Árvores-B$^+$

Profa. Dra. Cristina Dutra de Aguiar Ciferri

Baseado em material de vários professores

Acesso Seqüencial Indexado

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

Exemplo

- Arquivo indexado por um índice árvore-B
  - acesso indexado pela chave: desempenho excelente
  - acesso seqüencial aos registros ordenados pela chave: desempenho péssimo
- Arquivo com registros ordenados pela chave
  - processamento seqüencial: apropriado
  - processamento randômico: inapropriado

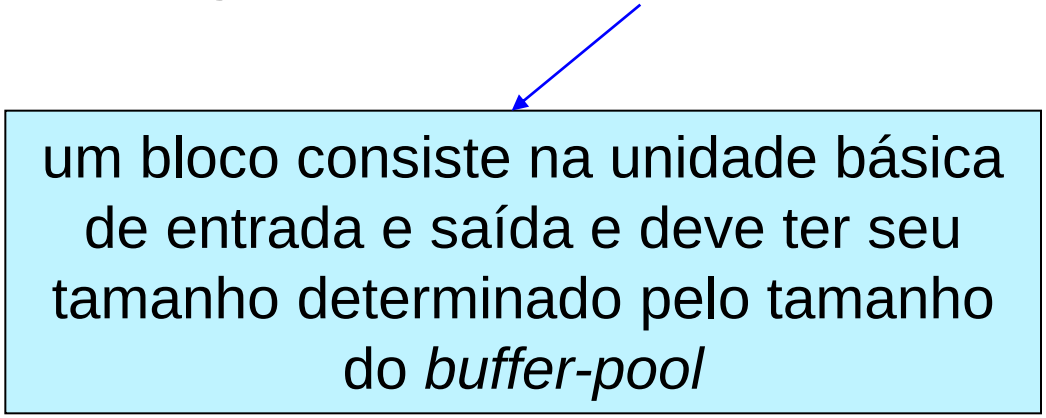
Exemplo

- Questão
  - registros organizados segundo a seqüência de entrada (*entry sequenced*)

  versus
  - registros ordenados fisicamente pela chave
- Cenários de utilização
  - registros de alunos em uma Universidade
  - registros com informações relacionadas a cartões de crédito

Foco 1

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



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 seqüencial ao arquivo

Problema 1

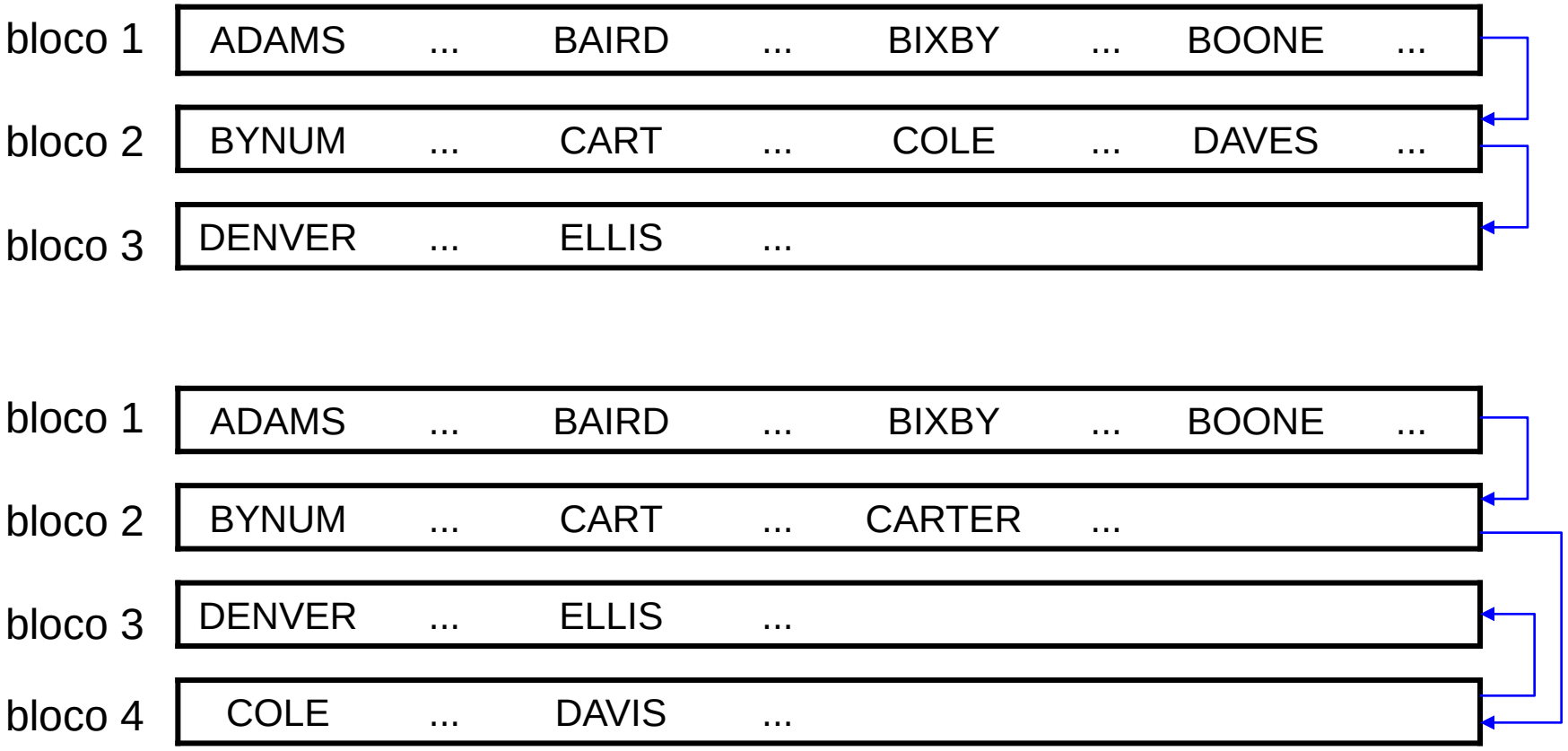
- Inserção de registros pode provocar *overflow* em um bloco
- Solução
  - dividir o bloco, em um processo análogo ao realizado em árvores-B
  - passos
    - divide os registros entre os dois blocos
    - rearranja os ponteiros

não existe promoção !

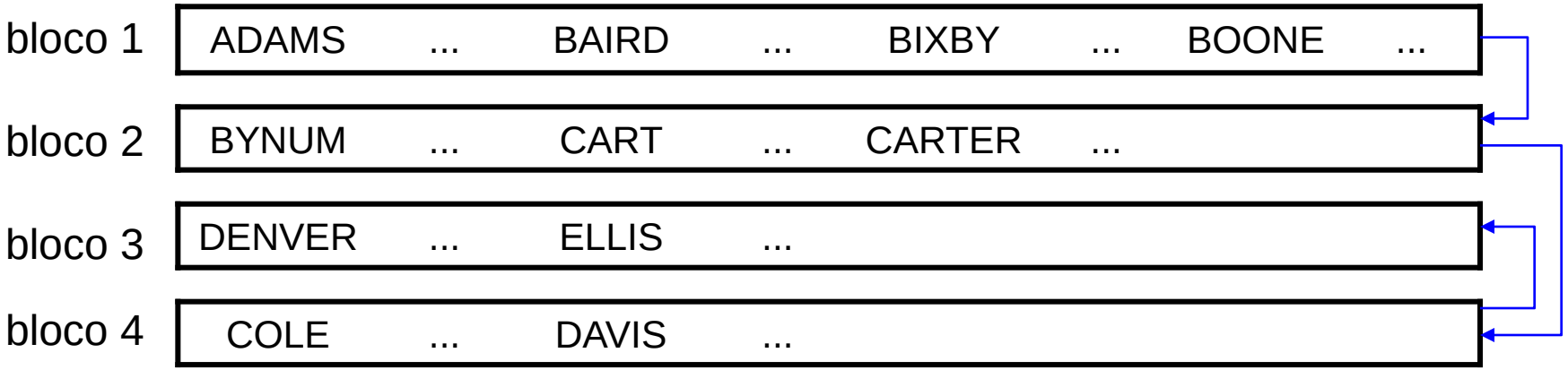
Problema 2

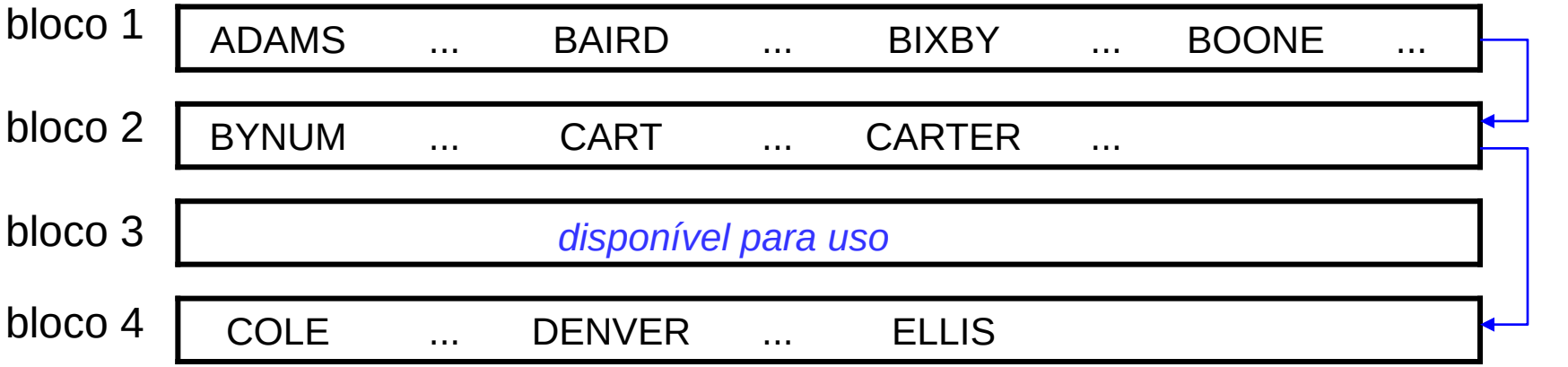
- Remoção de registros pode provocar *underflow* em um bloco
- Solução
  - concatenar o bloco com o seu antecessor ou sucessor na seqüência lógica
  - redistribuir os registros, movendo-os entre blocos logicamente adjacentes

Exemplo: Inserção de CARTER



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 seqüencial ao longo do arquivo

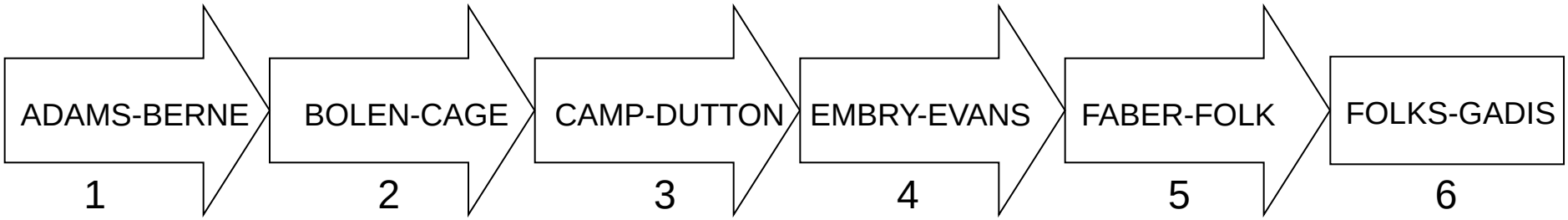
Tamanho do Bloco

- 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

Foco 2

- 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)



visão lógica

chave	bloco
BERNE	1
CAGE	2
DUTTON	3
EVANS	4
FOLK	5
GADIS	6

<u>Índice de 1 nível</u>

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

Acesso Seqüencial Indexado

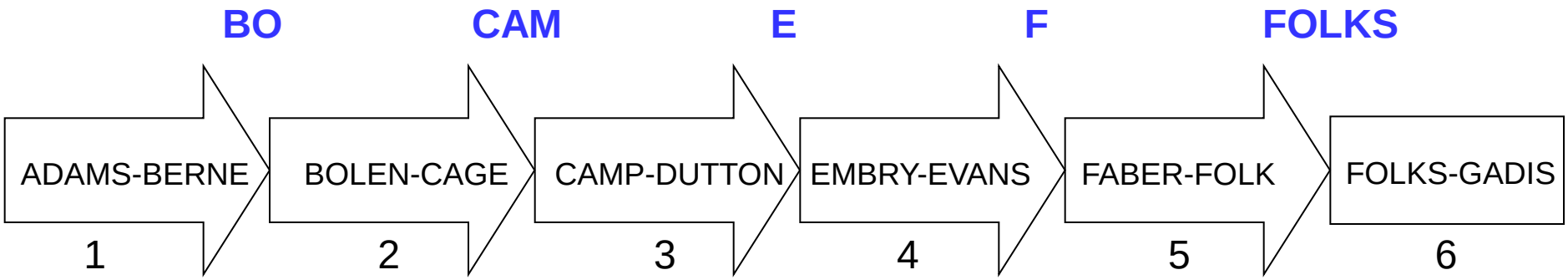
- Combina
  - registros ordenados fisicamente pela chave (*sequence set*)
  - índice simples para referenciar os blocos
- Restrição
  - a organização em tabela implica que o índice cabe na memória principal
    - busca binária no índice
    - atualização do índice em RAM

Árvore-B$^+$

Profa. Dra. Cristina Dutra de Aguiar Ciferri

Separadores

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



Separadores

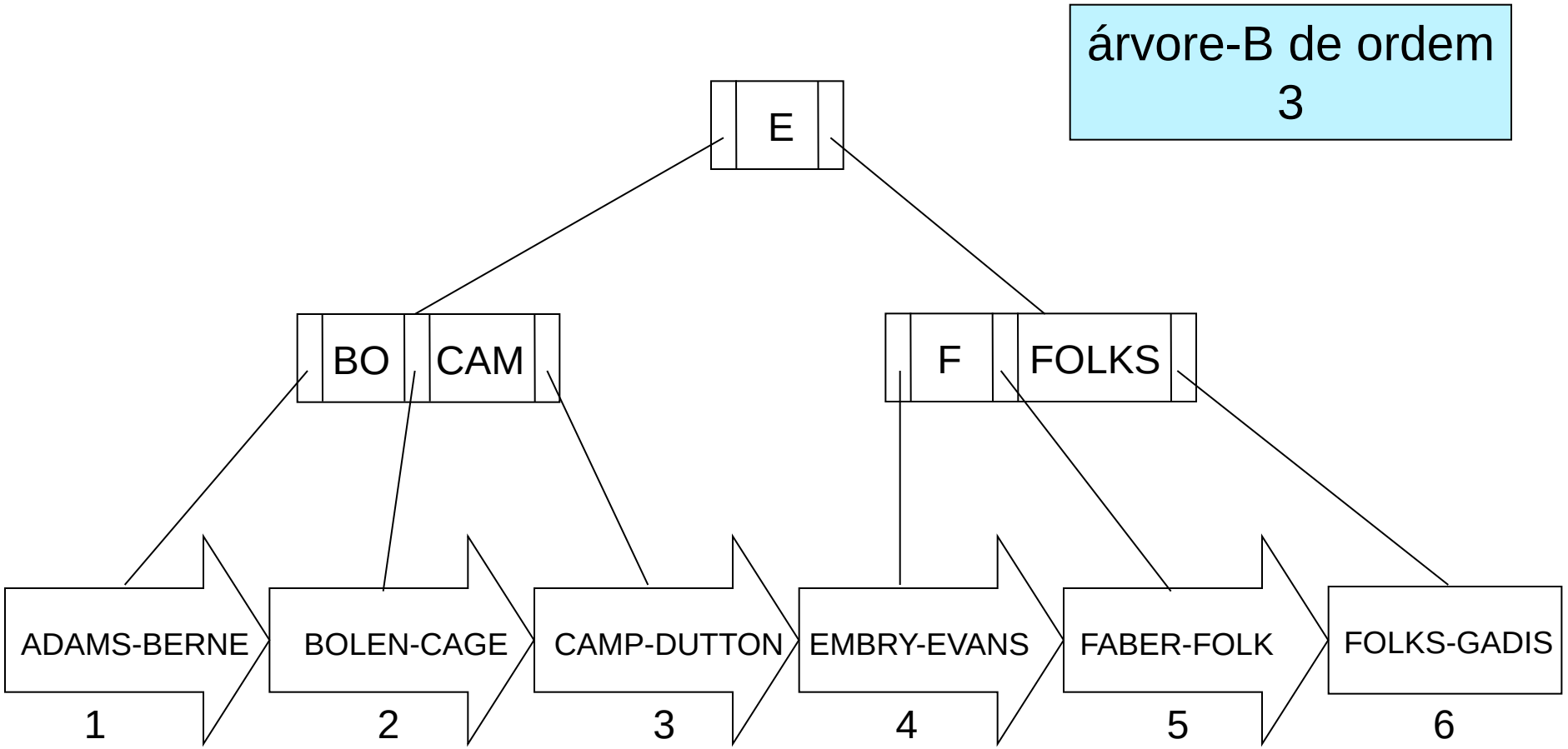
- Desafio
  - escolher o menor separador para utilizar no índice
- Tabela de decisão

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

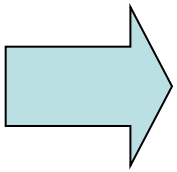
- Estrutura híbrida
  - chaves
    - organizadas como árvore-B (i.e., *index set*)
  - nós folhas
    - consistem em blocos de *sequence set*
- Pré-fixada simples
  - armazena na árvore as cadeias separadoras mínimas entre cada par de blocos

Árvore-B+ Pré-Fixada



Manutenção

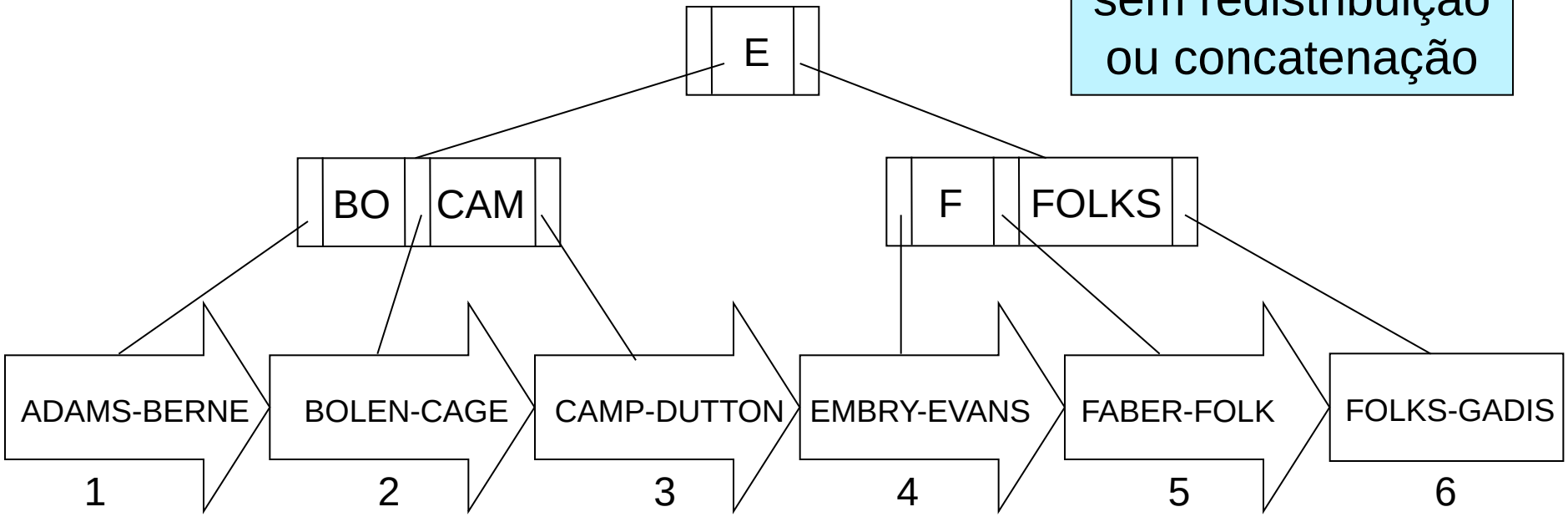
- Cenários
  - inserção
  - remoção
  - *overflow*
  - *underflow*



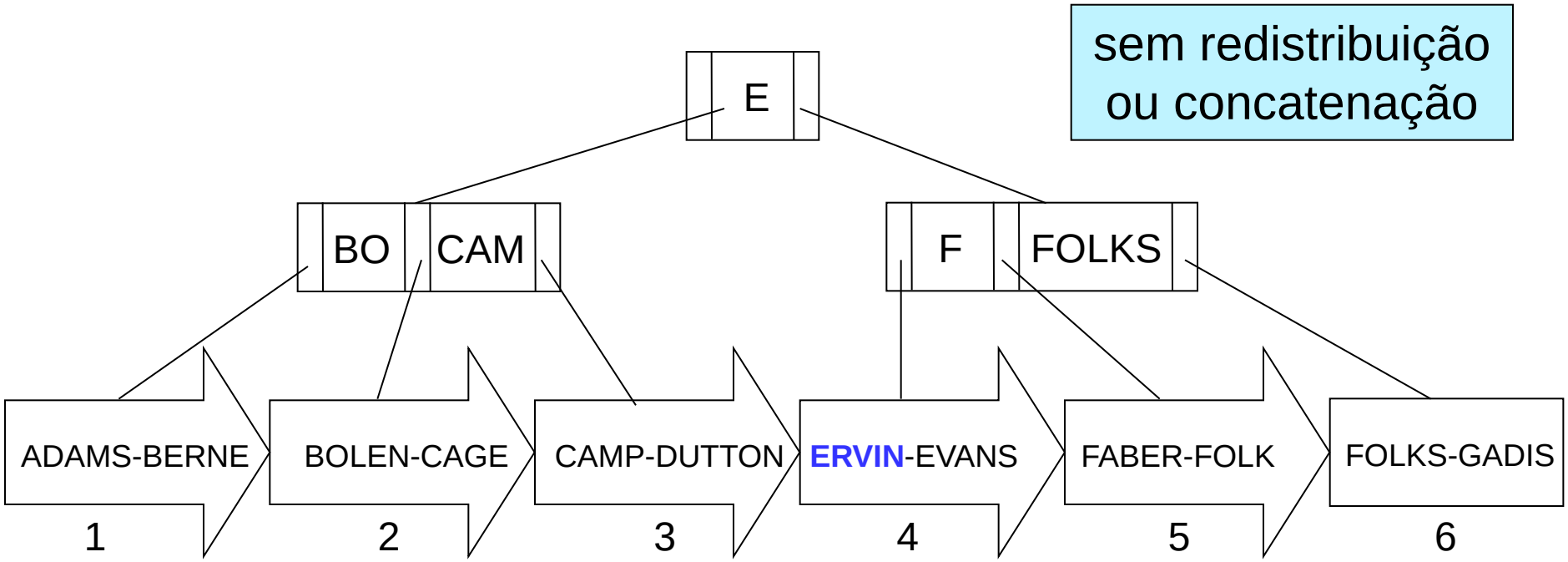
- Efeitos colaterais
  - *sequence set*
  - árvore-B+

Remoção de EMBRY

sem redistribuição
ou concatenação

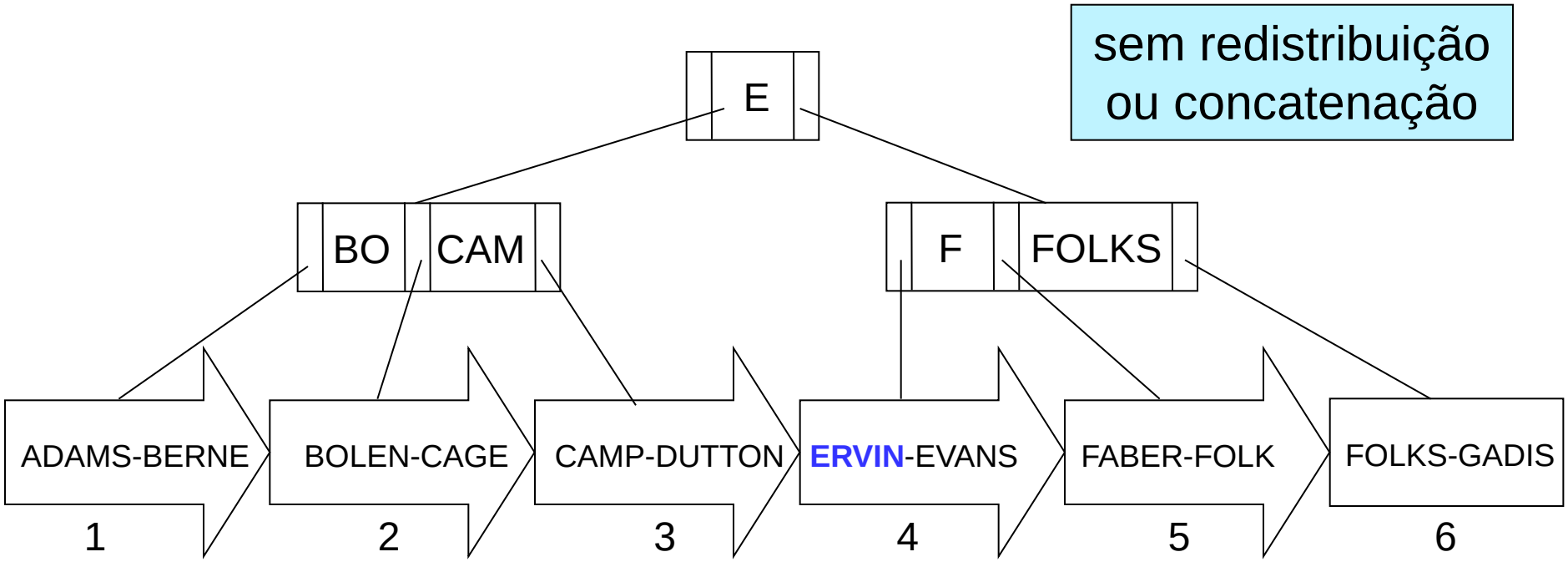


Remoção de EMBRY



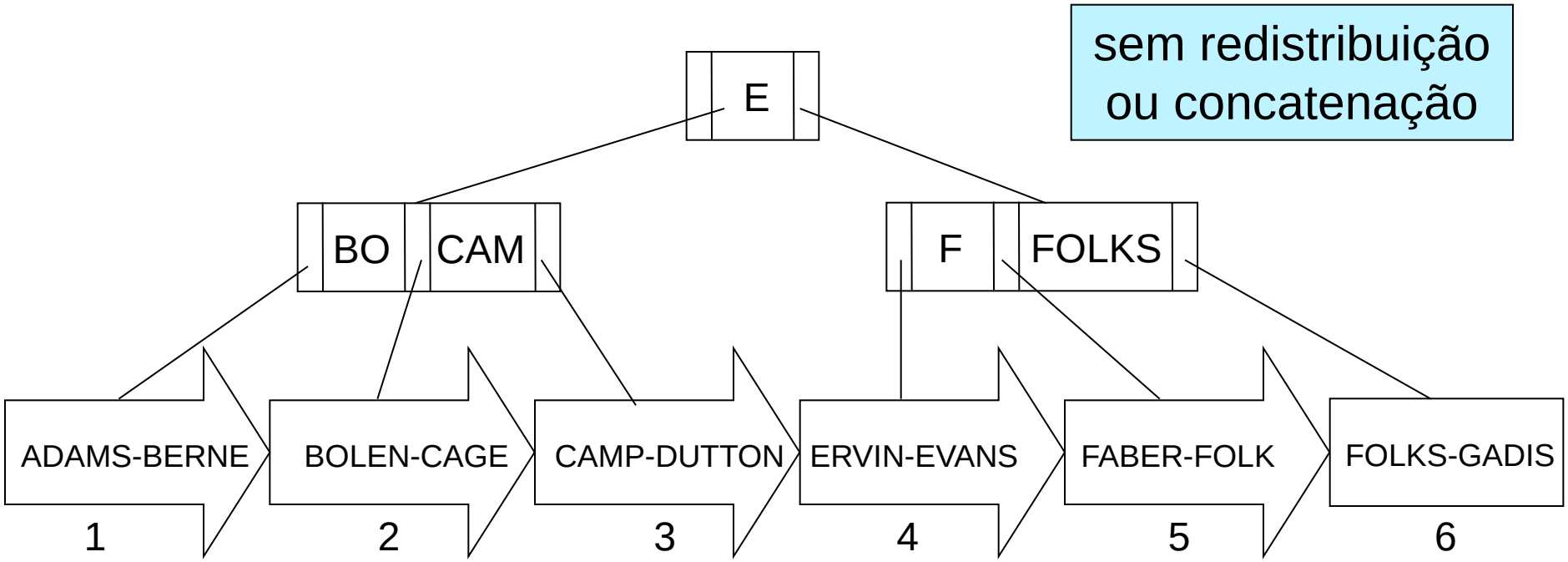
- Efeito no *sequence set*
  - limitado a alterações no bloco 4

Remoção de EMBRY

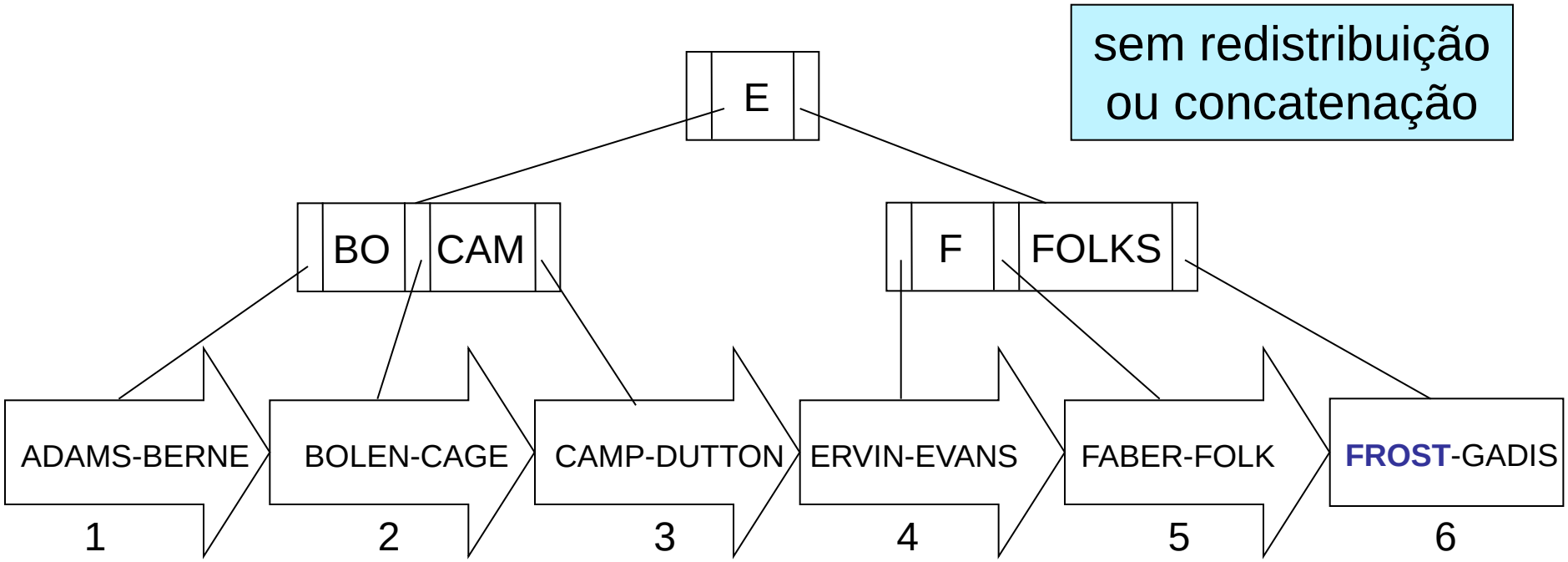


- Efeito na árvore-B+
  - nenhum: E é uma boa chave separadora

Remoção de FOLKS

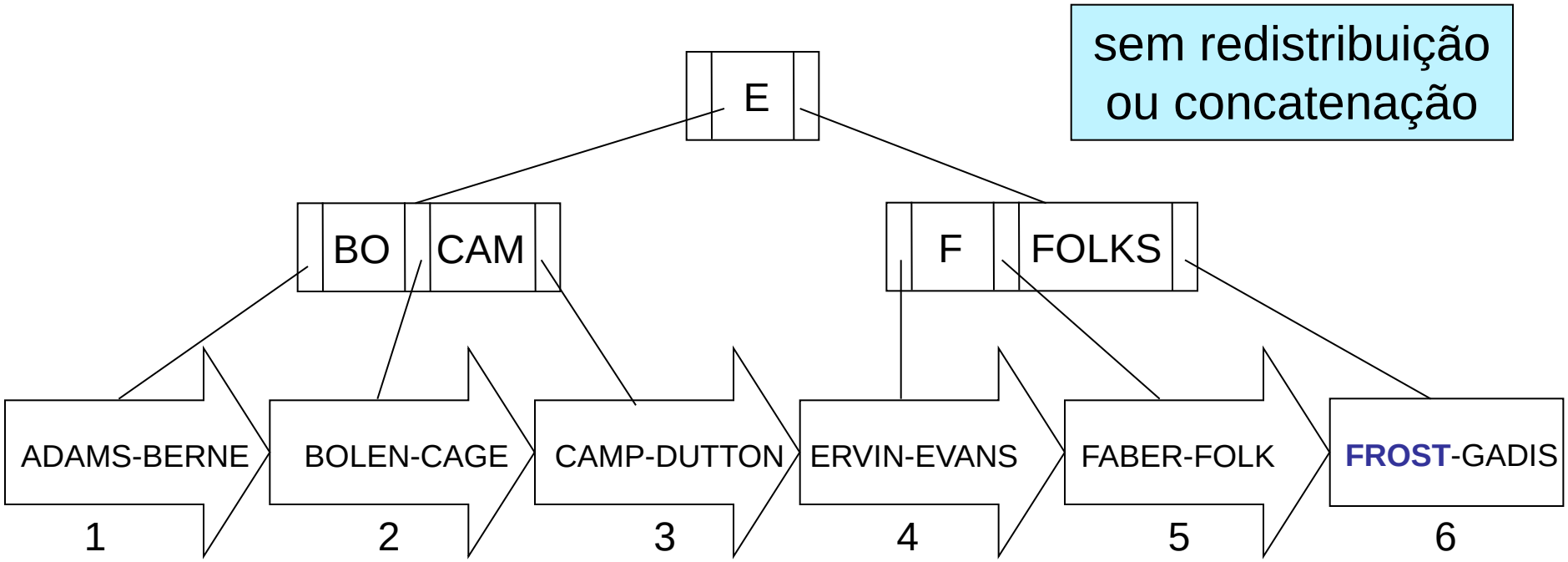


Remoção de FOLKS



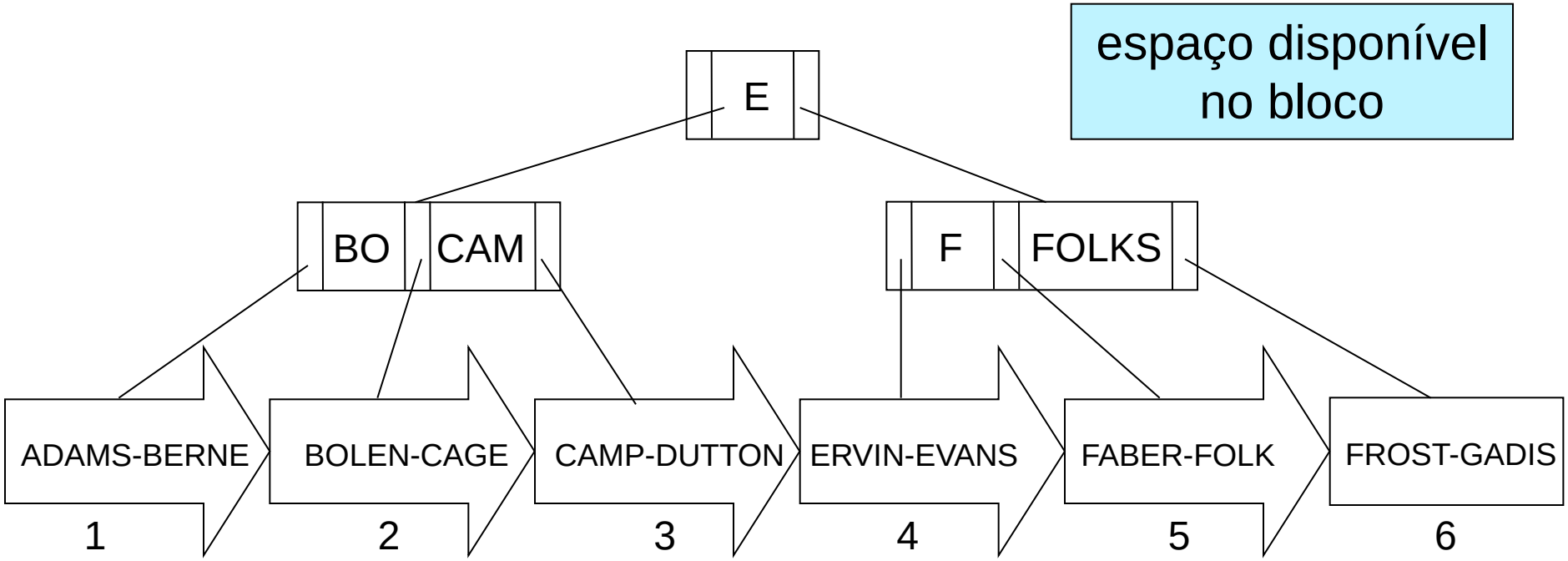
- Efeito no *sequence set*
  - limitado a alterações no bloco 6

Remoção de FOLKS

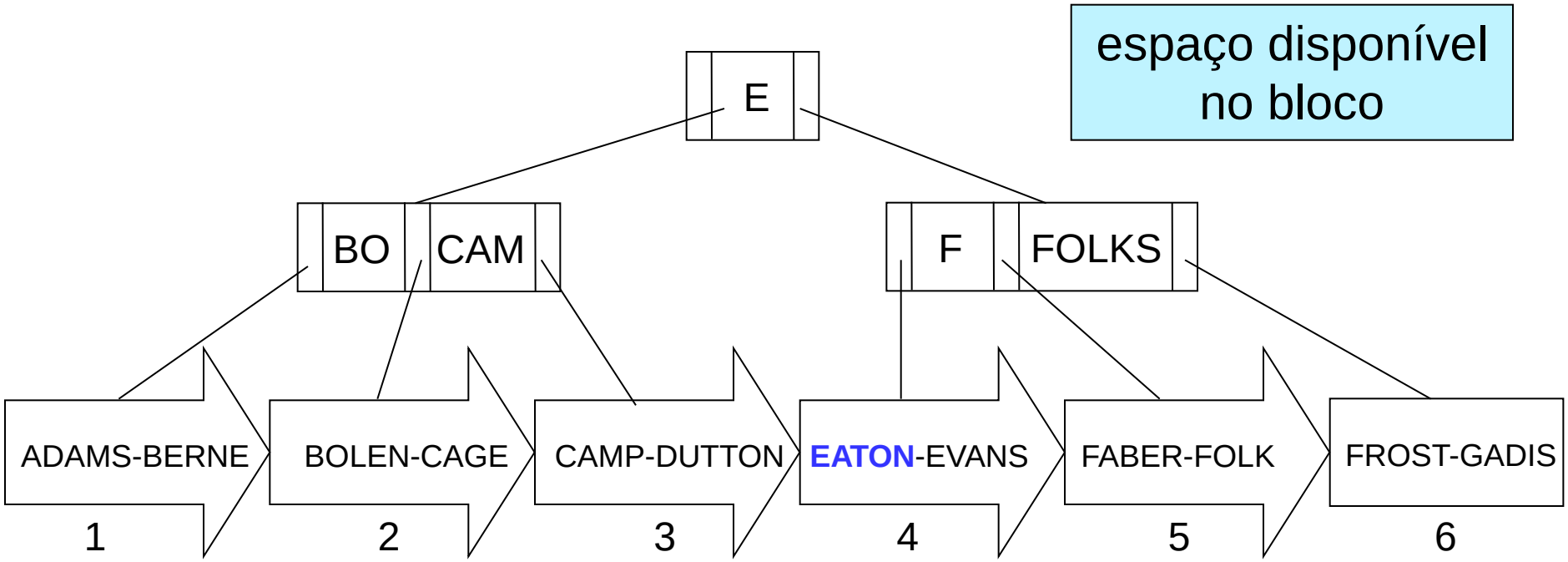


- Efeito na árvore-B+
  - nenhum: custos elevados

Inserção de EATON

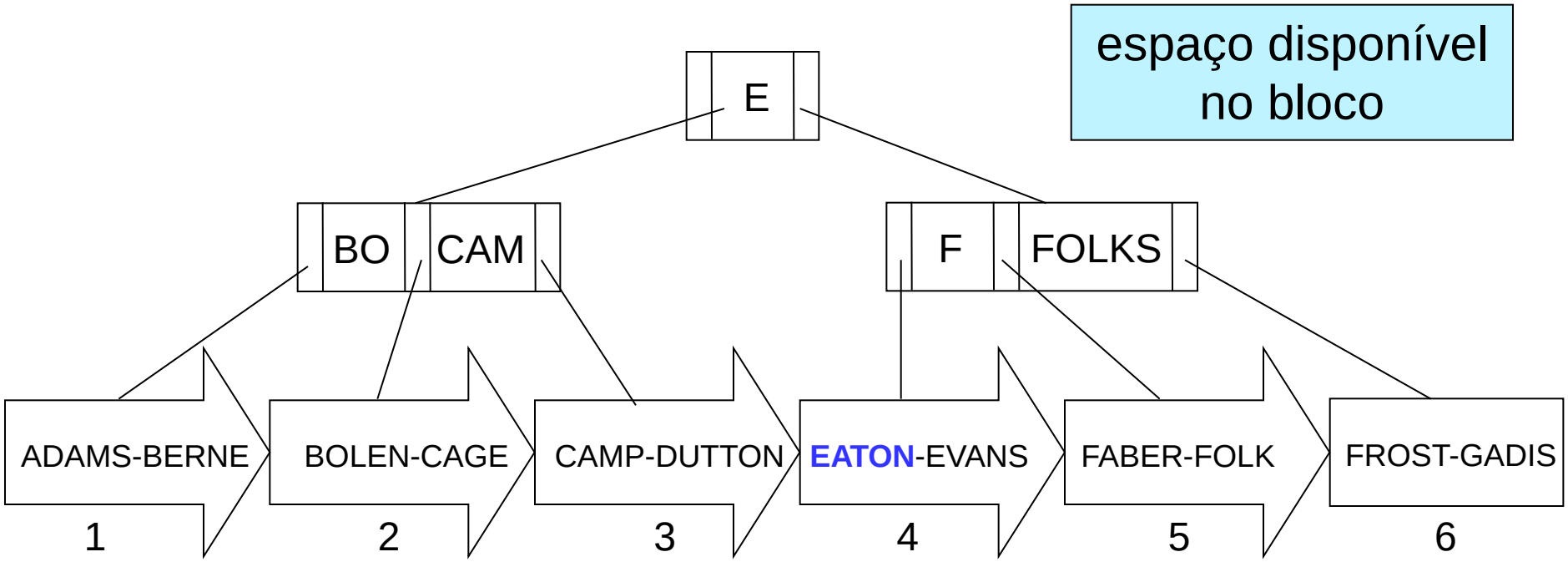


Inserção de EATON



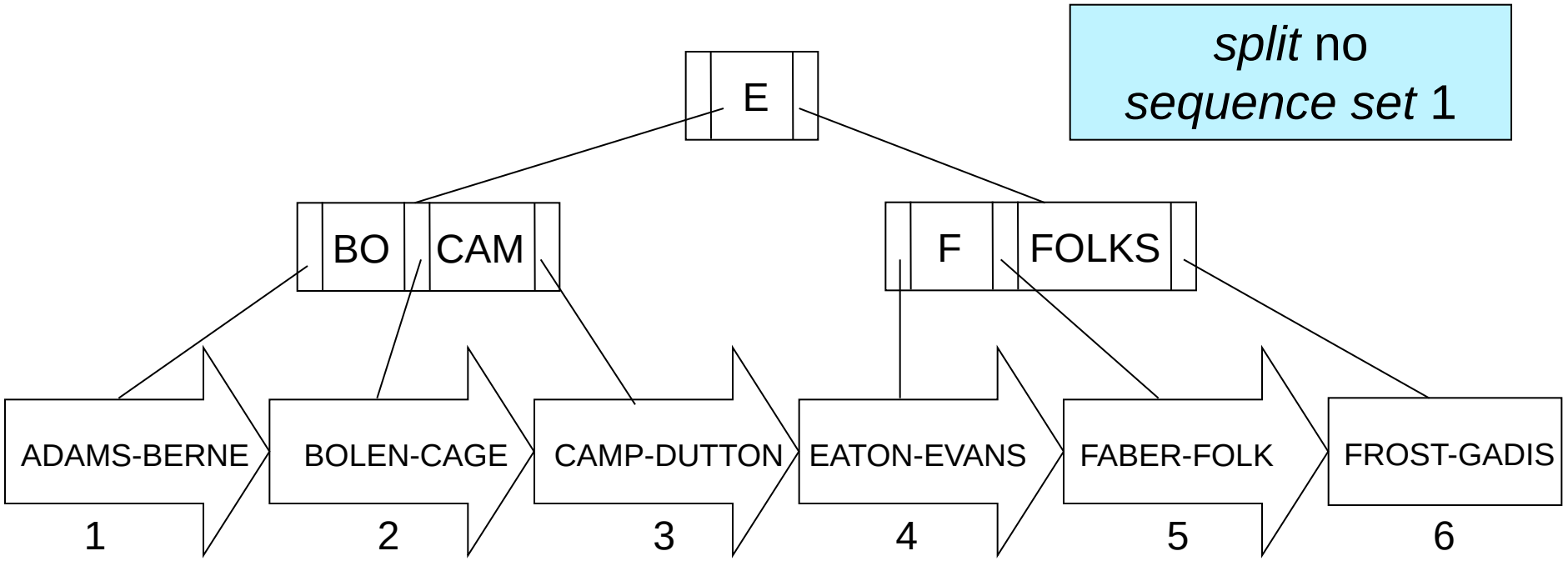
- Efeito no *sequence set*
  - limitado a alterações no bloco 4

Inserção de EATON

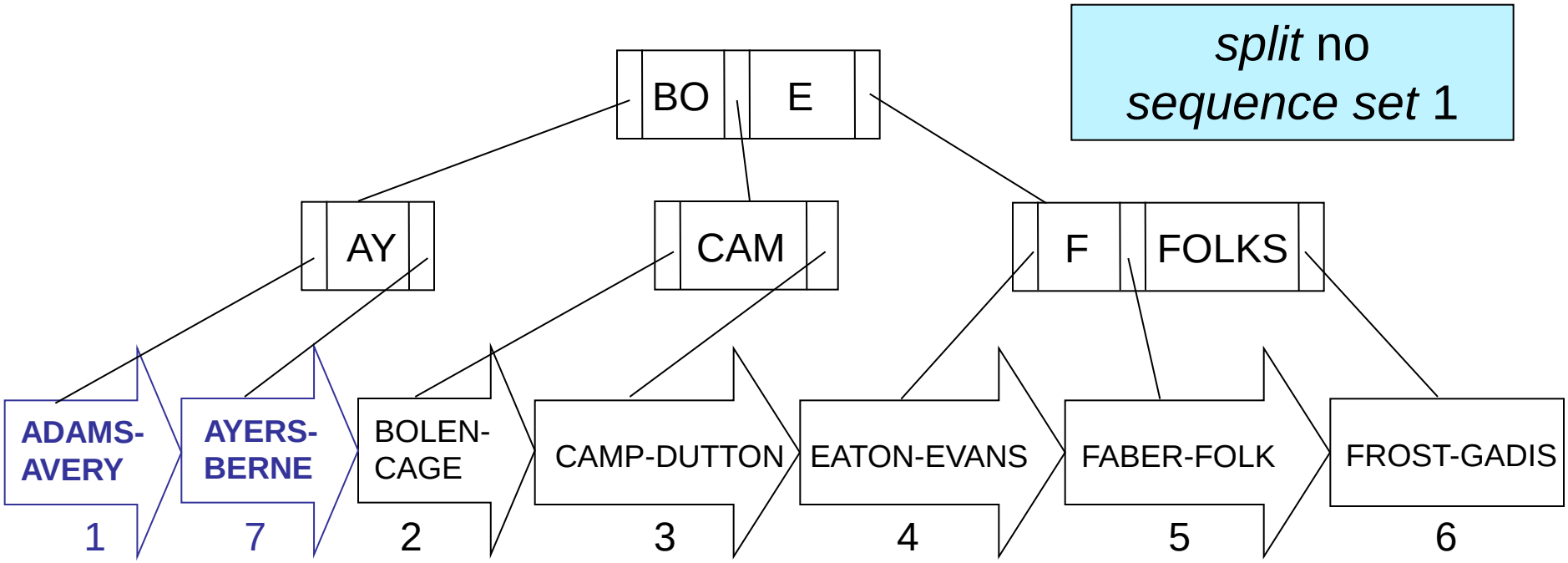


- Efeito na árvore-B+
  - nenhum: E é uma boa chave separadora

Inserção de AVERY

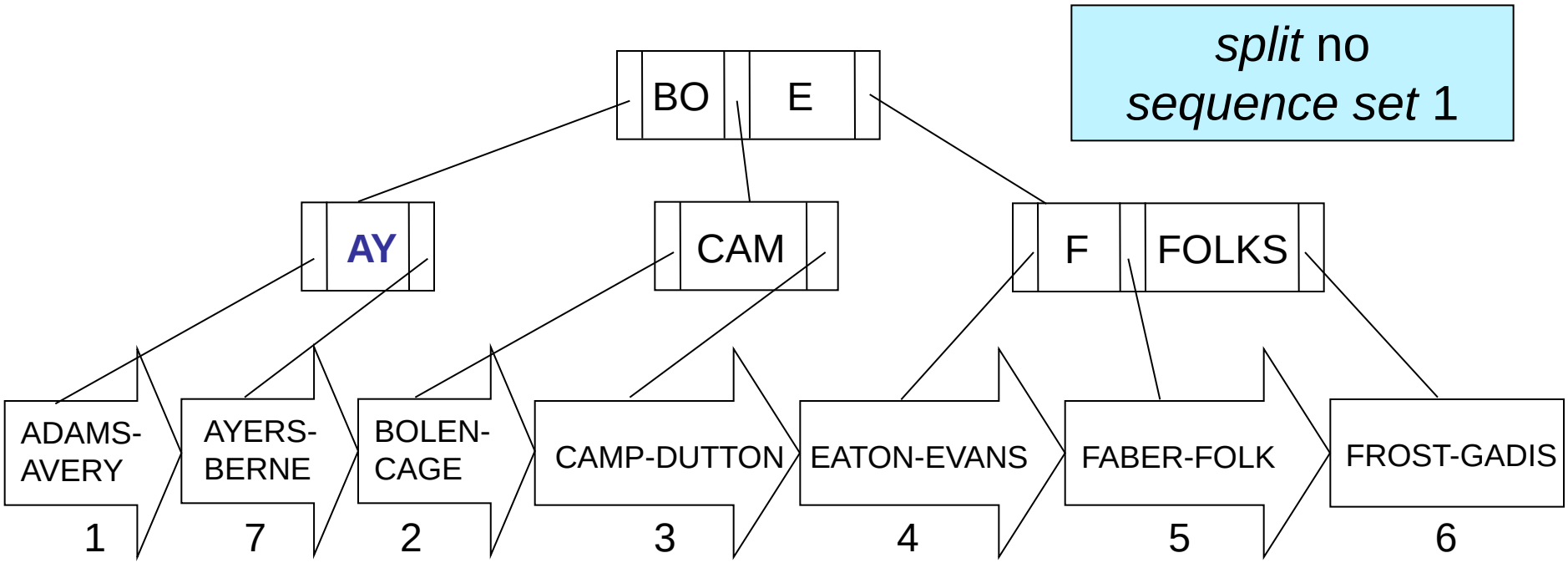


Inserção de AVERY



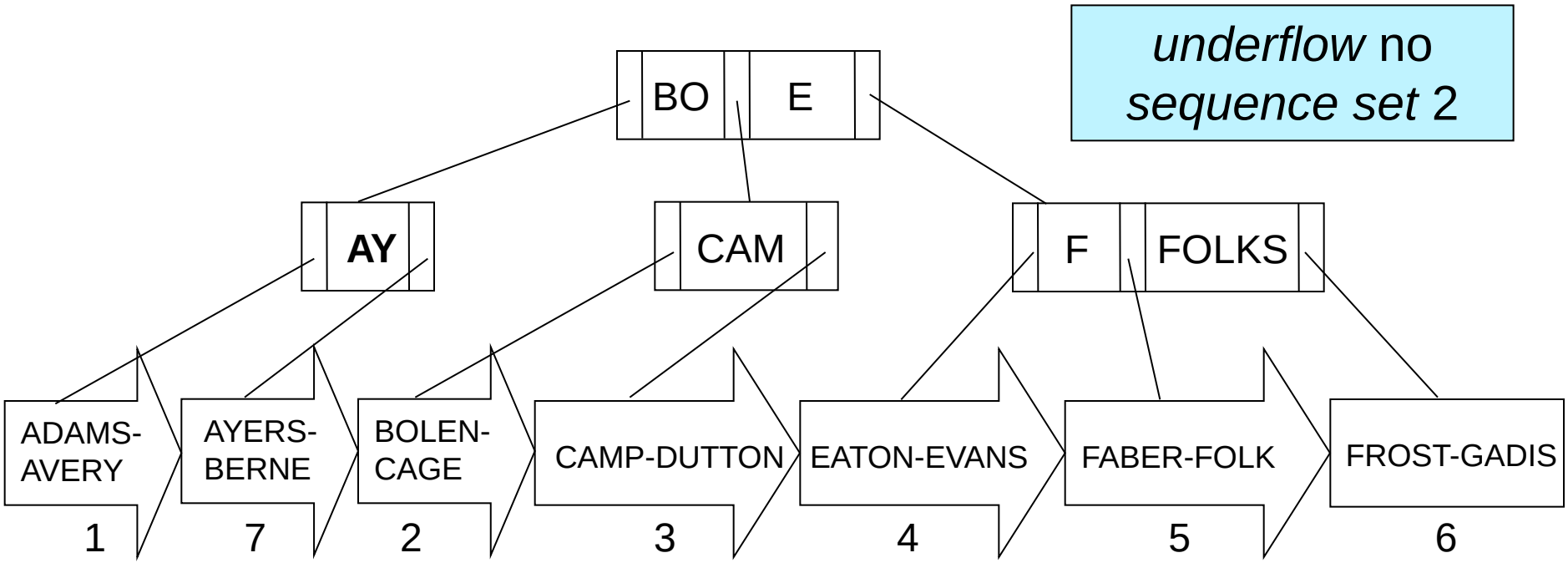
- Efeito no *sequence set*
  - dados do bloco 1 + AVERY distribuídos entre os blocos 1 e 7

Inserção de AVERY

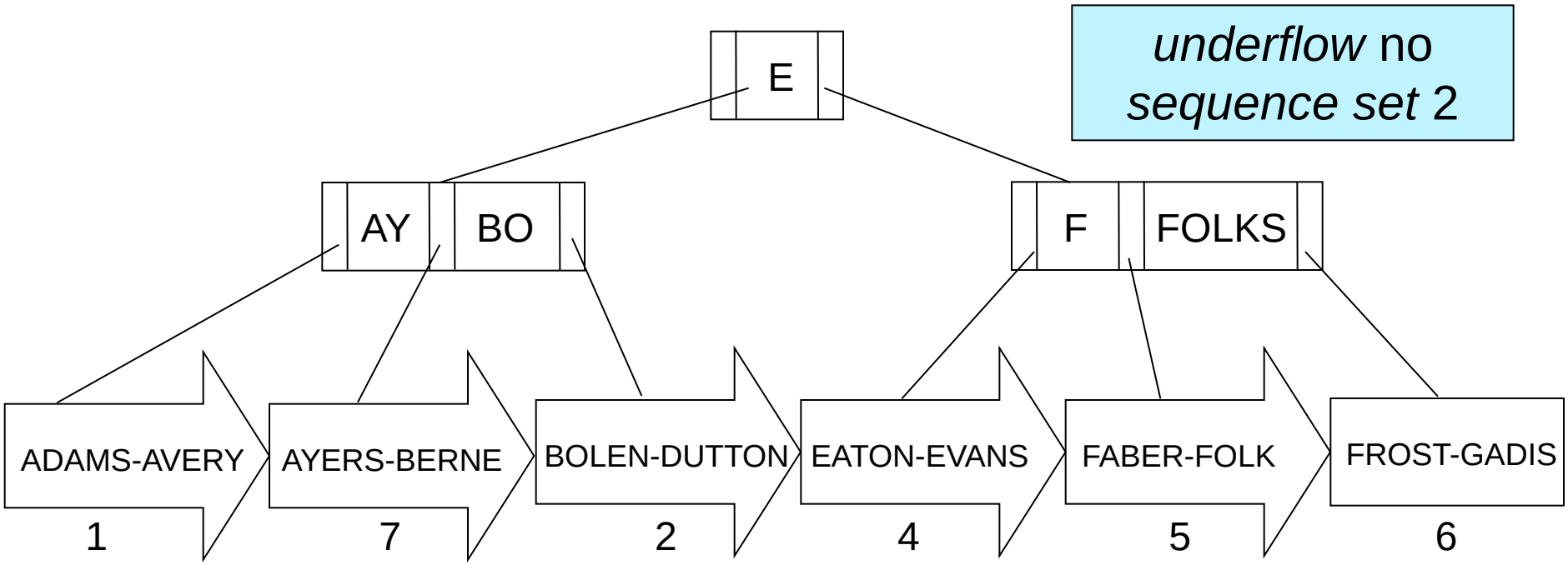


- Efeito na árvore-B+
  - separador adicional AY

Remoção de CAEL

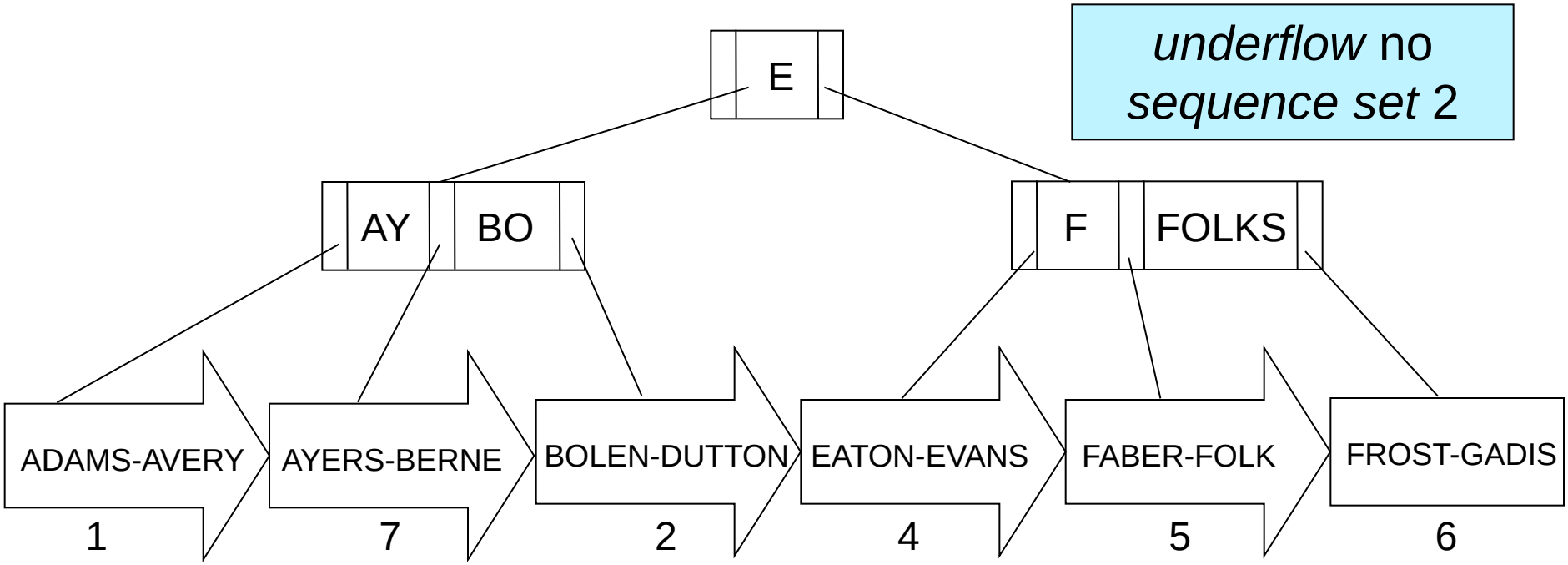


Remoção de CAEL



- Efeito no *sequence set*
  - concatenação dos blocos 2 e 3

Remoção de CAEL



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

Inserção e Remoção

- Primeiro passo: *Sequence Set*
  - inserir ou remover o dado
  - tratar, caso necessário
    - *split*
    - contatenação
    - redistribuição

alterações são sempre realizadas a partir do arquivo de dados

Inserção e Remoção

- 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

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

Observações Adicionais

- Tamanho físico de um nó no índice (i.e., árvore-B$^+$) = Tamanho físico de um bloco no *sequence set*

- Uso de um mesmo arquivo para armazenar os blocos do índice e os blocos do *sequence set*
  - evita *seeks* entre dois arquivos separados

Exercício de Árvore-B$^+$

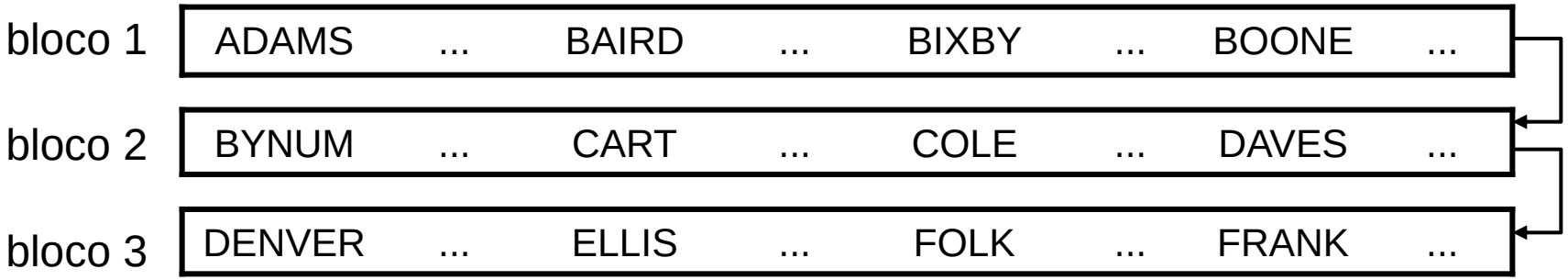
Profa. Dra. Cristina Dutra de Aguiar Ciferri

Características

- Árvore-B$^+$
  - ordem: 3
- *Sequence Set*
  - número máximo de registros: 2
  - número mínimo de registros: 1
    - *underflow*: 0 registros

Exercícios

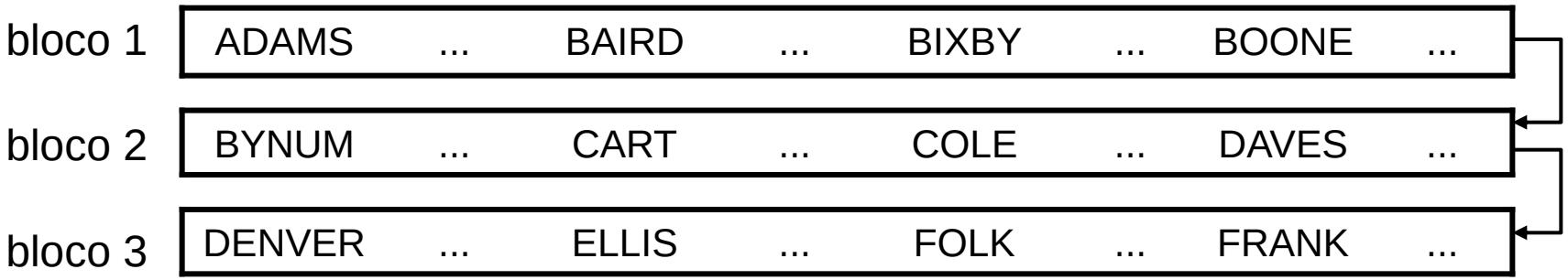
1. Quais os separadores dos *sequence sets*?



2. Construa a árvore-B$^+$
3. Realize as seguintes operações
   a) inserção de CARTER
   b) inserção de DRAG
   c) remoção de BIXBY
   d) remoção de COLE

Resposta

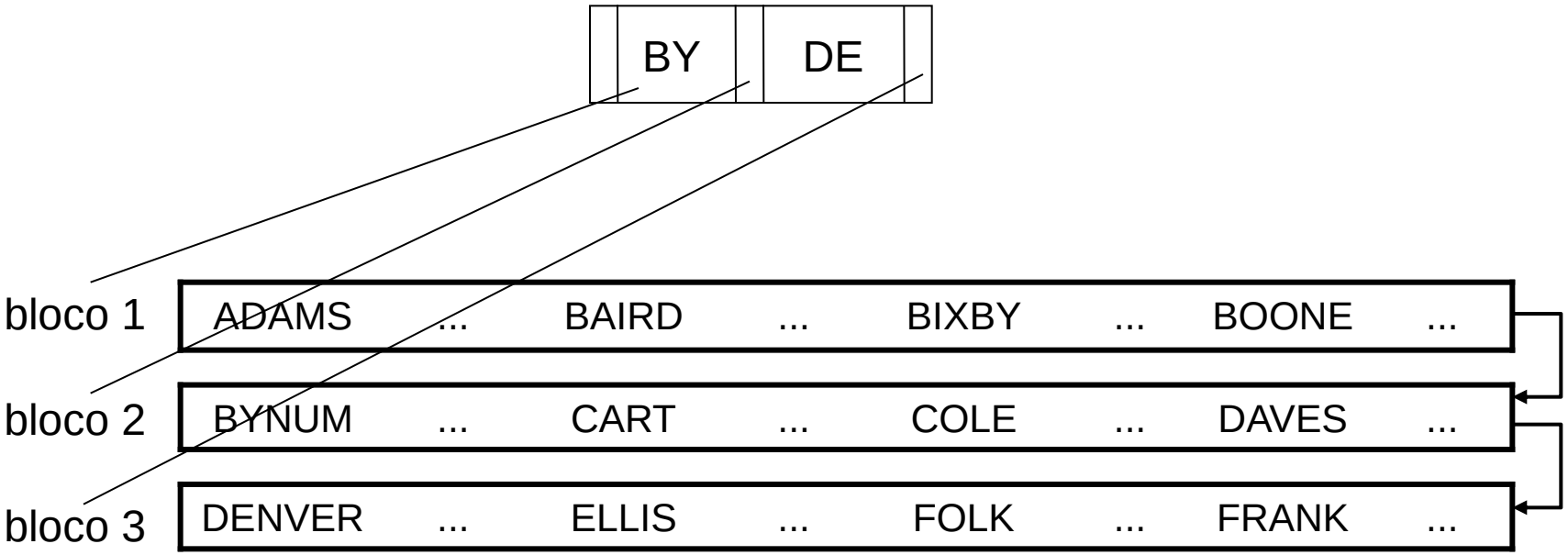
1. Quais os separadores dos *sequence sets*?



BY DE

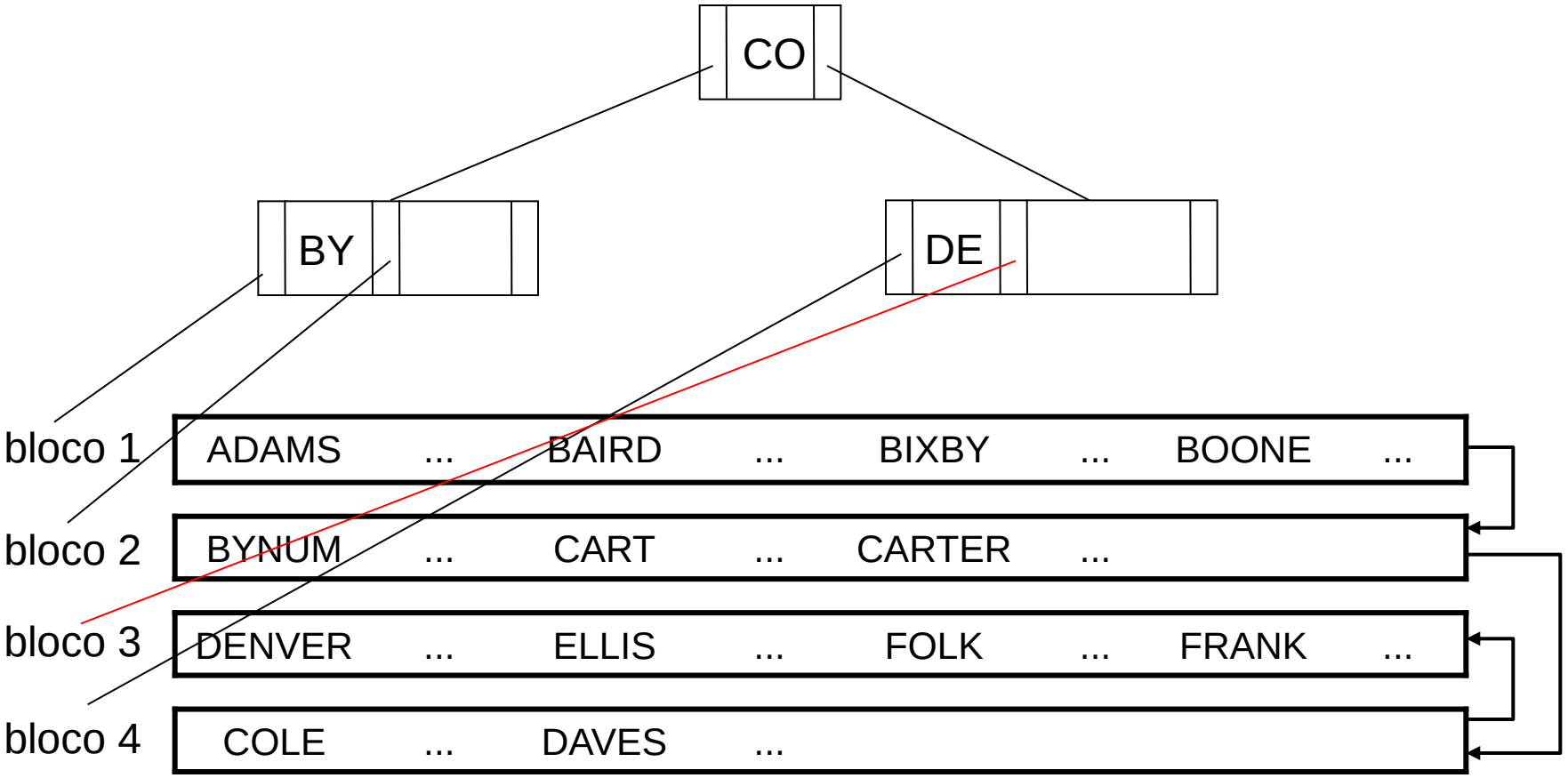
Resposta

2. Construa a árvore-B$^+$



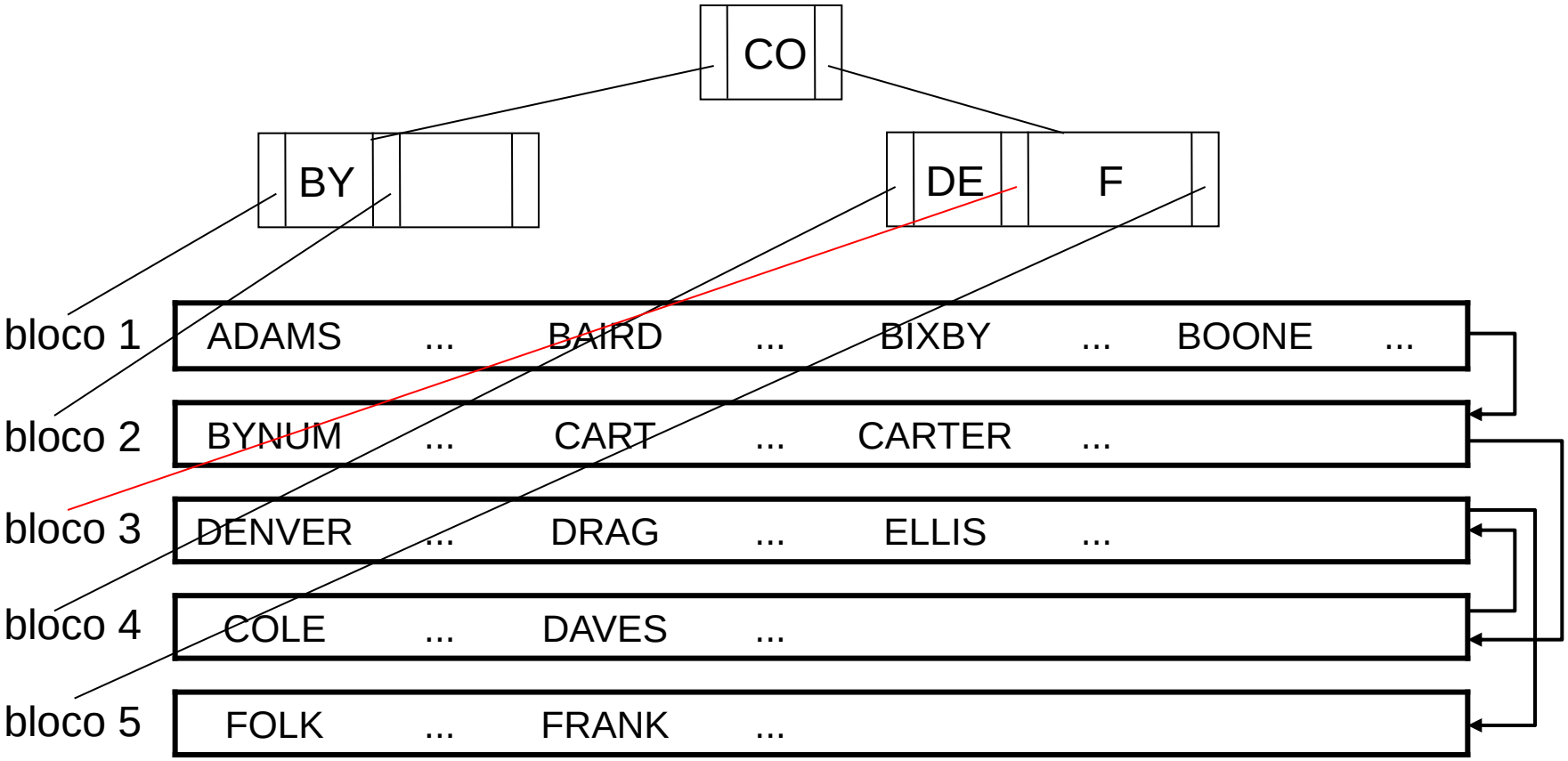
Resposta

3. inserção de CARTER



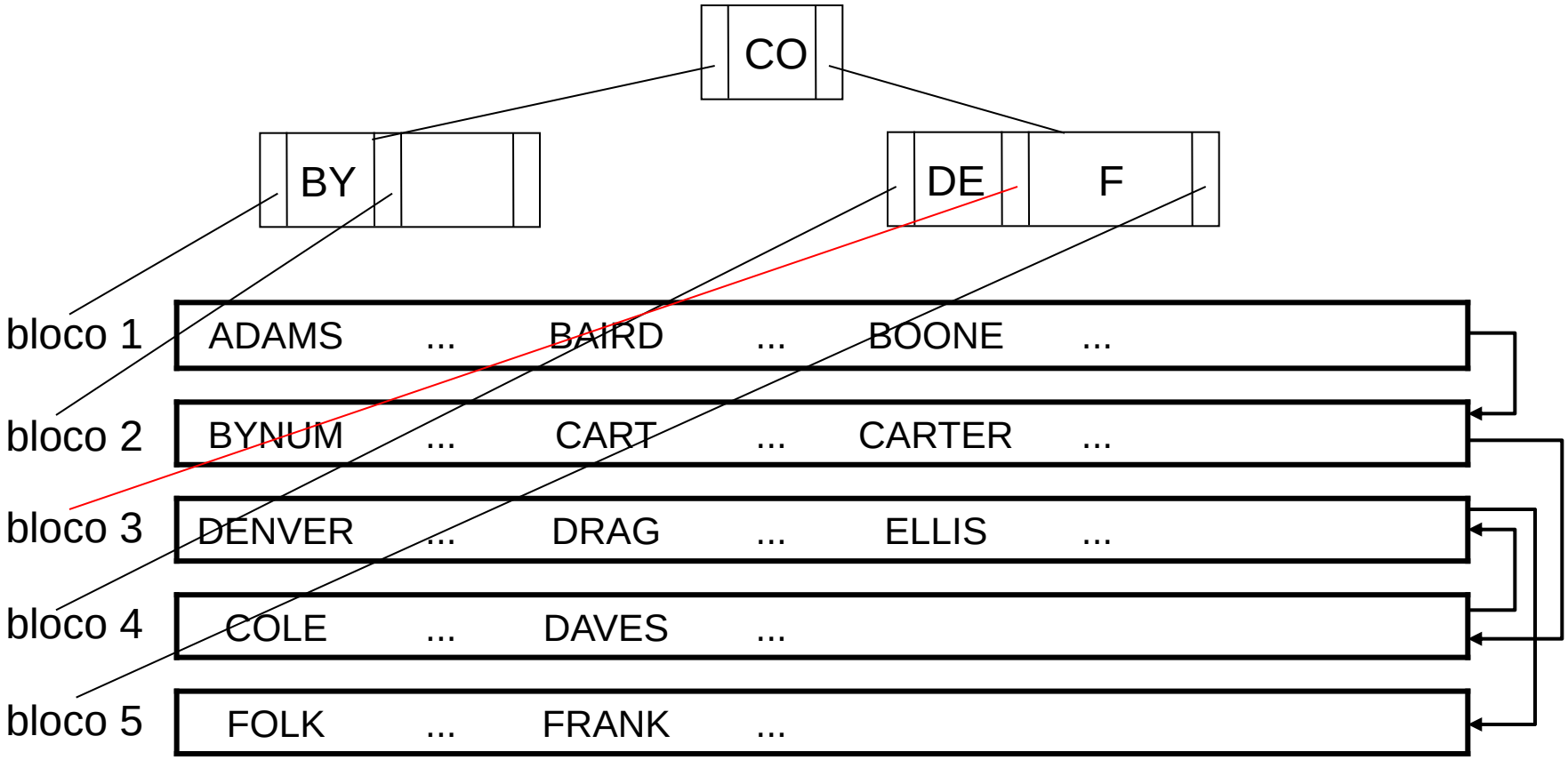
Resposta

3. inserção de DRAG



Resposta

3. remoção de BIXBY



Resposta

3. remoção de COLE