ESD2

Estrutura de Dados

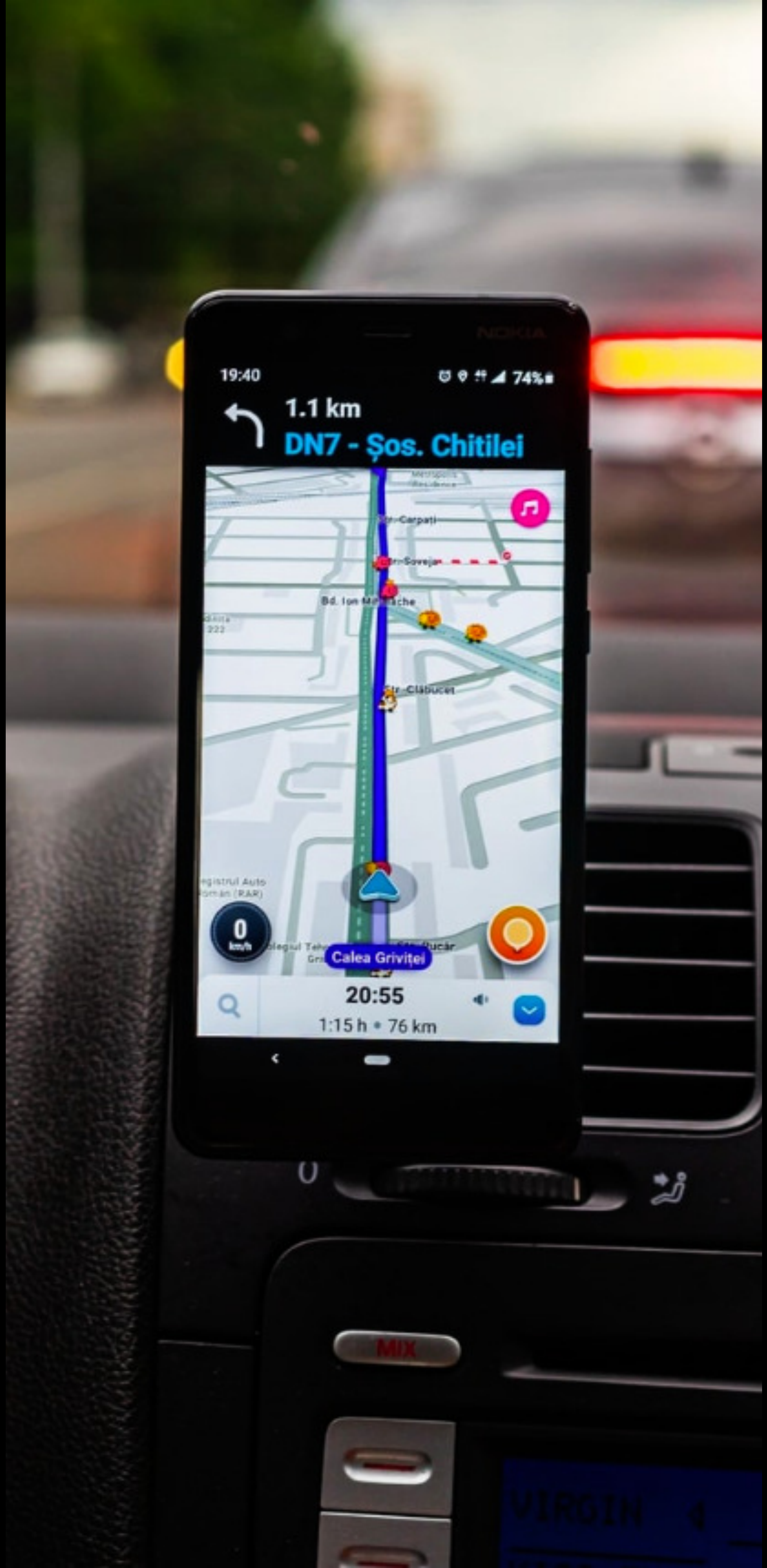
Introdução à Teoria de Grafos

Prof. Fernando Sambinelli

INSTITUTO FEDERAL
São Paulo

Motivação

- Muitas aplicações em computação necessitam considerar conjunto de conexões entre pares de objetos:
  - Existe um caminho para ir de um objeto a outro seguindo as conexões?
  - Qual é a menor distância entre um objeto e outro objeto?
  - Quantos outros objetos podem ser alcançados a partir de um determinado objeto?
- Existe um tipo abstrato da dados chamado **grafo** que é usado para modelar tais situações
  - Conceito introduzido por Euler, em 1736



Exemplos de Aplicações

ESD 2
Estrutura de Dados

Roteamento de Pacotes

Encontrar o caminho mais eficiente para enviar dados entre dispositivos

Análise de Conectividade

Estudar como indivíduos estão conectados e como as informações se propagam.

Ciência de Dados e Machine Learning

Modelar problemas em big data

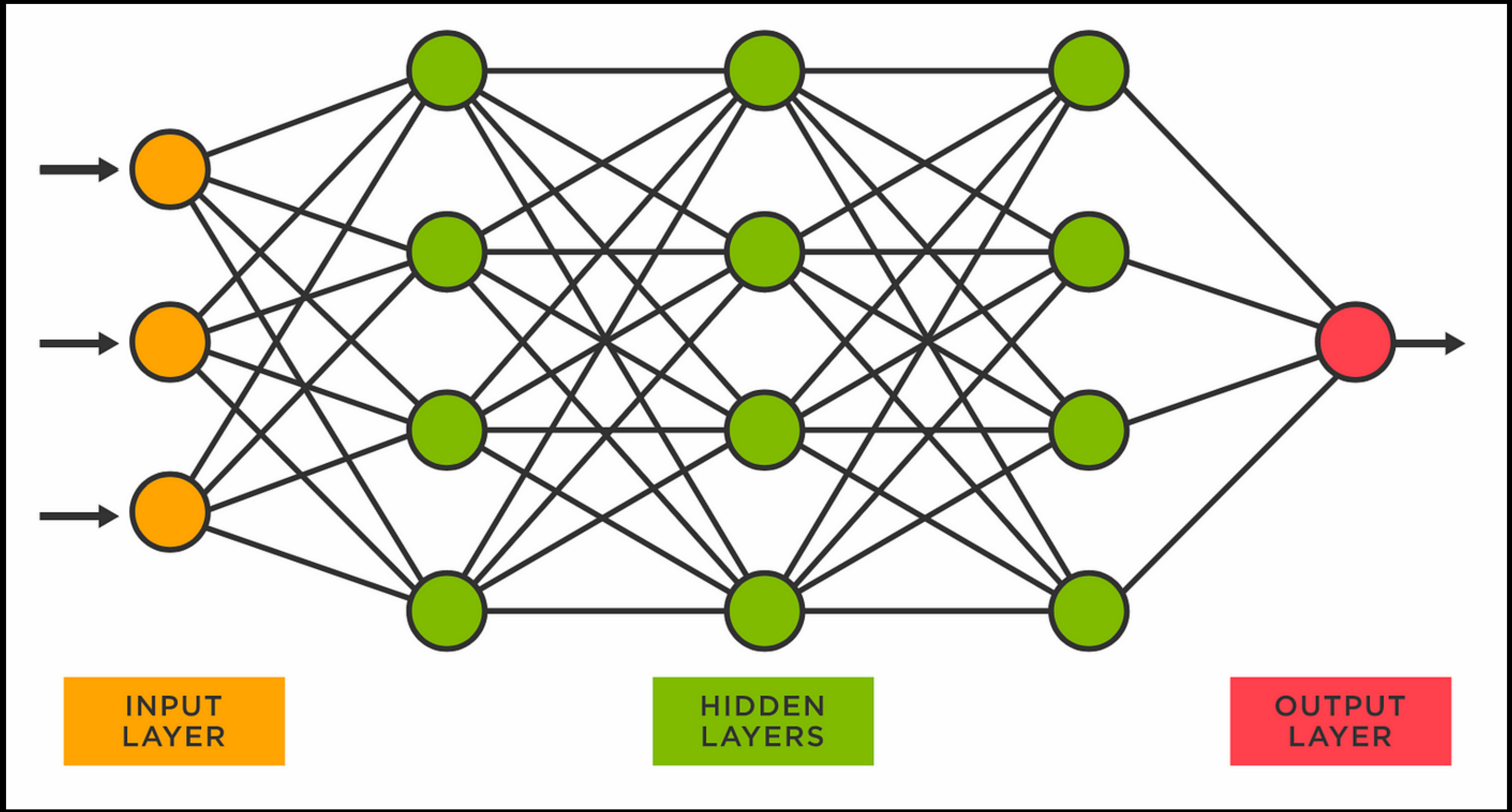
Modelagem de Sistemas

Utilizar grafos para modelar e analisar sistemas complexos

Exemplos de Aplicações

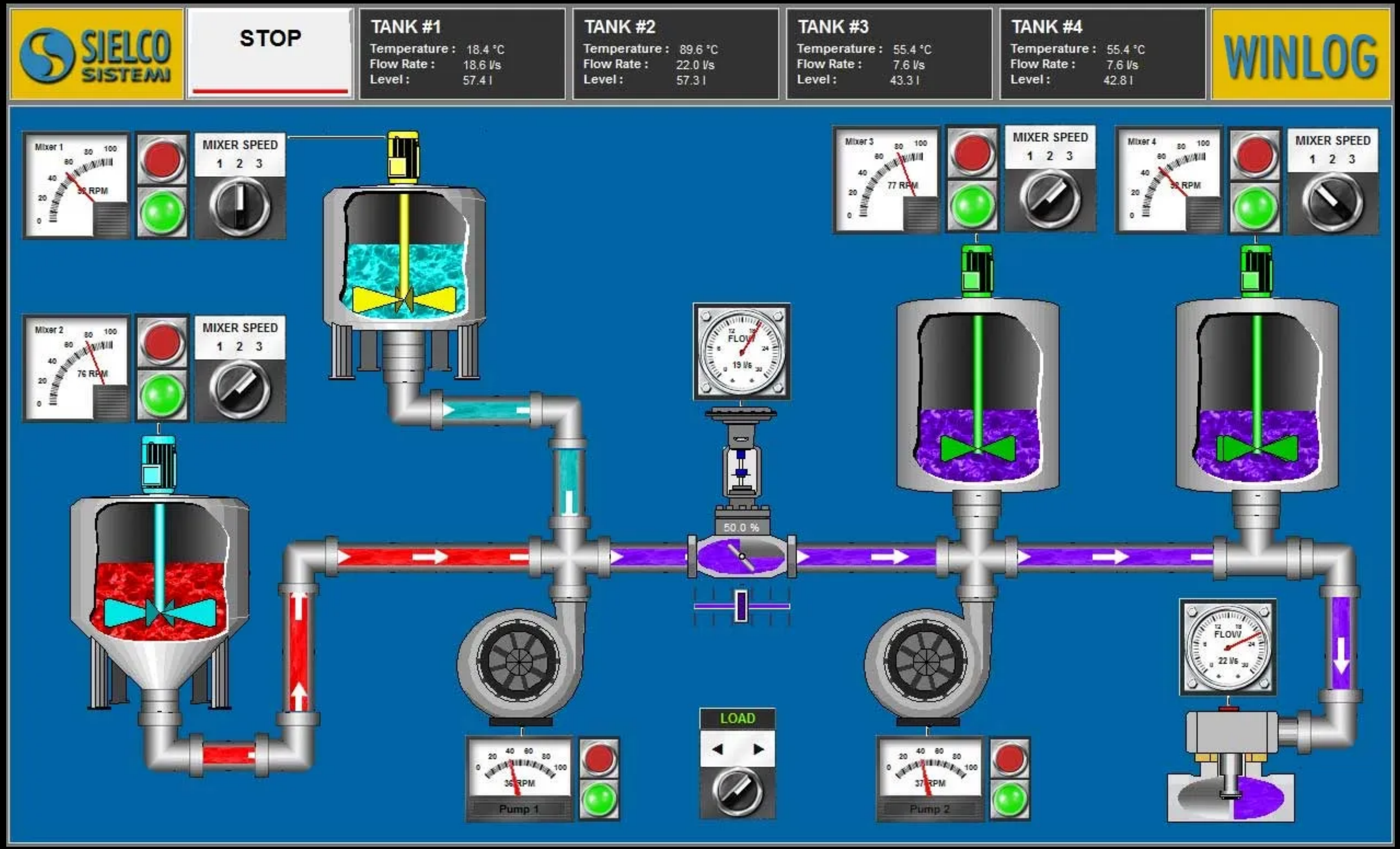
Inteligência Artificial

Rede Neural



Exemplos de Aplicações

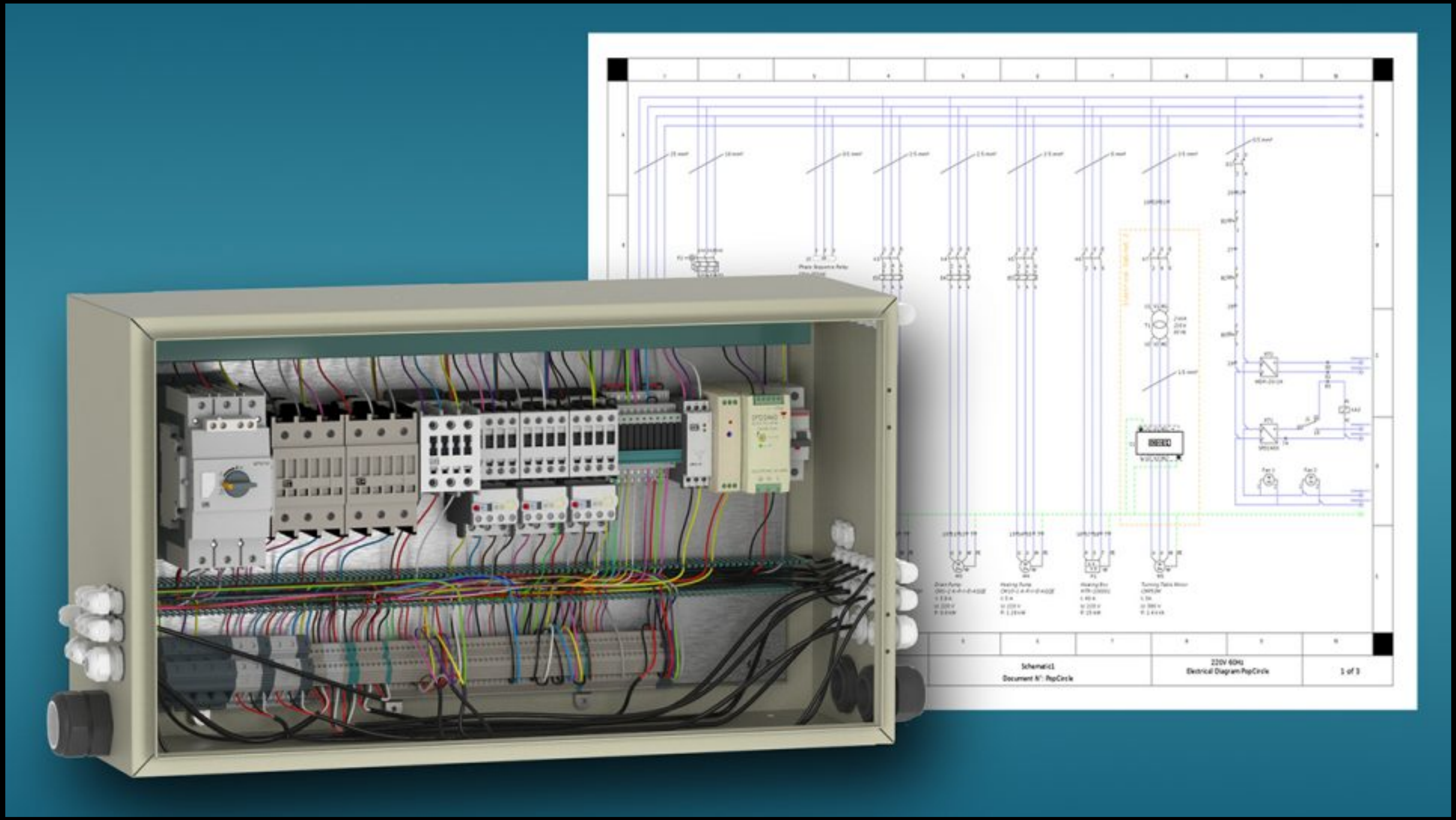
Sistema Supervisório



Exemplos de Aplicações

ESD 2
Estrutura de Dados

Simulação de Circuitos Eletrônicos



Conceitos Básicos

ESD 2
Estrutura de Dados

1 Grafo

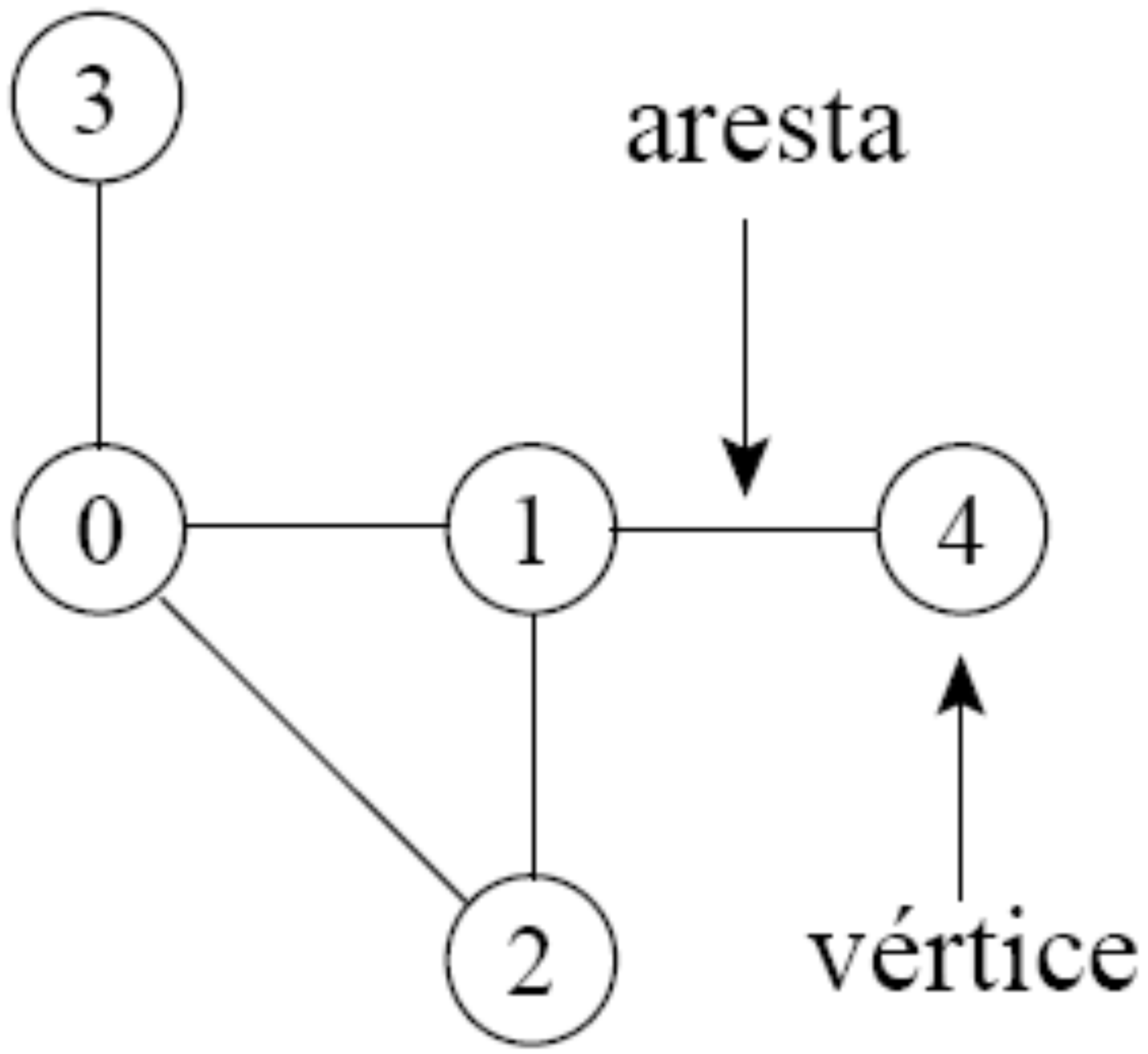
Conjunto de vértices e arestas

2 Vértice

Objeto simples que pode ter nome e outros atributos

3 Aresta

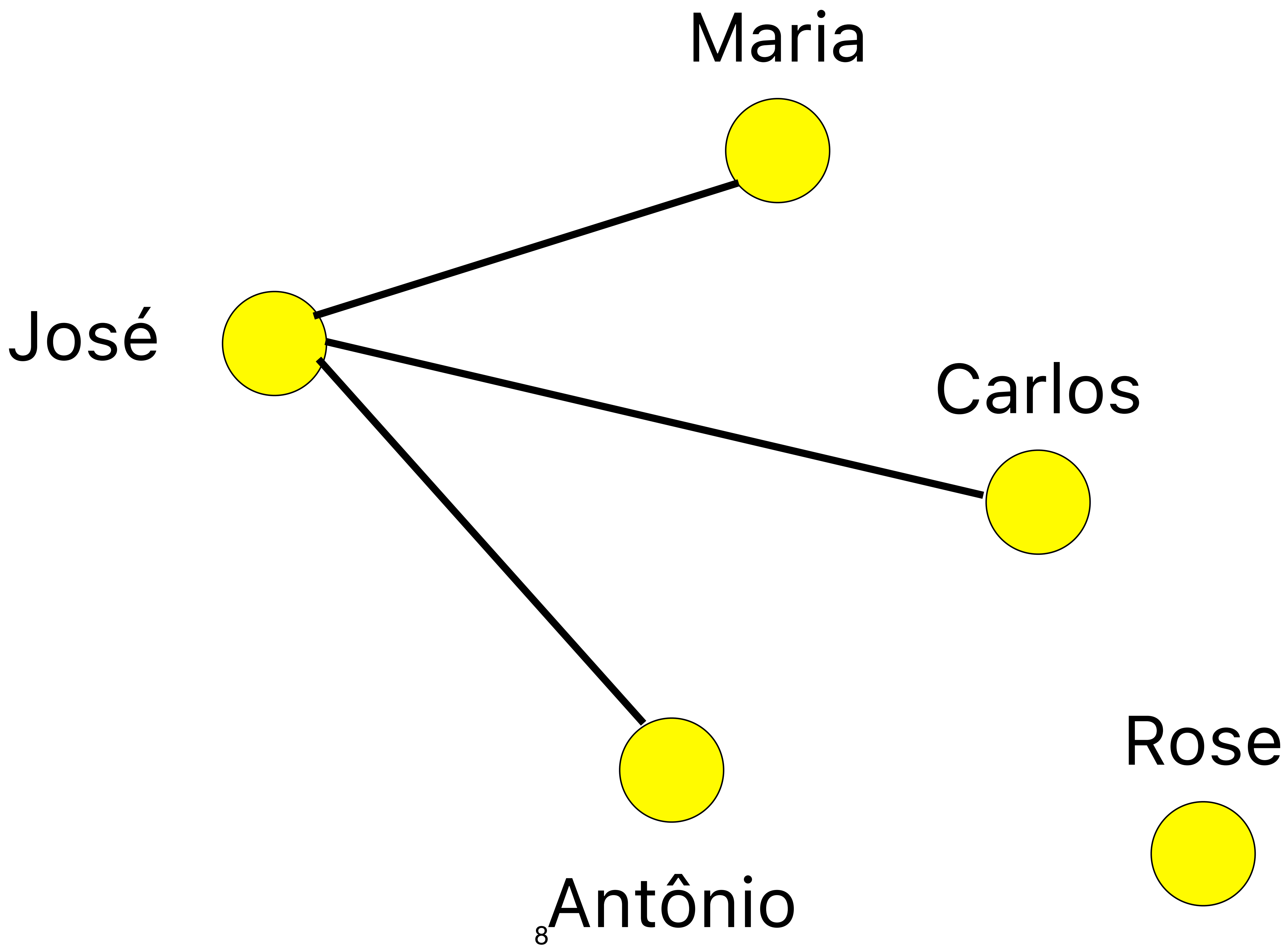
Conexão entre dois vértices



Notação: *G = (V,A)*
G: grafo
V: conjunto de vértices
A: conjunto de arestas

Conceitos Básicos

- Vamos imaginar que, em um grafo, os vértices representam pessoas e as arestas representam a relação de 'amizade'



Diversas Classificações dos Grafos

Porque?

Especificar suas Propriedades

Simplificação da Análise

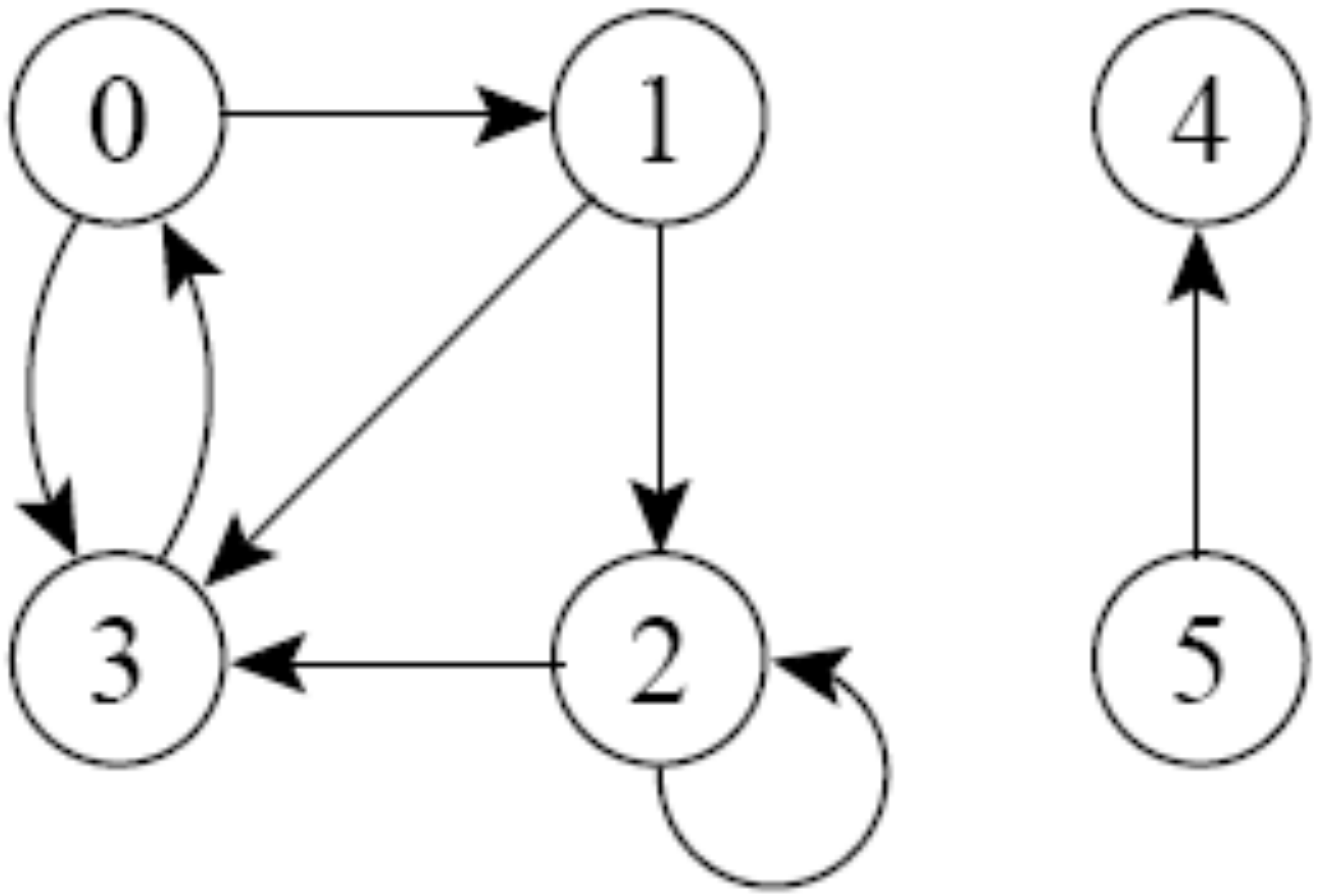
Adequação a Problemas Específicos

Facilidade Compreesão

Grafos Direcionados

Dígrafos

- Um grafo direcionado G é um par (V,A), onde V é um conjunto finito de vértices e A é uma relação binária em V.
- Uma aresta (u, v) sai do vértice u e entra no vértice v
    - *O vértice v é **adjacente** ao vértice u*
- Podem existir arestas de um vértice para ele mesmo, chamadas de self-loops.



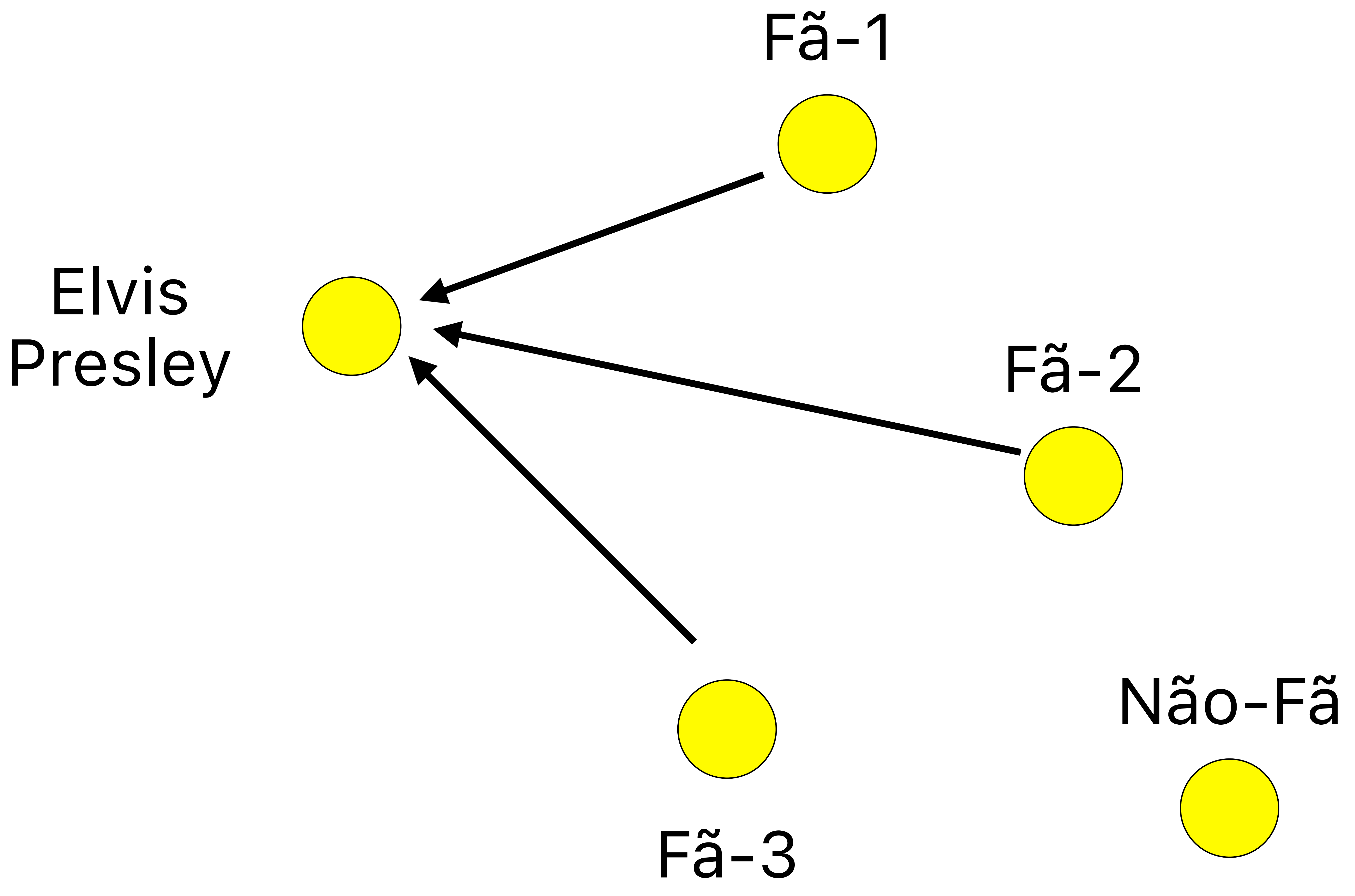
Self-loop

Grafos Direcionados

ESD 2
Estrutura de Dados

Dígrafos

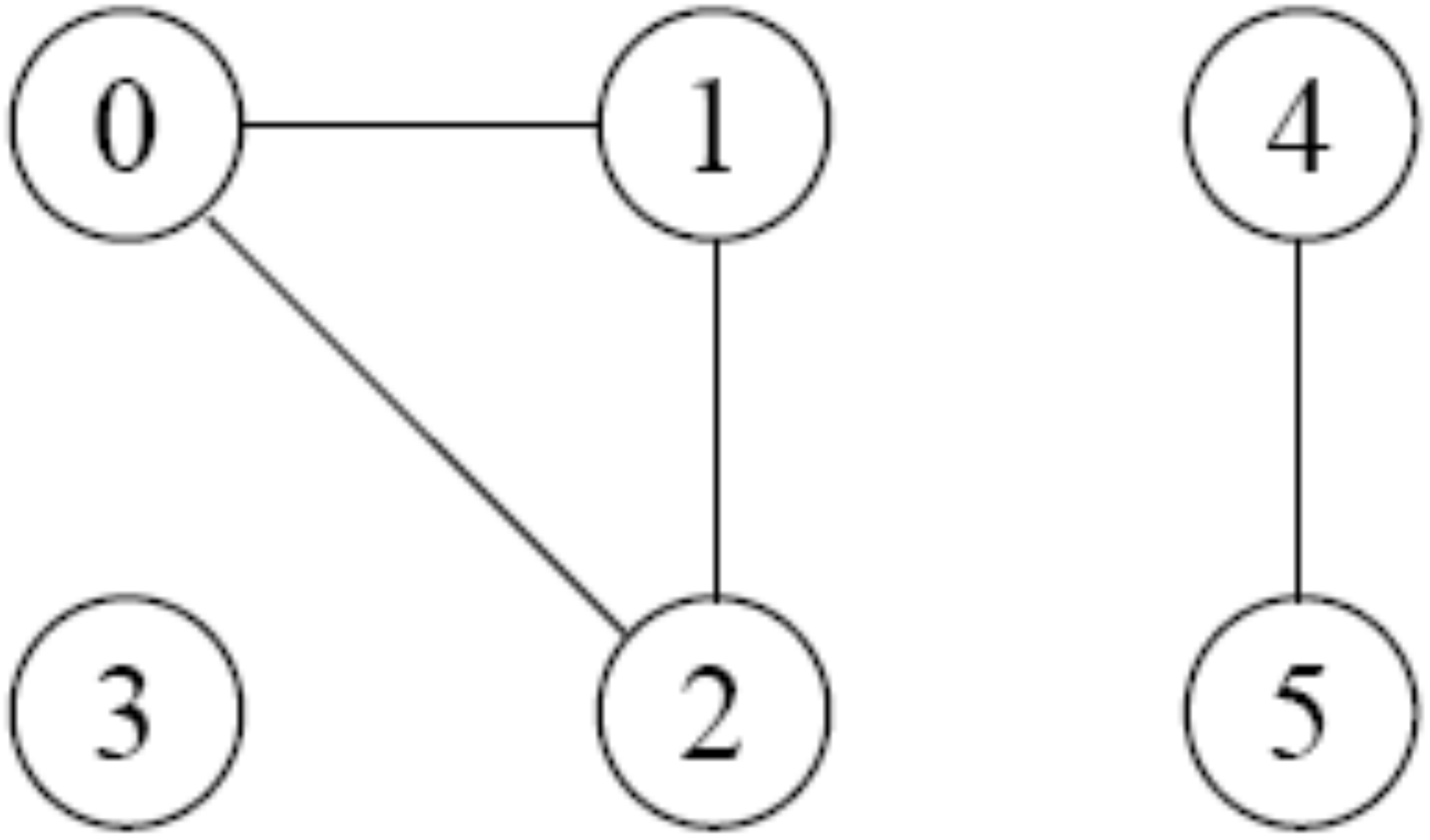
Sou fã de...



Grafos Não Direcionados

E S D 2
Estrutura de Dados

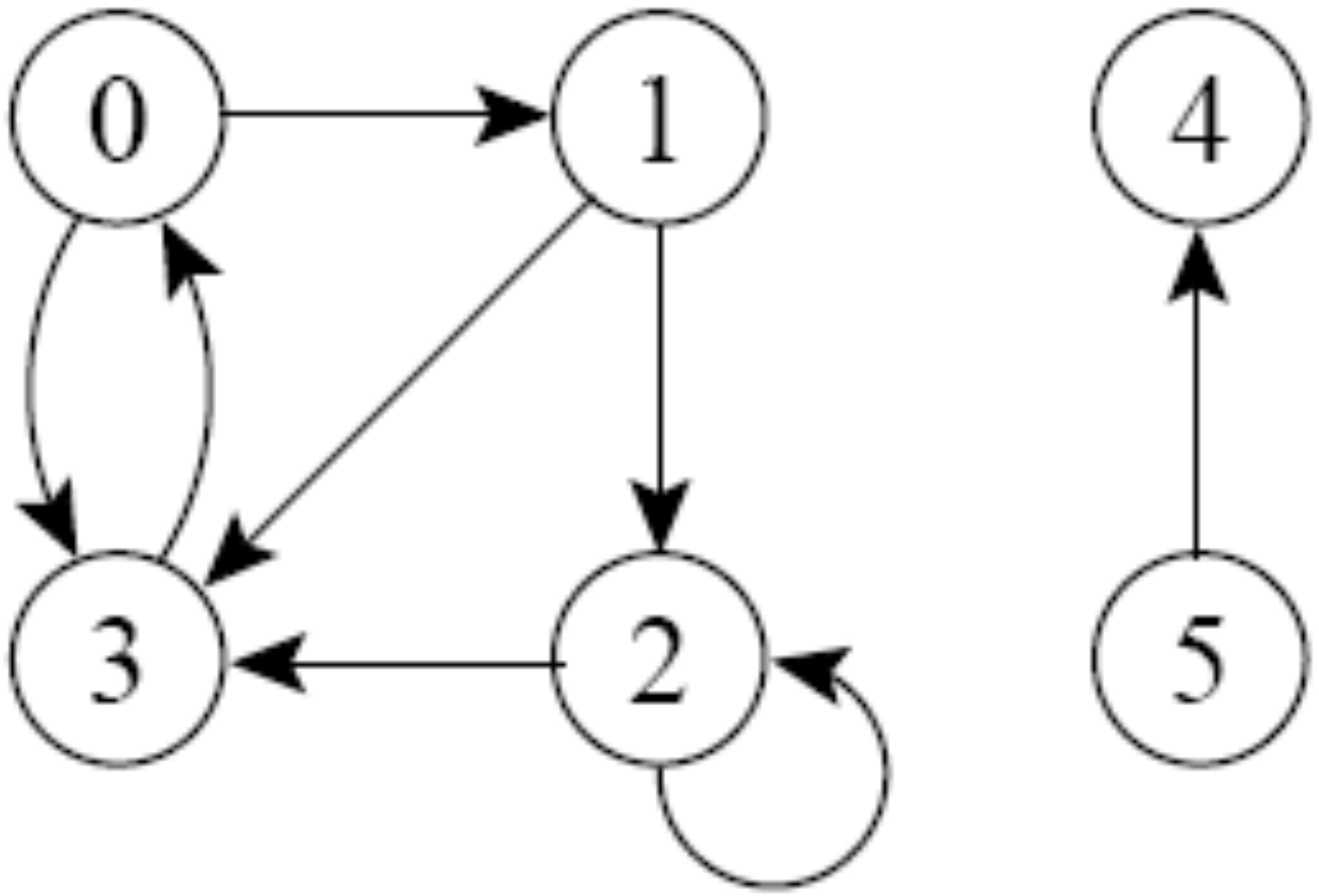
- Um grafo não direcionado G é um par (V,A), onde o conjunto de arestas A é constituído de pares de vértices não ordenados.
- As arestas (u, v) e (v, u) são consideradas como uma única aresta.
- A relação de adjacência é simétrica self-loops não são permitidos.



Grau de Vértice

Em grafos direcionados

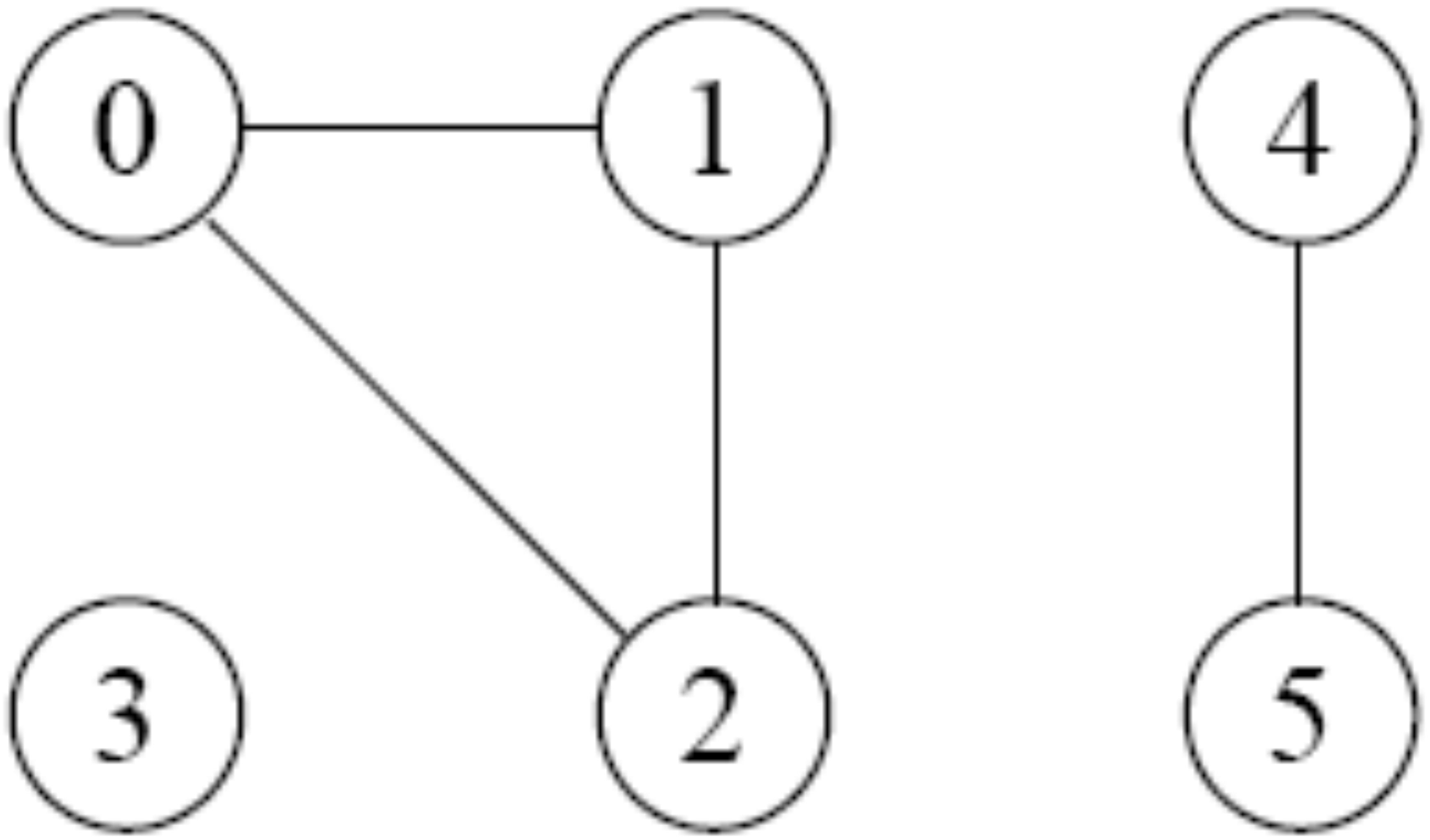
- O grau de um vértice é o número de arestas que saem dele (out-degree) mais o número de arestas que chegam nele (in-degree)
- Ex.: O vértice 2 tem in-degree 2, out-degree 2 e grau 4



Grau de Vértice

Em grafos não direcionados

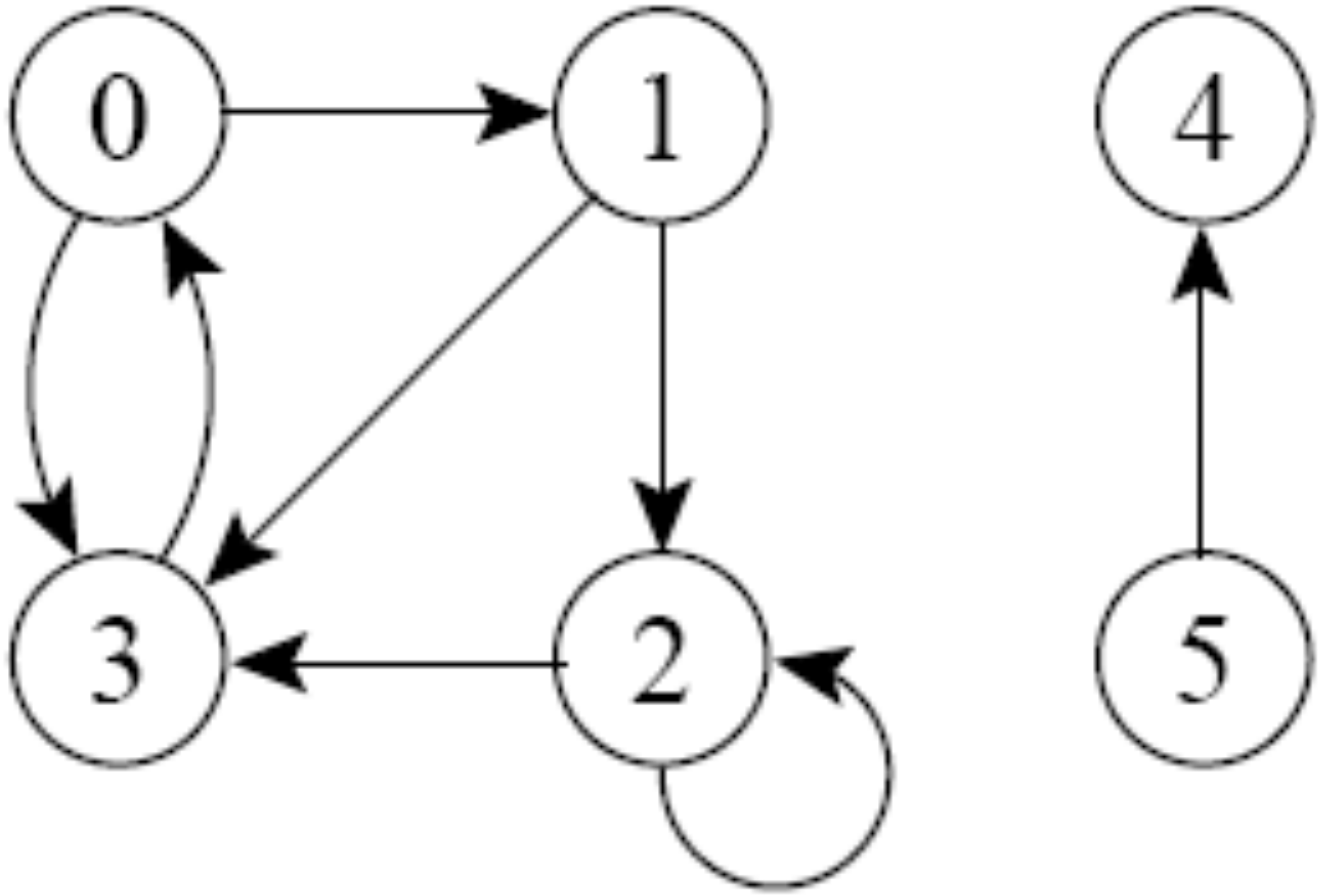
- O grau de um vértice é o número de arestas que incidem ("chegam") nele
- Um vértice de grau zero é dito isolado ou não conectado
- Ex.: O vértice 1 tem grau 2 e o vértice 3 é isolado



Caminho entre Vértices

ESD 2
Estrutura de Dados

- Um caminho de comprimento k entre um vértice x e um vértice y em um grafo G = (V,A) é uma sequência de vértices *(v0, v1, v2, … , vk)* tal que *x = v0* e *y = vk*, e *(vi-1, vi)* Є A para *i = 1, 2, … , k*.
- **O comprimento de um caminho é o número de arestas nele**, isto é, o caminho contém os vértices *v0, v1, v2, …, vk* e as arestas *(v0, v1), (v1, v2), …, (vk-1, vk)*
- Se existir um caminho c entre x a y, então y é **alcançável** a partir de x via c
- Um **caminho é simples** se todos os vértices do caminho são distintos
- Ex.: O caminho (0, 1, 2, 3) é simples e tem comprimento 3. O caminho (1, 3, 0, 3) não é simples

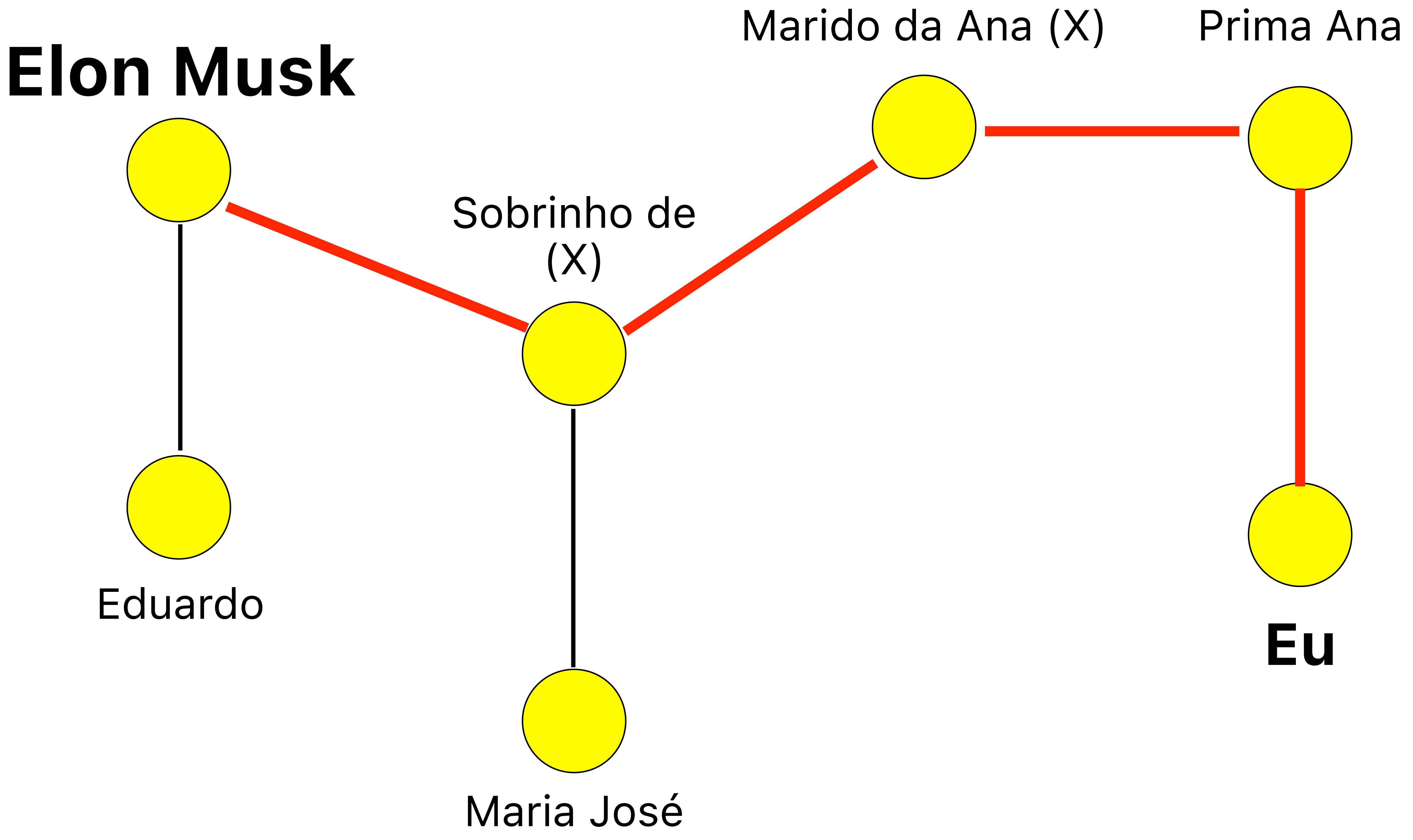


Caminho entre Vértices

ESD 2
Estrutura de Dados

- Um caminho é uma sequência de arestas que ligam dois vértices

Grafo de Amizade com CEO "Famoso"

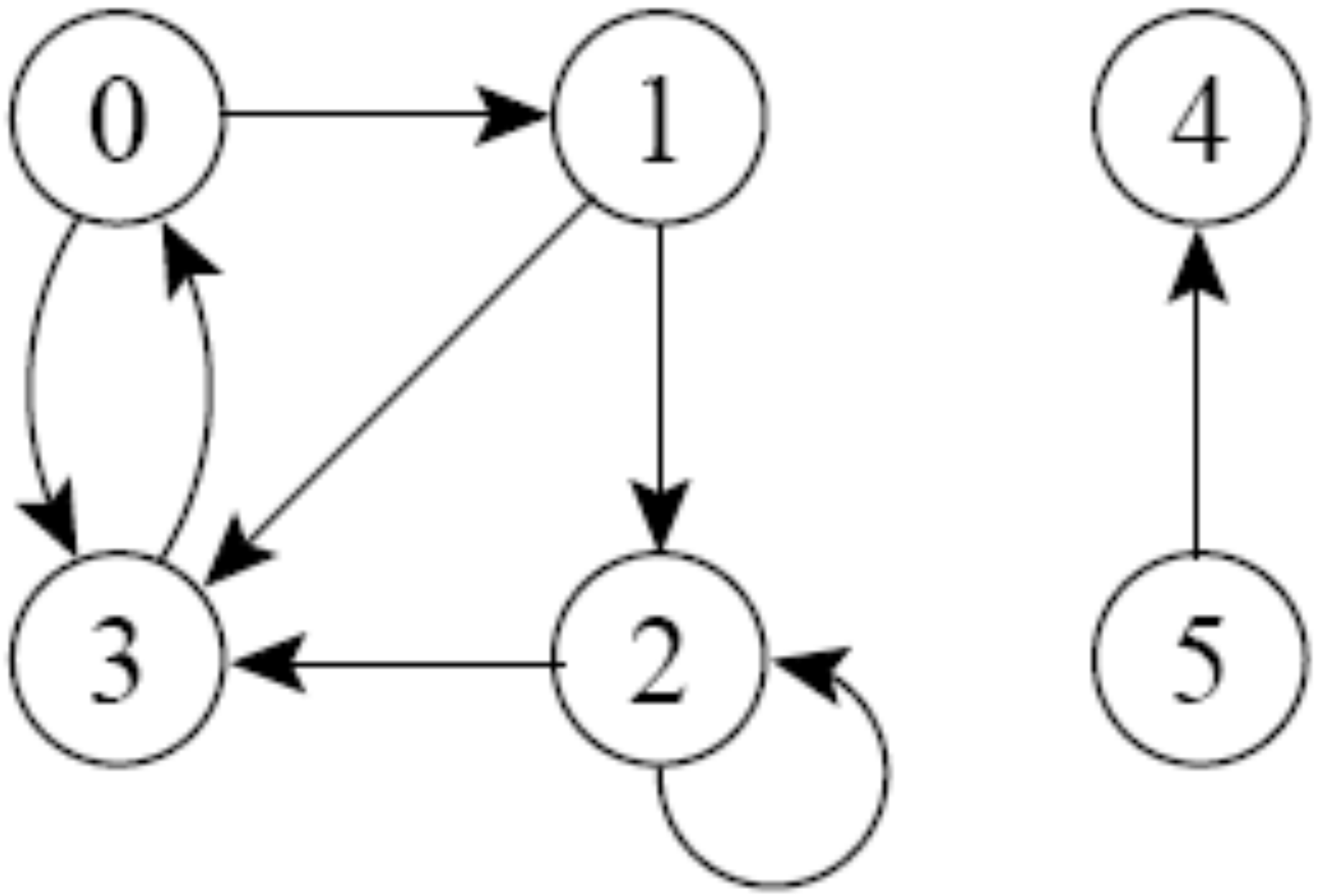


Ciclo

Em grafos direcionados

E S D 2
Estrutura de Dados

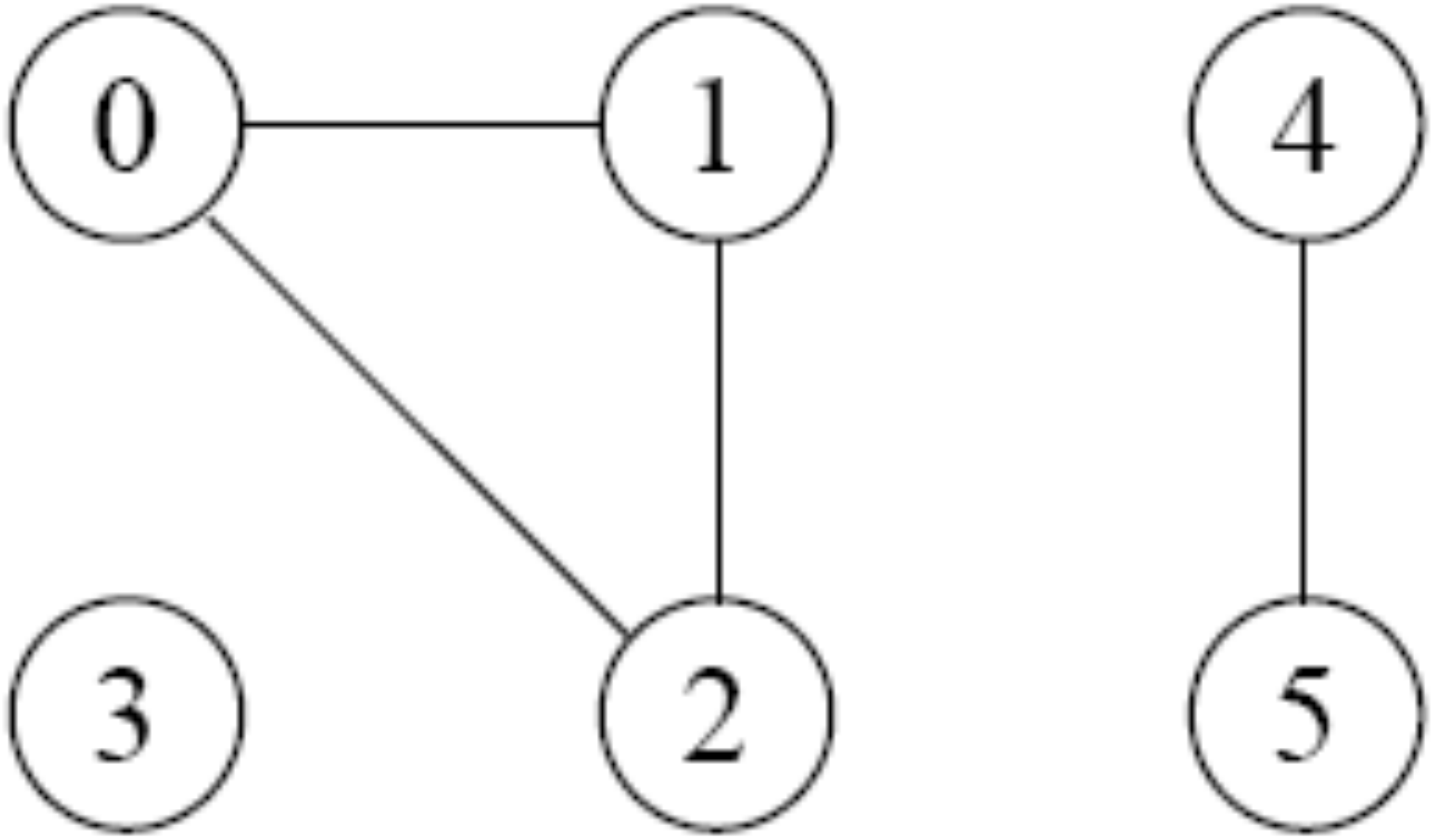
- Um caminho (v0, v1, ... , vk) forma um ciclo se v0 = vk e contém ao menos um outro vértice além de v0 e vk
- O ciclo é simples se os vértices v1, v2, ... , vk são distintos
- O self-loop é um ciclo de tamanho 1
- Ex.: O caminho (0, 1, 2, 3, 0) forma um ciclo. O caminho (0, 1, 3, 0) forma o mesmo ciclo que os caminhos (1, 3, 0, 1) e (3, 0, 1, 3)



Ciclo

Em grafos não direcionados

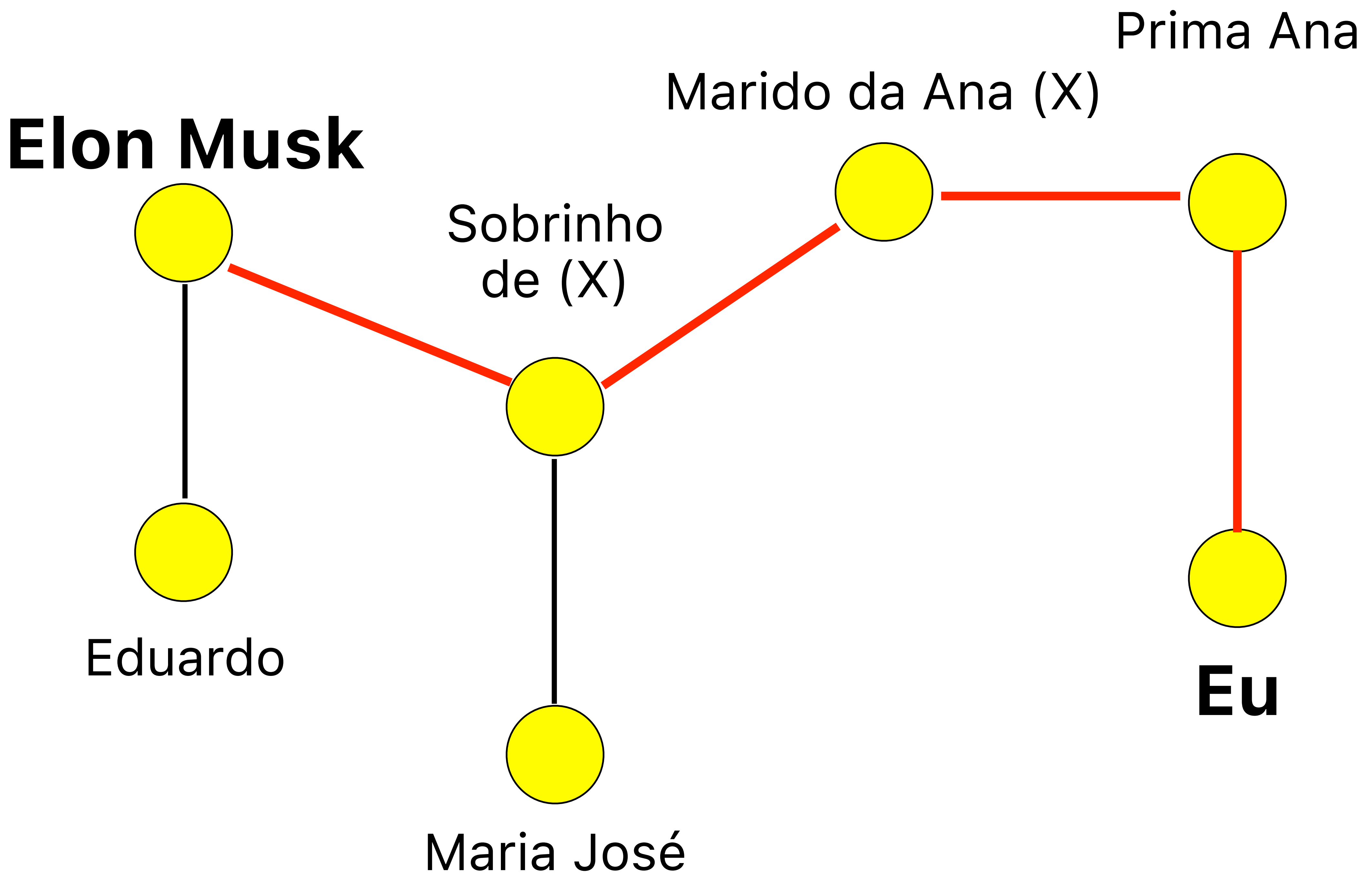
- Um caminho (v0, v1, ... , vk) forma um ciclo se v0 = vk e contém ao menos um outro vértice além de v0 e vk
- O ciclo é simples se os vértices v1, v2, ..., vk são distintos
- Ex.: O caminho (0, 1, 2, 0) é um ciclo. O caminho (4, 5, 4) não é um ciclo



Grafo Conexo

ESD 2 Estrutura de Dados

- Um grafo é conexo quando existe um caminho “entre cada par” de vértices. Caso contrário, G é desconexo
- Grafo Totalmente Desconexo:
  - Quando não possui arestas
  - Todos os vértices isolados
- Em um grafo conexo, é possível chegar em qualquer vértice partindo de qualquer outro vértice



Qual a diferença entre...

Grafo

Árvore

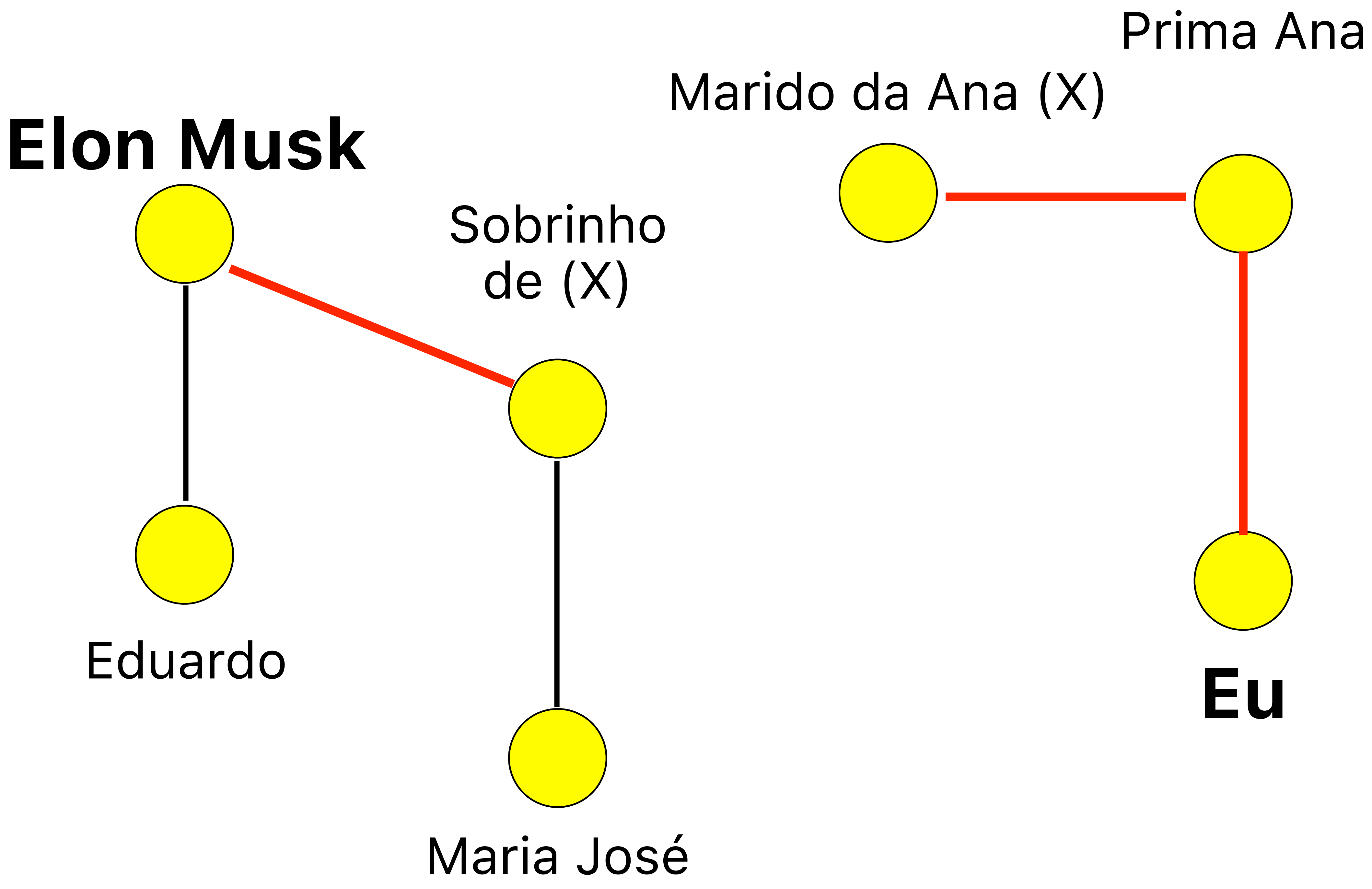
Uma árvore é um tipo especial de grafo que é **conexo** e **acíclico**.

Isso significa que há um caminho único entre qualquer par de vértices na árvore e não existem ciclos, ou seja, não há caminhos que começam e terminam no mesmo vértice

Grafo Desconexo

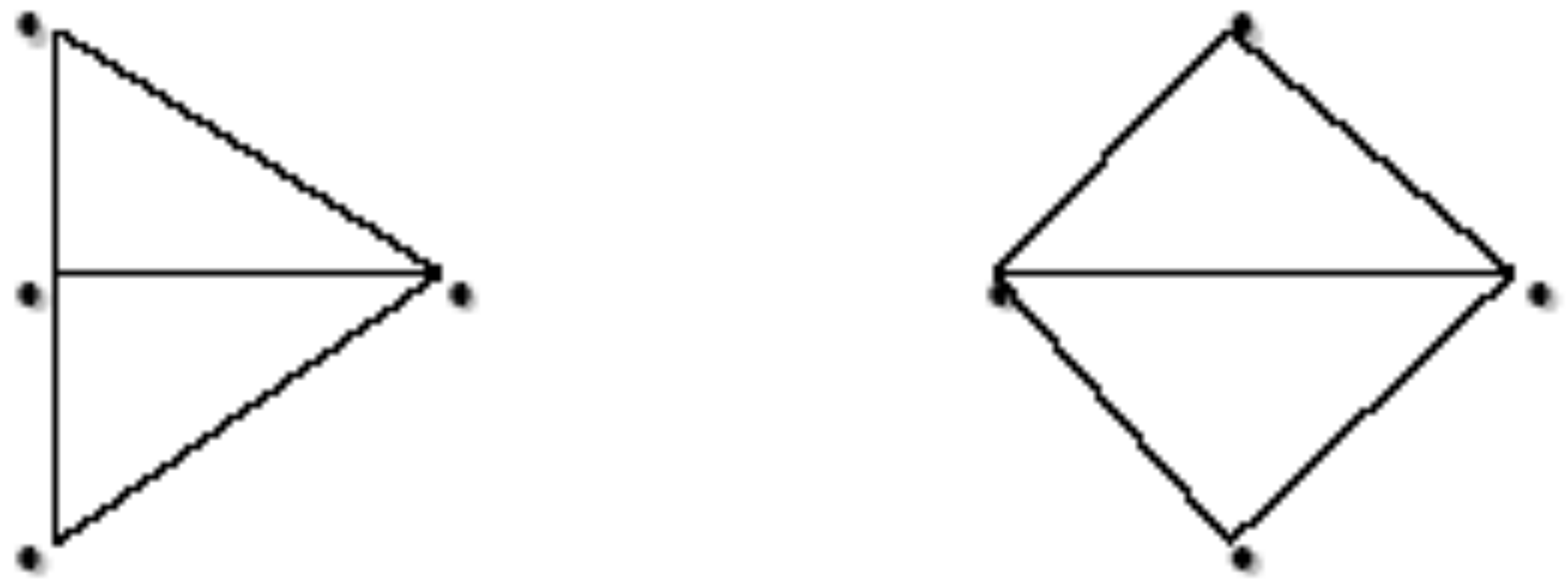
ESD 2 Estrutura de Dados

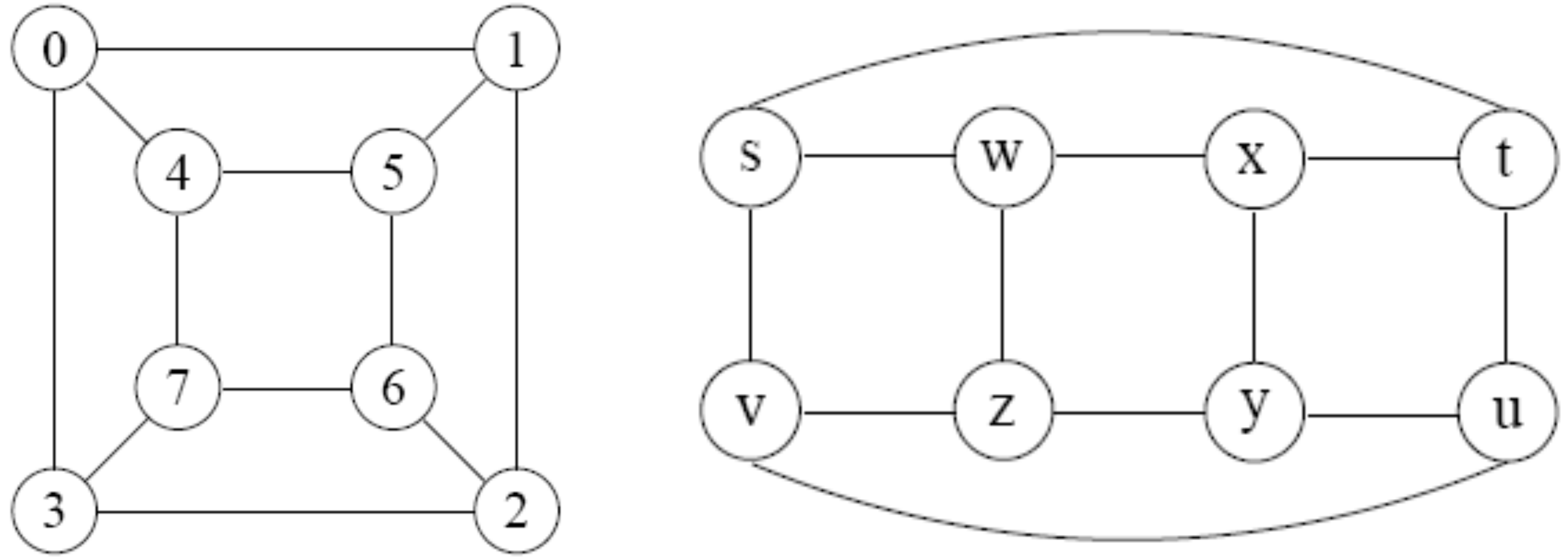
- Em um grafo desconexo, não é possível chegar em qualquer vértice partindo de qualquer outro vértice



Grafo Isomorfos

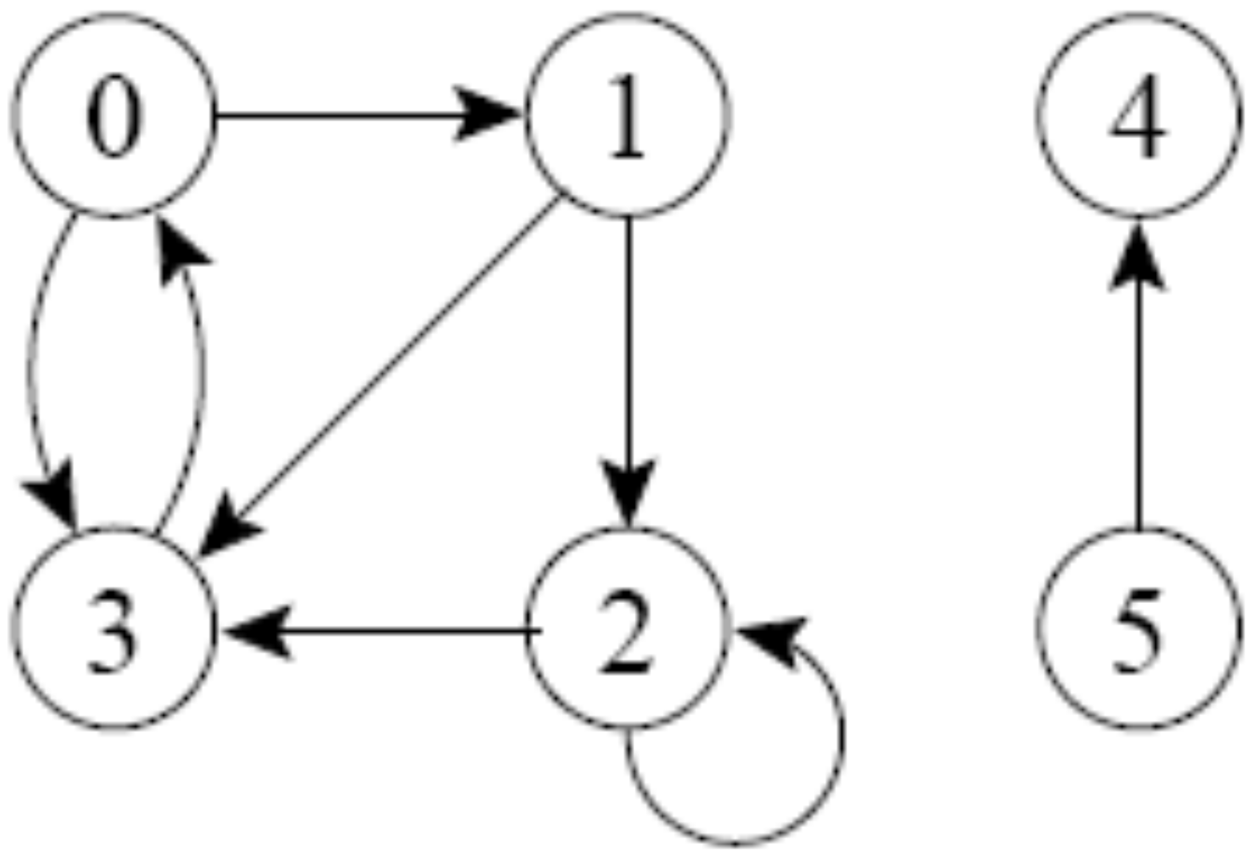
- Dois grafos são isomorfos se existe um mapeamento bionívoco entre seus conjuntos de vértices e arestas
- Em outras palavras, eles podem ter desenhos diferentes, mas têm propriedades iguais



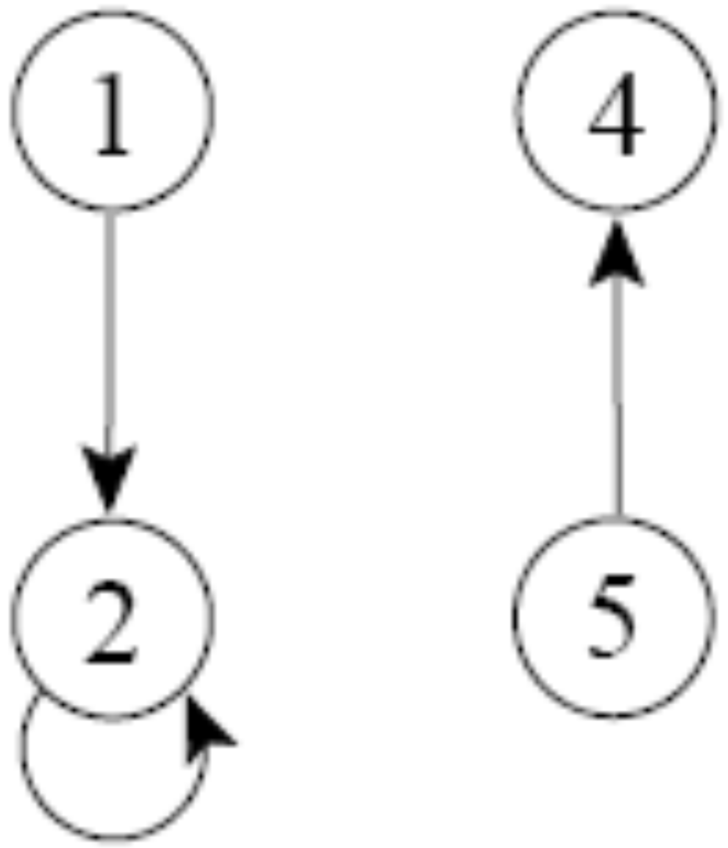


Subgrafos

- Um grafo G′ = (V′, A′) é um subgrafo de G = (V, A) se V′ C V e A′ C A
- Ex.: Subgrafo induzido pelo conjunto de vértices {1, 2, 4, 5}



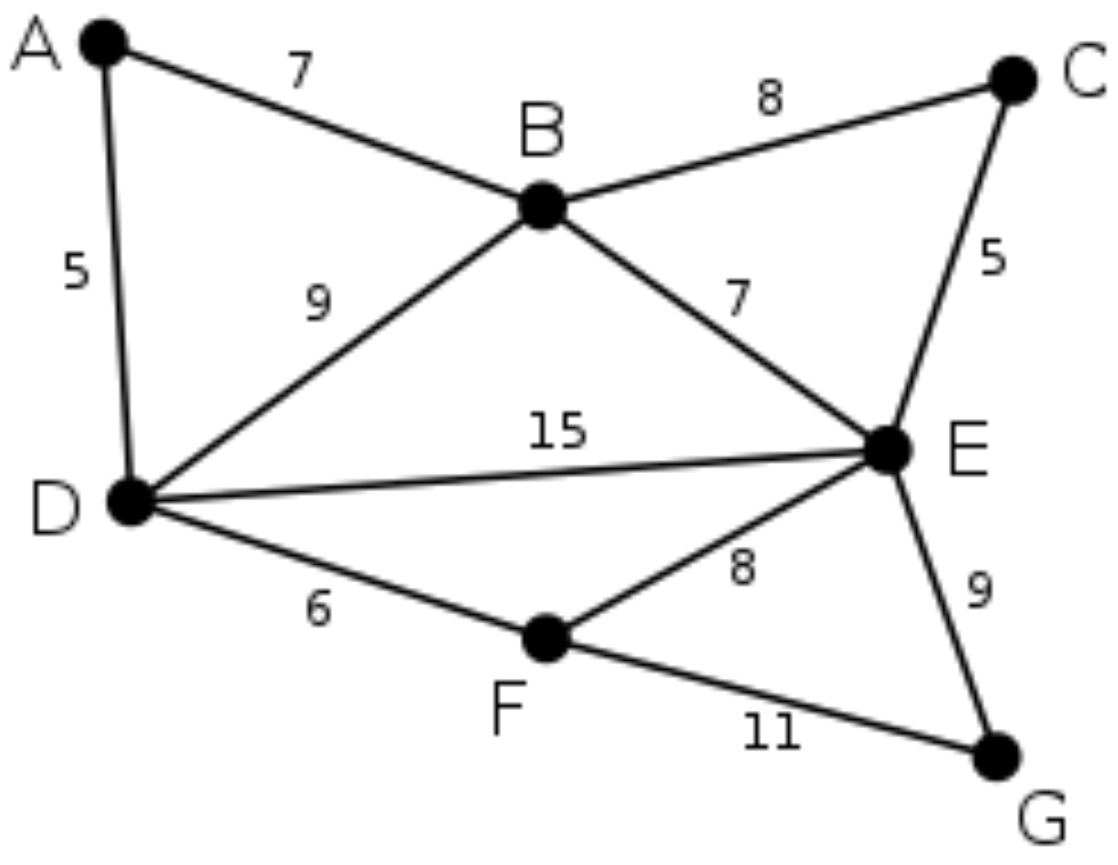
grafo G



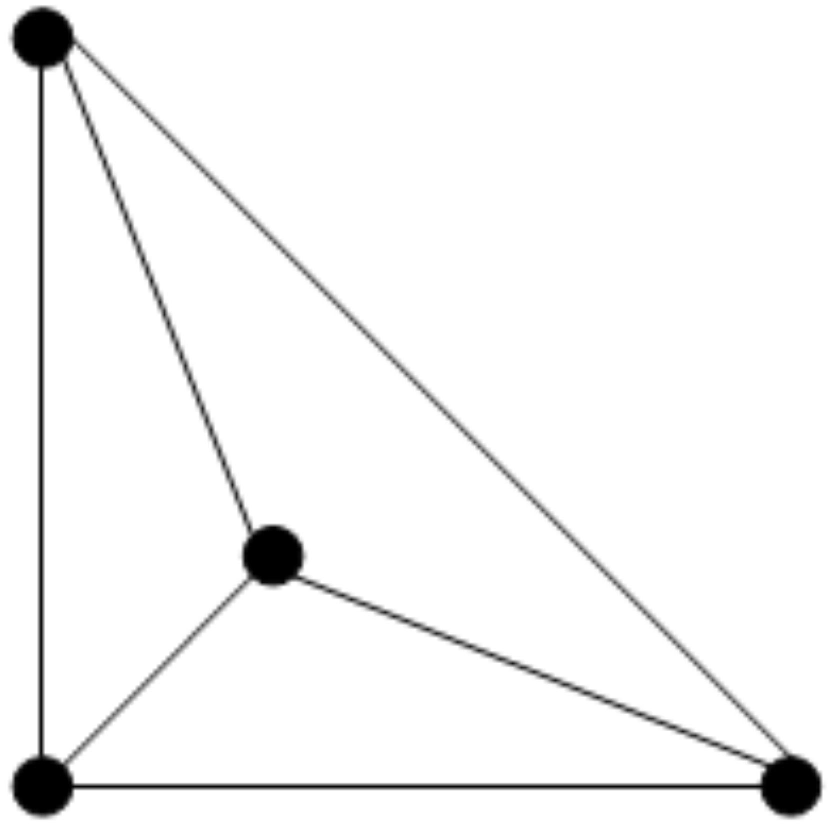
grafo G'

Outras Classificações

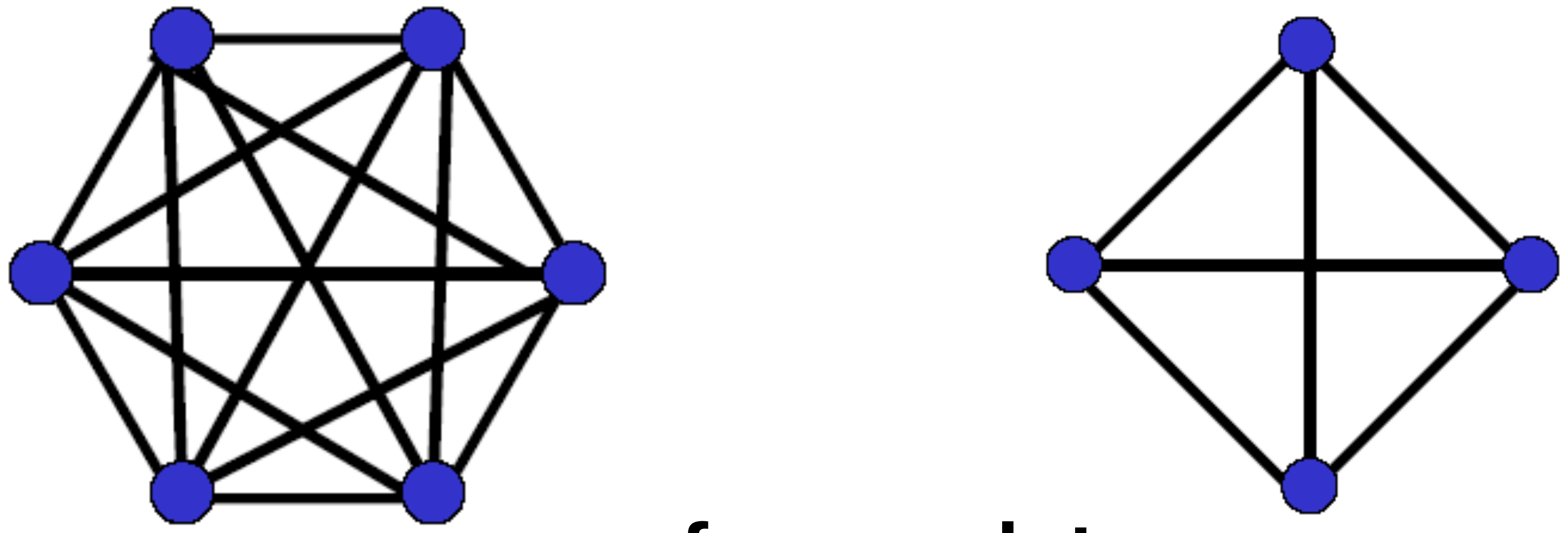
- **Grafo ponderado:** possui pesos associados às arestas
- **Grafo planar:** Um grafo é planar se puder ser desenhado sem cruzamentos
- **Grafo completo:** é um grafo não direcionado no qual todos os pares de vértices são adjacentes



grafo ponderado



grafo planar

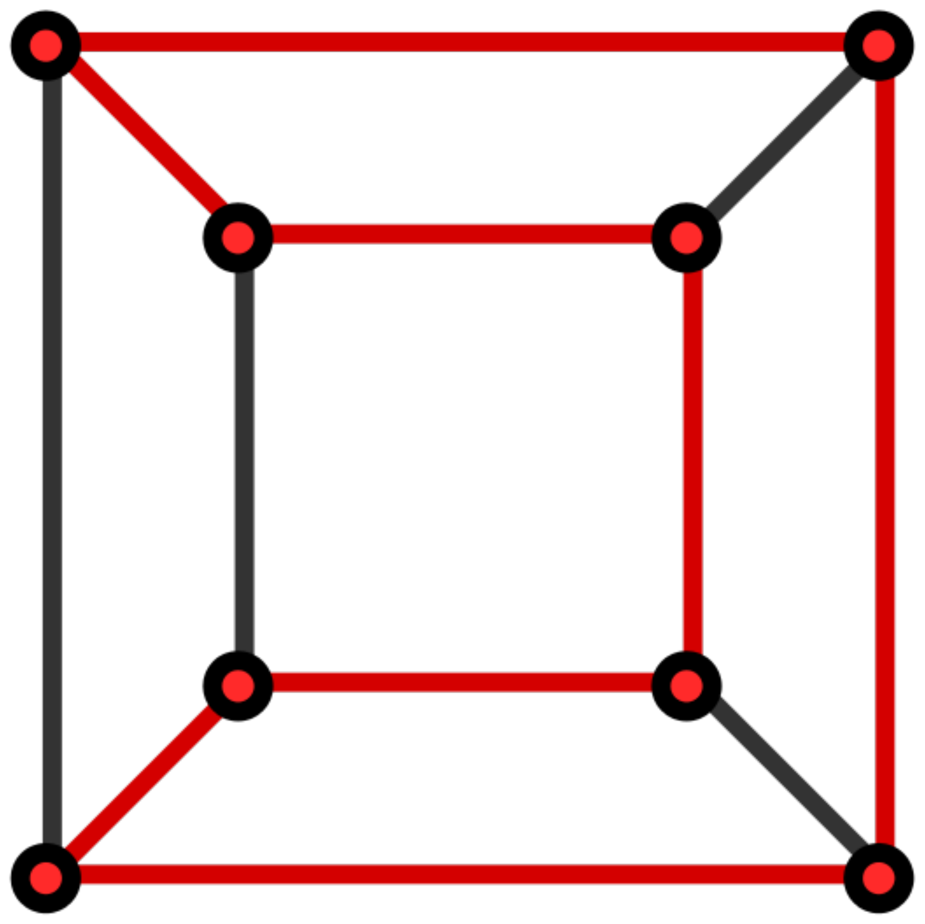


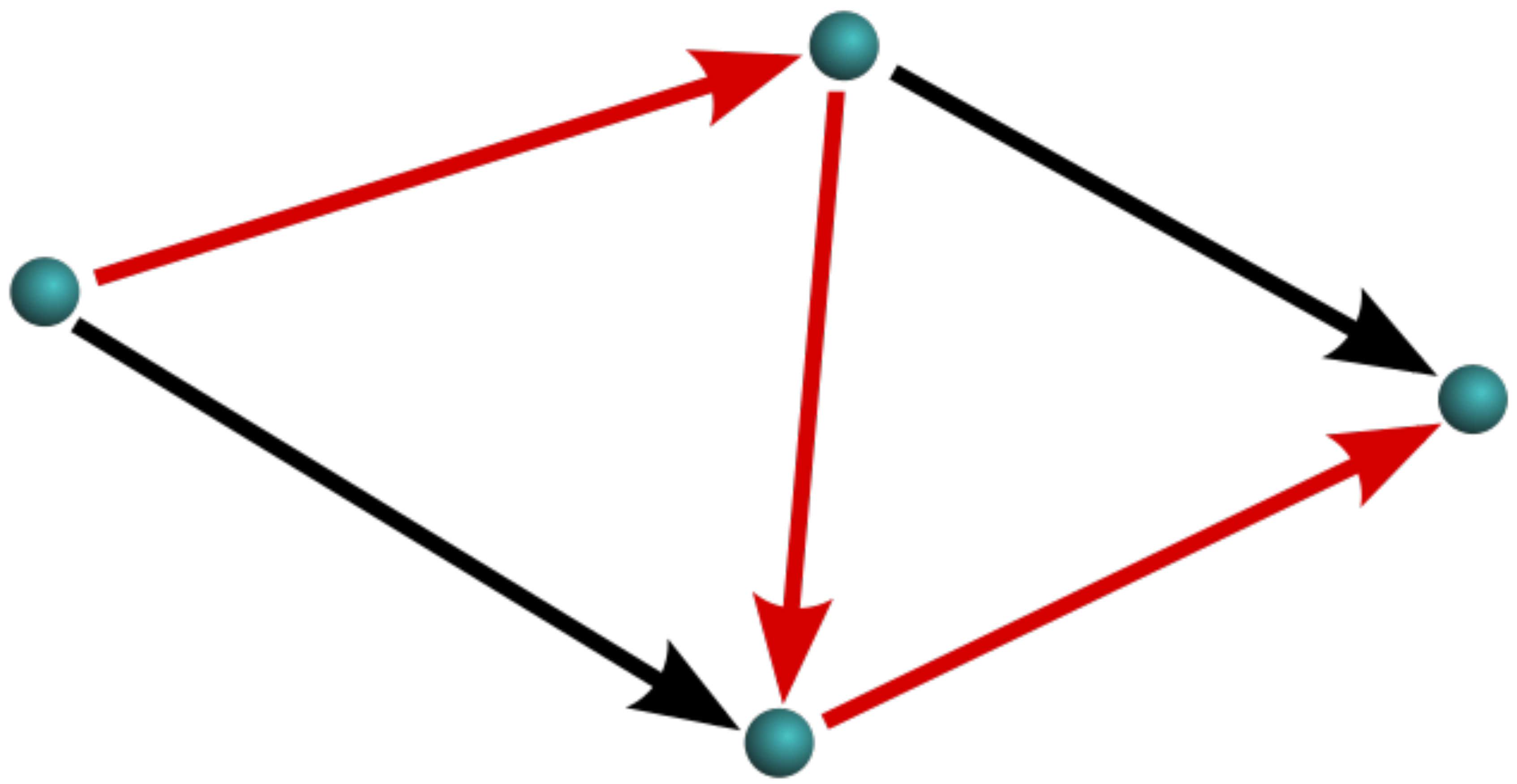
grafo completo

Ciclo Hamiltoniano

ESD 2
Estrutura de Dados

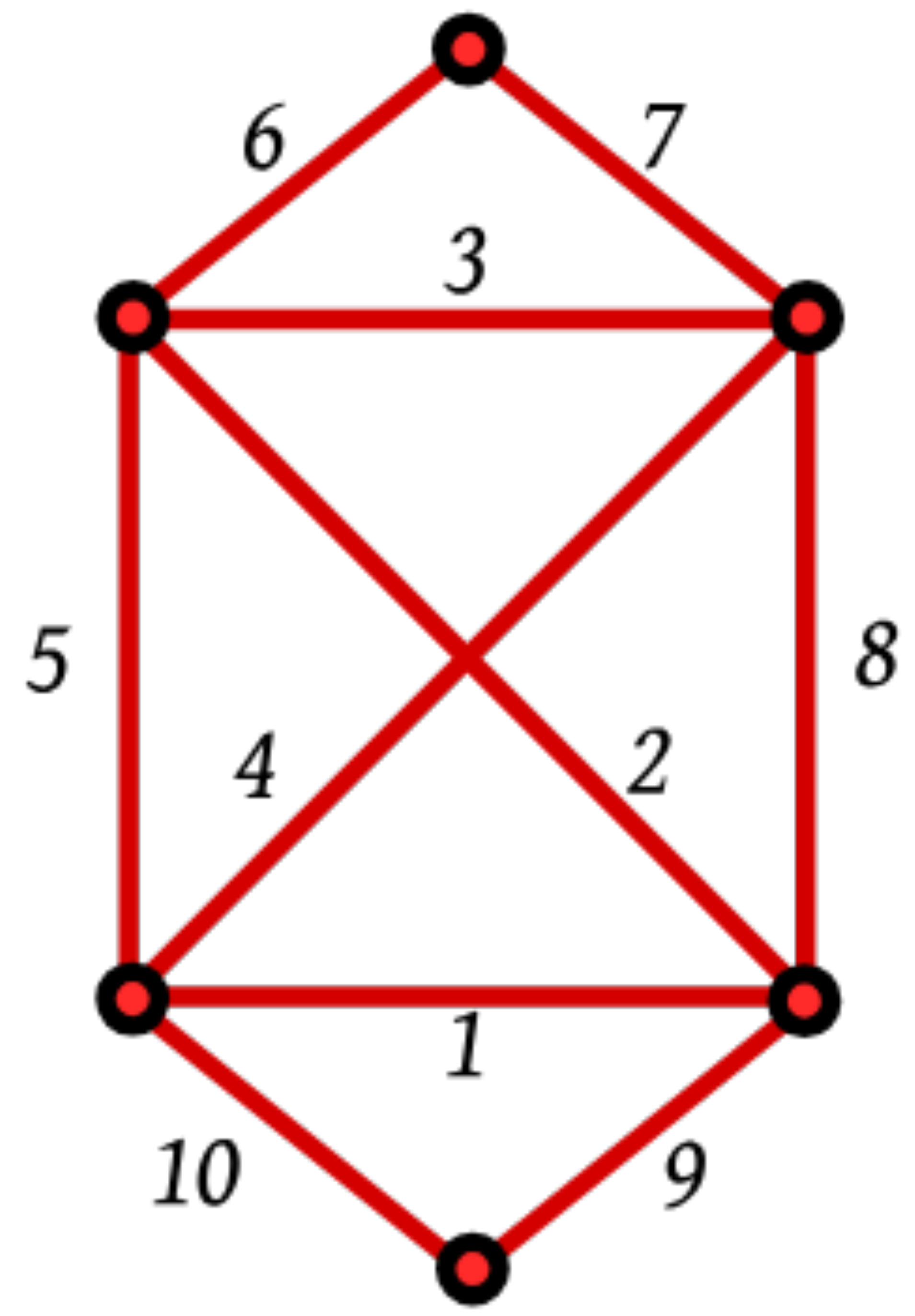
- Ciclo Hamiltoniano é um **ciclo simples** que contém **todos os vértices** do grafo
- Exemplos de aplicações: Logística, Projeto de Circuito Elétrico, Problemas de Sequenciamento, etc.
- O tempo para se achar todos os ciclos Hamiltonianos de um grafo G é exponencial (identificar todos os ciclos e depois ver se são hamiltonianos)





Ciclo Euleriano

- Ciclo Euleriano é um **ciclo simples**, que contém **todas as arestas** do grafo
- Um grafo G conexo possui ciclo Euleriano se e somente se todo vértice de G possuir grau par (≥ 2)
- Encontrar um ciclo Euleriano leva um tempo polinomial com relação ao tamanho do grafo
- Exemplos de aplicação: redes de distribuição de energia, rotas de coleta de lixo, sequenciamento de DNA, etc.



Representação de Grafos

ESD 2
Estrutura de Dados

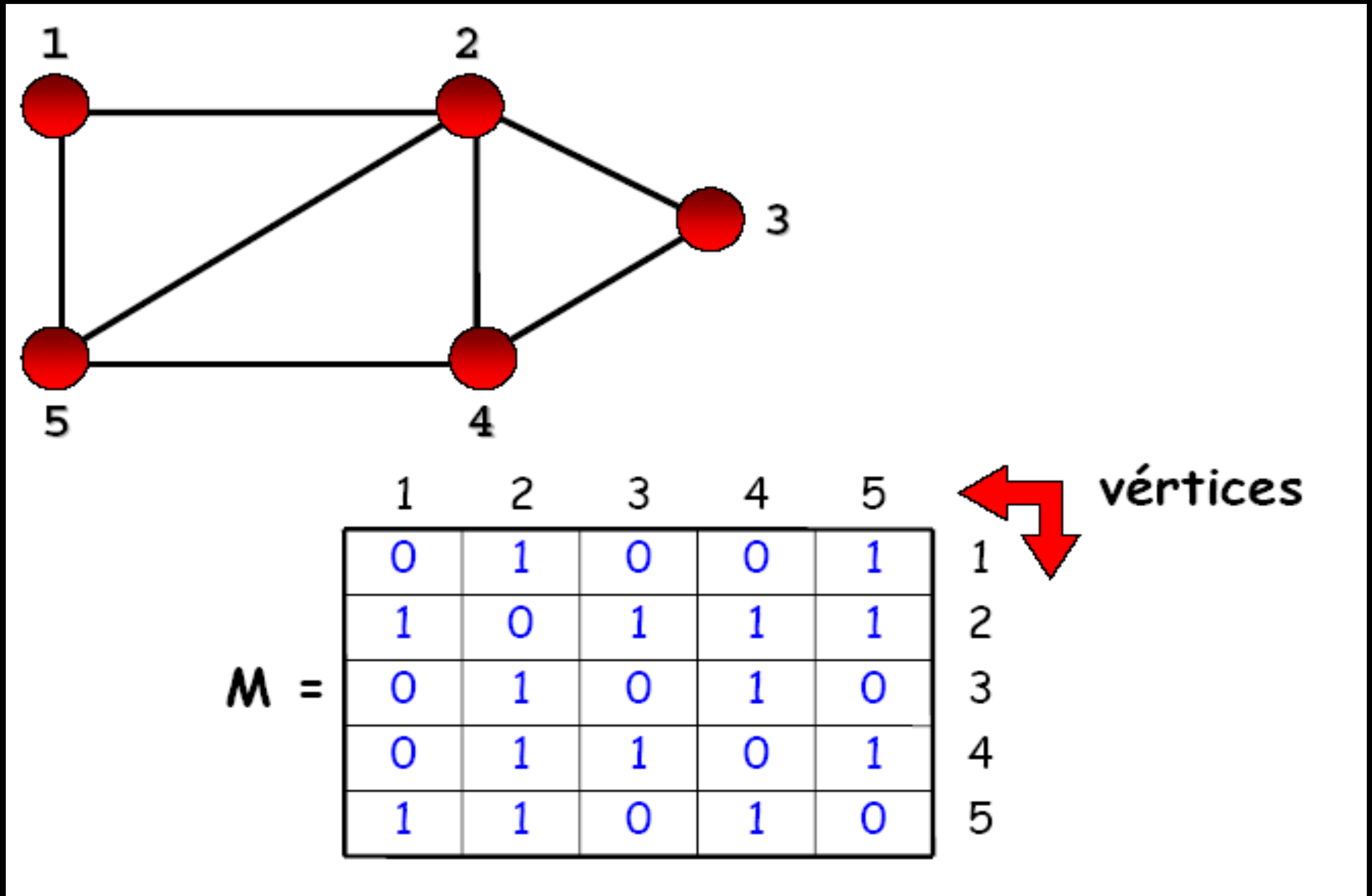
Lista Adjacência

Matriz Adjacência

A escolha da estrutura de dados certa para a representação de grafos tem impacto no desempenho do algoritmo e deve ser considerado o problema modelado para tomada desta decisão

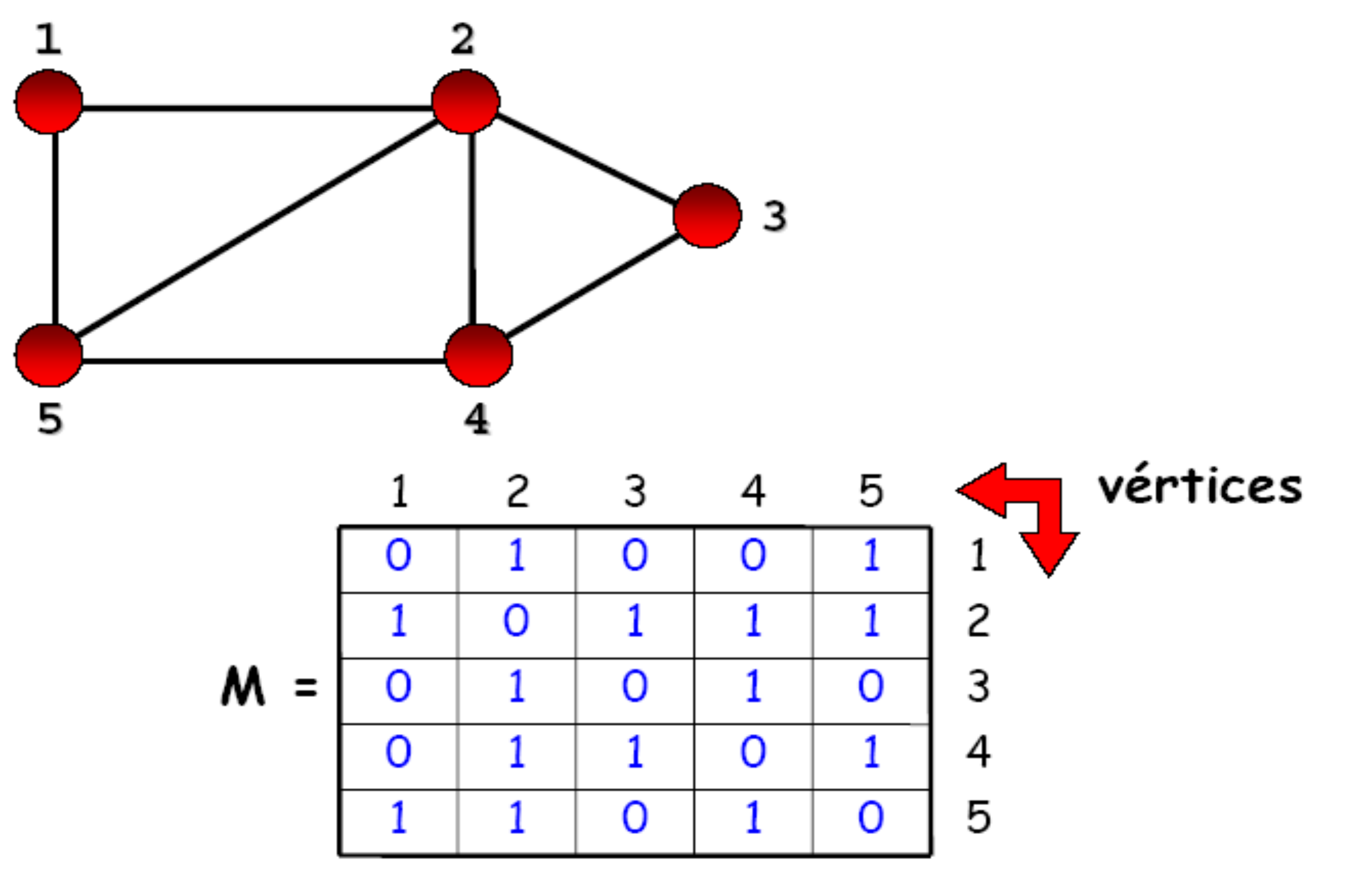
Matriz de Adjacência

- A matriz de adjacência de um grafo G = (V, A) contendo n vértices é uma matriz n x n de bits, onde A[i, j] é 1 (ou verdadeiro) se e somente se existe um arco do vértice i para o vértice j
- Para grafos ponderados A[i, j] contém o rótulo ou peso associado com a aresta e, neste caso, a matriz não é de bits
  - Se não existir uma aresta de i para j então é necessário utilizar um valor que não possa ser usado como rótulo ou peso



Matriz de Adjacência
Análise

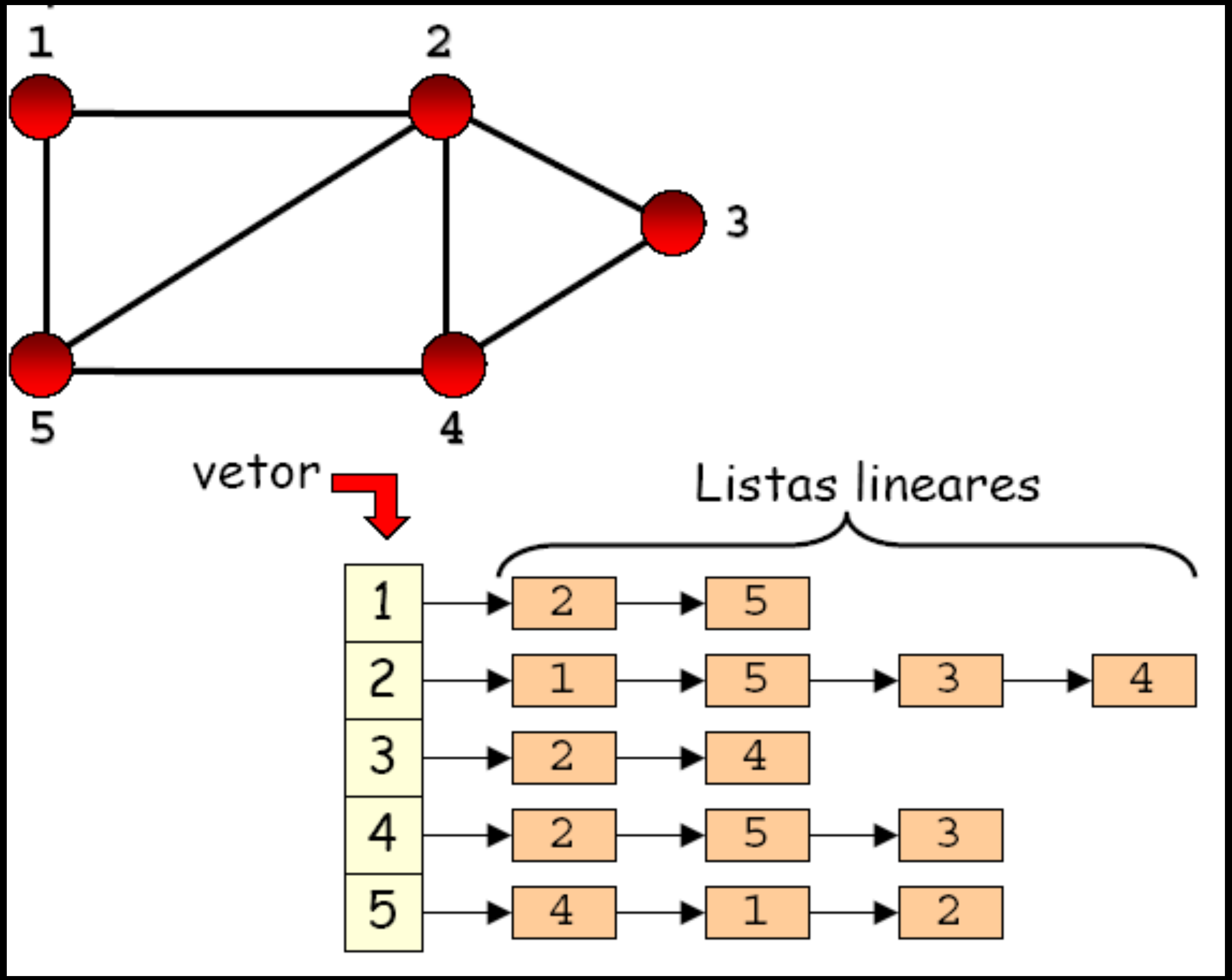
- É a forma mais simples de representação
- A maior desvantagem é que a matriz necessita $O(|V|^2)$ de espaço. Ler ou examinar a matriz tem complexidade de tempo $O(|V|^2)$
- Deve ser utilizada para grafos densos, onde $|A|$ é próximo de $|V|^2$
- É muito útil para algoritmos em que necessitamos saber com rapidez se existe uma aresta ligando dois vértices
    - Teste se aresta (i,j) está no grafo: $O(1)$



$$M = \begin{array}{c|ccccc|c} & 1 & 2 & 3 & 4 & 5 & \\ \hline & 0 & 1 & 0 & 0 & 1 & 1 \\ & 1 & 0 & 1 & 1 & 1 & 2 \\ & 0 & 1 & 0 & 1 & 0 & 3 \\ & 0 & 1 & 1 & 0 & 1 & 4 \\ & 1 & 1 & 0 & 1 & 0 & 5 \end{array}$$

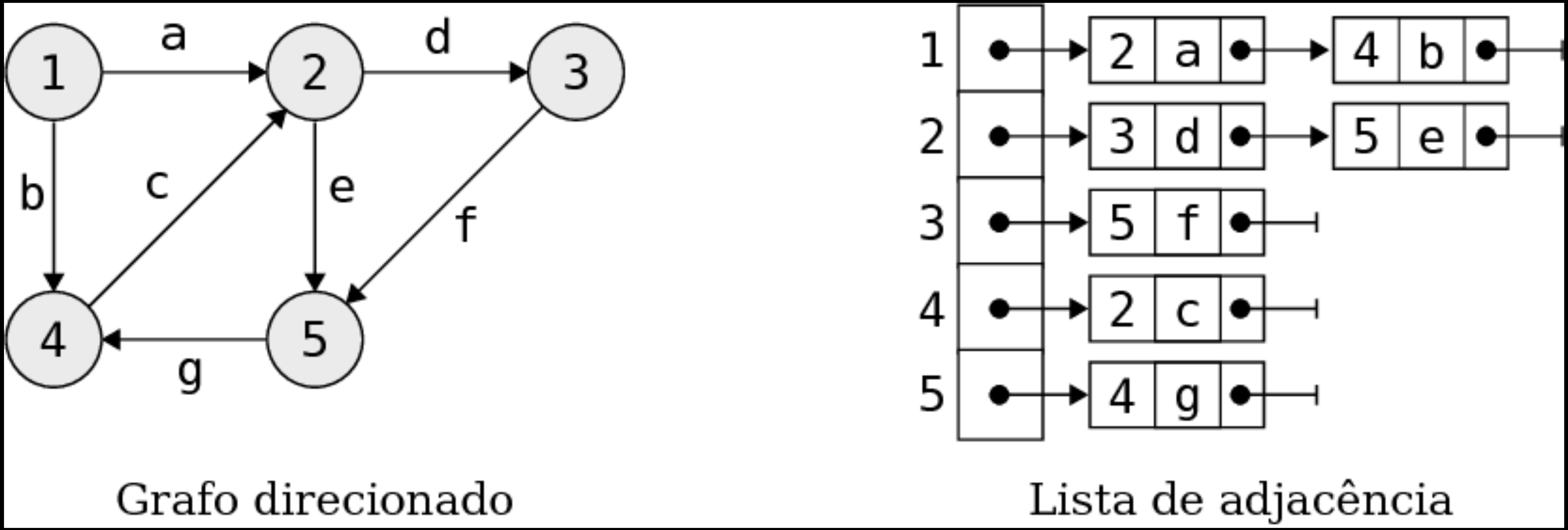
Lista de Adjacência

- Representar o grafo usando listas:
- Um vetor de V elementos (V é o número de vértices)
- O i-ésimo elemento do vetor aponta para a lista dos vértices adjacentes ao vértice i



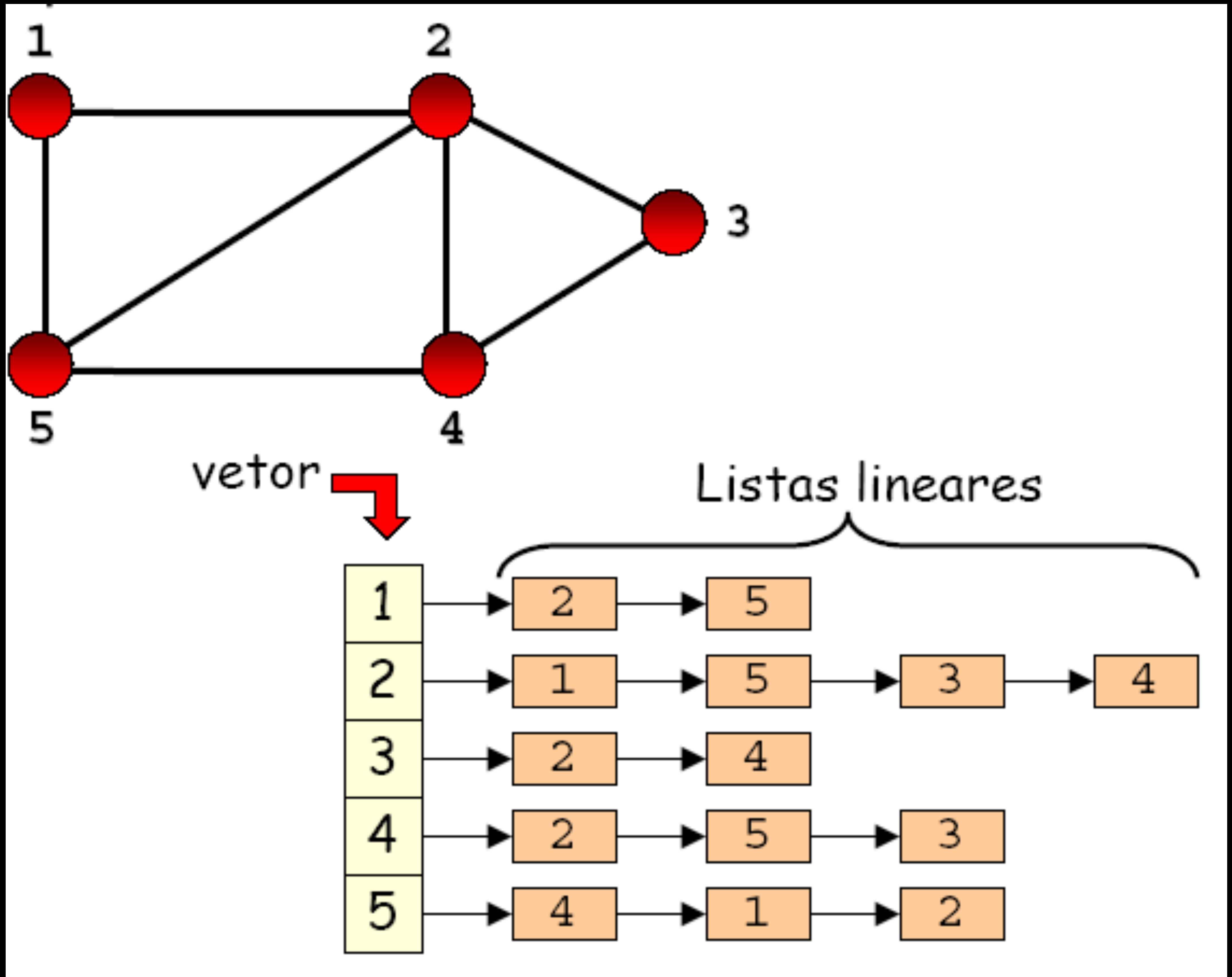
Lista de Adjacência

- Para grafos ponderados, que contém o rótulo ou peso associado às arestas, é necessário adicionar o atributo extra (peso) em cada representação dos vértices



Lista de Adjacência
Análise

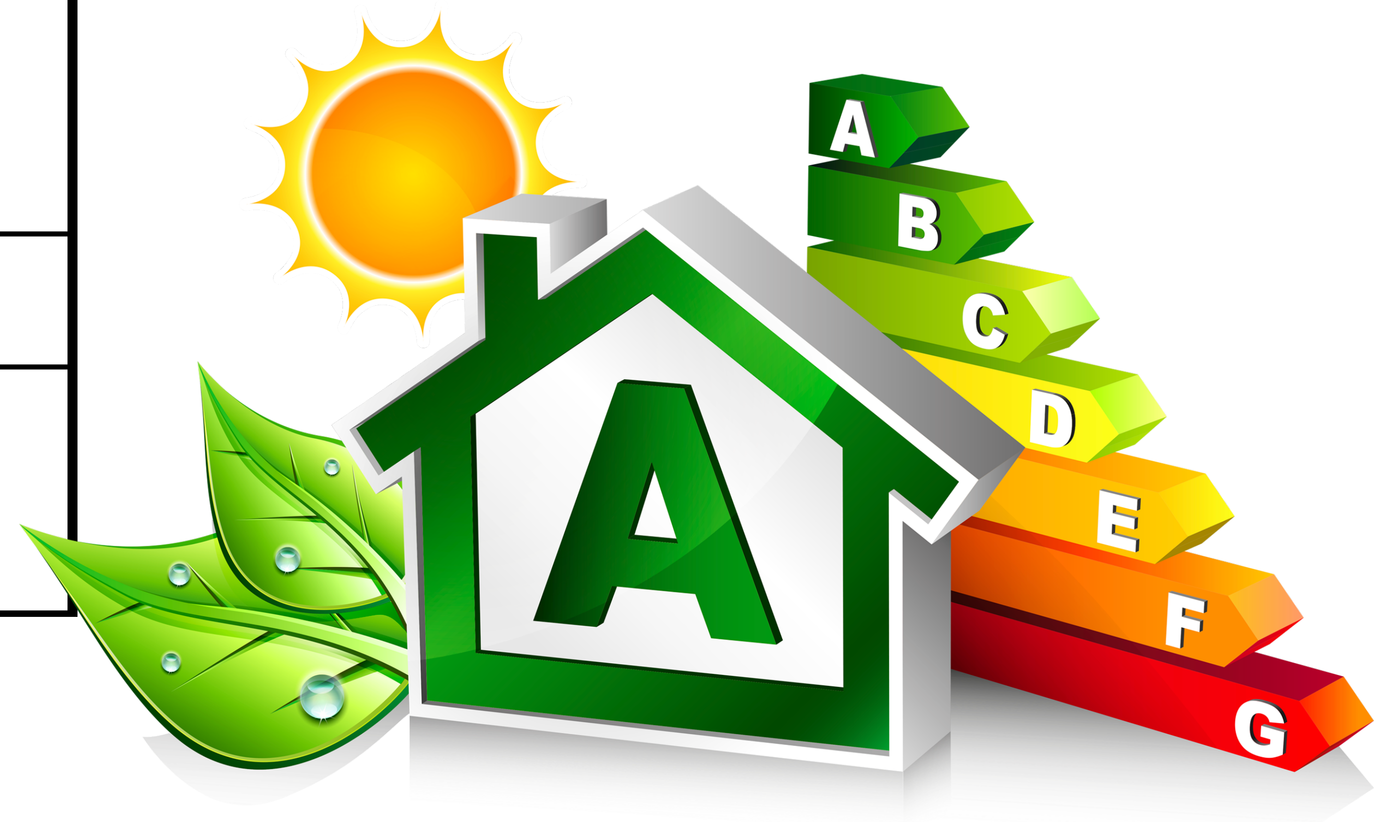
- Maior complexidade na representação
- Possui uma complexidade de espaço O(|V| + |A|). Pode ser bem menor que O(|V|²)
- Indicada para grafos esparsos, onde |A| é muito menor do que |V|²
- Teste se aresta (i,j) está no grafo: O(di), em que di é o grau do vértice i



Comparativo

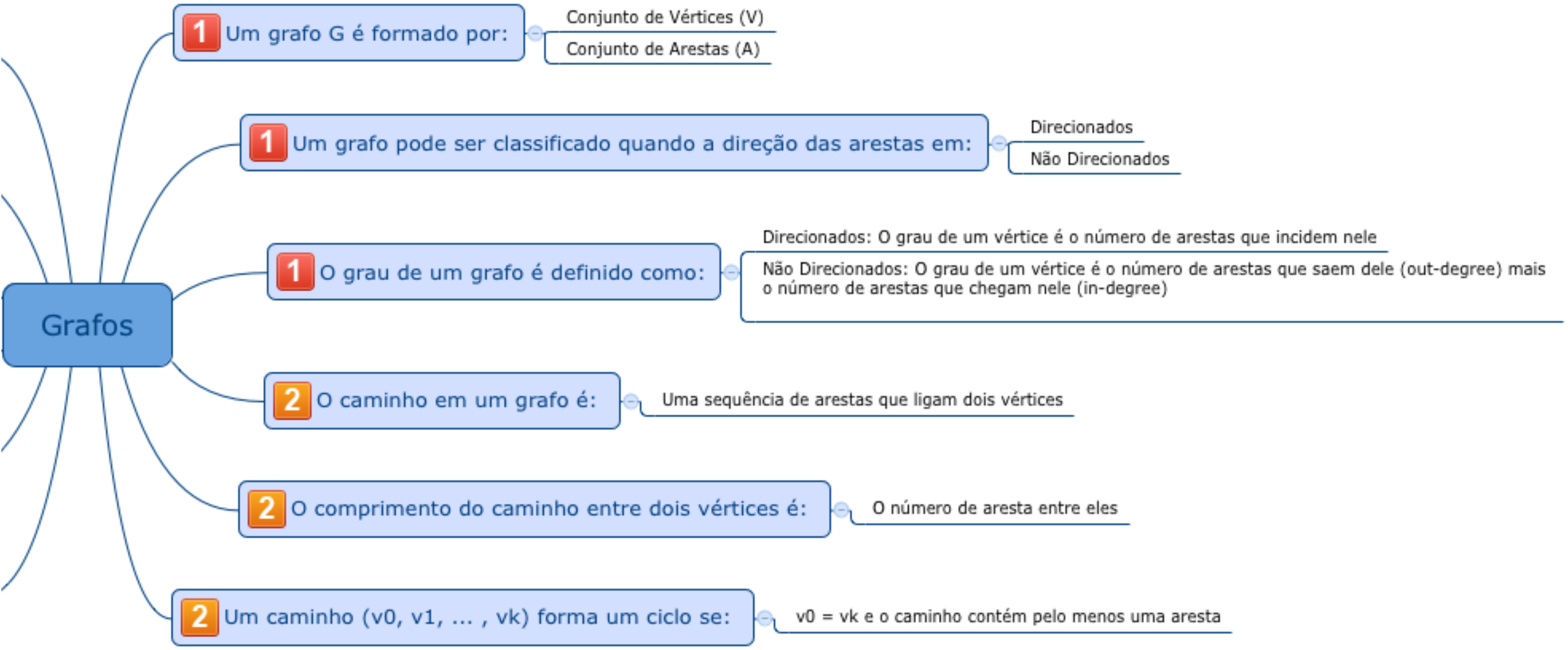
Lista x Matriz de Adjacência

Operação	Avaliação
Verificar se aresta (x,y) está no grafo	Mais Eficiente em: Matriz de adjacência
Determinar o grau de um vértice	Mais Eficiente em: Lista de adjacência
Uso de memória	Lista: (A + V) Matriz: V^2
Uso de memória em grafos grandes	Matriz de adjacência
Inserção/Remoção de arestas	Matriz: O(1) Lista: O(d)
Melhor na maioria dos problemas	Lista de adjacência
Rapidez para percorrer o grafo	Lista: O(V + A) Matriz: $O(V^2)$



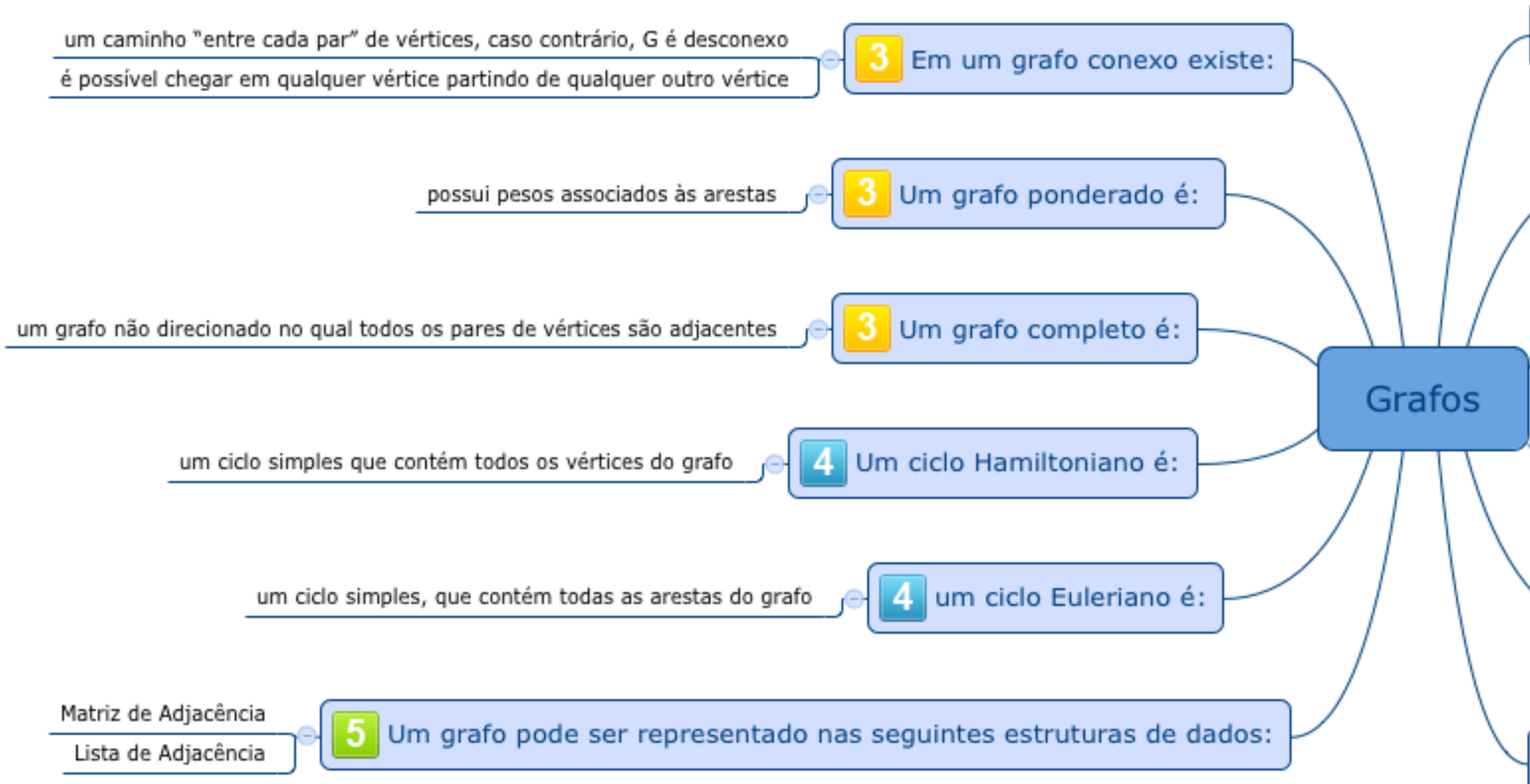
Resumo

ESD 2
Estrutura de Dados



Resumo

ESD2 Estrutura de Dados



ESD2

Estrutura de Dados

Prof. Fernando Sambinelli

sambinelli@ifsp.edu.br

INSTITUTO FEDERAL
São Paulo