Árvores

Prof. Andrei Braga

UNIVERSIDADE FEDERAL DA FRONTEIRA SUL

Conteúdo

Número de componentes conexas e de arestas

- Para um grafo *G* com 5 vértices, vamos estudar a relação entre o número de componentes conexas de *G* e o número de arestas de *G*
- Se *G* tem 5 componentes conexas, quantas arestas *G* tem?

Número de componentes conexas e de arestas

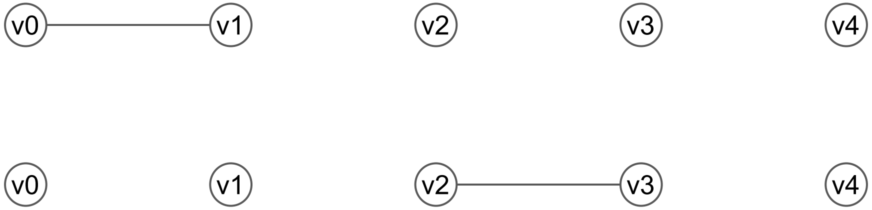
- Para um grafo *G* com 5 vértices, vamos estudar a relação entre o número de componentes conexas de *G* e o número de arestas de *G*
- Se *G* tem 5 componentes conexas, quantas arestas *G* tem? 0 arestas

Número de componentes conexas e de arestas

- Para um grafo *G* com 5 vértices, vamos estudar a relação entre o número de componentes conexas de *G* e o número de arestas de *G*
- Se *G* tem 4 componentes conexas, quantas arestas *G* tem?

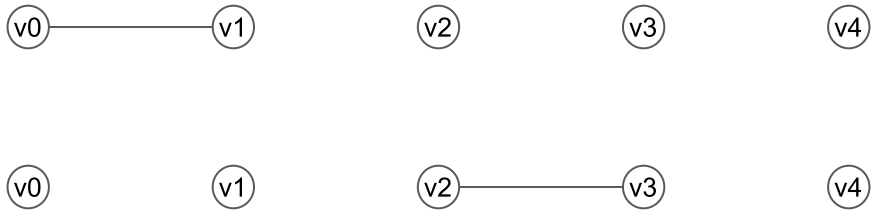
Número de componentes conexas e de arestas

- Para um grafo *G* com 5 vértices, vamos estudar a relação entre o número de componentes conexas de *G* e o número de arestas de *G*
- Se *G* tem 4 componentes conexas, quantas arestas *G* tem?



Número de componentes conexas e de arestas

- Para um grafo *G* com 5 vértices, vamos estudar a relação entre o número de componentes conexas de *G* e o número de arestas de *G*
- Se *G* tem 4 componentes conexas, quantas arestas *G* tem? 1 aresta

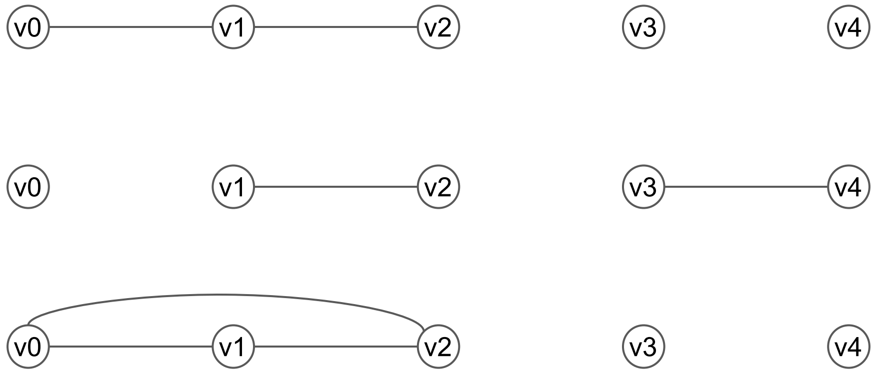


Número de componentes conexas e de arestas

- Para um grafo *G* com 5 vértices, vamos estudar a relação entre o número de componentes conexas de *G* e o número de arestas de *G*
- Se *G* tem 3 componentes conexas, quantas arestas *G* tem?

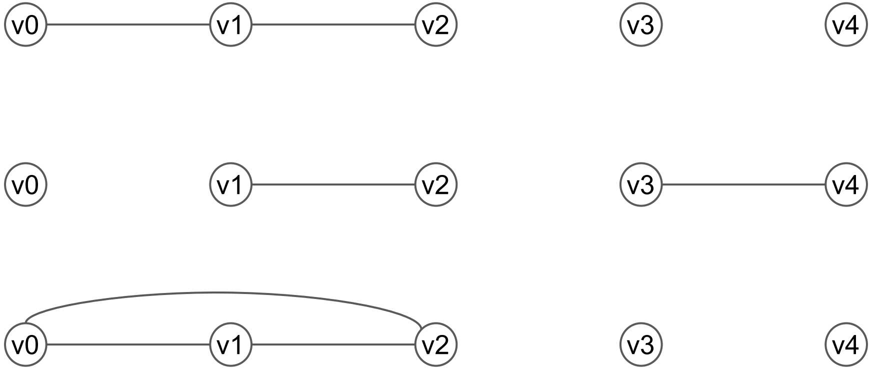
Número de componentes conexas e de arestas

- Para um grafo *G* com 5 vértices, vamos estudar a relação entre o número de componentes conexas de *G* e o número de arestas de *G*
- Se *G* tem 3 componentes conexas, quantas arestas *G* tem?



Número de componentes conexas e de arestas

- Para um grafo *G* com 5 vértices, vamos estudar a relação entre o número de componentes conexas de *G* e o número de arestas de *G*
- Se *G* tem 3 componentes conexas, quantas arestas *G* tem? 2 ou 3 arestas

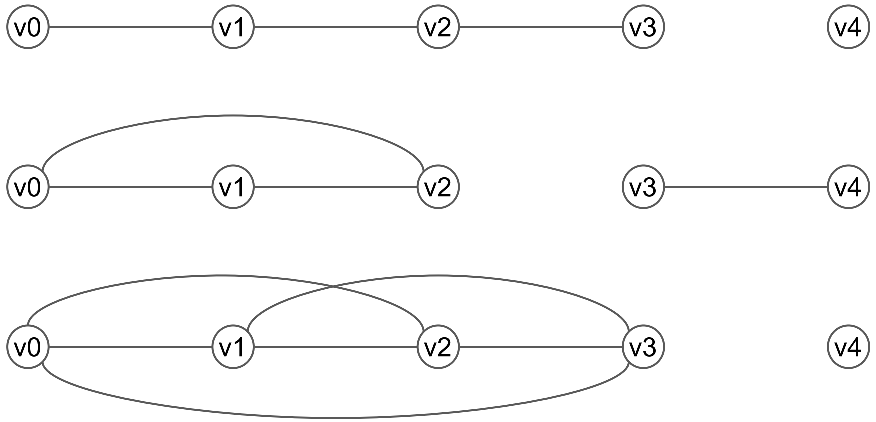


Número de componentes conexas e de arestas

- Para um grafo *G* com 5 vértices, vamos estudar a relação entre o número de componentes conexas de *G* e o número de arestas de *G*
- Se *G* tem 2 componentes conexas, quantas arestas *G* tem?

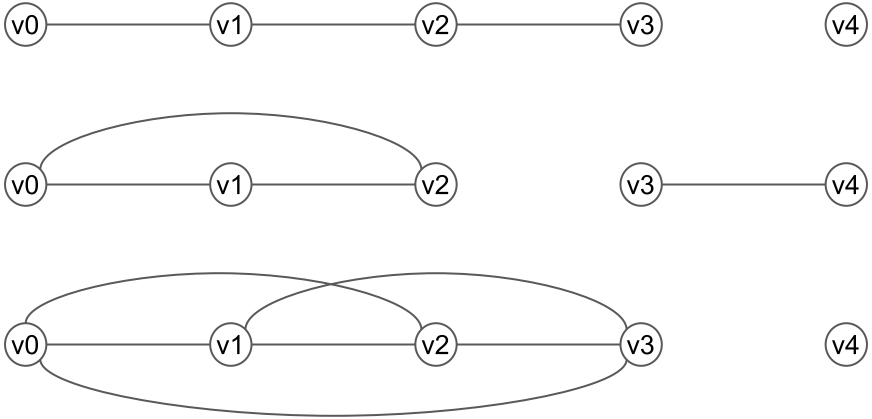
Número de componentes conexas e de arestas

- Para um grafo *G* com 5 vértices, vamos estudar a relação entre o número de componentes conexas de *G* e o número de arestas de *G*
- Se *G* tem 2 componentes conexas, quantas arestas *G* tem?



Número de componentes conexas e de arestas

- Para um grafo *G* com 5 vértices, vamos estudar a relação entre o número de componentes conexas de *G* e o número de arestas de *G*
- Se *G* tem 2 componentes conexas, quantas arestas *G* tem? 3, 4, 5 ou 6 arestas

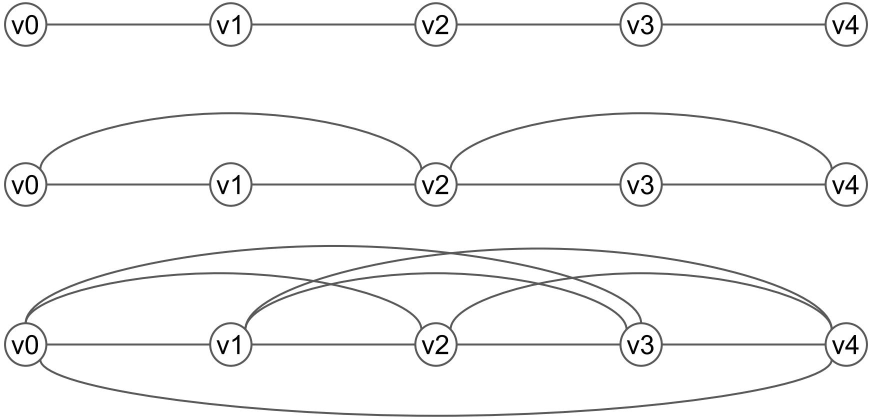


Número de componentes conexas e de arestas

- Para um grafo *G* com 5 vértices, vamos estudar a relação entre o número de componentes conexas de *G* e o número de arestas de *G*
- Se *G* tem 1 componente conexa, quantas arestas *G* tem?

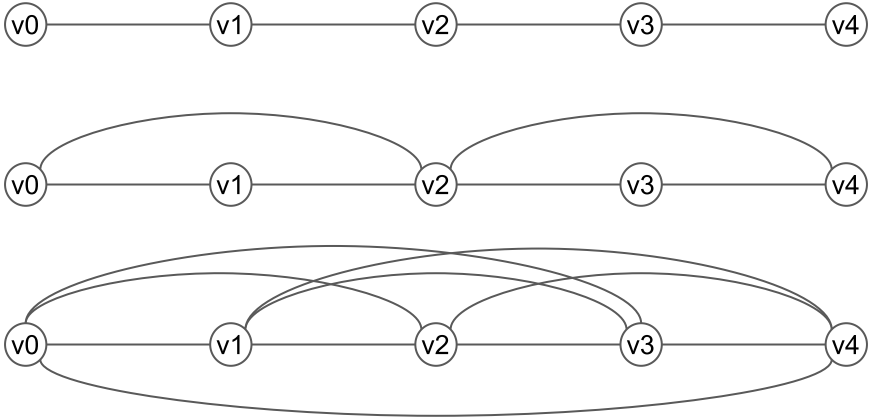
Número de componentes conexas e de arestas

- Para um grafo *G* com 5 vértices, vamos estudar a relação entre o número de componentes conexas de *G* e o número de arestas de *G*
- Se *G* tem 1 componente conexa, quantas arestas *G* tem?



Número de componentes conexas e de arestas

- Para um grafo *G* com 5 vértices, vamos estudar a relação entre o número de componentes conexas de *G* e o número de arestas de *G*
- Se *G* tem 1 componente conexa, quantas arestas *G* tem? 4, 5, 6, 7, 8, 9 ou 10 arestas



Número de componentes conexas e de arestas

- De acordo com as observações anteriores, podemos perceber uma relação entre o número de componentes conexas e o número mínimo de arestas de um grafo
- Vamos escrever o que percebemos de uma forma mais geral

Número de componentes conexas e de arestas

- Vamos escrever o que percebemos de uma forma mais geral
- Dado um grafo *G* com 5 vértices,
  - Se *G* tem 5, ou seja, 5 - 0 componentes conexas, então *G* tem pelo menos 0 arestas
  - Se *G* tem 4, ou seja, 5 - 1 componentes conexas, então *G* tem pelo menos 1 aresta
  - Se *G* tem 3, ou seja, 5 - 2 componentes conexas, então *G* tem pelo menos 2 arestas
  - Se *G* tem 2, ou seja, 5 - 3 componentes conexas, então *G* tem pelo menos 3 arestas
  - Se *G* tem 1, ou seja, 5 - 4 componentes conexas, então *G* tem pelo menos 4 arestas

Número de componentes conexas e de arestas

- **Teorema:** Dado um grafo G com n vértices, se G tem n - k componentes conexas, então G tem pelo menos k arestas

- Um grafo G com n vértices é conexo se G tem 1, ou seja, n - (n - 1) componente conexa
- **Teorema:** Dado um grafo G com n vértices, se G é conexo, então G tem pelo menos n - 1 arestas

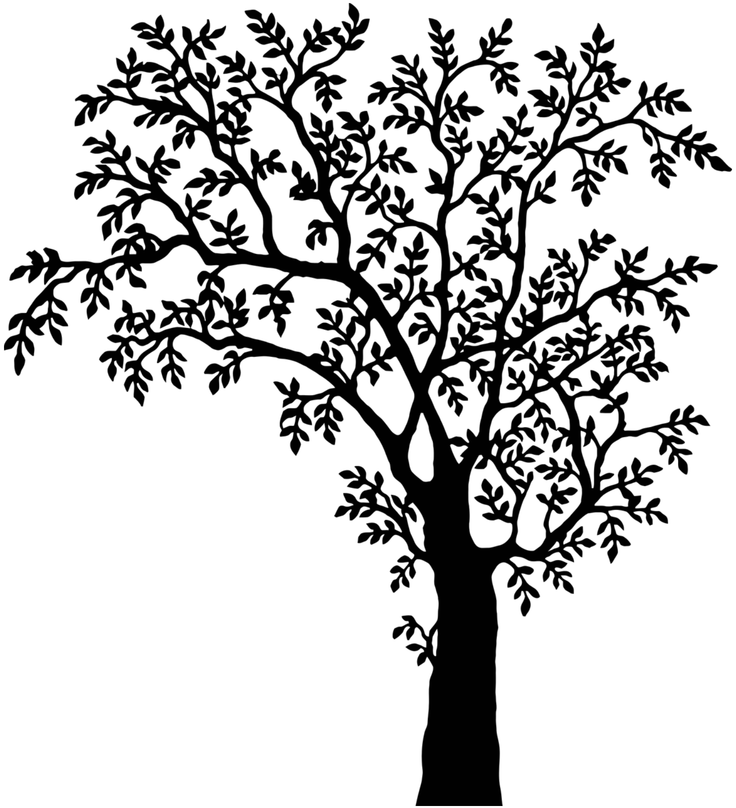
Árvores

- Vimos que, para ser conexo, um grafo precisa ter um determinado número mínimo de arestas
- **Árvores** são grafos conexos onde este **mínimo é atingido**

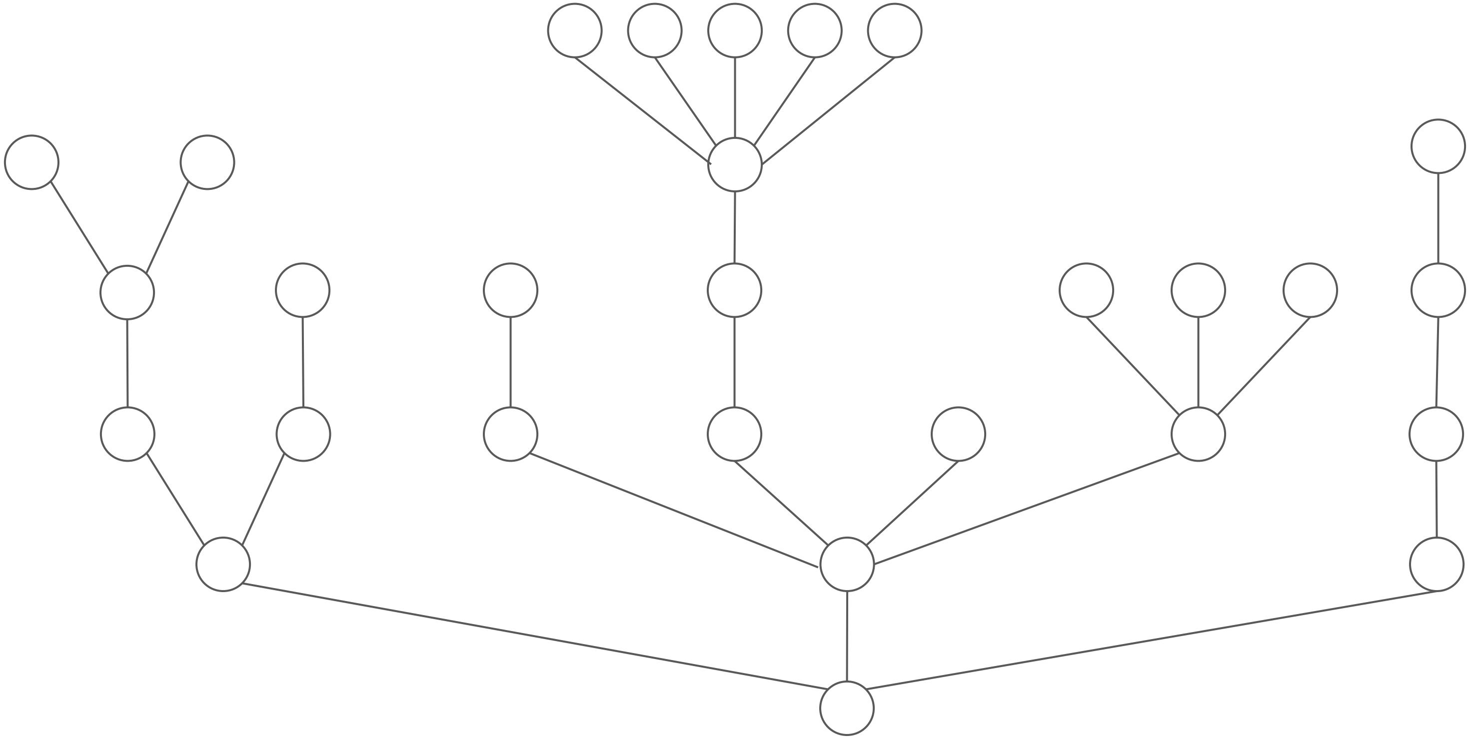
- Árvores são um tipo de grafo que possui muitas aplicações importantes
- Estas aplicações ocorrem em áreas como armazenamento e busca eficiente de dados e telecomunicações

Árvore

Abstraindo a estrutura de uma árvore

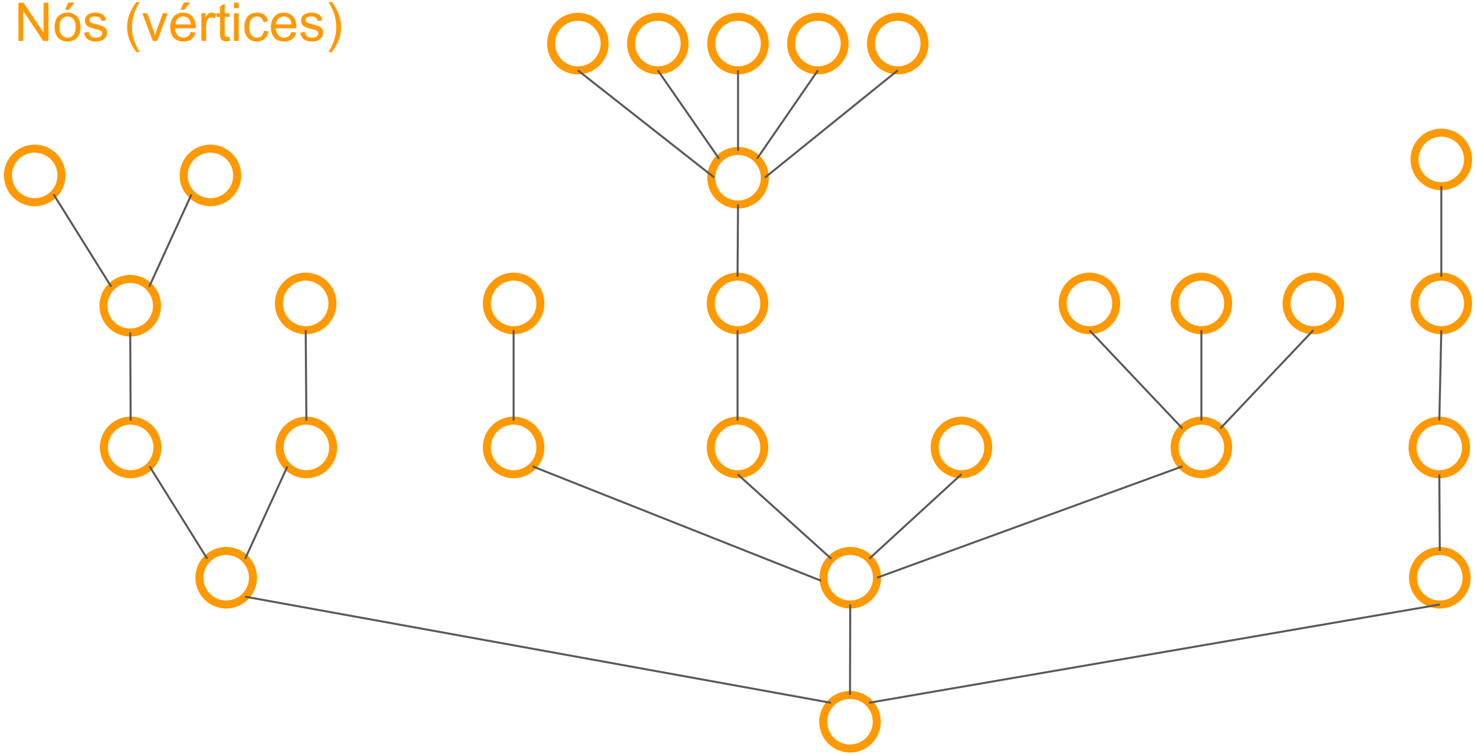


Abstraindo a estrutura de uma árvore

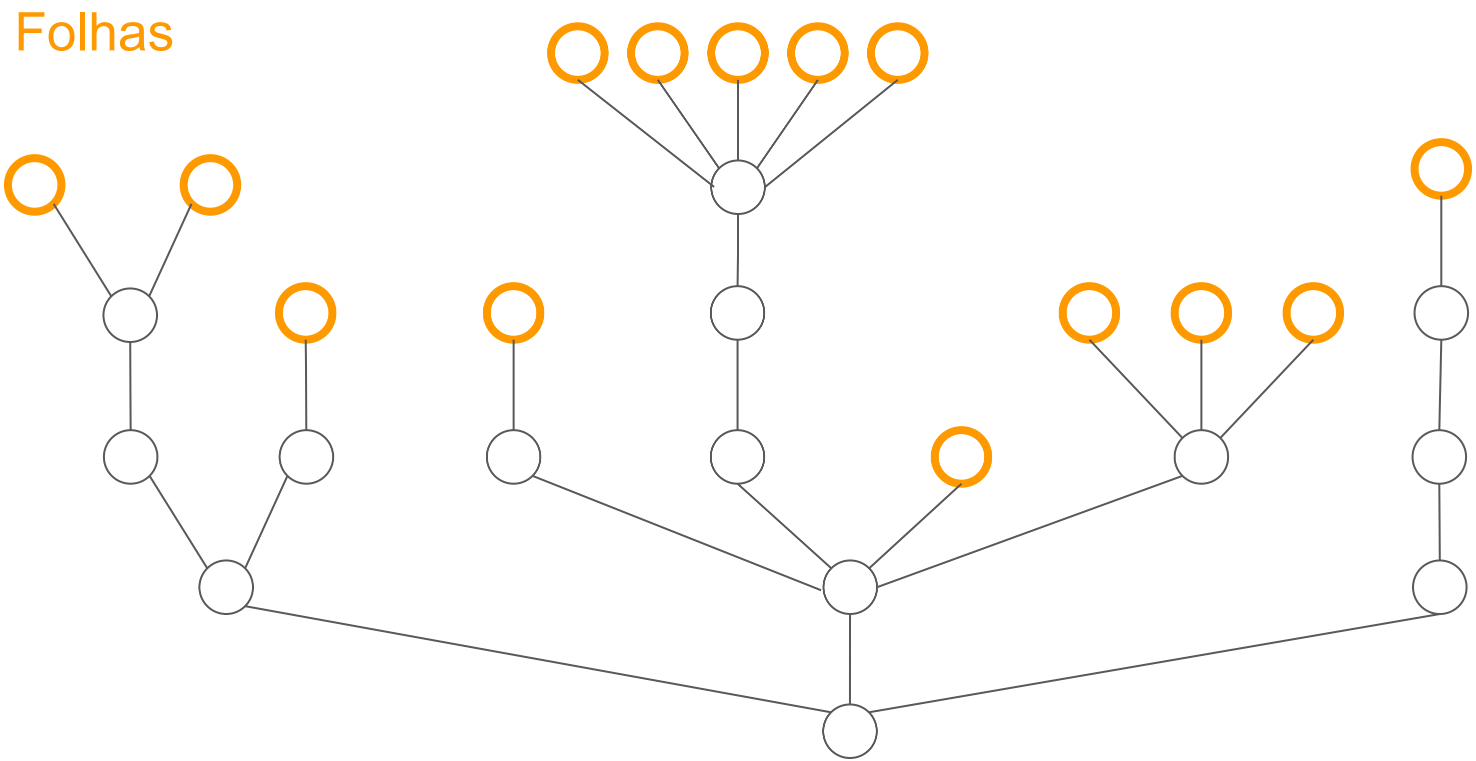


Abstraindo a estrutura de uma árvore

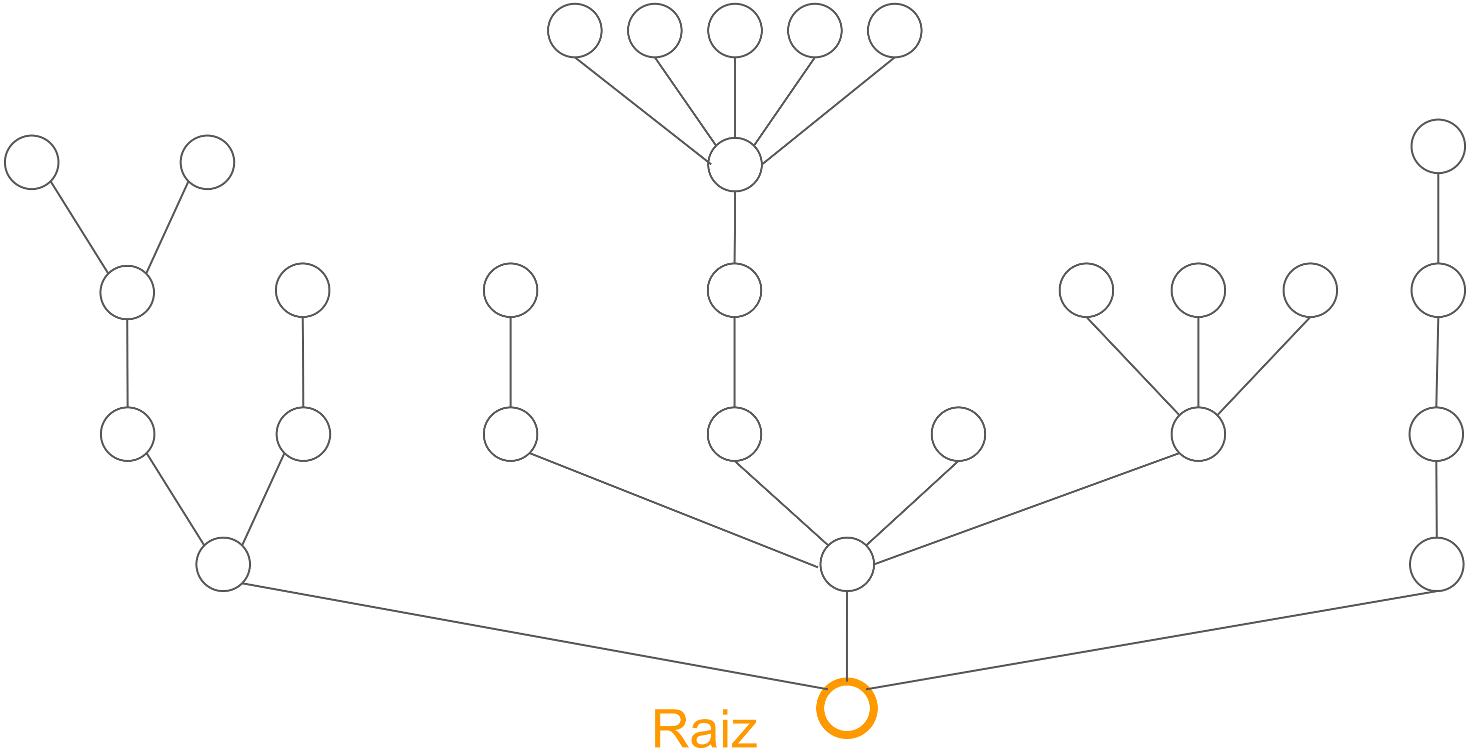
Nós (vértices)



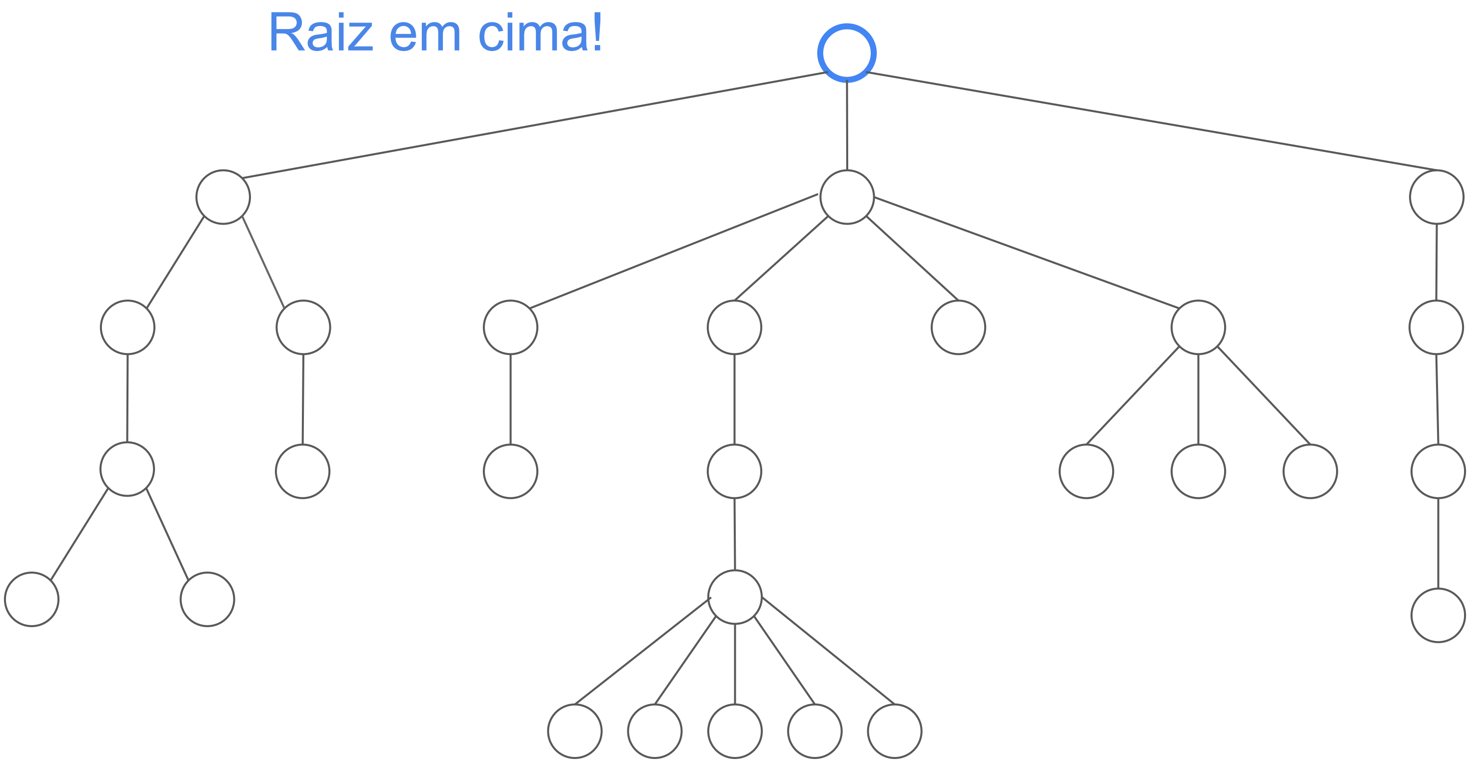
Abstraindo a estrutura de uma árvore



Abstraindo a estrutura de uma árvore

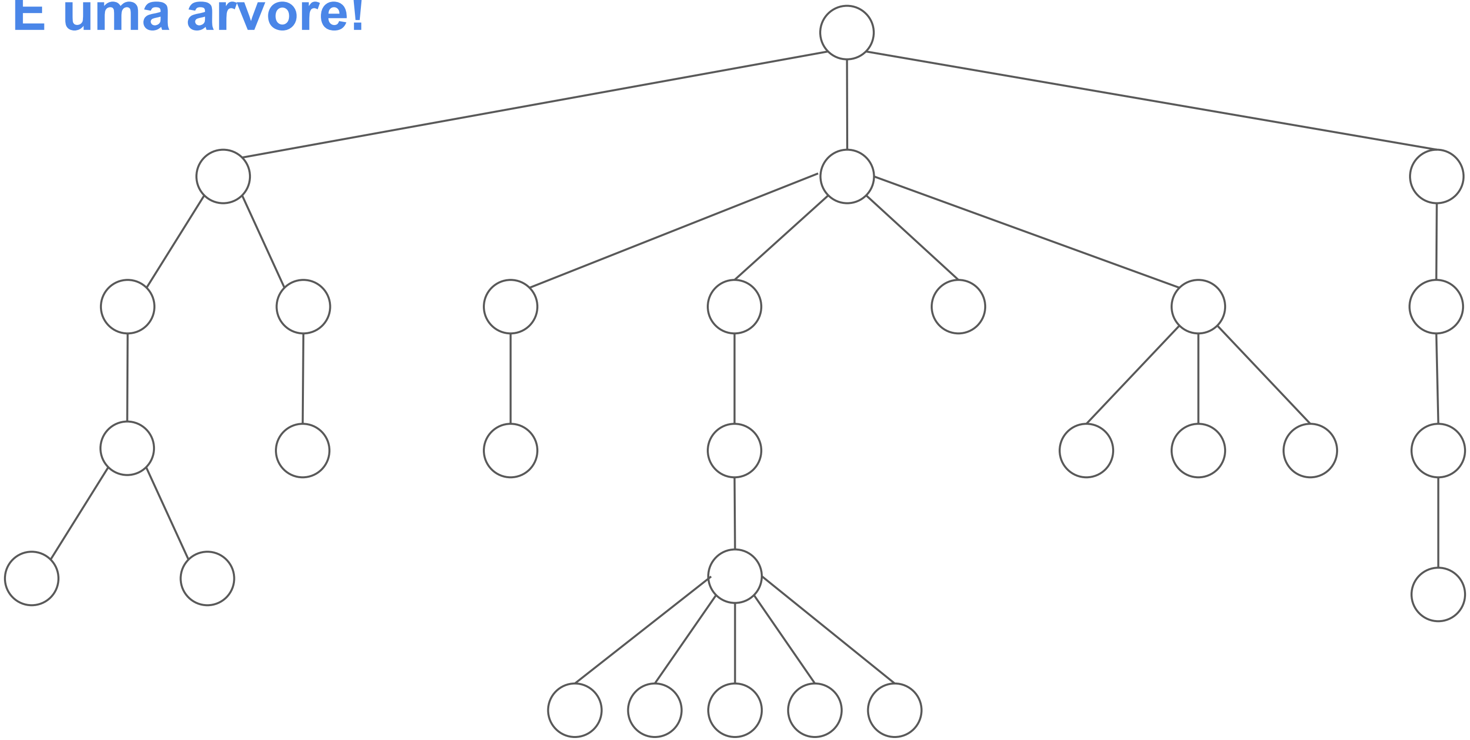


Abstraindo a estrutura de uma árvore



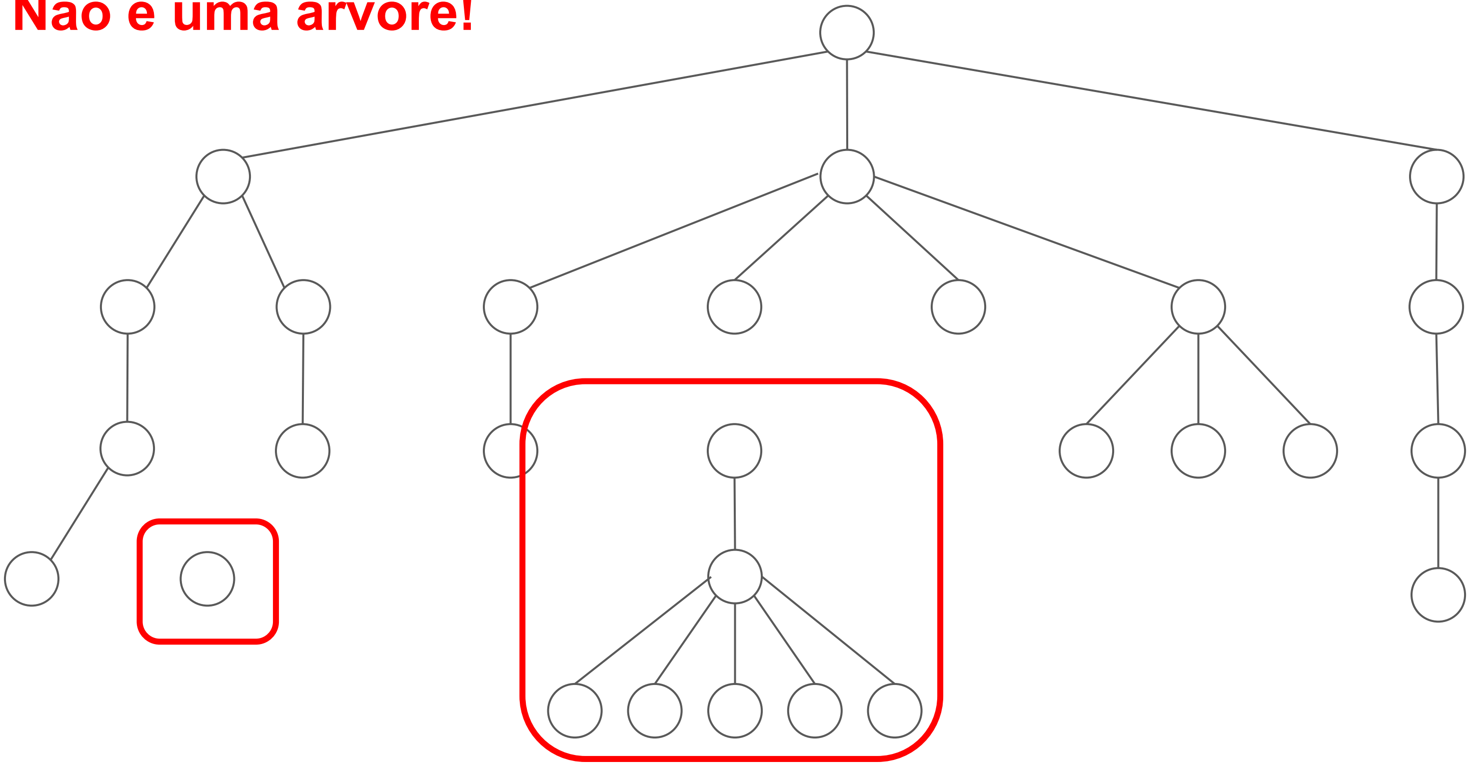
Árvore - grafo conexo

É uma árvore!



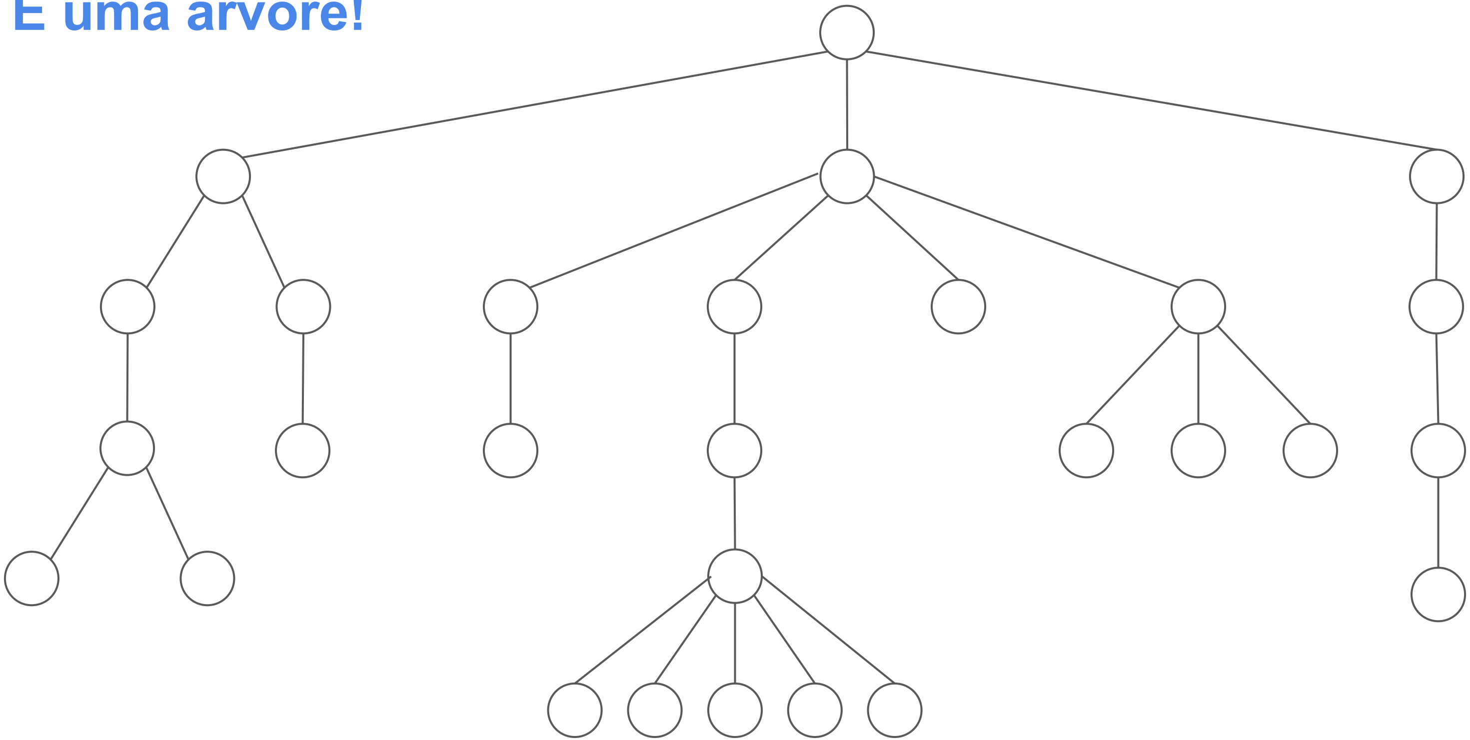
Árvore - grafo conexo

Não é uma árvore!



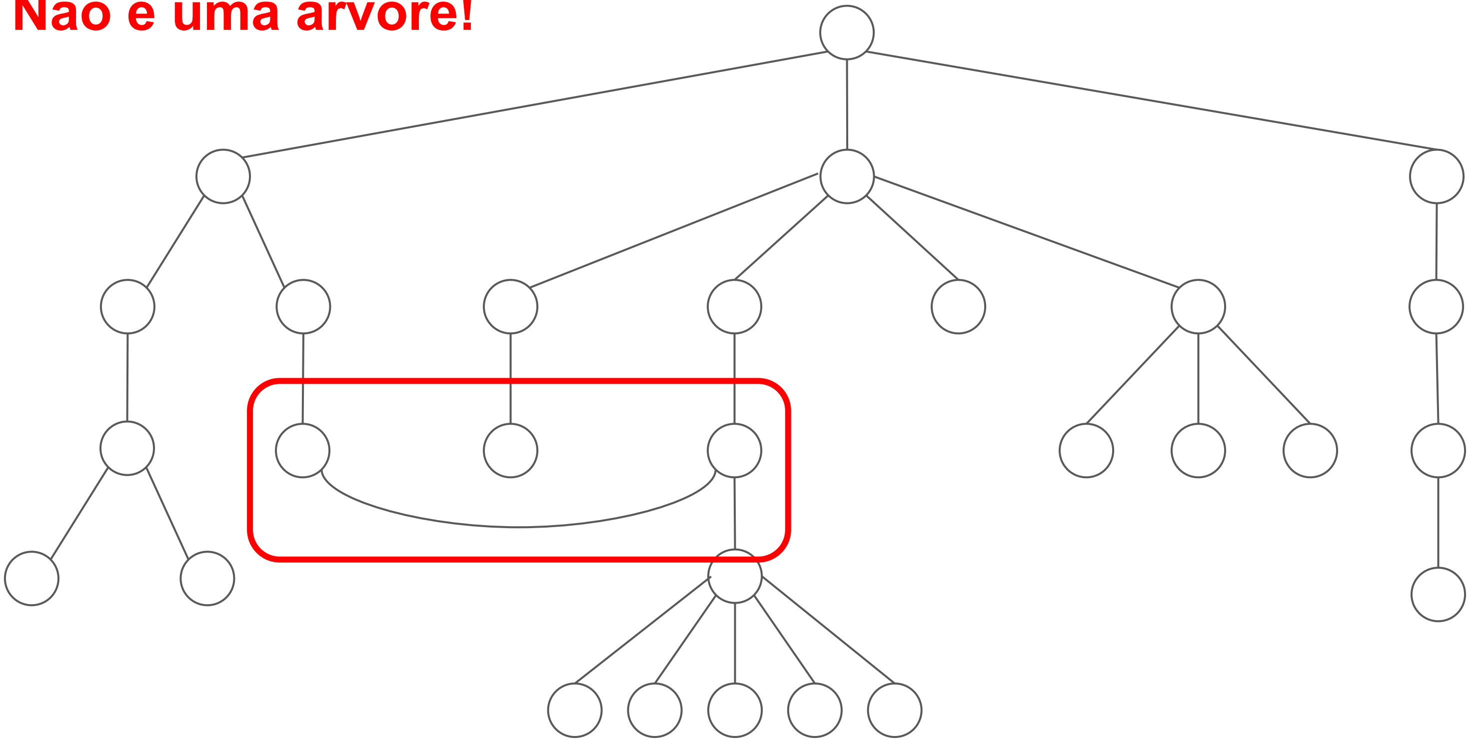
Árvore - grafo acíclico

É uma árvore!



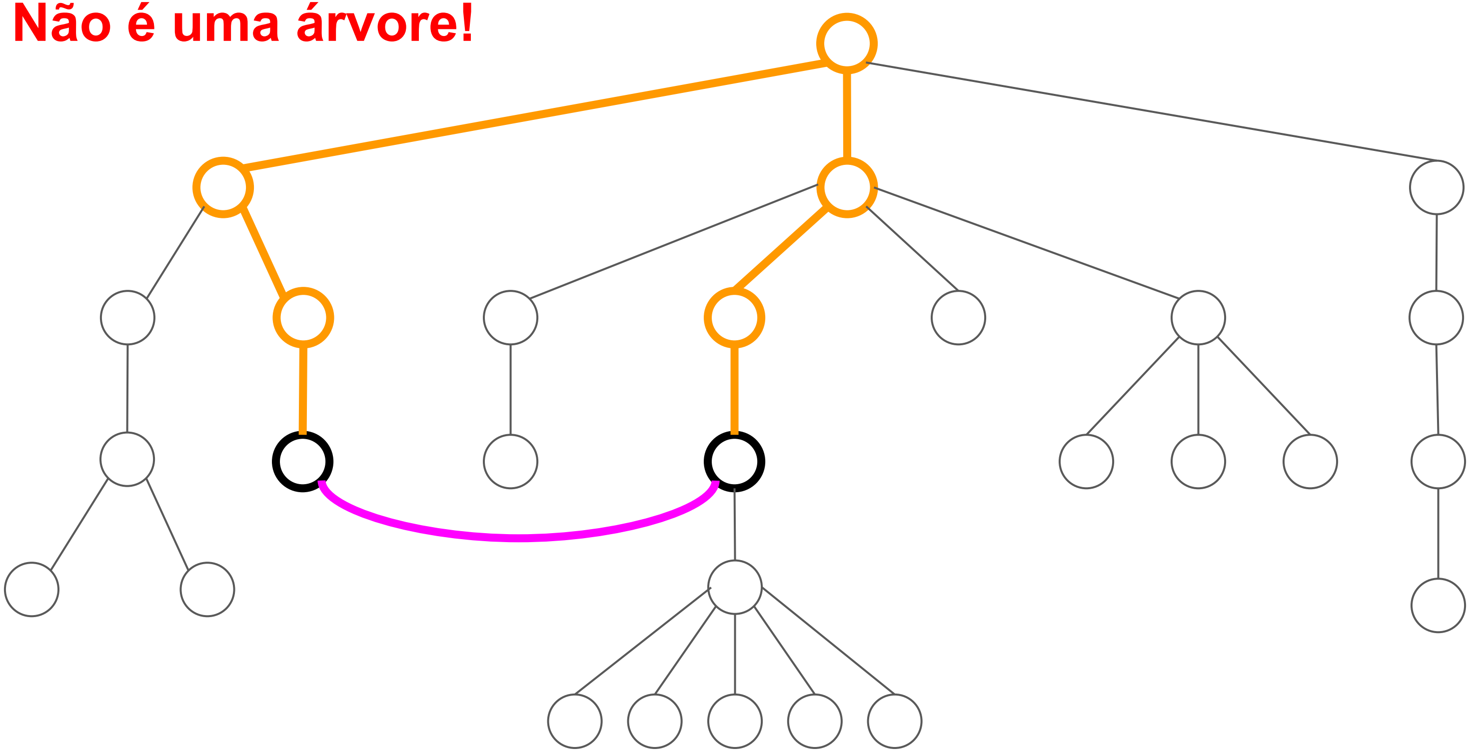
Árvore - grafo acíclico

Não é uma árvore!



Árvore - grafo acíclico

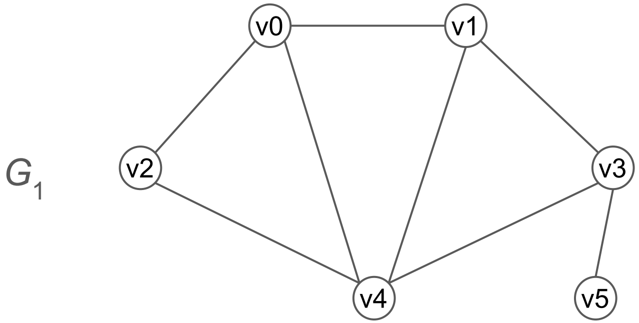
Não é uma árvore!



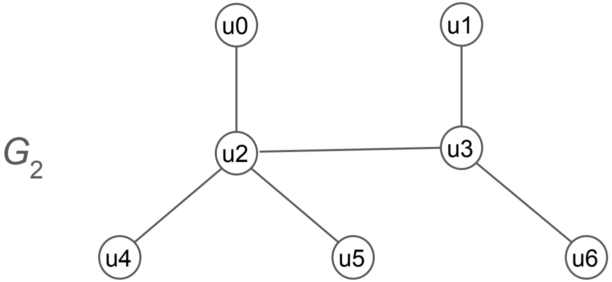
Grafo acíclico

- Um grafo é **acíclico** se não possui ciclos

- Exemplo:



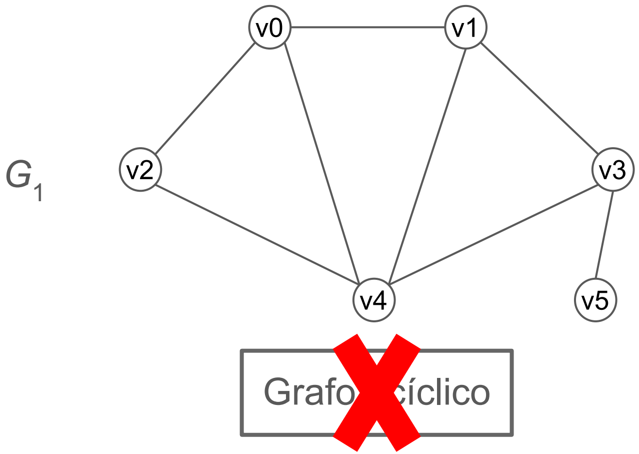
Grafo acíclico

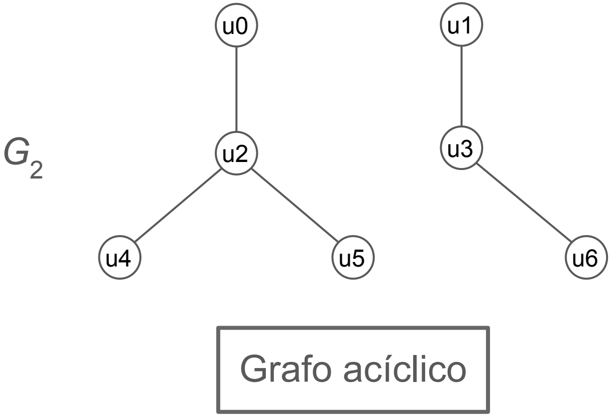


Grafo acíclico

Grafo acíclico

- Um grafo é **acíclico** se não possui ciclos
- Exemplo:



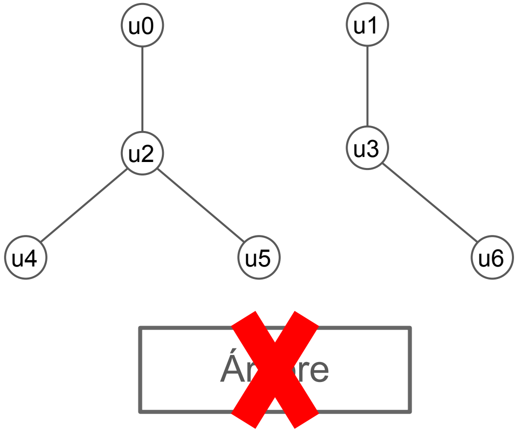


Árvore

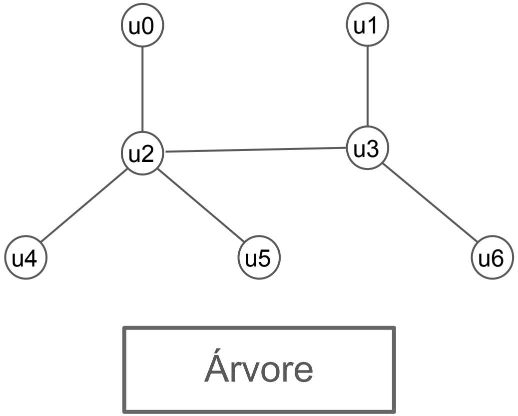
- Uma **árvore** é um grafo conexo acíclico

- Exemplo:

G_1



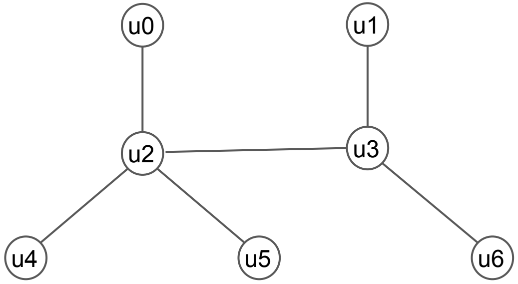
G_2



Árvore

- Uma **árvore** é um grafo conexo acíclico
- Uma **folha** de uma árvore é um vértice de grau 1 da árvore

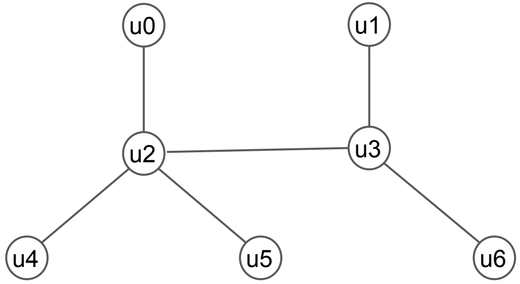
- Exemplo:
  - As folhas da árvore ao lado são



Árvore

- Uma **árvore** é um grafo conexo acíclico
- Uma **folha** de uma árvore é um vértice de grau 1 da árvore

- Exemplo:
  - As folhas da árvore ao lado são u_0, u_1, u_4, u_5, e u_6



Propriedades de uma árvore

- **Teorema:** Dado um grafo *G* com *n* vértices, as seguintes afirmações são equivalentes:
  1. *G* é uma árvore;
  2. *G* é conexo e possui *n* - 1 arestas;
  3. *G* é acíclico e possui *n* - 1 arestas;
  4. Existe exatamente um caminho entre quaisquer dois vértices de *G*;
  5. *G* é conexo, mas a remoção de qualquer aresta de *G* torna *G* desconexo;
  6. *G* é acíclico, mas a inserção de qualquer aresta em *G* faz com que *G* tenha um ciclo.

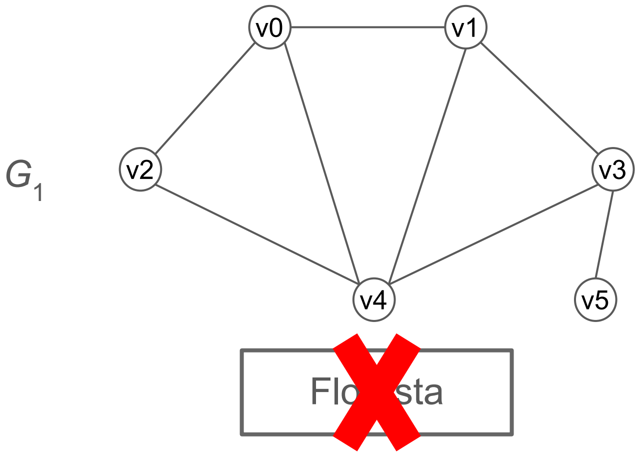
Exercícios

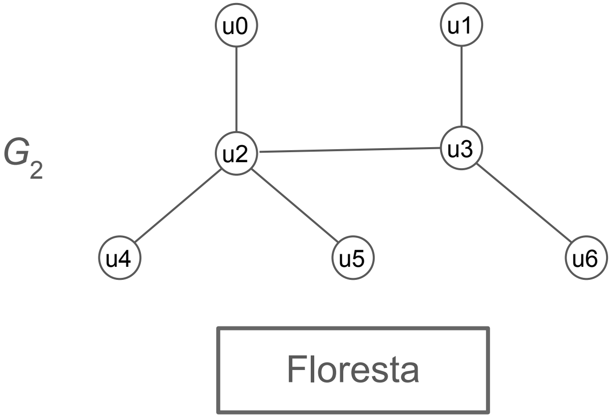
1. Considere o teorema do slide anterior e prove o seguinte:
   a. A Afirmação 1 implica a Afirmação 6;
   b. A Afirmação 6 implica a Afirmação 1.

Floresta

- Uma **floresta** é um grafo ~~conexo~~ acíclico

- Exemplo:

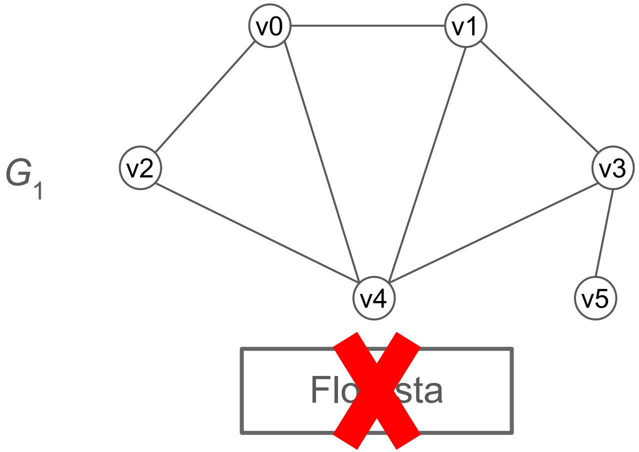


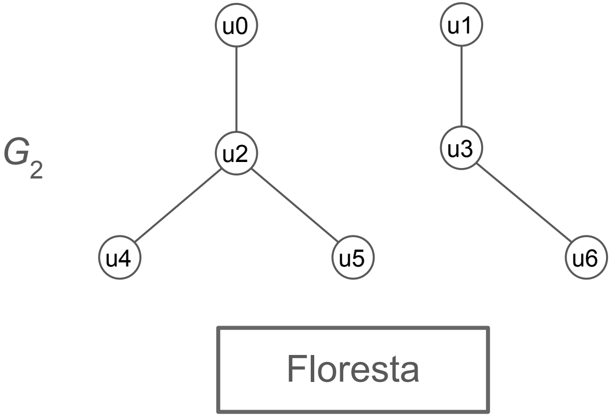


Floresta

- Uma **floresta** é um grafo ~~conexo~~ acíclico

- Exemplo:





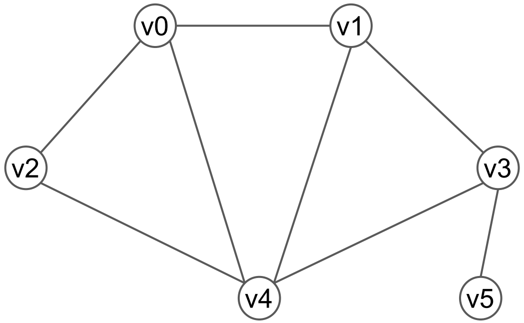
Floresta

- Uma **floresta** é um grafo ~~conexo~~ acíclico

> As **componentes conexas** de uma floresta são **árvores** 🌳🌳🌳🌳

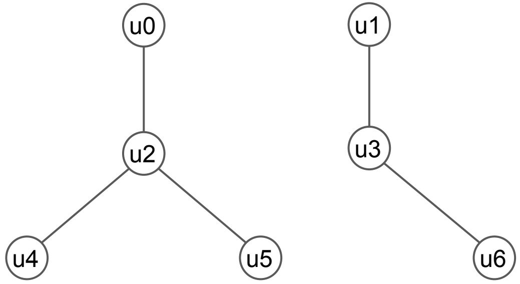
- Exemplo:

G_1



Floresta

G_2

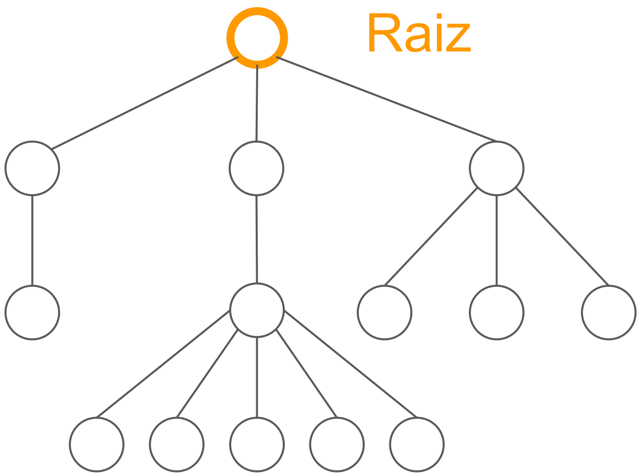


Floresta

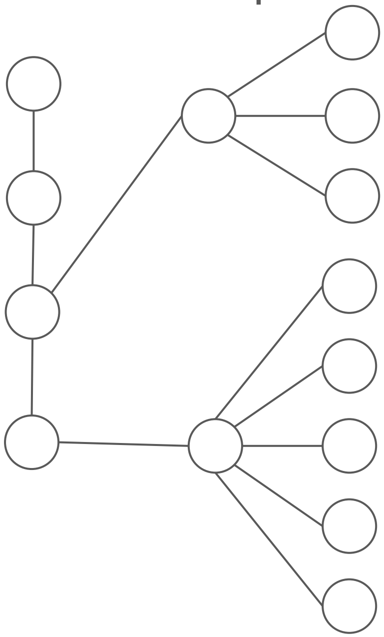
Árvore enraizada

- Uma **árvore enraizada** é uma árvore em que um dos vértices é especificado como a raiz

- Exemplo:



Árvore enraizada

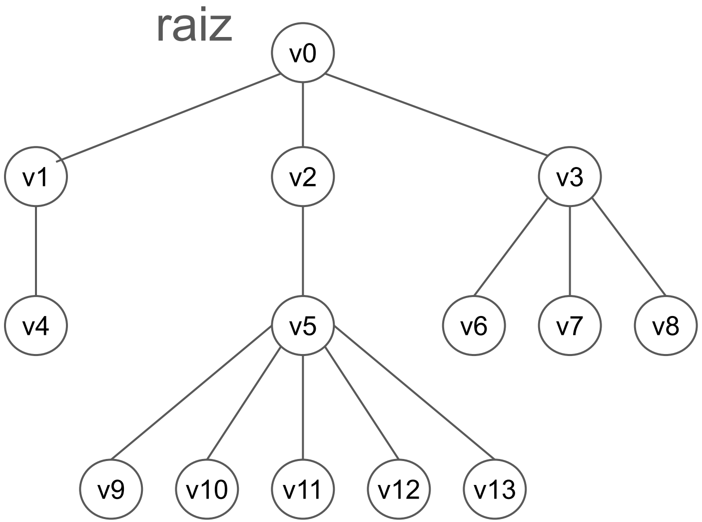


Árvore não enraizada

Árvore enraizada - terminologia

- Dada uma árvore com raiz *r*, se a última aresta do caminho entre o vértice *r* e um vértice *v* na árvore é a aresta *uv*, então dizemos que *u* é o **pai** de *v* e *v* é um **filho** de *u*

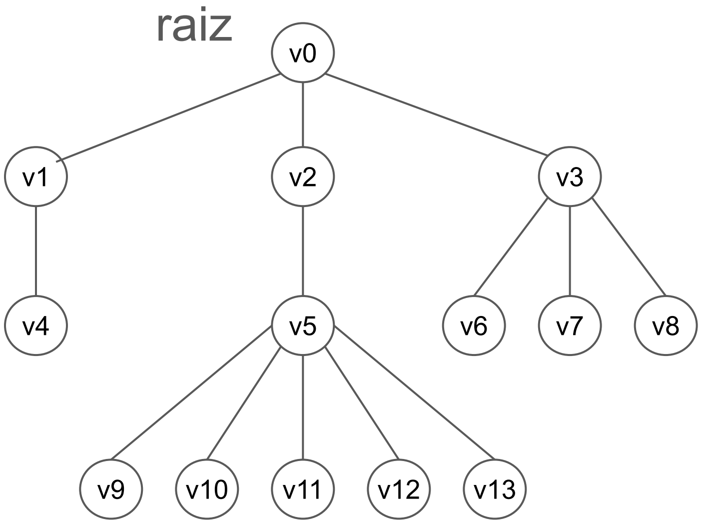
- Exemplo:
  - v2 é   de v0
  - v5 é   de v10
  - v13 é   de v5
  - v6 é   de v7



Árvore enraizada - terminologia

- Dada uma árvore com raiz *r*, se a última aresta do caminho entre o vértice *r* e um vértice *v* na árvore é a aresta *uv*, então dizemos que *u* é o **pai** de *v* e *v* é um **filho** de *u*

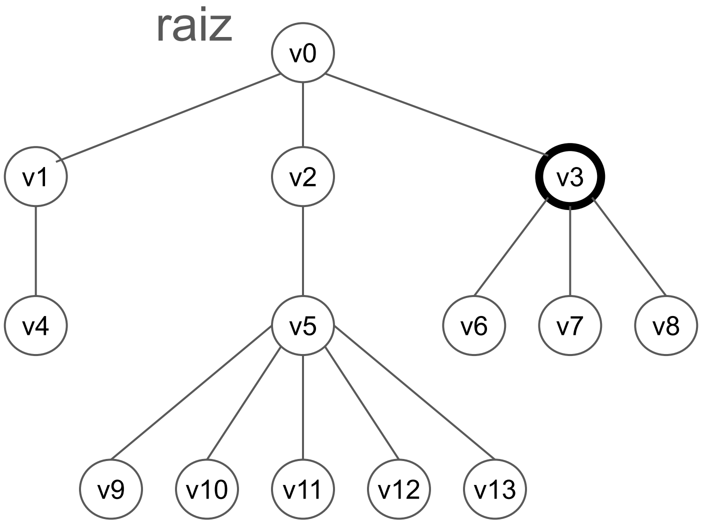
- Exemplo:
  - v2 é filho de v0
  - v5 é pai de v10
  - v13 é filho de v5
  - v6 não é pai nem filho de v7 (v6 é **irmão** de v7)



Árvore enraizada - terminologia

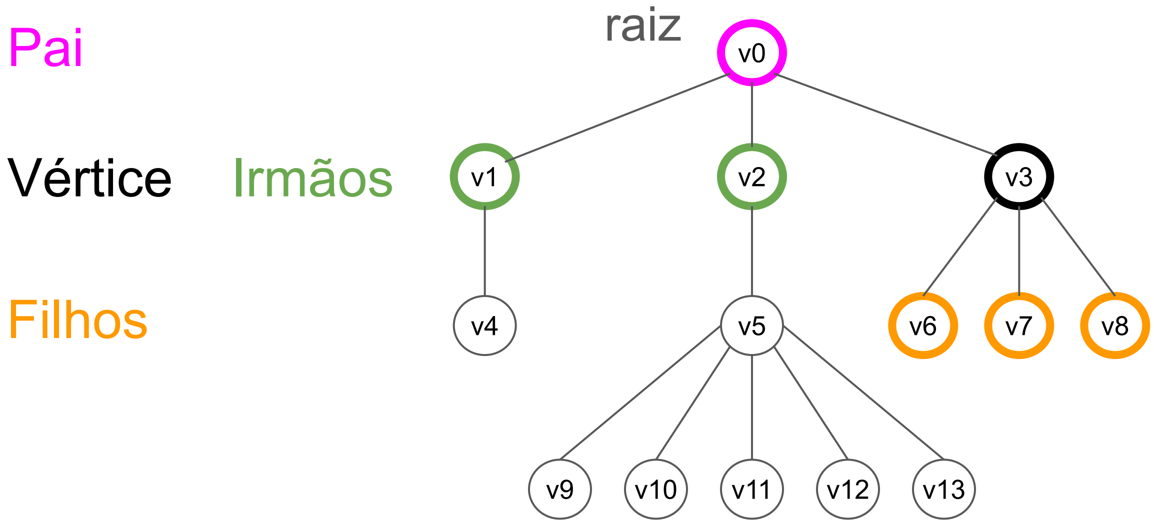
- Dada uma árvore com raiz *r*, se a última aresta do caminho entre o vértice *r* e um vértice *v* na árvore é a aresta *uv*, então dizemos que *u* é o **pai** de *v* e *v* é um **filho** de *u*. Vértices que têm o mesmo pai são chamados de **irmãos**

Vértice



Árvore enraizada - terminologia

- Dada uma árvore com raiz *r*, se a última aresta do caminho entre o vértice *r* e um vértice *v* na árvore é a aresta *uv*, então dizemos que *u* é o **pai** de *v* e *v* é um **filho** de *u*. Vértices que têm o mesmo pai são chamados de **irmãos**

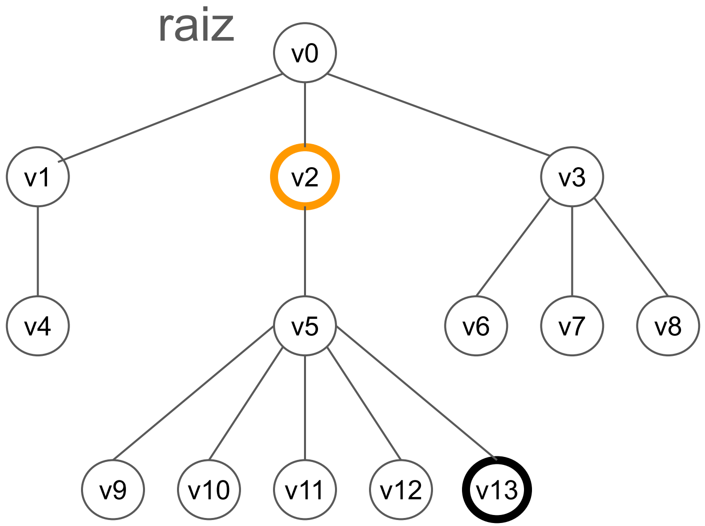


Árvore enraizada - terminologia

- Dada uma árvore com raiz *r*, se *u* é um vértice do caminho entre o vértice *r* e um vértice *v* na árvore, então dizemos que *u* é um **ancestral** de *v*

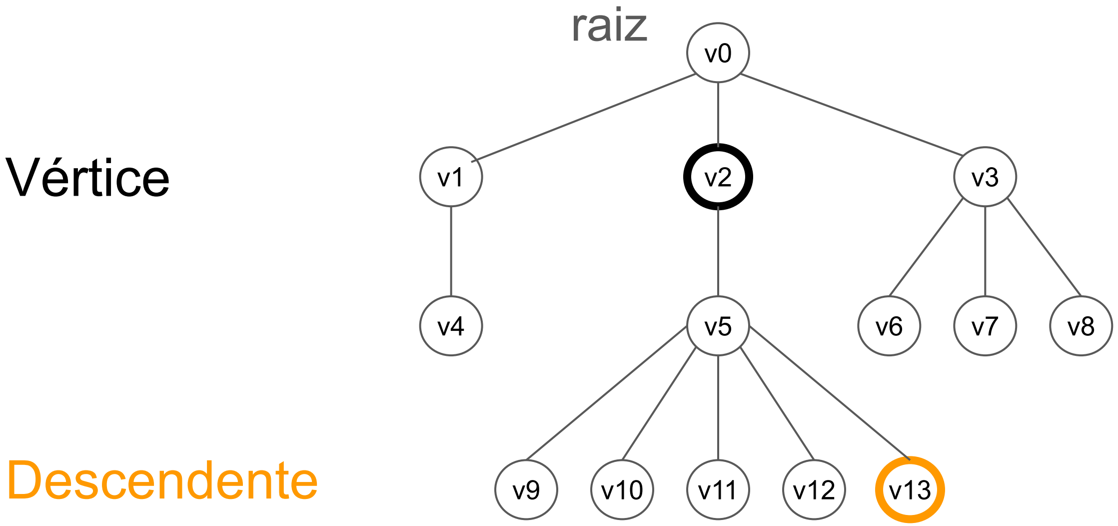
Ancestral

Vértice



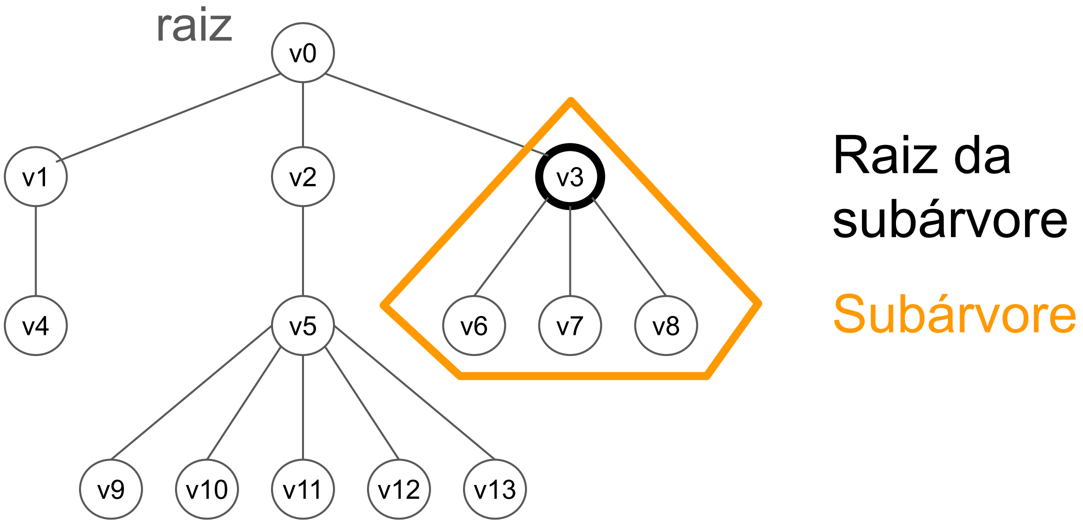
Árvore enraizada - terminologia

- Dada uma árvore com raiz *r*, se *u* é um vértice do caminho entre o vértice *r* e um vértice *v* na árvore, então dizemos que *u* é um **ancestral** de *v*. Se um vértice *u* é ancestral de um vértice *v*, chamamos *v* de um **descendente** de *u*



Árvore enraizada - terminologia

- Dada uma árvore enraizada *T*, chamamos de **subárvore** de *T* com raiz *v*, a árvore induzida por *v* e seus descendentes em *T*



Representação computacional de uma árvore

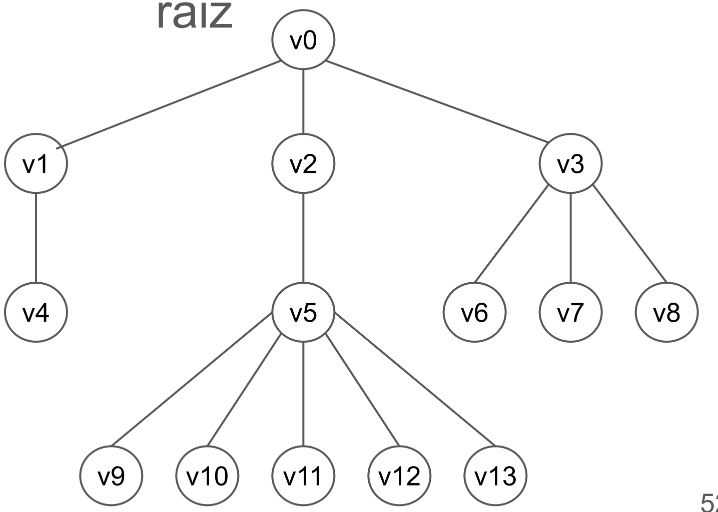
- Uma árvore é um grafo e, portanto, pode ser representada como uma matriz de adjacências ou listas de adjacência ou de outra forma usual de representar um grafo
- Além disso, uma árvore pode ser representada como uma estrutura mais simples

Representação computacional de uma árvore

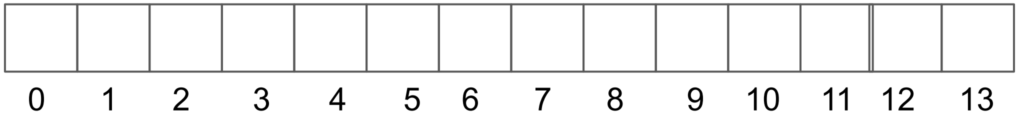
- Podemos representar uma árvore *G* com raiz *r* como um **vetor** *pai* de $|V(G)|$ elementos, com índices 0, 1, …, $|V(G)| - 1$, tal que
  - *pai*[*i*] é igual ao pai do vértice *i* em *G* caso $i \neq r$ e
  - *pai*[*r*] = -1

- Exemplo:

raiz



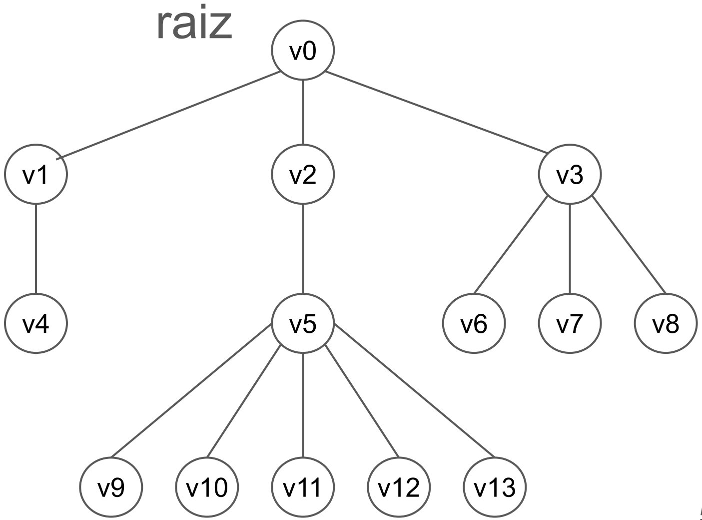
pai



Representação computacional de uma árvore

- Podemos representar uma árvore *G* com raiz *r* como um **vetor** *pai* de |*V*(*G*)| elementos, com índices 0, 1, …, |*V*(*G*)| - 1, tal que
  - *pai*[*i*] é igual ao pai do vértice *i* em *G* caso *i* ≠ *r* e
  - *pai*[*r*] = -1

- Exemplo:



pai

-1	0	0	0	1	2	3	3	3	5	5	5	5	5
0	1	2	3	4	5	6	7	8	9	10	11	12	13

Referências

- Esta apresentação é baseada nos seguintes materiais:

1. Apêndice B.5 do livro
   Cormen, T. H., Leiserson, C. E., Rivest, R. L., Stein, C. Introduction to Algorithms. 3rd. ed. MIT Press, 2009.
2. Capítulo 17 do livro
   Sedgewick, R. Algorithms in C++ – Part 5. Graph Algorithms. 3rd. ed. Addison-Wesley, 2002.