Universidade Federal de Ouro Preto
Campus João Monlevade

CSI 488 – Algoritmos e Estruturas de Dados I

TAD – Árvores

Prof. Mateus Ferreira Satler

Índice

1. Introdução

- Nesta aula veremos conceitos e definições sobre árvores.

- Diferentemente das estruturas de pilhas, filas e listas que são **lineares**, uma árvore é uma estrutura de dados **não linear**.

1. Introdução

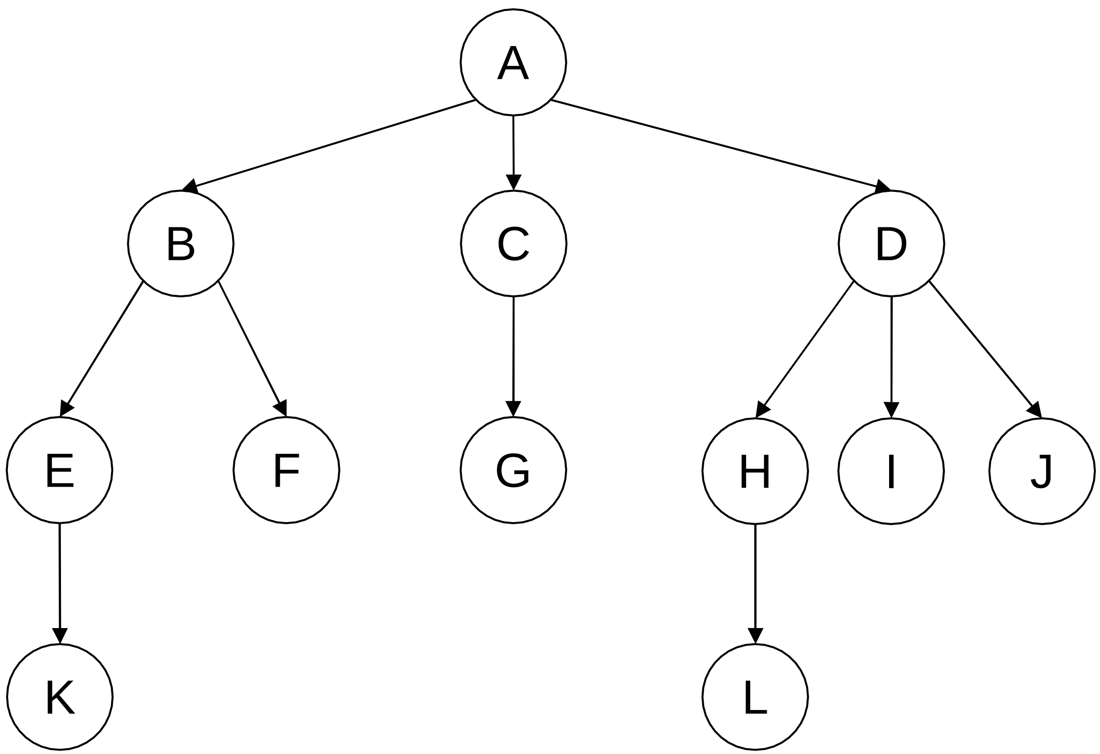
- **Definição:**
  - Uma árvore é um conjunto finito de um ou mais nós (ou vértices) tais que:
    - Existe um nó especial, denominado **raiz.**
  - Os demais nós encontram-se desdobrados em $n \geq 0$ conjuntos disjuntos $T_1, ..., T_n$ sendo que cada conjunto se constitui numa árvore.
  - $T_1, ..., T_n$ são denominadas **sub-árvores** da raiz.

1. Introdução

- Utilizaremos grafos para representar árvores.
- Contudo, existem outras representações equivalentes para árvores: conjuntos aninhados (diagrama de inclusão), parênteses aninhados, endentação, etc.

1. Introdução

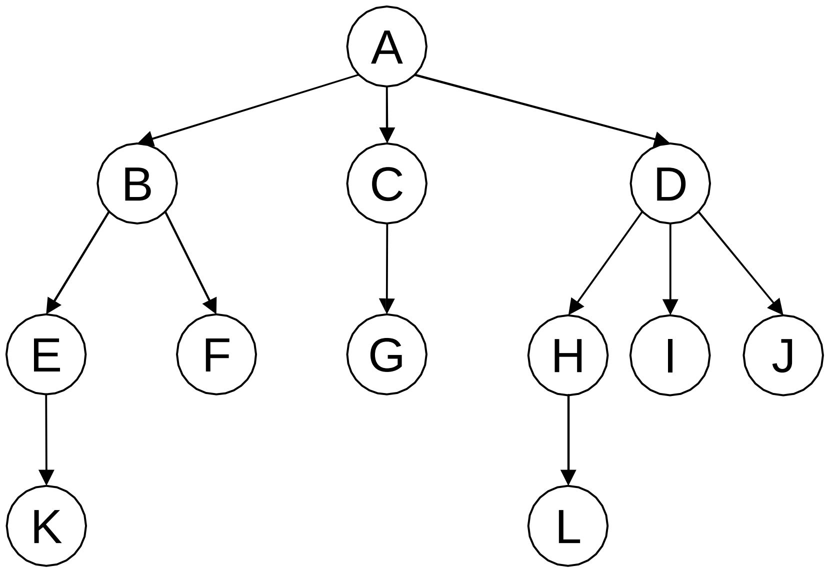
- Uma árvore é um grafo sem ciclos.

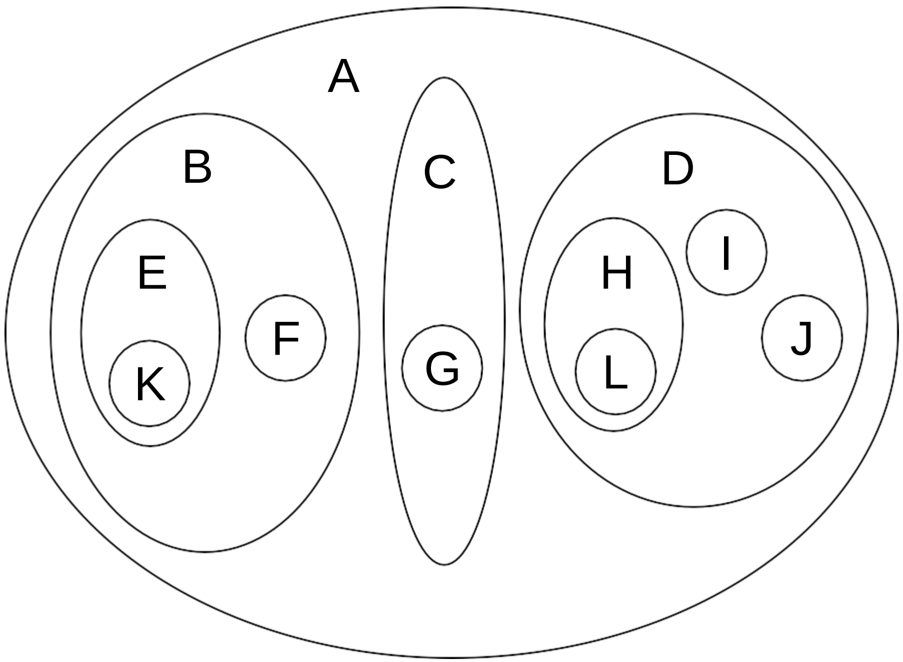


1. Introdução

- Grafo

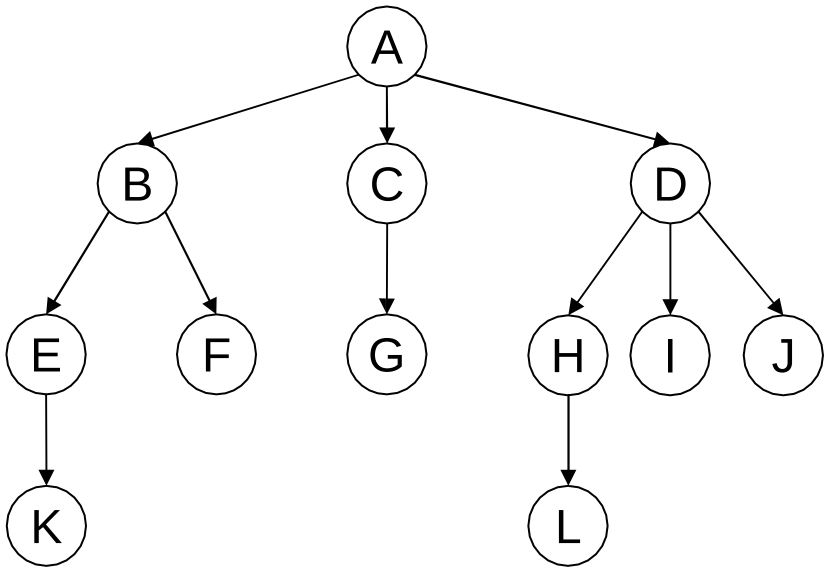
Conjuntos aninhados





1. Introdução

- Grafo

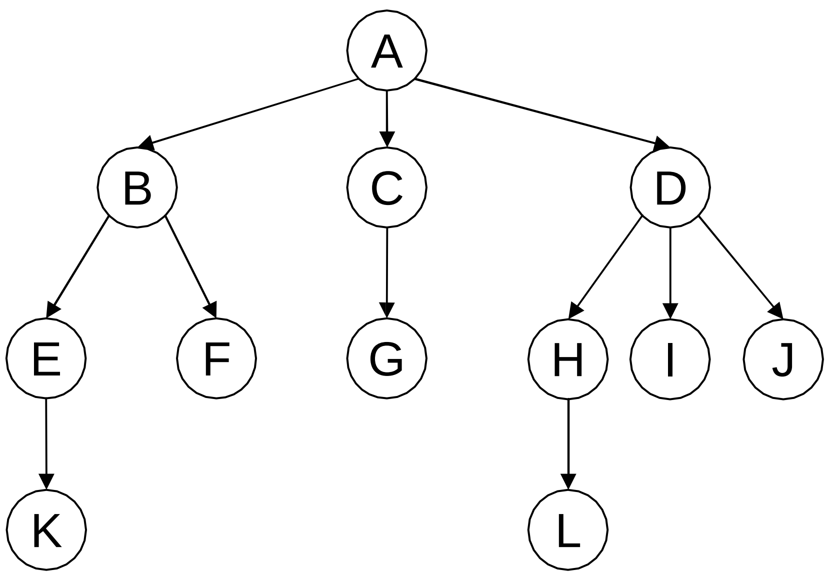


Parênteses aninhados

((A (B (E (K) (F)) C (G) D (H (L) (I) (J)))))

1. Introdução

- Grafo

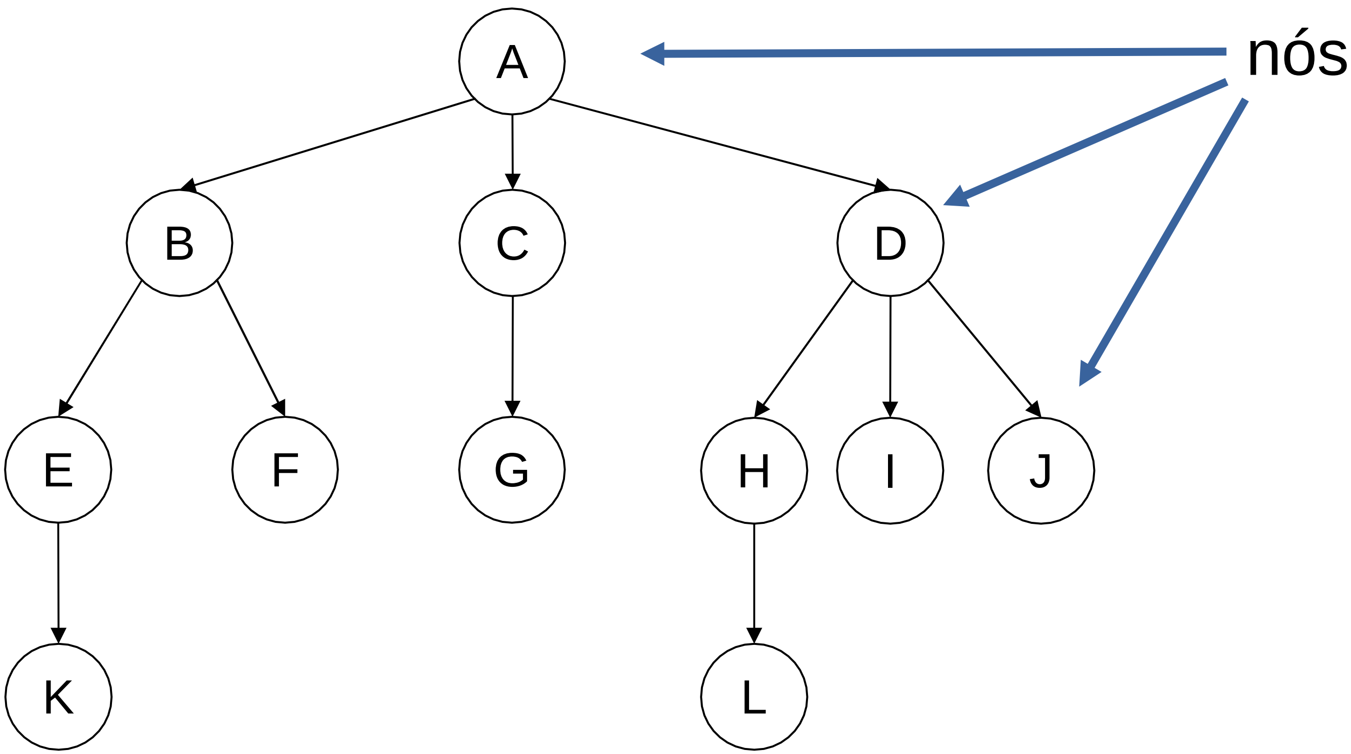


Endentação

```
A
    B
        E
            K
        F
    C
        G
    D
        H
            L
        I
        J
```

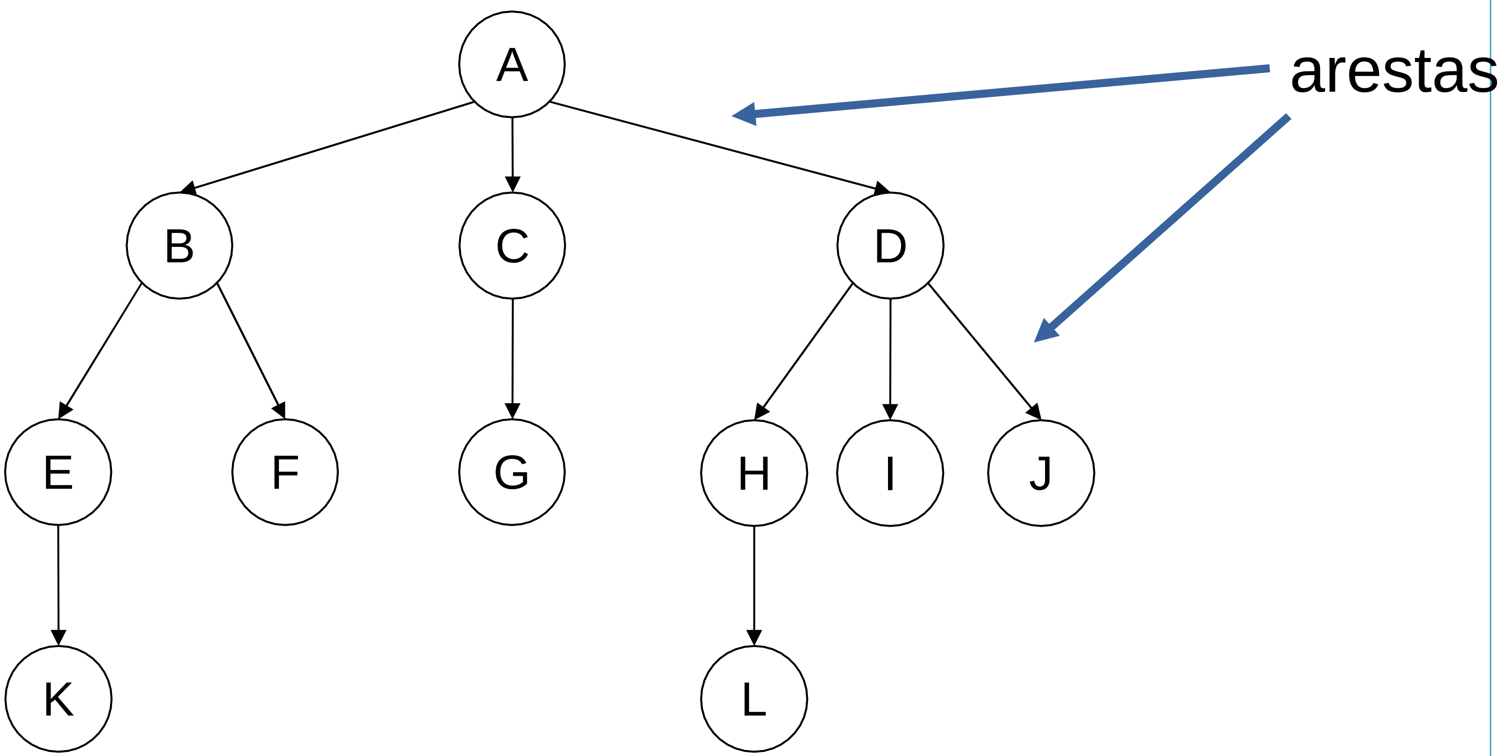
1. Introdução

- **Nós (Vértices)**
  - Esta árvore possui 12 **nós** (ou **vértices**).



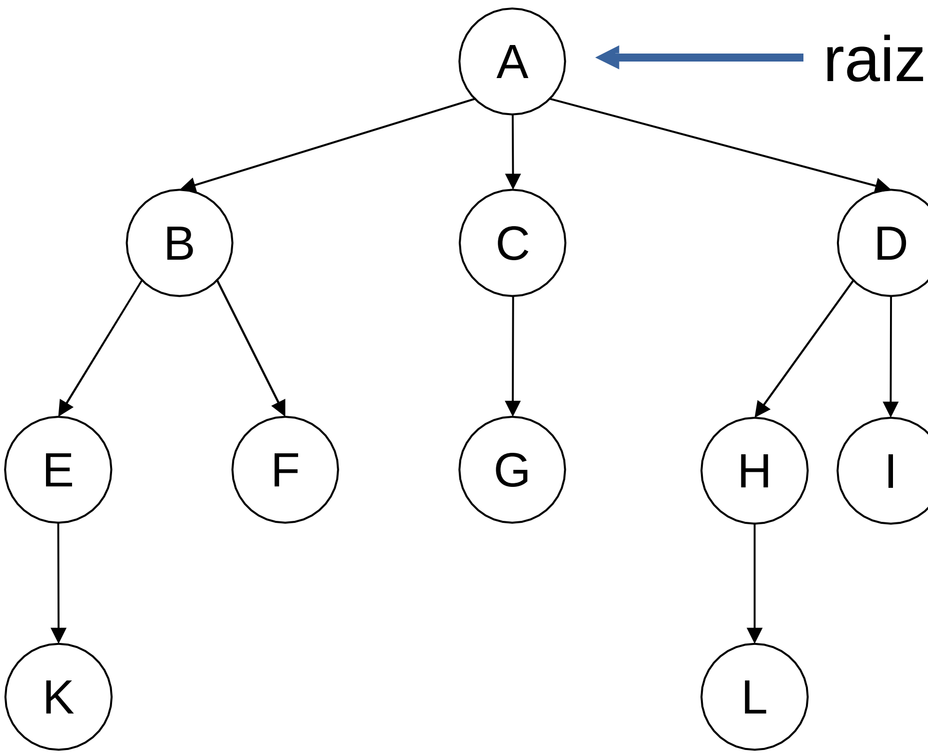
1. Introdução

- **Arestas (Arcos)**
  - Uma **aresta (arco)** liga um nó a outro.



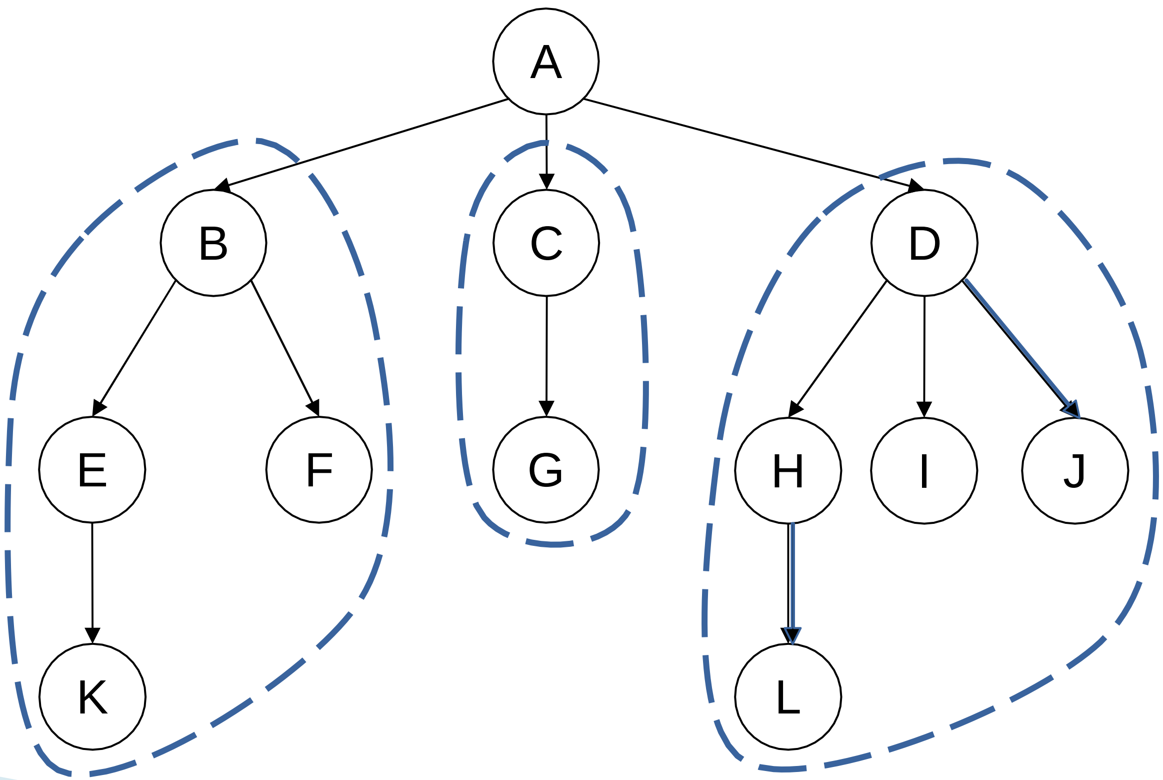
1. Introdução

- **Raiz**
  - Normalmente as árvores são desenhadas de forma invertida, com a raiz em cima.



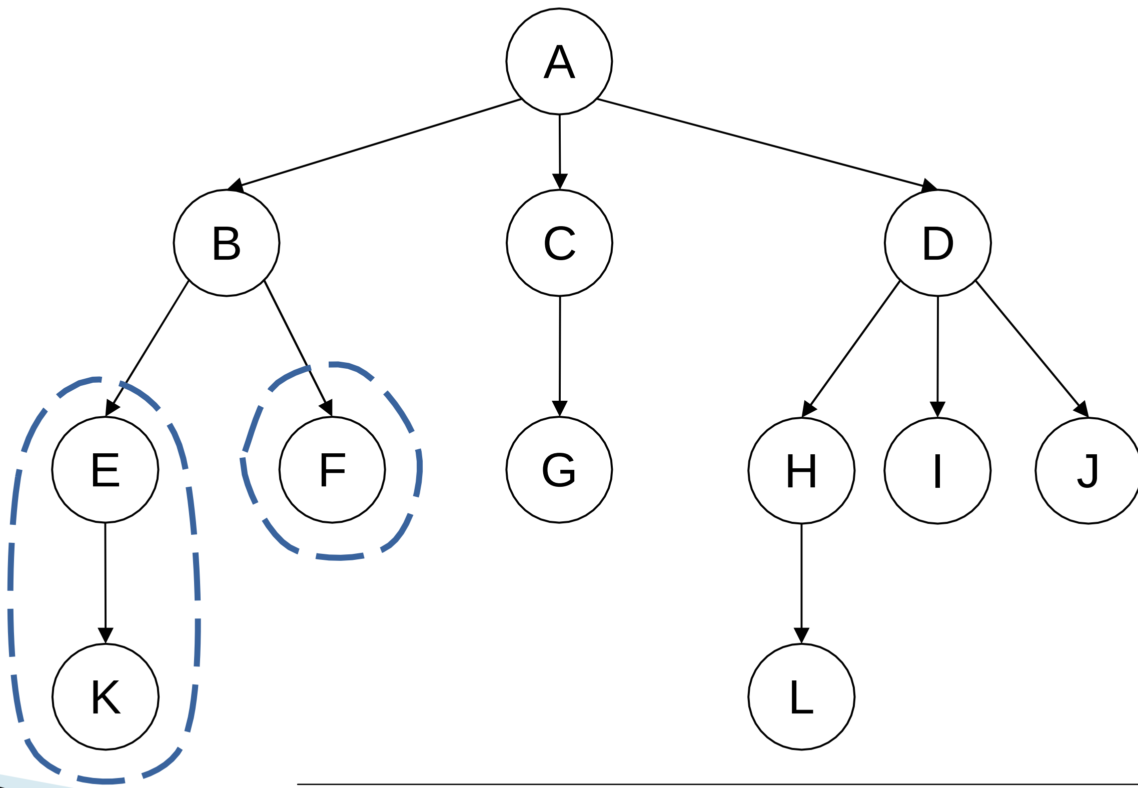
1. Introdução

- **Sub-árvores**
  - No exemplo, o nó A possui três **sub-árvores** (ramos) cujas raízes são B, C e D.



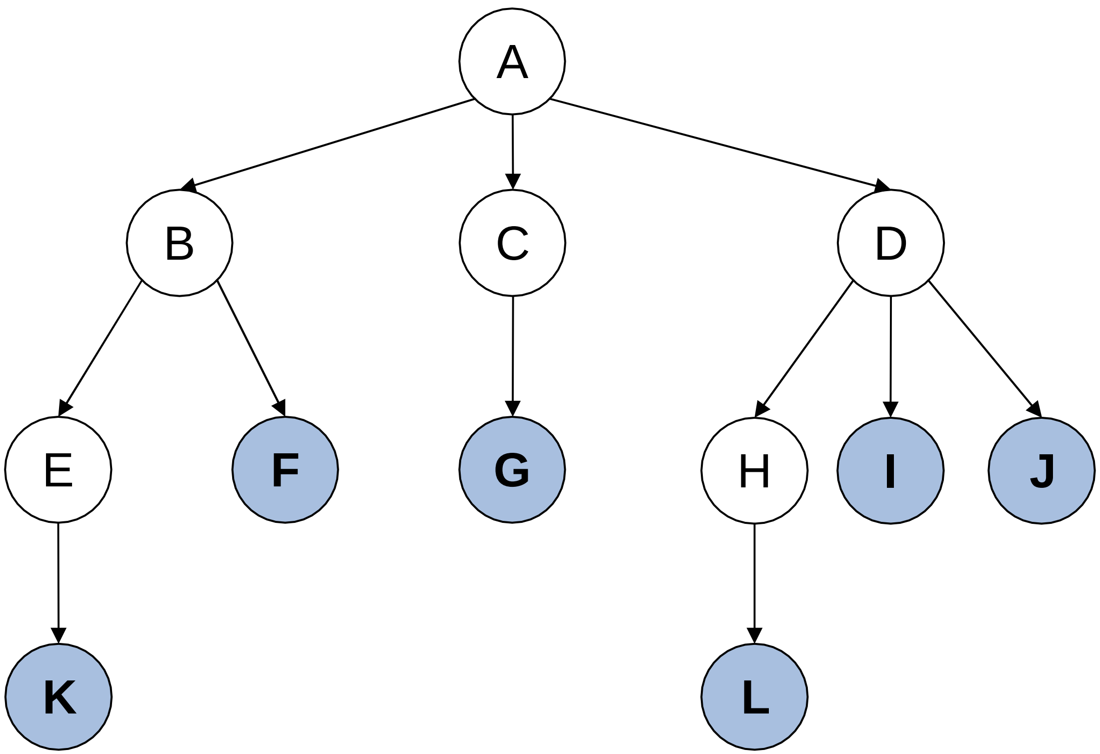
1. Introdução

- **Sub-árvores**
  - No exemplo, o nó B possui duas **sub-árvores** (ramos) cujas raízes são E e F.



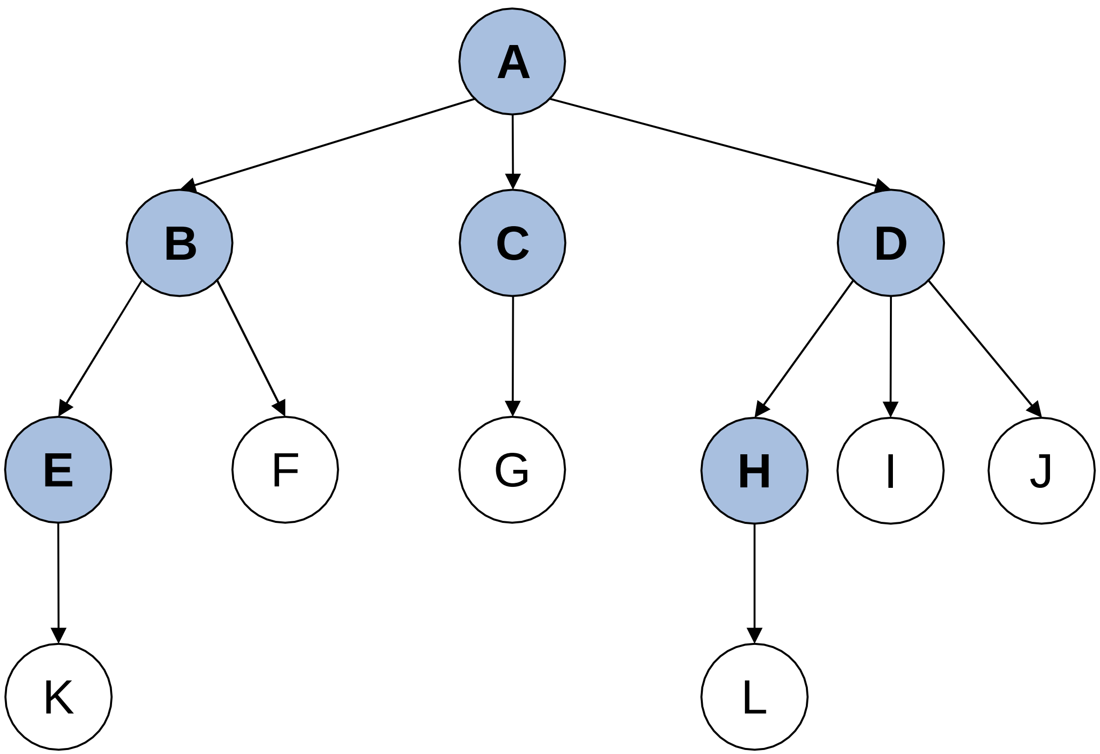
1. Introdução

- **Folha**
  - Um **nó** sem descendentes (sem filhos ou sem sucessores) é denominado **terminal** ou **folha.**



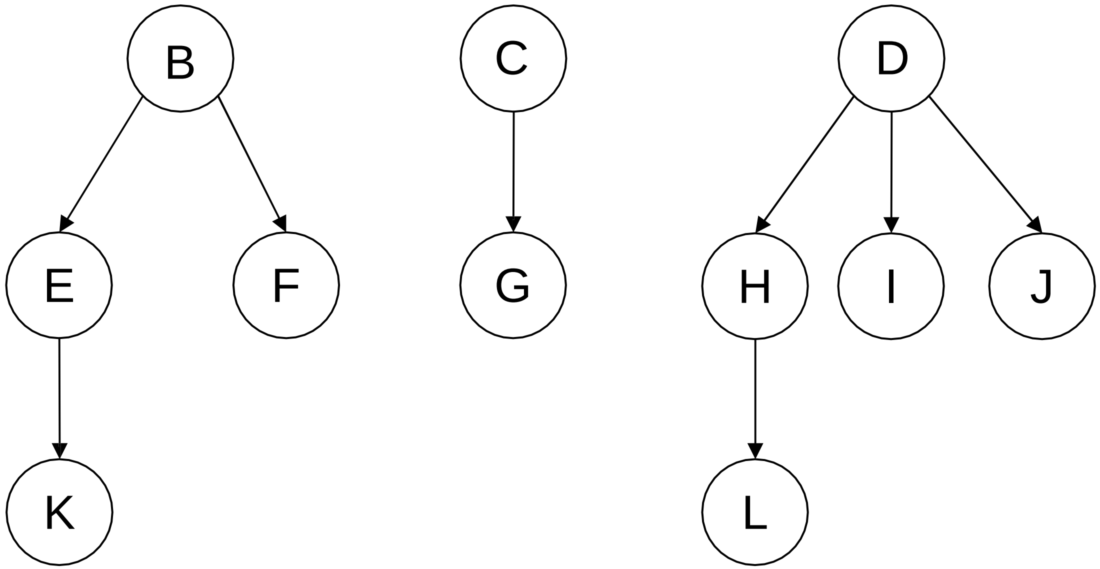
1. Introdução

- **Não-Folha**
  - Um **nó** com descendentes (com filhos ou com sucessores) é denominado **não-terminal** ou **não-folha** ou **interior**.



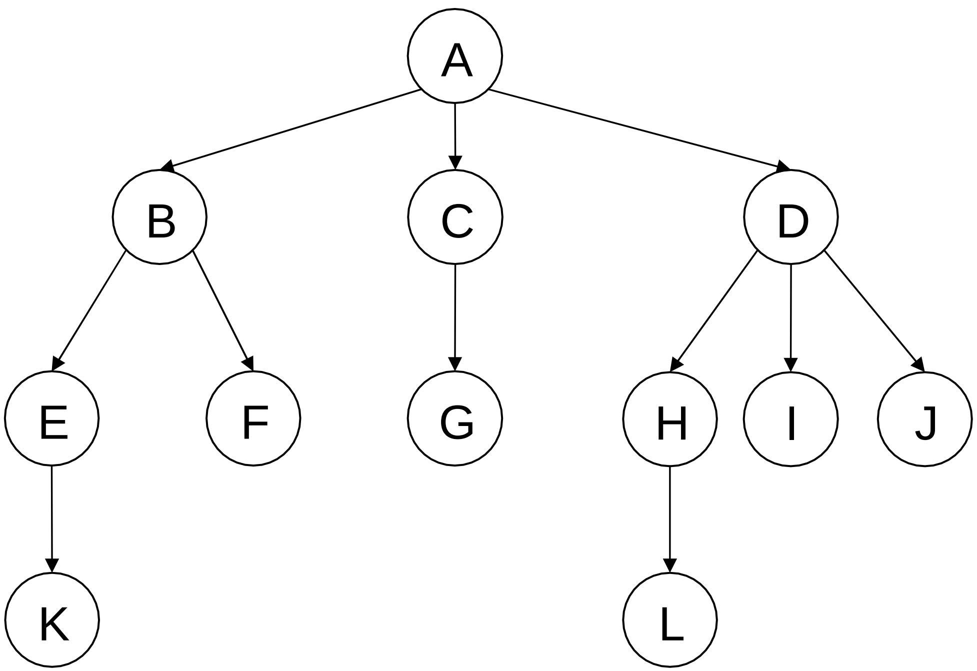
1. Introdução

- **Floresta**
  - Uma **floresta** é um conjunto de zero ou mais árvores.
  - No exemplo, temos 3 árvores que compõem uma floresta.



1. Introdução

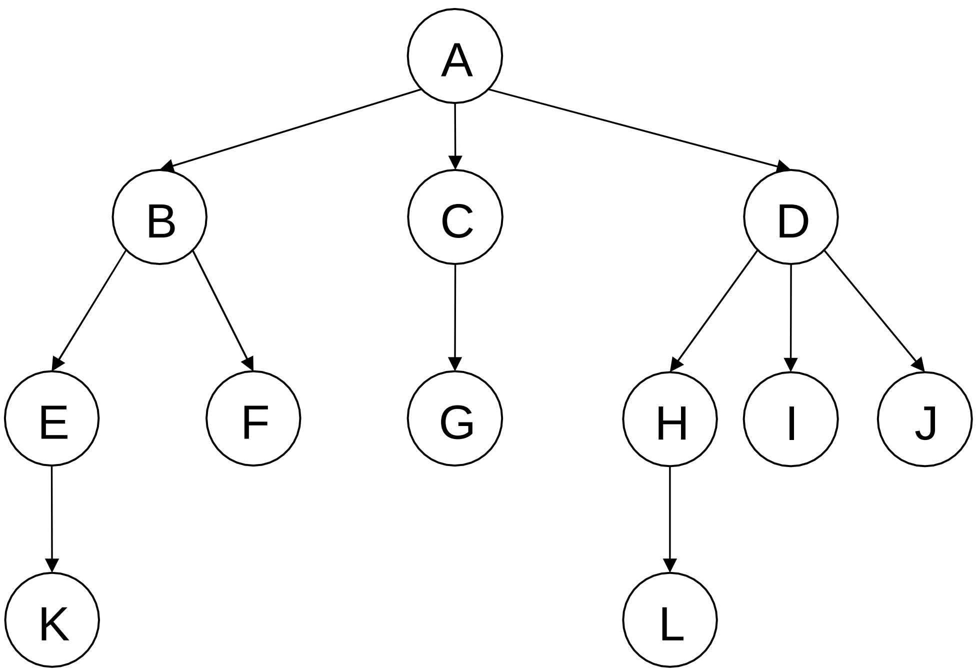
- **Grau de um Nó**
  - O número de descendentes (imediatos) de um nó é denominado **grau** deste nó.



Nó	Grau
A	
B	
C	
D	
E	
F	
G	
H	
I	
J	
K	
L	

1. Introdução

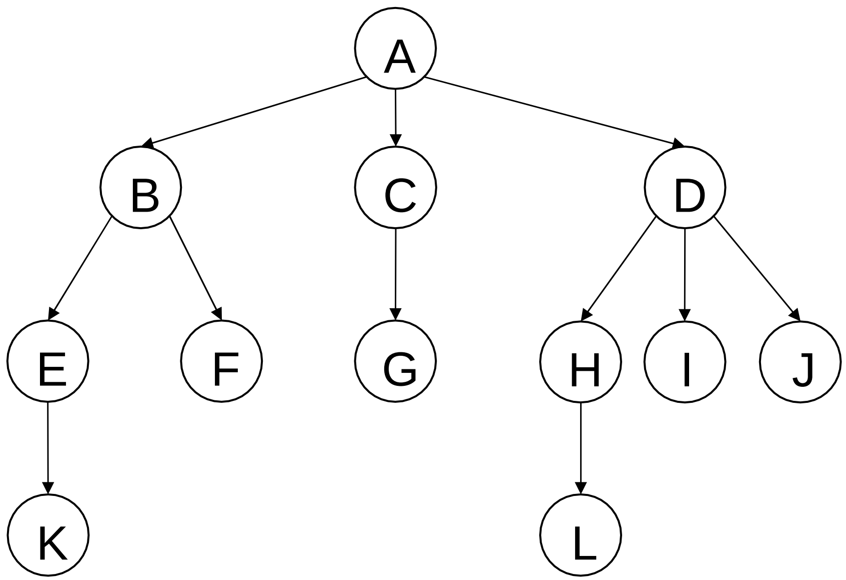
- **Grau de um Nó**
  - O **grau** de uma **folha** é zero.



Nó	Grau
A	3
B	2
C	1
D	3
E	1
F	0
G	0
H	1
I	0
J	0
K	0
L	0

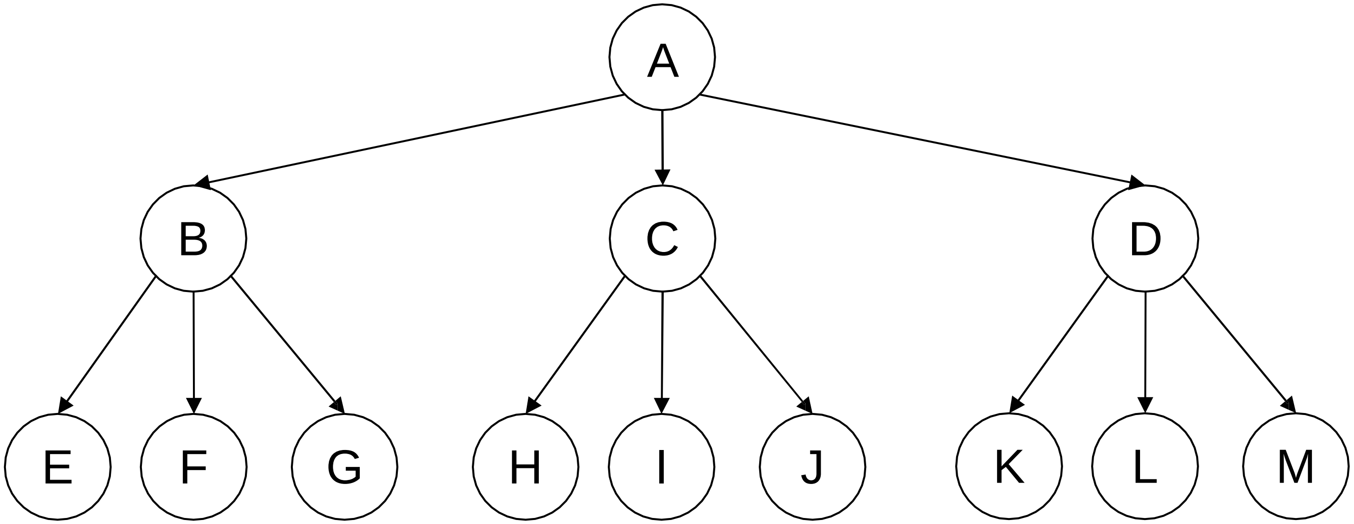
1. Introdução

- **Grau de uma Árvore**
  - O grau máximo atingido pelos nós de uma árvore é denominado **grau** desta árvore.
  - No exemplo, o grau da árvore é 3.



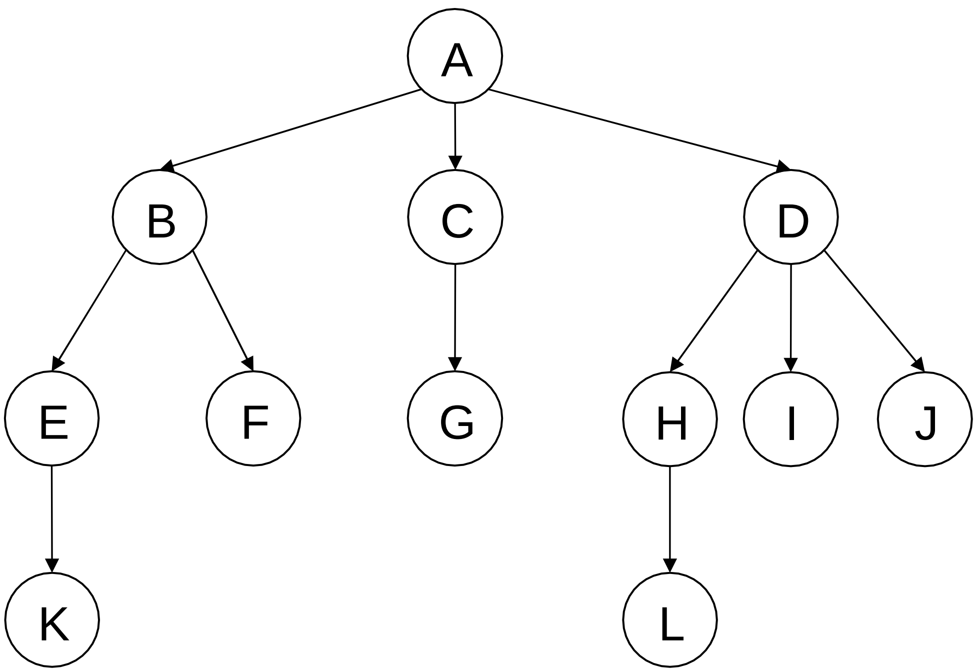
1. Introdução

- **Árvore Completa**
  - Uma árvore de grau **d** é uma árvore **completa** (cheia) se:
    - Todos os nós tem **exatamente d filhos**, exceto as folhas e,
    - Todas as folhas estão na mesma altura.
  - No exemplo, a árvore de grau **d=3** é completa.



1. Introdução

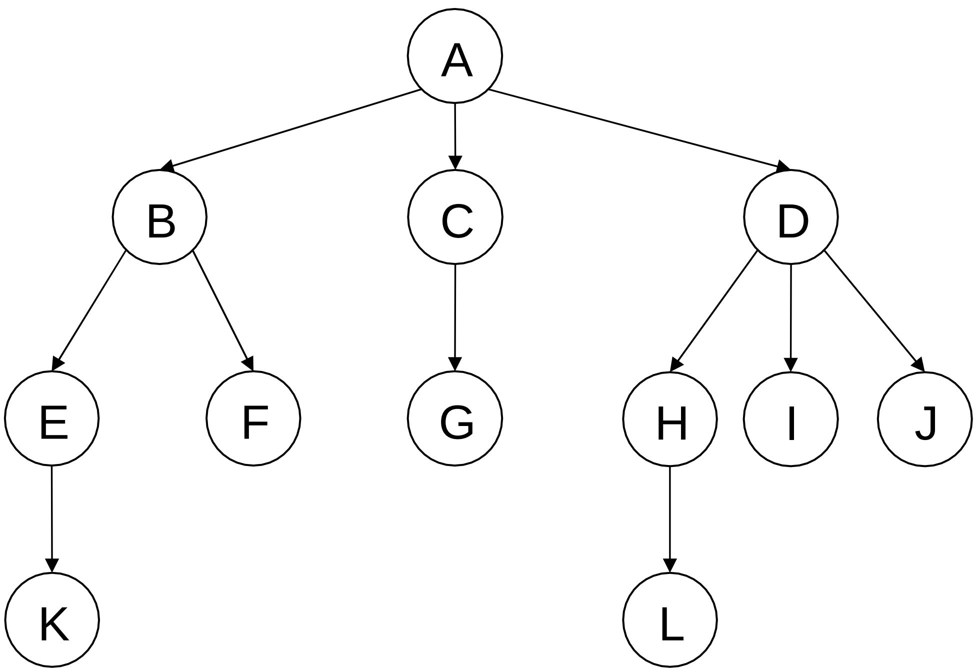
- **Irmão**
  - Os filhos (descendentes) de um mesmo nó pai (antecessor) são denominados **irmãos**.



Irmãos
B, C, D
E, F
H, I, J

1. Introdução

- **Avô & Demais Parentes**
  - Podemos estender essa terminologia para **avô, bisavô,** e demais parentes.



Nós	Avô
E,F,G,H,I,J	A
K	B
L	D

Nós	Bisavô
K, L	A

1. Introdução

- **Caminho**
  - Uma sequência de nós distintos $v_1, v_2, ..., v_k$ tal que sempre existe a relação:

    "v_i é filho de v_{i+1}" ou "v_i é pai de v_{i+1}", $1 \leq i < k$

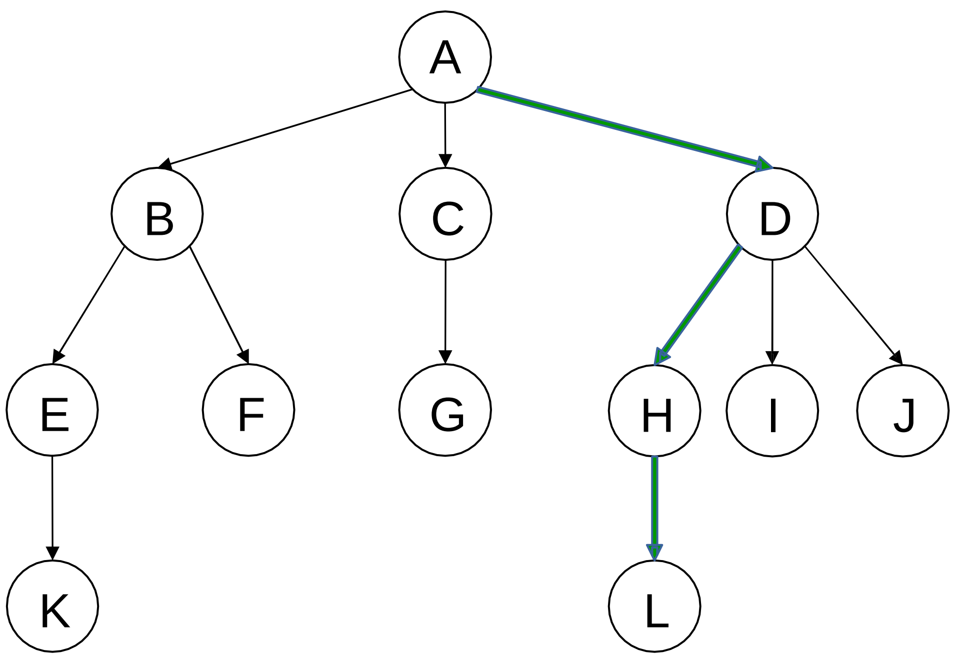
    é denominada um **caminho** entre v_1 e v_k
  - Diz-se que v_1 **alcança** v_k ou que v_k é **alcançado** por v_1

1. Introdução

- **Caminho**
  - Um caminho de k vértices v_1, v_2, ..., v_k é formado pela sequência de k−1 pares de nós (v_1,v_2), (v_2,v_3), ..., (v_{k-2},v_{k-1}), (v_{k-1},v_k)
    - k−1 é o comprimento do caminho
    - Cada par (v_i, v_{i+1}) é uma aresta ou arco, $1 \leq i < k$

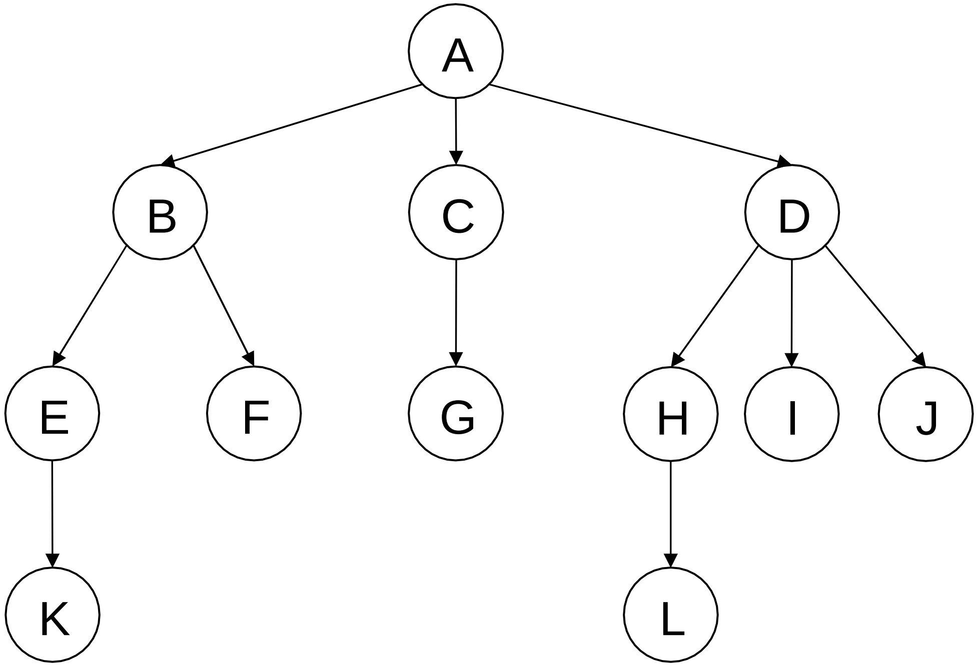
1. Introdução

- **Caminho**
  - **Exemplo:** A, D, H, L é um caminho entre A e L, formando pela sequência de arestas (A,D), (D,H), (H,L).
    - O comprimento do caminho entre A e L é 3.



1. Introdução

- **Antecessores**
  - Os **antecessores** (**antepassados**) de um nó são todos os nós no caminho entre a raiz e o respectivo nó.
  - No exemplo, os antecessores de L são A, D e H.

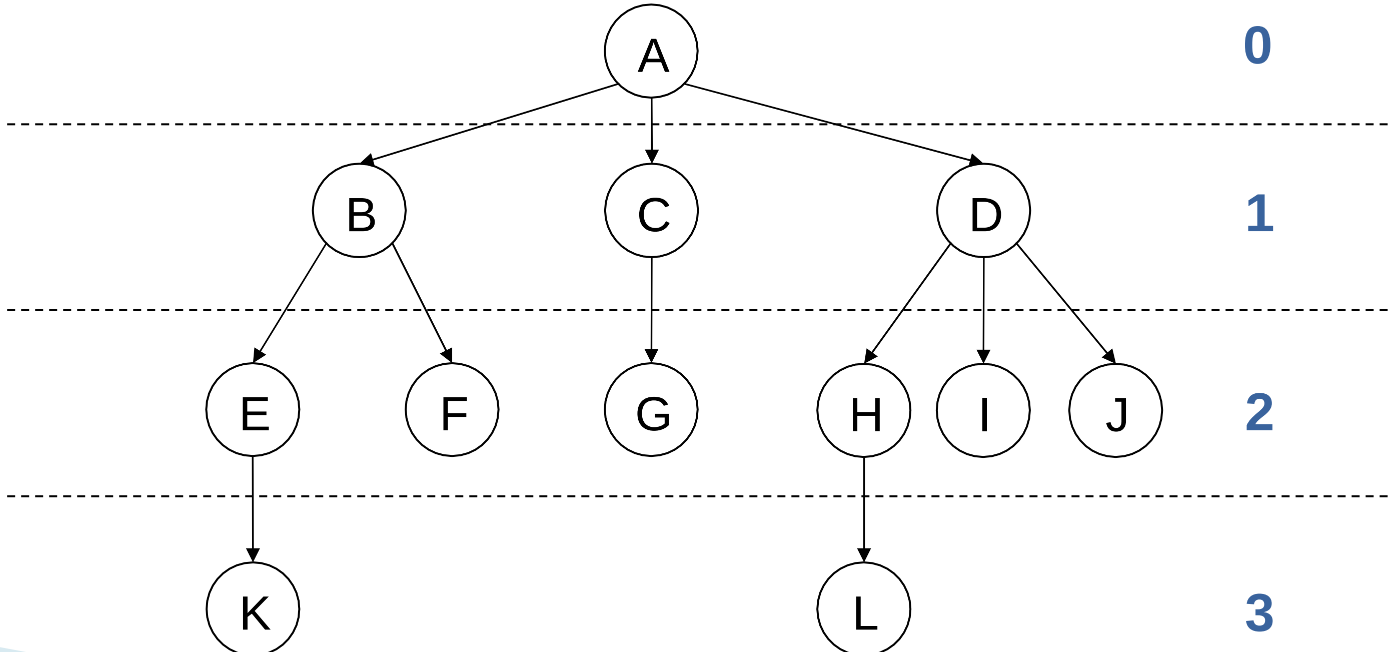


1. Introdução

- **Nível**
  - O **nível** (ou **profundidade**) de um nó é definido admitindo-se que a raiz está no nível zero (ou nível um).
  - Estando um nó no *nível i*, seus filhos estarão no **nível i+1.**
  - Não existe um padrão quanto ao nível adotado para a raiz, que determina o nível dos demais nós. Assim, a raiz pode ser admitida como estando:
    - No **nível zero**
    - Alternativamente, no **nível um**
  - No restante desta apresentação, vamos adotar a raiz no nível **zero.**

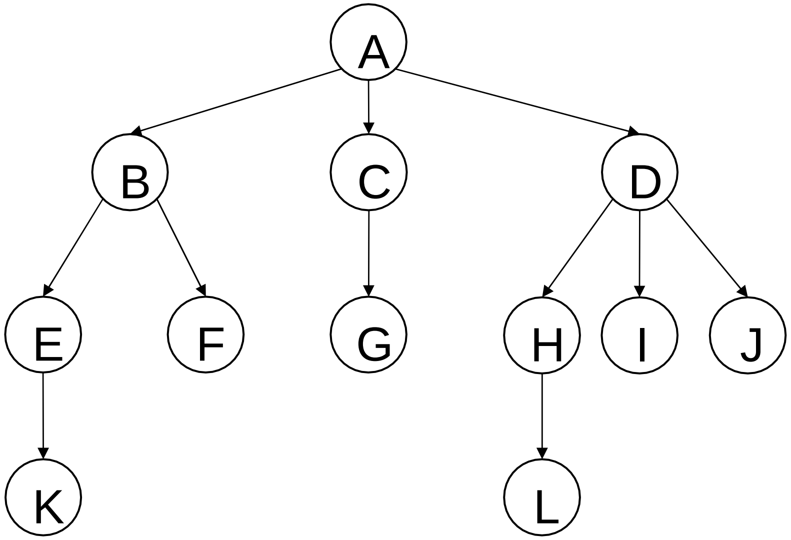
1. Introdução

- **Nível**
  - No exemplo, os nós:
    - B, C e D estão no nível 1
    - K e L estão no nível 3



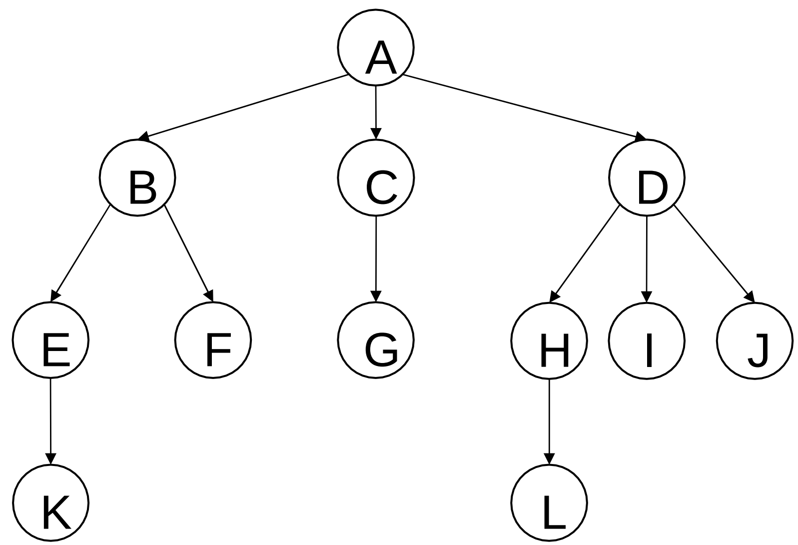
1. Introdução

- **Altura de um Nó**
  - A **altura de um nó** é o número de arestas no maior caminho desde o nó até um de seus descendentes.
  - Portanto, as folhas têm altura zero.
  - No exemplo, os nós:
    - K, F, G, L, I, J têm altura 0
    - E, C e H têm altura 1
    - B e D têm altura 2
    - A tem altura 3



1. Introdução

- **Altura de uma Árvore**
  - A **altura** (ou **profundidade**) de uma árvore é o nível máximo entre todos os nós da árvore ou, equivalentemente, é a altura da raiz.
  - No exemplo, a árvore possui altura 3

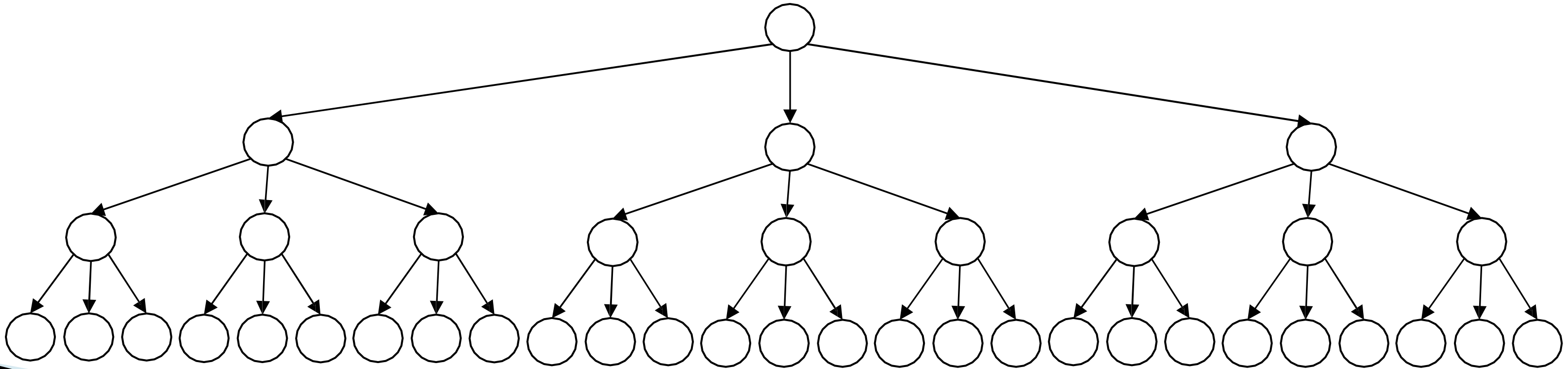


1. Introdução

- **Número Máximo de Nós**
  - O número máximo de nós **n (h,d)** em uma árvore de altura **h** é atingido quando todos os nós possuírem **d** sub-árvores, exceto os de nível **h**, que não possuem sub-árvores.
  - Para uma árvore de grau **d**:
    - Nível **0** contém $\mathbf{d}^0$ (um) nó (raiz)
    - Nível **1** contém $\mathbf{d}^1$ descendentes da raiz
    - Nível **2** contém $\mathbf{d}^2$ descendentes
    - ...
    - Nível **i** contém $\mathbf{d}^i$ descendentes

1. Introdução

- **Número Máximo de Nós**
  - Assumindo **d=3**
    - Nível 0: 1 nó (raiz)
    - Nível 1: 3 nós
    - Nível 2: $3^2 = 9$ nós
    - Nível 3: $3^3 = 27$ nós
  - $n(3,3) = 1 + 3 + 9 + 27 = 40$ nós



1. Introdução

- **Número Máximo de Nós**
  - Portanto, o número máximo de nós $\boldsymbol{n = n(h, d)}$ é soma do número de nós em cada nível, ou seja:

$$n = n(h, d) = \sum_{i=0}^{h} d^i = d^0 + d^1 + d^2 + \cdots + d^h$$

$$\sum_{i=0}^{h} d^i = \frac{d^{h+1} - 1}{d - 1}, d > 1$$

1. Introdução

- **Árvores (Perfeitamente) Balanceadas**

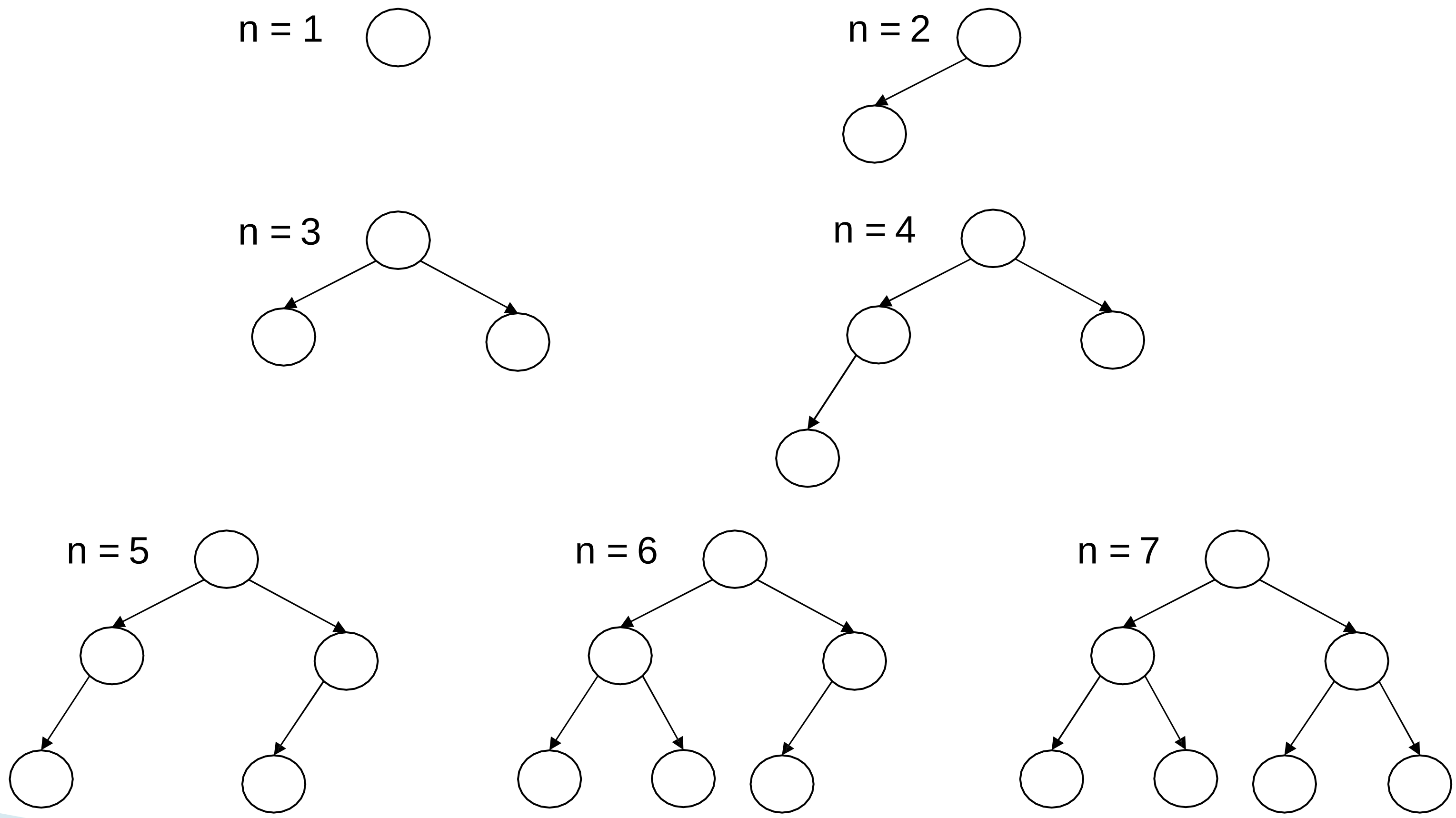
  - Uma árvore é **balanceada** se, para cada nó, a ***altura*** de suas sub-árvores diferem, no máximo, de uma unidade.

  - Uma árvore é **perfeitamente balanceada** se, para cada nó, os ***números de nós*** em suas sub-árvores diferem, no máximo, de uma unidade.

  - Todas as árvores perfeitamente balanceadas também são árvores balanceadas.

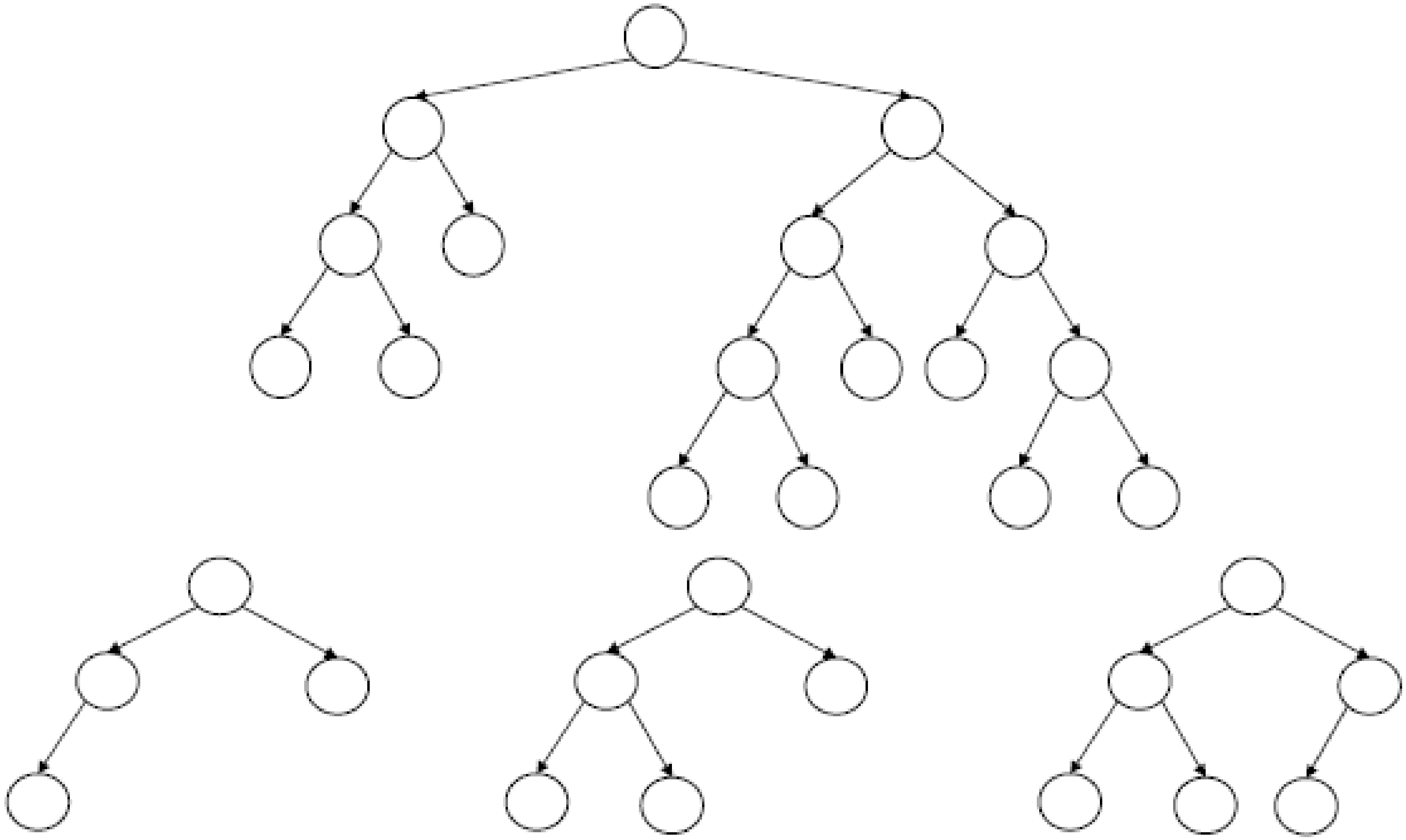
1. Introdução

- **Árvores Perfeitamente Balanceadas de Grau 2**



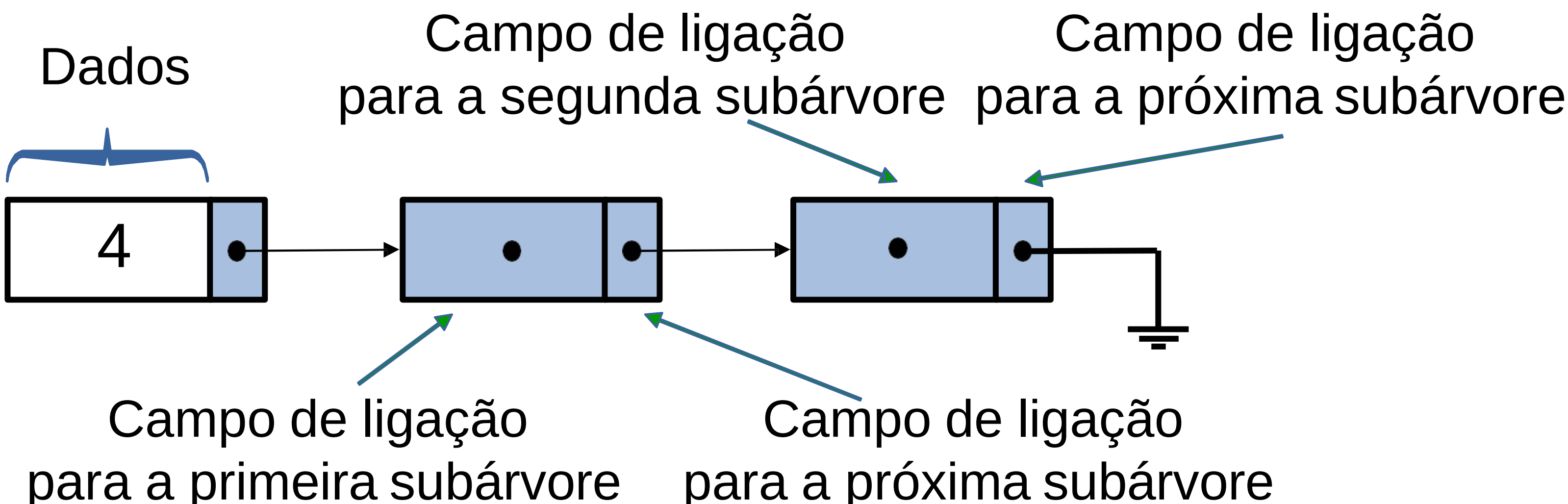
1. Introdução

- **Árvores Balanceadas de Grau 2**



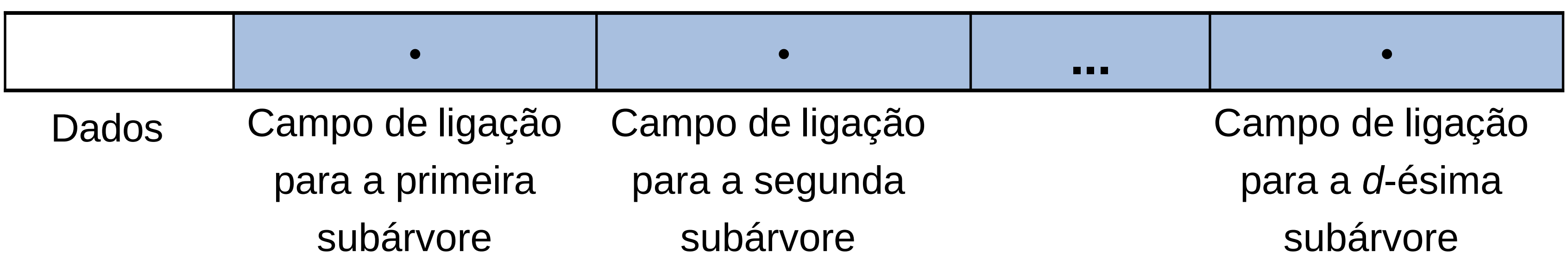
2. Implementação de Árvores

- Árvores podem ser implementadas utilizando estruturas de listas encadeadas.
  - Cada nó possui um campo de informação e uma série de campos de ligação, de acordo como número de filhos daquele nó.



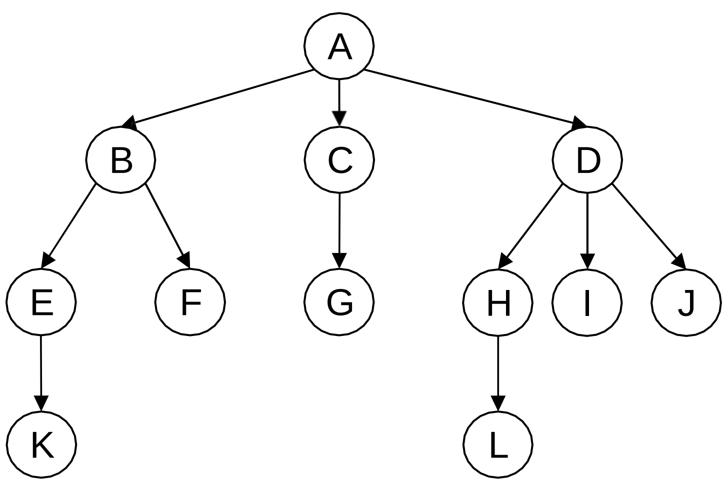
2. Implementação de Árvores

- Entretanto, é mais simples o caso em que cada nó tem um número máximo de filhos **d** pré-estabelecido.

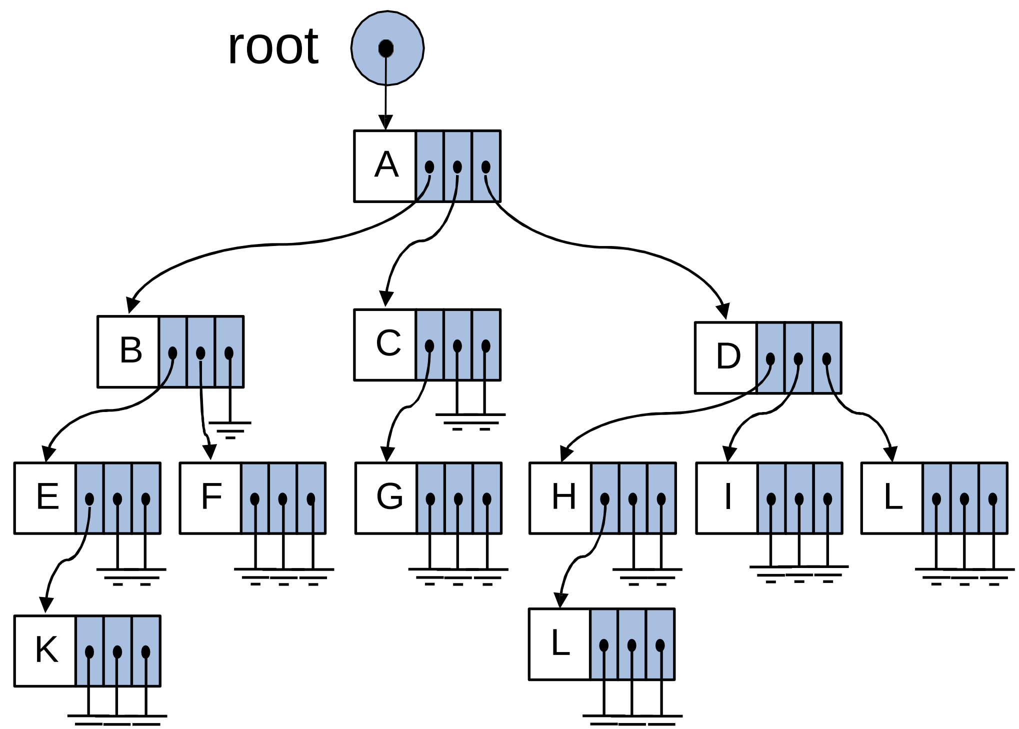


2. Implementação de Árvores

- Por exemplo, a árvore ternária seguinte (d=3)...



- ... pode ser implementada como

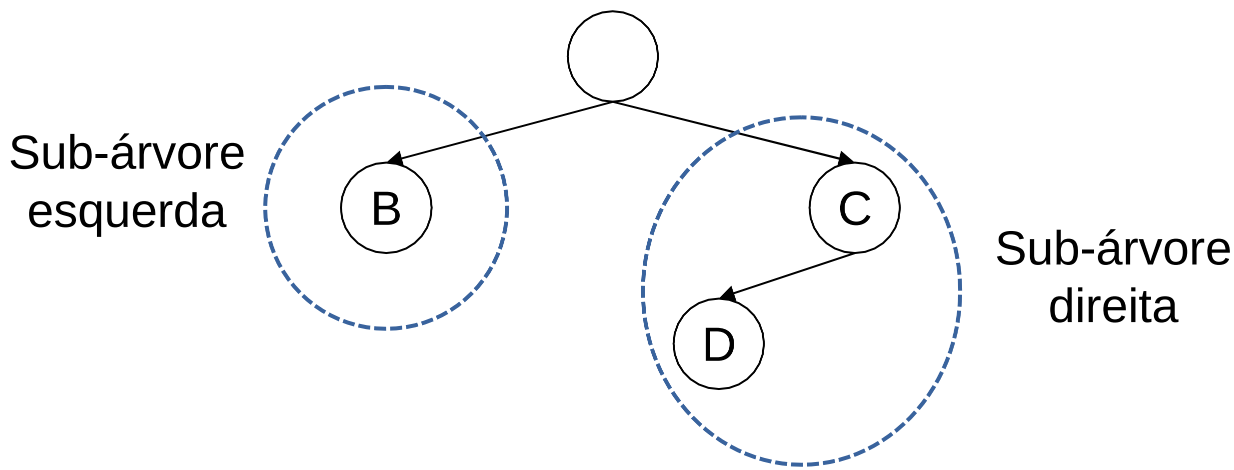


3. Árvores Binárias (AB)

- **Árvores binárias** são árvores de grau 2.

- Uma árvore binária é uma estrutura que é ou vazia ou possui 3 componentes:
  - Uma raiz.
  - Uma sub-árvore esquerda.
  - Uma sub-árvore direita.

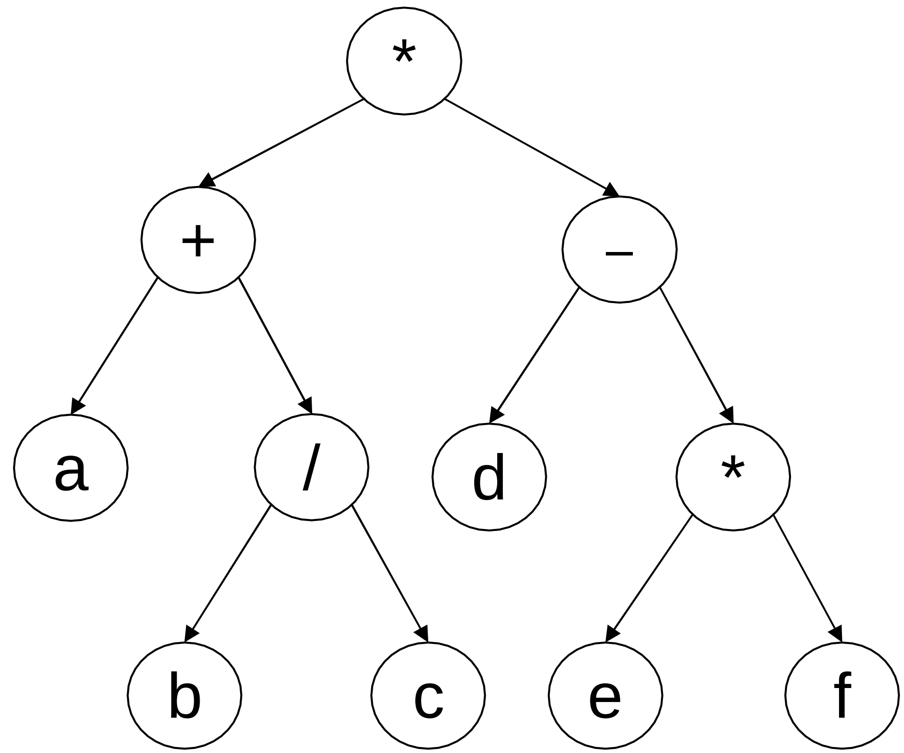
- As sub-árvores devem ser árvores binárias.



3. Árvores Binárias (AB)

- Podemos, por exemplo, representar uma expressão aritmética (com operadores binários) por meio de uma AB, na qual cada operador é um nó da árvore e seus dois operandos representados como sub-árvores.

- A árvore ao lado representa a expressão (a+b/c)*(d–e*f).



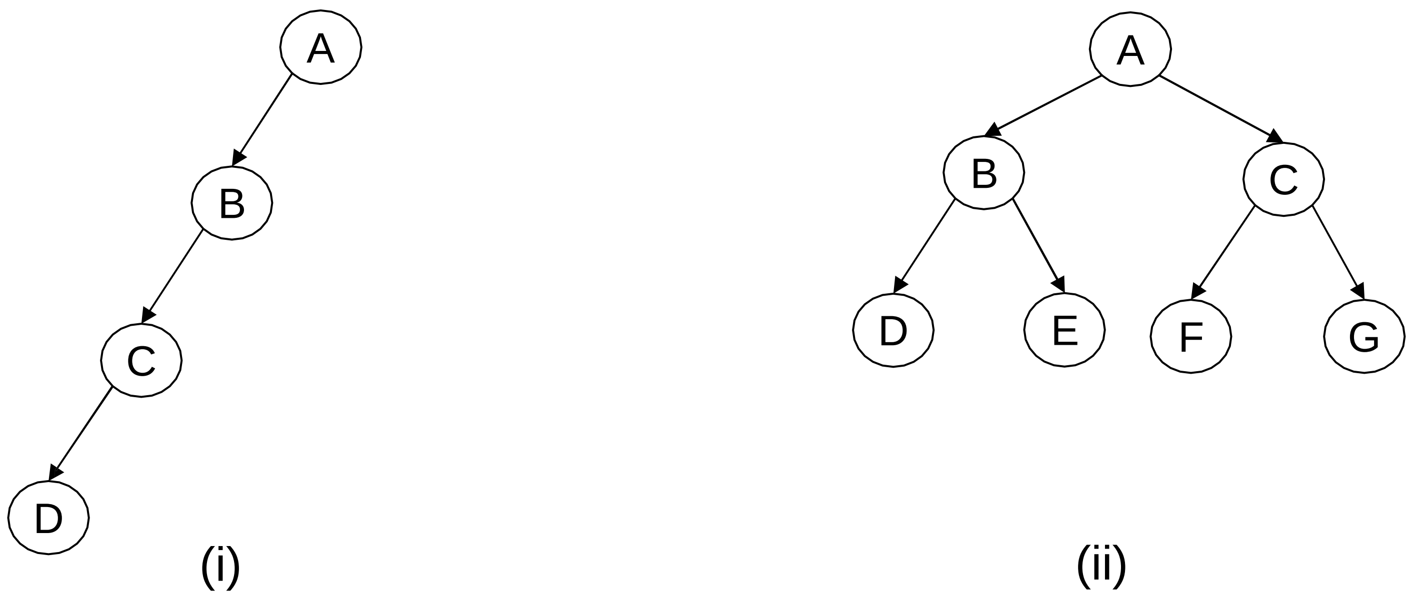
3. Árvores Binárias (AB)

- As duas AB seguintes são distintas:
  - (i) a primeira tem sub-árvore direita vazia.
  - (ii) a segunda tem sub-árvore esquerda vazia.

(i) (ii)

3. Árvores Binárias (AB)

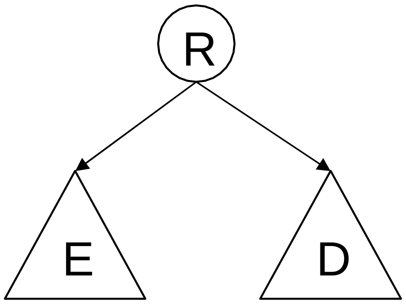
- Exemplos de AB:
  - (i) assimétrica à esquerda (degenerada)
  - (ii) completa



4. Percurso em Árvores Binárias

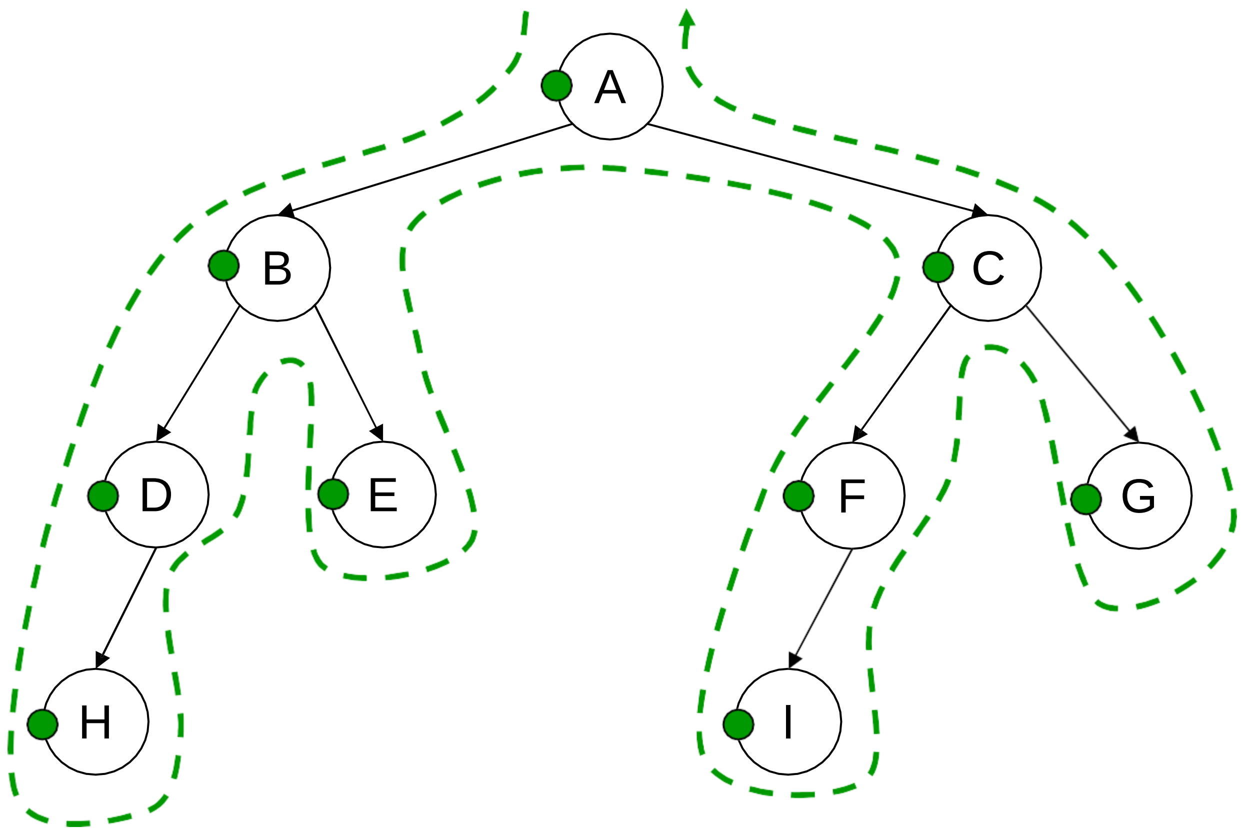
- Seja uma AB onde **R** denota sua raiz, **E** e **D** denotam as sub-árvores esquerda e direita, respectivamente.

- Os nós de uma AB podem ser visitados de três formas (varredura da árvore ou caminhamento):
  - **Pré-ordem** (pre-order): **R, E, D.**
    - Visitar a raiz antes das sub-árvores.
  - **Em-ordem** ou **central** (in-order): **E, R, D.**
  - **Pós-ordem** (post-order): **E, D, R.**
    - Visitar a raiz após visitar as sub-árvores.



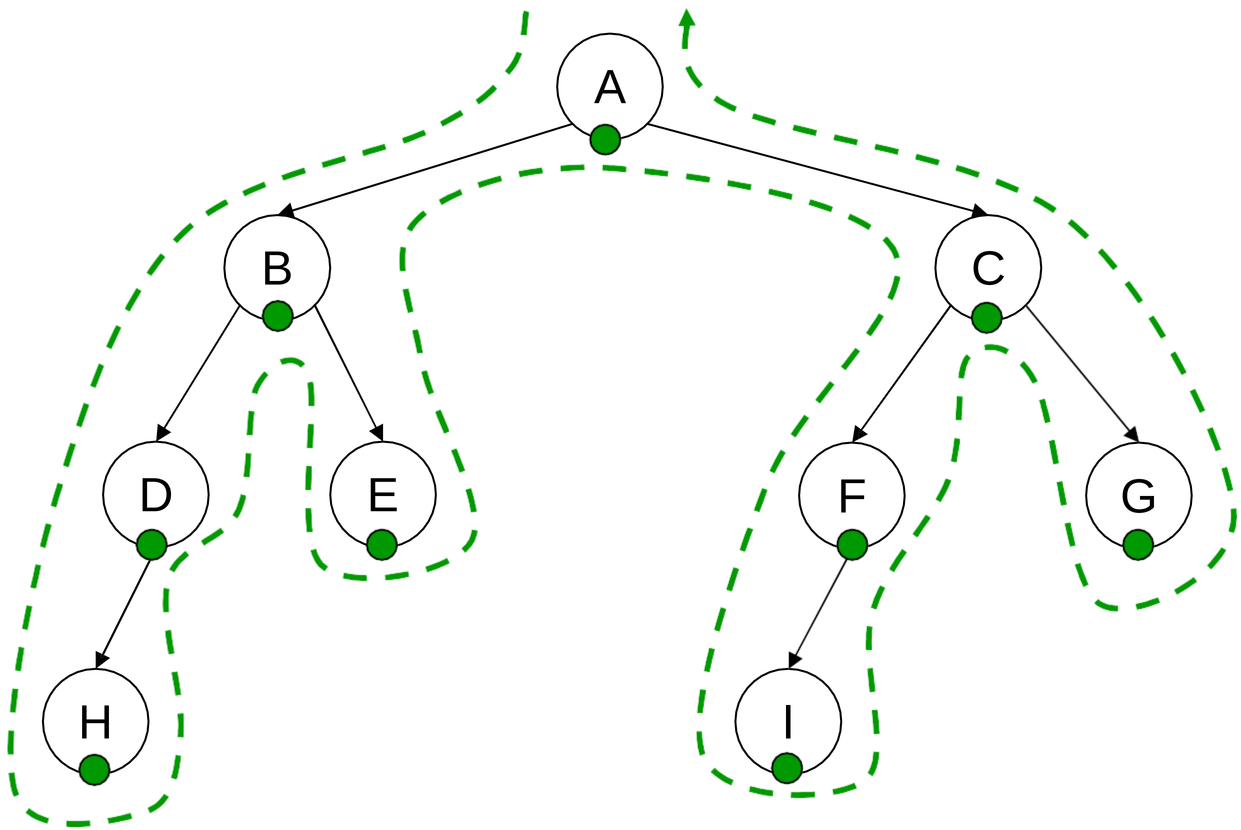
4. Percurso em Árvores Binárias

- **Pré-ordem:** A, B, D, H, E, C, F, I, G



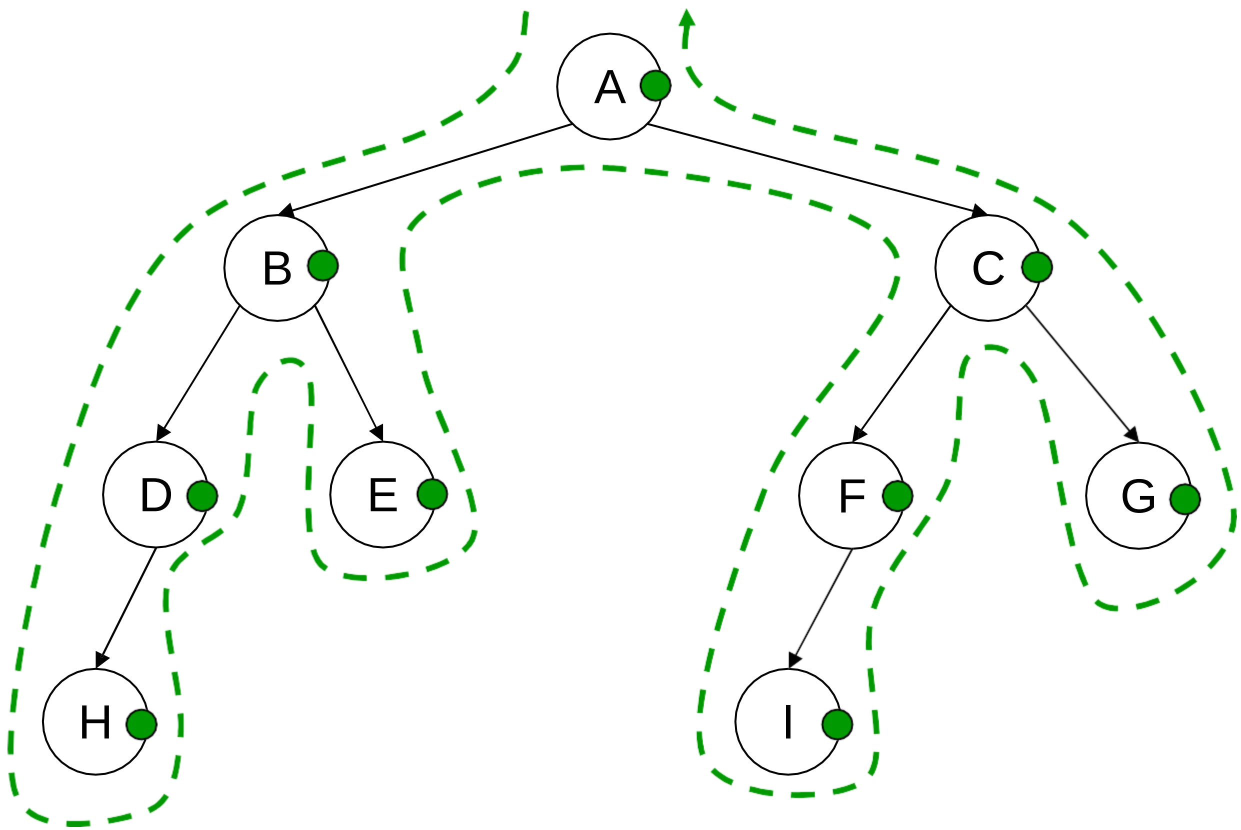
4. Percurso em Árvores Binárias

- **Em-ordem** ou **Cental:** H, D, B, E, A, I, F, C, G



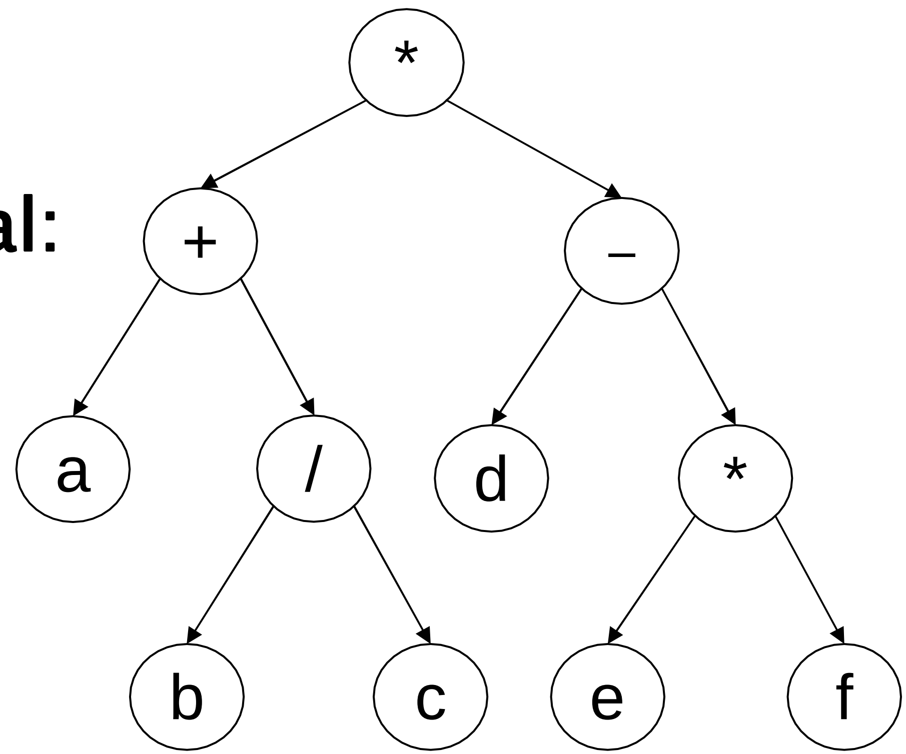
4. Percurso em Árvores Binárias

- **Pós-ordem:** H, D, E, B, I, F, G, C, A



4. Percurso em Árvores Binárias

- **Pré-ordem:**
  - * + a / b c – d * e f
- **Em-ordem** ou **Central:**
  - a + b / c * d – e * f
- **Pós-Ordem:**
  - a b c / + d e f * – *



5. Implementação de AB

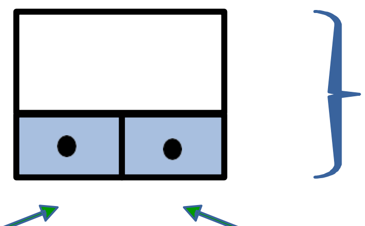
- As Árvores Binárias podem ser implementadas através de **ponteiros**.

- Como toda árvore possui uma raiz (root), uma árvore vazia pode ser representada por um ponteiro com valor **`NULL`**.

- Cada nó em uma árvore binária possui:
  - Um campo de dados
  - Um ponteiro para a sub-árvore esquerda
  - Um ponteiro para a sub-árvore direita

5. Implementação de AB

Campo de dados
(Entry)



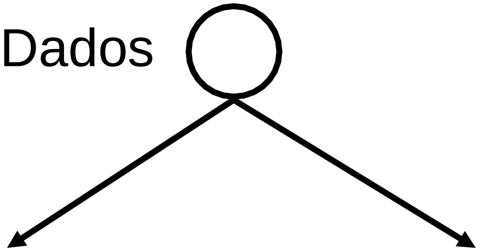
Nó
(TreeNode)

Ponteiro
para subárvore esquerda
(LeftNode)

Ponteiro
para subárvore direita
(RightNode)

Esquematicamente:

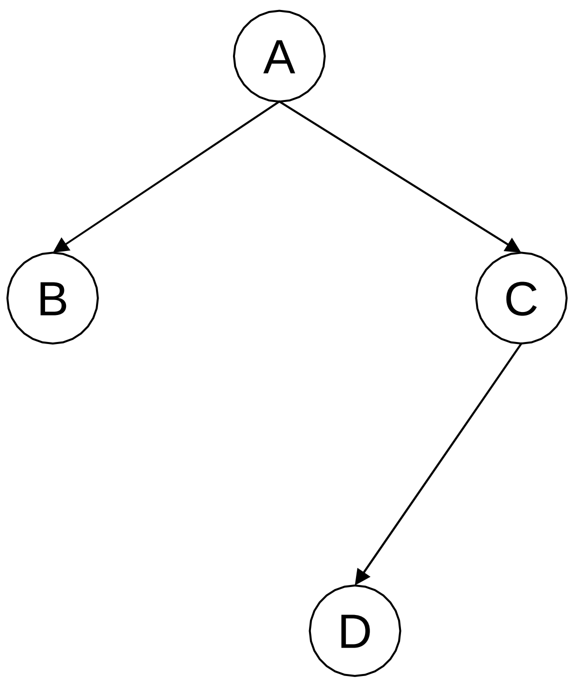
Dados

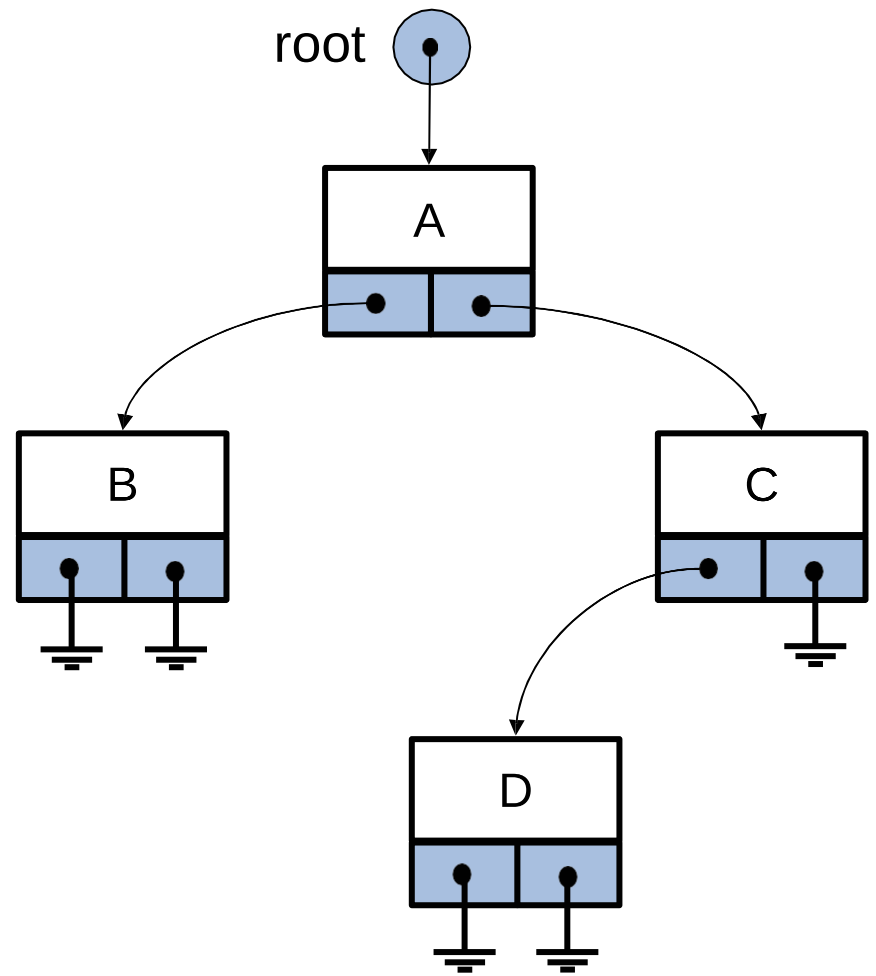


Ponteiro
para subárvore esquerda

Ponteiro
para subárvore direita

5. Implementação de AB





5. Implementação de AB

- **Exemplo:**
  - Criar um TAD **`TArvoreBin`** em C cuja informação seja um inteiro e possua ponteiros para duas sub-árvores: esquerda e direita.
  - Escrever funções que recebam um ponteiro para a raiz da árvore e façam:
    - o caminhamento pré-ordem
    - o caminhamento pós-ordem
    - o caminhamento central

5. Implementação de AB

- TAD **TArvoreBin**

```c
typedef int TItem;

typedef struct TArvoreBin {
  TItem item;
  struct TArvoreBin* pEsq;
  struct TArvoreBin* pDir;
}TArvore;
```

5. Implementação de AB

- **Caminhamento: PreOrder**

```
void PreOrderRec (TArvore* pRaiz) {

  if(pRaiz != NULL) {
    printf (“%d “, pRaiz->item);
    PreOrderRec (pRaiz->pEsq);
    PreOrderRec (pRaiz->pDir);
  }

}
```

5. Implementação de AB

- **Caminhamento: InOrder**

```
void InOrderRec (TArvore* pRaiz) {

  if (pRaiz != NULL) {
    InOrderRec (pRaiz->pEsq);
    printf (“%d “, pRaiz->item);
    InOrderRec (pRaiz->pDir);
  }

}
```

5. Implementação de AB

- **Caminhamento: PostOrder**

```
void PosOrderRec (TArvore* pRaiz) {

  if (pRaiz != NULL) {
    PosOrderRec (pRaiz->pEsq);
    PosOrderRec (pRaiz->pDir);
    printf (“%d “, pRaiz->item);
  }

}
```

6. Referências

- Material de aula dos Profs. Luiz Chaimowicz e Raquel O. Prates, da UFMG: https://homepages.dcc.ufmg.br/~glpappa/aeds2/AEDS2.1%20Conceitos%20Basicos%20TAD.pdf

- Horowitz, E. & Sahni, S.; Fundamentos de Estruturas de Dados, Editora Campus, 1984.

- Wirth, N.; Algoritmos e Estruturas de Dados, Prentice/Hall do Brasil, 1989.

- Material de aula do Prof. José Augusto Baranauskas, da USP: https://dcm.ffclrp.usp.br/~augusto/teaching.htm