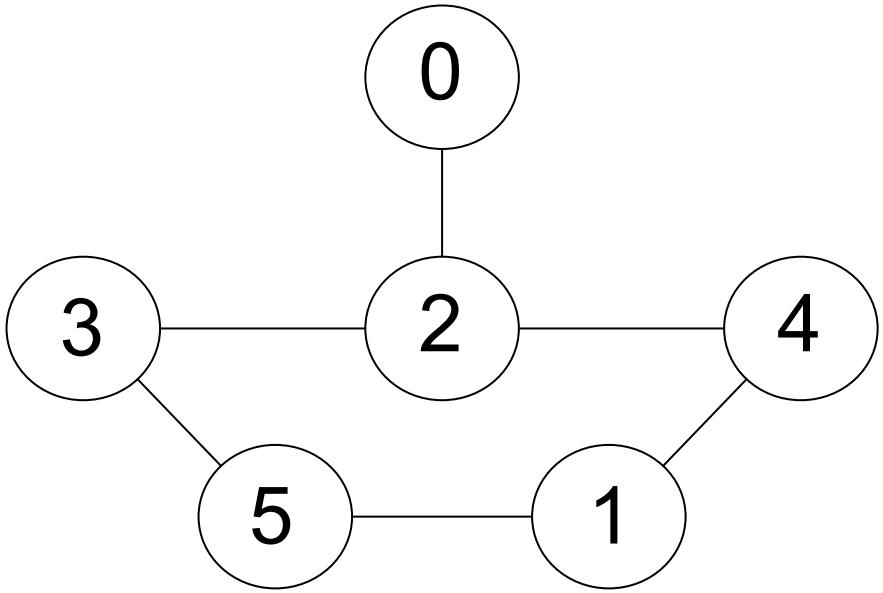
A[i][j] = 0 caso contrário.

Matriz de adjacência - Exemplo



0	0	1	0	0	0
0	0	0	0	1	1
1	0	0	1	1	0
0	0	1	0	0	1
0	1	1	0	0	0
0	1	0	1	0	0

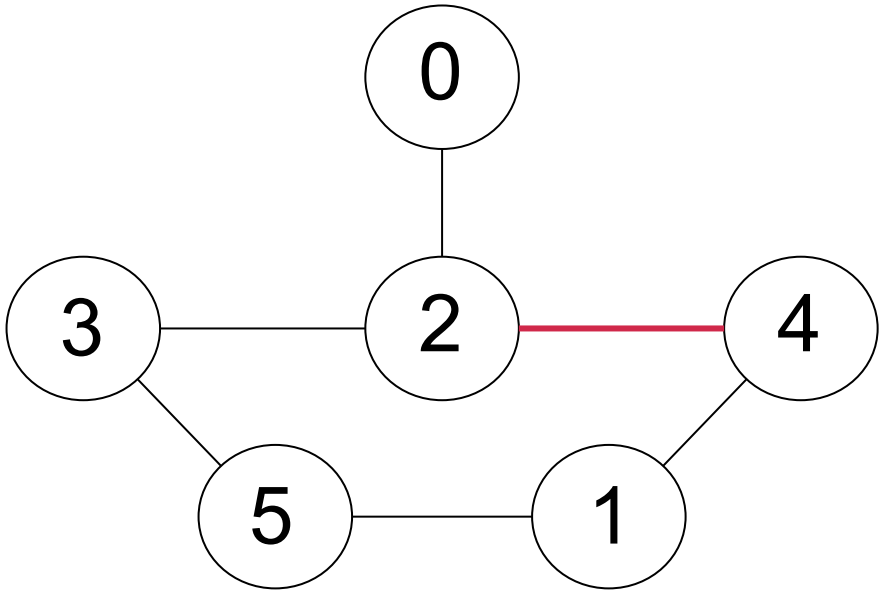
Matriz de adjacência - Exemplo



0	0	1	0	0	0
0	0	0	0	1	1
1	0	0	1	1	0
0	0	1	0	0	1
0	1	1	0	0	0
0	1	0	1	0	0

Observe que a matriz de adjacência é simétrica, uma vez que a aresta que conecta *u* com *v* também conecta *v* com *u*.

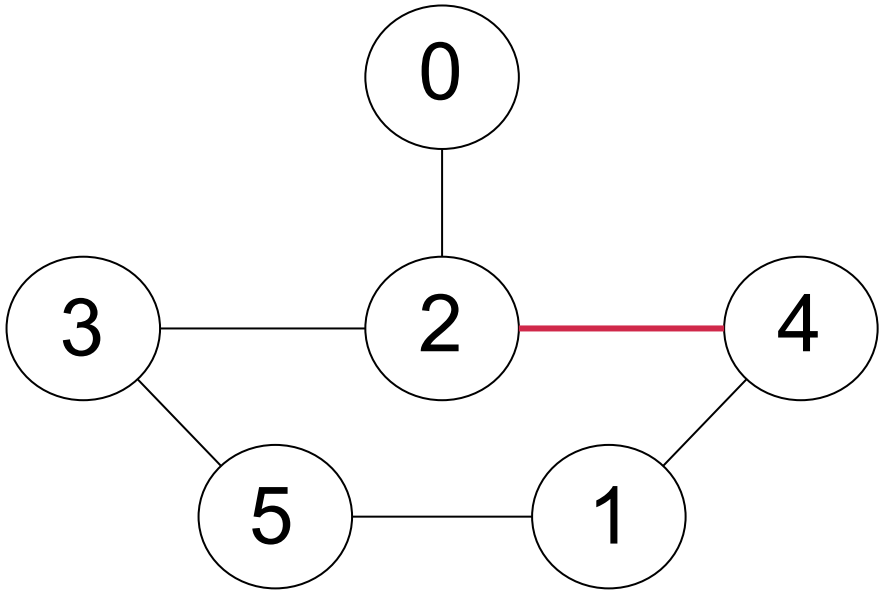
Existência de aresta



0	0	1	0	0	0
0	0	0	0	1	1
1	0	0	1	1	0
0	0	1	0	0	1
0	1	1	0	0	0
0	1	0	1	0	0

Para verificar a existência de uma aresta basta consultar a célula correspondente da matriz.

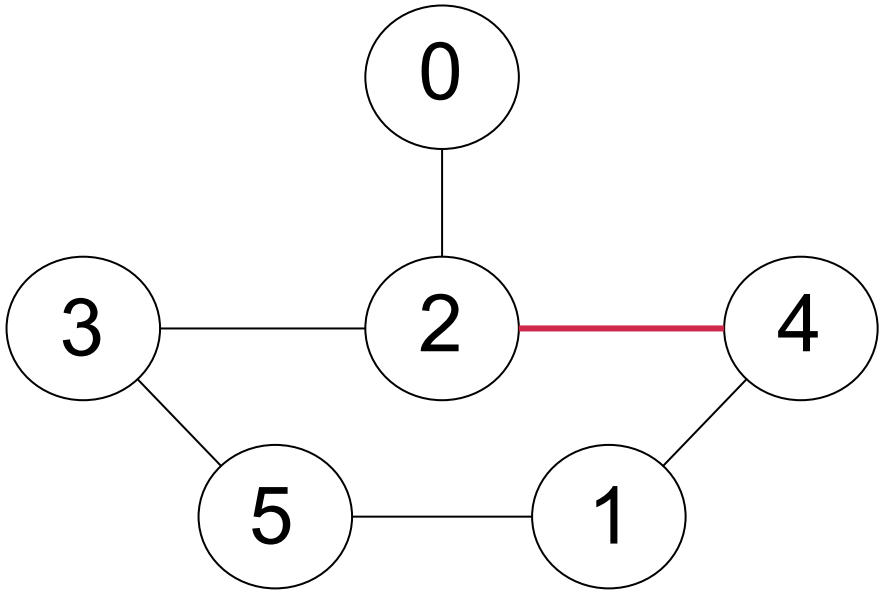
Existência de aresta



0	0	1	0	0	0
0	0	0	0	1	1
1	0	0	1	1	0
0	0	1	0	0	1
0	1	1	0	0	0
0	1	0	1	0	0

Para verificar a existência de uma aresta basta consultar a célula correspondente da matriz.

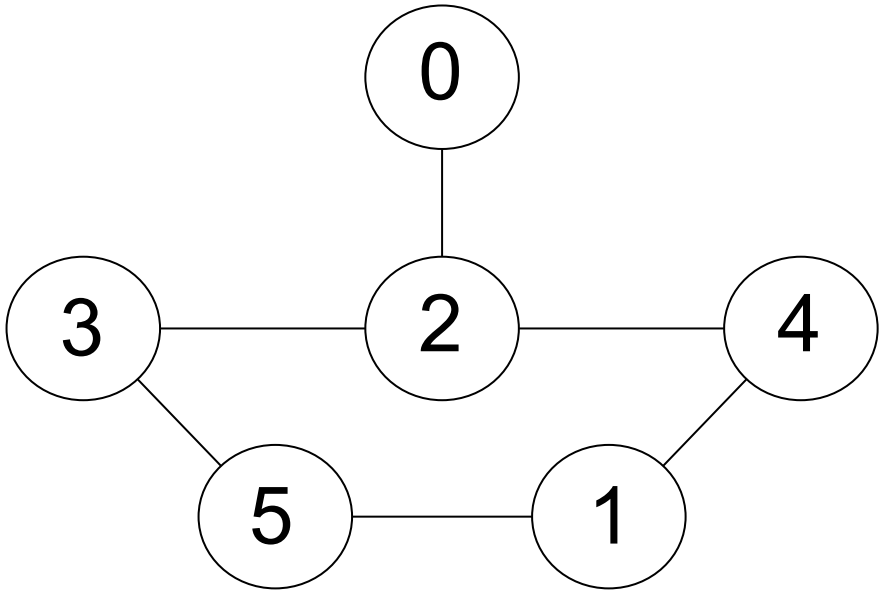
Existência de aresta



0	0	1	0	0	0
0	0	0	0	1	1
1	0	0	1	1	0
0	0	1	0	0	1
0	1	1	0	0	0
0	1	0	1	0	0

Para verificar a existência de uma aresta basta consultar a célula correspondente da matriz. Esta operação pode ser realizada em tempo *O(1)*.

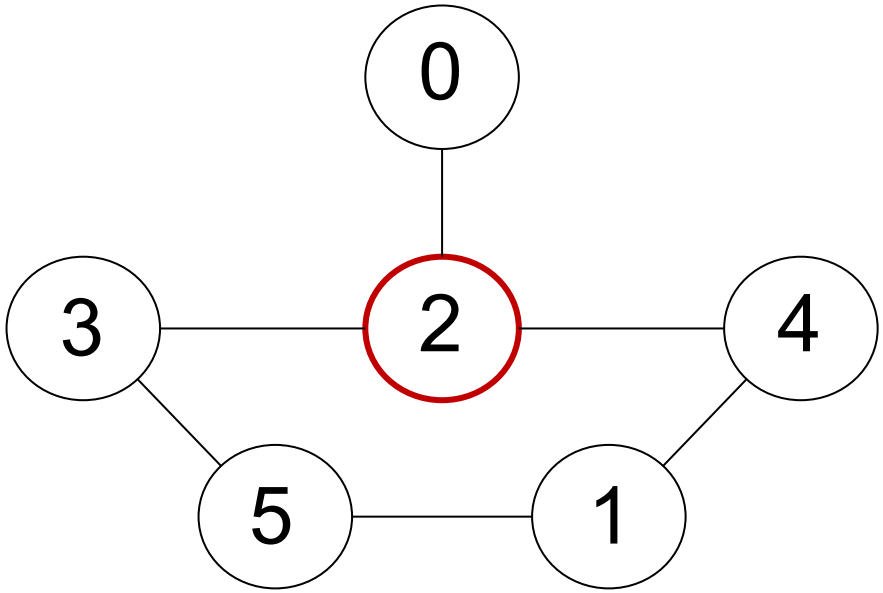
Verificar vizinhos de um vértice



0	0	1	0	0	0
0	0	0	0	1	1
1	0	0	1	1	0
0	0	1	0	0	1
0	1	1	0	0	0
0	1	0	1	0	0

Para encontrar os vizinhos de um vértice *v* devemos pensar onde suas possíveis arestas estão armazenadas.

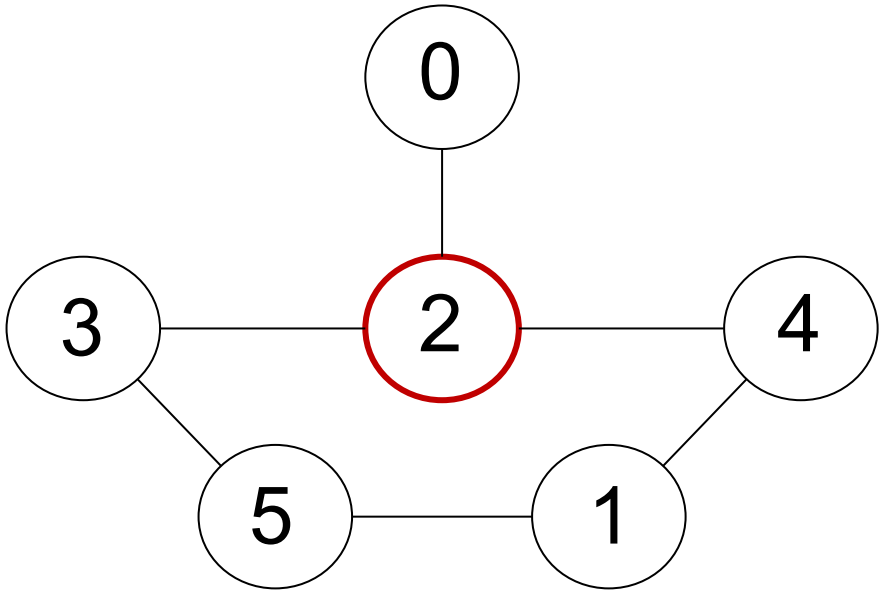
Verificar vizinhos de um vértice



0	0	1	0	0	0
0	0	0	0	1	1
1	0	0	1	1	0
0	0	1	0	0	1
0	1	1	0	0	0
0	1	0	1	0	0

Por exemplo vamos checar os vizinhos do vértice 2.

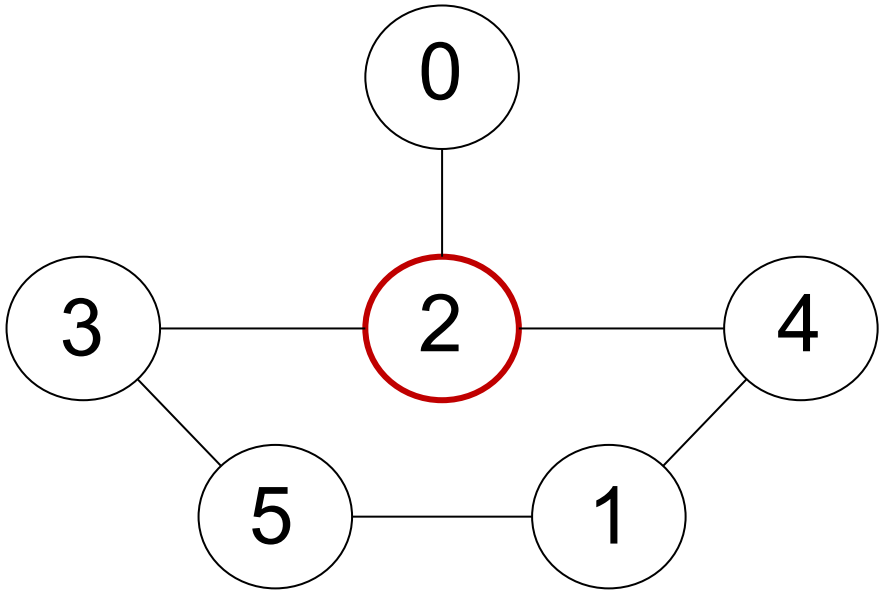
Verificar vizinhos de um vértice



0	0	1	0	0	0
0	0	0	0	1	1
1	0	0	1	1	0
0	0	1	0	0	1
0	1	1	0	0	0
0	1	0	1	0	0

Pela definição da matriz de adjacência as células A[2][j] indicam se existe aresta entre os vértices 2 e j.

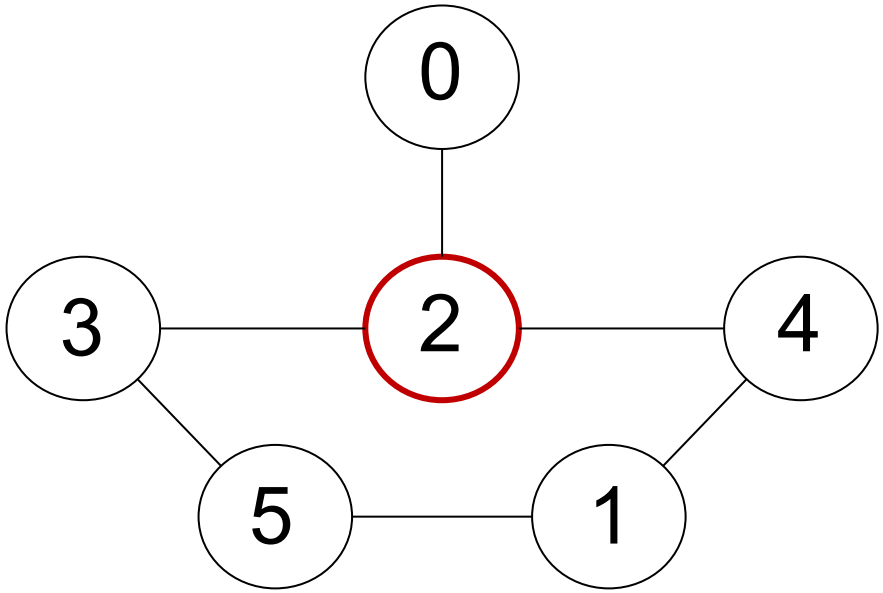
Verificar vizinhos de um vértice



0	0	1	0	0	0
0	0	0	0	1	1
1	0	0	1	1	0
0	0	1	0	0	1
0	1	1	0	0	0
0	1	0	1	0	0

O mesmo pode ser dito para as células da forma A[i][2].

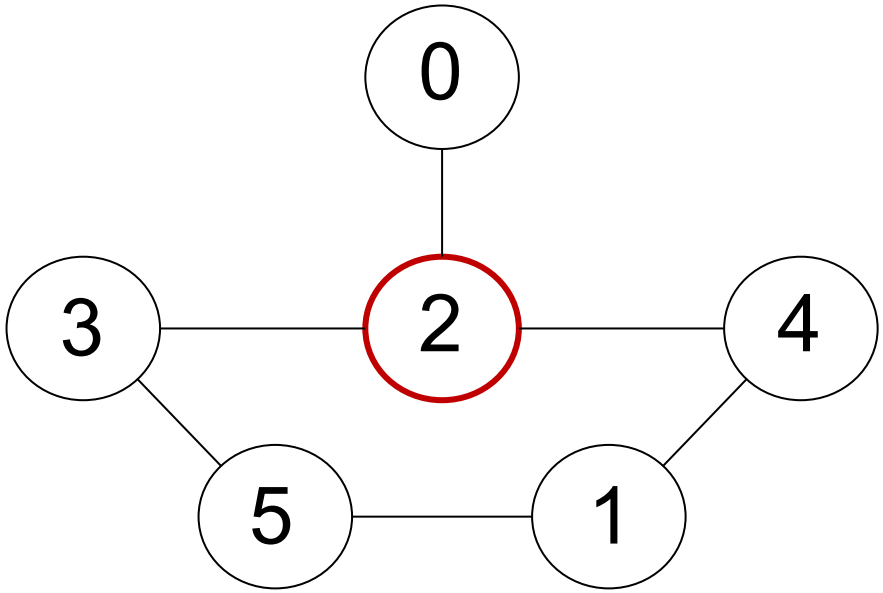
Verificar vizinhos de um vértice



0	0	1	0	0	0
0	0	0	0	1	1
1	0	0	1	1	0
0	0	1	0	0	1
0	1	1	0	0	0
0	1	0	1	0	0

O mesmo pode ser dito para as células da forma A[i][2]. No entanto pela forma que as matrizes são implementadas obtemos melhor localidade de referência utilizando as linhas.

Verificar vizinhos de um vértice



0	0	1	0	0	0
0	0	0	0	1	1
1	0	0	1	1	0
0	0	1	0	0	1
0	1	1	0	0	0
0	1	0	1	0	0

Conseguimos então computar os vizinhos de um vértice em tempo $\theta(n)$.

Matriz de adjacência

Vantagens:

- Acesso rápido a arestas. A existência de uma aresta pode ser conferida em $O(1)$.

Desvantagens:

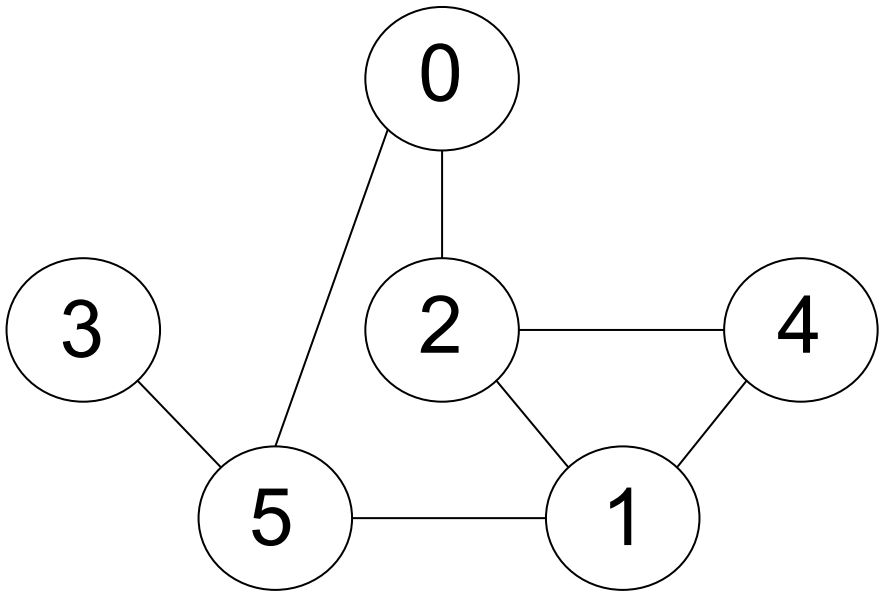
- Consultar a vizinhança de um vértice é feita em $\theta(n)$, pois sempre será necessário varrer a linha toda da matriz.
- Ocupa espaço $O(n^2)$ independente de quantas arestas o grafo possui.

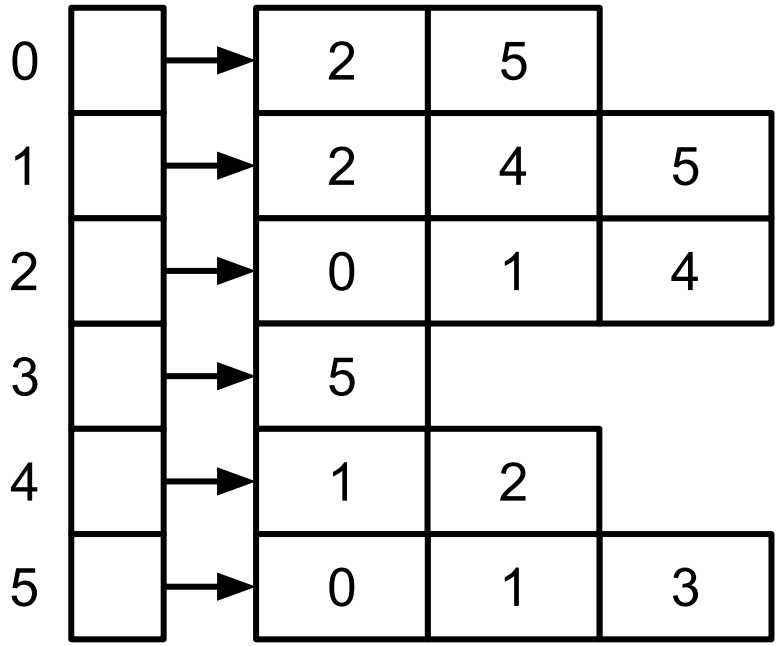
Lista de adjacência

Esta outra representação consiste em uma lista de listas. Teremos uma lista para cada vértice, contendo seus vizinhos.

Lista de adjacência

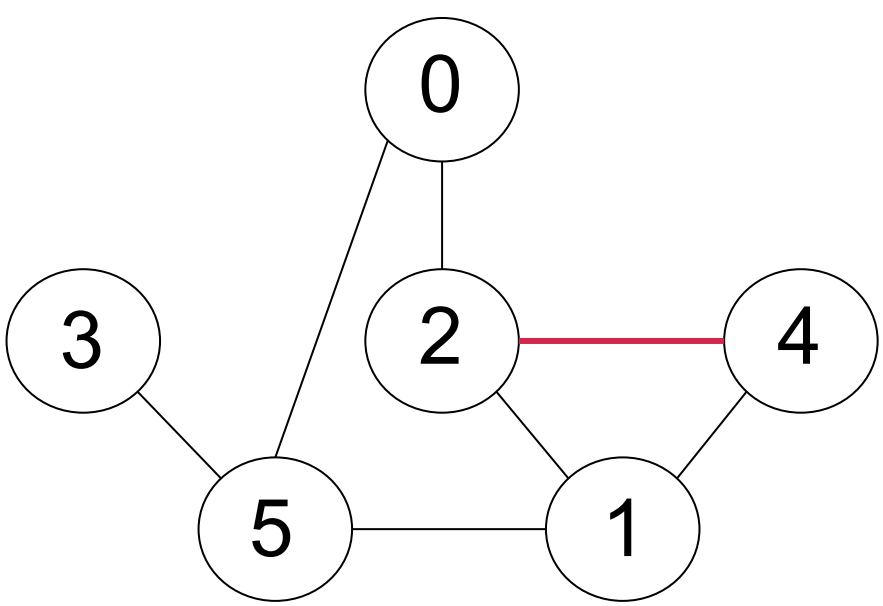
Esta outra representação consiste em uma lista de listas. Teremos uma lista para cada vértice, contendo seus vizinhos.

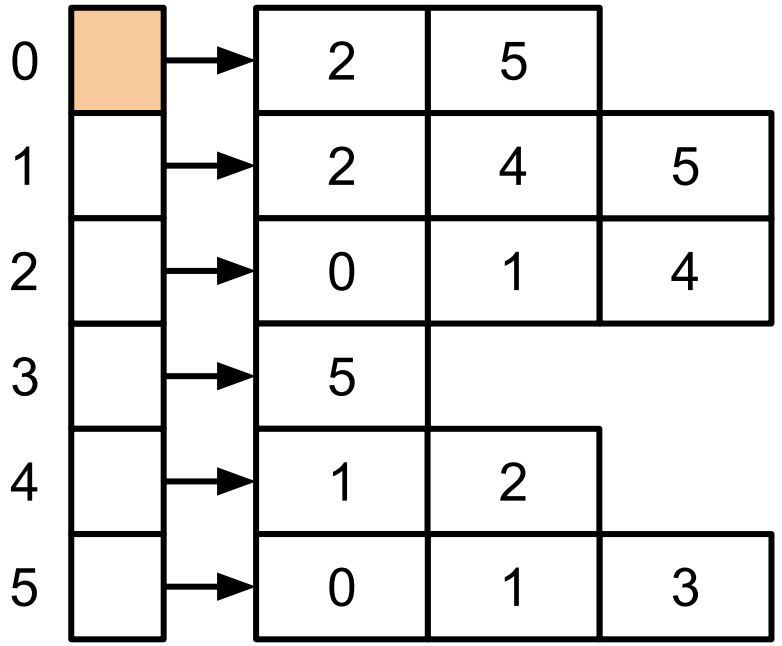




Existência de aresta

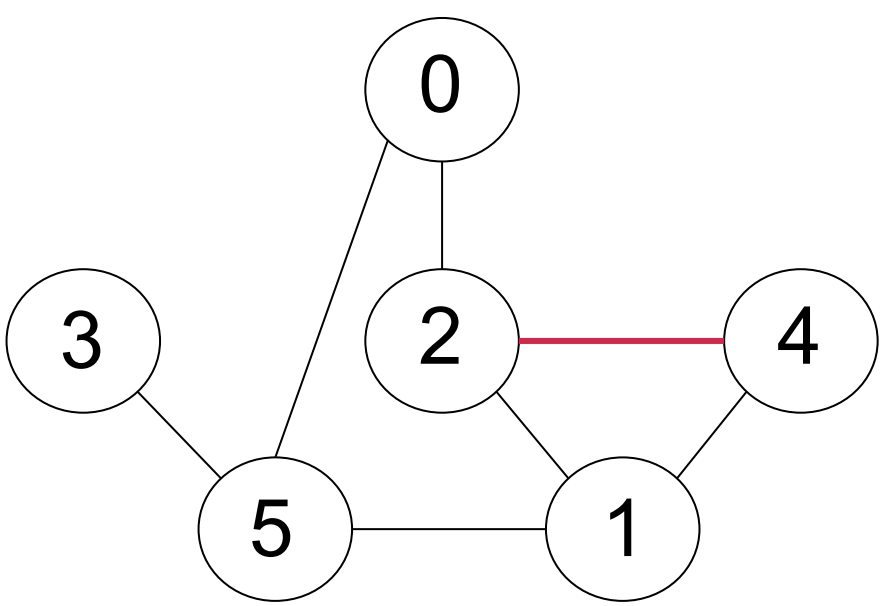
Para verificar a existência de uma aresta agora é necessário percorrer as listas.

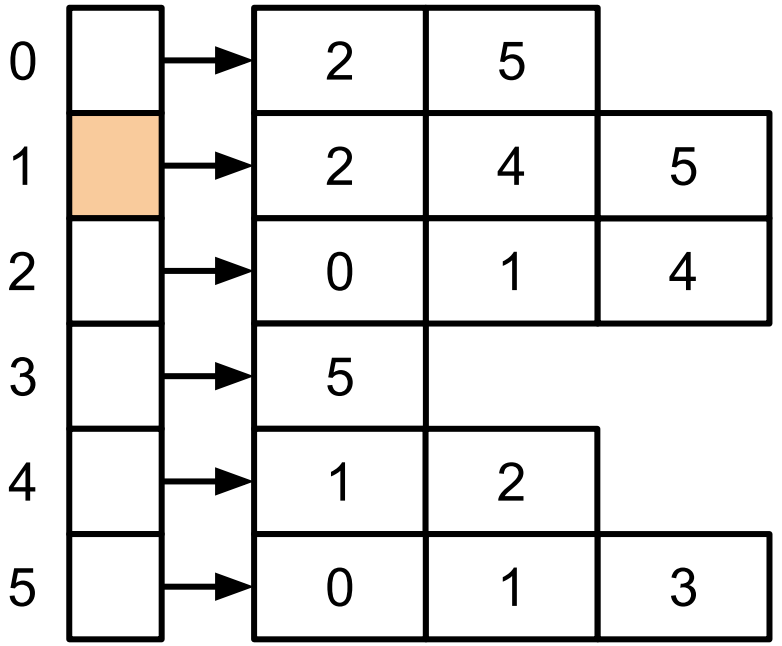




Existência de aresta

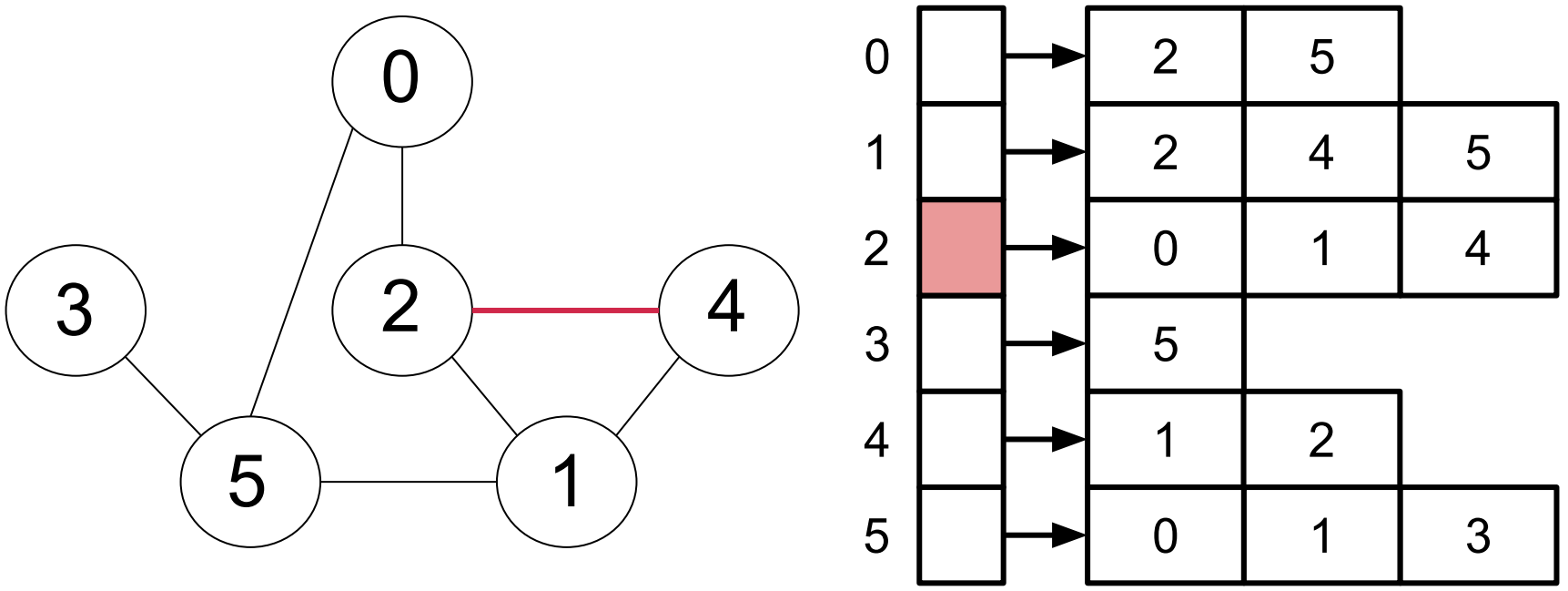
Para verificar a existência de uma aresta agora é necessário percorrer as listas.





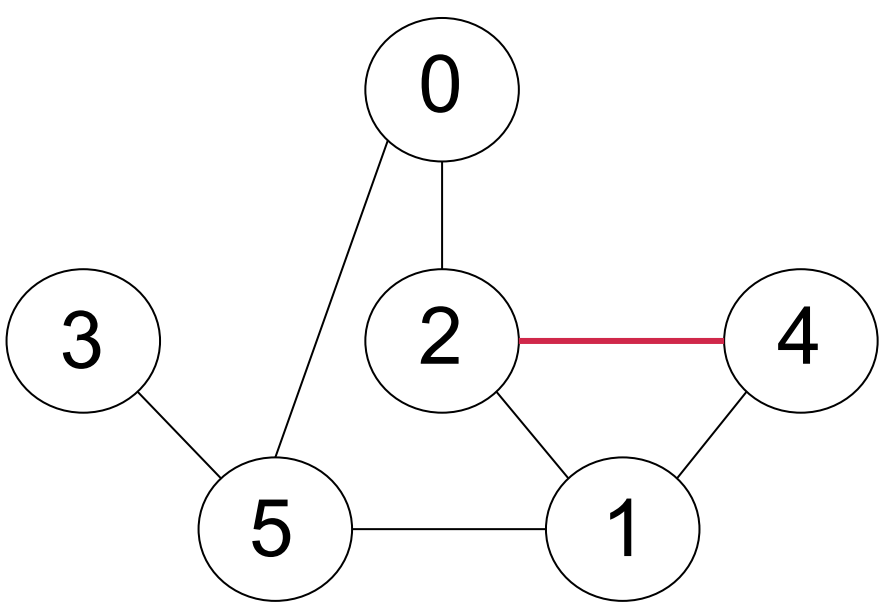
Existência de aresta

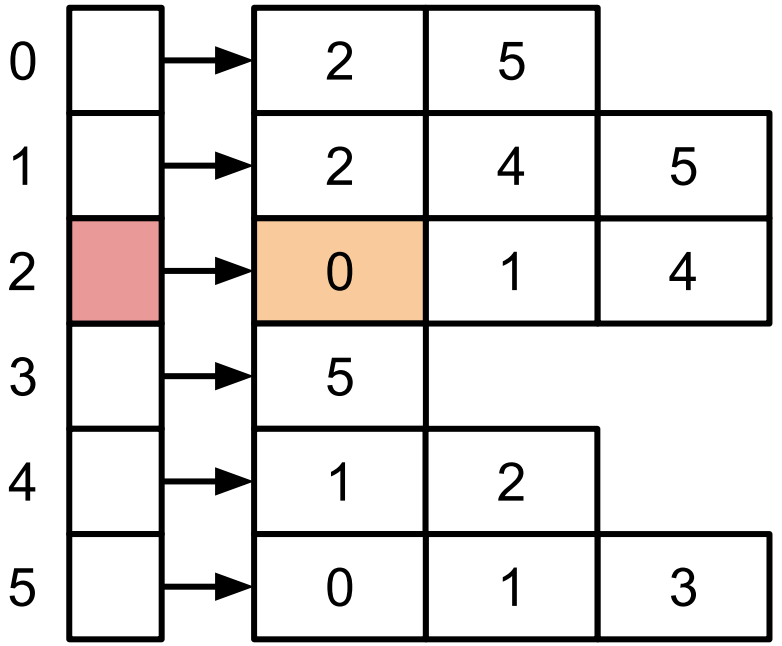
Para verificar a existência de uma aresta agora é necessário percorrer as listas.



Existência de aresta

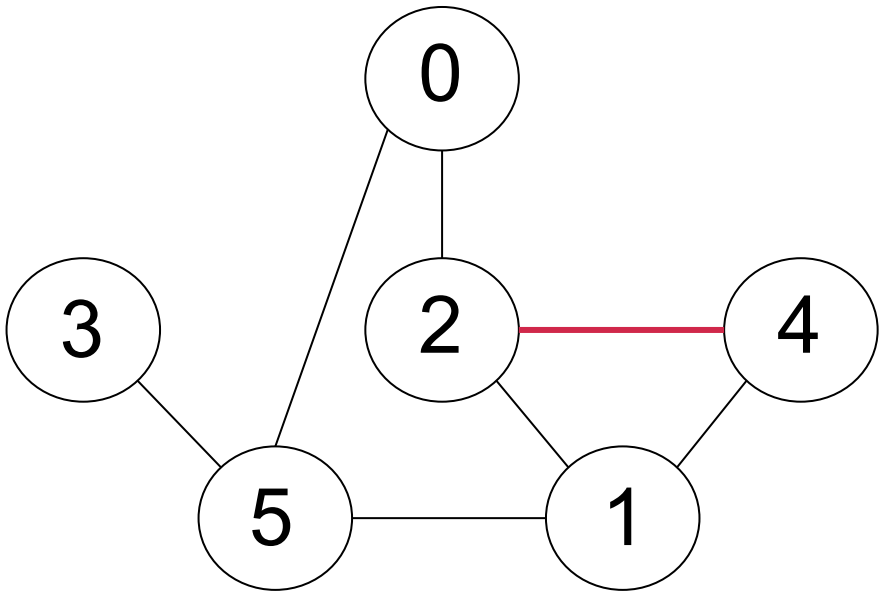
Para verificar a existência de uma aresta agora é necessário percorrer as listas.

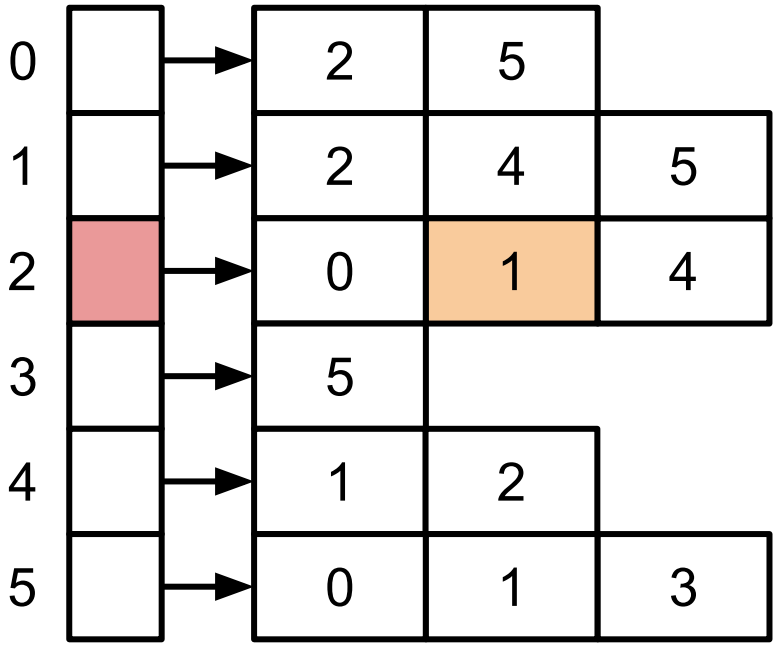




Existência de aresta

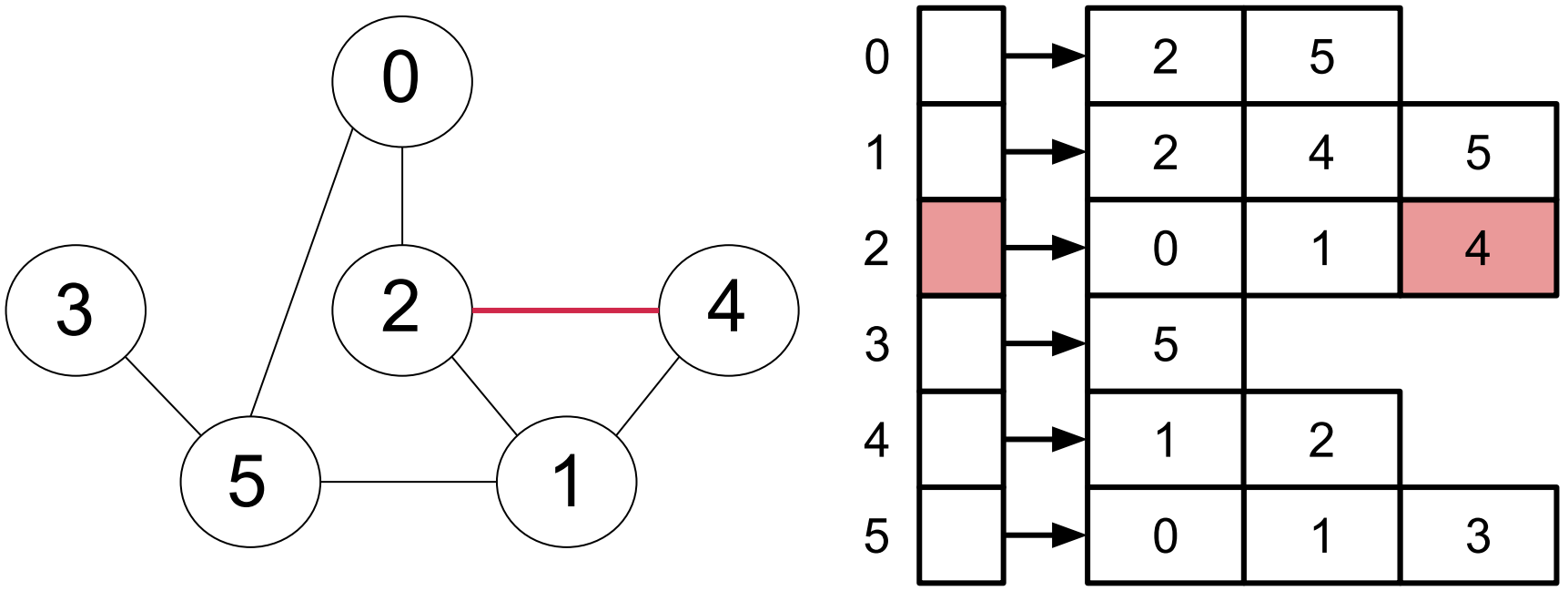
Para verificar a existência de uma aresta agora é necessário percorrer as listas.





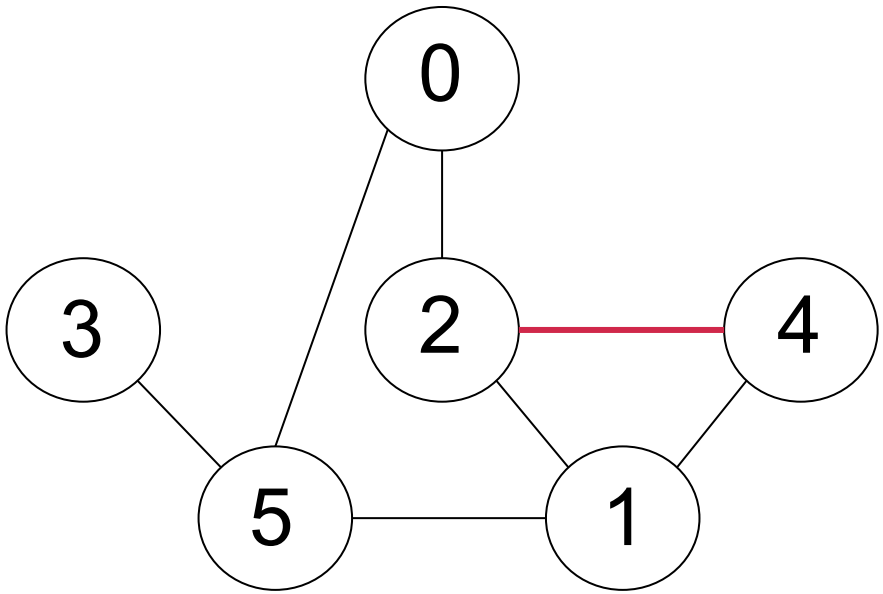
Existência de aresta

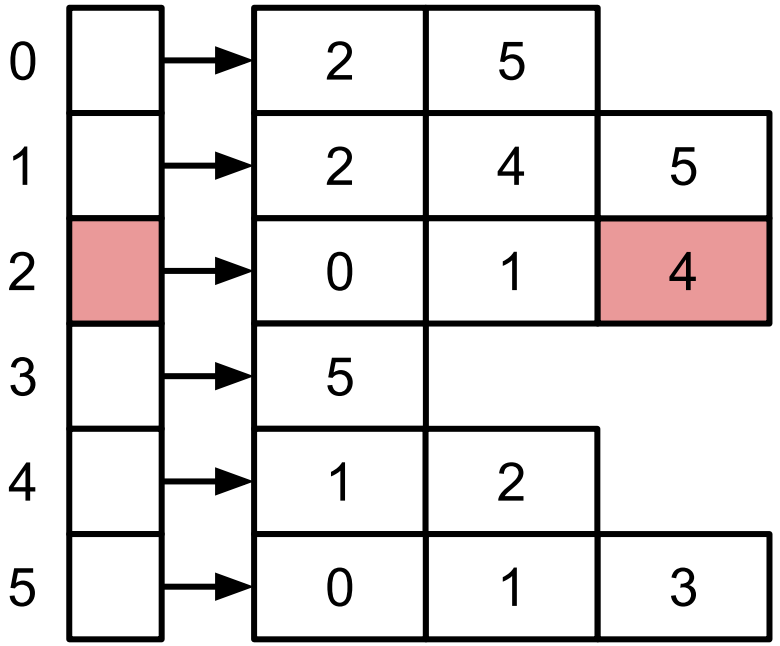
Para verificar a existência de uma aresta agora é necessário percorrer as listas.



Existência de aresta

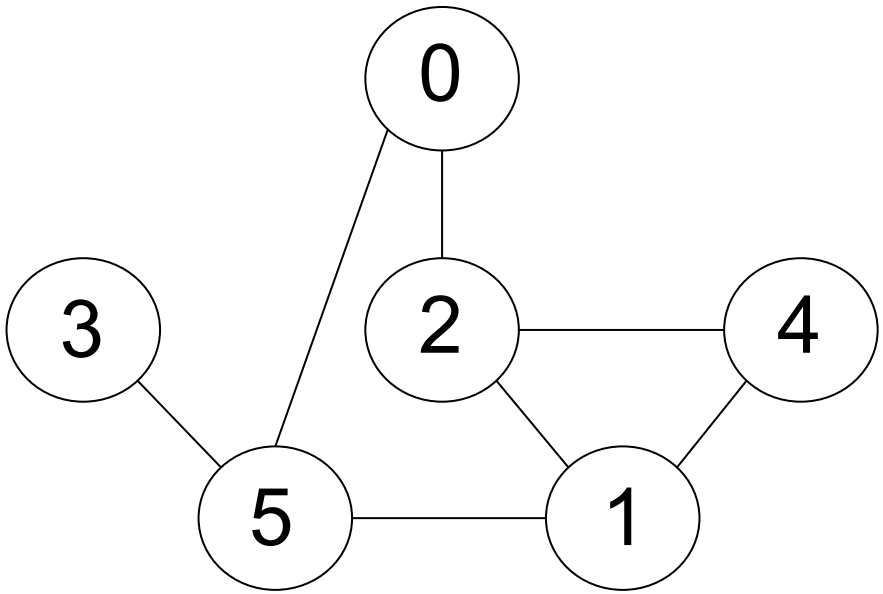
Se as listas forem implementadas como listas encadeadas esse processo é realizado em tempo *O(n)*.

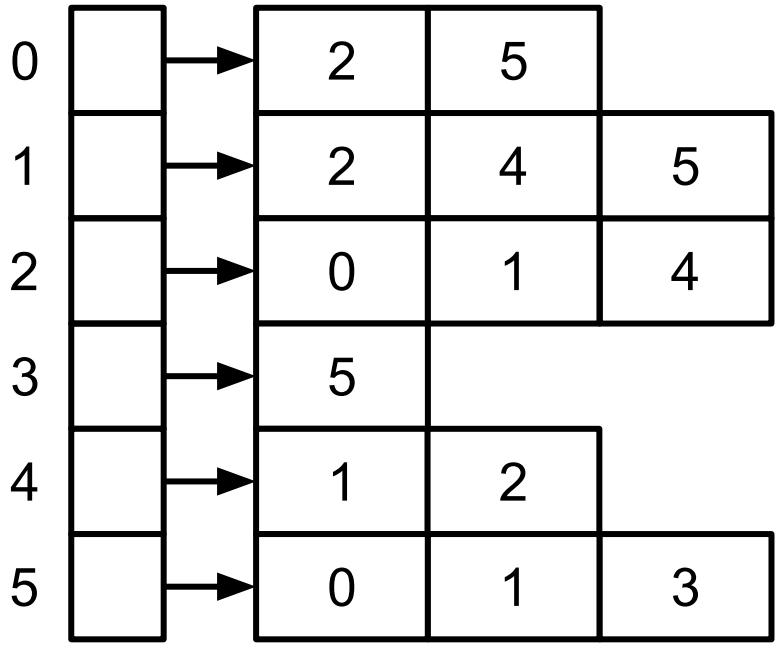




Vizinhança de um vértice

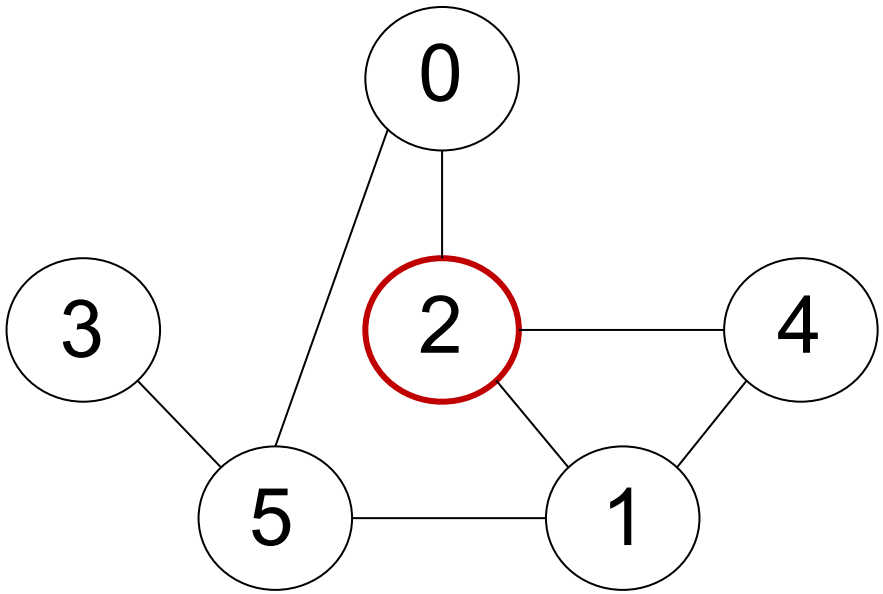
Pela própria construção da lista de adjacência, basta encontrarmos a lista referente ao vértice que desejamos.

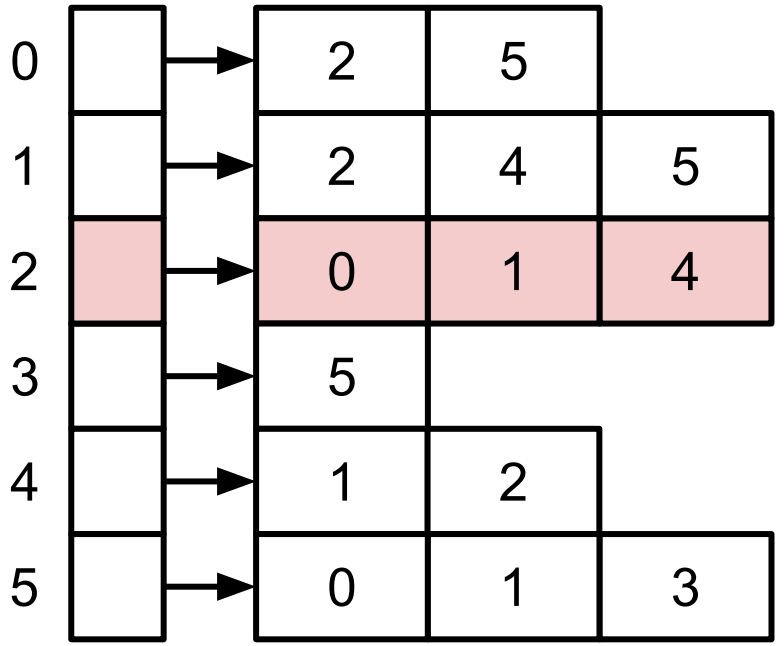




Vizinhança de um vértice

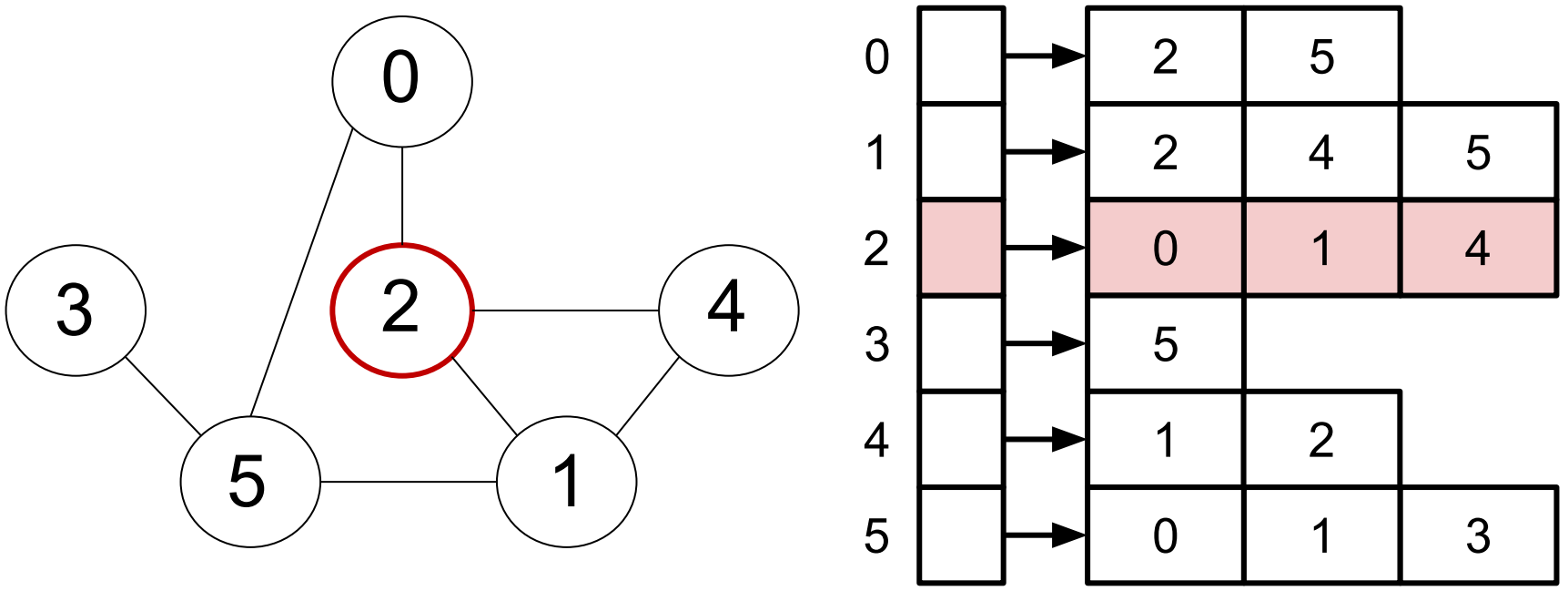
Pela própria construção da lista de adjacência, basta encontrarmos a lista referente ao vértice que desejamos.





Vizinhança de um vértice

Conseguimos computar os vizinhos de um vértice em tempo *O(n)*.



Lista de adjacência

Vantagens:

- Ocupa espaço $O(\ |V|+|E|\ )$. Note que |E| é $O(n^2)$, então no pior caso a lista ocupa espaço proporcional a de uma matriz de adjacência.
- Apesar de termos complexidade assintótica semelhante a matriz de adjacência, conseguimos tempos de execução significativamente menores quando o grafo possui poucas arestas.

Desvantagens:

- Verificar a existência de uma aresta custa $O(n)$.