Árvores B (part 3)

Prof. Dr. Lucas C. Ribas

Disciplina: Estrutura de Dados II

Departamento de Ciências de Computação e Estatística

UNIVERSIDADE ESTADUAL PAULISTA
"JÚLIO DE MESQUITA FILHO"

IBILCE / UNESP - CÂMPUS DE SÃO JOSÉ DO RIO PRETO

Propriedades das Árvores-B - Relembrando...

- Para uma árvore-B de ordem **m**

1. cada página tem, no máximo, **m** descendentes
2. cada página, exceto a raiz e as folhas, tem no mínimo ⌈**m/2**⌉ descendentes
3. a raiz tem, no mínimo, dois descendentes - a menos que seja uma folha
4. todas as folhas estão no mesmo nível
5. uma página não folha que possui **k** descendentes contém **k-1** chaves
6. uma página folha contém, no mínimo ⌈**m/2**⌉**-1** e, no máximo, **m-1** chaves

unesp

Eliminação, Redistribuição e Concatenação

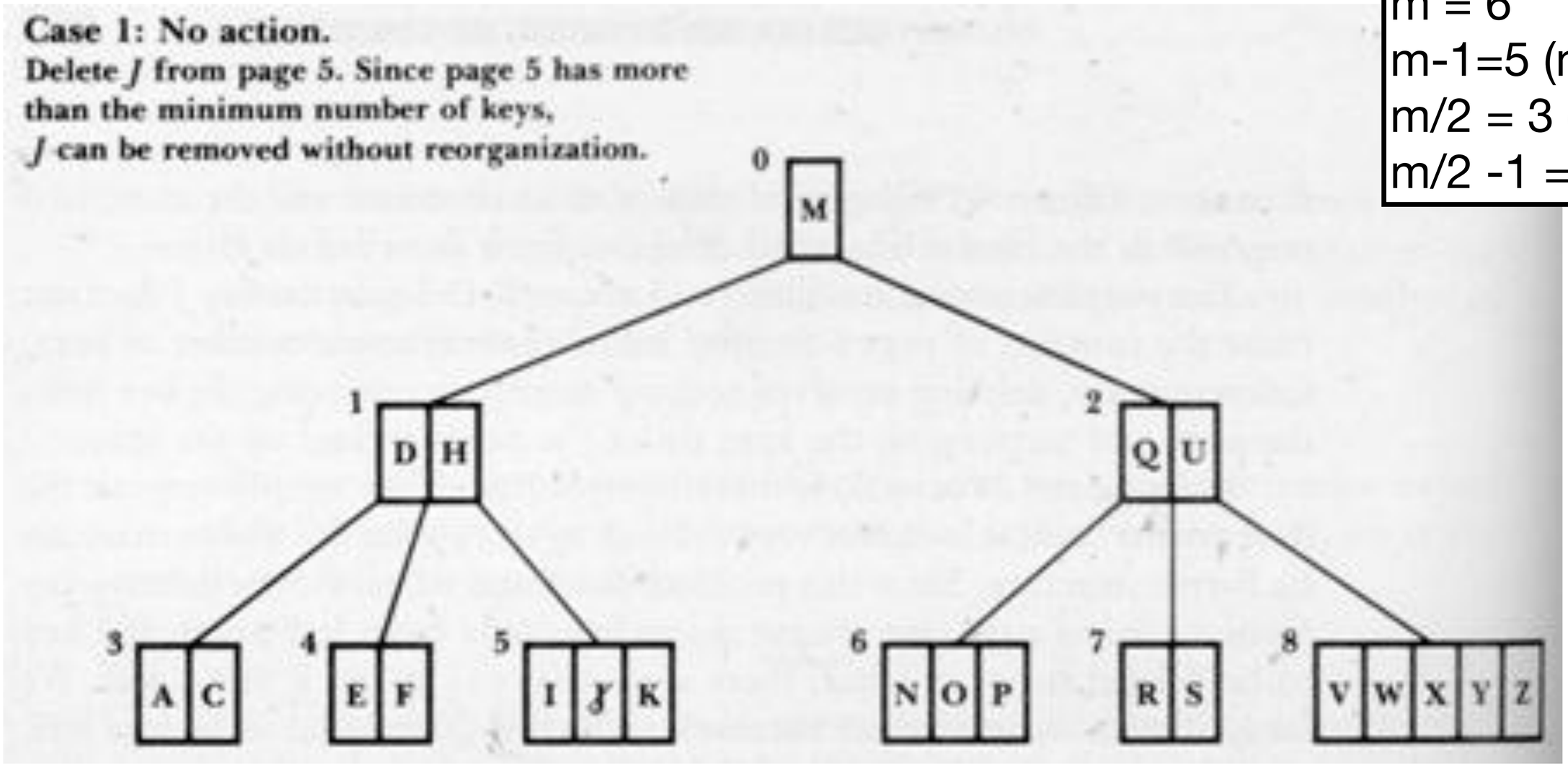
- O **split** garante a **manutenção das propriedades da árvore-B** durante a inserção
- Essas propriedades precisam ser mantidas, também, durante a eliminação de chaves
- Há vários casos para se analisar (árvore de ordem **m**)

Eliminação: Caso 1

- **Caso 1:** eliminação de uma chave em uma página folha, sendo que o número mínimo de chaves na página é respeitado: $\lceil m/2 \rceil - 1$
- **Solução:** chave é retirada e os registros internos à página são reorganizados

unesp

Eliminação: Caso 1



m = 6
m-1=5 (max)
m/2 = 3 (min)
m/2 -1 = 2 (min)

Eliminação: Caso 2

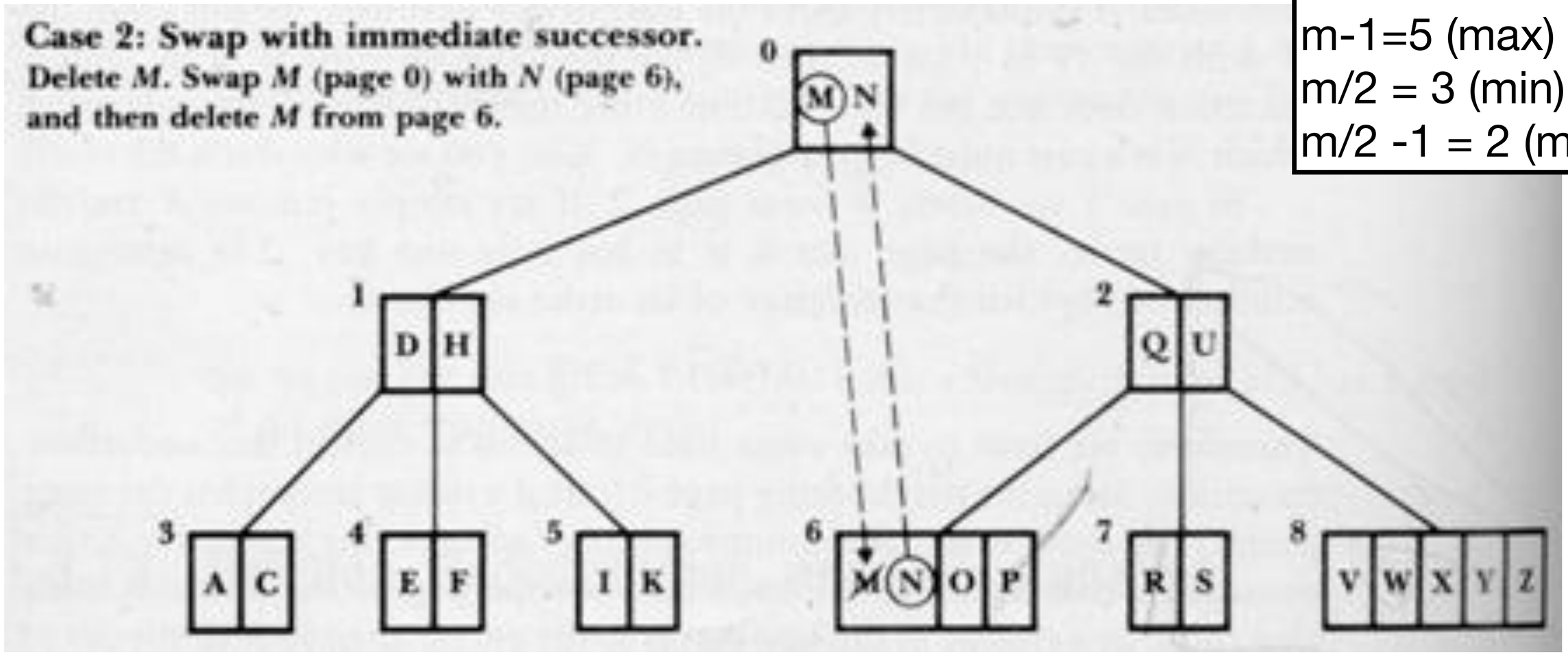
- **Caso 2:** eliminação de uma **chave que não está em uma folha**
- **Solução:** sempre eliminamos de páginas folha*
  - Se uma chave deve ser eliminada de uma página que não é folha, trocamos a chave com sua sucessora imediata (ou com a predecessora imediata) que está numa folha
  - A seguir, eliminamos a chave da folha

  * Análogo a AVL

unesp

Eliminação: Caso 2

m = 6
m-1=5 (max)
m/2 = 3 (min)
m/2 -1 = 2 (min)

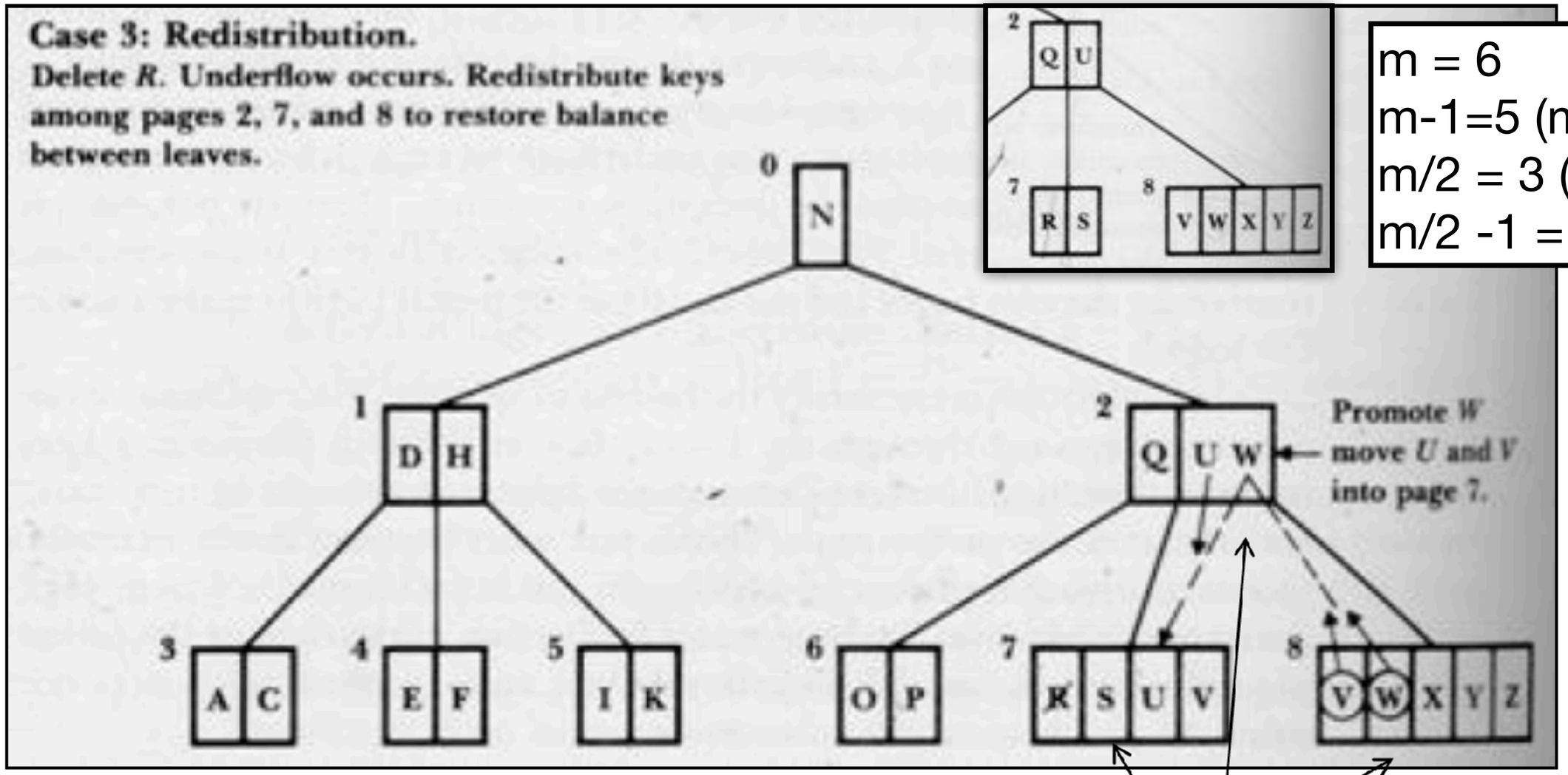


Eliminação: Caso 3

- **Caso 3:** eliminação causa *underflow* na página (folha)
- **Solução:** redistribuição
  - Procura-se uma página irmã (mesmo pai) que contenha mais chaves do que o mínimo: se existir, redistribuem-se as chaves entre essas páginas
  - A redistribuição pode provocar uma alteração na chave separadora que está no nó pai

unesp

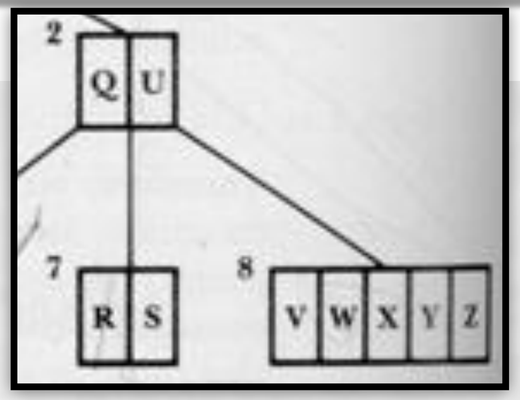
Eliminação: Caso 3



m = 6
m-1=5 (max)
m/2 = 3 (min)
m/2 -1 = 2 (min)

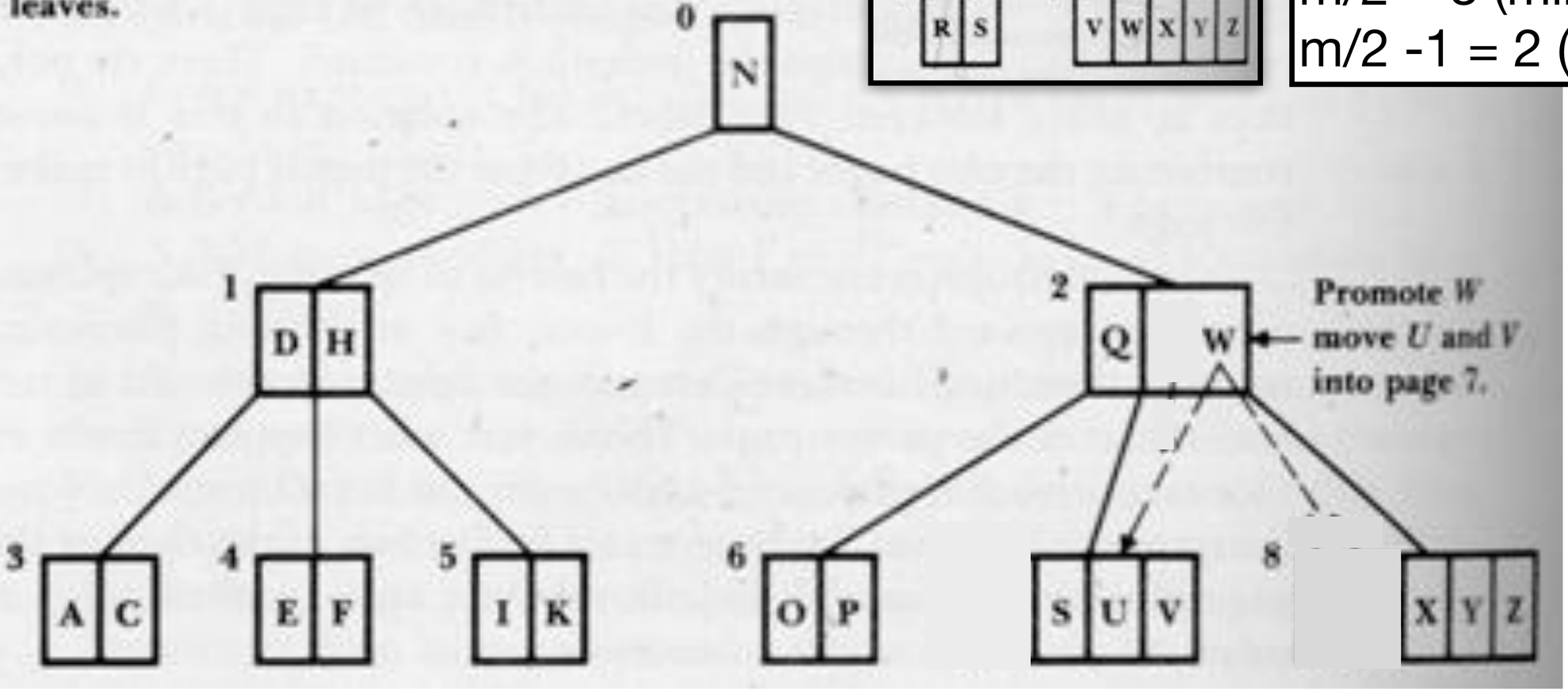
S U V W X Y Z

Eliminação: Caso 3



m = 6
m-1=5 (max)
m/2 = 3 (min)
m/2 -1 = 2 (min)

Case 3: Redistribution.
Delete *R*. Underflow occurs. Redistribute keys among pages 2, 7, and 8 to restore balance between leaves.

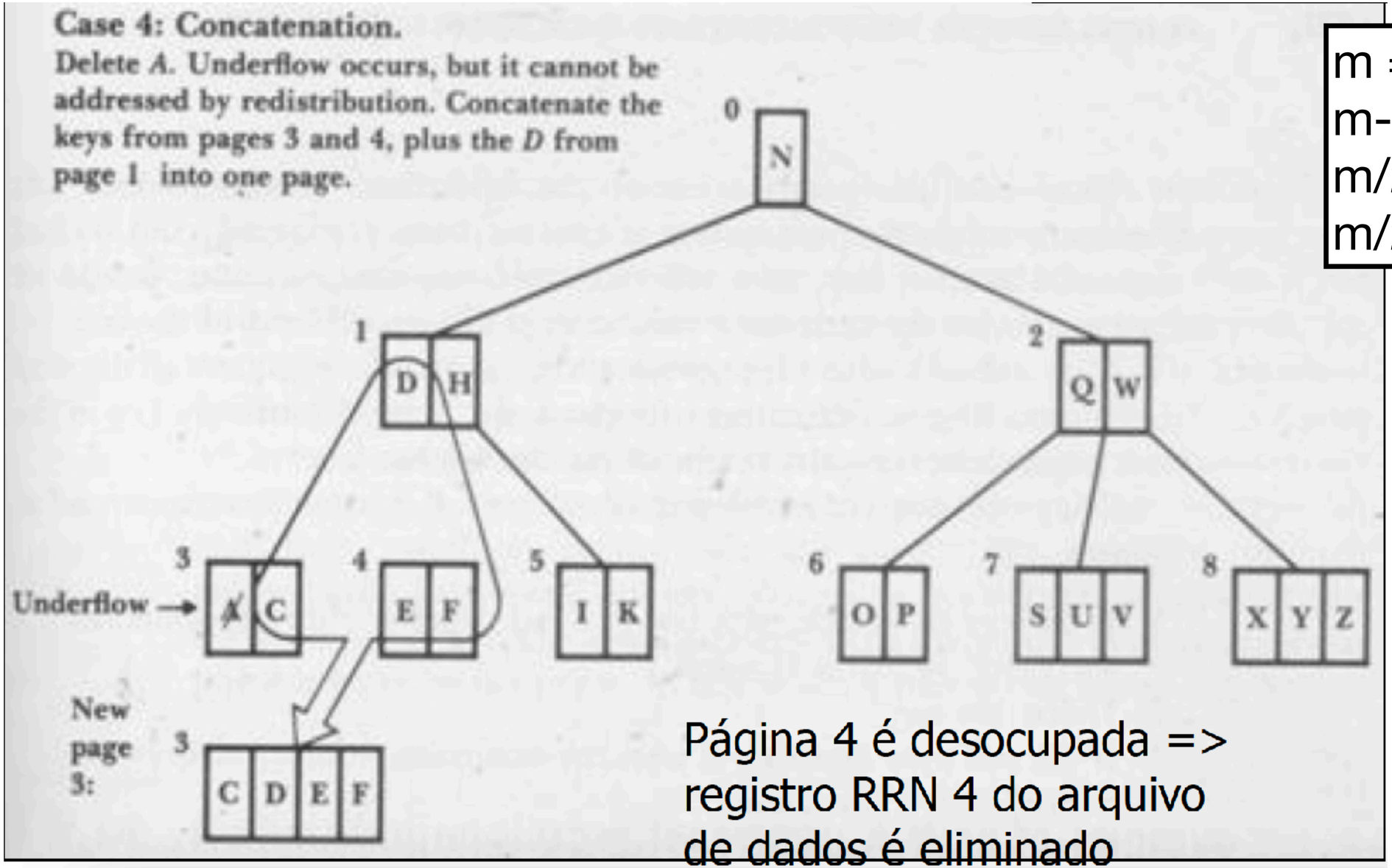


Eliminação: Caso 4

- **Caso 4:** ocorre *underflow* e a redistribuição não pode ser aplicada
- Não existem chaves suficientes para dividir entre as duas páginas irmãs. A que sofreu *underflow* tem m/2-2 chaves, e a outra, m/2-1 chaves
- **Solução:** concatenação
  - Combina-se o conteúdo das duas páginas (m-3) mais a chave separadora da página pai para formar uma única página com m-2 chaves
  - A concatenação é o inverso do processo de particionamento
    - Como consequência, a eliminação na página pai também pode causar *underflow*
    - Uma página é liberada (registro eliminado do arquivo de dados)

unesp

Eliminação: Caso 4



m = 6
m-1=5 (max)
m/2 = 3 (min)
m/2 -1 = 2 (min)

Página 4 é desocupada => registro RRN 4 do arquivo de dados é eliminado

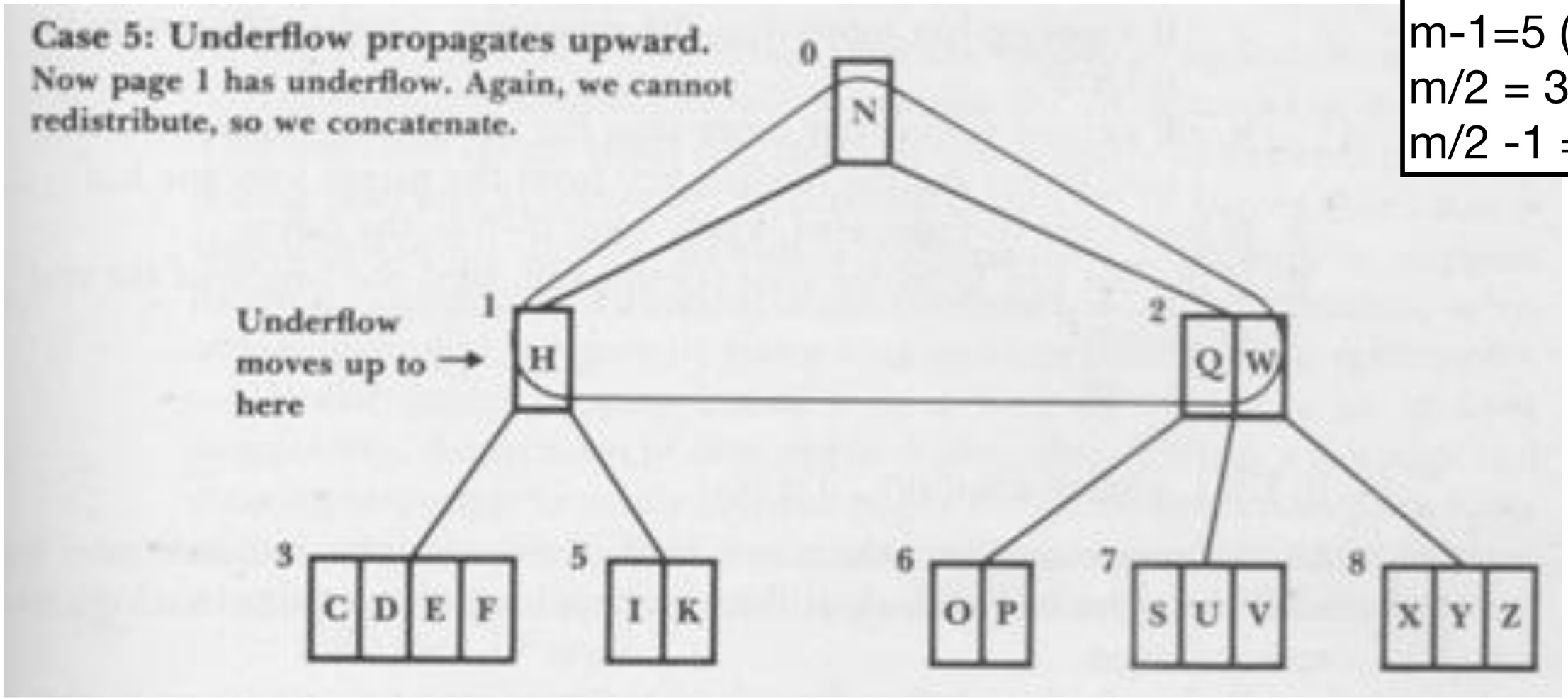
Eliminação: Caso 5

- **Caso 5:** *underflow* da página pai, como consequência da concatenação
- **Solução:** utiliza-se redistribuição ou concatenação novamente

unesp

Eliminação: Caso 5

m = 6
m-1=5 (max)
m/2 = 3 (min)
m/2 -1 = 2 (min)

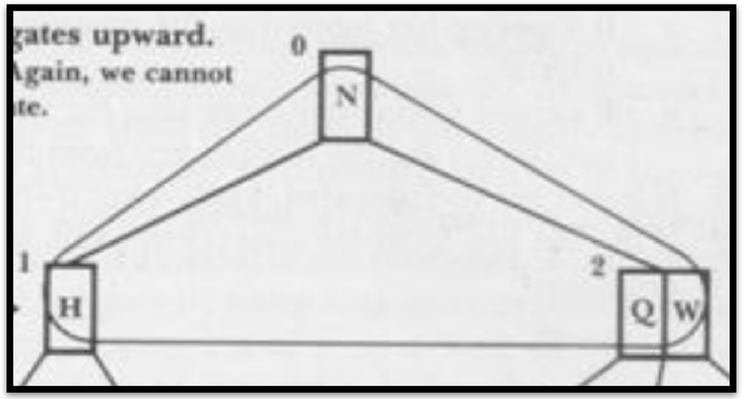


Eliminação: Caso 6

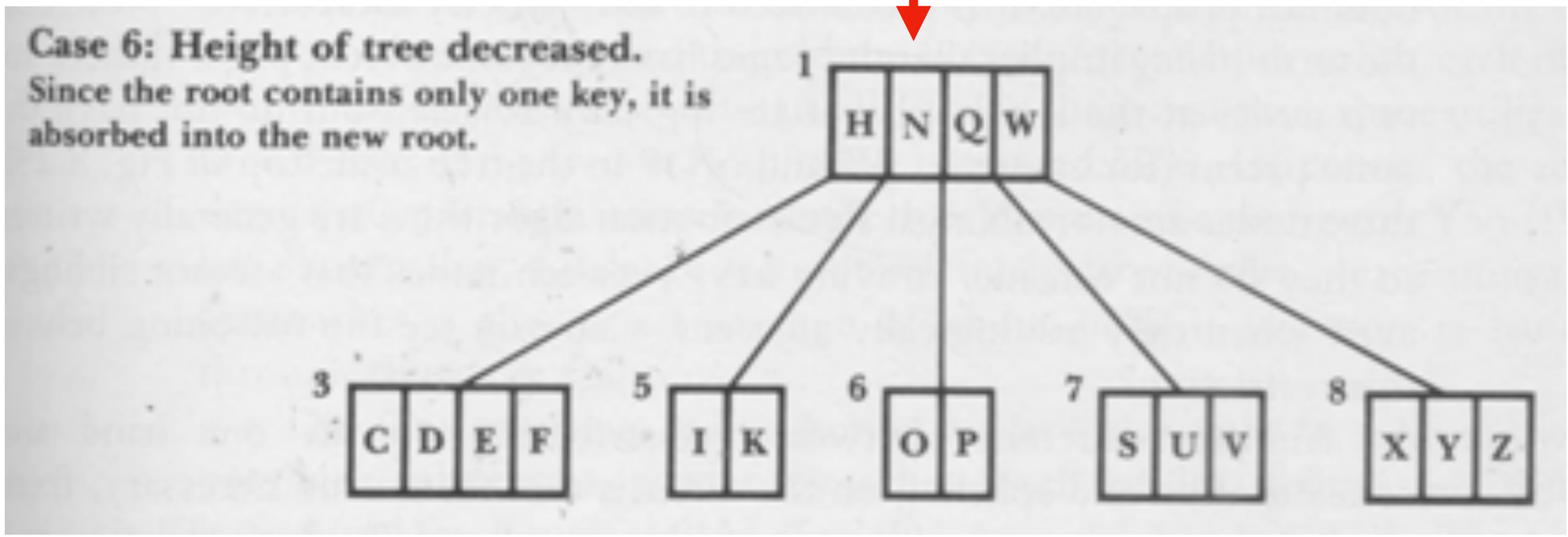
- **Caso 6:** diminuição da altura da árvore
  - Ocorre quando o nó raiz tem uma única chave
- **Solução:** concatenação nos seus nós filhos

unesp

Eliminação: Caso 6



m = 6
m-1=5 (max)
m/2 = 3 (min)
m/2 -1 = 2 (min)



Registros RRN 0 e 2 são eliminados do arquivo de dados

Algoritmo para eliminação de chaves em árvores-B

1. Se a chave não estiver numa folha, troque-a com sua sucessora*

2. Elimine a chave da folha

3. Se a página continuar com o número mínimo de chaves, fim

4. Se a página tem uma chave a menos que o mínimo, verifique as páginas irmãs a esquerda e a direita

4.1. se uma delas tiver mais do que o número mínimo de chaves, aplique redistribuição

4.2. senão concatene a página com uma das irmãs e a chave separadora do pai

5. Se ocorreu concatenação, aplique os passos de 3 a 6 para a página pai

6. Se a última chave da raiz for removida, a altura da árvore diminui

*primeira chave da página mais à esquerda da filha à direita; ou última da página mais à direita da filha à esquerda

unesp

Exercício

- Usando o algoritmo anterior, remova as chaves **A, B, Q, R** e **M** da árvore-B de ordem 5 abaixo

1. Se a chave não estiver numa folha, troque-a com sua sucessora*

2. Elimine a chave da folha

3. Se a página continuar com o número mínimo de chaves, fim

4. Se a página tem uma chave a menos que o mínimo, verifique as páginas irmãs a esquerda e a direita

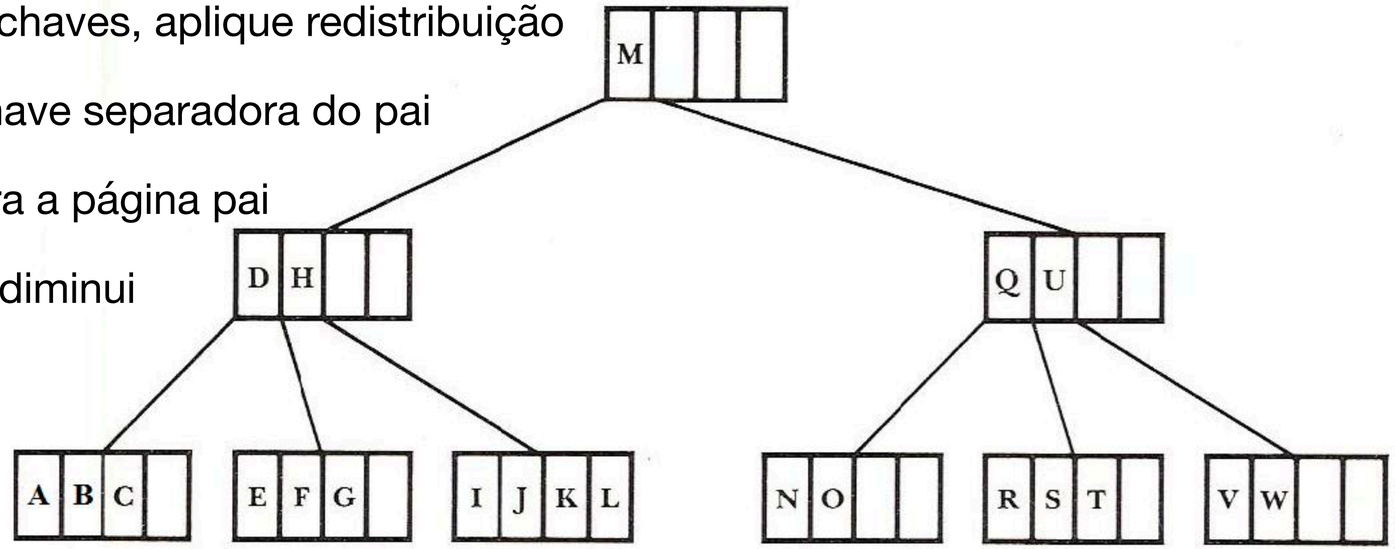
4.1. se uma delas tiver mais do que o número mínimo de chaves, aplique redistribuição

4.2. senão concatene a página com uma das irmãs e a chave separadora do pai

5. Se ocorreu concatenação, aplique os passos de 3 a 6 para a página pai

6. Se a última chave da raiz for removida, a altura da árvore diminui

m = 6
m-1=5 (max)
m/2 = 3 (min)
m/2 -1 = 2 (min)



Efeitos da Redistribuição entre páginas irmãs

unesp

Efeitos da Redistribuição entre páginas irmãs

- Diferentemente do particionamento e da concatenação, o efeito da redistribuição é local
  - Não existe propagação
- Outra diferença é que não existe regra fixa para o rearranjo das chaves
  - redistribuição pode restabelecer as propriedades da árvore-B movendo apenas uma chave de uma página irmã para a página com problema, ou
  - estratégia usual é redistribuir as chaves igualmente entre as páginas

unesp

Redistribuição durante inserção

- Redistribuição pode ser usada na inserção
- Seria uma opção desejável também na inserção
  - Em vez de particionar uma página cheia em duas páginas novas semi-vazias, pode-se optar por colocar a chave que sobra (ou mais de uma!) em outra página
    - **Melhor utilização do espaço** alocado para a árvore

unesp

Redistribuição vs. Particionamento (Splitting)

- Depois do particionamento de uma página, cada página fica 50% vazia
  - Portanto, a utilização do espaço, no pior caso, em uma árvore-B que utiliza splitting é de cerca de 50%
  - Em média, para árvores grandes, foi provado que o índice de ocupação de páginas é de ~69%
- Estudos empíricos indicam que a utilização de redistribuição pode elevar esse índice para 85%
  - Resultados sugerem que qualquer aplicação séria de árvore-B deve utilizar, de fato, **redistribuição** durante a inserção

unesp

Árvores-B* (B*-Trees)

- Proposta por Knuth em 1973, essa nova organização tenta redistribuir as chaves durante a inserção antes de particionar o nó
  - É uma variação de árvore-B na qual cada nó tem, no mínimo, 2/3 do número máximo de chaves
- A geração destas árvores utiliza uma variação do processo de particionamento
  - O particionamento é adiado até que duas páginas irmãs estejam cheias
  - Realiza-se, então, a divisão do conteúdo das duas páginas em 3 páginas (*two-to-three split*)

unesp

Two-to-three split

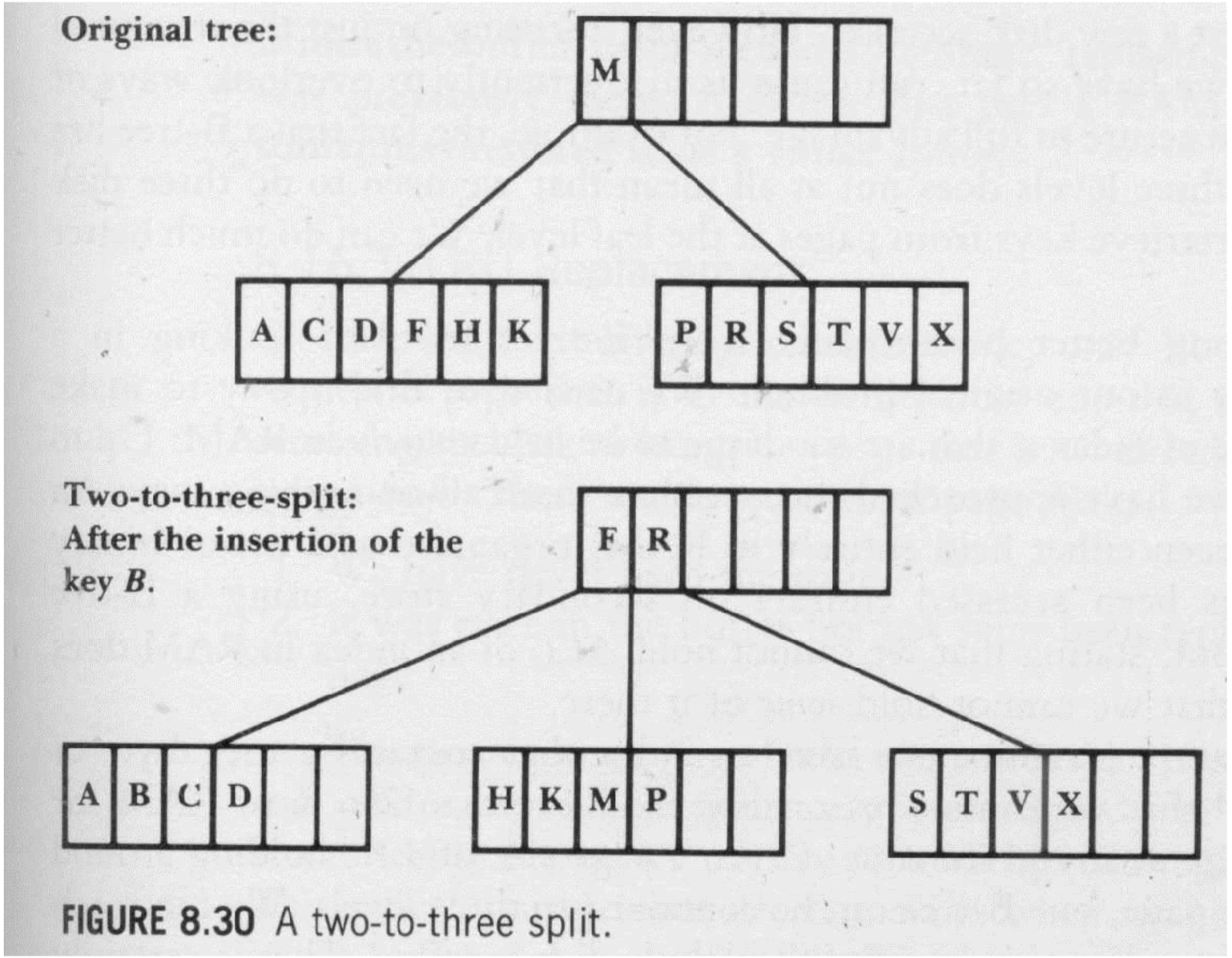


FIGURE 8.30 A two-to-three split.

Propriedades de árvores-B*

- Cada página tem no máximo **m** descendentes
- Toda página, **exceto a raiz e as folhas**, tem no mínimo **(2m-1)/3** descendentes
- A raiz tem pelo menos 2 descendentes
- Todas as folhas estão no mesmo nível
- Uma página não-folha com k descendentes contém k-1 chaves
- Uma página folha contém no mínimo **⌊(2m-1)/3⌋** e no máximo **m-1** chaves

unesp

Observações

- Esta propriedade afeta as regras para remoção e redistribuição
  - Com 2/3 de ocupação, toda página irmã tem condições de redistribuir após um *underflow* em página vizinha.
- Deve-se tomar cuidado na implementação, uma vez que a raiz nunca tem irmã e, portanto, requer tratamento especial
- Uma solução é dividir a raiz usando a divisão convencional (*one-to-two split*), outra é permitir que a raiz seja um nó com maior capacidade

Árvores-B Virtuais

- **Árvores-B** são muito eficientes, mas **podem ficar ainda melhores**
  - Observe, por exemplo, que o fato da árvore ter profundidade 3 não implica necessariamente fazer 3 acessos para recuperar as páginas folhas (se a página desejada já estiver na RAM, por buferização)
  - O fato de não podermos manter todo o índice na RAM não significa que não se possa manter pelo menos parte do índice em RAM

unesp

Árvores-B Virtuais

- **Exemplo**
  - Suponha que temos um índice que ocupa 1 MB, e que temos disponíveis 256KB de RAM
  - Supondo que uma página usa 4 KB, e armazena em torno de 64 chaves por página (ordem 65)
  - Nossa árvore-B pode estar totalmente contida em 3 níveis (altura 3 comporta 1.7 MB)
  - Podemos atingir qualquer página com, no máximo, 3 acessos a disco
  - Mas, se a raiz for mantida todo o tempo na memória, ainda sobraria muito espaço em RAM e, com essa solução simples, o pior caso do número de acessos diminui em 2 (um acesso a menos)

unesp

Árvores-B Virtuais

- Podemos generalizar esta idéia e ocupar toda a memória disponível com quantas páginas pudermos, sendo que, quando precisarmos da página, ela pode já estar na RAM
- Se não estiver, ela é carregada para a memória, substituindo uma página que estava em memória
- Tem-se um RAM *buffer* que, algumas vezes, é chamado de árvore-B virtual

unesp

Política de gerenciamento de substituição: LRU

- Se a página não estiver em RAM, e esta estiver cheia, precisamos escolher uma página para ser substituída
- Uma opção: **LRU (Last Recently Used)**
  - substitui-se a página que foi acessada menos recentemente
- O processo de acessar o disco para trazer uma página que não está no buffer é denominado ***page fault***

unesp

Substituição baseada na altura da árvore

- Podemos optar por colocar todos os níveis mais altos da árvore em RAM
  - No exemplo de 256KB de RAM e páginas de 4KB, podemos manter até 64 páginas em memória
- Isso comporta a raiz e mais, digamos, as 8 ou 10 páginas que compõem o segundo nível
  - Ainda sobra espaço (utiliza-se LRU), e o número de acessos diminui em mais uma unidade
- **Importante**: bufferização deve ser incluída em qualquer situação real de utilização de árvore-B

unesp

Alocação da informação associada à chave

- E a informação associada às chaves (os demais campos dos registros), onde fica?
- Se a informação for mantida junto com a chave, ganha-se um acesso a disco, mas perde-se no número de chaves que pode ser colocado em uma página
  - Isso reduz a ordem da árvore, e aumenta a sua altura
- Se ficar em um arquivo separado, a árvore é realmente usada como índice, e cada chave tem o RRN, ou *byte offset*, que dá a posição do registro associado no arquivo de dados

unesp

Registros e Chaves de tamanho variável

- Até agora adotamos chaves de tamanho fixo
- Em muitas situações, pode-se ter economia significativa de espaço usando chaves de tamanho variável
- Índices secundários referenciando listas invertidas são um bom exemplo desta situação
- As árvores-B^+ adotam uma estrutura de página apropriada para acomodar chaves de tamanho variável

unesp

Referências

- FOLK, M.J. File Structures, Addison-Wesley, 1992.
- File Structures: Theory and Pratice”, P. E. Livadas, Prentice-Hall, 1990;
- Contém material extraído e adaptado das notas de aula dos professores Moacir Ponti, Thiago Pardo, Leandro Cintra, Thelma Cecília Chiossi e Maria Cristina de Oliveira.