Estrutura de Dados

Hamilton José Brumatto

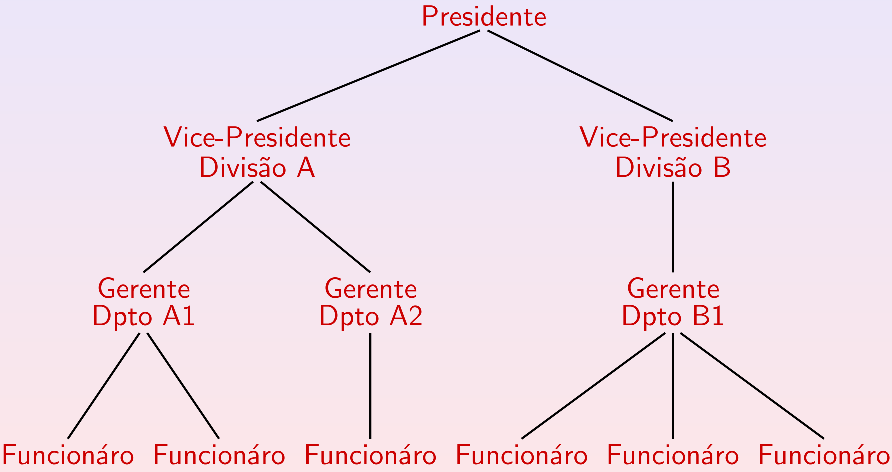
Bacharelado em Ciências da Computação - UESC

26 de maio de 2016

Árvores enraizadas: Árvores Binárias

Árvores enraizadas - Conceitos básicos

- Estrutura complexa que representa uma hierarquia.



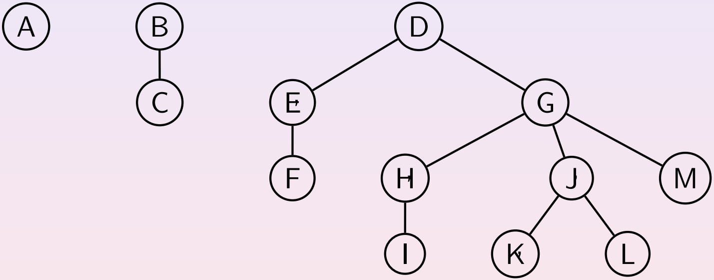
Representação Matemática

- Uma árvore T é um conjunto finito, não vazio de nós.
- $T = \{r\} \cup T_1 \cup T_2 \cup \ldots \cup T_n$, com as propriedades:
    - Um nó especial da árvore, r, é chamado de raiz da árvore; e
    - O restante dos nós é particionado em $n \geqslant 0$ subconjuntos, T_1, T_2, ..., T_n, cada um dos quais sendo uma árvore.

Resumindo, representa-se a árvore como:

$$T = \{r, T_1, T_2, \ldots, T_n\}$$

Representação Gráfica



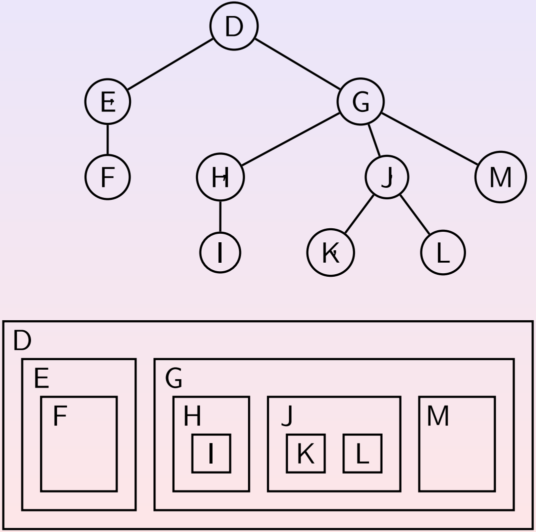
Terminologia

- Nó um elemento da árvore, pode ser a raiz (no topo), uma folha (terminadores) ou um ramo (intermediário)
- Grau - (de um nó) é o número de subárvores relacionadas com aquele nó (esta terminologia é diferente se a árvore não for enraizada)
- Folha - Um nó de grau 0.
- Filho/Neto - Nó raiz de uma subárvore, com relação à árvore que pertence.
- Raiz - Não é filho, origem das árvores.
- Irmãs - Raízes distintas de subárvores de uma mesma árvores
- Caminho - Sequência não vazia de nós.
- Comprimento - Quantos nós foram passados (com excessão do primeiro) em uma sequência.

Terminologia

- Altura - É o comprimento do caminho mais longo até uma folha.
- Profundidade - (de um nó) Comprimento do raiz da árvore ao nó.
- Altura da árvore - É a autura do raiz.
- Ancestral - (de um nó) Um nó de menor profundidade, em relação a este, desde que esteja no caminho do comprimento da profundidade.
- Descendente - Um nó de uma profundidade maior, em relação a este. Sendo este parte do caminho para o comprimento da profundidade.

Representações



Representações

```
class D {
    class E {
        class F {
        }
    }
    class G {
        class H {
            class I {
            }
        }
        class J {
            class K {
            }
            class L {
            }
        }
        class M {
        }
    }
}
```

Implementação de uma árvore

- São várias formas possíveis, estática, dinâmica, ...
- Algumas funções que são importantes:
  - void iniciar(arvore *a) → inicia a estrutura interna da árvore, atribuindo info.
  - obj_t info(arvore *a) → retorna o conteúdo de informação.
  - int altura(arvore *a) → retorna a altura da árvore.
  - int profundidade(arvore *a) → retorna a profundidade.
  - arvore *pai(arvore *a) → retorna quem é o pai da árvore.
  - int filhos(arvore *a) → retorna quantos filhos tem a árvore.
  - arvore *filho(arvore *a, int i) → retorna um filho, por índice.
  - int insereFilho(arvore *a, arvore *f) → insere o filho e retorna o índice.
  - void removeFilho(arvore *a, int i) → remove o filho na posição i.
- A implementação está no código que faz parte desta aula.

Árvores N-árias

- Uma árvore N-ária T é definida como:
  - O conjunto (árvore) é vazio, $T = \emptyset$ ou
  - O conjunto consiste de uma raíz R, e exatamente N árvores N-árias distintas:

$$T = \{R, T_1, T_2, \ldots, T_N\}$$

- Exemplo de árvores 3-árias (ternárias):

$$T_a = \{A, \emptyset, \emptyset, \emptyset\},$$

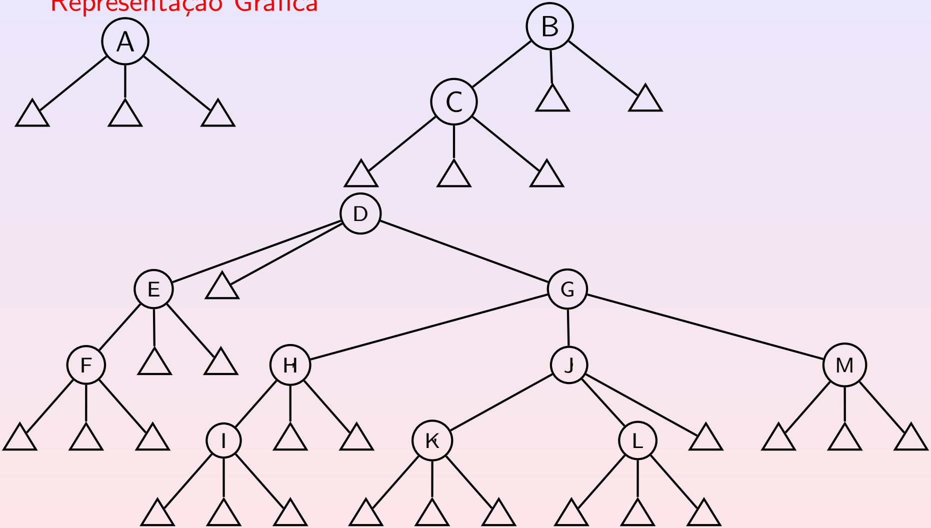
$$T_b = \{B, \{C, \emptyset, \emptyset, \emptyset\}, \emptyset, \emptyset\}$$

$$T_d = \{D, \{E, \{F, \emptyset, \emptyset, \emptyset\}, \emptyset, \emptyset\}, \{G, \{H, \{I, \emptyset, \emptyset, \emptyset\}, \emptyset, \emptyset\}, \{J, \{K, \emptyset, \emptyset, \emptyset\}, \{L, \emptyset, \emptyset, \emptyset\}, \emptyset\}, \{M, \emptyset, \emptyset, \emptyset\}\}, \emptyset\}$$

Árvores N-árias

- As árvores vazias são chamadas de nós externos
- As árvores não vazias são chamadas de nós internos.
- Folhas são nós internos que somente possuem subárvores que são nós externos.
- Uma árvore N-ária com $n \geqslant 0$ nós internos possui $(N-1)n+1$ nós externos.
- A altura de um nó externo é -1.
- A altura de uma folha é 0.

Representação Gráfica



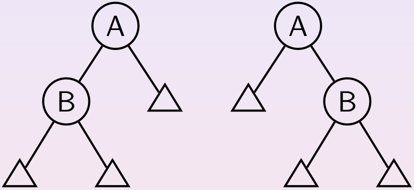
Implementações de árvore N-ária

- Utiliza as mesmas rotinas implementadas na árvore geral.
- A lista de filhos é um vetor.
- As árvores vazias são representadas como NULL na lista de filhos.
- A ordem importa, filhos $\{T_1, T_2, \emptyset\} \neq \{T_1, \emptyset, T_2\}$.
- Cada árvore filha tem sua posição fixada por um índice do vetor.
- As implementações estão em códigos que acompanham esta aula.

Árvores Binárias

- O conjunto é vazio, $T = \emptyset$; ou
- O conjunto consiste em uma raiz, R, e em exatamente duas árvores binárias distintas T_e e T_d, $T = \{R, T_e, T_d\}$
- A árvore T_e é dita subárvore esquerda de T, e a árvore T_d é dita a subárvore direita de T.

Exemplo de Árvores Binárias

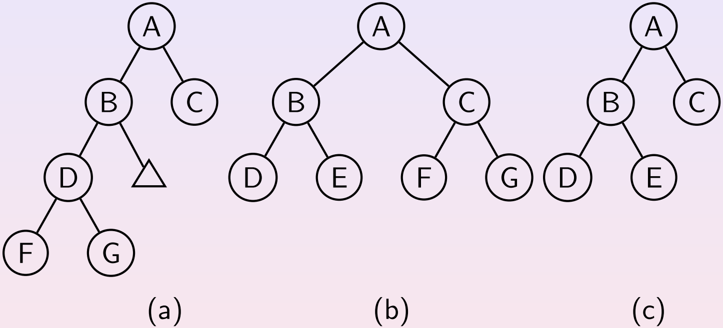


Árvores binárias distintas, a primeira possui a subárvore esquerda, como nó interno, e a segunda a subárvore direita: $\{A, \{B, \emptyset, \emptyset\}, \emptyset\}$ e $\{A, \emptyset, \{B, \emptyset, \emptyset\}\}$

Definições sobre árvores binárias

- Uma árvore binária completa de profundidade d é a árvore estritamente binária onde todas as folhas estejam no nível d.
- Uma árvore binária completa de profundidade d possui 2^d folhas e $2^d - 1$ nós não folhas.
- Uma árvore binária quase completa de profundidade d ocorre quando:
  - Cada folha na árvore está no nível d ou $d - 1$.
  - Para cada nó n na árvore que contenha um descendente direito folha no nível d, então todos descendentes esquerdos que forem folha também estão no nível d.
- Uma árvore (completa ou quase completa) estritamente binária com n folhas possui no total $2n - 1$ nós.
- Uma árvore quase completa que não seja estritamente binária com n folhas possui no total $2n$ nós
- Existe uma única árvore binária quase completa com n nós, esta árvore será estritamente binária se n for ímpar

Exemplos - omitindo-se quase todas árvores vazias

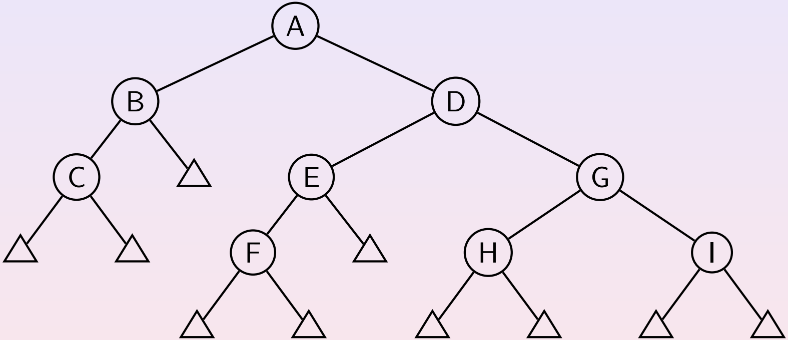


(a) Árvore não estritamente binária
(b) Árvore binária completa
(c) Árvore estritamente binária quase completa

Percurso em Árvores Binárias

- O percurso, tal qual em estruturas lineares, representa uma visita aos nós da árvore para exibir pesquisar seu conteúdo.
- São três percursos utilizados:
  - Percurso ém pré-ordem (ou em profundidade): Faz-se uma pesquisa sobre o conteúdo do raiz, em seguida, recursivamente aplica-se o percurso no filho esquerdo e depois no direito.
  - Percurso em ordem simétrica (ou em ordem): Aplica-se recursivamente o percurso no filho esquerdo, faz-se a pesquisa no raiz, e aplica-se recursivamente o percurso no filho direito.
  - Percurso em pós-ordem: Aplica-se recursivamente primeiro o percurso no filho esquerdo, em seguida no filho direito e então faz-se a pesquisa no raiz.
- O percurso em ordem é o único restrito para árvores binárias, os percursos em pré-ordem e em pós ordem pode-se aplicar em todos tipos de árvores, desde que se defina uma ordem para os filhos.

Exemplo de percurso em Árvores Binárias



Percurso em pré-ordem: ABCDEFGHI
Percurso em ordem: CBAFEDHGI
Percurso em pós-ordem: CBFEHIGDA