



Árvores-B (Parte II)

Leandro C. Cintra

M.C.F. de Oliveira

2004

Fonte: Folk & Zoelick, File Structures

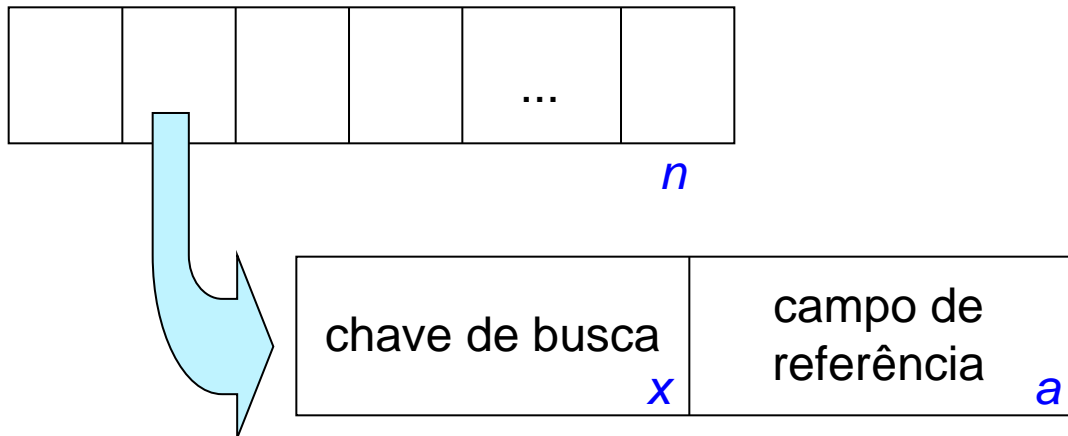
(atualizado 2007 c/ material Profa. Cristina Ciferri)



Construção de árvores-B

Características Gerais

- Organizar e manter um índice para um arquivo de acesso aleatório altamente dinâmico
- Índice
 - n elementos (x,a) de tamanho fixo





Características Gerais

- Índice

- extremamente volumoso

- *Buffer-pool* pequeno

- apenas uma parcela do índice pode ser carregada em memória principal
- operações baseadas em disco

- Desempenho

- proporcional a \log_K^I
ou melhor

- I: tamanho do índice
- K: tamanho da página de disco



Construção *Top-Down* de árvores paginadas

- É simples construir uma árvore paginada se todo o conjunto de chaves é conhecido antes de iniciar a construção
 - Inicia-se pela chave do meio para obter uma árvore balanceada
- Porém, é complicado se as chaves são recebidas em uma seqüência aleatória

Construção *Top-Down* de árvores paginadas

- **Ordem: 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**

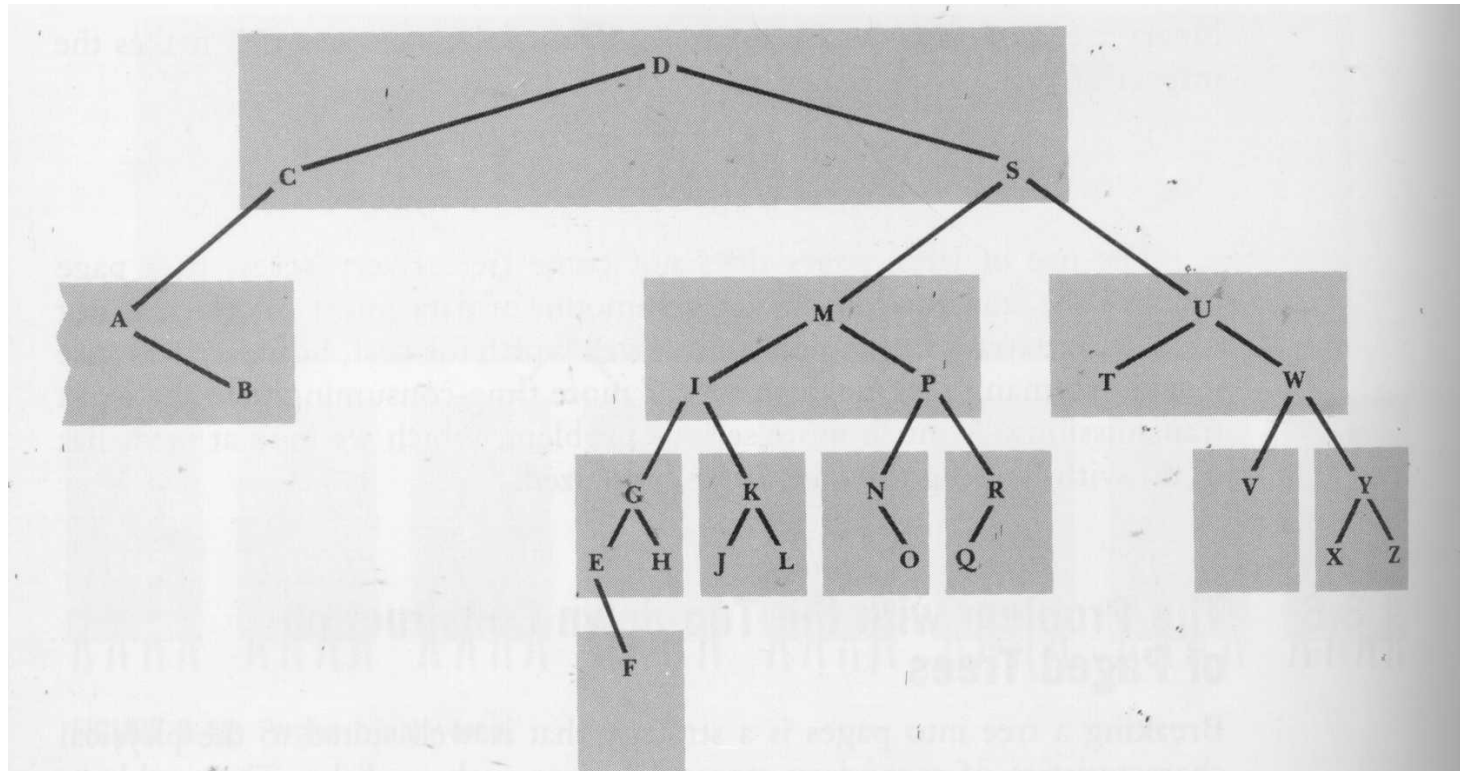


FIGURE 8.13 Paged tree constructed from keys arriving in random input sequence.



Construção *Top-Down* de árvores paginadas

- Na figura anterior, a construção foi feita *top-down*, a partir da raiz
- Sempre que uma chave é inserida a árvore dentro da página sofre uma rotação, sempre que necessário, para manter o balanceamento
- Construção a partir da raiz implica em que as chaves iniciais estarão necessariamente na raiz
- C e D não deveriam estar no topo, pois acabam desbalanceando a árvore de forma definitiva
- Esta árvore não está tão ruim, mas o que aconteceria se as chaves fossem fornecidas em ordem alfabética?



Construção *Top-Down* de árvores paginadas

■ Questões

- como garantir que as chaves na página raiz são boas separadoras, i.e., dividem o conjunto de chaves de maneira balanceada ?
- como impedir o agrupamento de chaves que não deveriam estar na mesma página (como C, D e S, por exemplo)
- como garantir que cada página contenha um número mínimo de chaves ?



Árvore B

- Características

- balanceada
- bottom-up para a criação (em disco)
 - nós folhas → nó raiz

- Inovação

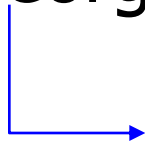
- não é necessário construir a árvore a partir do nó raiz, como é feito para árvores em memória principal e para as árvores anteriores



Construção *Bottom-Up*

■ Conseqüências

- chaves “erradas” não são mais alocadas no nó raiz
 - elimina as questões em aberto de *chaves separadoras* e de *chaves extremas*
- não é necessário tratar o problema de desbalanceamento usando algoritmos de reorganização da árvore



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



Características

- Nó (= página de disco)
 - seqüência ordenada de chaves
 - conjunto de ponteiros
 - número de ponteiros = número de chaves + 1
 - não há uma árvore explícita dentro de uma página (ou nó da árvore)



Características

■ Ordem

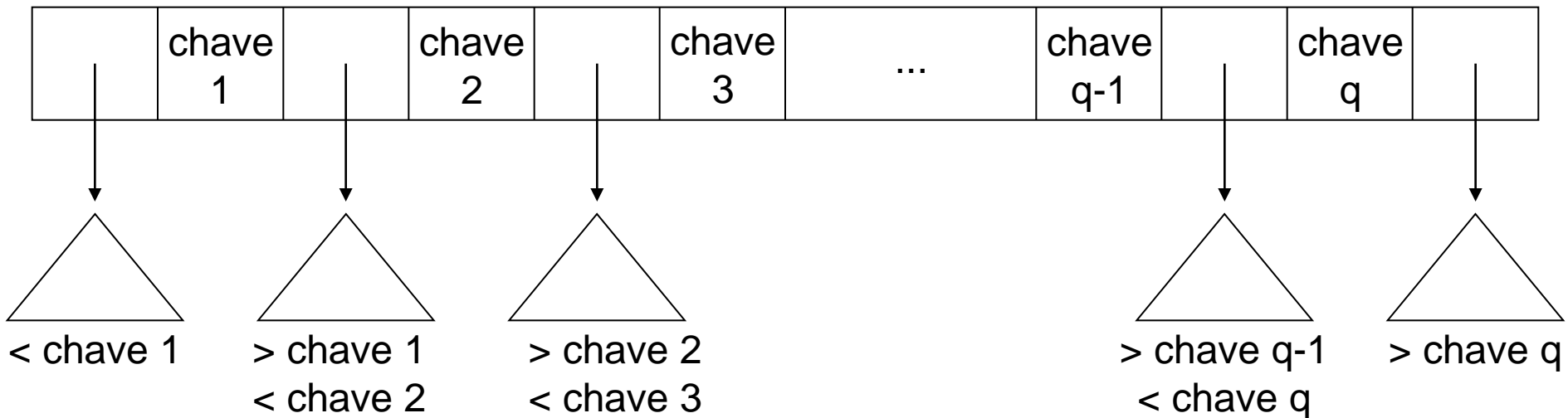
- número máximo de ponteiros que pode ser armazenado em um nó
- exemplo: árvore-B de ordem 8
 - máximo de 7 chaves e 8 ponteiros

■ Observações

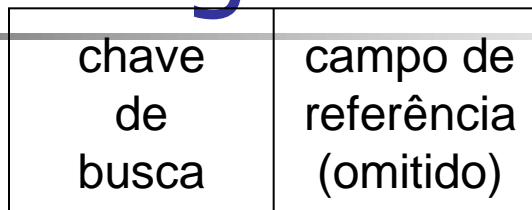
- número máximo de ponteiros é igual ao número máximo de descendentes de um nó
- nós folhas não possuem filhos, e seus ponteiros são nulos



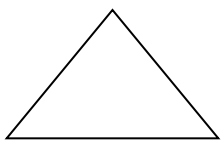
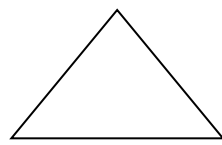
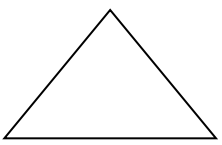
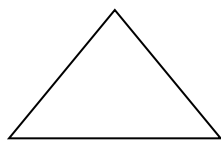
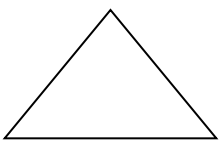
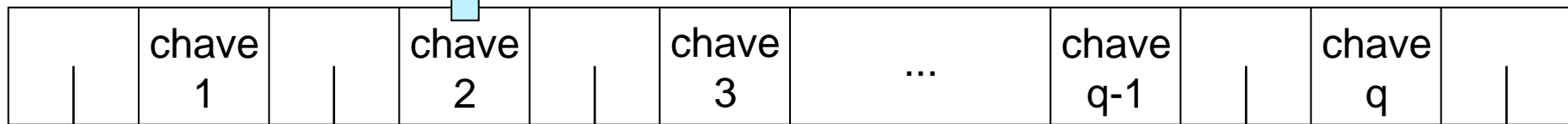
Estrutura Lógica de um Nó



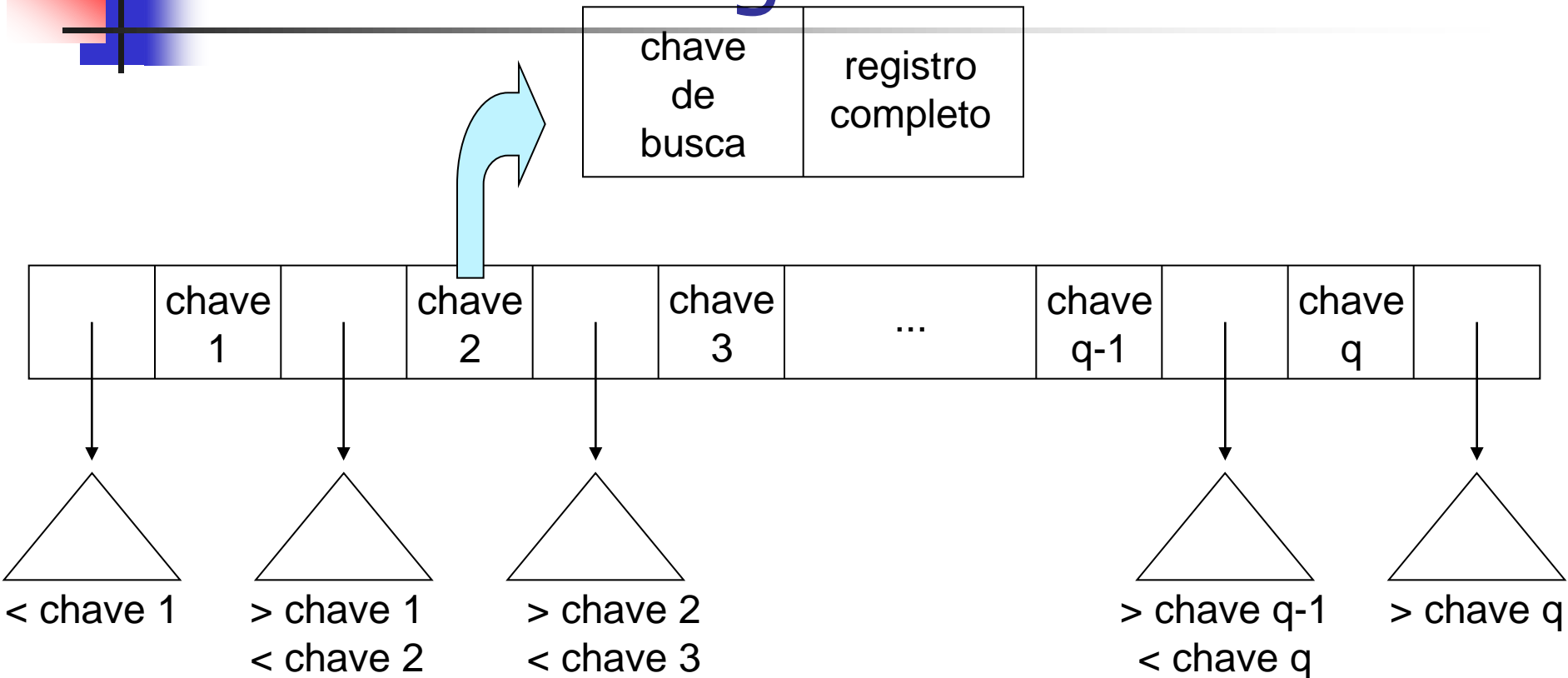
Estrutura Lógica de um Nó



campos de tamanho fixo

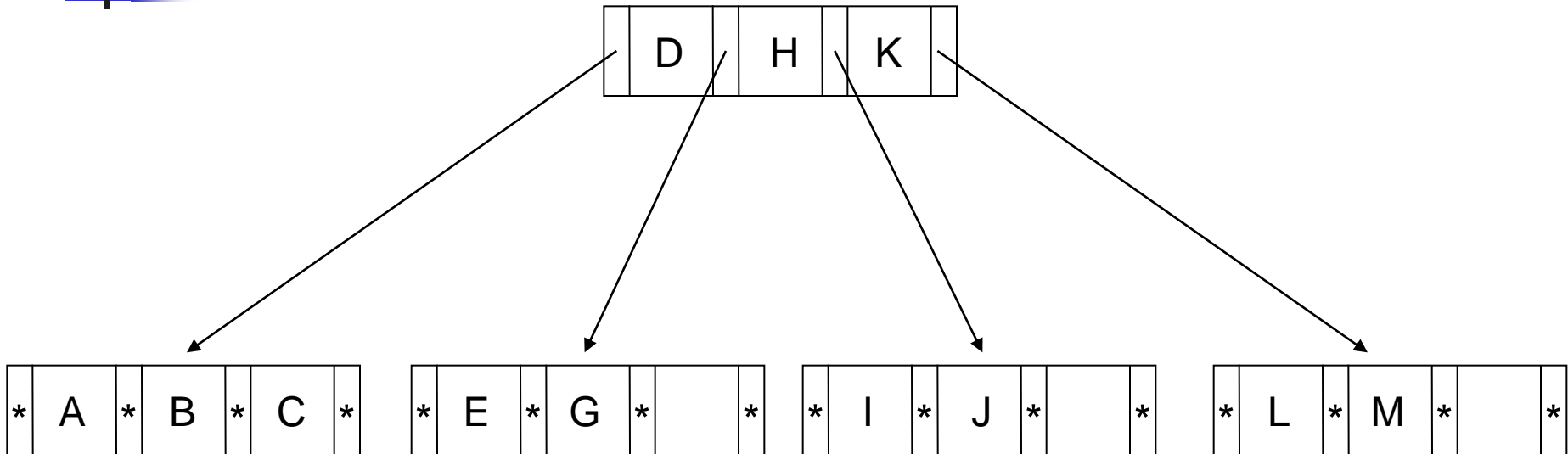


Estrutura Lógica de um Nó





Exemplo





Inserção de Dados (Chave)

- Característica
 - sempre realizada nos nós folhas
- Situações a serem analisadas
 - árvore vazia
 - overflow no nó raiz
 - inserção nos nós folhas



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
 - 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)

*	A	*	B	*	C	*	D	*	E	*	F	*	G	*
---	---	---	---	---	---	---	---	---	---	---	---	---	---	---

Inserção: *Overflow* Nó Raiz

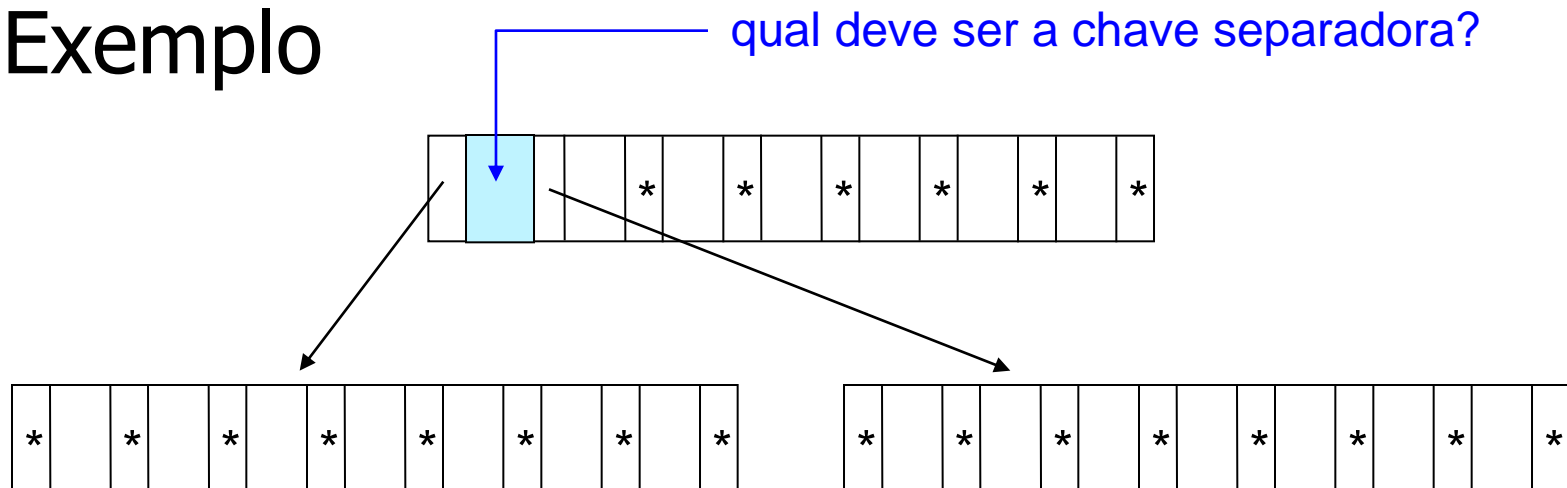
- Passo 1 – particionamento do nó (*split*)
 - nó original → nó original + novo nó
 - *split* 1-to-2
 - as chaves são distribuídas uniformemente nos dois nós
 - chaves do nó original + nova chave
- Exemplo: inserção de J

*	A	*	B	*	C	*	D	*		*		*		*
---	---	---	---	---	---	---	---	---	--	---	--	---	--	---

*	E	*	F	*	G	*	J	*		*		*		*
---	---	---	---	---	---	---	---	---	--	---	--	---	--	---

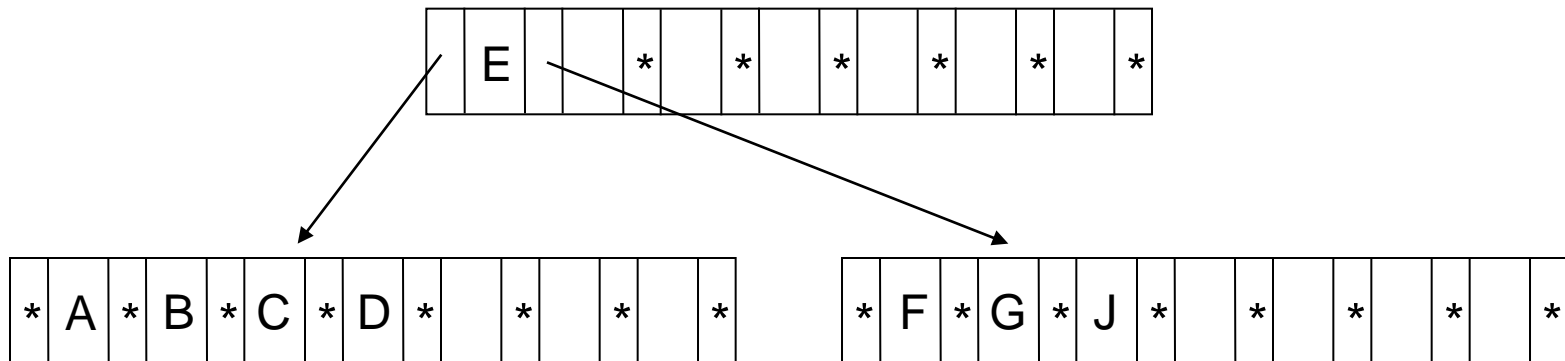
Inserção: *Overflow* Nó Raiz

- Passo 2 – criação de uma nova raiz
 - 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
- Exemplo





Inserção: Nós Folhas

- Passo 1 – pesquisa
 - a árvore é percorrida até encontrar o nó folha no qual a nova chave será inserida
- Passo 2 – inserção em nó com espaço
 - ordenação da chave após a inserção
 - alteração dos valores dos campos de referência

nó folha em
memória principal



Inserção: Nós Folhas

- Passo 2 – inserção em nó cheio
 - particionamento
 - criação de um novo nó
(nó original → nó original + novo nó)
 - distribuição uniforme das chaves nos dois nós
 - promoção
 - escolha da primeira chave do novo nó como chave separadora no nó pai
 - ajuste do nó pai para apontar para o novo nó
 - propagação 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 ...

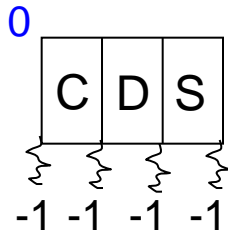
- Passo 1 – inserção de C, S, D

- criação do nó raiz

- C

- C S

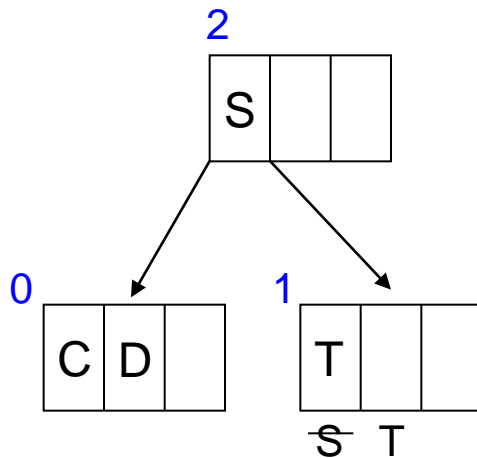
- C D S



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

K ...

- Passo 2 – inserção de T
 - nó raiz cheio

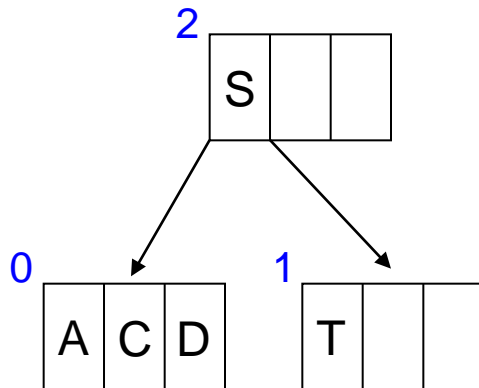


- particionamento do nó
- criação de uma nova raiz
- promoção de S

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

K ...

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

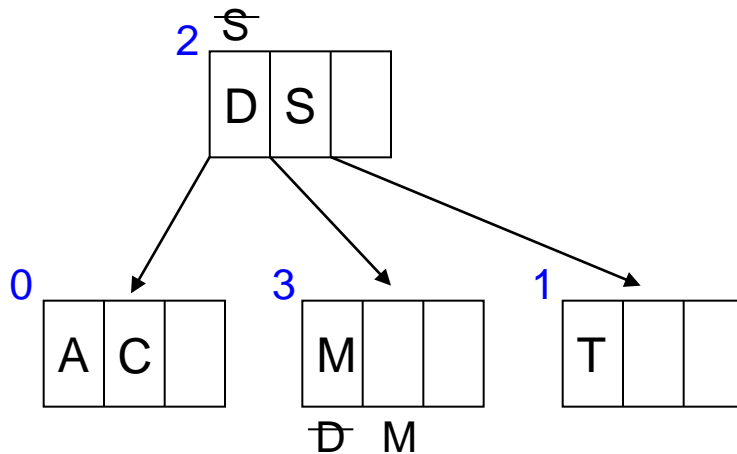


C S D T A M P I B W N G U R

K ...

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

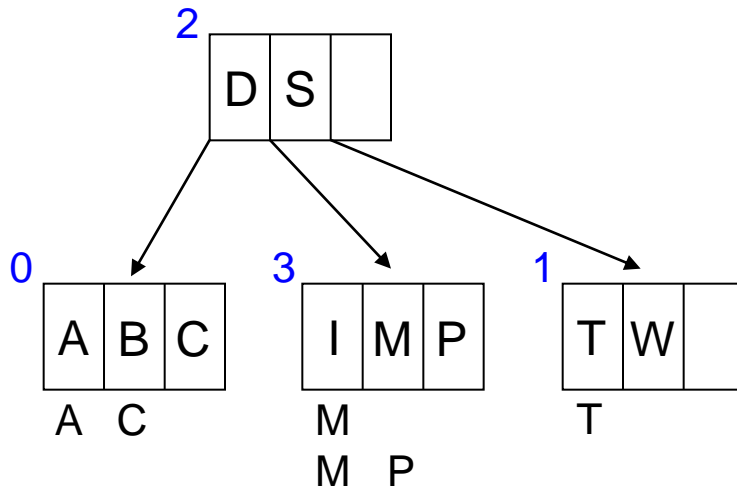
- particionamento do nó
- promoção de D



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

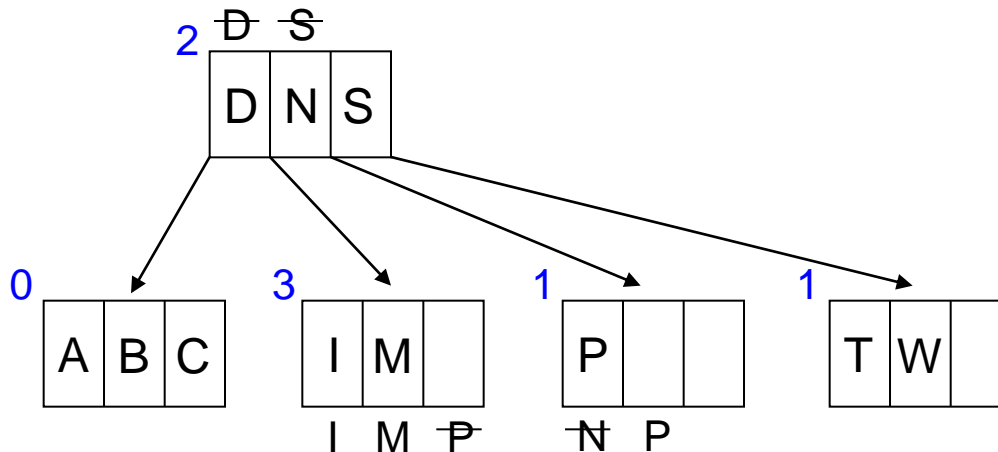


C S D T A M P I B W N G U R

K ...

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

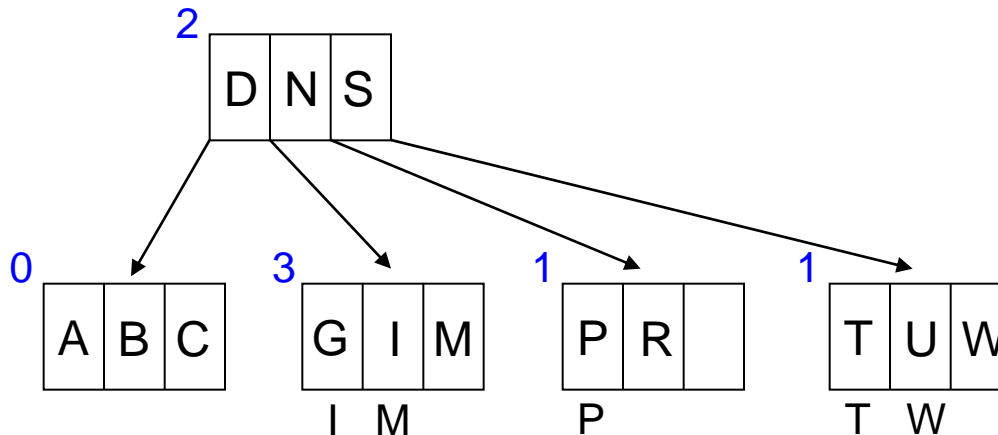
- particionamento do nó
- promoção de N



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

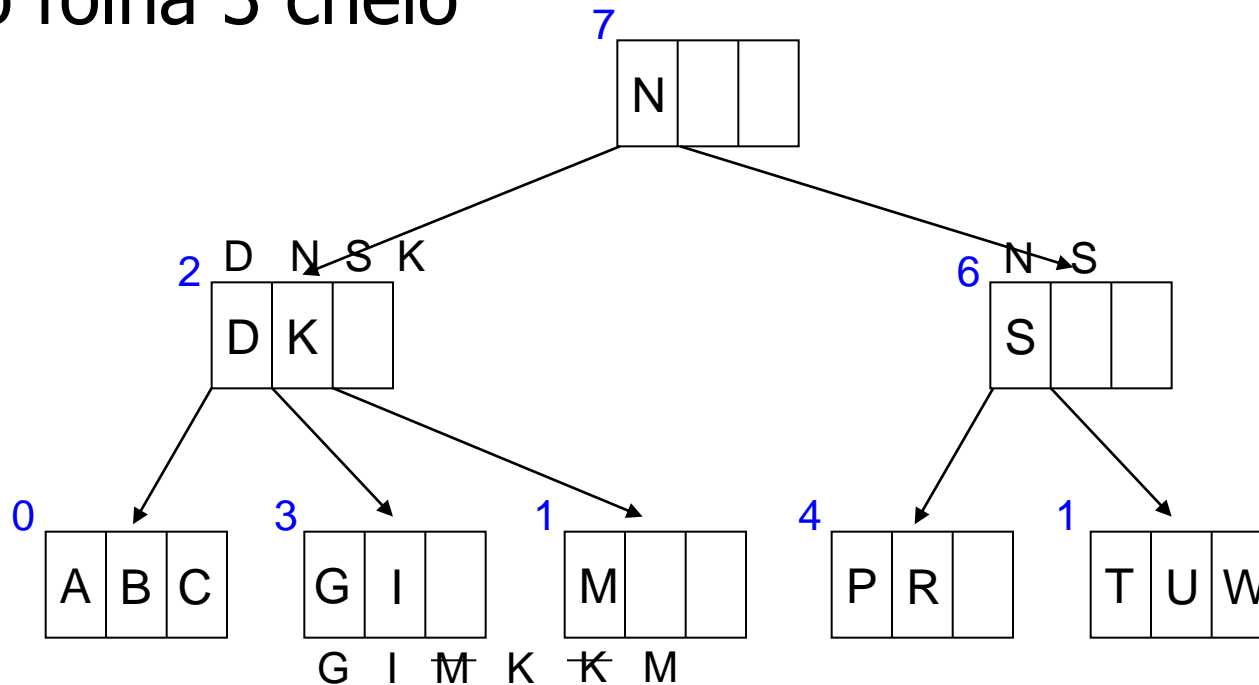


C S D T A M P I B W N G U R

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

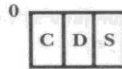


Exercícios

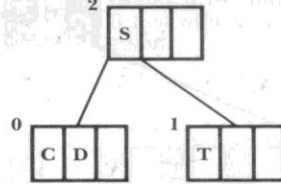
- Na árvore-B do exemplo anterior, insira a chave \$, sendo que $\$ < A$.
- Insira as seguintes chaves em um índice árvore-B
 - C S D T A M P I B W N G U R K E H O L J Y Q Z
F X V
 - diferentemente do exemplo anterior, escolha o último elemento do primeiro nó para promoção durante o particionamento do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

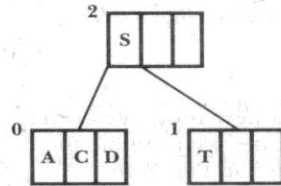
Insertion of C, S, and D into the initial page:



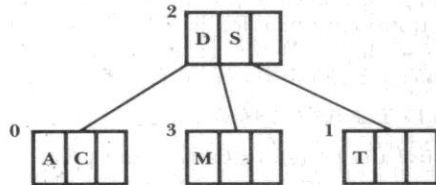
Insertion of T forces the split and the promotion of S:



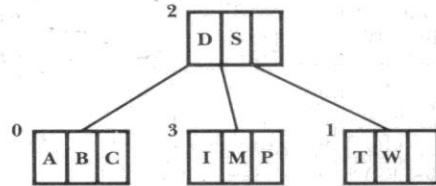
A added without incident:



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:

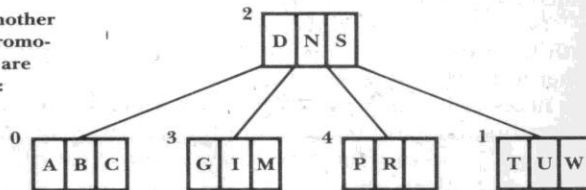
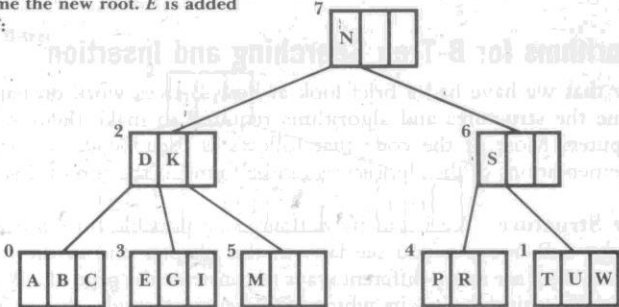
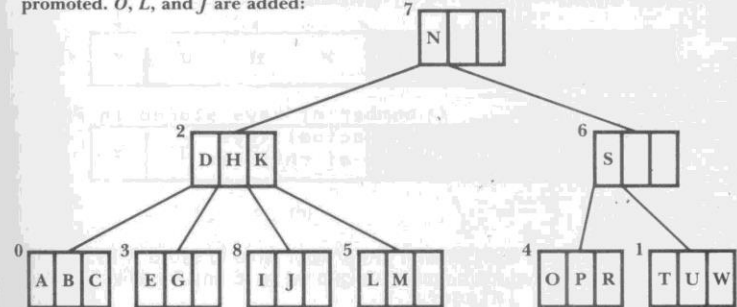


FIGURE 8.17 Growth of a B-tree, part I. The tree grows to a point at which splitting of the root is imminent.

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:

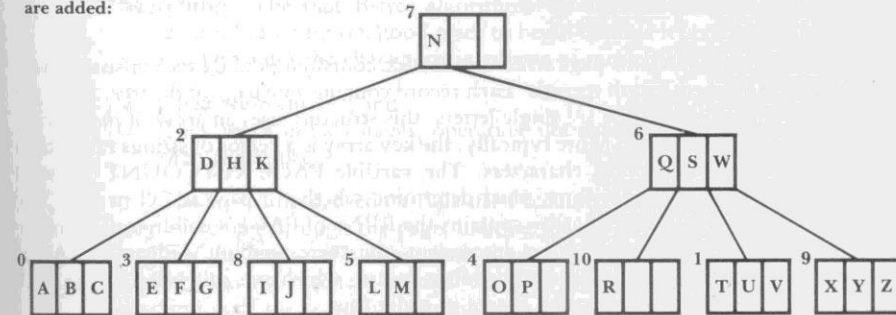


FIGURE 8.18 Growth of a B-tree, part II. The root splits to add a new level; remaining keys are inserted.