ÁRVORES B

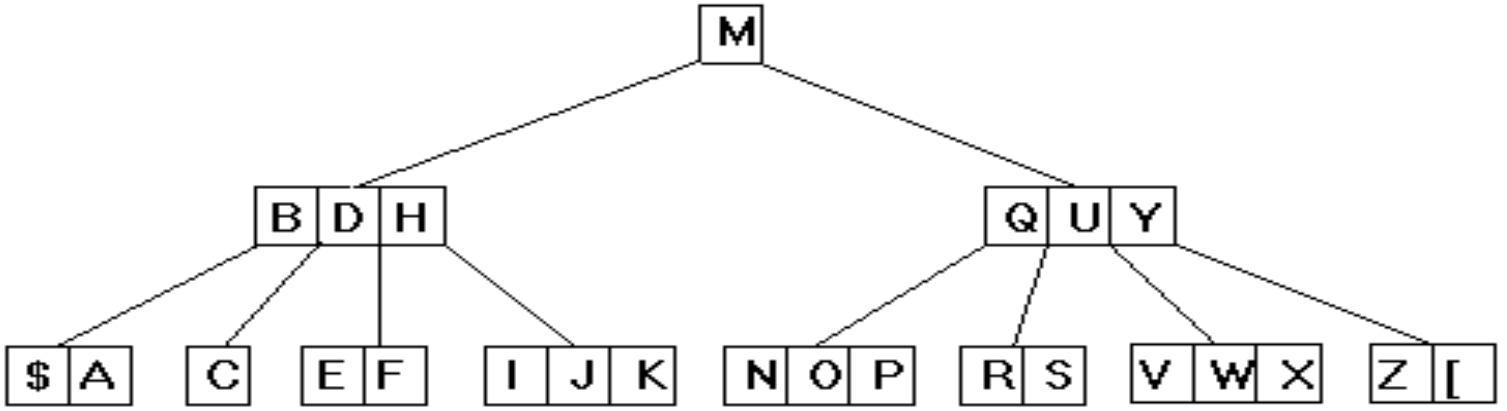
São árvores balanceadas, desenvolvidas para otimizar o acesso em armazenamento secundário.

Os nós da árvore B podem ter muitos filhos. Esse fator de ramificação é determinante para reduzir o número de acessos ao ''disco''.

Árvores B são balanceadas, ou seja, sua altura é O(lg(n)).

ÁRVORES B

Árvores B são generalizações de árvores binárias balanceadas. Um nó pode ter várias chaves e filhos.



ÁRVORES B

O armazenamento estável tem o custo de acesso muito alto quando comparado ao acesso à memória RAM.

Um acesso deve ser aproveitado da melhor maneira possível, trazendo o máximo de informação.

A quantidade de dados utilizados numa árvore B, obviamente, não cabe na memória interna de uma só vez, por isso é necessário paginá-la.

Observação: Usamos nó como sinônimo de *página*. Não confundir nó com chave, já que em uma árvore B um nó pode conter diversas chaves.

ÁRVORES B – Definição de ordem*

Seja d um número natural.

Uma árvore B de ordem d é uma árvore ordenada que é vazia, ou que satisfaz as seguintes condições:

a. A raiz é uma folha ou tem no mínimo 2 filhos;

b. Cada nó diferente da raiz e das folhas possui no mínimo (d + 1) filhos;

c. Cada nó tem no máximo (2d + 1) filhos;

d. Todas as folhas estão no mesmo nível.

*esta é a definição que usaremos !

ÁRVORES B – outra definição possível para a ordem

Seja g um inteiro, $g \geq 3$.

Uma árvore B de ordem d é uma árvore ordenada que é vazia, ou que satisfaz as seguintes condições:

a. A raiz é uma folha ou tem no mínimo 2 filhos;

b. Cada nó diferente da raiz e das folhas possui no mínimo teto de g/2 filhos;

c. Cada nó tem no máximo g filhos;

d. Todas as folhas estão no mesmo nível.

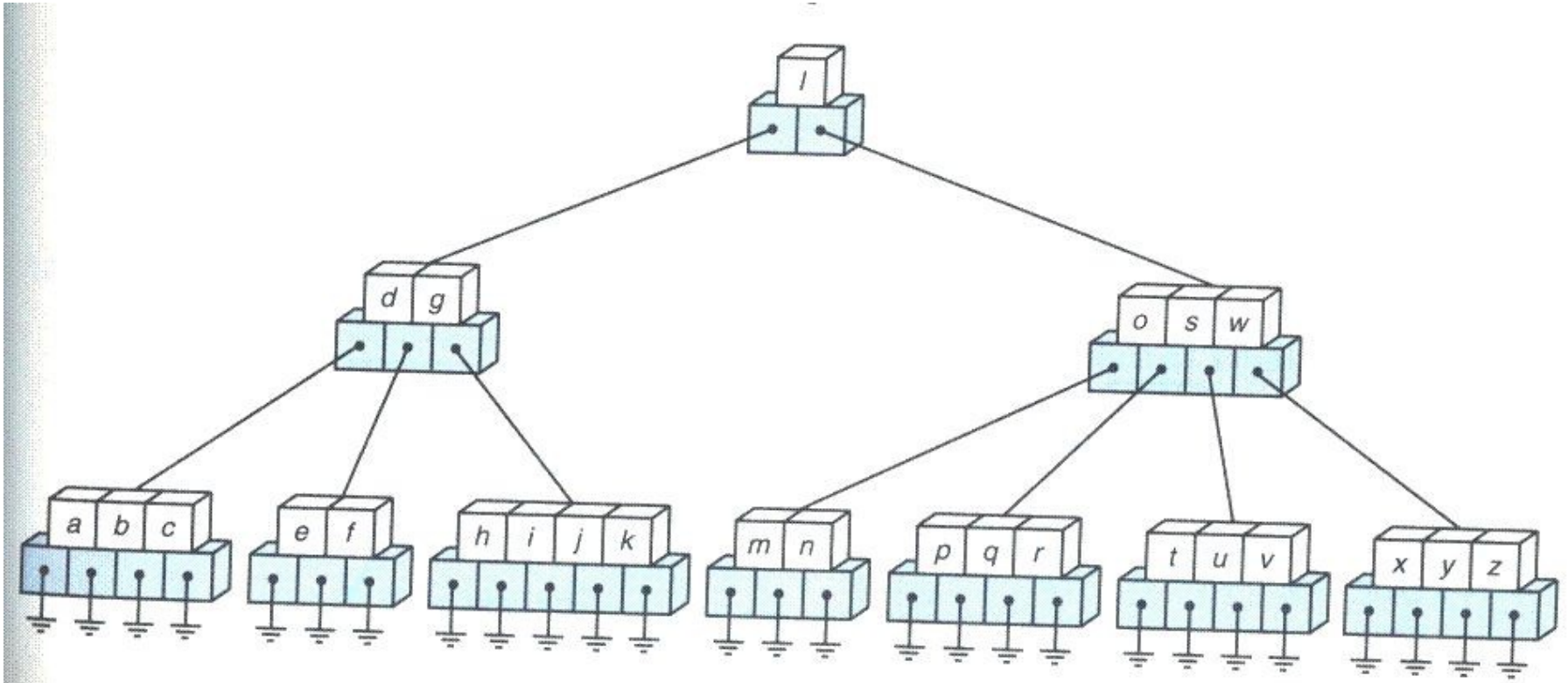


Figure 10.6. A B-tree of order 5

ÁRVORES B – Definição

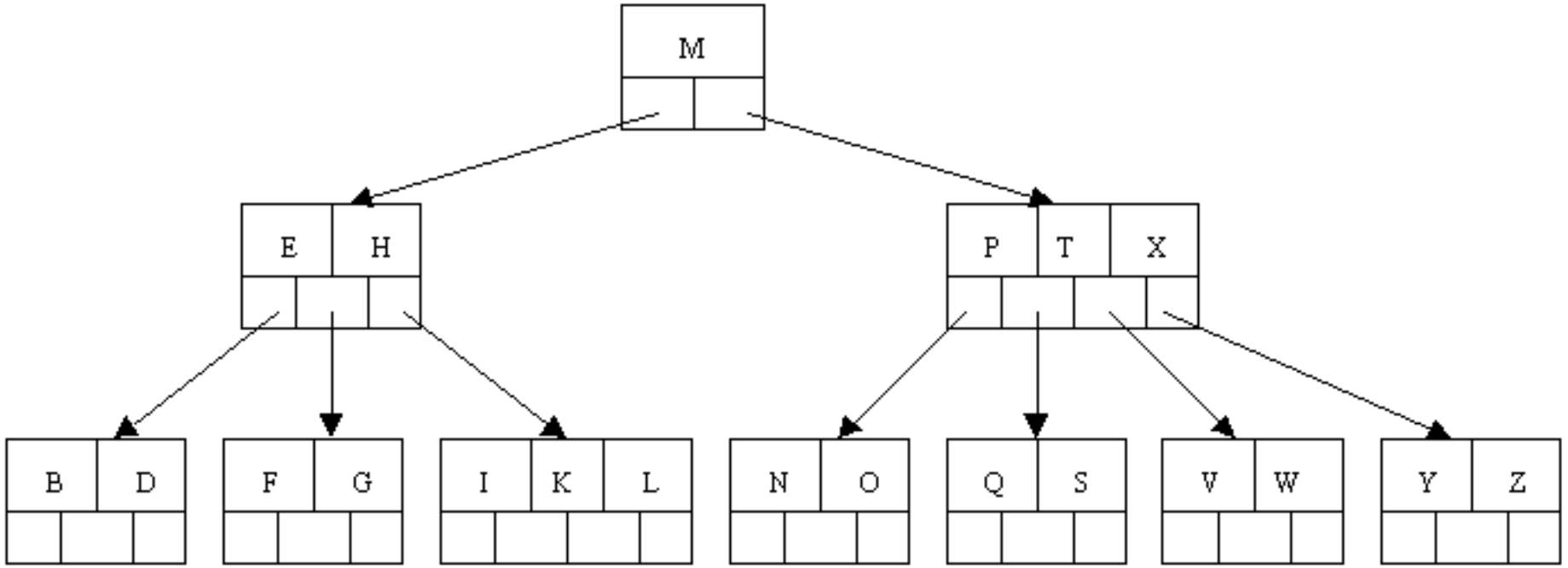
Um nó de uma árvore B é chamado PÁGINA e armazena m chaves.

Uma página não folha tem (m + 1) filhos:

raiz : $1 \leq m \leq 2d$
outras : $d \leq m \leq 2d$

Em cada página P as chaves estão ordenadas $s_1 \leq s_2 \leq ... \leq s_m$ e contém (m + 1) ponteiros $p_0, p_1, ..., p_m$ para os filhos de P (nas folhas estes ponteiros são NULL).

Árvore B de ordem d=2



ÁRVORES B – Definição

Critério de ordenação:

- Para qualquer chave y da página apontada por $p_{0,}$,

$y < s_1$

- Para qualquer chave y da página apontada por p_k, $1 \leq k \leq (m - 1)$,

$s_k < y < s_{(k + 1)}$

- Para qualquer chave y da página apontada por p_m,

$y > s_m$

ÁRVORES B

Podemos ver o poder da árvore B quando comparada a outros tipos de árvores balanceadas com altura $O(\log(n))$.

No caso da árvore B a base do logaritmo é proporcional ao fator de ramificação.

Por exemplo. Seja $d = 1000$ e $n = 1$ milhão de registros. Então, são necessários aproximadamente $\log_{1000}(10^6) = 3$ acessos ao disco.

ÁRVORES B

O fator de ramificação pode variar bastante, entre 2 a 2048, dependendo de características do hardware

O tempo de execução dos algoritmos que implementam uma árvore B é fortemente influenciado pelo tempo de leitura e escrita em disco

O fator de ramificação é inversamente proporcional a altura da árvore.

ÁRVORES B - Limites Superiores e Inferiores

Seja T uma árvore B de ordem d=9.

Suponha T armazene n=199 chaves.

Qual a altura mínima de T ?

…

Qual a altura máxima de T ?

…

ÁRVORES B - Limites Superiores e Inferiores

Cotas extremas para a altura:

?? ≤ h ≤ ??

Número mínimo de páginas ?

Número máximo de páginas ?

Limites superior para a altura h

Número mínimo de páginas ?

Número mínimo de páginas: ocorre quando a árvore possui uma página-raiz seguida do número mínimo de páginas para cada nível.

Seja d um número natural.

a. A raiz é uma folha ou tem no mínimo 2 filhos;

b. Cada nó diferente da raiz e das folhas possui no mínimo (d + 1) filhos;

Limites superior para a altura h

Número mínimo de páginas ?

$$P_{min} = 1 + 2[(d+1)^0 + (d+1)^1 + \ldots (d+1)^{h-2}]$$

$$= 1 + 2/d\ [\ (d+1)^{h-1} - 1\ ]$$

Número mínimo de chaves ?

Limites superior para a altura h

Número mínimo de chaves ?

$$P_{min} = 1 + 2/d \, [\, (d+1)^{h-1} - 1 \,]$$

$$n_{min} = 1 + d \, (\, 2/d \, [\, (d+1)^{h-1} - 1 \,] \,)$$

$$= \; 2(d+1)^{h-1} - 1$$

Portanto, $h \leq 1 + \log_{d+1}(n + 1)/2$

Limites inferior para a altura h

Número máximo de páginas ?

Número máximo de páginas: ocorre quando todas as páginas apontam para (2d + 1) páginas (filhos).

Seja d um número natural.

c. Cada nó tem no máximo (2d + 1) filhos;

Limites inferior para a altura h

Número máximo de páginas ?

$$P_{max} = (2d+1)^0 + (2d+1)^1 + \ldots + (2d+1)^{h-1}$$

$$= 1/2d \, [\, (2d+1)^h - 1 \,]$$

Número máximo de chaves ?

Limites inferior para a altura h

Número máximo de chaves ?

$$P_{max} = 1/2d\ [\ (2d+1)^h - 1\ ]$$

$$n_{max} = 2d\ (\ \ [(2d+1)^h - 1\ ]\ /2d\ )$$

$$= \ (2d+1)^h - 1$$

Portanto, $h \geq \log_{2d+1}(n + 1)$

ÁRVORES B - Limites Superiores e Inferiores

Limites quanto ao número de páginas e elementos

Número mínimo de páginas: ocorre quando a árvore possui uma página-raiz seguida do número mínimo de páginas para cada nível.

Número máximo de páginas: todas as páginas apontam para (2d + 1) páginas (filhos).

$$\log_{2d+1}(n+1) \leq h \leq 1 + \log_{d+1}(n+1/2)$$

ÁRVORES B - Limites Superiores e Inferiores

Seja T uma árvore B de ordem d=9. Suponha T armazene n=199 chaves.

Qual a altura mínima de T ? $h \geq \log_{2d+1}(n + 1)$

Qual a altura máxima de T ? $h \leq 1 + \log_{d+1}(n + 1)/2$

ÁRVORES B - Limites Superiores e Inferiores

Seja T uma árvore B de ordem d=9. Suponha T armazene n=199 chaves.

Qual a altura mínima de T ? $h \geq \log_{2d+1}(n + 1)$

$h \geq \log_{19}(200) \rightarrow h \geq 2$ (usando o teto)

Qual a altura máxima de T ? $h \leq 1 + \log_{d+1}(n + 1)/2$

$h \leq 1 + \log_{10}(100) = 3$

ÁRVORES B - Limites Superiores e Inferiores

Seja T uma árvore B de ordem d=9.

a. Suponha T armazene n=1999 chaves.

Qual a altura mínima de T ?
Qual a altura máxima de T ?

b. Suponha T armazene n=1999999 chaves.

Qual a altura mínima de T ?
Qual a altura máxima de T ?

c. Pra qual valor de n a altura máxima dessa árvore será 10?

ÁRVORES B - Limites Superiores e Inferiores

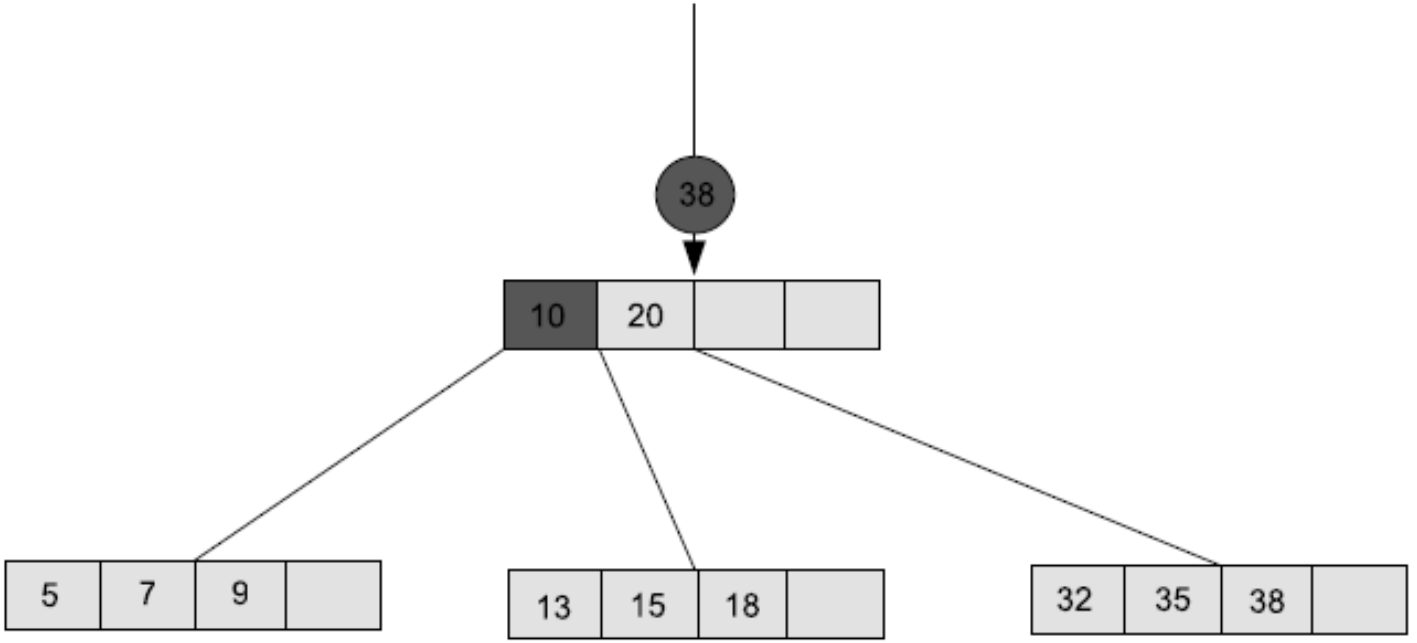
Seja T uma árvore B de altura $h \geq 10$.

Quais os valores possíveis para d, em cada hipótese:

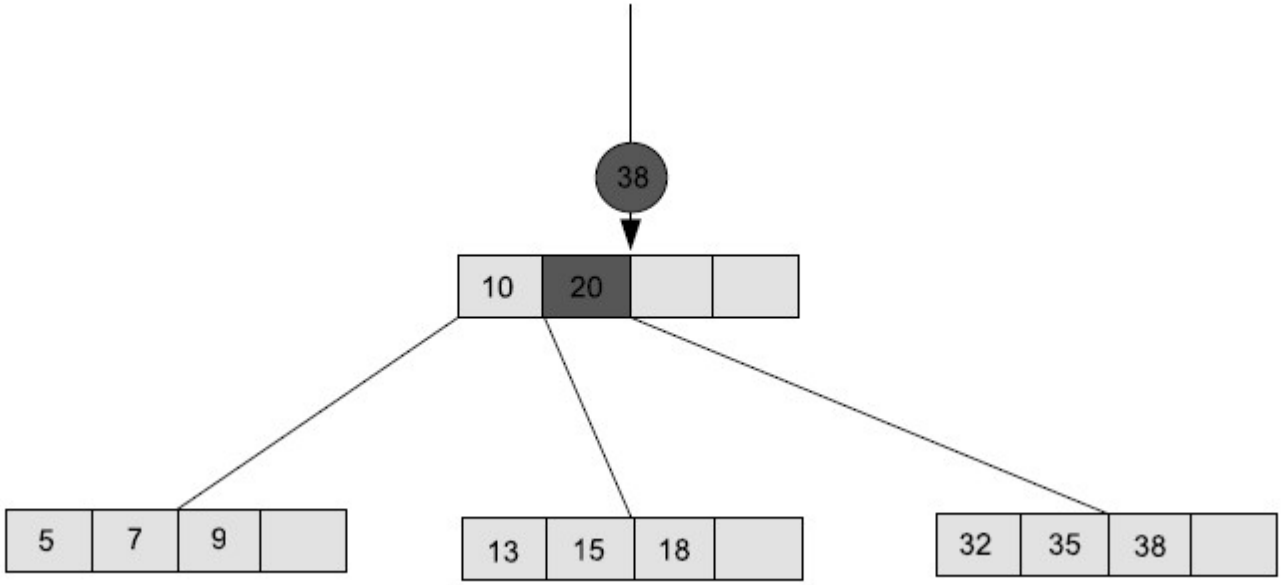
a. Suponha T armazene n=1000000 chaves.

b. Suponha T armazene n=25937424600 chaves.

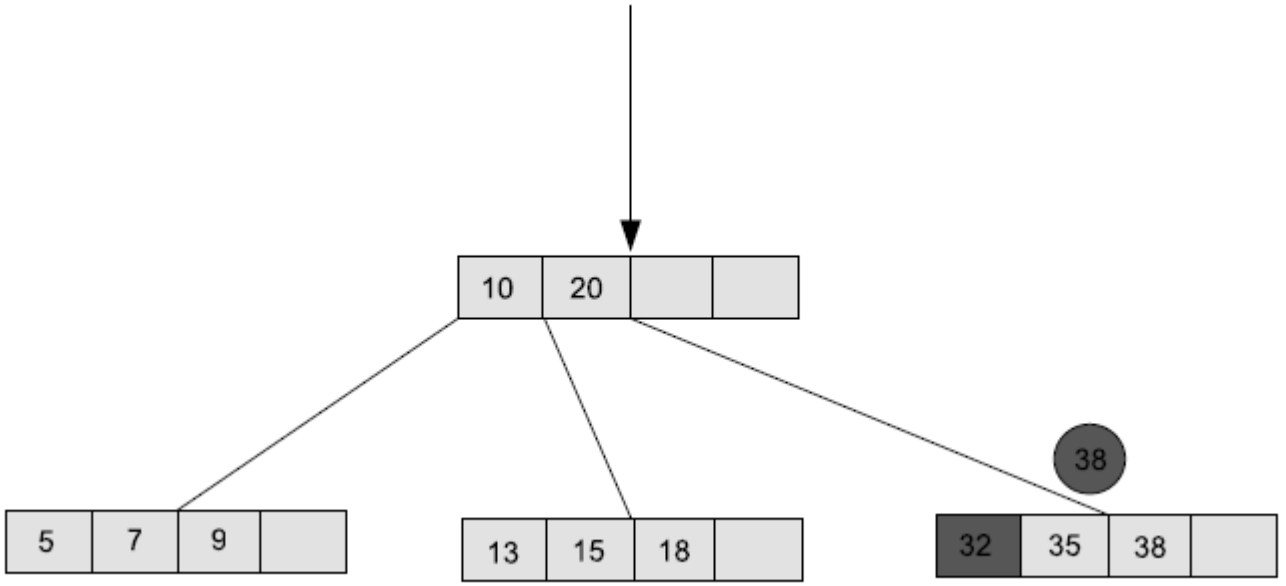
Exemplo de Busca (busca pelo elemento 38)



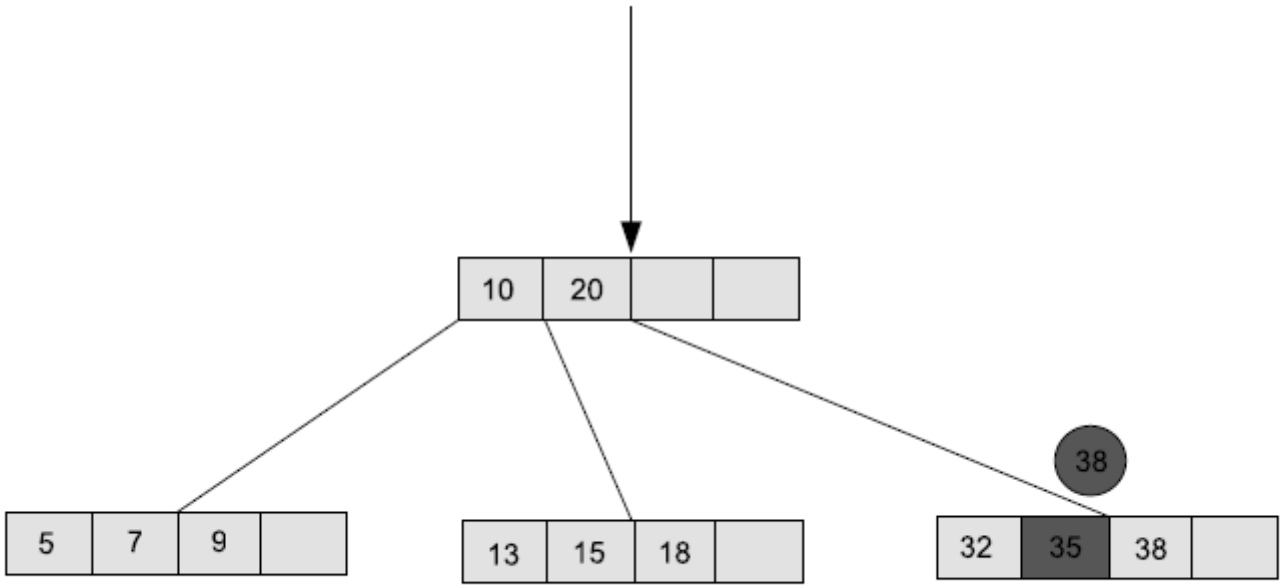
Exemplo de Busca (busca pelo elemento 38)



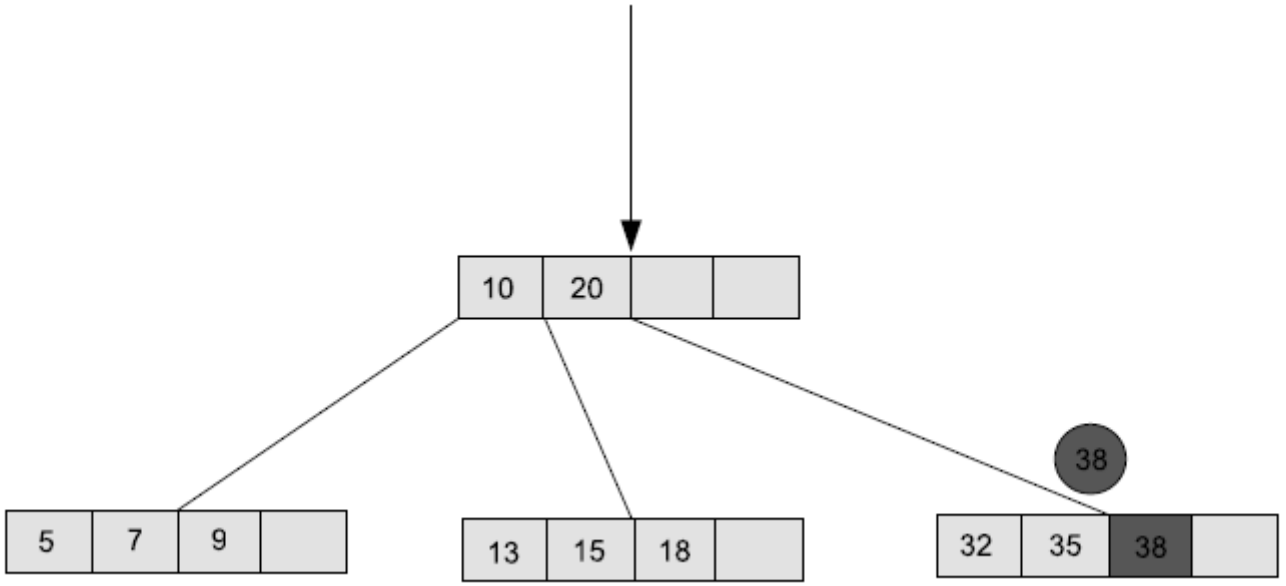
Exemplo de Busca (busca pelo elemento 38)



Exemplo de Busca (busca pelo elemento 38)



Exemplo de Busca (busca pelo elemento 38)



ÁRVORES B – Atributos da página

m: número de chaves

s[2d]: vetor de chaves // 1..2d

p[2d+1]: vetor de ponteiros (filhos) // 0..2d

ÁRVORES B – Algoritmo Busca

Entrada:

x: chave procurada.

Saída:

pt: aponta para a página onde a chave foi encontrada, ou deve ser inserida.

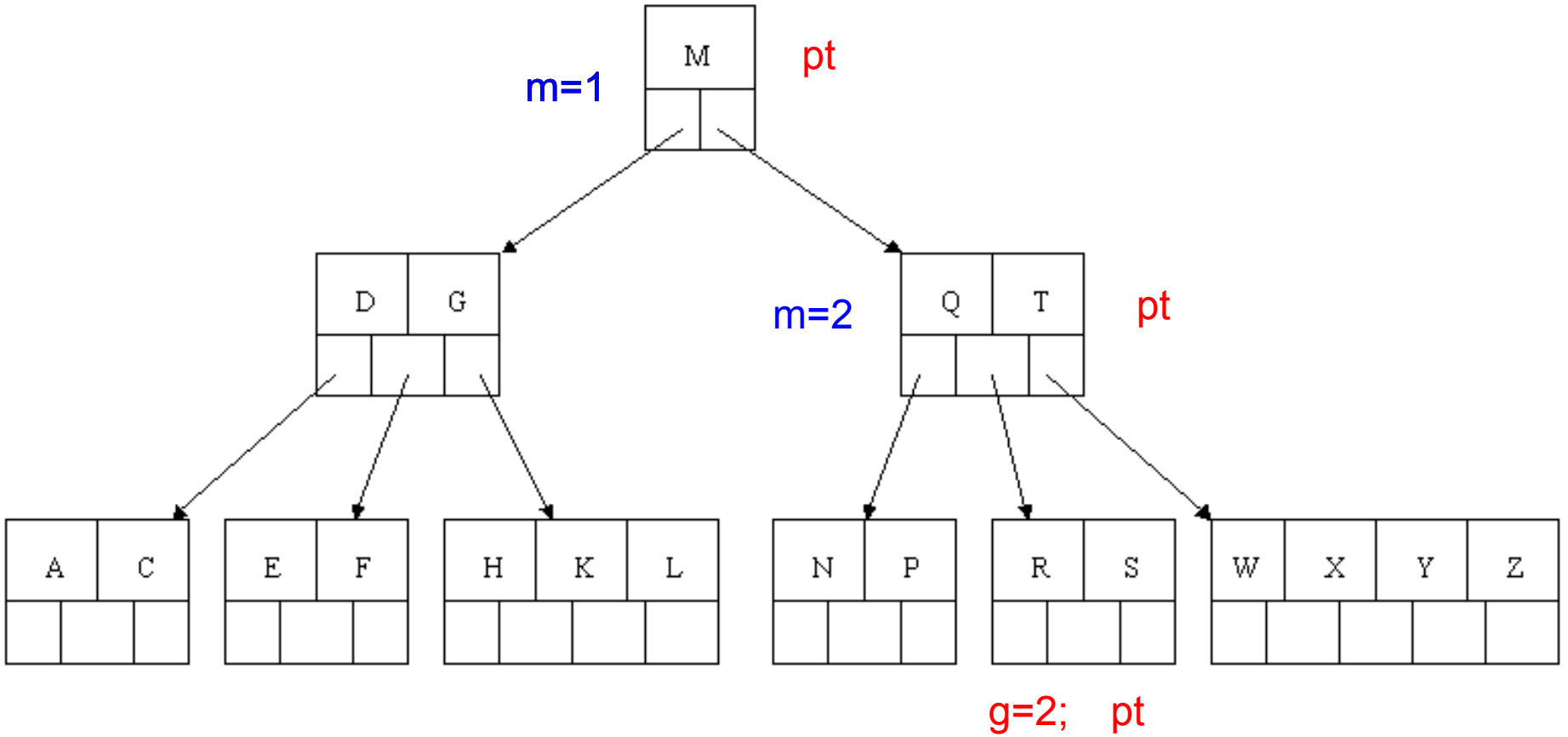
f: indica se a chave foi encontrada (f = 1) ou não (f = 0).

g: g-ésima posição da página apontada por pt, onde a chave foi encontrada; Se a chave não foi encontrada, pt aponta a última página examinada (uma folha) e g informa a posição, nesta página, onde x será incluída.

```
BuscaB ( x, pt, f, g)

    p = ptraiz; pt = NULL; f = 0;

    enquanto p ≠ NULL faça

        i = 1;  g = 1;  pt = p;
        enquanto i ≤ p→m faça

            se x > p→s[i] então
                  i = i + 1; g = i
            senão se x = = p→s[i] então
                        p = NULL; f = 1;        // chave encontrada
                    senão                       // s[i] > x
                        p = p→p[i-1];           // mudança de página
                    i = (p→m) + 2;

        se i = = (p→m) + 1 então p = p→p[m];
```

C N G A H E K Q M F W L T Z D P R X Y S

x='S'



ÁRVORES B - Inserção

Passo 1: Executar o procedimento de Busca

Passo 2: Se a chave já existe (f=1) então

Inserção Inválida

Passo 3: Senão inserir a chave x na página e na posição adequada

(g-ésima posição da página apontada por pt)

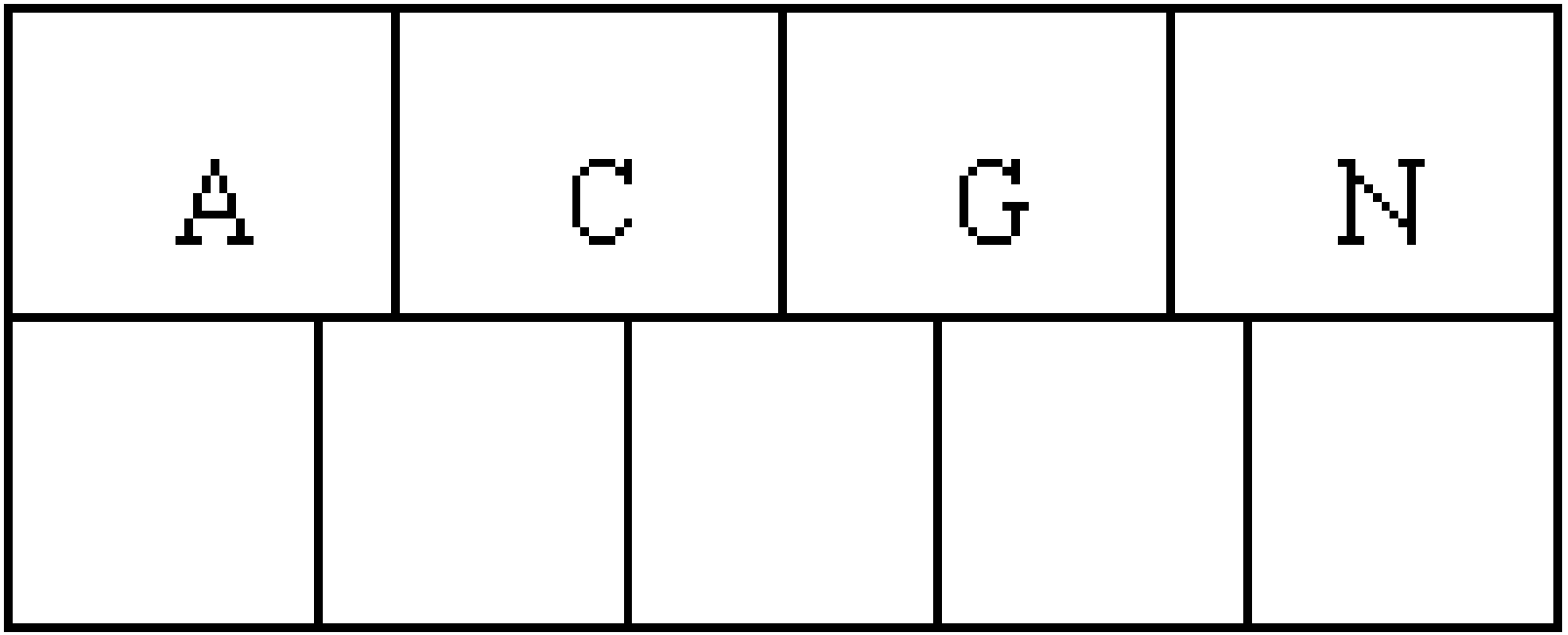
ÁRVORES B - Inserção

Seja T uma árvore B de ordem d=2

Inserir as seguintes chaves partindo de T vazia:

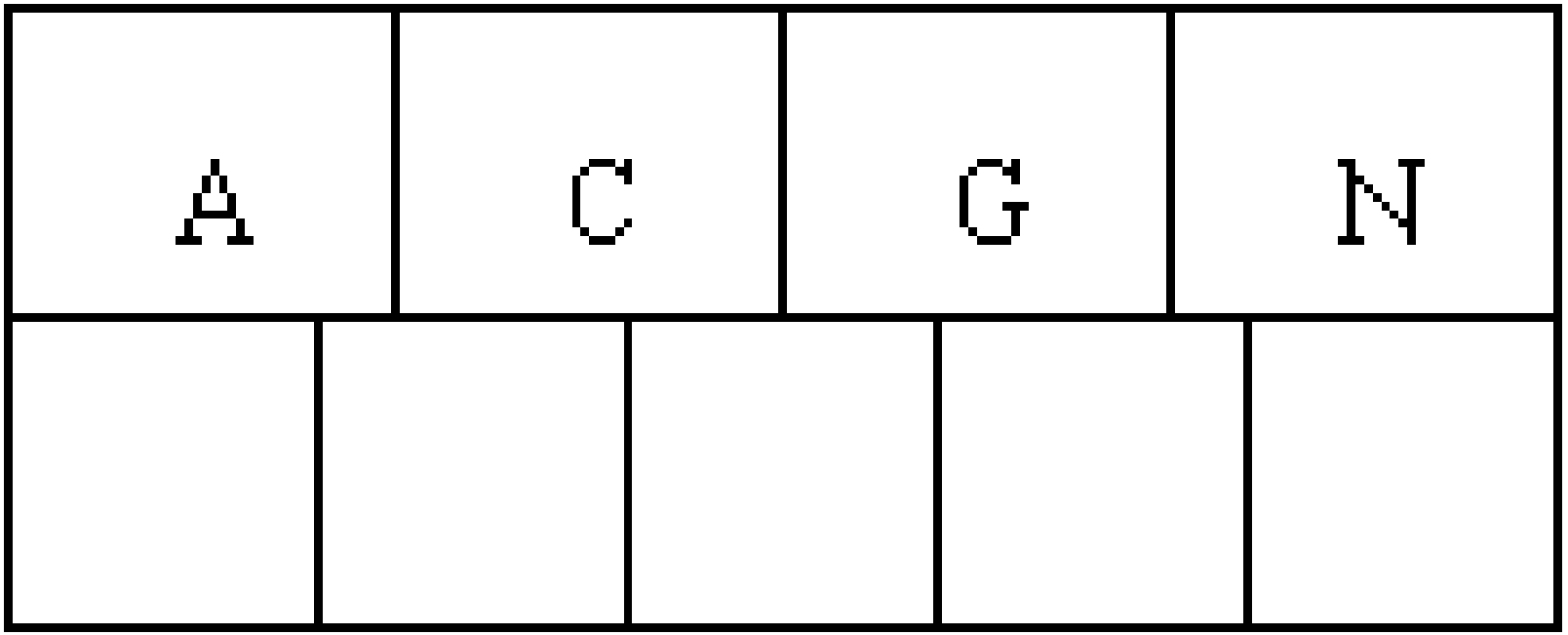
C N G A H E K Q M F W L T Z D P R X Y S

C N G A H E K Q M F W L T Z D P R X Y S



C N G A H E K Q M F W L T Z D P R X Y S

Inserir H !



ÁRVORES B - Inserção

Problema: A folha já possuía 2d chaves, com a inserção obtemos 2d + 1 chaves (IMPOSSÍVEL POR DEFINIÇÃO)

Solução: Cisão de página

ÁRVORES B – Inserção / Cisão

Seja P a página onde é feita uma inserção, resultando em 2d+1 chaves .

Disposição em P (apontada por pt)

$p_0, (s_1,p_1),(s_2,p_2),\ldots,(s_d,p_d),$

(s_{d+1},p_{d+1})

$(s_{d+2},p_{d+2}),\ldots,(s_{2d+1},p_{2d+1})$

ÁRVORES B – Inserção / Cisão

1. Em **P** permanecem d chaves

2. Criamos uma nova página **Q** apontada por pt1, a qual armazenará também d chaves
   $\mathbf{P_{d+1,}(s_{d+2},p_{d+2}),...,(s_{2d+1},p_{2d+1})}$

3. Acrescentamos no nó **W**, pai de **P**, a entrada $\mathbf{(s_{d+1},pt1)}$, na ordem adequada.

ÁRVORES B – Inserção / Cisão

A disposição de entradas em ***P*** após o processo de cisão é:

$$p_0, (s_1,p_1),(s_2,p_2),\ldots,(s_d,p_d),$$

O novo nó ***Q***, apontado por pt1, apresenta a seguinte disposição de entradas:

$$P_{d+1,}(s_{d+2},p_{d+2}),\ldots,(s_{2d+1},p_{2d+1})$$

O nó ***W***, agora também pai de ***Q***, possui a nova entrada

$$(s_{d+1},pt1)$$

ÁRVORES B – Inserção / Cisão

Situação Possível:

W também pode ultrapassar o limite de chaves

Solução:

Propagar a Cisão até (possivelmente) a raiz

ÁRVORES B - Inserção

1. Aplicar o procedimento de Busca, verificando a validade da inserção.
2. Se a inserção é válida, incluir a chave na folha **F** adequada.
3. Verificar se a página **F** necessita de cisão.
   Em caso positivo,
   propagar a cisão bottom-up enquanto necessário.

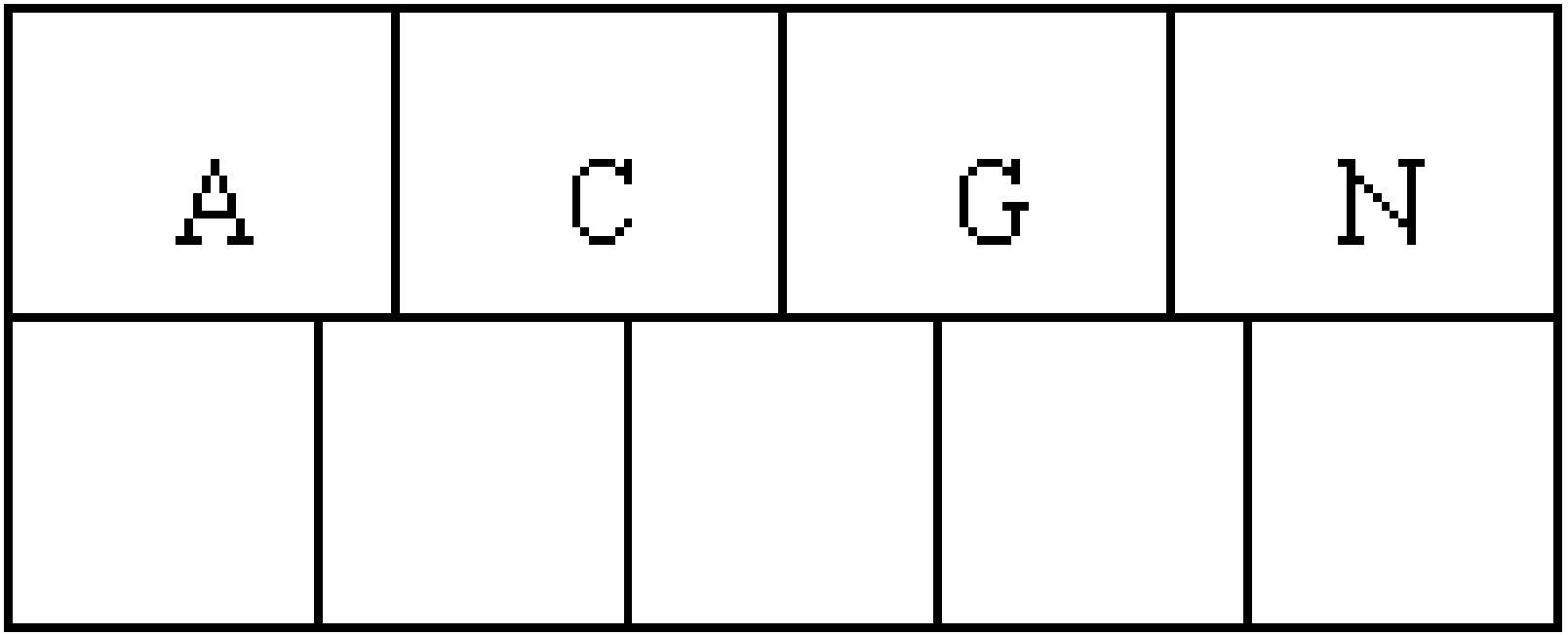
ÁRVORES B - Inserção

Seja T uma árvore B de ordem d=2

Inserir as seguintes chaves partindo de T vazia:

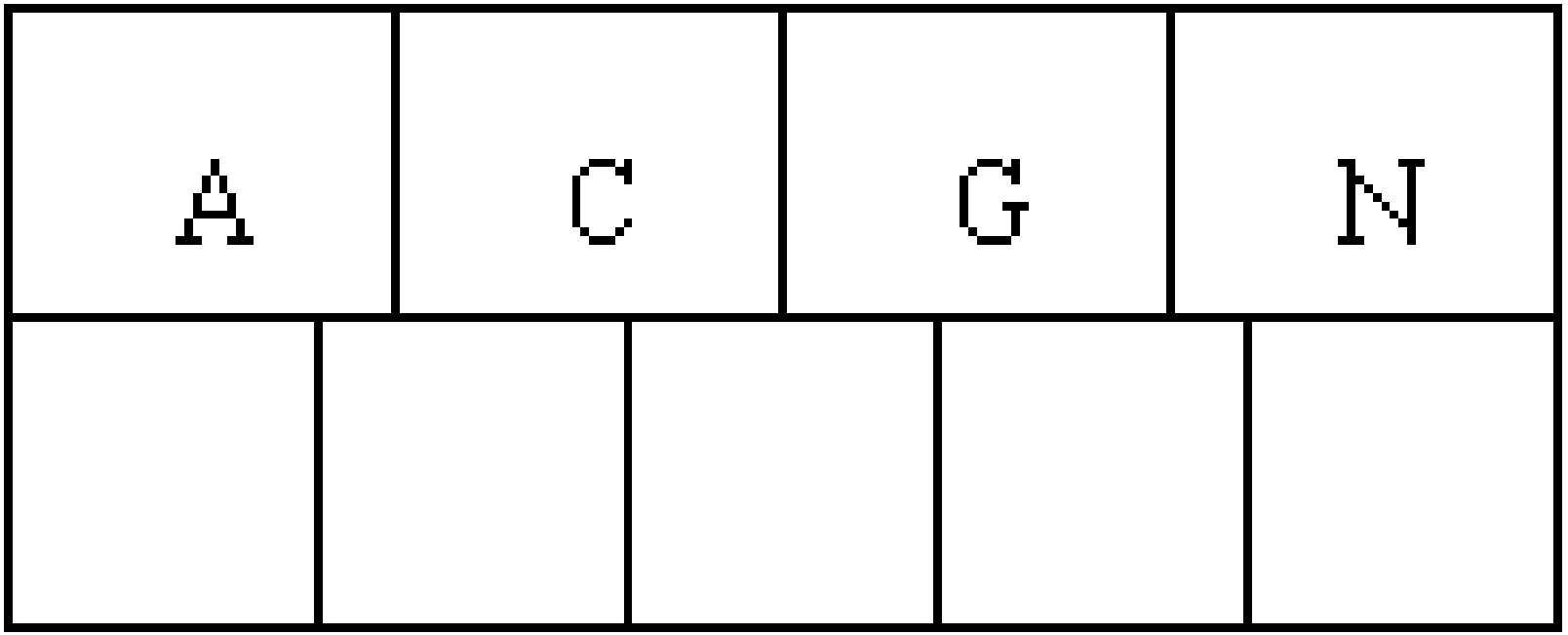
C N G A H E K Q M F W L T Z D P R X Y S

C N G A H E K Q M F W L T Z D P R X Y S

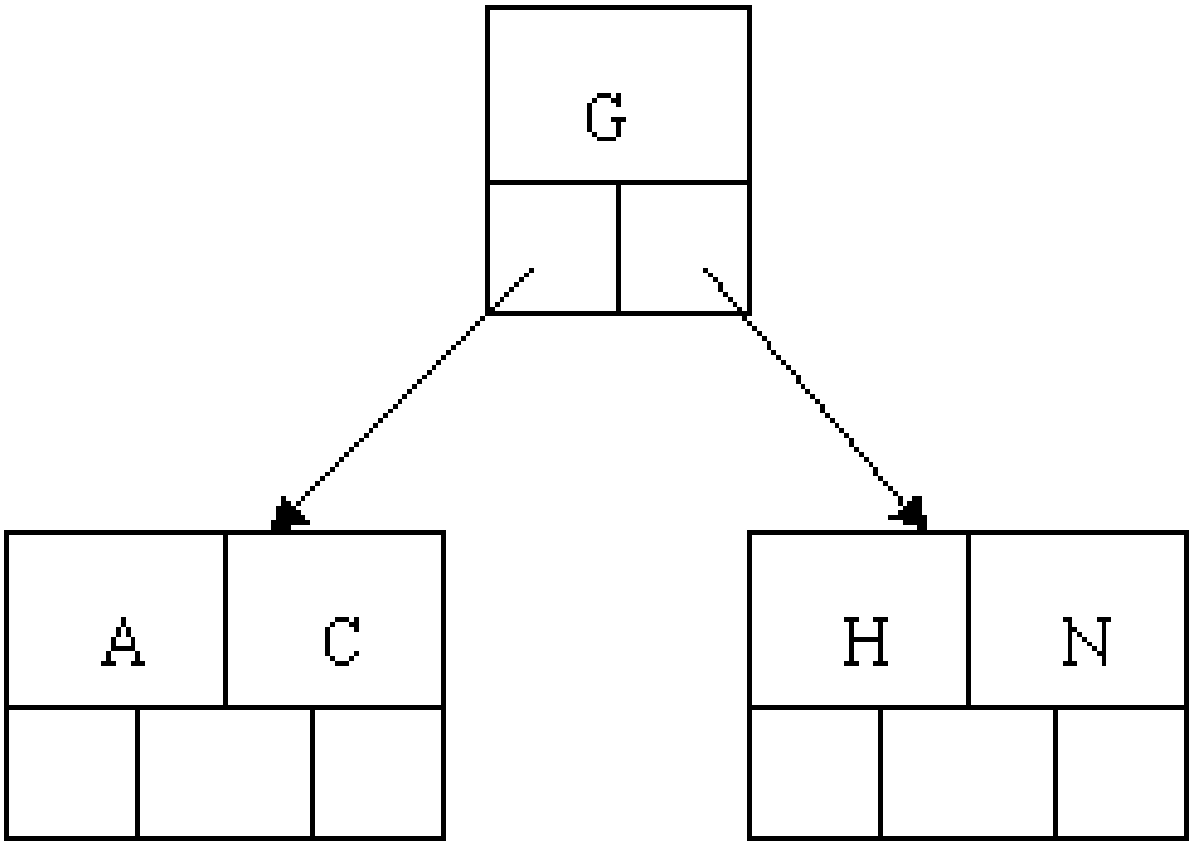


C N G A H E K Q M F W L T Z D P R X Y S

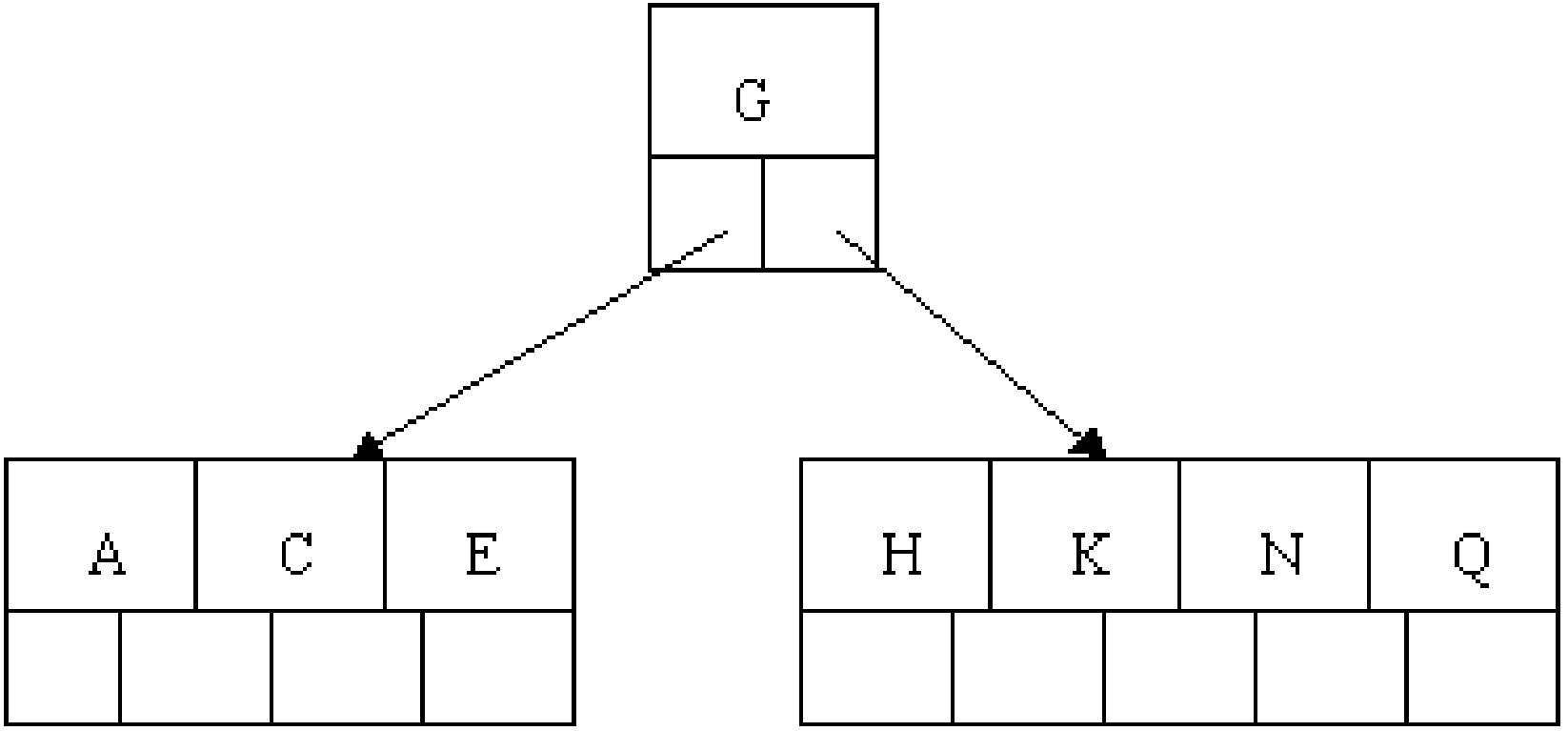
Inserir H !



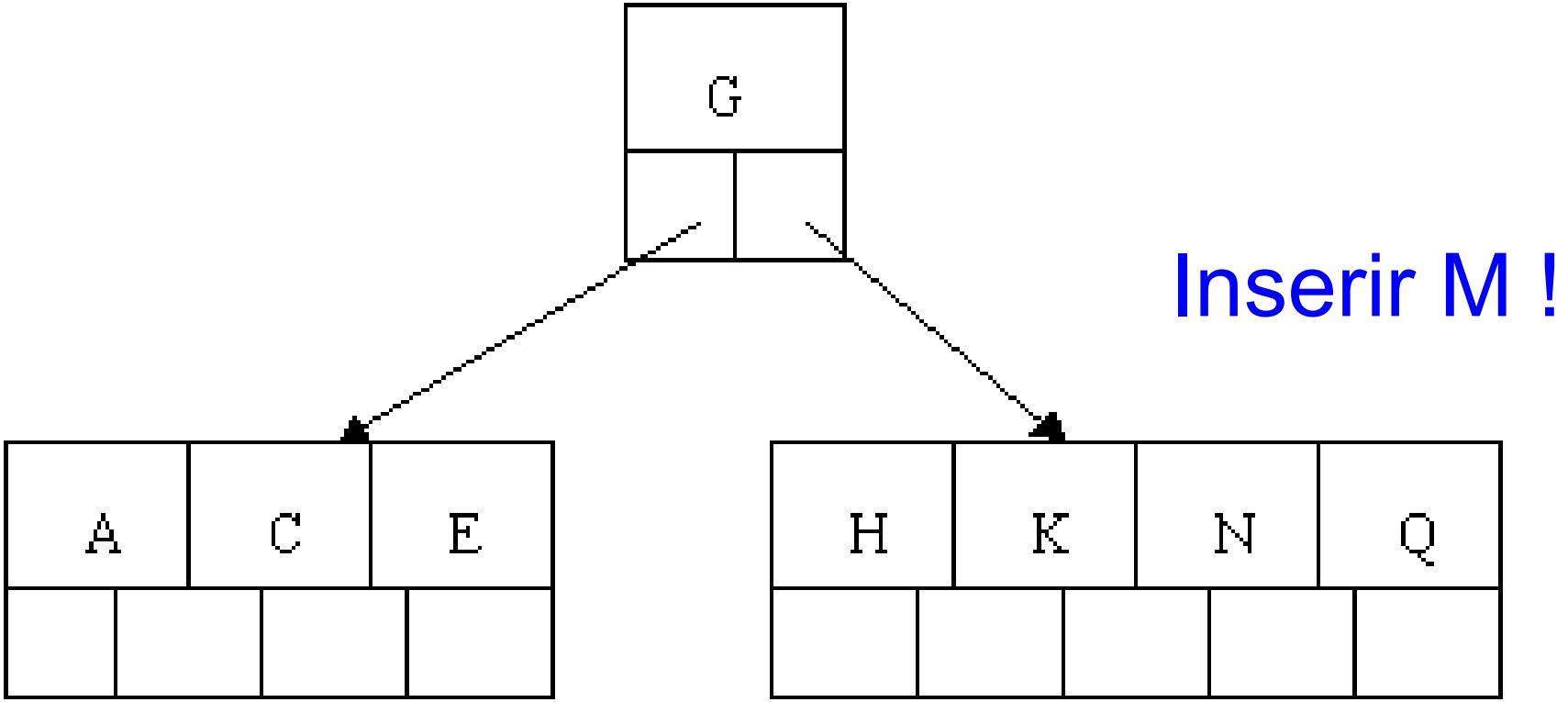
C N G A H E K Q M F W L T Z D P R X Y S



C N G A H E K Q M F W L T Z D P R X Y S

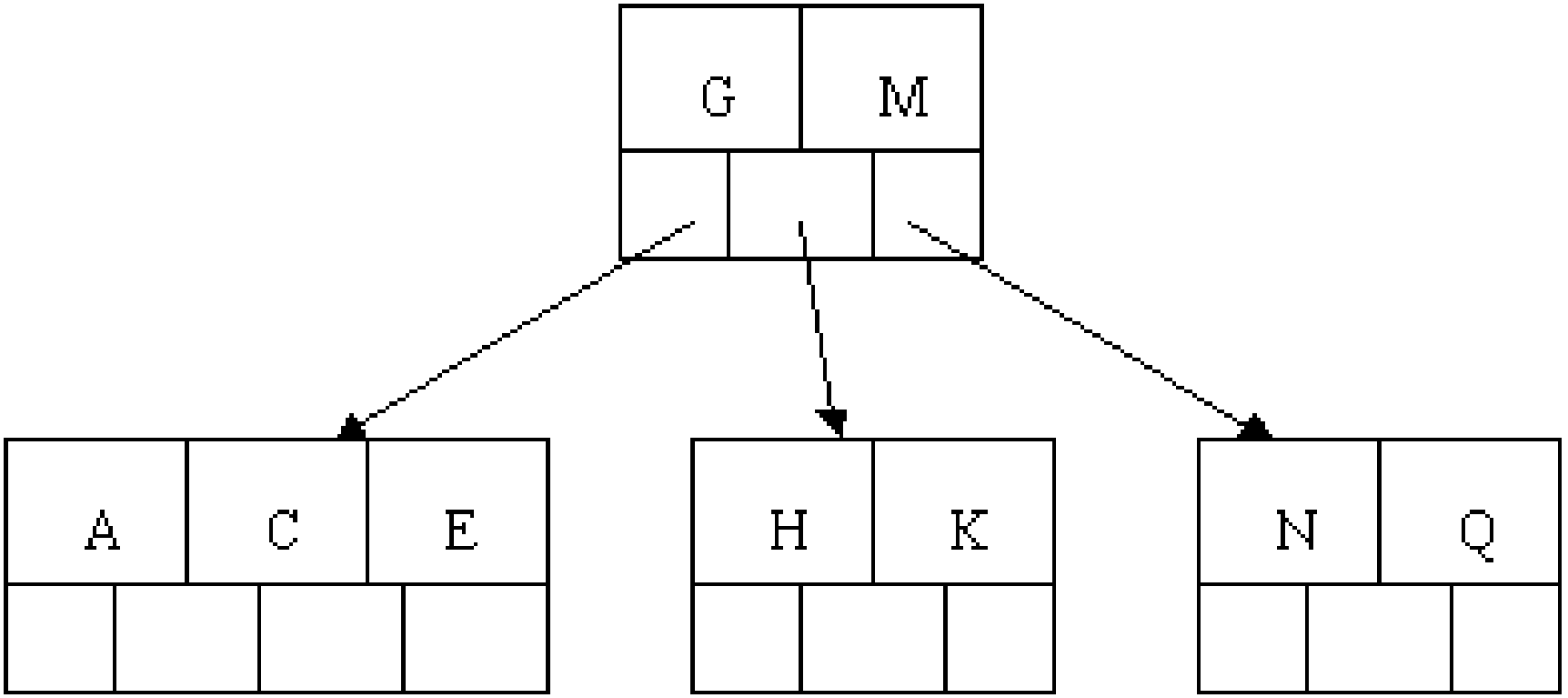


C N G A H E K Q M F W L T Z D P R X Y S

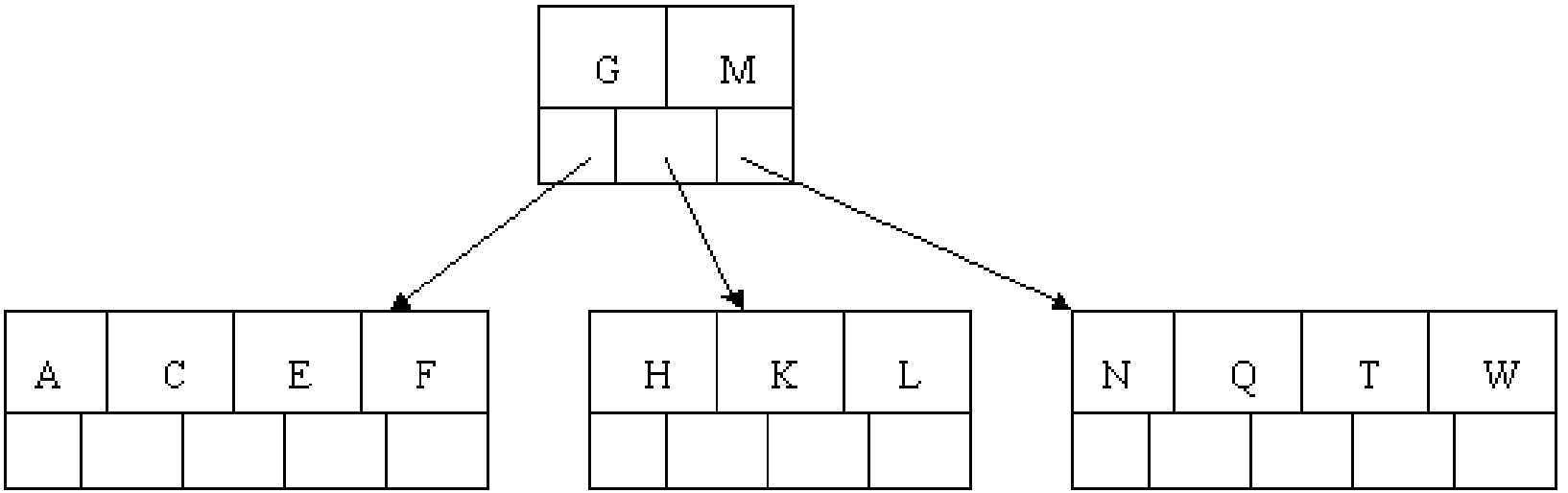


Inserir M !

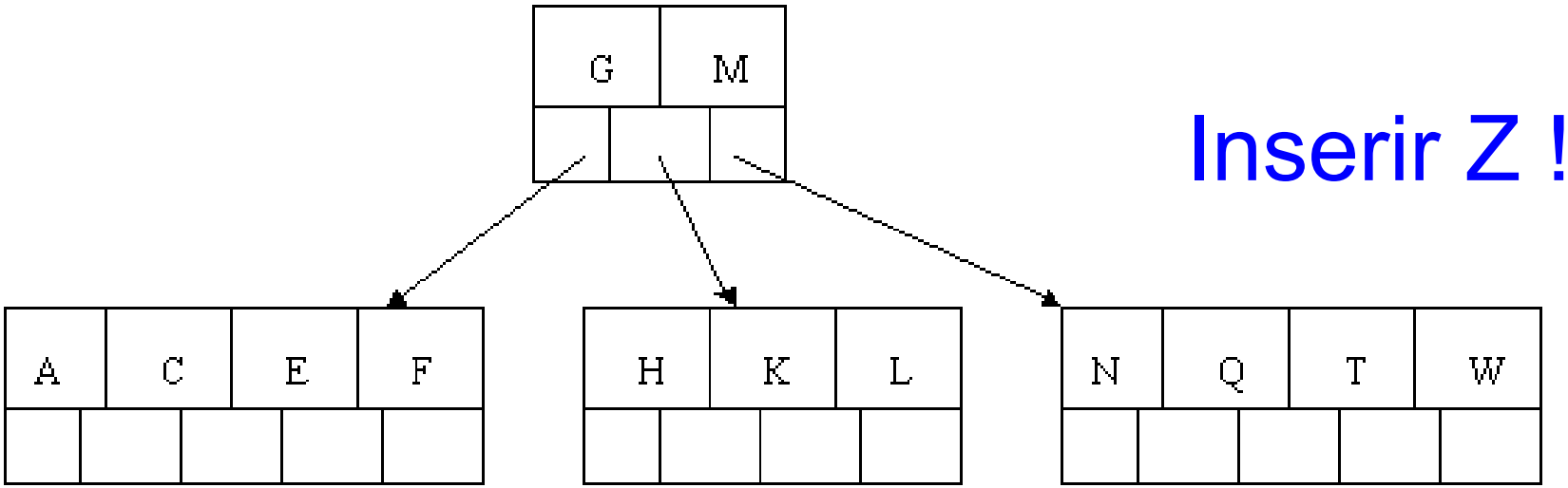
C N G A H E K Q M F W L T Z D P R X Y S



C N G A H E K Q M F W L T Z D P R X Y S

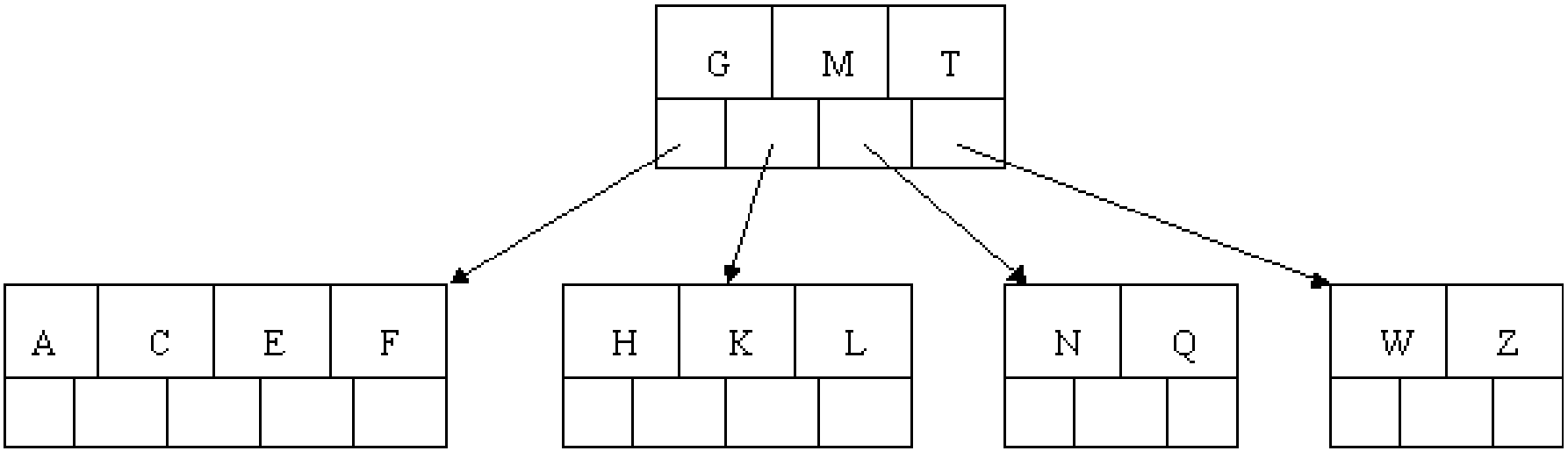


C N G A H E K Q M F W L T Z D P R X Y S



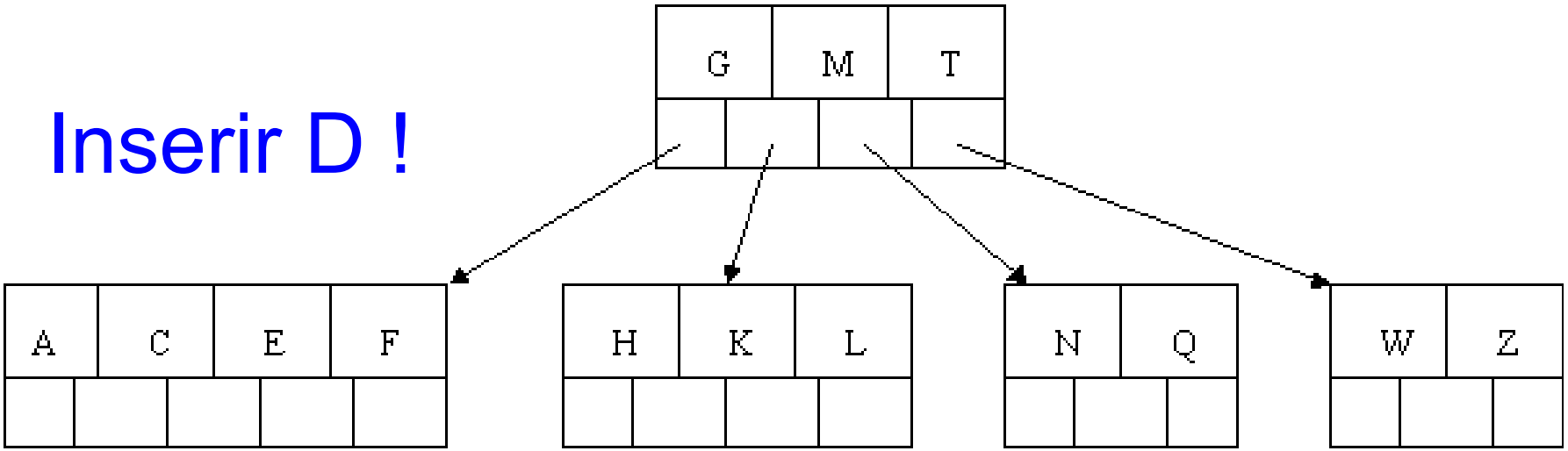
Inserir Z !

C N G A H E K Q M F W L T Z D P R X Y S

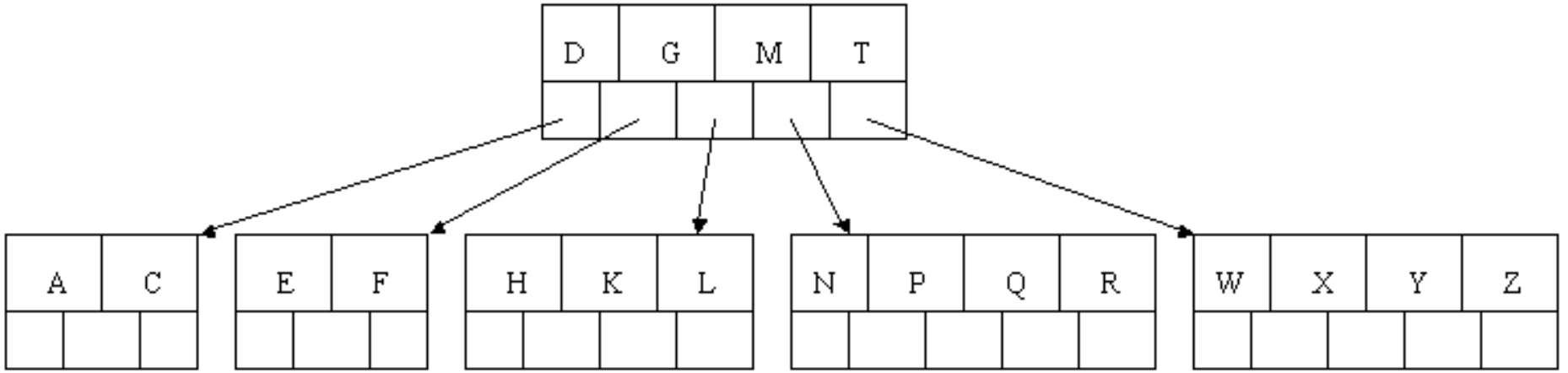


C N G A H E K Q M F W L T Z D P R X Y S

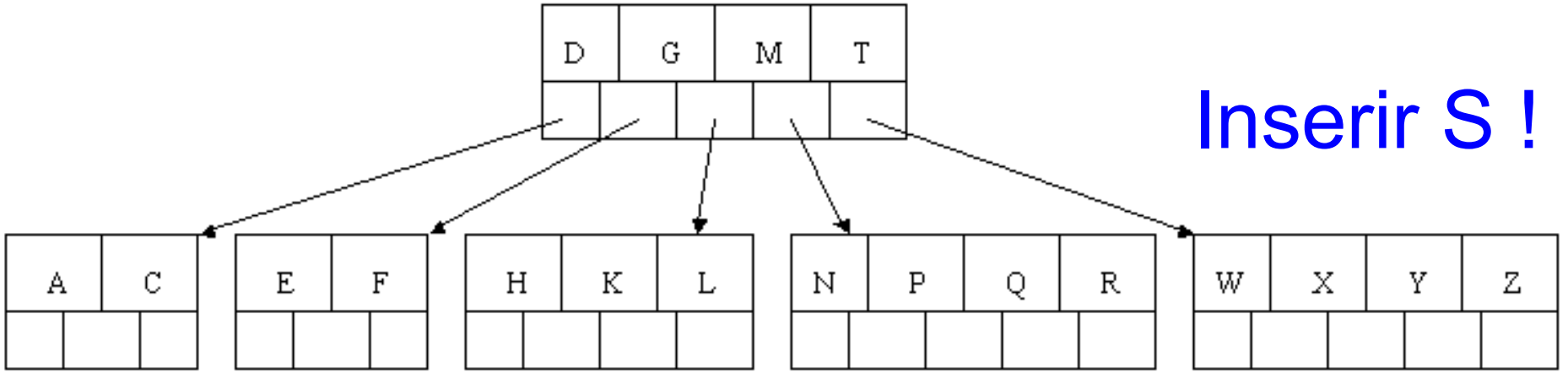
Inserir D !



C N G A H E K Q M F W L T Z D P R X Y S

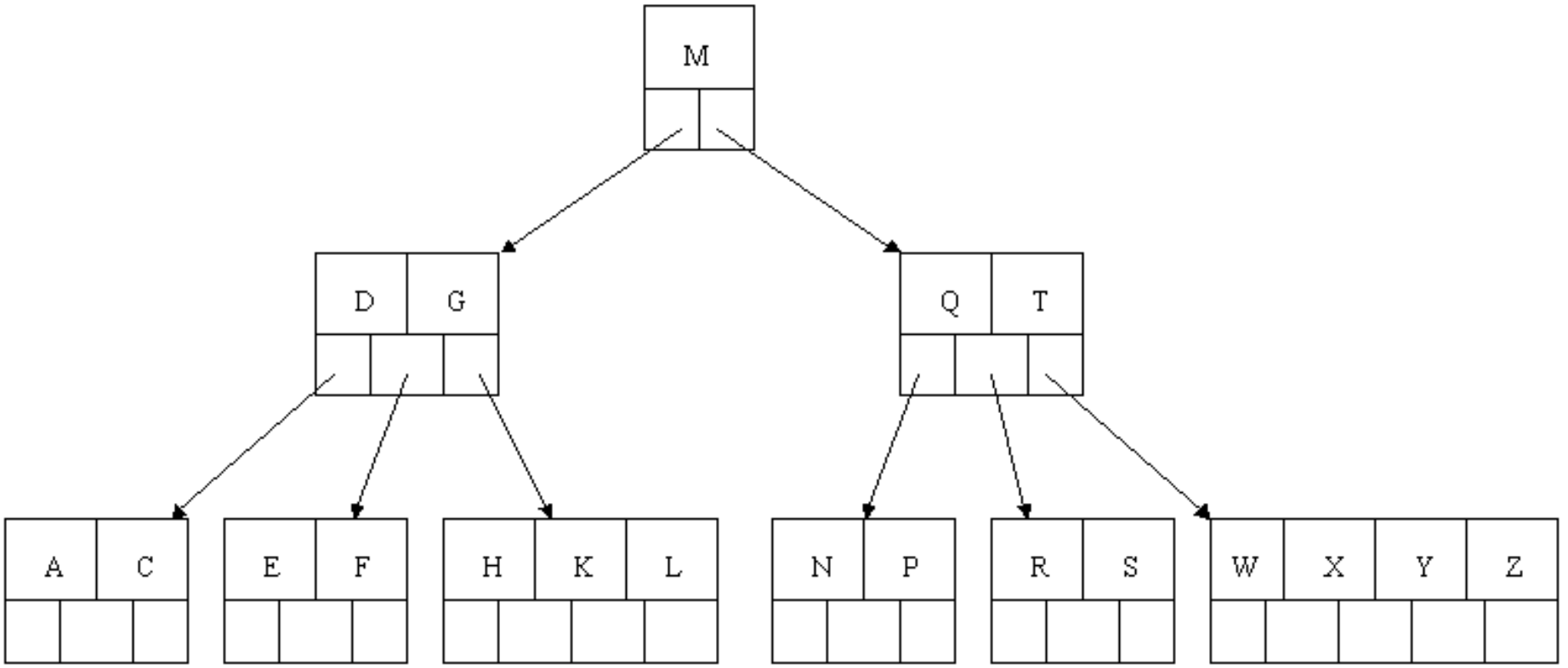


C N G A H E K Q M F W L T Z D P R X Y S



Inserir S !

C N G A H E K Q M F W L T Z D P R X Y S



ÁRVORES B - Remoção

Seja x a chave a ser removida. Temos 2 casos para considerar:

1. x se encontra em uma folha.
   Solução: a entrada é simplesmente retirada;

2. x não se encontra em uma folha
   Solução: x é substituída pelo sucessor (que está, necessariamente, numa folha)

Logo, a análise da remoção pode se restringir ao caso em que esta operação é realizada em uma folha.

ÁRVORES B - Remoção

Lembrando:

Os limites quanto ao número de chaves na página são:

raiz: $1 \leq m \leq 2d$

outras: $d \leq m \leq 2d$

Problemas na remoção:

Quando a chave é retirada, podemos ter $m < d$.

Existem dois tratamentos

- **Concatenação**
- **Redistribuição.**

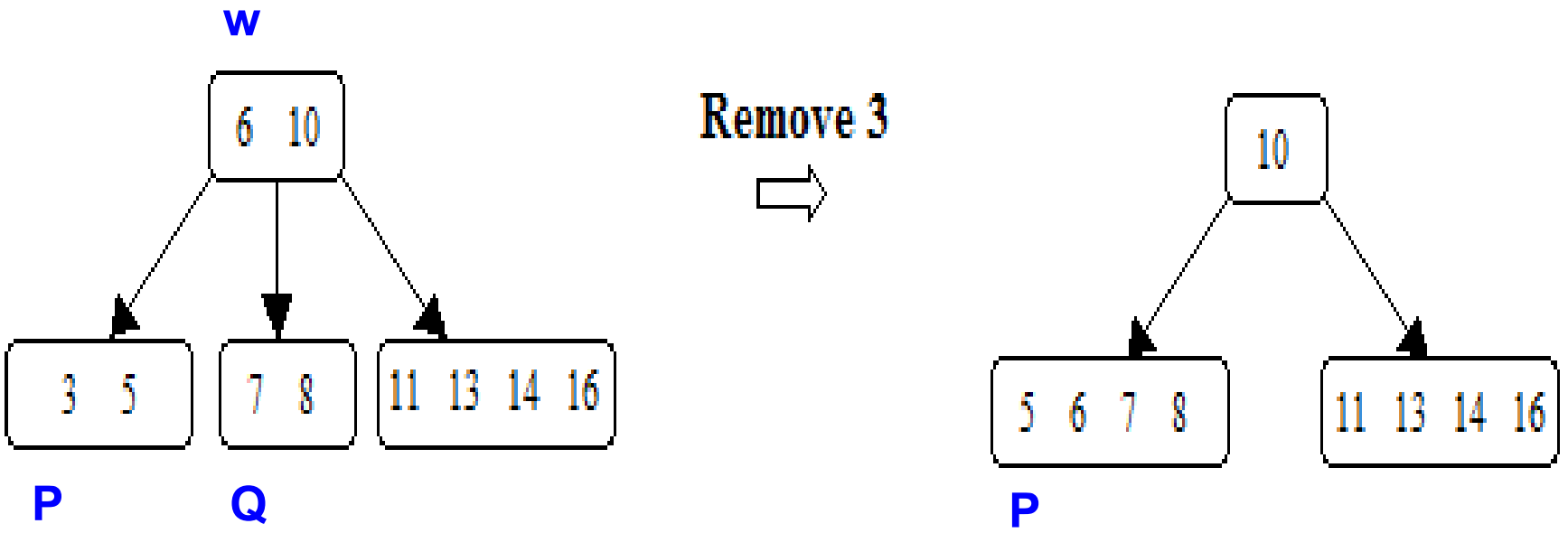
Árvores B - Concatenação

Duas páginas P e Q são chamadas *adjacentes* se têm o mesmo pai W e são apontadas por 2 ponteiros adjacentes em W.

- **Pré-condição:** Duas páginas podem ser concatenadas se são irmãs adjacentes e juntas possuem menos de 2d chaves.

Árvores B - Concatenação

Árvore B de ordem d=2



Árvores B - Concatenação

Agrupa as entradas de 2 páginas P e Q apenas em P;

Em W (pai) deixa de existir 1 entrada, exatamente aquela que existe entre os ponteiros adjacentes;

Esta chave passa a fazer parte da nova página concatenada P e seu ponteiro desaparece, visto que a página Q é devolvida;

Como uma chave foi retirada em W, esta página poderá ter menos do que d chaves e a concatenação pode ser propagada.

Árvores B - Concatenação

Antes da concatenação:

Página W: $...(y_{j-1},pt),(y_j,pt1),(y_{j+1},p_{j+1}),...$

Página P: $p_0,(s_1,p_1),...,(s_k,p_k)$

Página Q: $p'_0,(s'_1,p'_1),...,(s'_k,p'_k),...,(s'_m,p'_m)$

onde $k + m < 2d$

Árvores B - Concatenação

Depois da concatenação:

Página W:

$$\ldots(y_{j-1}, pt), (y_{j+1}, p_{j+1}), \ldots$$

Página P:

$$p_0, (s_1, p_1), \ldots, (s_k, p_k), (y_j, p'_0), (s'_1, p'_1), \ldots, (s'_k, p'_k), \ldots, (s'_m, p'_m)$$

Árvores B - Redistribuição

P e Q possuem em conjunto 2d ou mais chaves.

1. Concatenação de P e Q (P fica grande demais)

2. Efetua-se uma cisão (use a página Q)

OBS:

Redistribuição não é propagável, pois W (pai) mantém o mesmo número de chaves.

Árvores B - Algoritmo Remoção

1: Aplicar o procedimento de Busca, verificando a existência da chave x na árvore.

Seja P a página onde se encontra a chave.

2: Se P é uma folha, retirar a entrada correspondente a chave x. Caso contrário, buscar o sucessor.

Seja z essa chave e F a página onde z se encontra. Substitua a chave x por z.

Fazer P = F.

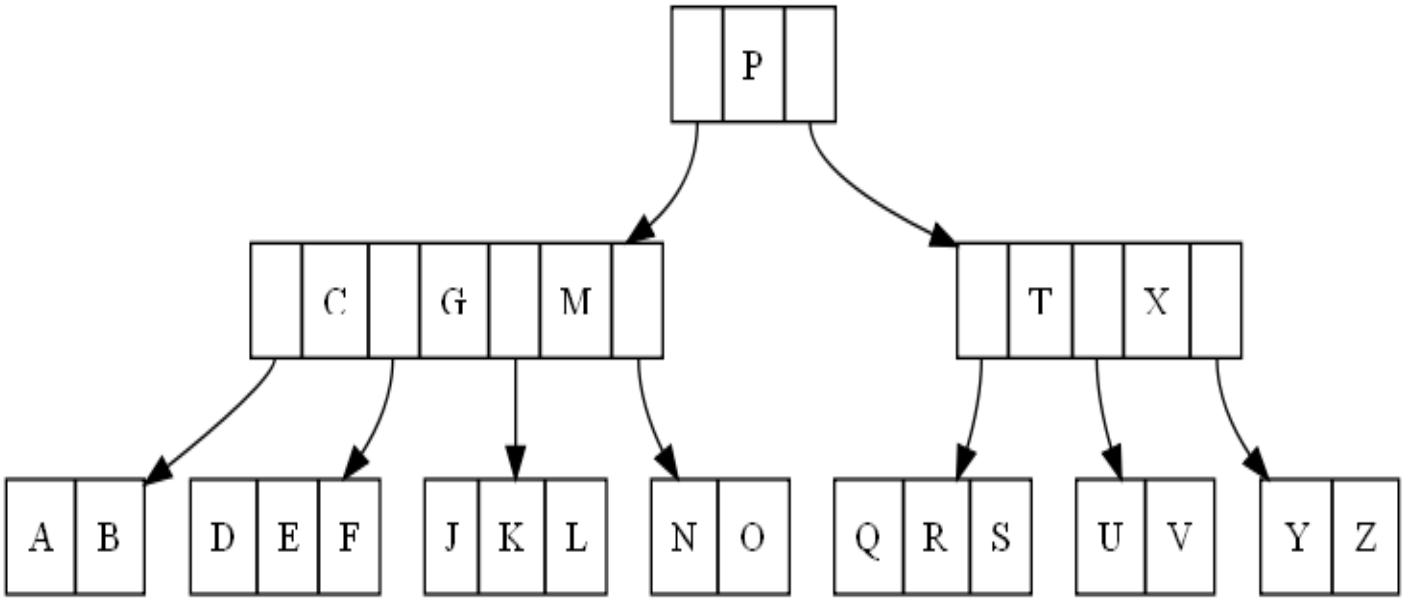
Árvores B - Algoritmo Remoção

3: Verificar se P contém d entradas. Caso contrário, executar a operação de concatenação ou redistribuição.

Árvores B - Remoção

Remover 'F'

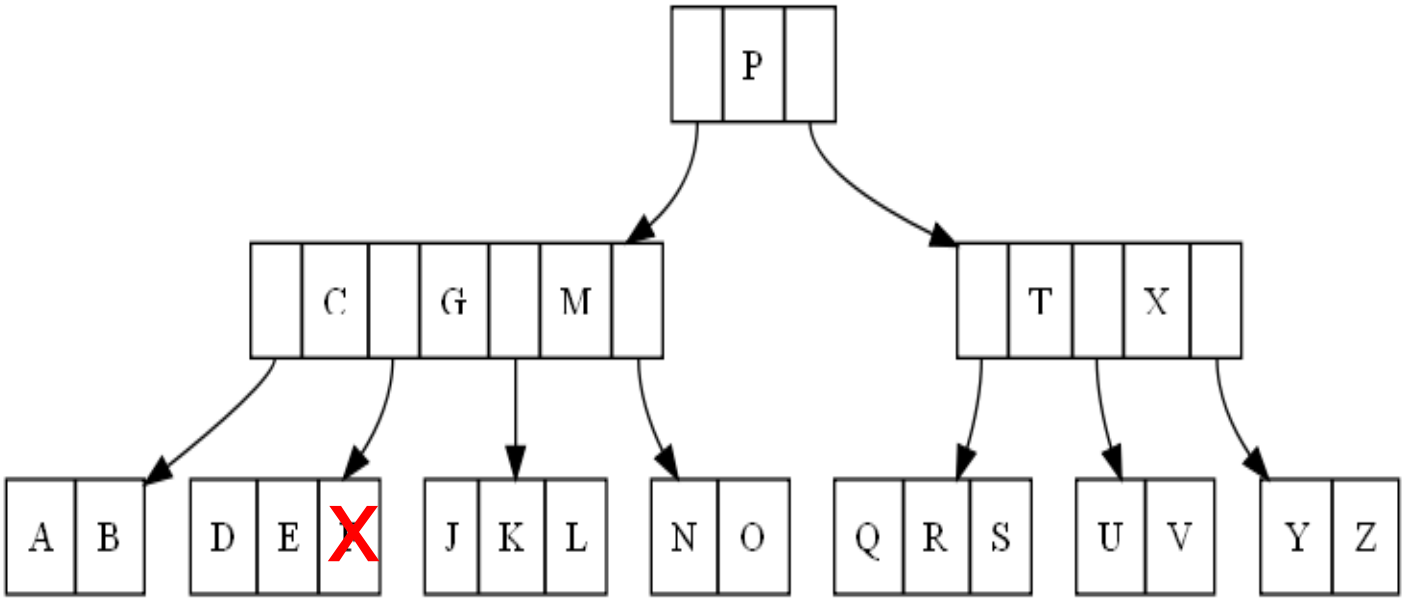
d=2



folha

Árvores B - Remoção

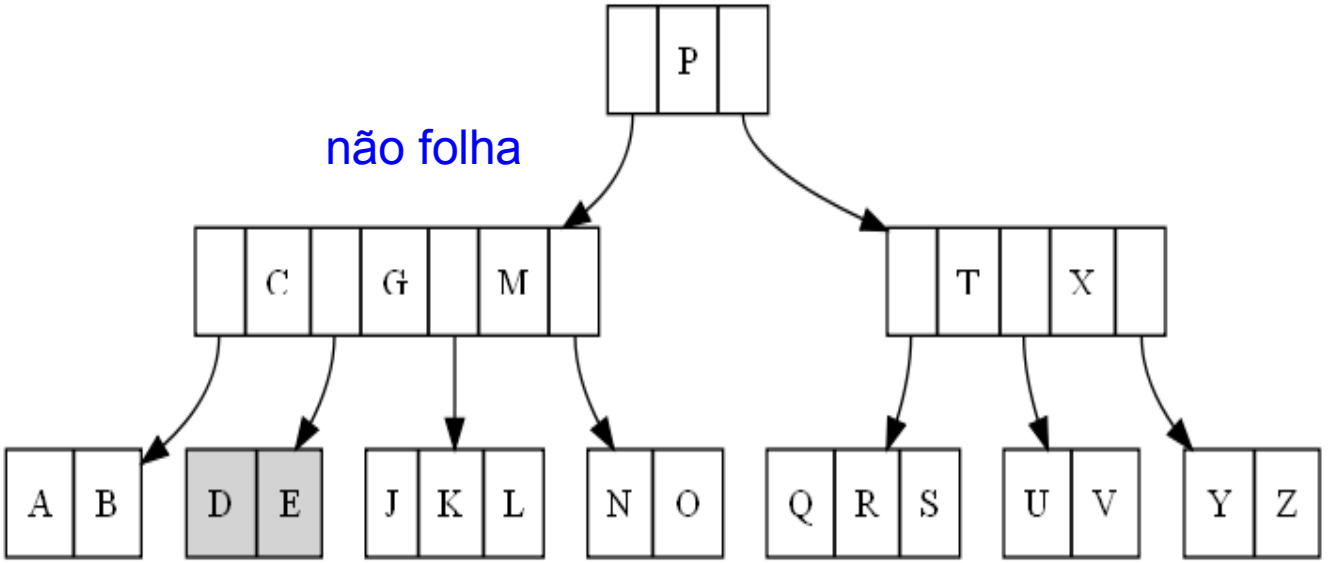
Remover 'F'



folha

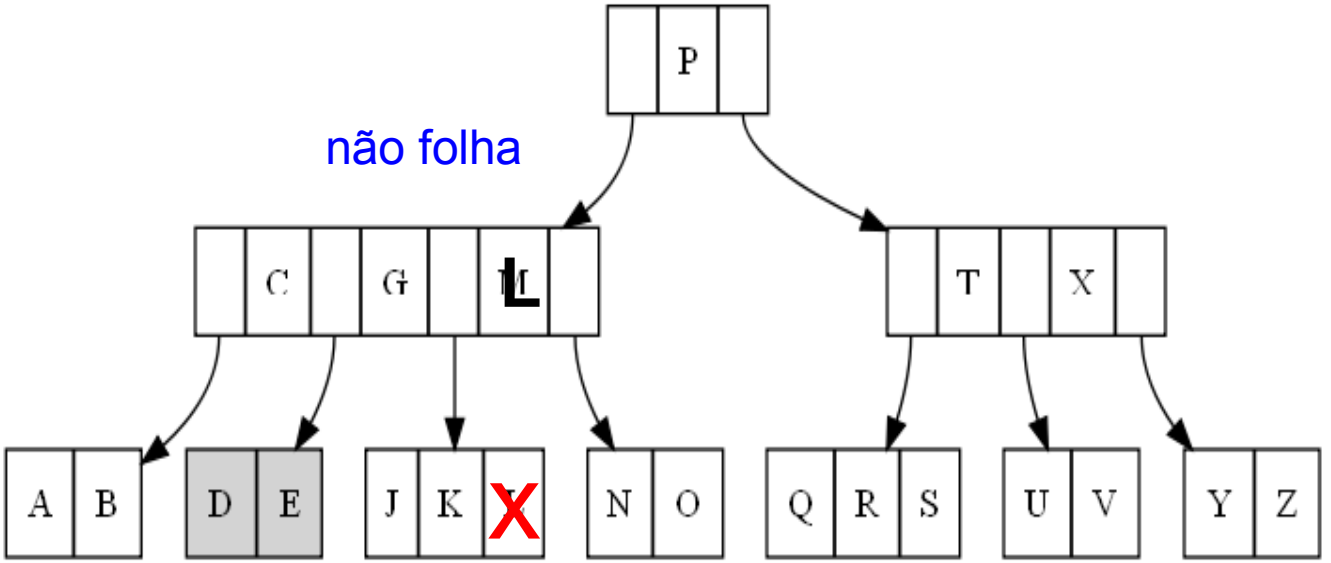
Árvores B - Remoção

Remover 'M'



Árvores B - Remoção

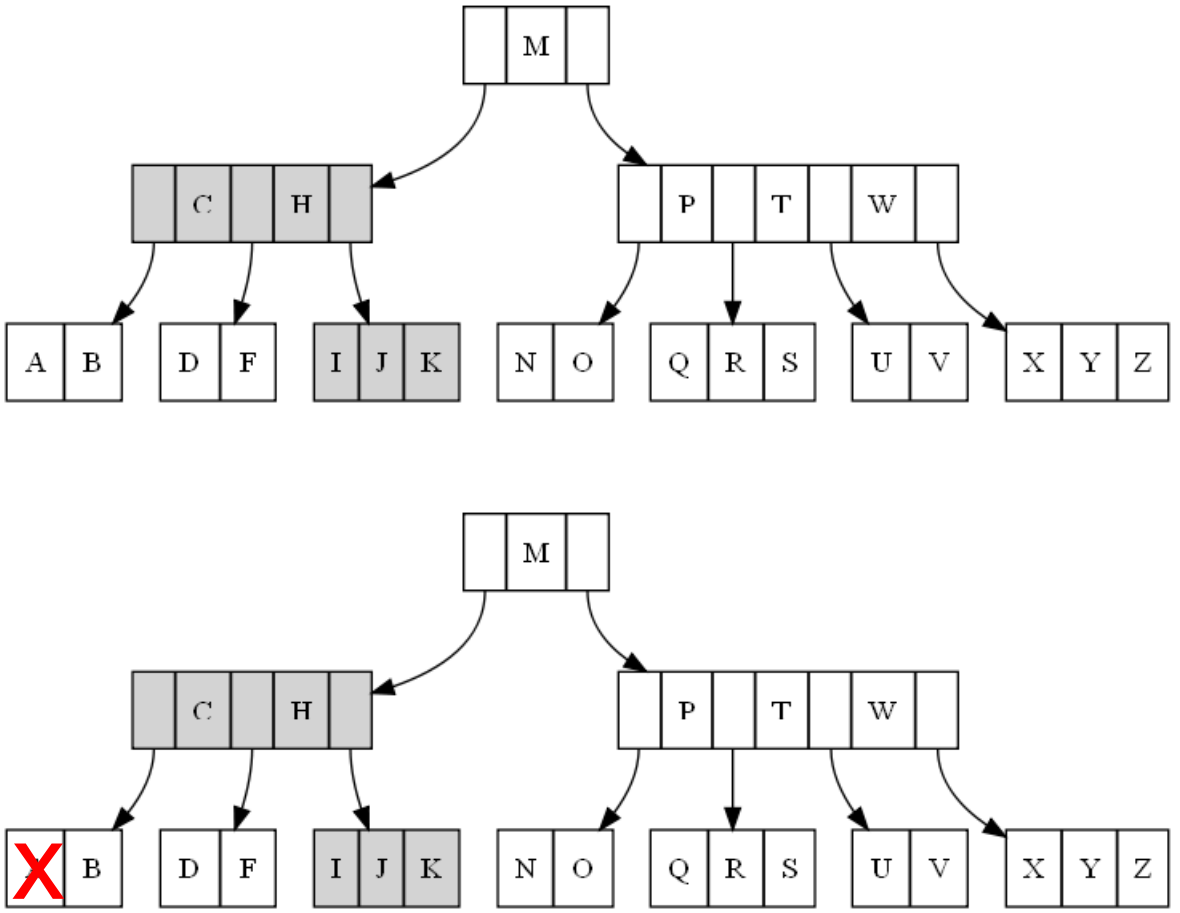
Remover 'M'



Árvores B - Remoção

Remover 'A'

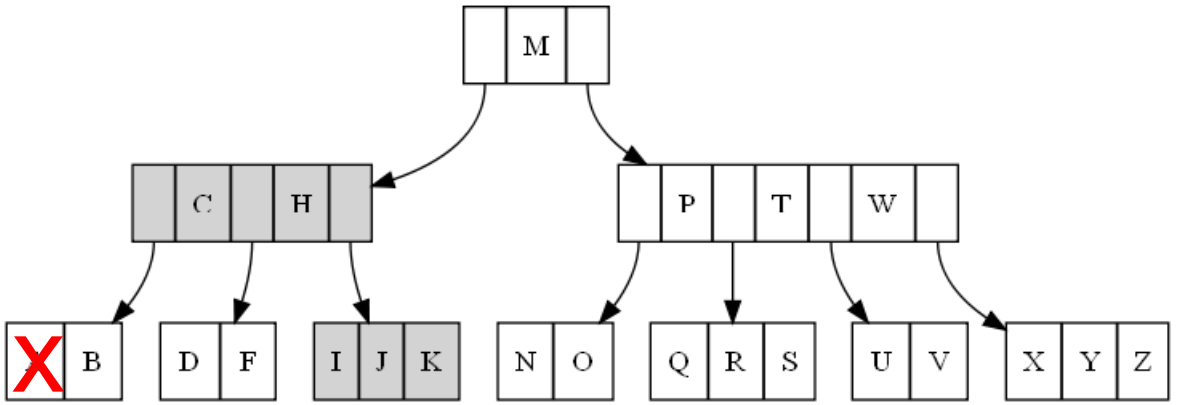
d=2



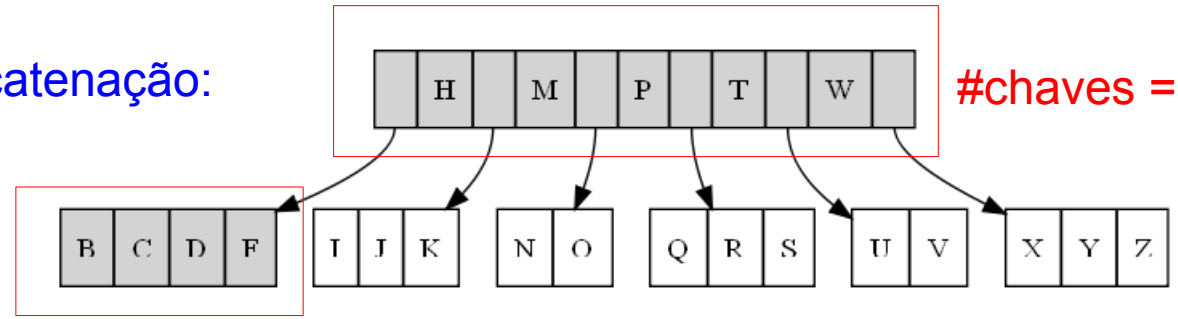
Árvores B - Remoção

Remover 'A'

d=2



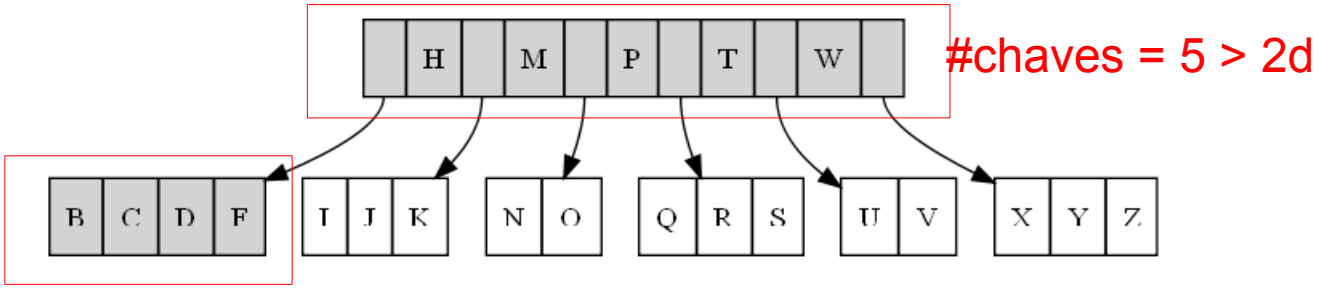
Concatenação:



#chaves = 5 > 2d

Árvores B - Remoção

Remover 'A'



Redistribuição (= concatenação seguida de cisão)

Ainda é necessário fazer a cisão da página raiz, o que implica a criação de uma nova raiz contendo a chave 'P' e dois ponteiros:

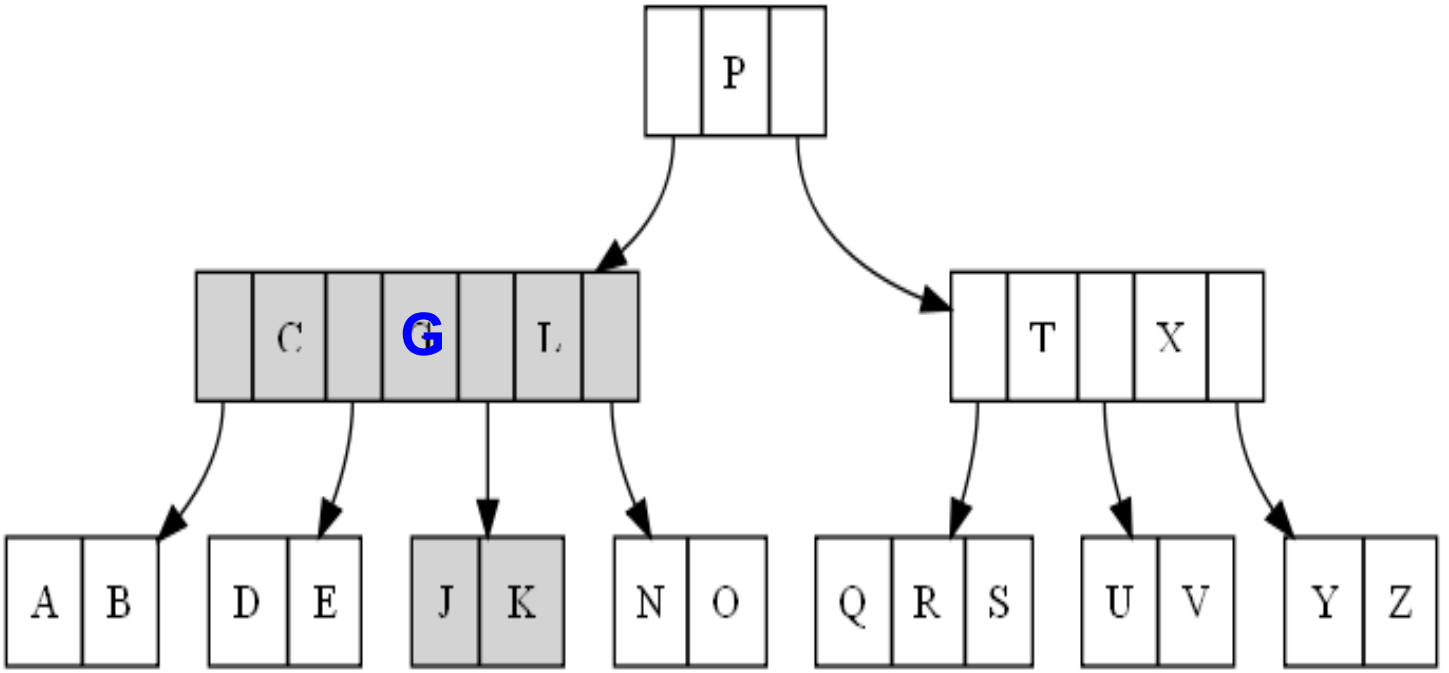
Nova raiz : [p_o,'P', p_1]

p_0 : { 'H', 'M' } p_1: { 'T', 'W' }

Árvores B - Remoção

Exercício:

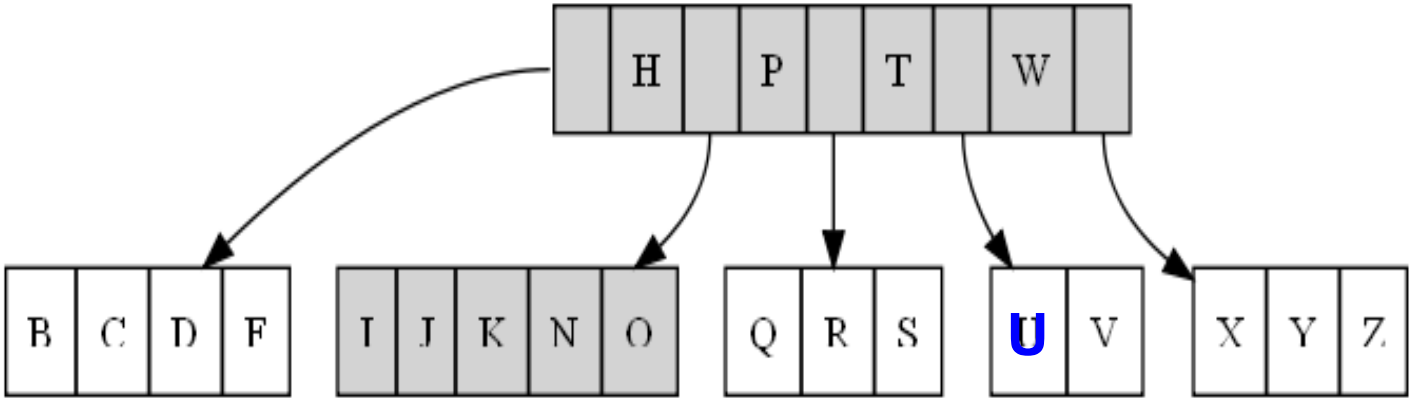
Remover 'G'



Árvores B - Remoção

Exercício:

Remover 'U'



Exercícios

1. Dê um exemplo de uma cisão de página que se propaga até à raiz em uma árvore B com d=3 e h=3.

2. Dê um exemplo de uma concatenação que se propaga até à raiz em uma árvore B com d=3 e h=3.

3. Escreva o algoritmo de inserção em uma árvore B.

4. Escreva o algoritmo de remoção em uma árvore B.