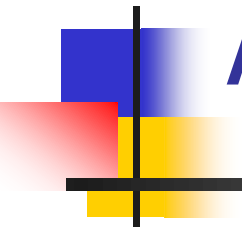
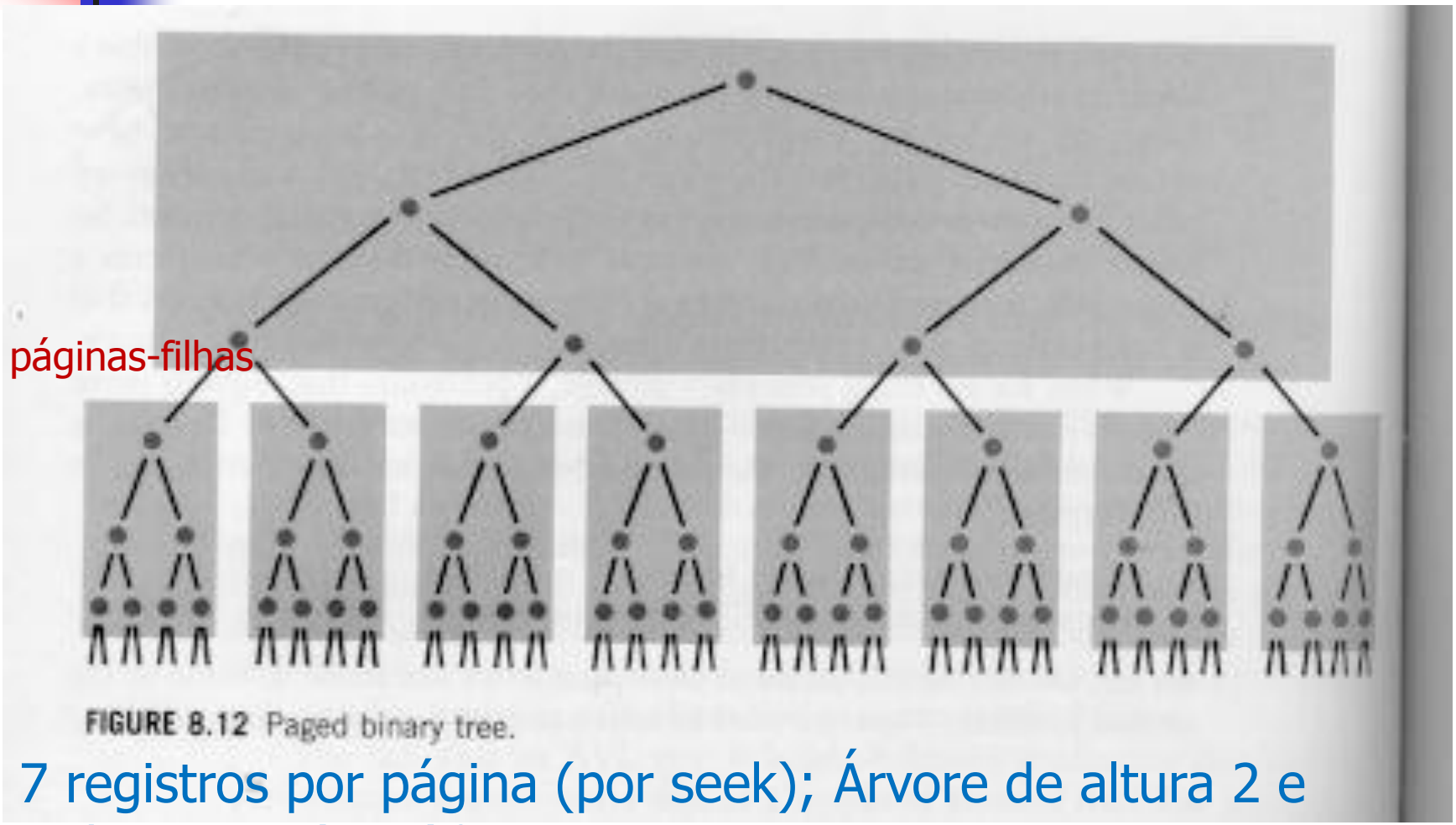


Árvores-B (Parte Ib)

SCC-203 – Algoritmos e Estruturas de Dados II

Graça Nunes

Árvores Binárias Paginadas (*Paged Binary Trees*)

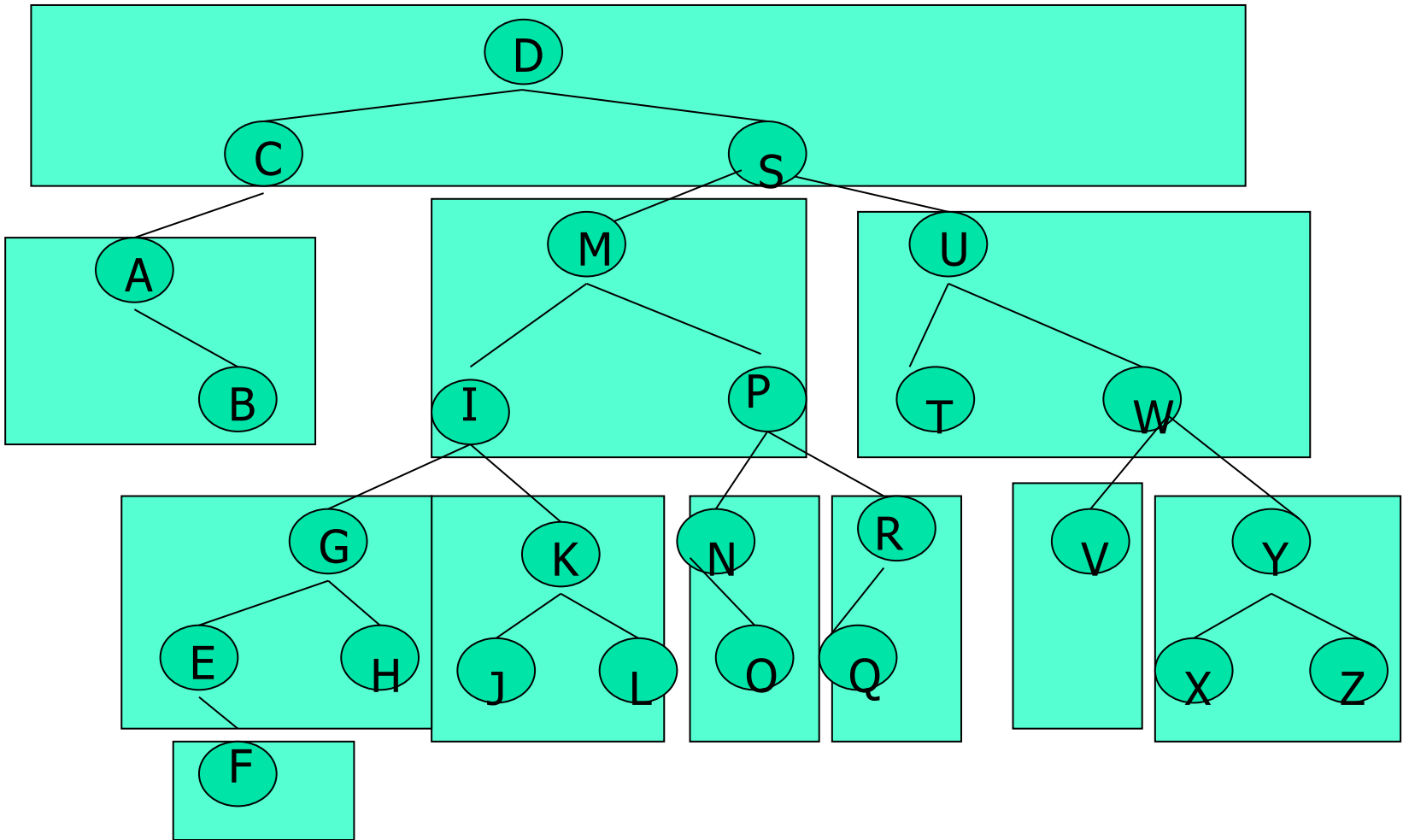


8 páginas-filhas

FIGURE 8.12 Paged binary tree.

7 registros por página (por seek); Árvore de altura 2 e ordem 8 ➔ (8+1)*7 registros, se completa

Exemplo – C S D T A M P I B W N G U R K E H O L J Y Q Z F X V



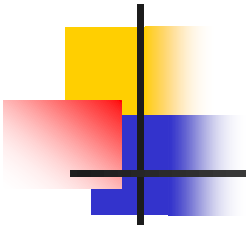
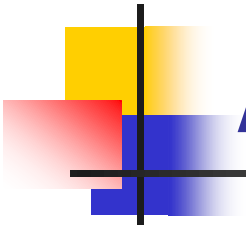


ABB Paginadas: Problemas a resolver

1. Como garantir que as chaves da raiz sejam boas separadoras, tal que dividam o conjunto mais ou menos ao meio?
2. Como evitar agrupamento de certas chaves na página raiz?
3. Como garantir que cada página contenha um certo número mínimo de chaves?

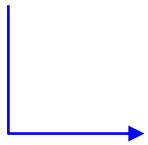


Árvores-B

- **Generalização** da ideia de ABB paginada
  - Não são binárias
  - Conteúdo de uma página não é mantido como uma árvore
  - Sempre balanceada
  - *bottom-up* para a criação (em disco)
    - nós folhas $\rightarrow$ nó raiz

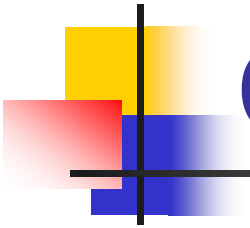
Construção *Bottom-Up*

- Consequências
  - Chaves "erradas" não são mais alocadas no nó raiz



na árvore-B, as chaves na raiz da árvore emergem naturalmente

  - Não é necessário tratar o problema de desbalanceamento

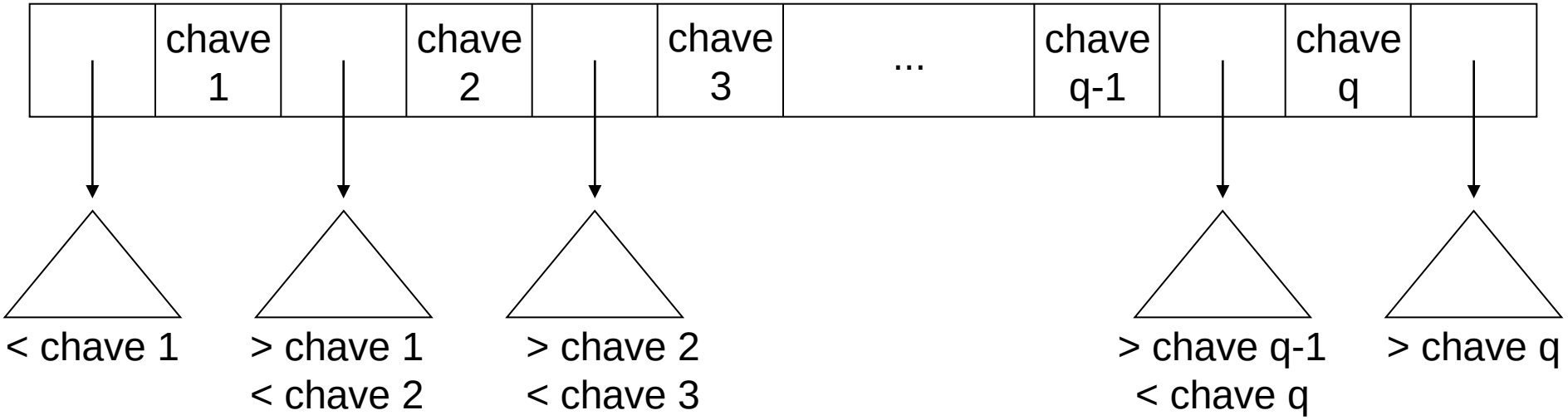


Características

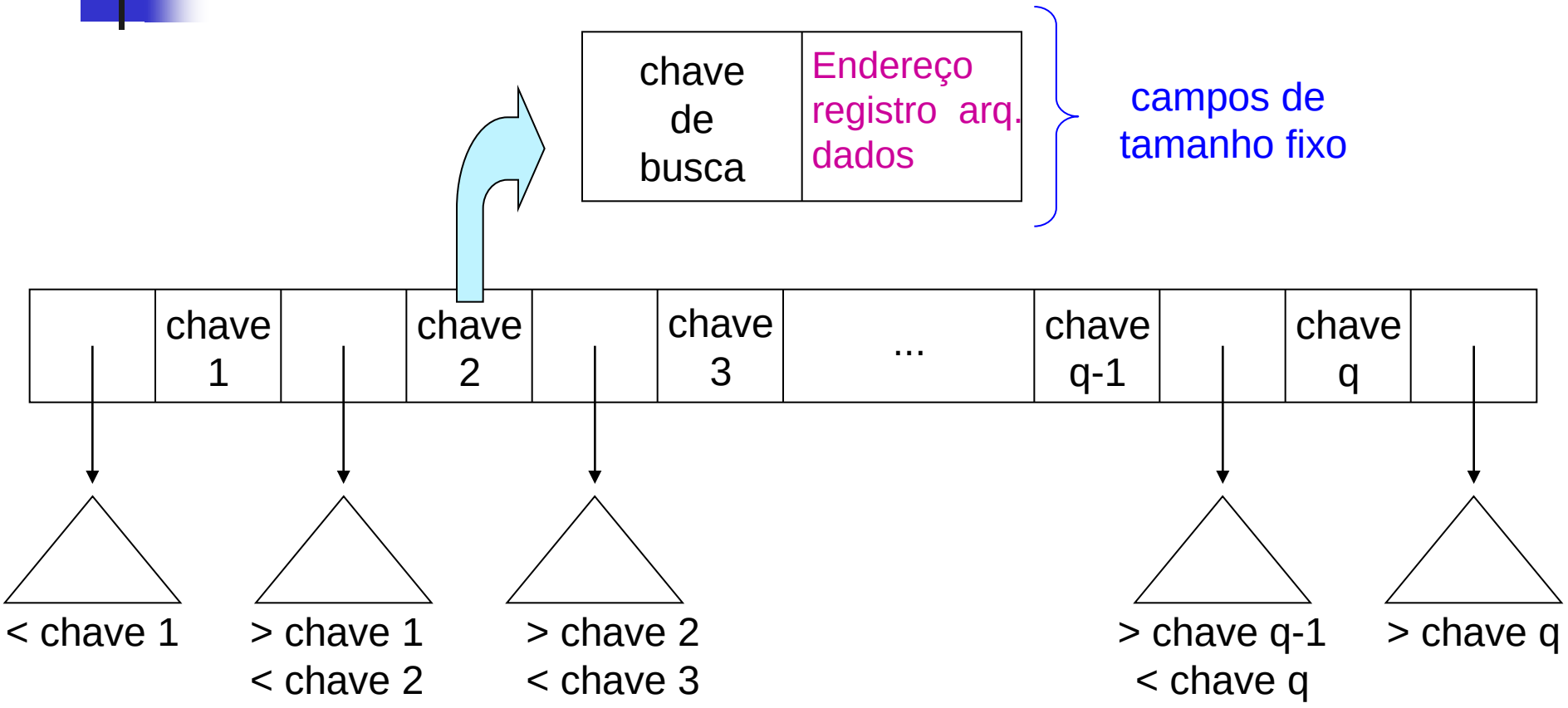
- Características do nó (= página de disco)
  - Sequência ordenada de chaves + Conjunto de ponteiros
    - Número de ponteiros = número de chaves + 1
    - Ordem da árvore: número máximo de ponteiros dos nós
  - Não há uma árvore explícita dentro de uma página (ou nó da árvore)
  - Registros de tamanho fixo para armazenar um nó

Estrutura Lógica de um Nó

Registro de tamanho fixo➔ RRN

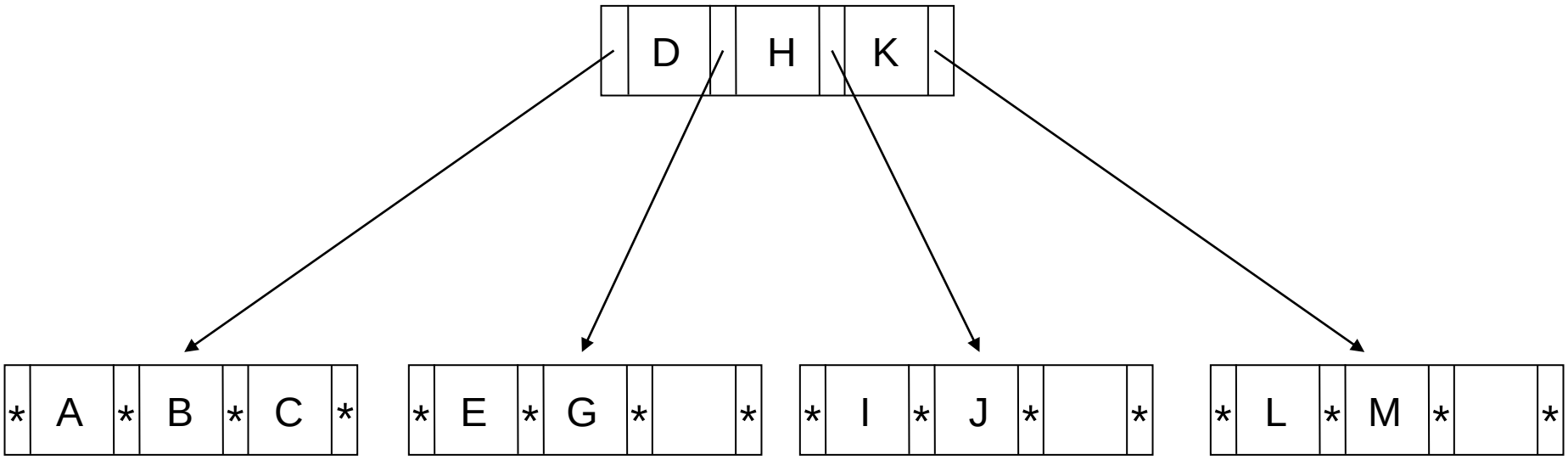


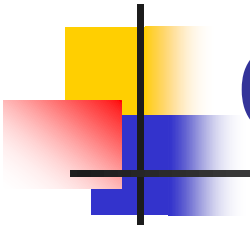
Estrutura Lógica de um Nó



Exemplo

Ordem 4: até 3 chaves por nó; até 4 filhos

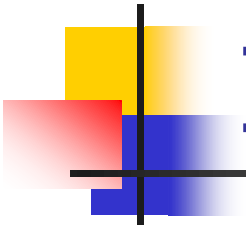




Características

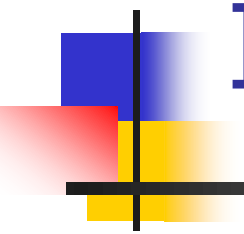
- Ordem
  - Número máximo de ponteiros que pode ser armazenado em um nó
  - Exemplo: árvore-B de ordem 8
    - máximo de 7 chaves e 8 ponteiros

- Observações
  - Número máximo de ponteiros é igual ao número máximo de descendentes de um nó
  - Nós folhas não possuem filhos, e seus ponteiros são nulos

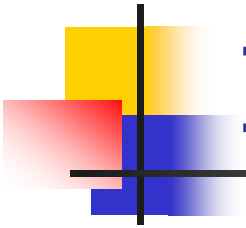


Inserção de Dados (Chave)

- Característica
  - Sempre realizada nos nós folha (a busca binária por uma chave inexistente termina sempre no nó folha)
- Situações a serem analisadas
  1. árvore vazia
  2. overflow no nó raiz
  3. inserção em nós folha



Inserção em árvore vazia



Inserção: situação inicial

- Criação e preenchimento do nó
  - primeira chave: criação do nó raiz
  - demais chaves: inserção até a capacidade limite do nó

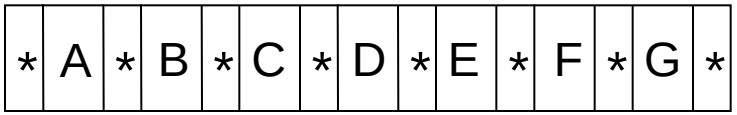
- Exemplo
  - nó com capacidade para 7 chaves ➔ ordem 8
  - chaves: letras do alfabeto
  - situação inicial: árvore vazia

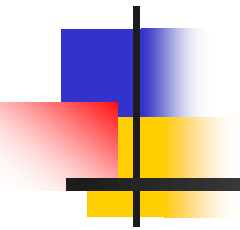
Inserção: situação inicial

- Chaves B C G E F D A
  - inseridas desordenadamente
  - mantidas ordenadas no nó
- Ponteiros (*)
  - nós folhas: -1 ou fim de lista (NIL)
  - nós internos: RRN do nó filho ou -1
- Nó raiz (= nó folha)

Raiz=0

RRN 0



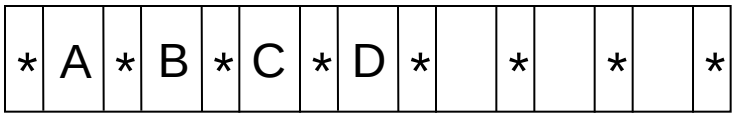


Overflow no nó raiz

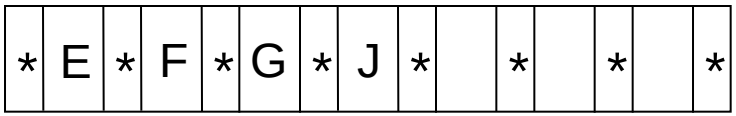
Inserção: *overflow* nó raiz

- Passo 1 – particionamento do nó (*split*)
  - nó original → nó original + novo nó
    - *split* 1-to-2
  - as chaves são distribuídas uniformemente nos dois nós
    - chaves do nó original + nova chave
- Exemplo: inserção de J

0



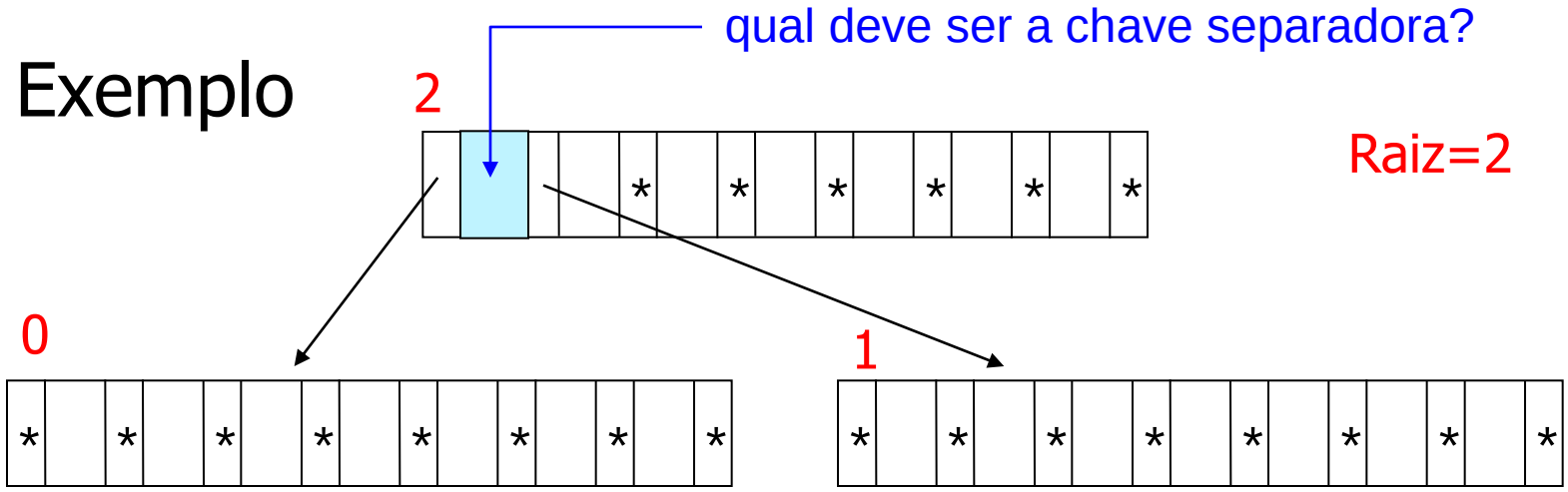
1



Inserção: *overflow* nó raiz

- Passo 2 – criação de uma nova raiz (efeito *bottom-up*); aumenta altura
  - a existência de um nível mais alto na árvore permite a escolha das folhas durante a pesquisa

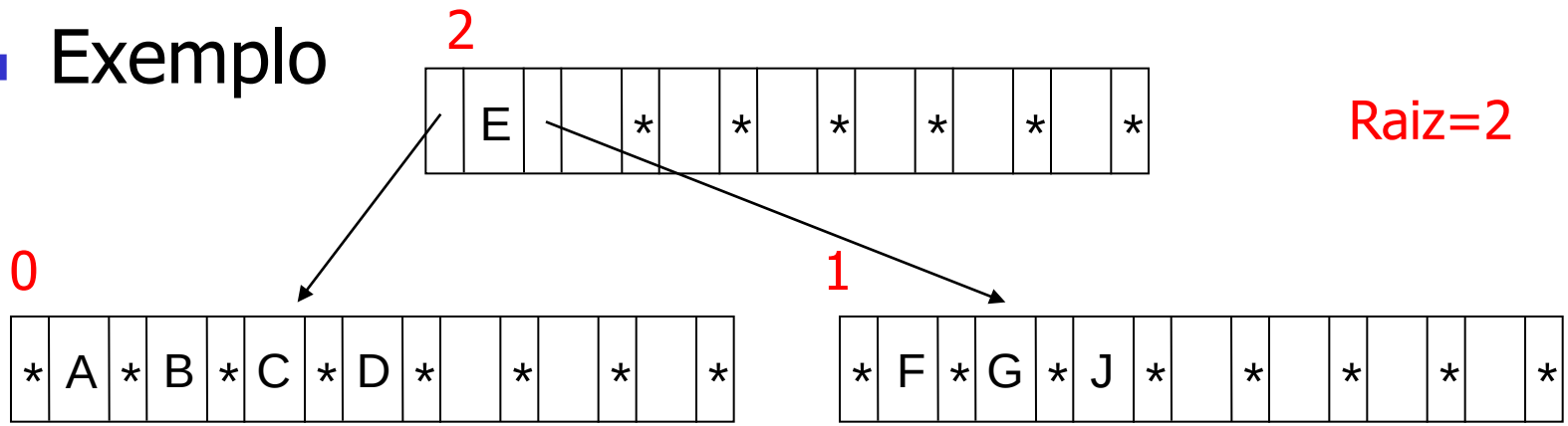
- Exemplo

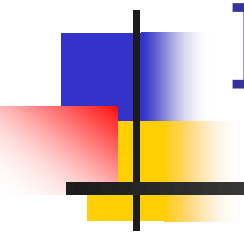


Inserção: *overflow* nó raiz

- Passo 3 – promoção de chave (*promotion*)
  - a primeira chave do novo nó resultante do particionamento é promovida para o nó raiz ➔ é a mediana do conjunto dos dois nós

- Exemplo





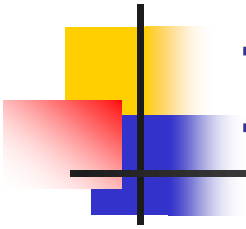
Inserção em nós folha

Inserção: nós folhas

- Passo 1 – pesquisa binária (inserção sempre nas folhas)
  - a árvore é percorrida até encontrar o nó folha no qual a nova chave será inserida

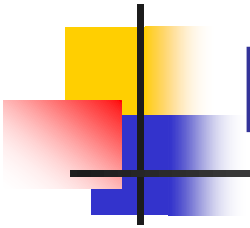
nó folha em memória principal

- Passo 2(a) – inserção em nó com lugar disponível
  - inserção ordenada da chave no nó (sequencial)
  - alteração dos valores dos campos de referência



Inserção: nós folhas

- Passo 2(b) – inserção em nó cheio *(overflow)*
  - Particionamento *(split)*
    - criação de um novo nó
      (nó original $\rightarrow$ nó original + novo nó)
    - distribuição uniforme das chaves nos dois nós
  - Promoção *(promotion)*
    - escolha da primeira chave do novo nó como chave separadora no nó pai
    - ajuste do nó pai para apontar para o novo nó
    - propagação recursiva de *overflow*



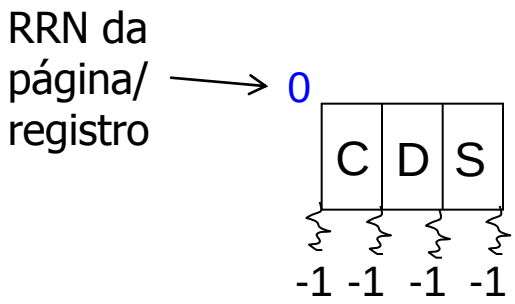
Exemplo

- Insira as seguintes chaves em um índice árvore-B
  - C S D T A M P I B W N G U R K E H O L J Y Q Z F X V

- Ordem da árvore-B: 4
  - em cada nó (página de disco)
    - número de chaves: 3
    - número de ponteiros: 4

C S D T A M P I B W N G U R K ...

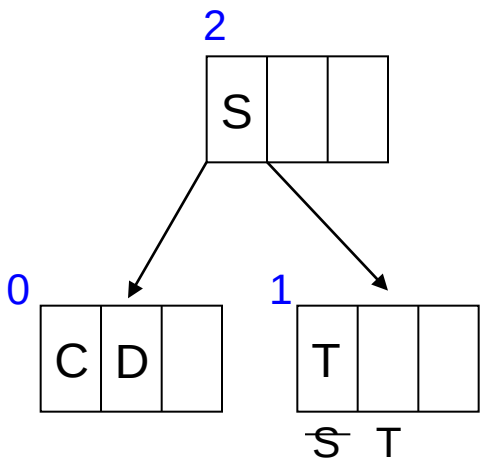
- Passo 1 – inserção de C, S, D
  - criação do nó raiz
    - C
    - C S
    - C D S



C S D T A M P I B W N G U R K ...

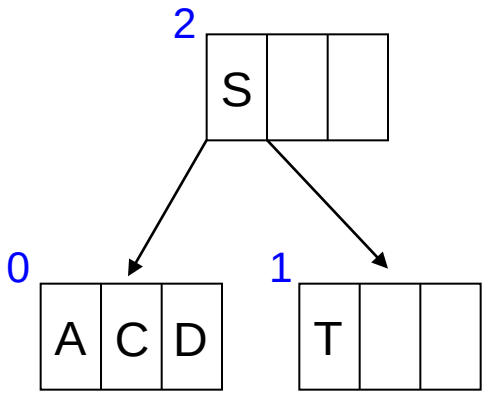
- Passo 2 – inserção de T
  - nó raiz cheio

- particionamento do nó
- criação de uma nova raiz
- promoção de S



C S D T A M P I B W N G U R K ...

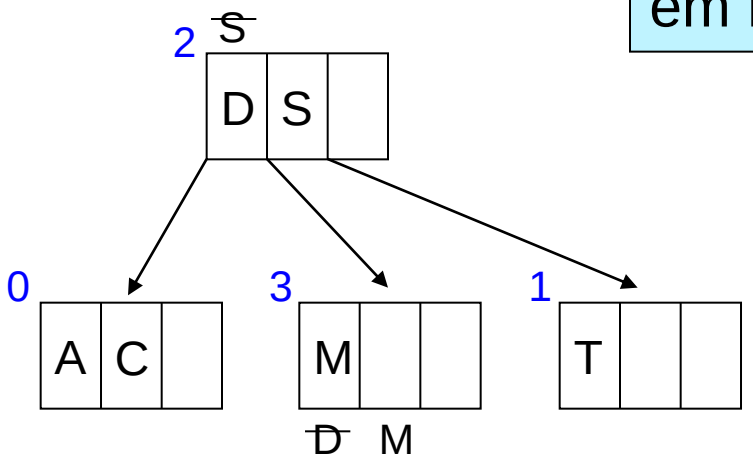
- Passo 3 – inserção de A
  - nó folha com espaço



C S D T A M P I B W N G U R K ...

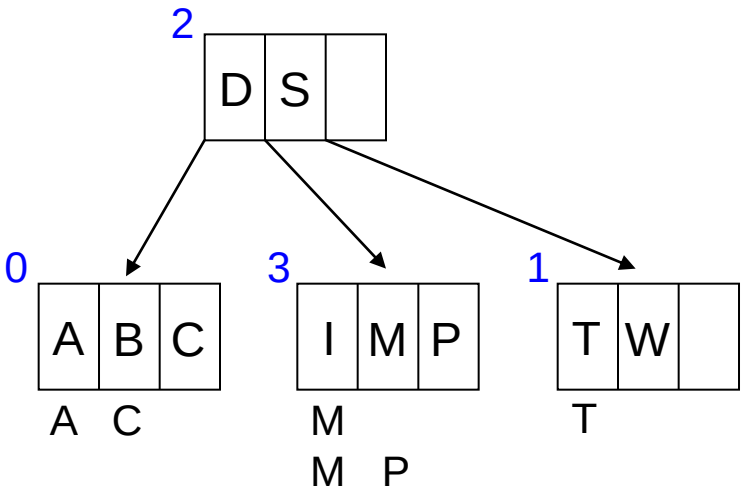
- Passo 4 – inserção de M
  - nó folha 0 cheio

• particionamento do nó
• promoção de D; inserção em nó (raiz) com espaço



C S D T A M P I B W N G U R K ...

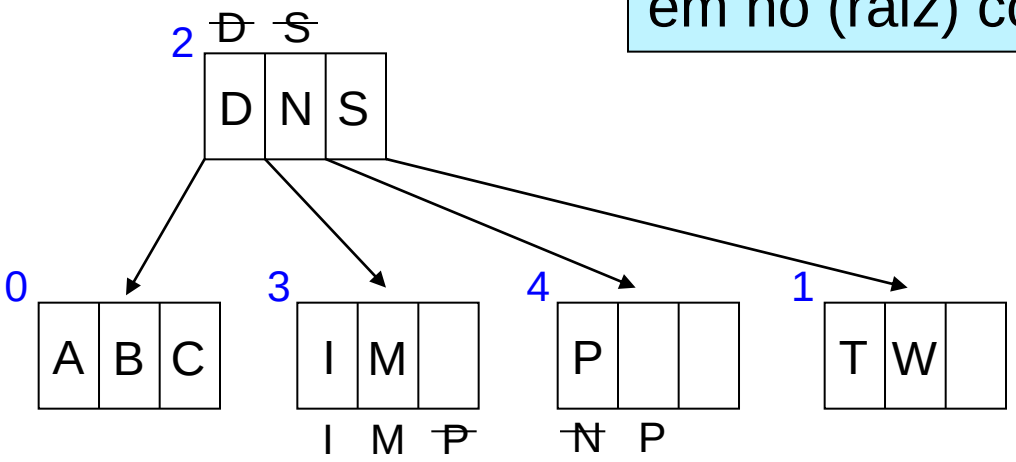
- Passo 5 – inserção de P, I, B, W
  - nós folhas com espaço



C S D T A M P I B W N G U R K ...

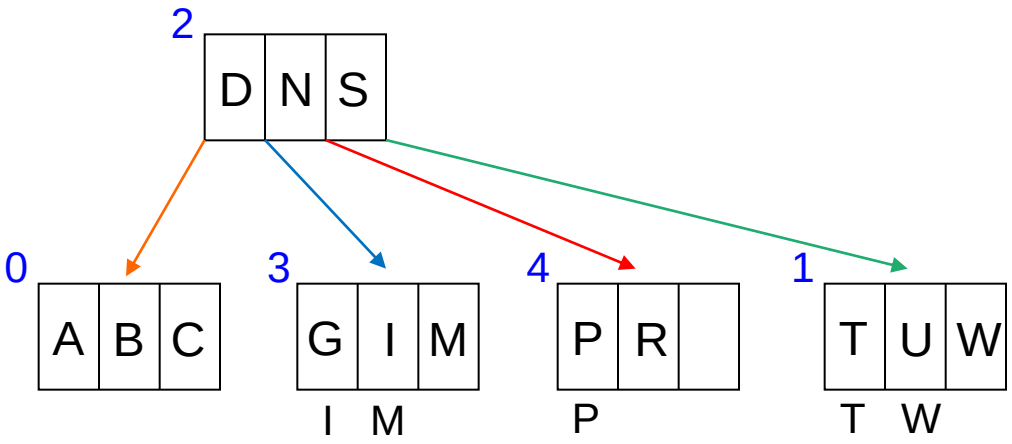
- Passo 6 – inserção de N
  - nó folha 3 cheio

• particionamento do nó
• promoção de N; inserção em nó (raiz) com espaço



C S D T A M P I B W N G U R K ...

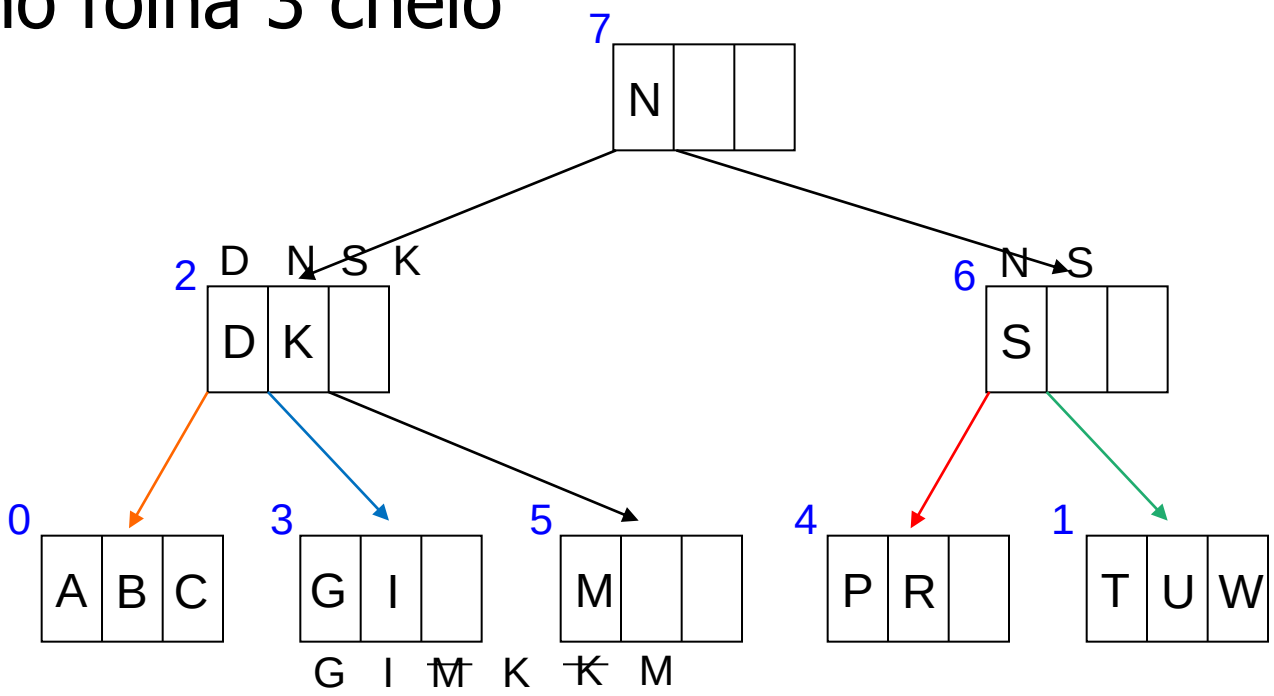
- Passo 7 – inserção de G, U, R
  - nós folhas com espaço



C S D T A M P I B W N G U R K ...

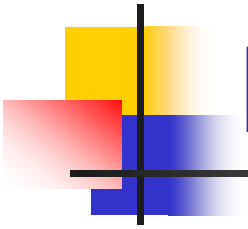
- Passo 8 – inserção de K
  - nó folha 3 cheio

- particionamento do nó 3
- promoção de K
- particionamento do nó 2
- promoção de N



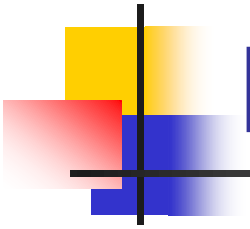
... E H O L J Y Q Z F X V

- Finalizar a construção da árvore



Exercícios

- Na árvore-B do exemplo anterior, insira a chave $, sendo que $ < A

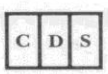


Exercícios

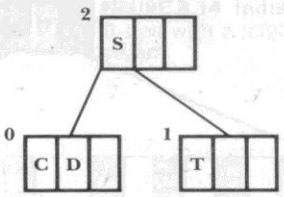
- Insira as seguintes chaves em um índice árvore-B
  - C S D T A M P I B W N G U R K E H O L J Y Q Z F X V
  - diferentemente do exemplo anterior, escolha o último elemento do primeiro nó para promoção durante o particionamento do nó.

Exemplo Inserção: **C S D T A M P I B W N G U R K E H O L J Y Q Z F X V**

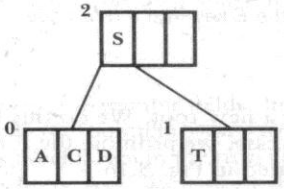
Insertion of *C*, *S*, and *D* into the initial page:



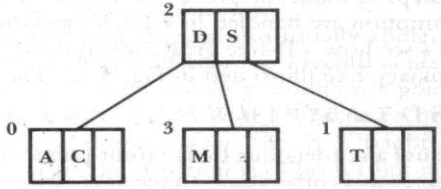
Insertion of *T* forces the split and the promotion of *S*:



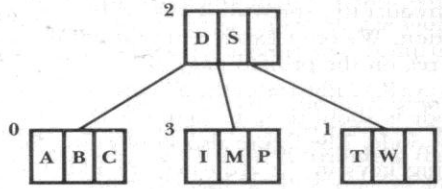
A added without incident:



Insertion of *M* forces another split and the promotion of *D*:



P, *I*, *B*, and *W* inserted into existing pages:



Insertion of *N* causes another split, followed by the promotion of *N*. *G*, *U*, and *R* are added to existing pages:

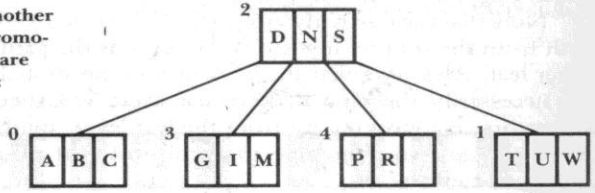
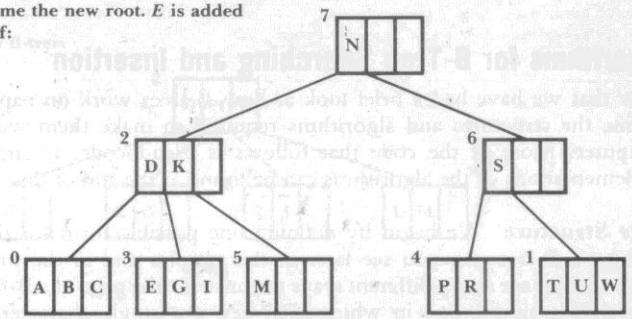
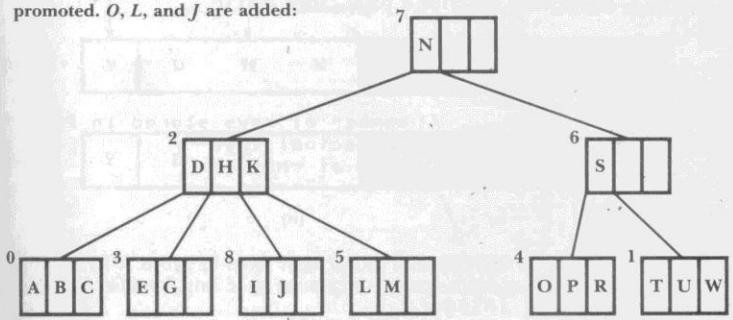


FIGURE 8.17 Growth of a B-tree, part I. The tree grows to a point at which splitting of the root is imminent.

Insertion of *K* causes a split at leaf level, followed by the promotion of *K*. This causes a split of the root. *N* is promoted to become the new root. *E* is added to a leaf:



Insertion of *H* causes a leaf to split. *H* is promoted. *O*, *L*, and *J* are added:



Insertion of *Y* and *Q* force two more leaf splits and promotions. Remaining letters are added:

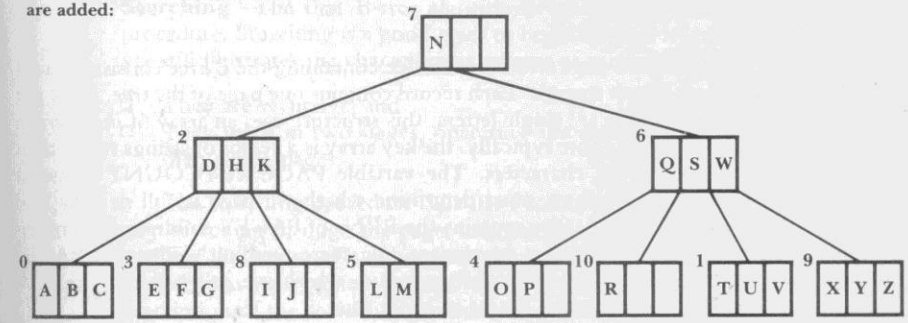
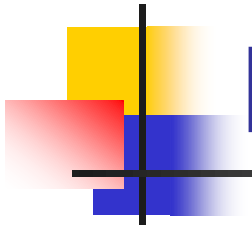


FIGURE 8.18 Growth of a B-tree, part II. The root splits to add a new level; remaining keys are inserted.



Exercício

- Esboce um algoritmo recursivo de busca em uma árvore-B
- Esboce um algoritmo de inserção de chaves em uma árvore-B