

Árvore-B⁺

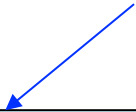
Profa. Dra. Cristina Dutra de Aguiar Ciferri

Acesso Sequencial Indexado

- Alternativas (até o momento)
 - acesso indexado
 - o arquivo pode ser visto como um conjunto de registros que são indexados por uma chave
 - acesso sequencial
 - o arquivo pode ser acessado sequencialmente (i.e., registros fisicamente contínuos)
 - Ideia
 - arquivos devem permitir acesso indexado eficiente, e também acesso sequencial
-

Organização dos Registros

- Problema
 - manter os registros ordenados fisicamente pela chave (*sequence set*)
- Solução
 - organizar registros em blocos



um bloco consiste na unidade básica de entrada e saída e deve ter seu tamanho determinado pelo tamanho do *buffer-pool*

Uso de Blocos

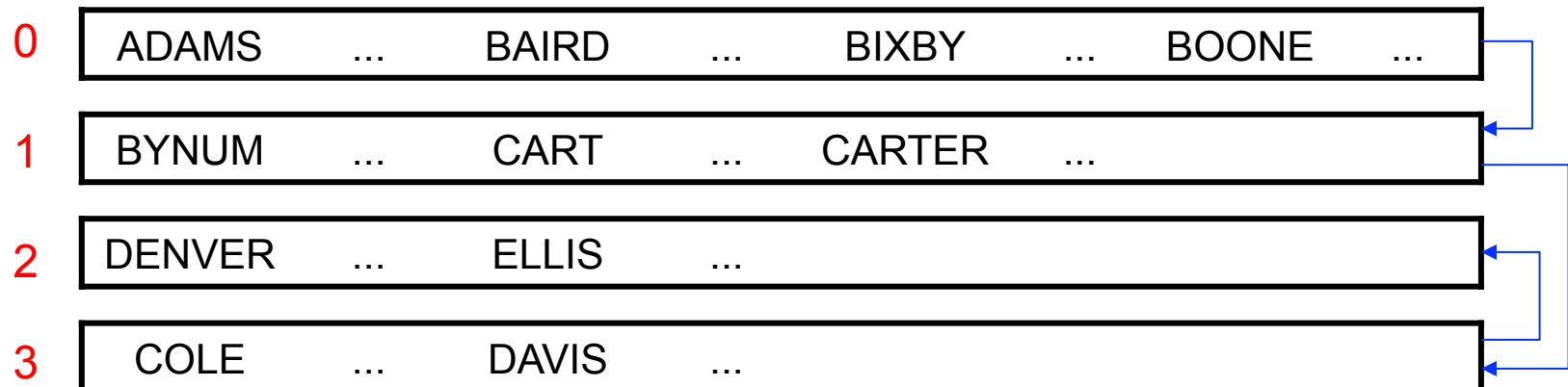
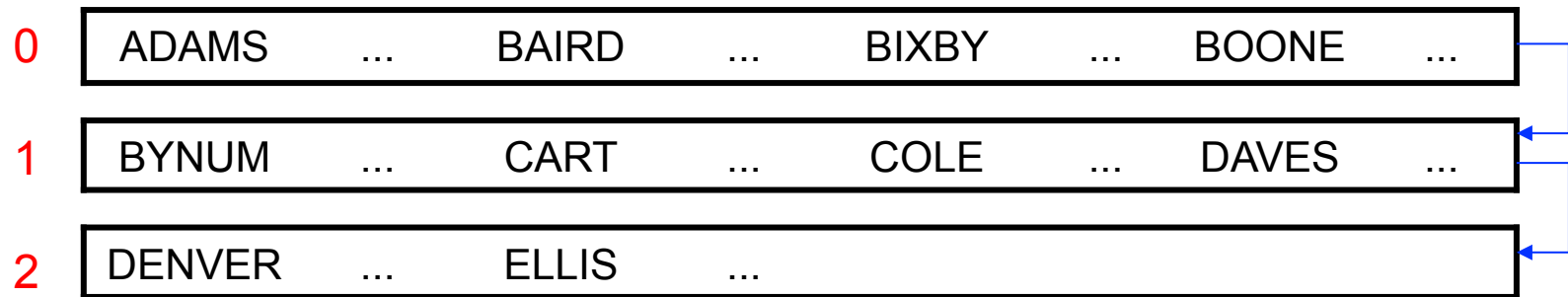
- Características
 - o conteúdo de cada bloco está ordenado, e pode ser recuperado em um acesso
 - cada bloco mantém um ‘ponteiro’ para o bloco antecessor e um ‘ponteiro’ para o bloco sucessor
 - blocos logicamente adjacentes não estão (necessariamente) fisicamente adjacentes
 - Garante acesso sequencial ao arquivo
-

Problema 1

- Inserção de registros pode provocar *overflow* em um bloco
- Solução
 - dividir o bloco, em um processo análogo ao realizado em árvores-B
 - passos
 - divide os registros entre os dois blocos
 - rearranja os ponteiros

não existe
promoção !

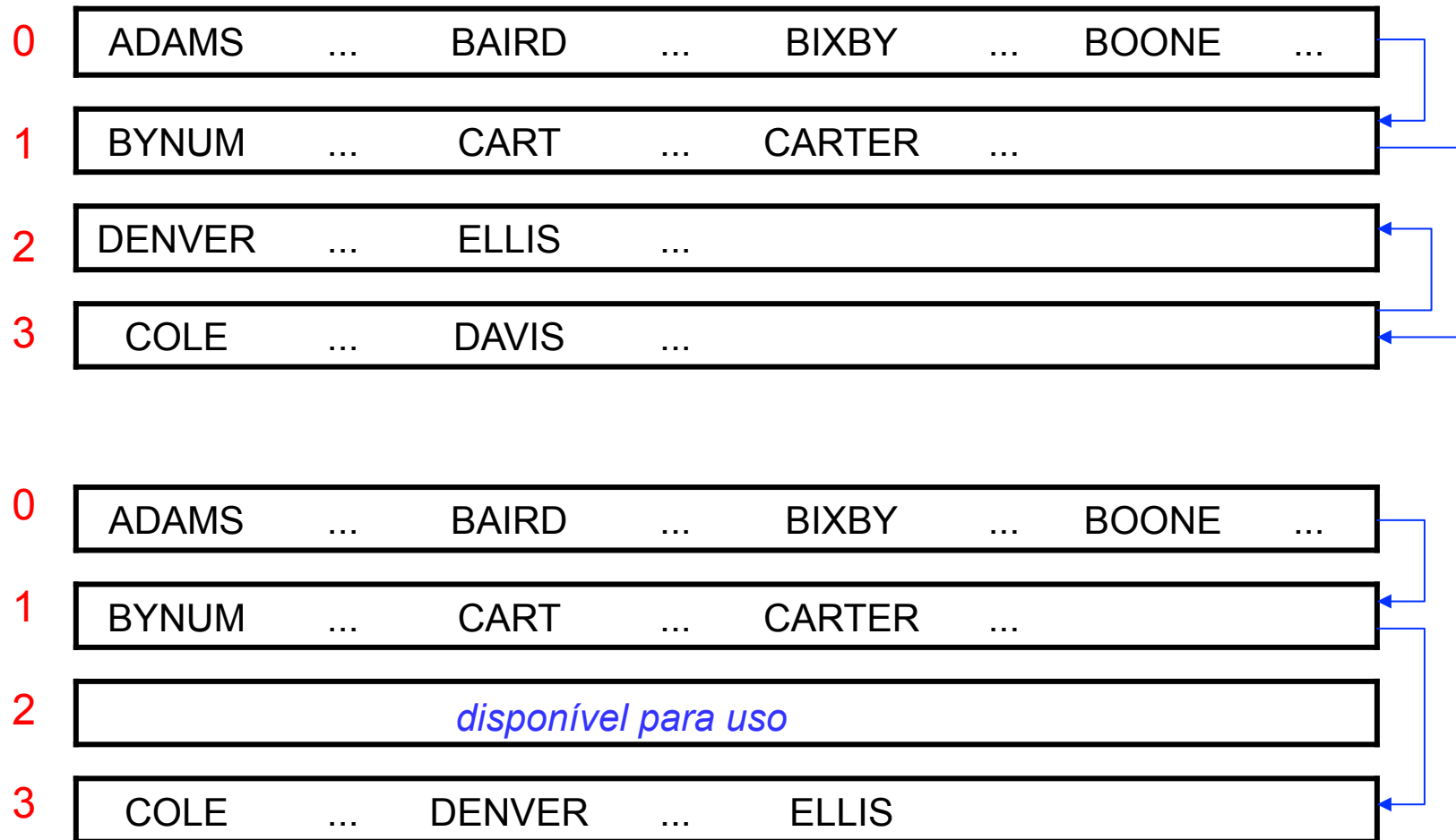
Exemplo: Inserção de CARTER



Problema 2

- Remoção de registros pode provocar *underflow* em um bloco
 - Solução
 - concatenar o bloco com o seu antecessor ou sucessor na sequência lógica
 - redistribuir os registros, movendo-os entre blocos logicamente adjacentes
-

Exemplo: Remoção de DAVIS



Uso de Blocos

- Custos associados
 - devido à fragmentação gerada pelas inserções, o arquivo pode ocupar mais espaço do que um arquivo ordenado comum
 - melhorias incluem redistribuição antes do particionamento, *split 2-to-3*, etc
 - a ordem física dos registros não é necessariamente sequencial ao longo do arquivo
-

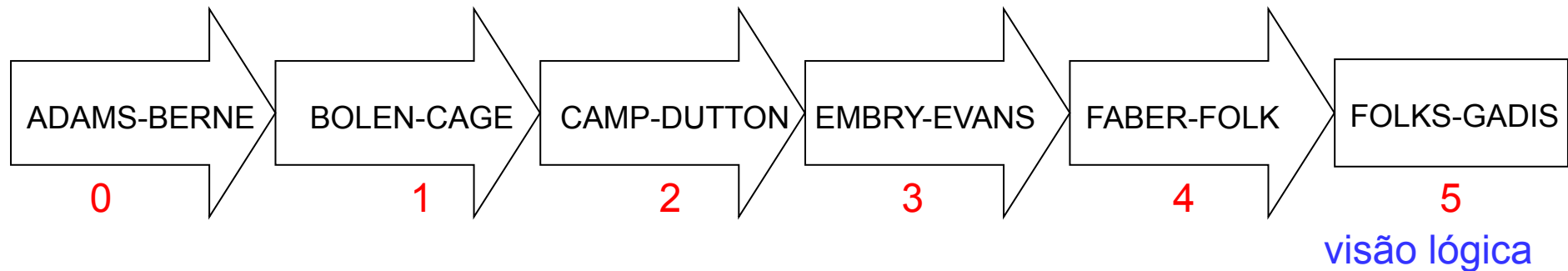
Tamanho do Bloco

- Consideração 1
 - deve permitir que diversos blocos possam ser armazenados em RAM ao mesmo tempo
 - Consideração 2
 - deve permitir que um bloco possa ser acessado sem se pagar o custo de um *seek* com a operação de leitura ou escrita do bloco
 - a leitura ou a escrita de um bloco não deve consumir muito tempo
-

Acesso aos Registros

- Característica
 - os registros podem ser acessados em ordem, sequencialmente, pela chave
 - Problema
 - localizar eficientemente um bloco com um registro particular, dado a chave do registro
 - Soluções
 - índice simples para referenciar os blocos
 - árvore-B+
-

Índice Simples (Tabela)



chave	bloco
BERGE	0
CAGE	1
DUTTON	2
EVANS	3
FOLK	4
GADIS	5

Índice de 1 nível

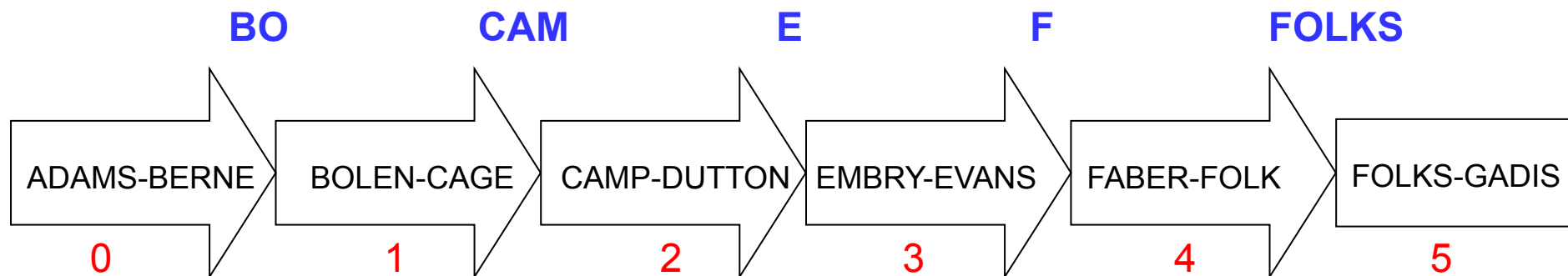
- registros de tamanho fixo
- contém a chave para o último registro no bloco

Acesso Sequencial Indexado

- Combina
 - registros ordenados fisicamente pela chave (*sequence set*)
 - índice simples para referenciar os blocos
 - Restrição
 - a organização em tabela implica que o **índice cabe na memória principal**
 - busca binária no índice
 - atualização do índice em RAM
-

Separadores

- Características
 - são mantidos no índice, ao invés das chaves de busca
 - possuem tamanho variável
- Exemplo



Separadores

- Desafio
 - escolher o menor separador para utilizar no índice
- Tabela de decisão

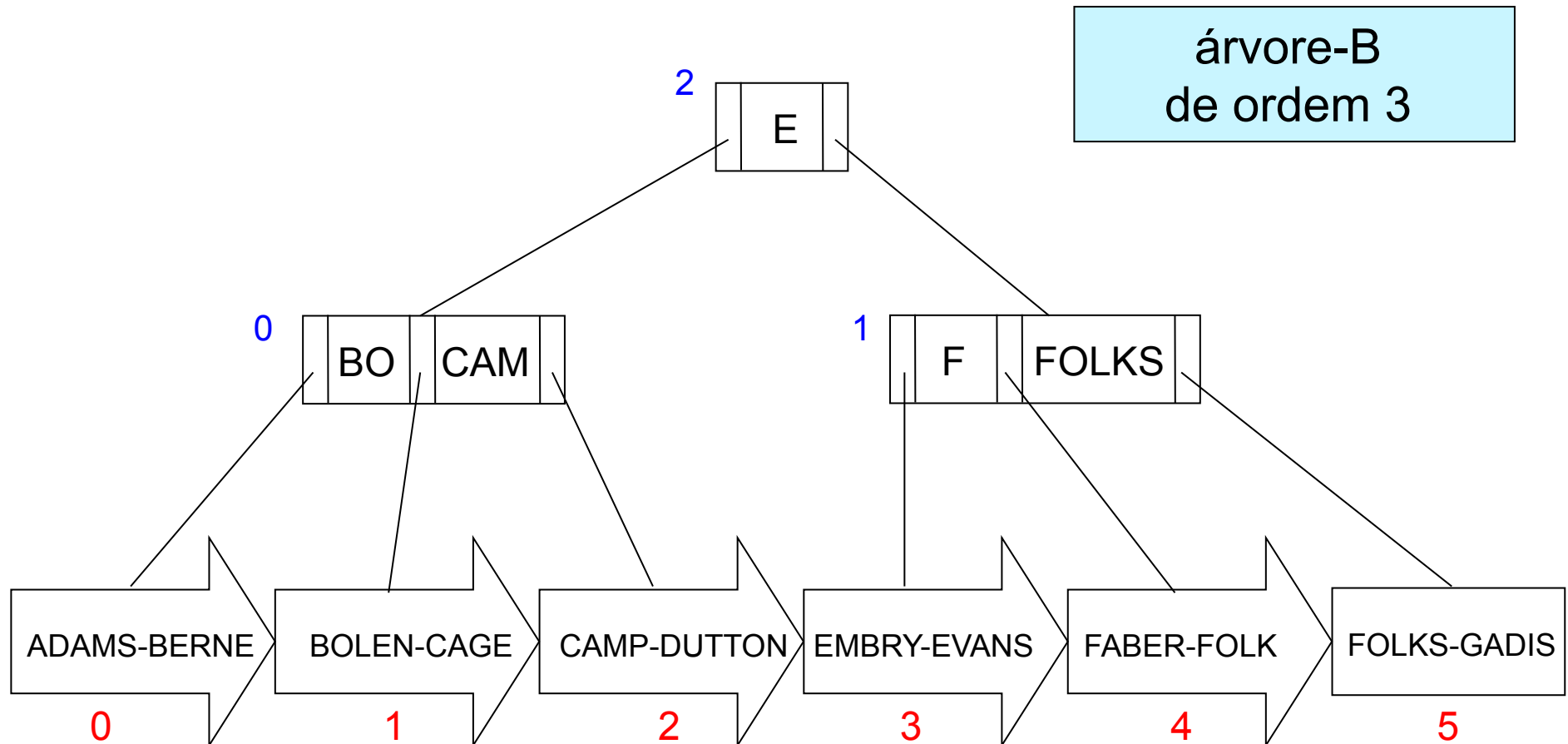
chave de busca x separador	decisão
chave < separador	procure à esquerda
chave = separador	procure à direita
chave > separador	procure à direita

Árvore-B+ Pré-Fixada

- Estrutura híbrida
 - chaves
 - organizadas como *árvore-B*
 - nós folhas
 - consistem em blocos de *sequence set*
 - Pré-fixada simples
 - armazena na árvore as cadeias separadoras mínimas entre cada par de blocos
-

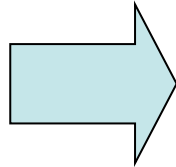
Árvore-B+ Pré-Fixada

árvore-B
de ordem 3



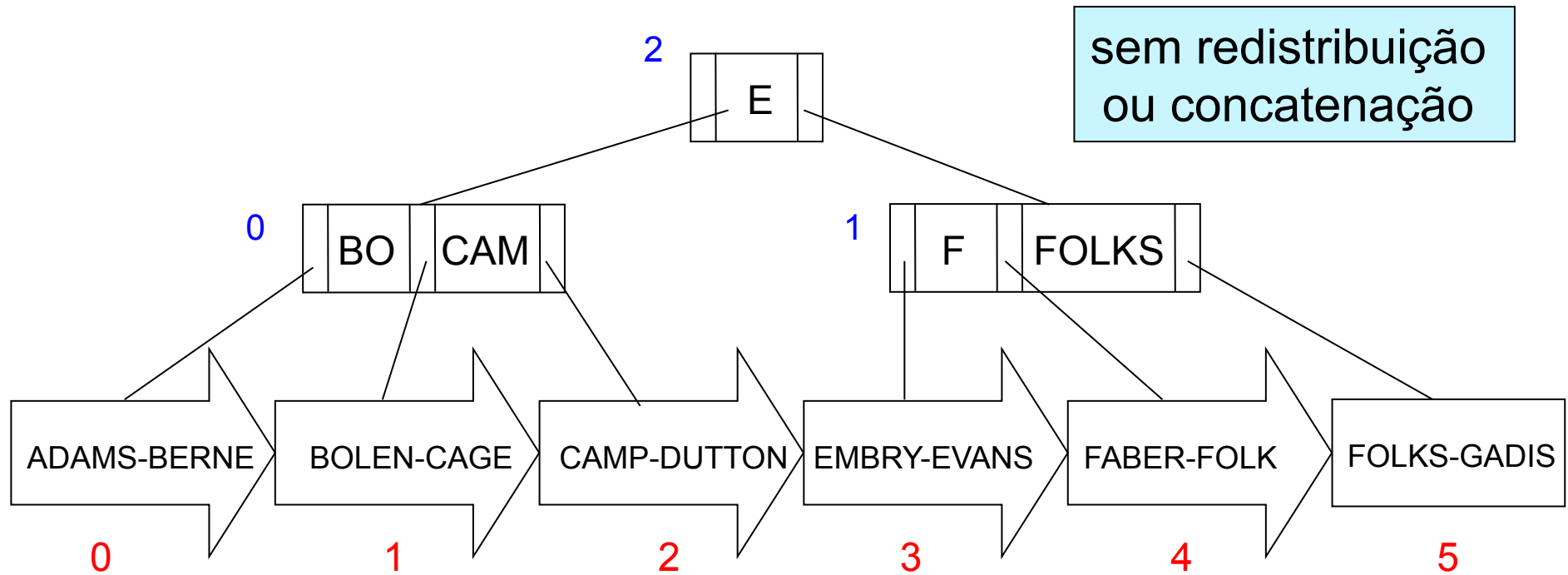
Manutenção

- Cenários
 - inserção
 - remoção
 - *overflow*
 - *underflow*

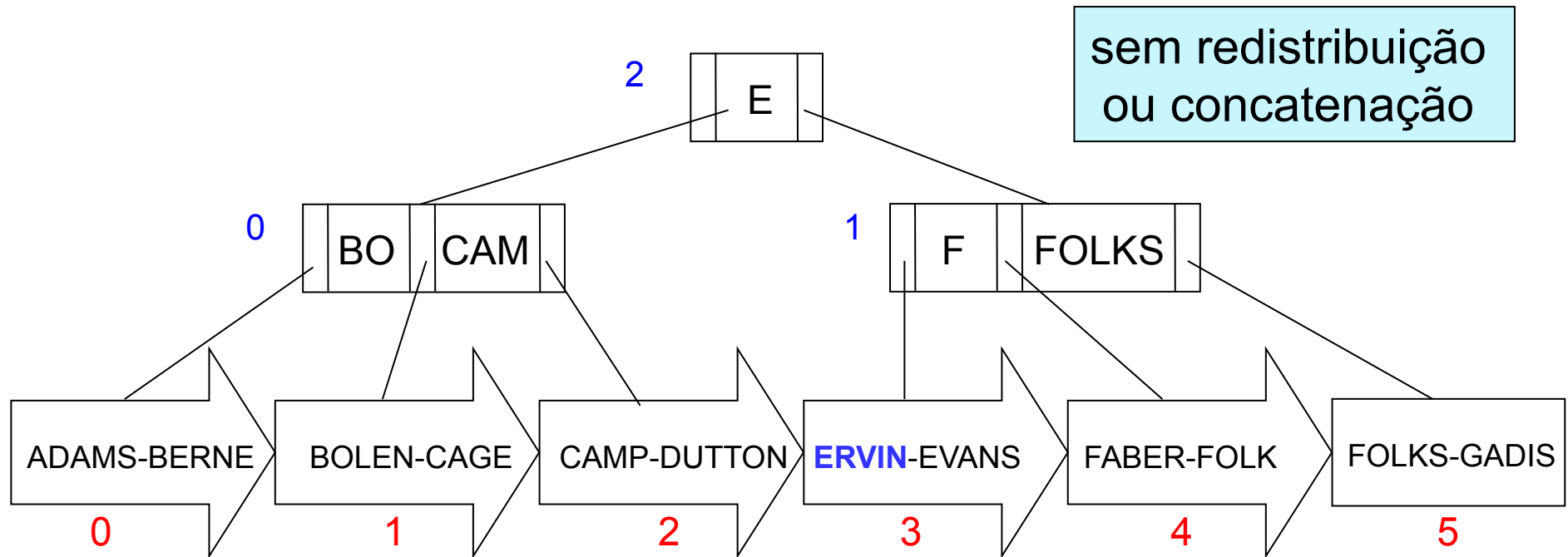


- Efeitos colaterais
 - *sequence set*
 - árvore-B+

Remoção de EMBRY (1/3)

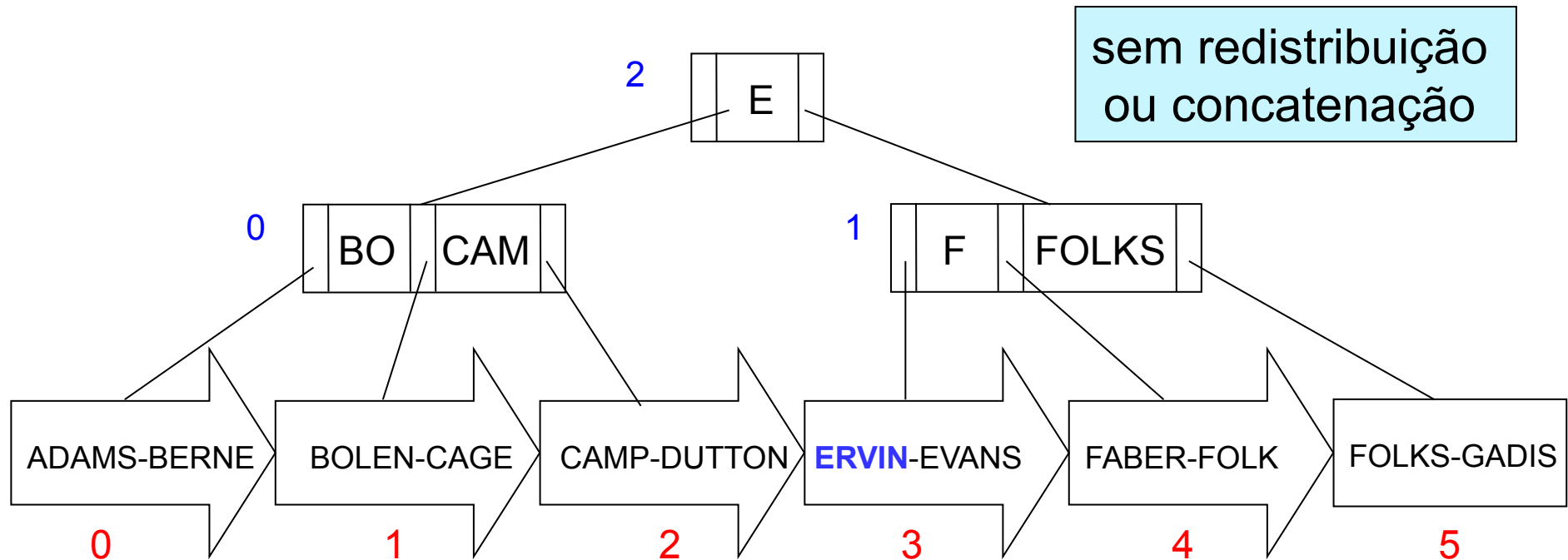


Remoção de EMBRY (2/3)



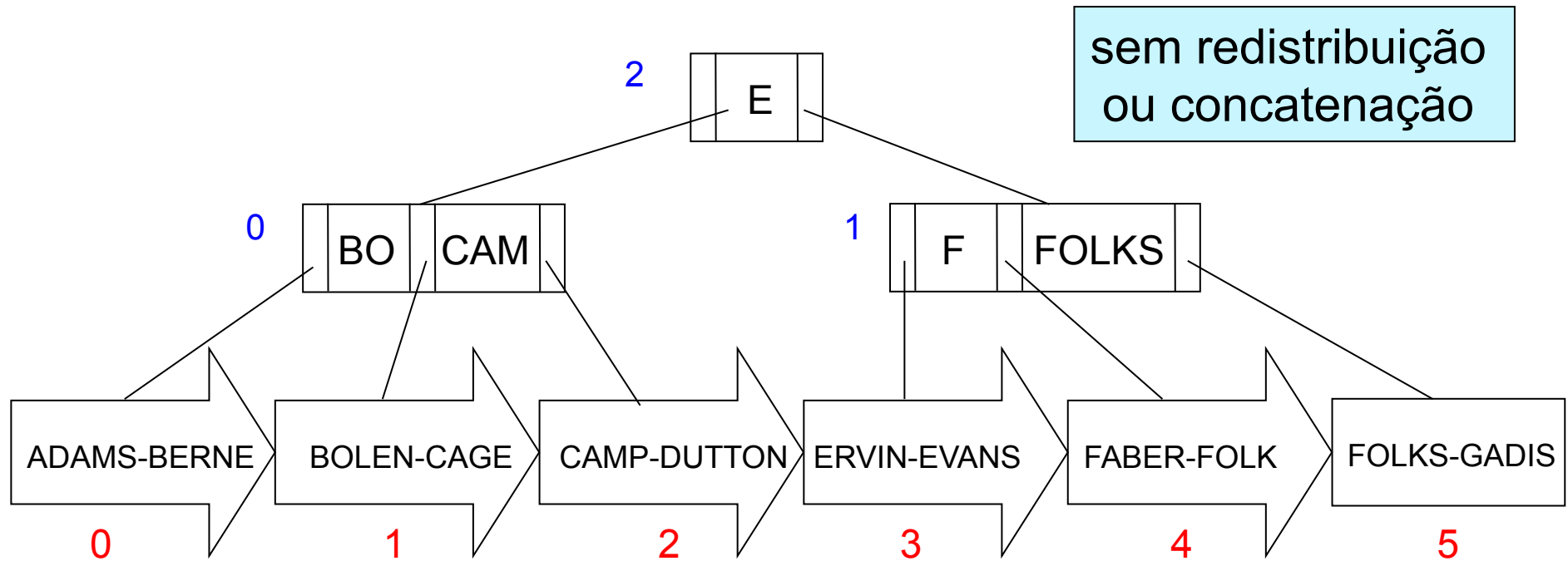
- Efeito no *sequence set*
 - limitado a alterações no bloco 3
-

Remoção de EMBRY (3/3)

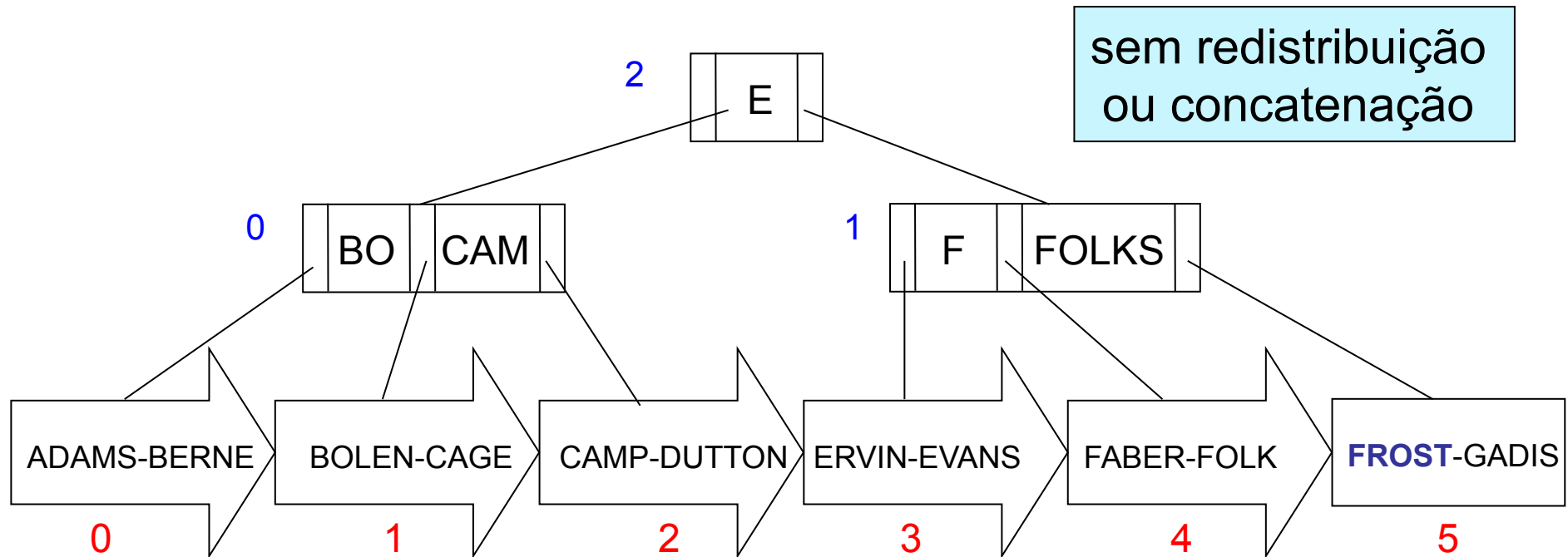


- Efeito na **árvore-B+**
 - nenhum: E é uma boa chave separadora
-

Remoção de FOLKS (1/3)

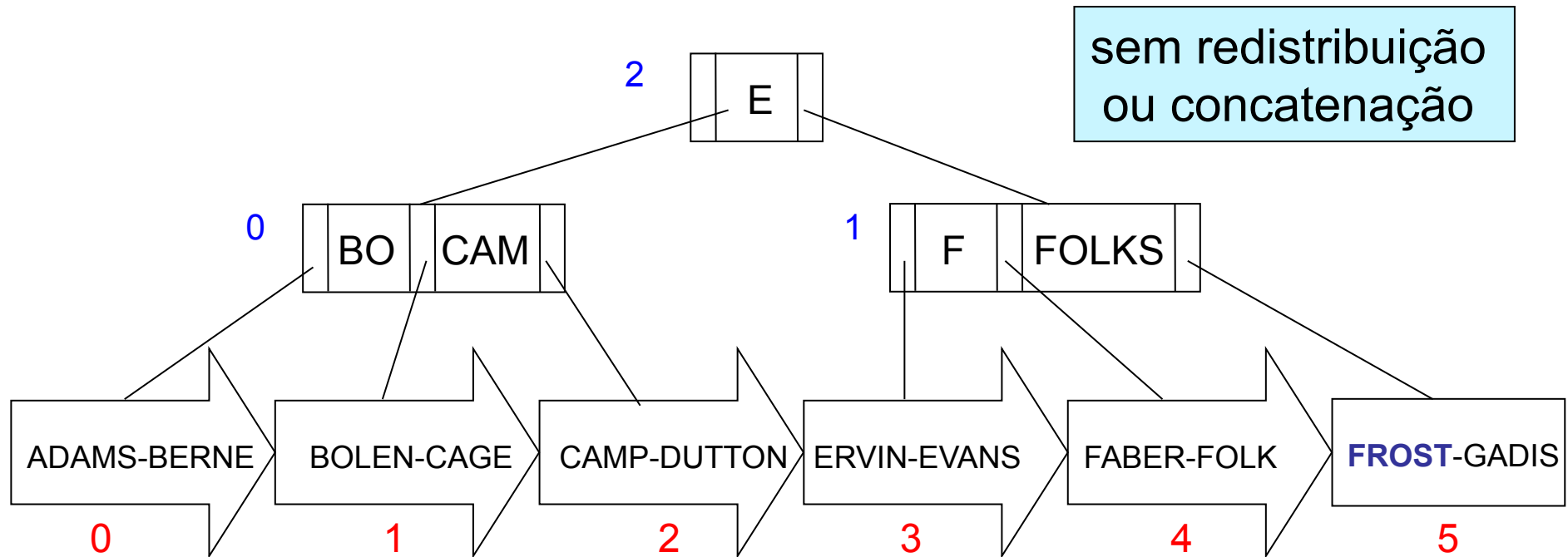


Remoção de FOLKS (2/3)



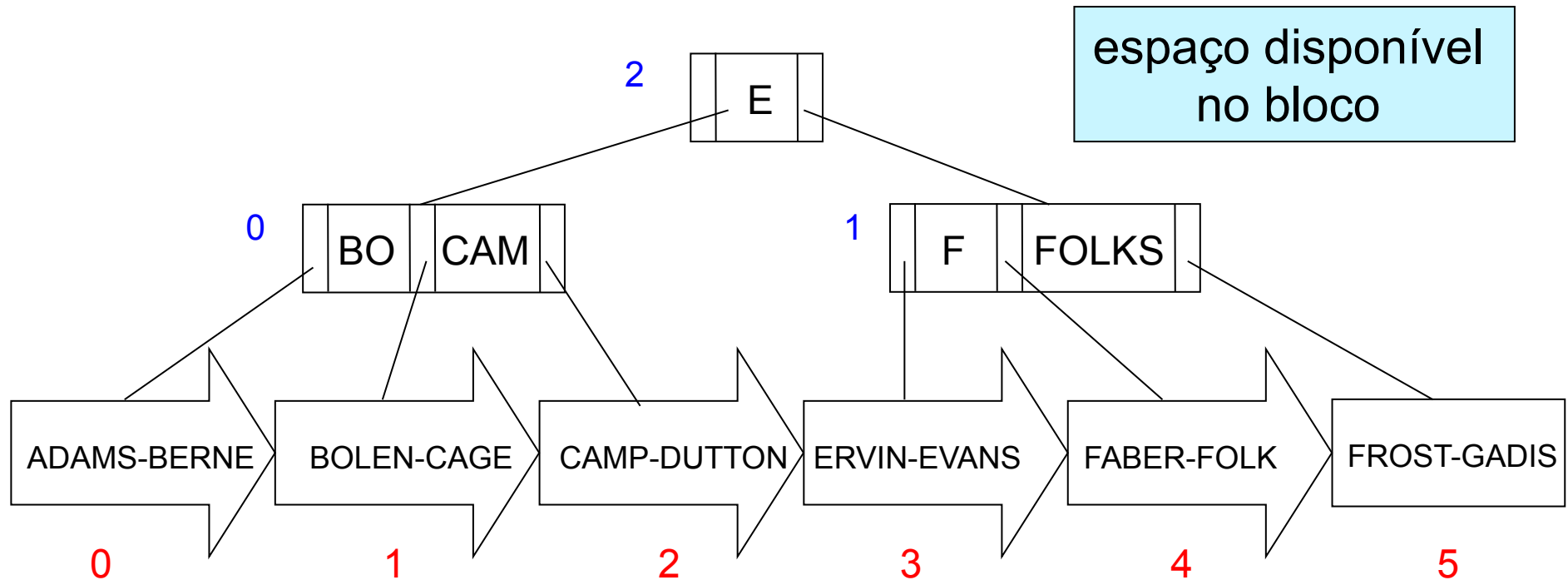
- Efeito no *sequence set*
 - limitado a alterações no bloco 5
-

Remoção de FOLKS (3/3)

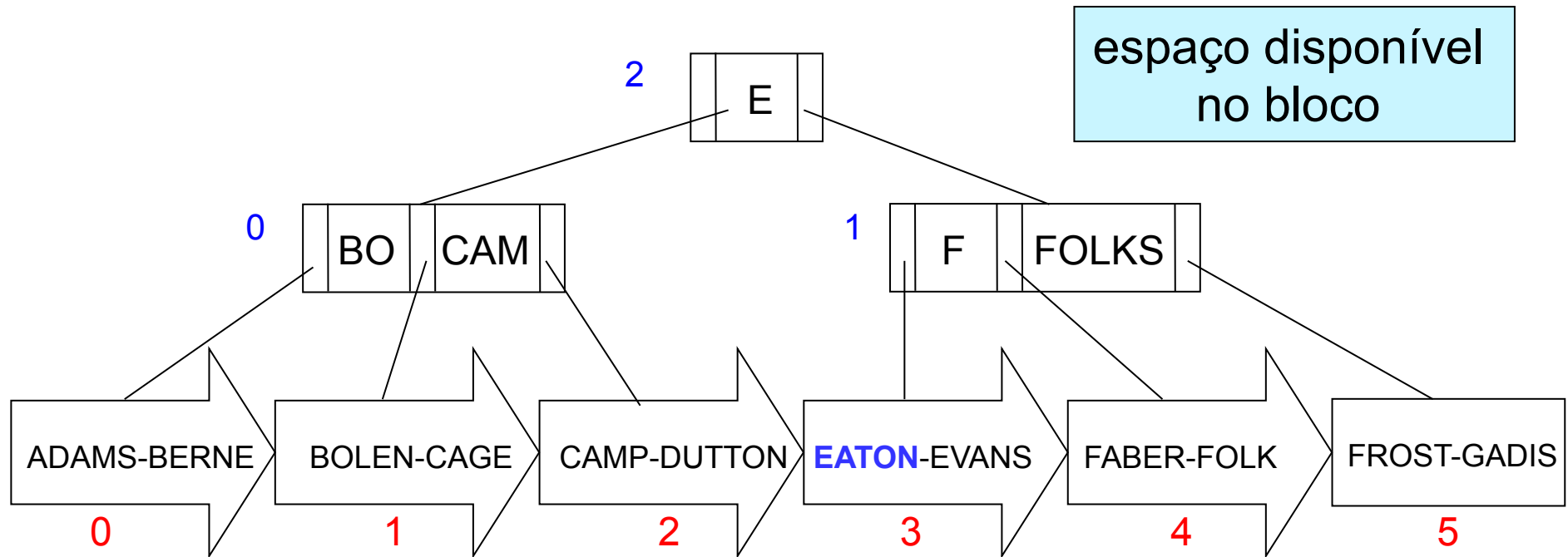


- Efeito na **árvore-B+**
 - nenhum: custos elevados

Inserção de **EATON** (1/3)

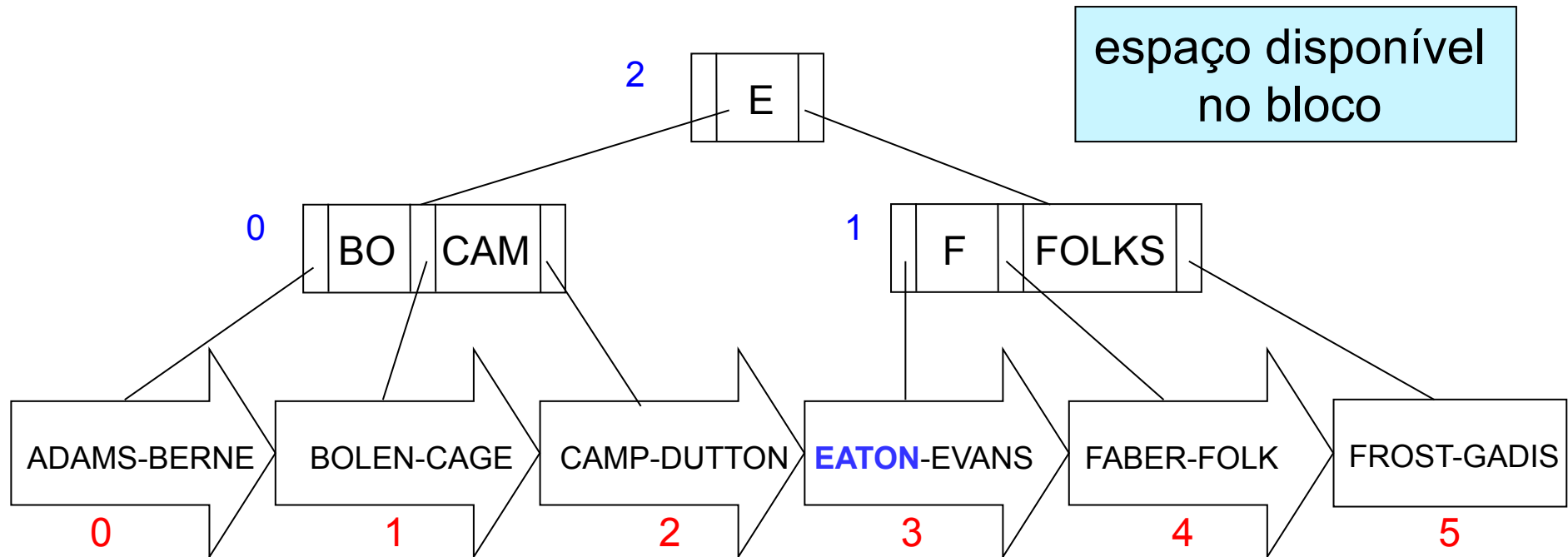


Inserção de EATON (2/3)



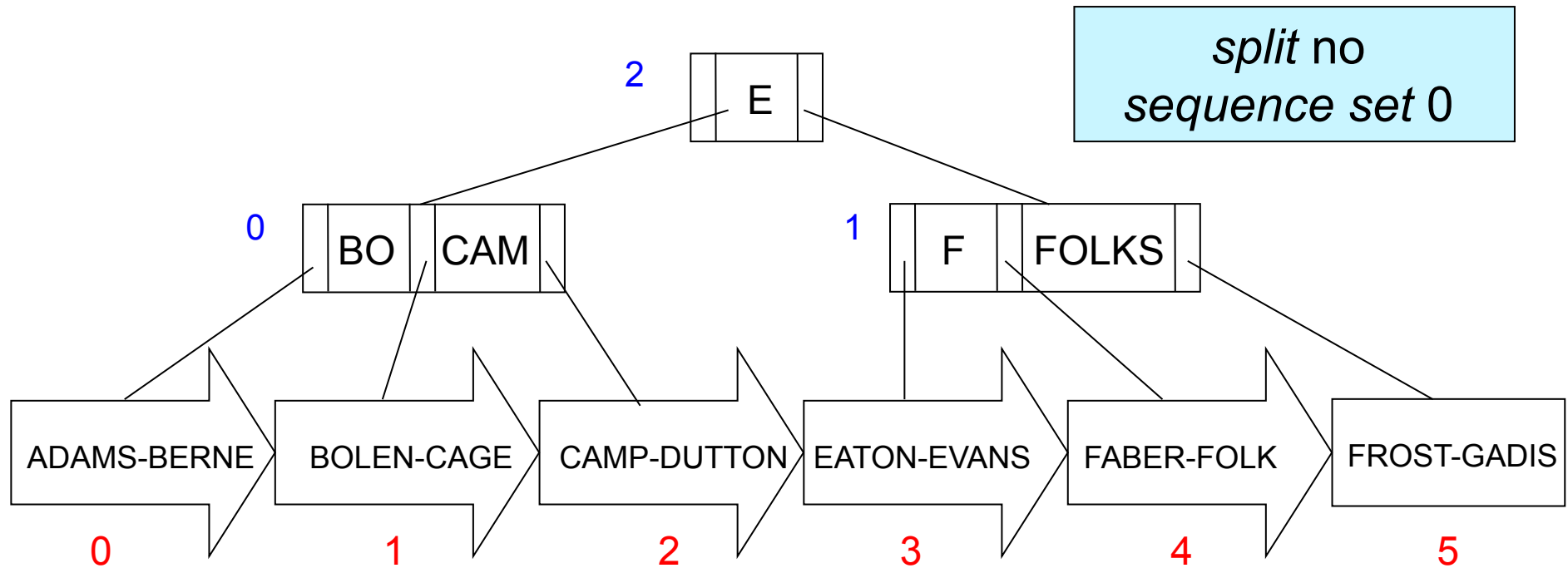
- Efeito no *sequence set*
 - limitado a alterações no bloco 3

Inserção de **EATON** (3/3)

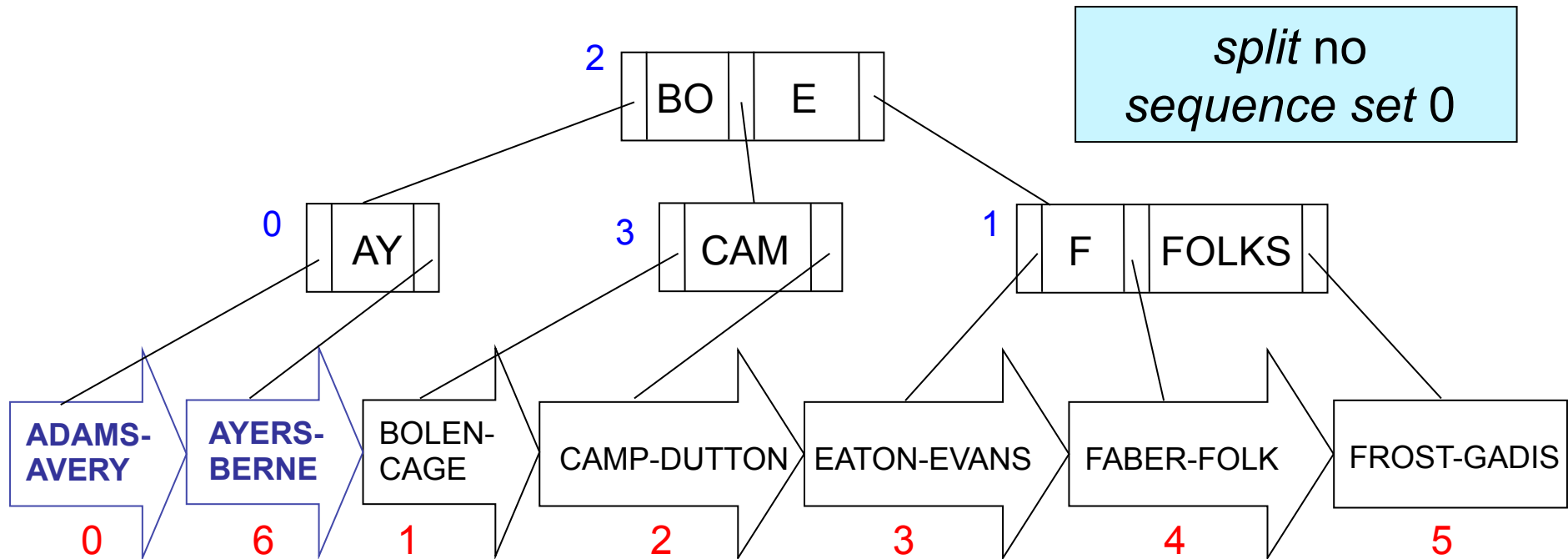


- Efeito na **árvore-B+**
 - nenhum: E é uma boa chave separadora

Inserção de **AVERY** (1/3)

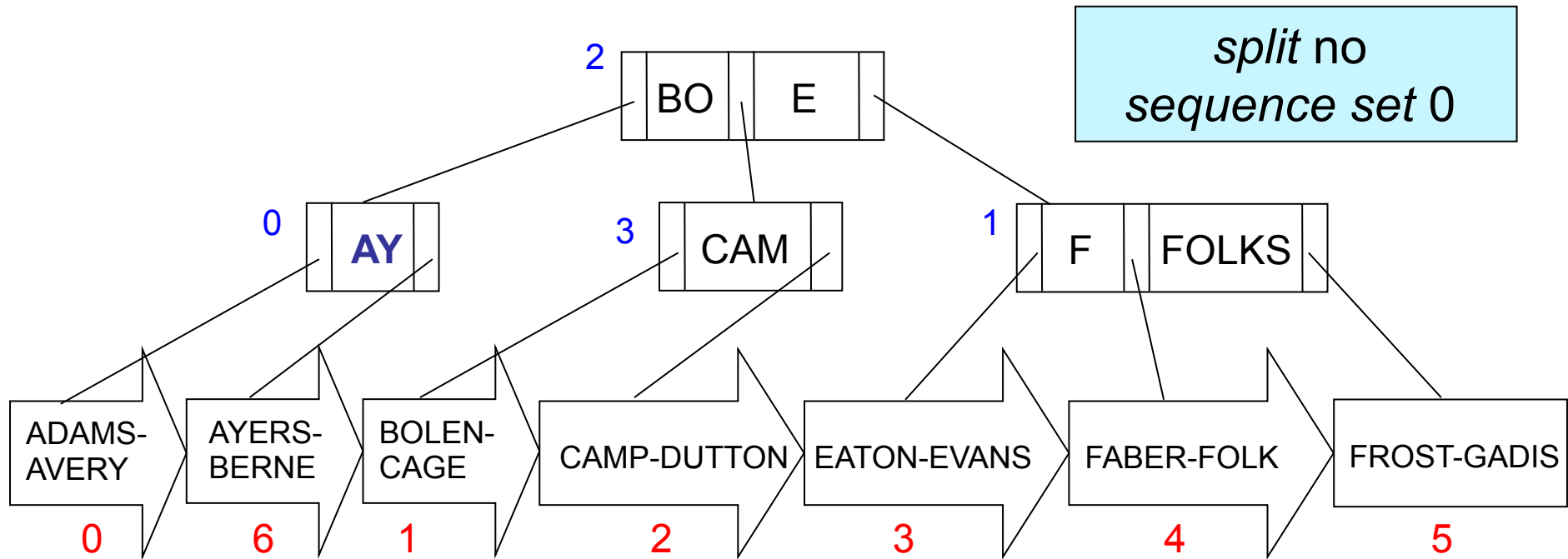


Inserção de **AVERY** (2/3)



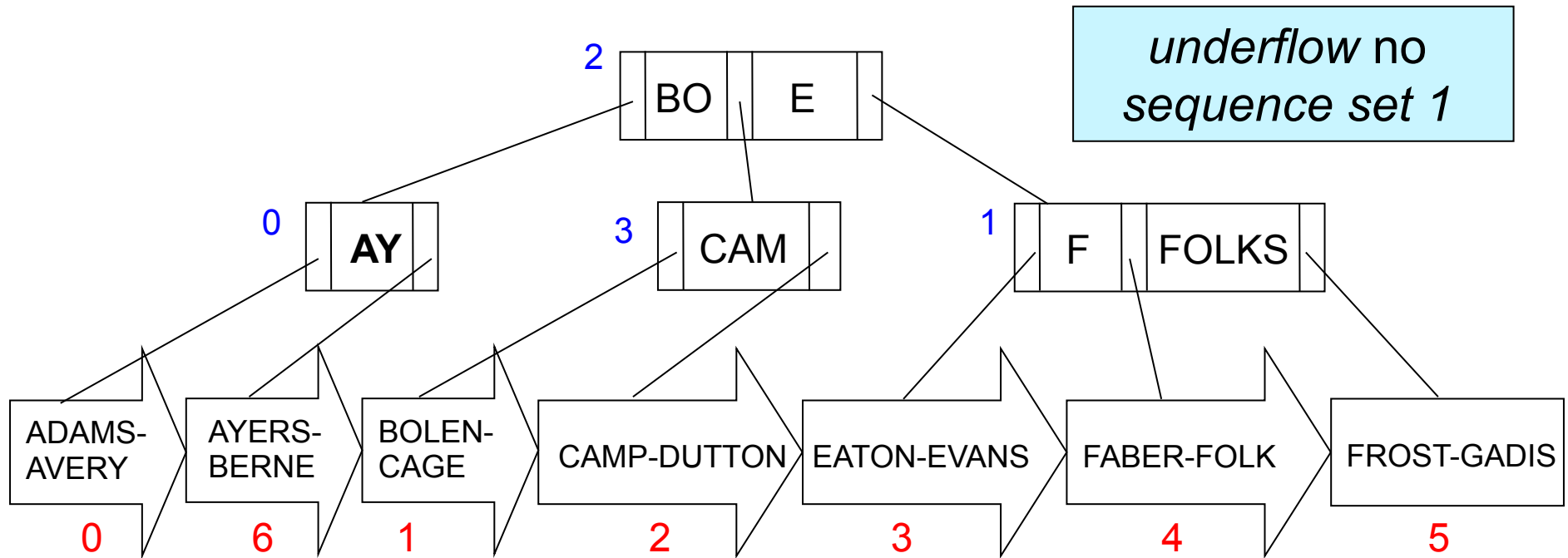
- Efeito no *sequence set*
 - dados do bloco 0 + AVERY distribuídos entre os blocos 0 e 6
-

Inserção de **AVERY** (3/3)

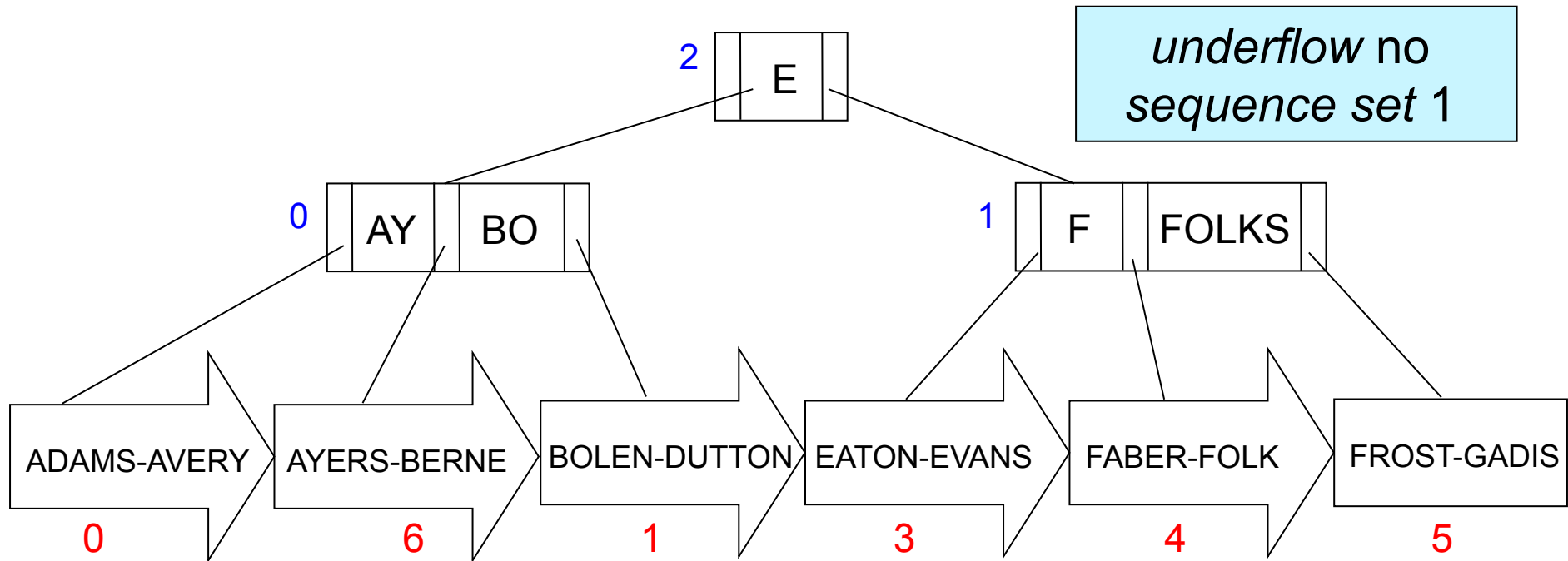


- Efeito na **árvore-B+**
 - separador adicional **AY** + *split* + promoção de chave
-

Remoção de CAEL (1/3)

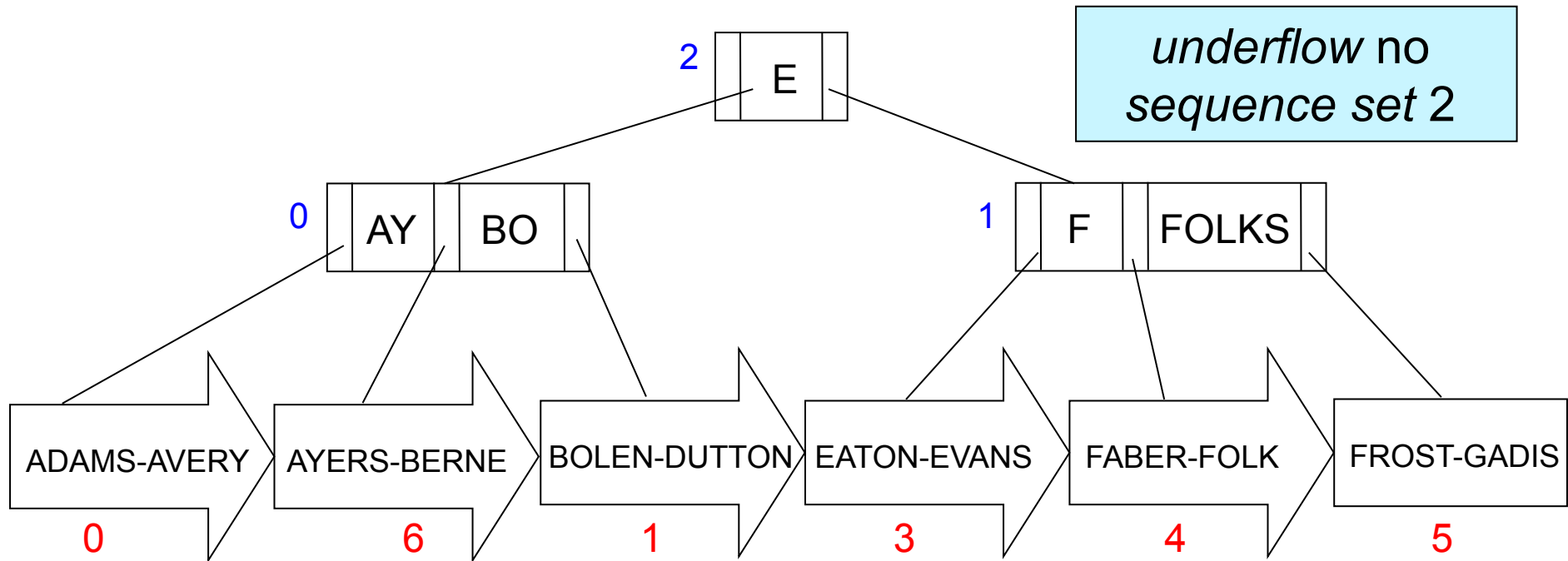


Remoção de CAEL (2/3)



- Efeito no *sequence set*
 - concatenação dos blocos 1 e 2

Remoção de CAEL (3/3)



- Efeito na **árvore-B+**
 - remoção de CAM e concatenação de nós

Pesquisa

- Passos
 - primeiro: *Árvore-B⁺*
 - segundo: *Sequence Set*
- Inserção e Remoção
 - iniciam-se pela pesquisa

buscas são
sempre realizadas
a partir do arquivo
arquivo de índice !

Inserção e Remoção

- Primeiro passo: *Sequence Set*
 - inserir ou remover o dado
 - tratar *split*, concatenação e redistribuição (se necessário)
- Segundo passo: *Árvore-B⁺*
 - se *split* no *sequence set*,
inserir um novo separador no índice
 - se *concatenação* no *sequence set*
remover um separador do índice
 - se *distribuição* no *sequence set*
alterar o valor do separador no índice

inserções e
remoções são
sempre
realizadas a
partir do arquivo
de dados !

Observações Adicionais

- Tamanho físico de um nó no índice (i.e., árvore-B⁺) = Tamanho físico de um bloco no *sequence set*
 - Escolha direcionada pelos mesmos quesitos
 - tamanho do bloco
 - características do disco
 - quantidade de memória disponível
-

Observações Adicionais

- Tamanho físico de um nó no índice (i.e., árvore-B⁺) = Tamanho físico de um bloco no *sequence set*
 - Facilidade para a implementação da árvore-B⁺ virtual
 - Uso de um mesmo arquivo para armazenar os blocos do índice e os blocos do *sequence set*
 - evita *seeks* entre dois arquivos separados
-