CEP-UFPR-UTFPR-PUC/Pr-UP Sistemas de Informação 12/08/2024 - 11:35:52.4
Matemática aplicada Prof Dr P Kantek (pkantek@gmail.com)
Evolução de árvores: Árvore-B (e B+) VIVXq92a V: 1.02
76506 CARLOS EDUARDO ANTAO
24CC2401 - 1 primeiro exercicio do T3 _____ / / _____

Árvore B

O conceito de árvore B foi desenvolvido por Rudolf Bayer e Edward McCreight em 1972.

Eles publicaram seu trabalho no artigo *"Organization and Maintenance of Large Ordered Indices"*, que descreve as propriedades e operações básicas das árvores B.

O trabalho de Bayer e McCreight teve um impacto significativo na área de ciência da computação, e as árvores B são agora amplamente utilizadas em bancos de dados e outros sistemas de software.

As árvores B são um tipo de árvore de busca balanceada que pode armazenar um grande número de chaves e valores de maneira eficiente. Elas são mais eficientes do que as árvores de busca binárias tradicionais, pois podem ter mais filhos por nó. As árvores B são usadas em uma variedade de aplicações, incluindo bancos de dados, sistemas de arquivos e sistemas de roteamento.

Eles não explicaram o significado do nome, só nos resta especular que esse B seja de Balanceada. Se você se lembrar das árvores binárias de pesquisa, deve recordar o problema que existe quando as chaves vem (quasi) ordenadas na entrada: a árvore gerada é desbalanceada e pode chegar a ter um desempenho terrível.

Outra qualidade da árvore-B é que ela tem grau k (k filhos por nodo). Em geral este valor é escolhido de modo a otimizar a blocagem de dados no arquivo de memória secundária. De qualquer modo este k modifica o desempenho, que lembrando é proporcional a $\log_k n$, onde n é a quantidade de chaves.

Como toda árvore, ela tem uma raiz que tem de 0 a $2k-1$ chaves e consequentemente tem de 0 a $2k$ filhos. Cada um dos filhos, tem de $k-1$ até $2k-1$ chaves de 0 até $2k$ filhos. Note que a raiz é o único nodo que sofre processamento especial, necessário para quando o arquivo tem uma única chave (por exemplo).

Um caso prático real Suponha um colégio com milhares de alunos, que têm seus dados guardados em um banco de dados. Por hipótese, os dados de cada aluno vão sendo recolhidos quando eles fazem a matrícula, pelo que a ordem física dos dados no arquivo é a data da matrícula. Parece óbvio assinalar que um determinado conjunto de dados (um arquivo) só pode ter uma única ordem física em um determinado instante. Mas nada impede que existam muitas ordens lógicas diferentes para o mesmo arquivo. Tais ordens lógicas são providenciadas por estruturas muito parecidas a uma árvore-b. Neste exemplo, poderíamos ter índices por nome (ordem alfabética), curso, turma, endereço etc.

Vai-se descrever aqui uma árvore-b que dará acesso aos dados por código do aluno.

Suponha uma árvore-b de nome M que providenciará este acesso. Vamos descrevê-la com $k=4$, que é muito pouco para aplicações reais, mas fica razoável para trabalhar com este exemplo. De acordo com a definição, os nodos de M têm entre 3 e 7 códigos de aluno. Para esta dimensão, o nodo de M terá 17 colunas:

coluna 0 um asterisco nesta posição indica que este é o nodo raiz da árvore. Conta como coluna 0 porque na verdade esta informação não está fisicamente aqui, está em outro lugar.

coluna 1 indicativo de folha (1=sim; 0=não)

coluna 2 quantidade de chaves válidas neste nodo (nem sempre o valor inválido é zero)

cols 3 a 9 chaves, sempre em ordem crescente

cols 10 a 17 endereços dos filhos associados a cada chave. Só estão preenchidas para nodos não folha.

Veja um exemplo:

```
    1 2    3    4    5    6    7    8    9 10 11 12 13 14 15 16 17
  +-------------------------------------------------------------------
 1| 1 5    3    6    7    8   10    0    0
 2| 0 4   11   20   32   44    0    0    0  1 20 15  7 12
 3| 1 4  128  131  132  134    0    0    0
 4| 1 5  149  150  151  153  160    0    0
 5| 1 6   65   73   76   79   80   83    0
 6| 1 5  182  183  188  189  191    0    0
 7| 1 3   34   35   37    0    0    0    0
 8| 1 4  106  109  111  114    0    0    0
 9| 1 6  136  137  138  140  143  145    0
*10| 0 2   64  127    0    0    0    0    0  2 17 11
 11| 0 6  135  146  162  171  181  192    0  3  9  4 14 21  6 18
 12| 1 7   46   49   50   55   56   57   63
 13| 1 4   88   89   91   94    0    0    0
 14| 1 5  164  166  167  168  169    0    0
 15| 1 6   21   22   23   28   29   30    0
 16| 1 7  117  118  120  122  123  125  126
 17| 0 4   84   95  105  116    0    0    0  5 13 19  8 16
 18| 1 5  193  195  197  199  200    0    0
 19| 1 5   96   98  100  102  103    0    0
 20| 1 4   14   15   17   19    0    0    0
 21| 1 3  174  175  177    0    0    0    0
```

Antes de continuar no exercício, você deve tentar entender o funcionamento da árvore-b.

- Desenhe a árvore
- identifique a necessidade da coluna 1. Idem para a coluna 2
- Qual a altura da árvore ?
- Quando e como a altura da árvore cresce ?

Agoram suponha que o elemento 187 é incluído. Qual a linha;coluna da matriz acima que é modificada, ou seja onde o elemento recém incluido entra ? Localize-o e veja:

```
    1 2    3    4    5    6    7    8    9 10 11 12 13 14 15 16 17
  +-------------------------------------------------------------------
 1| 1 5    3    6    7    8   10    0    0
 2| 0 4   11   20   32   44    0    0    0  1 20 15  7 12
 3| 1 4  128  131  132  134    0    0    0
 4| 1 5  149  150  151  153  160    0    0
 5| 1 6   65   73   76   79   80   83    0
 6| 1 6  182  183(187)188  189  191    0
 7| 1 3   34   35   37    0    0    0    0
 8| 1 4  106  109  111  114    0    0    0
 9| 1 6  136  137  138  140  143  145    0
*10| 0 2   64  127    0    0    0    0    0  2 17 11
 11| 0 6  135  146  162  171  181  192    0  3  9  4 14 21  6 18
 12| 1 7   46   49   50   55   56   57   63
 13| 1 4   88   89   91   94    0    0    0
 14| 1 5  164  166  167  168  169    0    0
 15| 1 6   21   22   23   28   29   30    0
 16| 1 7  117  118  120  122  123  125  126
 17| 0 4   84   95  105  116    0    0    0  5 13 19  8 16
 18| 1 5  193  195  197  199  200    0    0
 19| 1 5   96   98  100  102  103    0    0
 20| 1 4   14   15   17   19    0    0    0
 21| 1 3  174  175  177    0    0    0    0
```

Como voce pode ver ele está na linha ____ e coluna ____. Algumas características notáveis da árvore-b

- Figurinha carimbada no mundo do software: usam B-trees: SYBASE e do Harbour (antigo Clipper). Também são usados no Oracle, MySQL, PostgreSQL, Microsoft SQL e DB2, da IBM.
- muitas estruturas usam a árvore-b+. O + significa que há uma ligação adicional entre as folhas, para permitir o processamento sequencial dos dados.
- Existem nodos folha e não-folha. Os endereços dos filhos (colunas 10 a 17) só existem para os nodos que os têm (nodos não folhas).
- Os códigos de aluno (colunas 3 a 9 não mostram, para diminuir o ruído), mas ao lado de cada código está o endereço físico real dos dados deste aluno, que afinal é o dado que nos interessa nesta busca.
- Os códigos (colunas 3 a 9) estão em ordem crescente. Isto é fundamental e não pode ser modificado.
- Perceba como, em média, apenas a metade dos códigos está preenchida. Isto é necessário para permitir inclusão de novos dados. Sinaliza também um certo desperdício (overhead) de área em disco que qualquer método de acesso consome.
- Quando um nodo tem todas as chaves preenchidas e chega uma nova, que deveria (deverá) entrar nele, o nodo sofre um processo chamado split e ele é transformado em 2 nodos (cada um com metade dos espaços vazios). Note que o número máximo de chaves em qualquer nodo é sempre ímpar ($2n-1$) o que implica em que o elemento do meio sobe para o *pai* do nodo que está sendo quebrado. A metade anterior vai para um nodo *splitado* e a metade posterior vai para o outro nodo *splitado*.
- Quando as chaves vão sendo excluídas, e um nodo passa a ter menos do que a metade das chaves (o que é proibido) ele é juntado com um nodo irmão e ambos passam a ser um único nodo. Este processo recebe o nome de merge. Não é tão simples assim, há diversos casos particulares, mas a ideia é essa.

Mais um exemplo. Suponha a inclusão do número 26. Em que linha;coluna ele entra ?

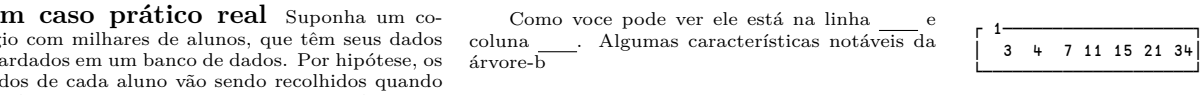
Algoritmo de Inclusão em árvore B

```
1: função INCLUA-BTREE (K:inteiro)
2: X ← raiz da árvore B
3: enquanto nodo X não é folha
4:     X ← filho correto onde K deve ser incluído
5: fim{enquanto}
6: se nodo X está cheio
7:     faça split do nodo X
8:     INCLUA-BTREE (K)
9:     abandone
10: senão
11:     inclua K no nodo X mantendo as chaves ordenadas
12: fim{se}
```

Algoritmos Estão disseminados pela comunidade. Deixo de apresentá-los aqui, pois são bastante específicos, complexos até, e mais importante do que implementá-los é entender-lhes o funcionamento. Até porque se você tiver que usar uma árvore b que não seja para aprender como ela funciona, recomendo fortemente que você use pacotes prontos, já testados e confiáveis.

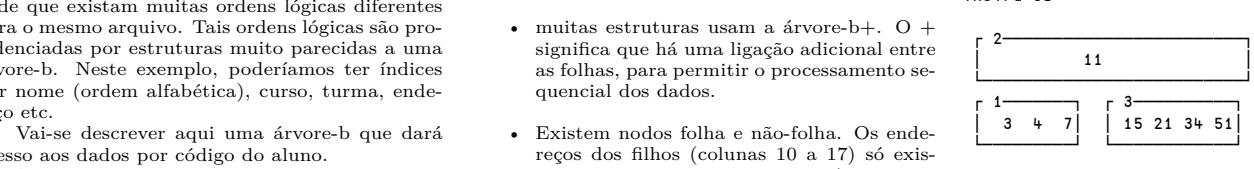
Um exemplo, passo a passo Seja uma árvore-b com $k=4$. Pela definição os nodos terão entre 3 e 7 chaves.

```
criabtree
insira 4, 7, 21, 11, 34, 15, 3
```

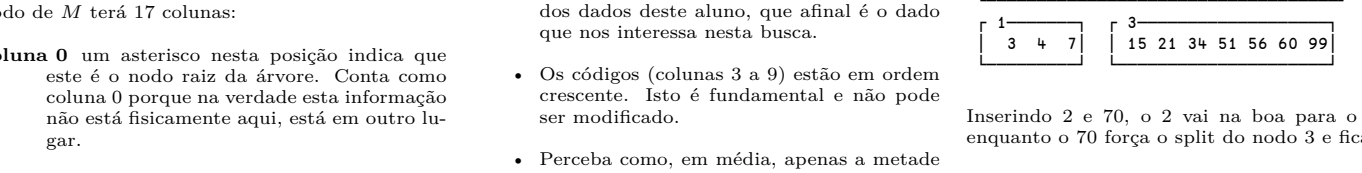


Agora, vamos inserir uma nova chave, o que vai provocar o split

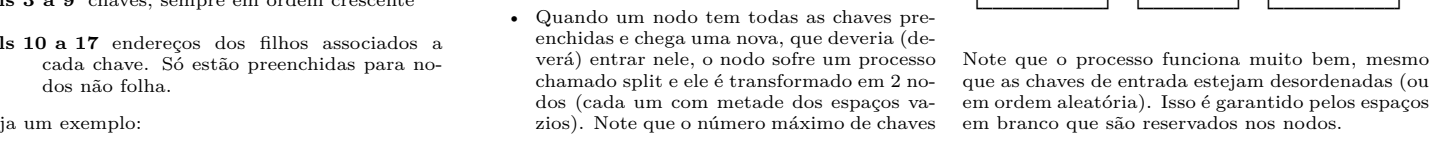
```
insira 51
```



Inserindo 60, 99, 56, fica



Inserindo 2 e 70, o 2 vai na boa para o nodo 1 enquanto o 70 força o split do nodo 3 e fica



Note que o processo funciona muito bem, mesmo que as chaves de entrada estejam desordenadas (ou em ordem aleatória). Isso é garantido pelos espaços em branco que são reservados nos nodos.

Um caso quasi-real: SYBASE

O SYBASE usa árvores B para indexar as tabelas. Existem 2 tipos de árvores B: as granuladas e as não granuladas (clustered). As granuladas são árvores esparsas (nem todo registro de dados tem entrada nos índices) nas quais os registros (na tabela original) são mantidos em ordem dos valores de índice e apenas o primeiro registro em cada página de dados tem uma entrada de índice. Árvores não granuladas são densas e cada um dos registros da tabela tem 1 entrada de índice.

Uma tabela SYBASE com índice granulado exige que seus elementos sejam mantidos em ordem dentro da tabela. Assim, inclusões podem se dar em qualquer página, a depender do conteúdo do campo de índice. Tabelas SYBASE que tem apenas índices não granulados fazem as inclusões de novos elementos sempre ao final das tabelas (na última página).

Tabelas SYBASE podem ter apenas 1 índice granulado, mas podem ter até 250 índices não granulados.

Na criação de um índice granulado, os dados da tabela devem estar colocados nas páginas em ordem.

Cada entrada de índice em sybase pode ser composta por até 16 campos e ter até 256 bytes. Estes campos não precisam estar contiguos. Só não podem ser partes de campos e resultados de cálculos.

O tamanho da página varia, mas o tamanho mais comum é 2K. Existem 5 tipos de páginas em sybase, a saber:

Páginas de Dados contém registros de dados ou registros de log. Ambos são estruturalmente diferentes, mas são construídos do mesmo jeito.

Páginas de índices contém registros de índice

Páginas de texto ou imagem contém BLOBS

Páginas de alocação contém estruturas de dados usadas para gerenciar o processo de alocacão de páginas

páginas de Estatísticas contém estatística de distribuição e uso para os índices.

Página de Dados Se os registros tem tamanho fixo, não haverá tabela de deslocamento na página e o endereço de cada registro é obtido mediante simples cálculo. Neste caso, o tamanho do registro (fixo) é guardado no campo TAMANHO MÍNIMO DE REGISTRO do header.

O header da página é um conjunto de 32 bytes contendo (entre outros): número lógico da página, anterior e próxima página lógica, identificação do objeto ao qual esta página pertence, primeira entrada de registro disponível nesta página, deslocamento do espaço livre, tamanho mínimo do registro

Depois vem a área de registros, que contém um número inteiro de slots para guardar registros. Um registro não atravessa limites de páginas

Tabela de deslocamento: Um conjunto de deslocamentos desde o início da página dando o início de cada um dos registros. Este conjunto cresce do fim da página para o início e só existe quando o tamanho do registro é variável.

Cada registro na página recebe um número de registro. É um campo de 1 byte e assim restringe-se o número de registros em uma página a 256. O número do registro usado nos índices não granulados (densos) é uma combinação do número da página com o número do registro na página.

O registro pode ocupar a página inteira (2048-32=2016). Não há tamanho mínimo para o registro, mas só pode haver 256 registros na página.

Os índices SYBASE são organizados na forma de arvores B + (o + significa que há uma ligação adicional entre as folhas, para permitir o processamento sequencial dos dados).

No exemplo aqui estudado, vão-se fazer algumas considerações/simplificações descritas a seguir.

Suponha um arquivo de pessoas contendo nome, cor preferida, bicho de estimação, alimento que gosta e idade.

Construindo um arquivo SYBASE contendo estes dados e que esteja sujeito às seguintes características:

- Registros de tamanho fixo igual a 40 bytes (9+9+9+9+4).
- Um índice granulado ou esparso (clustered) por um dos campos. No exemplo que é igual para todos, o índice esparso é por cor.
- Quatro índices não granulados (densos) pelos demais atributos.
- Páginas de dados com espaço para 4 registros e fator de preenchimento de 4:1 (usar apenas 3 espaços na carga original).
- Páginas de índices com espaço para 5 chaves (nome, cor, bicho, alimento ou idade). Lembrar que este número é impar, o que indica que no *split* vai subir a terceira chave (das 5 que existem).
- A identificação de cada registro deve ser página:número do registro na página. Por exemplo, a identificação 3:2, indica o segundo registro da terceira página.
- Nas páginas de índice (árvores-B) despreze as informações de endereço dos filhos. No exemplo, elas são irrelevantes. Faça como no exemplo.
- Os dados de entrada estão ordenados pela coluna do índice granulado.

Exemplo: Seja o conjunto de dados, no qual se fará o granulado por cor:

NOME	COR	ANIMAL	COMIDA	id.
josé	dourado	chopim	feijão	34
gabi	laranja	perú	aipim	22
sueli	lilas	gato	alho	40
antônio	musgo	vaca	batata	41
tatiana	petróleo	cavalo	arroz	37
juca	preto	águia	goiaba	36
romeu	roxo	dragão	maracujá	27
paulo	verde	urubú	rabanete	20

Com este conjunto de dados, ter-se-á a seguintes disposição nos arquivos

Páginas de Dados					
Pág1	josé	dourado	chopim	feijão	34
	gabi	laranja	perú	aipim	22
	sueli	lilas	gato	alho	40
Pág2	antônio	musgo	vaca	batata	41
	tatiana	petróleo	cavalo	arroz	37
	juca	preto	águia	goiaba	36
Pág3	romeu	roxo	dragão	maracujá	27
	paulo	verde	urubú	rabanete	20

A seguir, as páginas de índices

Páginas de índices	
Índice I1	granulado por COR
Página 1 ←raiz	dourado(1:1), musgo(2:1), roxo(3:1)
Índice I2	por NOME
Página 1	antônio(2:1), gabi(1:2)
Página 2 ←raiz	jose(1:1)
Página 3	juca(2:3), paulo(3:2), romeu(3:1), sueli(1:3), tatiana(2:2)
Índice I3	por BICHO
P1	águia(2:3), cavalo(2:2), chopim(1:1), dragão(3:1)
P2 ←raiz	gato(1:3)
P3	perú(1:2),urubú(3:2), vaca(2:1)
Índice I4	por COMIDA
P1	aipim(1:2),alho(1:3)
P2 ←raiz	arroz(2:2)
P3	batata(2:1), feijão(1:1), goiaba(2:3), maracujá(3:1), rabanete(3:2)
Índice I5	por IDADE
P1	20(3:2), 22(1:2), 27(3:1), 34(1:1), 36(2:3)
P2 ←raiz	37(2:2)
P3	40(1:3), 41(2:1)

☞ Para você fazer

Desenhe a árvore B a seguir:

```
    1 2    3   4   5   6   7   8   9 10 11 12 13 14 15 16 17
  +-----------------------------------------------------------
 1| 1 5   32  33  35  37  38   0   0
 2| 0 6   40  51  69  78  85  93   0  1  8  7 18  4 20 13
 3| 1 7  126 128 129 130 131 132 133
 4| 1 3   80  81  82   0   0   0   0
 5| 1 6  168 169 174 175 176 177   0
 6| 1 5  111 112 113 114 115   0   0
 7| 1 7   53  54  55  56  57  60  68
 8| 1 7   41  44  46  47  48  49  50
 9| 1 6  186 187 188 189 190 192   0
*10| 0 2  103 165   0   0   0   0   0  2 11 17
 11| 0 4  117 124 136 146   0   0   0  6 14  3 12 19
 12| 1 4  137 139 141 144   0   0   0
 13| 1 5   94  96  98 101 102   0   0
 14| 1 5  118 119 120 122 123   0   0
 15| 1 5  194 195 196 197 198   0   0
 16| 1 5  179 180 182 183 184   0   0
 17| 0 3  178 185 193   0   0   0   0  5 16  9 15
 18| 1 6   70  72  73  74  75  77   0
 19| 1 5  150 153 154 157 161   0   0
 20| 1 4   86  87  88  89   0   0   0
```

Responda agora:

Qual a altura da árvore B ?	Se o 90 for incluido, qual linha,coluna na matriz?	Se o 42 for incluído, qual linha,coluna na matriz?

Mais uma Instância do mesmo exercício Desenhe a árvore B a seguir:

```
    1 2    3   4   5   6   7   8   9 10 11 12 13 14 15 16 17
  +-----------------------------------------------------------
 1| 1 4   32  34  35  36   0   0   0
 2| 0 4   38  50  56  67   0   0   0  1 14 13  8 19
 3| 1 4  100 101 106 107   0   0   0
 4| 1 4  151 152 153 154   0   0   0
 5| 1 5   83  84  85  87  88   0   0
 6| 1 5  124 125 126 127 131   0   0
 7| 1 6  159 160 161 162 163 165   0
 8| 1 5   57  58  60  61  63   0   0
 9| 1 7  183 187 188 189 190 192 194
*10| 0 3   81 123 158   0   0   0   0  2 18 11 22
 11| 0 3  134 142 149   0   0   0   0  6 15 21  4
 12| 1 4   94  95  96  98   0   0   0
 13| 1 5   51  52  53  54  55   0   0
 14| 1 5   43  44  45  46  47   0   0
 15| 1 4  135 136 137 138   0   0   0
 16| 1 3  196 197 198   0   0   0   0
 17| 1 7  112 113 114 117 118 119 121
 18| 0 3   92  99 109   0   0   0   0  5 12  3 17
 19| 1 7   68  69  70  71  73  77  79
 20| 1 5  168 171 174 175 178   0   0
 21| 1 4  144 145 147 148   0   0   0
 22| 0 3  167 181 195   0   0   0   0  7 20  9 16
```

Responda agora:

Qual a altura da árvore B ?	Se o 128 for incluido, qual linha,coluna na matriz?	Se o 49 for incluído, qual linha,coluna na matriz?

CEP-UFPR-UTFPR-PUC/Pr-UP Sistemas de Informação 12/08/2024 - 11:35:52.4
Matemática aplicada Prof Dr P Kantek (pkantek@gmail.com)
Evolução de árvores: Árvore-B (e B+) VIVXq92a V: 1.02
76663 DANILO DEPETRIS SOARES
24CC2401 - 2 primeiro exercicio do T3 _____ / / _____

Árvore B

O conceito de árvore B foi desenvolvido por Rudolf Bayer e Edward McCreight em 1972.

Eles publicaram seu trabalho no artigo *"Organization and Maintenance of Large Ordered Indices"*, que descreve as propriedades e operações básicas das árvores B.

O trabalho de Bayer e McCreight teve um impacto significativo na área de ciência da computação, e as árvores B são agora amplamente utilizadas em bancos de dados e outros sistemas de software.

As árvores B são um tipo de árvore de busca balanceada que pode armazenar um grande número de chaves e valores de maneira eficiente. Elas são mais eficientes do que as árvores de busca binárias tradicionais, pois podem ter mais filhos por nó. As árvores B são usadas em uma variedade de aplicações, incluindo bancos de dados, sistemas de arquivos e sistemas de roteamento.

Eles não explicaram o significado do nome, só nos resta especular que esse B seja de Balanceada. Se você se lembrar das árvores binárias de pesquisa, deve recordar o problema que existe quando as chaves vem (quasi) ordenadas na entrada: a árvore gerada é desbalanceada e pode chegar a ter um desempenho terrível.

Outra qualidade da árvore-B é que ela tem grau k (k filhos por nodo). Em geral este valor é escolhido de modo a otimizar a blocagem de dados no arquivo de memória secundária. De qualquer modo este k modifica o desempenho, que lembrando é proporcional a $\log_k n$, onde n é a quantidade de chaves.

Como toda árvore, ela tem uma raiz que tem de 0 a $2k-1$ chaves e consequentemente tem de 0 a $2k$ filhos. Cada um dos filhos, tem de $k-1$ até $2k-1$ chaves de 0 até $2k$ filhos. Note que a raiz é o único nodo que sofre processamento especial, necessário para quando o arquivo tem uma única chave (por exemplo).

Um caso prático real Suponha um colégio com milhares de alunos, que têm seus dados guardados em um banco de dados. Por hipótese, os dados de cada aluno vão sendo recolhidos quando eles fazem a matrícula, pelo que a ordem física dos dados no arquivo é a data da matrícula. Parece óbvio assinalar que um determinado conjunto de dados (um arquivo) só pode ter uma única ordem física em um determinado instante. Mas nada impede que existam muitas ordens lógicas diferentes para o mesmo arquivo. Tais ordens lógicas são providenciadas por estruturas muito parecidas a uma árvore-b. Neste exemplo, poderíamos ter índices por nome (ordem alfabética), curso, turma, endereço etc.

Vai-se descrever aqui uma árvore-b que dará acesso aos dados por código do aluno.

Suponha uma árvore-b de nome M que providenciará este acesso. Vamos descrevê-la com $k=4$, que é muito pouco para aplicações reais, mas fica razoável para trabalhar com este exemplo. De acordo com a definição, os nodos de M têm entre 3 e 7 códigos de aluno. Para esta dimensão, o nodo de M terá 17 colunas:

coluna 0 um asterisco nesta posição indica que este é o nodo raiz da árvore. Conta como coluna 0 porque na verdade esta informação não está fisicamente aqui, está em outro lugar.

coluna 1 indicativo de folha (1=sim; 0=não)

coluna 2 quantidade de chaves válidas neste nodo (nem sempre o valor inválido é zero)

cols 3 a 9 chaves, sempre em ordem crescente

cols 10 a 17 endereços dos filhos associados a cada chave. Só estão preenchidas para nodos não folha.

Veja um exemplo:

```
    1 2    3    4    5    6    7    8    9 10 11 12 13 14 15 16 17
  +-------------------------------------------------------------------
 1| 1 5    3    6    7    8   10    0    0
 2| 0 4   11   20   32   44    0    0    0  1 20 15  7 12
 3| 1 4  128  131  132  134    0    0    0
 4| 1 5  149  150  151  153  160    0    0
 5| 1 6   65   73   76   79   80   83    0
 6| 1 5  182  183  188  189  191    0    0
 7| 1 3   34   35   37    0    0    0    0
 8| 1 4  106  109  111  114    0    0    0
 9| 1 6  136  137  138  140  143  145    0
*10| 0 2   64  127    0    0    0    0    0  2 17 11
 11| 0 6  135  146  162  171  181  192    0  3  9  4 14 21  6 18
 12| 1 7   46   49   50   55   56   57   63
 13| 1 4   88   89   91   94    0    0    0
 14| 1 5  164  166  167  168  169    0    0
 15| 1 6   21   22   23   28   29   30    0
 16| 1 7  117  118  120  122  123  125  126
 17| 0 4   84   95  105  116    0    0    0  5 13 19  8 16
 18| 1 5  193  195  197  199  200    0    0
 19| 1 5   96   98  100  102  103    0    0
 20| 1 4   14   15   17   19    0    0    0
 21| 1 3  174  175  177    0    0    0    0
```

Antes de continuar no exercício, você deve tentar entender o funcionamento da árvore-b.

- Desenhe a árvore
- identifique a necessidade da coluna 1. Idem para a coluna 2
- Qual a altura da árvore ?
- Quando e como a altura da árvore cresce ?

Agoram suponha que o elemento 187 é incluído. Qual a linha;coluna da matriz acima que é modificada, ou seja onde o elemento recém incluido entra ? Localize-o e veja:

```
    1 2    3    4    5    6    7    8    9 10 11 12 13 14 15 16 17
  +-------------------------------------------------------------------
 1| 1 5    3    6    7    8   10    0    0
 2| 0 4   11   20   32   44    0    0    0  1 20 15  7 12
 3| 1 4  128  131  132  134    0    0    0
 4| 1 5  149  150  151  153  160    0    0
 5| 1 6   65   73   76   79   80   83    0
 6| 1 6  182  183(187)188  189  191    0
 7| 1 3   34   35   37    0    0    0    0
 8| 1 4  106  109  111  114    0    0    0
 9| 1 6  136  137  138  140  143  145    0
*10| 0 2   64  127    0    0    0    0    0  2 17 11
 11| 0 6  135  146  162  171  181  192    0  3  9  4 14 21  6 18
 12| 1 7   46   49   50   55   56   57   63
 13| 1 4   88   89   91   94    0    0    0
 14| 1 5  164  166  167  168  169    0    0
 15| 1 6   21   22   23   28   29   30    0
 16| 1 7  117  118  120  122  123  125  126
 17| 0 4   84   95  105  116    0    0    0  5 13 19  8 16
 18| 1 5  193  195  197  199  200    0    0
 19| 1 5   96   98  100  102  103    0    0
 20| 1 4   14   15   17   19    0    0    0
 21| 1 3  174  175  177    0    0    0    0
```

Como voce pode ver ele está na linha ____ e coluna ____. Algumas características notáveis da árvore-b

- Figurinha carimbada no mundo do software: usam B-trees: SYBASE e do Harbour (antigo Clipper). Também são usados no Oracle, MySQL, PostgreSQL, Microsoft SQL e DB2, da IBM.
- muitas estruturas usam a árvore-b+. O + significa que há uma ligação adicional entre as folhas, para permitir o processamento sequencial dos dados.
- Existem nodos folha e não-folha. Os endereços dos filhos (colunas 10 a 17) só existem para os nodos que os têm (nodos não folhas).
- Os códigos de aluno (colunas 3 a 9 não mostram, para diminuir o ruído), mas ao lado de cada código está o endereço físico real dos dados deste aluno, que afinal é o dado que nos interessa nesta busca.
- Os códigos (colunas 3 a 9) estão em ordem crescente. Isto é fundamental e não pode ser modificado.
- Perceba como, em média, apenas a metade dos códigos está preenchida. Isto é necessário para permitir inclusão de novos dados. Sinaliza também um certo desperdício (overhead) de área em disco que qualquer método de acesso consome.
- Quando um nodo tem todas as chaves preenchidas e chega uma nova, que deveria (deverá) entrar nele, o nodo sofre um processo chamado split e ele é transformado em 2 nodos (cada um com metade dos espaços vazios). Note que o número máximo de chaves em qualquer nodo é sempre ímpar ($2n-1$) o que implica em que o elemento do meio sobe para o *pai* do nodo que está sendo quebrado. A metade anterior vai para um nodo *splitado* e a metade posterior vai para o outro nodo *splitado*.
- Quando as chaves vão sendo excluídas, e um nodo passa a ter menos do que a metade das chaves (o que é proibido) ele é juntado com um nodo irmão e ambos passam a ser um único nodo. Este processo recebe o nome de merge. Não é tão simples assim, há diversos casos particulares, mas a ideia é essa.

Mais um exemplo. Suponha a inclusão do número 26. Em que linha;coluna ele entra ?

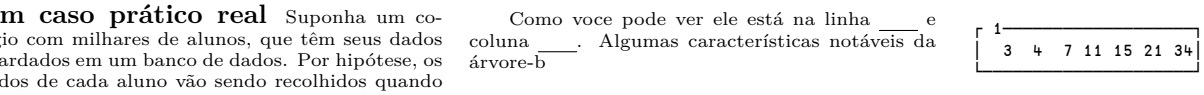
Algoritmo de Inclusão em árvore B

```
1: função INCLUA-BTREE (K:inteiro)
2: X ← raiz da árvore B
3: enquanto nodo X não é folha
4:     X ← filho correto onde K deve ser incluído
5: fim{enquanto}
6: se nodo X está cheio
7:     faça split do nodo X
8:     INCLUA-BTREE (K)
9:     abandone
10: senão
11:     inclua K no nodo X mantendo as chaves ordenadas
12: fim{se}
```

Algoritmos Estão disseminados pela comunidade. Deixo de apresentá-los aqui, pois são bastante específicos, complexos até, e mais importante do que implementá-los é entender-lhes o funcionamento. Até porque se você tiver que usar uma árvore b que não seja para aprender como ela funciona, recomendo fortemente que você use pacotes prontos, já testados e confiáveis.

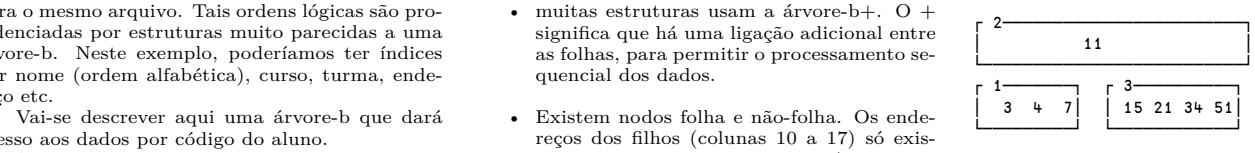
Um exemplo, passo a passo Seja uma árvore-b com $k=4$. Pela definição os nodos terão entre 3 e 7 chaves.

```
criabtree
insira 4, 7, 21, 11, 34, 15, 3
```

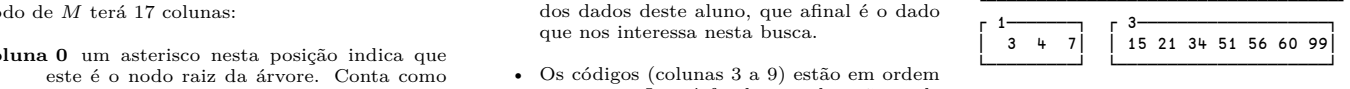


Agora, vamos inserir uma nova chave, o que vai provocar o split

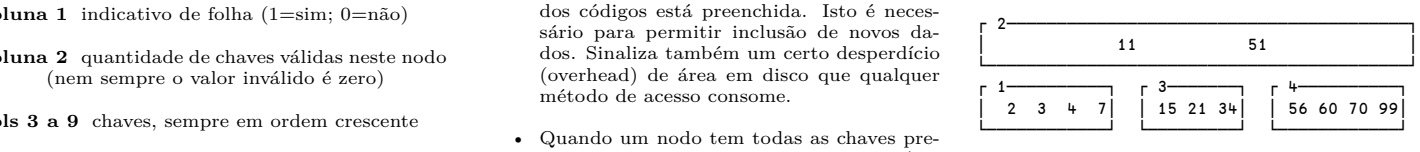
```
insira 51
```



Inserindo 60, 99, 56, fica



Inserindo 2 e 70, o 2 vai na boa para o nodo 1 enquanto o 70 força o split do nodo 3 e fica



Note que o processo funciona muito bem, mesmo que as chaves de entrada estejam desordenadas (ou em ordem aleatória). Isso é garantido pelos espaços em branco que são reservados nos nodos.

Um caso quasi-real: SYBASE

O SYBASE usa árvores B para indexar as tabelas. Existem 2 tipos de árvores B: as granuladas e as não granuladas (clustered). As granuladas são árvores esparsas (nem todo registro de dados tem entrada nos índices) nas quais os registros (na tabela original) são mantidos em ordem dos valores de índice e apenas o primeiro registro em cada página de dados tem uma entrada de índice. Árvores não granuladas são densas e cada um dos registros da tabela tem 1 entrada de índice.

Uma tabela SYBASE com índice granulado exige que seus elementos sejam mantidos em ordem dentro da tabela. Assim, inclusões podem se dar em qualquer página, a depender do conteúdo do campo de índice. Tabelas SYBASE que tem apenas índices não granulados fazem as inclusões de novos elementos sempre ao final das tabelas (na última página).

Tabelas SYBASE podem ter apenas 1 índice granulado, mas podem ter até 250 índices não granulados.

Na criação de um índice granulado, os dados da tabela devem estar colocados nas páginas em ordem.

Cada entrada de índice em sybase pode ser composta por até 16 campos e ter até 256 bytes. Estes campos não precisam estar contiguos. Só não podem ser partes de campos e resultados de cálculos.

O tamanho da página varia, mas o tamanho mais comum é 2K. Existem 5 tipos de páginas em sybase, a saber:

Páginas de Dados contém registros de dados ou registros de log. Ambos são estruturalmente diferentes, mas são construídos do mesmo jeito.

Páginas de índices contém registros de índice

Páginas de texto ou imagem contém BLOBS

Páginas de alocação contém estruturas de dados usadas para gerenciar o processo de alocacão de páginas

páginas de Estatísticas contém estatística de distribuição e uso para os índices.

Página de Dados Se os registros tem tamanho fixo, não haverá tabela de deslocamento na página e o endereço de cada registro é obtido mediante simples cálculo. Neste caso, o tamanho do registro (fixo) é guardado no campo TAMANHO MÍNIMO DE REGISTRO do header.

O header da página é um conjunto de 32 bytes contendo (entre outros): número lógico da página, anterior e próxima página lógica, identificação do objeto ao qual esta página pertence, primeira entrada de registro disponível nesta página, deslocamento do espaço livre, tamanho mínimo do registro

Depois vem a área de registros, que contém um número inteiro de slots para guardar registros. Um registro não atravessa limites de páginas

Tabela de deslocamento: Um conjunto de deslocamentos desde o início da página dando o início de cada um dos registros. Este conjunto cresce do fim da página para o início e só existe quando o tamanho do registro é variável.

Cada registro na página recebe um número de registro. É um campo de 1 byte e assim restringe-se o número de registros em uma página a 256. O número do registro usado nos índices não granulados (densos) é uma combinação do número da página com o número do registro na página.

O registro pode ocupar a página inteira (2048-32=2016). Não há tamanho mínimo para o registro, mas só pode haver 256 registros na página.

Os índices SYBASE são organizados na forma de arvores B + (o + significa que há uma ligação adicional entre as folhas, para permitir o processamento sequencial dos dados).

No exemplo aqui estudado, vão-se fazer algumas considerações/simplificações descritas a seguir.

Suponha um arquivo de pessoas contendo nome, cor preferida, bicho de estimação, alimento que gosta e idade.

Construindo um arquivo SYBASE contendo estes dados e que esteja sujeito às seguintes características:

- Registros de tamanho fixo igual a 40 bytes (9+9+9+9+4).
- Um índice granulado ou esparso (clustered) por um dos campos. No exemplo que é igual para todos, o índice esparso é por cor.
- Quatro índices não granulados (densos) pelos demais atributos.
- Páginas de dados com espaço para 4 registros e fator de preenchimento de 4:1 (usar apenas 3 espaços na carga original).
- Páginas de índices com espaço para 5 chaves (nome, cor, bicho, alimento ou idade). Lembrar que este número é impar, o que indica que no *split* vai subir a terceira chave (das 5 que existem).
- A identificação de cada registro deve ser página:número do registro na página. Por exemplo, a identificação 3:2, indica o segundo registro da terceira página.
- Nas páginas de índice (árvores-B) despreze as informações de endereço dos filhos. No exemplo, elas são irrelevantes. Faça como no exemplo.
- Os dados de entrada estão ordenados pela coluna do índice granulado.

Exemplo: Seja o conjunto de dados, no qual se fará o granulado por cor:

NOME	COR	ANIMAL	COMIDA	id.
josé	dourado	chopim	feijão	34
gabi	laranja	perú	aipim	22
sueli	lilas	gato	alho	40
antônio	musgo	vaca	batata	41
tatiana	petróleo	cavalo	arroz	37
juca	preto	águia	goiaba	36
romeu	roxo	dragão	maracujá	27
paulo	verde	urubú	rabanete	20

Com este conjunto de dados, ter-se-á a seguintes disposição nos arquivos

Páginas de Dados					
Pág1	josé	dourado	chopim	feijão	34
	gabi	laranja	perú	aipim	22
	sueli	lilas	gato	alho	40
Pág2	antônio	musgo	vaca	batata	41
	tatiana	petróleo	cavalo	arroz	37
	juca	preto	águia	goiaba	36
Pág3	romeu	roxo	dragão	maracujá	27
	paulo	verde	urubú	rabanete	20

A seguir, as páginas de índices

Páginas de índices	
Índice I1	granulado por COR
Página 1 ←raiz	dourado(1:1), musgo(2:1), roxo(3:1)
Índice I2	por NOME
Página 1	antônio(2:1), gabi(1:2)
Página 2 ←raiz	jose(1:1)
Página 3	juca(2:3), paulo(3:2), romeu(3:1), sueli(1:3), tatiana(2:2)
Índice I3	por BICHO
P1	águia(2:3), cavalo(2:2), chopim(1:1), dragão(3:1)
P2 ←raiz	gato(1:3)
P3	perú(1:2),urubú(3:2), vaca(2:1)
Índice I4	por COMIDA
P1	aipim(1:2),alho(1:3)
P2 ←raiz	arroz(2:2)
P3	batata(2:1), feijão(1:1), goiaba(2:3), maracujá(3:1), rabanete(3:2)
Índice I5	por IDADE
P1	20(3:2), 22(1:2), 27(3:1), 34(1:1), 36(2:3)
P2 ←raiz	37(2:2)
P3	40(1:3), 41(2:1)

☞ Para você fazer

Desenhe a árvore B a seguir:

```
     1  2    3    4    5    6    7    8    9 10 11 12 13 14 15 16 17
   +-------------------------------------------------------------------
 1| 1  6   32   33   38   39   40   43    0
 2| 0  3   44   56   66    0    0    0    0  1 15  7 21
 3| 1  4  150  151  152  153    0    0    0
 4| 1  6   97  100  101  102  103  105    0
 5| 1  6  179  180  183  184  185  186    0
 6| 1  4  127  130  131  132    0    0    0
 7| 1  4   58   61   63   65    0    0    0
 8| 1  4   79   81   82   83    0    0    0
 9| 1  4  167  168  169  170    0    0    0
*10| 0  3   77  126  166    0    0    0    0  2 22 11 18
 11| 0  4  134  141  149  155    0    0    0  6 19 13  3 16
 12| 1  7  191  195  196  197  198  199  200
 13| 1  7  142  143  144  145  146  147  148
 14| 1  7  108  109  114  115  116  117  125
 15| 1  7   45   46   47   48   51   52   53
 16| 1  4  157  158  161  163    0    0    0
 17| 1  4  172  173  174  175    0    0    0
 18| 0  3  171  176  189    0    0    0    0  9 17  5 12
 19| 1  3  135  137  138    0    0    0    0
 20| 1  4   86   89   91   94    0    0    0
 21| 1  3   67   74   75    0    0    0    0
 22| 0  3   85   96  106    0    0    0    0  8 20  4 14
```

Responda agora:

Qual a altura da árvore B ?	Se o 76 for incluido, qual linha,coluna na matriz?	Se o 119 for incluído, qual linha,coluna na matriz?

Mais uma Instância do mesmo exercício Desenhe a árvore B a seguir:

```
     1  2    3    4    5    6    7    8    9 10 11 12 13 14 15 16 17
   +-------------------------------------------------------------------
 1| 1  5   31   32   35   37   38    0    0
 2| 0  3   39   50   60    0    0    0    0  1 18  9  6
 3| 1  6  138  142  145  146  148  150    0
 4| 1  3   87   89   90    0    0    0    0
 5| 1  4  178  179  181  183    0    0    0
 6| 1  6   61   62   63   64   65   66    0
 7| 1  4  162  163  164  165    0    0    0
 8| 1  3  119  121  123    0    0    0    0
 9| 1  7   51   53   54   55   56   57   59
*10| 0  3   71  118  161    0    0    0    0  2 21 11 19
 11| 0  3  125  136  152    0    0    0    0  8 15  3 14
 12| 1  5  191  195  196  198  200    0    0
 13| 1  4   73   74   77   78    0    0    0
 14| 1  5  153  155  156  159  160    0    0
 15| 1  5  127  128  129  133  135    0    0
 16| 1  7  108  109  110  113  114  115  116
 17| 1  5  167  168  170  171  172    0    0
 18| 1  3   41   43   44    0    0    0    0
 19| 0  4  166  173  184  190    0    0    0  7 17  5 23 12
 20| 1  5   92   94   96   97  103    0    0
 21| 0  4   82   86   91  106    0    0    0 13 22  4 20 16
 22| 1  3   83   84   85    0    0    0    0
 23| 1  3  186  187  189    0    0    0    0
```

Responda agora:

Qual a altura da árvore B ?	Se o 182 for incluido, qual linha,coluna na matriz?	Se o 141 for incluído, qual linha,coluna na matriz?

CEP-UFPR-UTFPR-PUC/Pr-UP Sistemas de Informação 12/08/2024 - 11:35:52.4
Matemática aplicada Prof Dr P Kantek (pkantek@gmail.com)
Evolução de árvores: Árvore-B (e B+) VIVXq92a V: 1.02
76513 EDUARDO CAVALHEIRO DIAS
24CC2401 - 3 primeiro exercicio do T3 _____ / / _____

Árvore B

O conceito de árvore B foi desenvolvido por Rudolf Bayer e Edward McCreight em 1972.

Eles publicaram seu trabalho no artigo *"Organization and Maintenance of Large Ordered Indices"*, que descreve as propriedades e operações básicas das árvores B.

O trabalho de Bayer e McCreight teve um impacto significativo na área de ciência da computação, e as árvores B são agora amplamente utilizadas em bancos de dados e outros sistemas de software.

As árvores B são um tipo de árvore de busca balanceada que pode armazenar um grande número de chaves e valores de maneira eficiente. Elas são mais eficientes do que as árvores de busca binárias tradicionais, pois podem ter mais filhos por nó. As árvores B são usadas em uma variedade de aplicações, incluindo bancos de dados, sistemas de arquivos e sistemas de roteamento.

Eles não explicaram o significado do nome, só nos resta especular que esse B seja de Balanceada. Se você se lembrar das árvores binárias de pesquisa, deve recordar o problema que existe quando as chaves vem (quasi) ordenadas na entrada: a árvore gerada é desbalanceada e pode chegar a ter um desempenho terrível.

Outra qualidade da árvore-B é que ela tem grau k (k filhos por nodo). Em geral este valor é escolhido de modo a otimizar a blocagem de dados no arquivo de memória secundária. De qualquer modo este k modifica o desempenho, que lembrando é proporcional a $\log_k n$, onde n é a quantidade de chaves.

Como toda árvore, ela tem uma raiz que tem de 0 a $2k-1$ chaves e consequentemente tem de 0 a $2k$ filhos. Cada um dos filhos, tem de $k-1$ até $2k-1$ chaves de 0 até $2k$ filhos. Note que a raiz é o único nodo que sofre processamento especial, necessário para quando o arquivo tem uma única chave (por exemplo).

Um caso prático real Suponha um colégio com milhares de alunos, que têm seus dados guardados em um banco de dados. Por hipótese, os dados de cada aluno vão sendo recolhidos quando eles fazem a matrícula, pelo que a ordem física dos dados no arquivo é a data da matrícula. Parece óbvio assinalar que um determinado conjunto de dados (um arquivo) só pode ter uma única ordem física em um determinado instante. Mas nada impede que existam muitas ordens lógicas diferentes para o mesmo arquivo. Tais ordens lógicas são providenciadas por estruturas muito parecidas a uma árvore-b. Neste exemplo, poderíamos ter índices por nome (ordem alfabética), curso, turma, endereço etc.

Vai-se descrever aqui uma árvore-b que dará acesso aos dados por código do aluno.

Suponha uma árvore-b de nome M que providenciará este acesso. Vamos descrevê-la com $k=4$, que é muito pouco para aplicações reais, mas fica razoável para trabalhar com este exemplo. De acordo com a definição, os nodos de M têm entre 3 e 7 códigos de aluno. Para esta dimensão, o nodo de M terá 17 colunas:

coluna 0 um asterisco nesta posição indica que este é o nodo raiz da árvore. Conta como coluna 0 porque na verdade esta informação não está fisicamente aqui, está em outro lugar.

coluna 1 indicativo de folha (1=sim; 0=não)

coluna 2 quantidade de chaves válidas neste nodo (nem sempre o valor inválido é zero)

cols 3 a 9 chaves, sempre em ordem crescente

cols 10 a 17 endereços dos filhos associados a cada chave. Só estão preenchidas para nodos não folha.

Veja um exemplo:

```
    1 2    3    4    5    6    7    8    9 10 11 12 13 14 15 16 17
  +-------------------------------------------------------------------
 1| 1 5    3    6    7    8   10    0    0
 2| 0 4   11   20   32   44    0    0    0  1 20 15  7 12
 3| 1 4  128  131  132  134    0    0    0
 4| 1 5  149  150  151  153  160    0    0
 5| 1 6   65   73   76   79   80   83    0
 6| 1 5  182  183  188  189  191    0    0
 7| 1 3   34   35   37    0    0    0    0
 8| 1 4  106  109  111  114    0    0    0
 9| 1 6  136  137  138  140  143  145    0
*10| 0 2   64  127    0    0    0    0    0  2 17 11
 11| 0 6  135  146  162  171  181  192    0  3  9  4 14 21  6 18
 12| 1 7   46   49   50   55   56   57   63
 13| 1 4   88   89   91   94    0    0    0
 14| 1 5  164  166  167  168  169    0    0
 15| 1 6   21   22   23   28   29   30    0
 16| 1 7  117  118  120  122  123  125  126
 17| 0 4   84   95  105  116    0    0    0  5 13 19  8 16
 18| 1 5  193  195  197  199  200    0    0
 19| 1 5   96   98  100  102  103    0    0
 20| 1 4   14   15   17   19    0    0    0
 21| 1 3  174  175  177    0    0    0    0
```

Antes de continuar no exercício, você deve tentar entender o funcionamento da árvore-b.

- Desenhe a árvore
- identifique a necessidade da coluna 1. Idem para a coluna 2
- Qual a altura da árvore ?
- Quando e como a altura da árvore cresce ?

Agoram suponha que o elemento 187 é incluído. Qual a linha;coluna da matriz acima que é modificada, ou seja onde o elemento recém incluido entra ? Localize-o e veja:

```
    1 2    3    4    5    6    7    8    9 10 11 12 13 14 15 16 17
  +-------------------------------------------------------------------
 1| 1 5    3    6    7    8   10    0    0
 2| 0 4   11   20   32   44    0    0    0  1 20 15  7 12
 3| 1 4  128  131  132  134    0    0    0
 4| 1 5  149  150  151  153  160    0    0
 5| 1 6   65   73   76   79   80   83    0
 6| 1 6  182  183(187)188  189  191    0
 7| 1 3   34   35   37    0    0    0    0
 8| 1 4  106  109  111  114    0    0    0
 9| 1 6  136  137  138  140  143  145    0
*10| 0 2   64  127    0    0    0    0    0  2 17 11
 11| 0 6  135  146  162  171  181  192    0  3  9  4 14 21  6 18
 12| 1 7   46   49   50   55   56   57   63
 13| 1 4   88   89   91   94    0    0    0
 14| 1 5  164  166  167  168  169    0    0
 15| 1 6   21   22   23   28   29   30    0
 16| 1 7  117  118  120  122  123  125  126
 17| 0 4   84   95  105  116    0    0    0  5 13 19  8 16
 18| 1 5  193  195  197  199  200    0    0
 19| 1 5   96   98  100  102  103    0    0
 20| 1 4   14   15   17   19    0    0    0
 21| 1 3  174  175  177    0    0    0    0
```

Como voce pode ver ele está na linha ____ e coluna ____. Algumas características notáveis da árvore-b

- Figurinha carimbada no mundo do software: usam B-trees: SYBASE e do Harbour (antigo Clipper). Também são usados no Oracle, MySQL, PostgreSQL, Microsoft SQL e DB2, da IBM.
- muitas estruturas usam a árvore-b+. O + significa que há uma ligação adicional entre as folhas, para permitir o processamento sequencial dos dados.
- Existem nodos folha e não-folha. Os endereços dos filhos (colunas 10 a 17) só existem para os nodos que os têm (nodos não folhas).
- Os códigos de aluno (colunas 3 a 9 não mostram, para diminuir o ruído), mas ao lado de cada código está o endereço físico real dos dados deste aluno, que afinal é o dado que nos interessa nesta busca.
- Os códigos (colunas 3 a 9) estão em ordem crescente. Isto é fundamental e não pode ser modificado.
- Perceba como, em média, apenas a metade dos códigos está preenchida. Isto é necessário para permitir inclusão de novos dados. Sinaliza também um certo desperdício (overhead) de área em disco que qualquer método de acesso consome.
- Quando um nodo tem todas as chaves preenchidas e chega uma nova, que deveria (deverá) entrar nele, o nodo sofre um processo chamado split e ele é transformado em 2 nodos (cada um com metade dos espaços vazios). Note que o número máximo de chaves em qualquer nodo é sempre ímpar ($2n-1$) o que implica em que o elemento do meio sobe para o *pai* do nodo que está sendo quebrado. A metade anterior vai para um nodo *splitado* e a metade posterior vai para o outro nodo *splitado*.
- Quando as chaves vão sendo excluídas, e um nodo passa a ter menos do que a metade das chaves (o que é proibido) ele é juntado com um nodo irmão e ambos passam a ser um único nodo. Este processo recebe o nome de merge. Não é tão simples assim, há diversos casos particulares, mas a ideia é essa.

Mais um exemplo. Suponha a inclusão do número 26. Em que linha;coluna ele entra ?

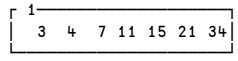
Algoritmo de Inclusão em árvore B

```
1: função INCLUA-BTREE (K:inteiro)
2: X ← raiz da árvore B
3: enquanto nodo X não é folha
4:     X ← filho correto onde K deve ser incluído
5: fim{enquanto}
6: se nodo X está cheio
7:     faça split do nodo X
8:     INCLUA-BTREE (K)
9:     abandone
10: senão
11:     inclua K no nodo X mantendo as chaves ordenadas
12: fim{se}
```

Algoritmos Estão disseminados pela comunidade. Deixo de apresentá-los aqui, pois são bastante específicos, complexos até, e mais importante do que implementá-los é entender-lhes o funcionamento. Até porque se você tiver que usar uma árvore b que não seja para aprender como ela funciona, recomendo fortemente que você use pacotes prontos, já testados e confiáveis.

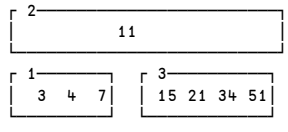
Um exemplo, passo a passo Seja uma árvore-b com $k=4$. Pela definição os nodos terão entre 3 e 7 chaves.

```
criabtree
insira 4, 7, 21, 11, 34, 15, 3
```

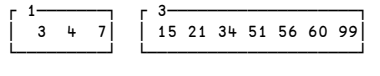


Agora, vamos inserir uma nova chave, o que vai provocar o split

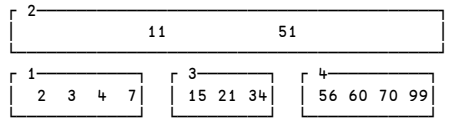
```
insira 51
```



Inserindo 60, 99, 56, fica



Inserindo 2 e 70, o 2 vai na boa para o nodo 1 enquanto o 70 força o split do nodo 3 e fica



Note que o processo funciona muito bem, mesmo que as chaves de entrada estejam desordenadas (ou em ordem aleatória). Isso é garantido pelos espaços em branco que são reservados nos nodos.

Um caso quasi-real: SYBASE

O SYBASE usa árvores B para indexar as tabelas. Existem 2 tipos de árvores B: as granuladas e as não granuladas (clustered). As granuladas são árvores esparsas (nem todo registro de dados tem entrada nos índices) nas quais os registros (na tabela original) são mantidos em ordem dos valores de índice e apenas o primeiro registro em cada página de dados tem uma entrada de índice. Árvores não granuladas são densas e cada um dos registros da tabela tem 1 entrada de índice.

Uma tabela SYBASE com índice granulado exige que seus elementos sejam mantidos em ordem dentro da tabela. Assim, inclusões podem se dar em qualquer página, a depender do conteúdo do campo de índice. Tabelas SYBASE que tem apenas índices não granulados fazem as inclusões de novos elementos sempre ao final das tabelas (na última página).

Tabelas SYBASE podem ter apenas 1 índice granulado, mas podem ter até 250 índices não granulados.

Na criação de um índice granulado, os dados da tabela devem estar colocados nas páginas em ordem.

Cada entrada de índice em sybase pode ser composta por até 16 campos e ter até 256 bytes. Estes campos não precisam estar contiguos. Só não podem ser partes de campos e resultados de cálculos.

O tamanho da página varia, mas o tamanho mais comum é 2K. Existem 5 tipos de páginas em sybase, a saber:

Páginas de Dados contém registros de dados ou registros de log. Ambos são estruturalmente diferentes, mas são construídos do mesmo jeito.

Páginas de índices contém registros de índice

Páginas de texto ou imagem contém BLOBS

Páginas de alocação contém estruturas de dados usadas para gerenciar o processo de alocacão de páginas

páginas de Estatísticas contém estatística de distribuição e uso para os índices.

Página de Dados Se os registros tem tamanho fixo, não haverá tabela de deslocamento na página e o endereço de cada registro é obtido mediante simples cálculo. Neste caso, o tamanho do registro (fixo) é guardado no campo TAMANHO MÍNIMO DE REGISTRO do header.

O header da página é um conjunto de 32 bytes contendo (entre outros): número lógico da página, anterior e próxima página lógica, identificação do objeto ao qual esta página pertence, primeira entrada de registro disponível nesta página, deslocamento do espaço livre, tamanho mínimo do registro

Depois vem a área de registros, que contém um número inteiro de slots para guardar registros. Um registro não atravessa limites de páginas

Tabela de deslocamento: Um conjunto de deslocamentos desde o início da página dando o início de cada um dos registros. Este conjunto cresce do fim da página para o início e só existe quando o tamanho do registro é variável.

Cada registro na página recebe um número de registro. É um campo de 1 byte e assim restringe-se o número de registros em uma página a 256. O número do registro usado nos índices não granulados (densos) é uma combinação do número da página com o número do registro na página.

O registro pode ocupar a página inteira (2048-32=2016). Não há tamanho mínimo para o registro, mas só pode haver 256 registros na página.

Os índices SYBASE são organizados na forma de arvores B + (o + significa que há uma ligação adicional entre as folhas, para permitir o processamento sequencial dos dados).

No exemplo aqui estudado, vão-se fazer algumas considerações/simplificações descritas a seguir.

Suponha um arquivo de pessoas contendo nome, cor preferida, bicho de estimação, alimento que gosta e idade.

Construindo um arquivo SYBASE contendo estes dados e que esteja sujeito às seguintes características:

- Registros de tamanho fixo igual a 40 bytes (9+9+9+9+4).
- Um índice granulado ou esparso (clustered) por um dos campos. No exemplo que é igual para todos, o índice esparso é por cor.
- Quatro índices não granulados (densos) pelos demais atributos.
- Páginas de dados com espaço para 4 registros e fator de preenchimento de 4:1 (usar apenas 3 espaços na carga original).
- Páginas de índices com espaço para 5 chaves (nome, cor, bicho, alimento ou idade). Lembrar que este número é impar, o que indica que no *split* vai subir a terceira chave (das 5 que existem).
- A identificação de cada registro deve ser página:número do registro na página. Por exemplo, a identificação 3:2, indica o segundo registro da terceira página.
- Nas páginas de índice (árvores-B) despreze as informações de endereço dos filhos. No exemplo, elas são irrelevantes. Faça como no exemplo.
- Os dados de entrada estão ordenados pela coluna do índice granulado.

Exemplo: Seja o conjunto de dados, no qual se fará o granulado por cor:

NOME	COR	ANIMAL	COMIDA	id.
josé	dourado	chopim	feijão	34
gabi	laranja	perú	aipim	22
sueli	lilas	gato	alho	40
antônio	musgo	vaca	batata	41
tatiana	petróleo	cavalo	arroz	37
juca	preto	águia	goiaba	36
romeu	roxo	dragão	maracujá	27
paulo	verde	urubú	rabanete	20

Com este conjunto de dados, ter-se-á a seguintes disposição nos arquivos

Páginas de Dados					
Pág1	josé	dourado	chopim	feijão	34
	gabi	laranja	perú	aipim	22
	sueli	lilas	gato	alho	40
Pág2	antônio	musgo	vaca	batata	41
	tatiana	petróleo	cavalo	arroz	37
	juca	preto	águia	goiaba	36
Pág3	romeu	roxo	dragão	maracujá	27
	paulo	verde	urubú	rabanete	20

A seguir, as páginas de índices

Páginas de índices	
Índice I1	granulado por COR
Página 1 ←raiz	dourado(1:1), musgo(2:1), roxo(3:1)
Índice I2	por NOME
Página 1	antônio(2:1), gabi(1:2)
Página 2 ←raiz	jose(1:1)
Página 3	juca(2:3), paulo(3:2), romeu(3:1), sueli(1:3), tatiana(2:2)
Índice I3	por BICHO
P1	águia(2:3), cavalo(2:2), chopim(1:1), dragão(3:1)
P2 ←raiz	gato(1:3)
P3	perú(1:2),urubú(3:2), vaca(2:1)
Índice I4	por COMIDA
P1	aipim(1:2),alho(1:3)
P2 ←raiz	arroz(2:2)
P3	batata(2:1), feijão(1:1), goiaba(2:3), maracujá(3:1), rabanete(3:2)
Índice I5	por IDADE
P1	20(3:2), 22(1:2), 27(3:1), 34(1:1), 36(2:3)
P2 ←raiz	37(2:2)
P3	40(1:3), 41(2:1)

☞ Para você fazer

Desenhe a árvore B a seguir:

```
     1  2    3    4    5    6    7    8    9 10 11 12 13 14 15 16 17
   +-------------------------------------------------------------------
 1| 1  4   31   33   36   37    0    0    0
 2| 0  3   39   49   54    0    0    0    0  1 17  8 20
 3| 1  7  158  159  161  162  163  164  165
 4| 1  7  124  125  127  129  130  132  137
 5| 1  6  167  168  170  172  173  174    0
 6| 1  5   87   91   92   93   98    0    0
 7| 1  4  140  142  143  144    0    0    0
 8| 1  4   50   51   52   53    0    0    0
 9| 1  7  114  115  117  118  119  120  122
*10| 0  3   64  123  166    0    0    0    0  2 21 11 19
11| 0  3  139  145  155    0    0    0    0  4  7 15  3
12| 1  6  102  103  104  105  106  108    0
13| 1  4   65   69   71   73    0    0    0
14| 1  4  195  196  197  200    0    0    0
15| 1  5  146  149  151  152  153    0    0
16| 1  4  180  181  186  187    0    0    0
17| 1  4   41   42   43   47    0    0    0
18| 1  3  176  177  178    0    0    0    0
19| 0  4  175  179  188  194    0    0    0  5 18 16 23 14
20| 1  3   58   61   62    0    0    0    0
21| 0  4   76   85   99  112    0    0    0 13 22  6 12  9
22| 1  3   78   80   81    0    0    0    0
23| 1  3  189  191  193    0    0    0    0
```

Responda agora:

Qual a altura da árvore B ?	Se o 86 for incluido, qual linha,coluna na matriz?	Se o 57 for incluído, qual linha,coluna na matriz?

Mais uma Instância do mesmo exercício Desenhe a árvore B a seguir:

```
     1  2    3    4    5    6    7    8    9 10 11 12 13 14 15 16 17
   +-------------------------------------------------------------------
 1| 1  7   31   33   34   35   37   38   39
 2| 0  4   41   50   58   69    0    0    0  1 12 16  4 18
 3| 1  6  156  157  160  161  164  165    0
 4| 1  3   61   62   63    0    0    0    0
 5| 1  3  108  109  111    0    0    0    0
 6| 1  5  173  174  177  181  184    0    0
 7| 1  5  134  135  136  137  138    0    0
 8| 1  5  186  187  189  191  192    0    0
 9| 1  4   81   82   84   86    0    0    0
*10| 0  2   80  133    0    0    0    0    0  2 17 11
11| 0  6  139  153  166  172  185  194    0  7 14  3 19  6  8 20
12| 1  7   42   43   44   45   47   48   49
13| 1  5  128  129  130  131  132    0    0
14| 1  6  140  142  144  146  147  149    0
15| 1  5   97  101  102  103  104    0    0
16| 1  5   52   54   55   56   57    0    0
17| 0  4   88  105  113  124    0    0    0  9 15  5 21 13
18| 1  5   70   72   74   75   76    0    0
19| 1  3  167  169  170    0    0    0    0
20| 1  4  195  196  197  199    0    0    0
21| 1  6  115  116  118  120  121  122    0
```

Responda agora:

Qual a altura da árvore B ?	Se o 91 for incluido, qual linha,coluna na matriz?	Se o 151 for incluído, qual linha,coluna na matriz?

CEP-UFPR-UTFPR-PUC/Pr-UP Sistemas de Informação 12/08/2024 - 11:35:52.4
Matemática aplicada Prof Dr P Kantek (pkantek@gmail.com)
Evolução de árvores: Árvore-B (e B+) VIVXq92a V: 1.02
76520 ERIC CORDEIRO
24CC2401 - 4 primeiro exercicio do T3 _____ / / _____

Árvore B

O conceito de árvore B foi desenvolvido por Rudolf Bayer e Edward McCreight em 1972.

Eles publicaram seu trabalho no artigo *"Organization and Maintenance of Large Ordered Indices"*, que descreve as propriedades e operações básicas das árvores B.

O trabalho de Bayer e McCreight teve um impacto significativo na área de ciência da computação, e as árvores B são agora amplamente utilizadas em bancos de dados e outros sistemas de software.

As árvores B são um tipo de árvore de busca balanceada que pode armazenar um grande número de chaves e valores de maneira eficiente. Elas são mais eficientes do que as árvores de busca binárias tradicionais, pois podem ter mais filhos por nó. As árvores B são usadas em uma variedade de aplicações, incluindo bancos de dados, sistemas de arquivos e sistemas de roteamento.

Eles não explicaram o significado do nome, só nos resta especular que esse B seja de Balanceada. Se você se lembrar das árvores binárias de pesquisa, deve recordar o problema que existe quando as chaves vem (quasi) ordenadas na entrada: a árvore gerada é desbalanceada e pode chegar a ter um desempenho terrível.

Outra qualidade da árvore-B é que ela tem grau k (k filhos por nodo). Em geral este valor é escolhido de modo a otimizar a blocagem de dados no arquivo de memória secundária. De qualquer modo este k modifica o desempenho, que lembrando é proporcional a $\log_k n$, onde n é a quantidade de chaves.

Como toda árvore, ela tem uma raiz que tem de 0 a $2k-1$ chaves e consequentemente tem de 0 a $2k$ filhos. Cada um dos filhos, tem de $k-1$ até $2k-1$ chaves de 0 até $2k$ filhos. Note que a raiz é o único nodo que sofre processamento especial, necessário para quando o arquivo tem uma única chave (por exemplo).

Um caso prático real Suponha um colégio com milhares de alunos, que têm seus dados guardados em um banco de dados. Por hipótese, os dados de cada aluno vão sendo recolhidos quando eles fazem a matrícula, pelo que a ordem física dos dados no arquivo é a data da matrícula. Parece óbvio assinalar que um determinado conjunto de dados (um arquivo) só pode ter uma única ordem física em um determinado instante. Mas nada impede que existam muitas ordens lógicas diferentes para o mesmo arquivo. Tais ordens lógicas são providenciadas por estruturas muito parecidas a uma árvore-b. Neste exemplo, poderíamos ter índices por nome (ordem alfabética), curso, turma, endereço etc.

Vai-se descrever aqui uma árvore-b que dará acesso aos dados por código do aluno.

Suponha uma árvore-b de nome M que providenciará este acesso. Vamos descrevê-la com $k=4$, que é muito pouco para aplicações reais, mas fica razoável para trabalhar com este exemplo. De acordo com a definição, os nodos de M têm entre 3 e 7 códigos de aluno. Para esta dimensão, o nodo de M terá 17 colunas:

coluna 0 um asterisco nesta posição indica que este é o nodo raiz da árvore. Conta como coluna 0 porque na verdade esta informação não está fisicamente aqui, está em outro lugar.

coluna 1 indicativo de folha (1=sim; 0=não)

coluna 2 quantidade de chaves válidas neste nodo (nem sempre o valor inválido é zero)

cols 3 a 9 chaves, sempre em ordem crescente

cols 10 a 17 endereços dos filhos associados a cada chave. Só estão preenchidas para nodos não folha.

Veja um exemplo:

```
    1 2    3    4    5    6    7    8    9 10 11 12 13 14 15 16 17
  +-------------------------------------------------------------------
 1| 1 5    3    6    7    8   10    0    0
 2| 0 4   11   20   32   44    0    0    0  1 20 15  7 12
 3| 1 4  128  131  132  134    0    0    0
 4| 1 5  149  150  151  153  160    0    0
 5| 1 6   65   73   76   79   80   83    0
 6| 1 5  182  183  188  189  191    0    0
 7| 1 3   34   35   37    0    0    0    0
 8| 1 4  106  109  111  114    0    0    0
 9| 1 6  136  137  138  140  143  145    0
*10| 0 2   64  127    0    0    0    0    0  2 17 11
 11| 0 6  135  146  162  171  181  192    0  3  9  4 14 21  6 18
 12| 1 7   46   49   50   55   56   57   63
 13| 1 4   88   89   91   94    0    0    0
 14| 1 5  164  166  167  168  169    0    0
 15| 1 6   21   22   23   28   29   30    0
 16| 1 7  117  118  120  122  123  125  126
 17| 0 4   84   95  105  116    0    0    0  5 13 19  8 16
 18| 1 5  193  195  197  199  200    0    0
 19| 1 5   96   98  100  102  103    0    0
 20| 1 4   14   15   17   19    0    0    0
 21| 1 3  174  175  177    0    0    0    0
```

Antes de continuar no exercício, você deve tentar entender o funcionamento da árvore-b.

- Desenhe a árvore
- identifique a necessidade da coluna 1. Idem para a coluna 2
- Qual a altura da árvore ?
- Quando e como a altura da árvore cresce ?

Agoram suponha que o elemento 187 é incluído. Qual a linha;coluna da matriz acima que é modificada, ou seja onde o elemento recém incluido entra ? Localize-o e veja:

```
    1 2    3    4    5    6    7    8    9 10 11 12 13 14 15 16 17
  +-------------------------------------------------------------------
 1| 1 5    3    6    7    8   10    0    0
 2| 0 4   11   20   32   44    0    0    0  1 20 15  7 12
 3| 1 4  128  131  132  134    0    0    0
 4| 1 5  149  150  151  153  160    0    0
 5| 1 6   65   73   76   79   80   83    0
 6| 1 6  182  183(187)188  189  191    0
 7| 1 3   34   35   37    0    0    0    0
 8| 1 4  106  109  111  114    0    0    0
 9| 1 6  136  137  138  140  143  145    0
*10| 0 2   64  127    0    0    0    0    0  2 17 11
 11| 0 6  135  146  162  171  181  192    0  3  9  4 14 21  6 18
 12| 1 7   46   49   50   55   56   57   63
 13| 1 4   88   89   91   94    0    0    0
 14| 1 5  164  166  167  168  169    0    0
 15| 1 6   21   22   23   28   29   30    0
 16| 1 7  117  118  120  122  123  125  126
 17| 0 4   84   95  105  116    0    0    0  5 13 19  8 16
 18| 1 5  193  195  197  199  200    0    0
 19| 1 5   96   98  100  102  103    0    0
 20| 1 4   14   15   17   19    0    0    0
 21| 1 3  174  175  177    0    0    0    0
```

Como voce pode ver ele está na linha ____ e coluna ____. Algumas características notáveis da árvore-b

- Figurinha carimbada no mundo do software: usam B-trees: SYBASE e do Harbour (antigo Clipper). Também são usados no Oracle, MySQL, PostgreSQL, Microsoft SQL e DB2, da IBM.
- muitas estruturas usam a árvore-b+. O + significa que há uma ligação adicional entre as folhas, para permitir o processamento sequencial dos dados.
- Existem nodos folha e não-folha. Os endereços dos filhos (colunas 10 a 17) só existem para os nodos que os têm (nodos não folhas).
- Os códigos de aluno (colunas 3 a 9 não mostram, para diminuir o ruído), mas ao lado de cada código está o endereço físico real dos dados deste aluno, que afinal é o dado que nos interessa nesta busca.
- Os códigos (colunas 3 a 9) estão em ordem crescente. Isto é fundamental e não pode ser modificado.
- Perceba como, em média, apenas a metade dos códigos está preenchida. Isto é necessário para permitir inclusão de novos dados. Sinaliza também um certo desperdício (overhead) de área em disco que qualquer método de acesso consome.
- Quando um nodo tem todas as chaves preenchidas e chega uma nova, que deveria (deverá) entrar nele, o nodo sofre um processo chamado split e ele é transformado em 2 nodos (cada um com metade dos espaços vazios). Note que o número máximo de chaves em qualquer nodo é sempre ímpar ($2n-1$) o que implica em que o elemento do meio sobe para o *pai* do nodo que está sendo quebrado. A metade anterior vai para um nodo *splitado* e a metade posterior vai para o outro nodo *splitado*.
- Quando as chaves vão sendo excluídas, e um nodo passa a ter menos do que a metade das chaves (o que é proibido) ele é juntado com um nodo irmão e ambos passam a ser um único nodo. Este processo recebe o nome de merge. Não é tão simples assim, há diversos casos particulares, mas a ideia é essa.

Mais um exemplo. Suponha a inclusão do número 26. Em que linha;coluna ele entra ?

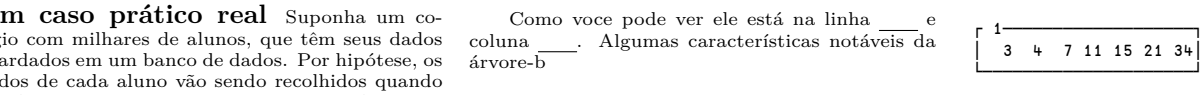
Algoritmo de Inclusão em árvore B

```
1: função INCLUA-BTREE (K:inteiro)
2: X ← raiz da árvore B
3: enquanto nodo X não é folha
4:     X ← filho correto onde K deve ser incluído
5: fim{enquanto}
6: se nodo X está cheio
7:     faça split do nodo X
8:     INCLUA-BTREE (K)
9:     abandone
10: senão
11:     inclua K no nodo X mantendo as chaves ordenadas
12: fim{se}
```

Algoritmos Estão disseminados pela comunidade. Deixo de apresentá-los aqui, pois são bastante específicos, complexos até, e mais importante do que implementá-los é entender-lhes o funcionamento. Até porque se você tiver que usar uma árvore b que não seja para aprender como ela funciona, recomendo fortemente que você use pacotes prontos, já testados e confiáveis.

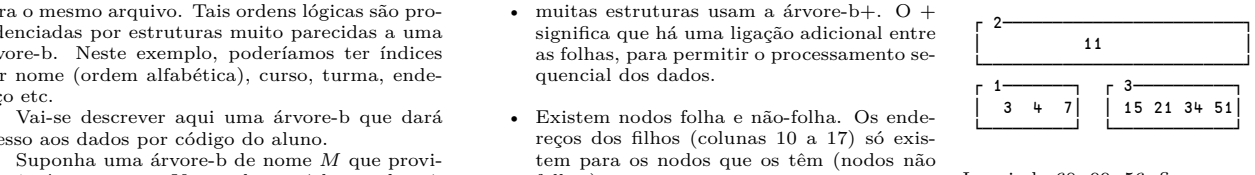
Um exemplo, passo a passo Seja uma árvore-b com $k=4$. Pela definição os nodos terão entre 3 e 7 chaves.

```
criabtree
insira 4, 7, 21, 11, 34, 15, 3
```

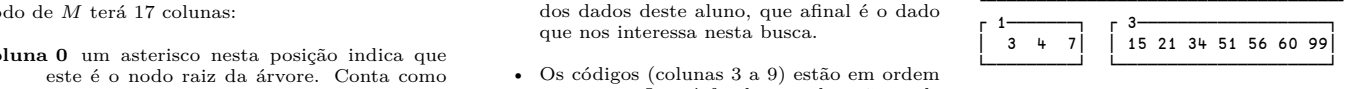


Agora, vamos inserir uma nova chave, o que vai provocar o split

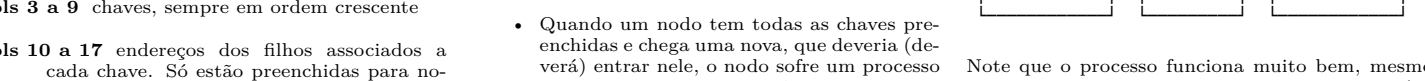
```
insira 51
```



Inserindo 60, 99, 56, fica



Inserindo 2 e 70, o 2 vai na boa para o nodo 1 enquanto o 70 força o split do nodo 3 e fica



Note que o processo funciona muito bem, mesmo que as chaves de entrada estejam desordenadas (ou em ordem aleatória). Isso é garantido pelos espaços em branco que são reservados nos nodos.

Um caso quasi-real: SYBASE

O SYBASE usa árvores B para indexar as tabelas. Existem 2 tipos de árvores B: as granuladas e as não granuladas (clustered). As granuladas são árvores esparsas (nem todo registro de dados tem entrada nos índices) nas quais os registros (na tabela original) são mantidos em ordem dos valores de índice e apenas o primeiro registro em cada página de dados tem uma entrada de índice. Árvores não granuladas são densas e cada um dos registros da tabela tem 1 entrada de índice.

Uma tabela SYBASE com índice granulado exige que seus elementos sejam mantidos em ordem dentro da tabela. Assim, inclusões podem se dar em qualquer página, a depender do conteúdo do campo de índice. Tabelas SYBASE que tem apenas índices não granulados fazem as inclusões de novos elementos sempre ao final das tabelas (na última página).

Tabelas SYBASE podem ter apenas 1 índice granulado, mas podem ter até 250 índices não granulados.

Na criação de um índice granulado, os dados da tabela devem estar colocados nas páginas em ordem.

Cada entrada de índice em sybase pode ser composta por até 16 campos e ter até 256 bytes. Estes campos não precisam estar contiguos. Só não podem ser partes de campos e resultados de cálculos.

O tamanho da página varia, mas o tamanho mais comum é 2K. Existem 5 tipos de páginas em sybase, a saber:

Páginas de Dados contém registros de dados ou registros de log. Ambos são estruturalmente diferentes, mas são construídos do mesmo jeito.

Páginas de índices contém registros de índice

Páginas de texto ou imagem contém BLOBS

Páginas de alocação contém estruturas de dados usadas para gerenciar o processo de alocacão de páginas

páginas de Estatísticas contém estatística de distribuição e uso para os índices.

Página de Dados Se os registros tem tamanho fixo, não haverá tabela de deslocamento na página e o endereço de cada registro é obtido mediante simples cálculo. Neste caso, o tamanho do registro (fixo) é guardado no campo TAMANHO MÍNIMO DE REGISTRO do header.

O header da página é um conjunto de 32 bytes contendo (entre outros): número lógico da página, anterior e próxima página lógica, identificação do objeto ao qual esta página pertence, primeira entrada de registro disponível nesta página, deslocamento do espaço livre, tamanho mínimo do registro

Depois vem a área de registros, que contém um número inteiro de slots para guardar registros. Um registro não atravessa limites de páginas

Tabela de deslocamento: Um conjunto de deslocamentos desde o início da página dando o início de cada um dos registros. Este conjunto cresce do fim da página para o início e só existe quando o tamanho do registro é variável.

Cada registro na página recebe um número de registro. É um campo de 1 byte e assim restringe-se o número de registros em uma página a 256. O número do registro usado nos índices não granulados (densos) é uma combinação do número da página com o número do registro na página.

O registro pode ocupar a página inteira (2048-32=2016). Não há tamanho mínimo para o registro, mas só pode haver 256 registros na página.

Os índices SYBASE são organizados na forma de arvores B + (o + significa que há uma ligação adicional entre as folhas, para permitir o processamento sequencial dos dados).

No exemplo aqui estudado, vão-se fazer algumas considerações/simplificações descritas a seguir.

Suponha um arquivo de pessoas contendo nome, cor preferida, bicho de estimação, alimento que gosta e idade.

Construindo um arquivo SYBASE contendo estes dados e que esteja sujeito às seguintes características:

- Registros de tamanho fixo igual a 40 bytes (9+9+9+9+4).
- Um índice granulado ou esparso (clustered) por um dos campos. No exemplo que é igual para todos, o índice esparso é por cor.
- Quatro índices não granulados (densos) pelos demais atributos.
- Páginas de dados com espaço para 4 registros e fator de preenchimento de 4:1 (usar apenas 3 espaços na carga original).
- Páginas de índices com espaço para 5 chaves (nome, cor, bicho, alimento ou idade). Lembrar que este número é impar, o que indica que no *split* vai subir a terceira chave (das 5 que existem).
- A identificação de cada registro deve ser página:número do registro na página. Por exemplo, a identificação 3:2, indica o segundo registro da terceira página.
- Nas páginas de índice (árvores-B) despreze as informações de endereço dos filhos. No exemplo, elas são irrelevantes. Faça como no exemplo.
- Os dados de entrada estão ordenados pela coluna do índice granulado.

Exemplo: Seja o conjunto de dados, no qual se fará o granulado por cor:

NOME	COR	ANIMAL	COMIDA	id.
josé	dourado	chopim	feijão	34
gabi	laranja	perú	aipim	22
sueli	lilas	gato	alho	40
antônio	musgo	vaca	batata	41
tatiana	petróleo	cavalo	arroz	37
juca	preto	águia	goiaba	36
romeu	roxo	dragão	maracujá	27
paulo	verde	urubú	rabanete	20

Com este conjunto de dados, ter-se-á a seguintes disposição nos arquivos

Páginas de Dados					
Pág1	josé	dourado	chopim	feijão	34
	gabi	laranja	perú	aipim	22
	sueli	lilas	gato	alho	40
Pág2	antônio	musgo	vaca	batata	41
	tatiana	petróleo	cavalo	arroz	37
	juca	preto	águia	goiaba	36
Pág3	romeu	roxo	dragão	maracujá	27
	paulo	verde	urubú	rabanete	20

A seguir, as páginas de índices

Páginas de índices	
Índice I1	granulado por COR
Página 1 ←raiz	dourado(1:1), musgo(2:1), roxo(3:1)
Índice I2	por NOME
Página 1	antônio(2:1), gabi(1:2)
Página 2 ←raiz	jose(1:1)
Página 3	juca(2:3), paulo(3:2), romeu(3:1), sueli(1:3), tatiana(2:2)
Índice I3	por BICHO
P1	águia(2:3), cavalo(2:2), chopim(1:1), dragão(3:1)
P2 ←raiz	gato(1:3)
P3	perú(1:2),urubú(3:2), vaca(2:1)
Índice I4	por COMIDA
P1	aipim(1:2),alho(1:3)
P2 ←raiz	arroz(2:2)
P3	batata(2:1), feijão(1:1), goiaba(2:3), maracujá(3:1), rabanete(3:2)
Índice I5	por IDADE
P1	20(3:2), 22(1:2), 27(3:1), 34(1:1), 36(2:3)
P2 ←raiz	37(2:2)
P3	40(1:3), 41(2:1)

☞ Para você fazer

Desenhe a árvore B a seguir:

```
    1 2    3    4    5    6    7    8    9 10 11 12 13 14 15 16 17
  +-------------------------------------------------------------------
 1| 1 4    31   33   34   35    0    0    0
 2| 0 6    36   41   47   59   69   74    0  1  8 22 13  5 20  9
 3| 1 4   127  128  129  130    0    0    0
 4| 1 3   147  148  149    0    0    0    0
 5| 1 6    61   62   63   65   66   68    0
 6| 1 4    89   92   93   94    0    0    0
 7| 1 5   163  164  165  166  169    0    0
 8| 1 3    37   38   40    0    0    0    0
 9| 1 7    75   77   78   81   83   84   86
*10| 0 2    88  146    0    0    0    0    0  2 11 17
 11| 0 6    95  104  114  126  131  141    0  6 19 16 21  3 23 14
 12| 1 4   172  173  175  176    0    0    0
 13| 1 6    49   50   53   54   56   57    0
 14| 1 3   142  143  144    0    0    0    0
 15| 1 7   182  187  188  189  190  194  196
 16| 1 4   105  110  111  112    0    0    0
 17| 0 4   151  160  171  178    0    0    0  4 18  7 12 15
 18| 1 4   152  153  154  156    0    0    0
 19| 1 5    96   98  100  101  103    0    0
 20| 1 3    70   71   72    0    0    0    0
 21| 1 3   115  119  122    0    0    0    0
 22| 1 4    42   44   45   46    0    0    0
 23| 1 3   133  137  140    0    0    0    0
```

Responda agora:

Qual a altura da árvore B ?	Se o 139 for incluido, qual linha,coluna na matriz?	Se o 79 for incluído, qual linha,coluna na matriz?

Mais uma Instância do mesmo exercício Desenhe a árvore B a seguir:

```
    1 2    3    4    5    6    7    8    9 10 11 12 13 14 15 16 17
  +-------------------------------------------------------------------
 1| 1 4    31   34   35   36    0    0    0
 2| 0 5    38   54   64   76   83    0    0  1 17  8  6  9 22
 3| 1 7   136  137  138  140  141  143  144
 4| 1 6   174  178  180  181  182  183    0
 5| 1 3   100  102  103    0    0    0    0
 6| 1 5    68   69   70   72   74    0    0
 7| 1 4   129  131  132  134    0    0    0
 8| 1 5    55   58   59   60   63    0    0
 9| 1 4    77   79   80   82    0    0    0
*10| 0 2    99  145    0    0    0    0    0  2 11 16
 11| 0 5   104  112  120  128  135    0    0  5 20 13 18  7  3
 12| 1 7   147  151  153  154  155  156  157
 13| 1 4   115  117  118  119    0    0    0
 14| 1 5   160  161  162  163  164    0    0
 15| 1 4   185  190  192  195    0    0    0
 16| 0 5   158  165  172  184  196    0    0  0 12 14 19  4 15 21
 17| 1 6    41   42   44   48   50   52    0
 18| 1 6   121  122  123  125  126  127    0
 19| 1 3   167  168  170    0    0    0    0
 20| 1 4   105  106  107  110    0    0    0
 21| 1 3   197  199  200    0    0    0    0
 22| 1 3    89   93   98    0    0    0    0
```

Responda agora:

Qual a altura da árvore B ?	Se o 171 for incluido, qual linha,coluna na matriz?	Se o 194 for incluído, qual linha,coluna na matriz?

CEP-UFPR-UTFPR-PUC/Pr-UP Sistemas de Informação 12/08/2024 - 11:35:52.4
Matemática aplicada Prof Dr P Kantek (pkantek@gmail.com)
Evolução de árvores: Árvore-B (e B+) VIVXq92a V: 1.02
76537 FELIPE EIJI KANO
24CC2401 - 5 primeiro exercicio do T3 _____ / / _____

Árvore B

O conceito de árvore B foi desenvolvido por Rudolf Bayer e Edward McCreight em 1972.

Eles publicaram seu trabalho no artigo *"Organization and Maintenance of Large Ordered Indices"*, que descreve as propriedades e operações básicas das árvores B.

O trabalho de Bayer e McCreight teve um impacto significativo na área de ciência da computação, e as árvores B são agora amplamente utilizadas em bancos de dados e outros sistemas de software.

As árvores B são um tipo de árvore de busca balanceada que pode armazenar um grande número de chaves e valores de maneira eficiente. Elas são mais eficientes do que as árvores de busca binárias tradicionais, pois podem ter mais filhos por nó. As árvores B são usadas em uma variedade de aplicações, incluindo bancos de dados, sistemas de arquivos e sistemas de roteamento.

Eles não explicaram o significado do nome, só nos resta especular que esse B seja de Balanceada. Se você se lembrar das árvores binárias de pesquisa, deve recordar o problema que existe quando as chaves vem (quasi) ordenadas na entrada: a árvore gerada é desbalanceada e pode chegar a ter um desempenho terrível.

Outra qualidade da árvore-B é que ela tem grau k (k filhos por nodo). Em geral este valor é escolhido de modo a otimizar a blocagem de dados no arquivo de memória secundária. De qualquer modo este k modifica o desempenho, que lembrando é proporcional a $\log_k n$, onde n é a quantidade de chaves.

Como toda árvore, ela tem uma raiz que tem de 0 a $2k-1$ chaves e consequentemente tem de 0 a $2k$ filhos. Cada um dos filhos, tem de $k-1$ até $2k-1$ chaves de 0 até $2k$ filhos. Note que a raiz é o único nodo que sofre processamento especial, necessário para quando o arquivo tem uma única chave (por exemplo).

Um caso prático real Suponha um colégio com milhares de alunos, que têm seus dados guardados em um banco de dados. Por hipótese, os dados de cada aluno vão sendo recolhidos quando eles fazem a matrícula, pelo que a ordem física dos dados no arquivo é a data da matrícula. Parece óbvio assinalar que um determinado conjunto de dados (um arquivo) só pode ter uma única ordem física em um determinado instante. Mas nada impede que existam muitas ordens lógicas diferentes para o mesmo arquivo. Tais ordens lógicas são providenciadas por estruturas muito parecidas a uma árvore-b. Neste exemplo, poderíamos ter índices por nome (ordem alfabética), curso, turma, endereço etc.

Vai-se descrever aqui uma árvore-b que dará acesso aos dados por código do aluno.

Suponha uma árvore-b de nome M que providenciará este acesso. Vamos descrevê-la com $k=4$, que é muito pouco para aplicações reais, mas fica razoável para trabalhar com este exemplo. De acordo com a definição, os nodos de M têm entre 3 e 7 códigos de aluno. Para esta dimensão, o nodo de M terá 17 colunas:

coluna 0 um asterisco nesta posição indica que este é o nodo raiz da árvore. Conta como coluna 0 porque na verdade esta informação não está fisicamente aqui, está em outro lugar.

coluna 1 indicativo de folha (1=sim; 0=não)

coluna 2 quantidade de chaves válidas neste nodo (nem sempre o valor inválido é zero)

cols 3 a 9 chaves, sempre em ordem crescente

cols 10 a 17 endereços dos filhos associados a cada chave. Só estão preenchidas para nodos não folha.

Veja um exemplo:

```
    1 2    3    4    5    6    7    8    9 10 11 12 13 14 15 16 17
  +-------------------------------------------------------------------
 1| 1 5    3    6    7    8   10    0    0
 2| 0 4   11   20   32   44    0    0    0  1 20 15  7 12
 3| 1 4  128  131  132  134    0    0    0
 4| 1 5  149  150  151  153  160    0    0
 5| 1 6   65   73   76   79   80   83    0
 6| 1 5  182  183  188  189  191    0    0
 7| 1 3   34   35   37    0    0    0    0
 8| 1 4  106  109  111  114    0    0    0
 9| 1 6  136  137  138  140  143  145    0
*10| 0 2   64  127    0    0    0    0    0  2 17 11
 11| 0 6  135  146  162  171  181  192    0  3  9  4 14 21  6 18
 12| 1 7   46   49   50   55   56   57   63
 13| 1 4   88   89   91   94    0    0    0
 14| 1 5  164  166  167  168  169    0    0
 15| 1 6   21   22   23   28   29   30    0
 16| 1 7  117  118  120  122  123  125  126
 17| 0 4   84   95  105  116    0    0    0  5 13 19  8 16
 18| 1 5  193  195  197  199  200    0    0
 19| 1 5   96   98  100  102  103    0    0
 20| 1 4   14   15   17   19    0    0    0
 21| 1 3  174  175  177    0    0    0    0
```

Antes de continuar no exercício, você deve tentar entender o funcionamento da árvore-b.

- Desenhe a árvore
- identifique a necessidade da coluna 1. Idem para a coluna 2
- Qual a altura da árvore ?
- Quando e como a altura da árvore cresce ?

Agoram suponha que o elemento 187 é incluído. Qual a linha;coluna da matriz acima que é modificada, ou seja onde o elemento recém incluido entra ? Localize-o e veja:

```
    1 2    3    4    5    6    7    8    9 10 11 12 13 14 15 16 17
  +-------------------------------------------------------------------
 1| 1 5    3    6    7    8   10    0    0
 2| 0 4   11   20   32   44    0    0    0  1 20 15  7 12
 3| 1 4  128  131  132  134    0    0    0
 4| 1 5  149  150  151  153  160    0    0
 5| 1 6   65   73   76   79   80   83    0
 6| 1 6  182  183(187)188  189  191    0
 7| 1 3   34   35   37    0    0    0    0
 8| 1 4  106  109  111  114    0    0    0
 9| 1 6  136  137  138  140  143  145    0
*10| 0 2   64  127    0    0    0    0    0  2 17 11
 11| 0 6  135  146  162  171  181  192    0  3  9  4 14 21  6 18
 12| 1 7   46   49   50   55   56   57   63
 13| 1 4   88   89   91   94    0    0    0
 14| 1 5  164  166  167  168  169    0    0
 15| 1 6   21   22   23   28   29   30    0
 16| 1 7  117  118  120  122  123  125  126
 17| 0 4   84   95  105  116    0    0    0  5 13 19  8 16
 18| 1 5  193  195  197  199  200    0    0
 19| 1 5   96   98  100  102  103    0    0
 20| 1 4   14   15   17   19    0    0    0
 21| 1 3  174  175  177    0    0    0    0
```

Como voce pode ver ele está na linha ____ e coluna ____. Algumas características notáveis da árvore-b

- Figurinha carimbada no mundo do software: usam B-trees: SYBASE e do Harbour (antigo Clipper). Também são usados no Oracle, MySQL, PostgreSQL, Microsoft SQL e DB2, da IBM.
- muitas estruturas usam a árvore-b+. O + significa que há uma ligação adicional entre as folhas, para permitir o processamento sequencial dos dados.
- Existem nodos folha e não-folha. Os endereços dos filhos (colunas 10 a 17) só existem para os nodos que os têm (nodos não folhas).
- Os códigos de aluno (colunas 3 a 9 não mostram, para diminuir o ruído), mas ao lado de cada código está o endereço físico real dos dados deste aluno, que afinal é o dado que nos interessa nesta busca.
- Os códigos (colunas 3 a 9) estão em ordem crescente. Isto é fundamental e não pode ser modificado.
- Perceba como, em média, apenas a metade dos códigos está preenchida. Isto é necessário para permitir inclusão de novos dados. Sinaliza também um certo desperdício (overhead) de área em disco que qualquer método de acesso consome.
- Quando um nodo tem todas as chaves preenchidas e chega uma nova, que deveria (deverá) entrar nele, o nodo sofre um processo chamado split e ele é transformado em 2 nodos (cada um com metade dos espaços vazios). Note que o número máximo de chaves em qualquer nodo é sempre ímpar ($2n-1$) o que implica em que o elemento do meio sobe para o *pai* do nodo que está sendo quebrado. A metade anterior vai para um nodo *splitado* e a metade posterior vai para o outro nodo *splitado*.
- Quando as chaves vão sendo excluídas, e um nodo passa a ter menos do que a metade das chaves (o que é proibido) ele é juntado com um nodo irmão e ambos passam a ser um único nodo. Este processo recebe o nome de merge. Não é tão simples assim, há diversos casos particulares, mas a ideia é essa.

Mais um exemplo. Suponha a inclusão do número 26. Em que linha;coluna ele entra ?

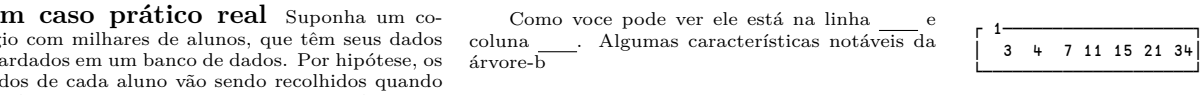
Algoritmo de Inclusão em árvore B

```
1: função INCLUA-BTREE (K:inteiro)
2: X ← raiz da árvore B
3: enquanto nodo X não é folha
4:     X ← filho correto onde K deve ser incluído
5: fim{enquanto}
6: se nodo X está cheio
7:     faça split do nodo X
8:     INCLUA-BTREE (K)
9:     abandone
10: senão
11:     inclua K no nodo X mantendo as chaves ordenadas
12: fim{se}
```

Algoritmos Estão disseminados pela comunidade. Deixo de apresentá-los aqui, pois são bastante específicos, complexos até, e mais importante do que implementá-los é entender-lhes o funcionamento. Até porque se você tiver que usar uma árvore b que não seja para aprender como ela funciona, recomendo fortemente que você use pacotes prontos, já testados e confiáveis.

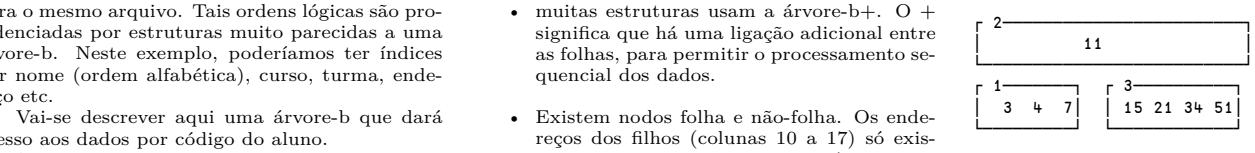
Um exemplo, passo a passo Seja uma árvore-b com $k=4$. Pela definição os nodos terão entre 3 e 7 chaves.

```
criabtree
insira 4, 7, 21, 11, 34, 15, 3
```

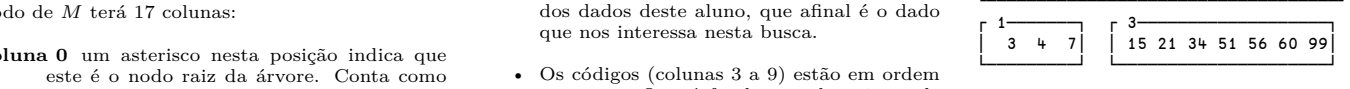


Agora, vamos inserir uma nova chave, o que vai provocar o split

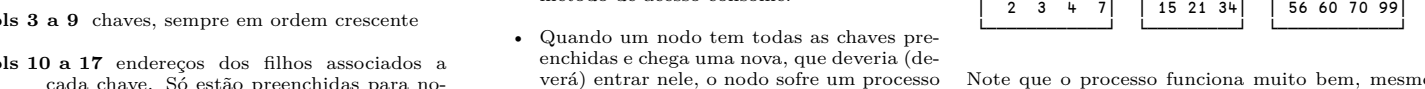
```
insira 51
```



Inserindo 60, 99, 56, fica



Inserindo 2 e 70, o 2 vai na boa para o nodo 1 enquanto o 70 força o split do nodo 3 e fica



Note que o processo funciona muito bem, mesmo que as chaves de entrada estejam desordenadas (ou em ordem aleatória). Isso é garantido pelos espaços em branco que são reservados nos nodos.

Um caso quasi-real: SYBASE

O SYBASE usa árvores B para indexar as tabelas. Existem 2 tipos de árvores B: as granuladas e as não granuladas (clustered). As granuladas são árvores esparsas (nem todo registro de dados tem entrada nos índices) nas quais os registros (na tabela original) são mantidos em ordem dos valores de índice e apenas o primeiro registro em cada página de dados tem uma entrada de índice. Árvores não granuladas são densas e cada um dos registros da tabela tem 1 entrada de índice.

Uma tabela SYBASE com índice granulado exige que seus elementos sejam mantidos em ordem dentro da tabela. Assim, inclusões podem se dar em qualquer página, a depender do conteúdo do campo de índice. Tabelas SYBASE que tem apenas índices não granulados fazem as inclusões de novos elementos sempre ao final das tabelas (na última página).

Tabelas SYBASE podem ter apenas 1 índice granulado, mas podem ter até 250 índices não granulados.

Na criação de um índice granulado, os dados da tabela devem estar colocados nas páginas em ordem.

Cada entrada de índice em sybase pode ser composta por até 16 campos e ter até 256 bytes. Estes campos não precisam estar contiguos. Só não podem ser partes de campos e resultados de cálculos.

O tamanho da página varia, mas o tamanho mais comum é 2K. Existem 5 tipos de páginas em sybase, a saber:

Páginas de Dados contém registros de dados ou registros de log. Ambos são estruturalmente diferentes, mas são construídos do mesmo jeito.

Páginas de índices contém registros de índice

Páginas de texto ou imagem contém BLOBS

Páginas de alocação contém estruturas de dados usadas para gerenciar o processo de alocacão de páginas

páginas de Estatísticas contém estatística de distribuição e uso para os índices.

Página de Dados Se os registros tem tamanho fixo, não haverá tabela de deslocamento na página e o endereço de cada registro é obtido mediante simples cálculo. Neste caso, o tamanho do registro (fixo) é guardado no campo TAMANHO MÍNIMO DE REGISTRO do header.

O header da página é um conjunto de 32 bytes contendo (entre outros): número lógico da página, anterior e próxima página lógica, identificação do objeto ao qual esta página pertence, primeira entrada de registro disponível nesta página, deslocamento do espaço livre, tamanho mínimo do registro

Depois vem a área de registros, que contém um número inteiro de slots para guardar registros. Um registro não atravessa limites de páginas

Tabela de deslocamento: Um conjunto de deslocamentos desde o início da página dando o início de cada um dos registros. Este conjunto cresce do fim da página para o início e só existe quando o tamanho do registro é variável.

Cada registro na página recebe um número de registro. É um campo de 1 byte e assim restringe-se o número de registros em uma página a 256. O número do registro usado nos índices não granulados (densos) é uma combinação do número da página com o número do registro na página.

O registro pode ocupar a página inteira (2048-32=2016). Não há tamanho mínimo para o registro, mas só pode haver 256 registros na página.

Os índices SYBASE são organizados na forma de arvores B + (o + significa que há uma ligação adicional entre as folhas, para permitir o processamento sequencial dos dados).

No exemplo aqui estudado, vão-se fazer algumas considerações/simplificações descritas a seguir.

Suponha um arquivo de pessoas contendo nome, cor preferida, bicho de estimação, alimento que gosta e idade.

Construindo um arquivo SYBASE contendo estes dados e que esteja sujeito às seguintes características:

- Registros de tamanho fixo igual a 40 bytes (9+9+9+9+4).
- Um índice granulado ou esparso (clustered) por um dos campos. No exemplo que é igual para todos, o índice esparso é por cor.
- Quatro índices não granulados (densos) pelos demais atributos.
- Páginas de dados com espaço para 4 registros e fator de preenchimento de 4:1 (usar apenas 3 espaços na carga original).
- Páginas de índices com espaço para 5 chaves (nome, cor, bicho, alimento ou idade). Lembrar que este número é impar, o que indica que no *split* vai subir a terceira chave (das 5 que existem).
- A identificação de cada registro deve ser página:número do registro na página. Por exemplo, a identificação 3:2, indica o segundo registro da terceira página.
- Nas páginas de índice (árvores-B) despreze as informações de endereço dos filhos. No exemplo, elas são irrelevantes. Faça como no exemplo.
- Os dados de entrada estão ordenados pela coluna do índice granulado.

Exemplo: Seja o conjunto de dados, no qual se fará o granulado por cor:

NOME	COR	ANIMAL	COMIDA	id.
josé	dourado	chopim	feijão	34
gabi	laranja	perú	aipim	22
sueli	lilas	gato	alho	40
antônio	musgo	vaca	batata	41
tatiana	petróleo	cavalo	arroz	37
juca	preto	águia	goiaba	36
romeu	roxo	dragão	maracujá	27
paulo	verde	urubú	rabanete	20

Com este conjunto de dados, ter-se-á a seguintes disposição nos arquivos

Páginas de Dados					
Pág1	josé	dourado	chopim	feijão	34
	gabi	laranja	perú	aipim	22
	sueli	lilas	gato	alho	40
Pág2	antônio	musgo	vaca	batata	41
	tatiana	petróleo	cavalo	arroz	37
	juca	preto	águia	goiaba	36
Pág3	romeu	roxo	dragão	maracujá	27
	paulo	verde	urubú	rabanete	20

A seguir, as páginas de índices

Páginas de índices	
Índice I1	granulado por COR
Página 1 ←raiz	dourado(1:1), musgo(2:1), roxo(3:1)
Índice I2	por NOME
Página 1	antônio(2:1), gabi(1:2)
Página 2 ←raiz	jose(1:1)
Página 3	juca(2:3), paulo(3:2), romeu(3:1), sueli(1:3), tatiana(2:2)
Índice I3	por BICHO
P1	águia(2:3), cavalo(2:2), chopim(1:1), dragão(3:1)
P2 ←raiz	gato(1:3)
P3	perú(1:2),urubú(3:2), vaca(2:1)
Índice I4	por COMIDA
P1	aipim(1:2),alho(1:3)
P2 ←raiz	arroz(2:2)
P3	batata(2:1), feijão(1:1), goiaba(2:3), maracujá(3:1), rabanete(3:2)
Índice I5	por IDADE
P1	20(3:2), 22(1:2), 27(3:1), 34(1:1), 36(2:3)
P2 ←raiz	37(2:2)
P3	40(1:3), 41(2:1)

☞ Para você fazer

Desenhe a árvore B a seguir:

```
    1 2    3    4    5    6    7    8    9 10 11 12 13 14 15 16 17
  +-----------------------------------------------------------------
 1| 1 6    31   34   35   36   37   38    0
 2| 0 3    39   59   64    0    0    0    0  1 13   7 12
 3| 1 3   141  143  144    0    0    0    0
 4| 1 7    85   89   92   93   95   96   97
 5| 1 5   159  160  161  165  167    0    0
 6| 1 3   118  121  122    0    0    0    0
 7| 1 3    61   62   63    0    0    0    0
 8| 1 6   180  181  182  183  184  189    0
 9| 1 5    99  101  102  107  108    0    0
*10| 0 3    74  117  157    0    0    0    0  2 22 11 18
 11| 0 4   123  131  140  146    0    0    0  6 20 15  3 14
 12| 1 7    65   66   67   68   70   72   73
 13| 1 7    40   43   44   47   49   55   58
 14| 1 7   148  149  151  152  153  154  155
 15| 1 4   133  134  136  138    0    0    0
 16| 1 5   169  173  174  176  177    0    0
 17| 1 4   194  195  198  199    0    0    0
 18| 0 3   168  178  190    0    0    0    0  5 16   8 17
 19| 1 4    76   77   78   79    0    0    0
 20| 1 4   125  126  128  129    0    0    0
 21| 1 4   112  114  115  116    0    0    0
 22| 0 3    80   98  110    0    0    0    0 19   4   9 21
```

Responda agora:

Qual a altura da árvore B ?	Se o 41 for incluido, qual linha,coluna na matriz?	Se o 42 for incluído, qual linha,coluna na matriz?

Mais uma Instância do mesmo exercício Desenhe a árvore B a seguir:

```
    1 2    3    4    5    6    7    8    9 10 11 12 13 14 15 16 17
  +-----------------------------------------------------------------
 1| 1 6    31   34   35   36   37   38    0
 2| 0 6    39   47   57   68   82   89    0  1 15   7 17   4   9 14
 3| 1 6   122  123  125  126  127  129    0
 4| 1 7    69   70   71   72   73   76   80
 5| 1 4   180  185  187  191    0    0    0
 6| 1 6   102  104  105  107  108  110    0
 7| 1 4    50   52   53   56    0    0    0
 8| 1 5   143  145  147  151  153    0    0
 9| 1 5    83   84   85   87   88    0    0
*10| 0 2   101  141    0    0    0    0    0  2 11 19
 11| 0 3   112  119  131    0    0    0    0  6 18   3 16
 12| 1 3   161  164  167    0    0    0    0
 13| 1 5   193  194  196  199  200    0    0
 14| 1 6    90   92   93   96   98   99    0
 15| 1 6    41   42   43   44   45   46    0
 16| 1 5   132  136  137  139  140    0    0
 17| 1 5    60   62   64   66   67    0    0
 18| 1 4   113  116  117  118    0    0    0
 19| 0 5   154  160  168  179  192    0    0  8 20 12 21   5 13
 20| 1 3   155  158  159    0    0    0    0
 21| 1 4   171  174  177  178    0    0    0
```

Responda agora:

Qual a altura da árvore B ?	Se o 146 for incluido, qual linha,coluna na matriz?	Se o 81 for incluído, qual linha,coluna na matriz?

CEP-UFPR-UTFPR-PUC/Pr-UP Sistemas de Informação 12/08/2024 - 11:35:52.4
Matemática aplicada Prof Dr P Kantek (pkantek@gmail.com)
Evolução de árvores: Árvore-B (e B+) VIVXq92a V: 1.02
76544 FELIPE JESMIEL LEITE
24CC2401 - 6 primeiro exercicio do T3 _____ / / _____

Árvore B

O conceito de árvore B foi desenvolvido por Rudolf Bayer e Edward McCreight em 1972.

Eles publicaram seu trabalho no artigo *"Organization and Maintenance of Large Ordered Indices"*, que descreve as propriedades e operações básicas das árvores B.

O trabalho de Bayer e McCreight teve um impacto significativo na área de ciência da computação, e as árvores B são agora amplamente utilizadas em bancos de dados e outros sistemas de software.

As árvores B são um tipo de árvore de busca balanceada que pode armazenar um grande número de chaves e valores de maneira eficiente. Elas são mais eficientes do que as árvores de busca binárias tradicionais, pois podem ter mais filhos por nó. As árvores B são usadas em uma variedade de aplicações, incluindo bancos de dados, sistemas de arquivos e sistemas de roteamento.

Eles não explicaram o significado do nome, só nos resta especular que esse B seja de Balanceada. Se você se lembrar das árvores binárias de pesquisa, deve recordar o problema que existe quando as chaves vem (quasi) ordenadas na entrada: a árvore gerada é desbalanceada e pode chegar a ter um desempenho terrível.

Outra qualidade da árvore-B é que ela tem grau k (k filhos por nodo). Em geral este valor é escolhido de modo a otimizar a blocagem de dados no arquivo de memória secundária. De qualquer modo este k modifica o desempenho, que lembrando é proporcional a $\log_k n$, onde n é a quantidade de chaves.

Como toda árvore, ela tem uma raiz que tem de 0 a $2k-1$ chaves e consequentemente tem de 0 a $2k$ filhos. Cada um dos filhos, tem de $k-1$ até $2k-1$ chaves de 0 até $2k$ filhos. Note que a raiz é o único nodo que sofre processamento especial, necessário para quando o arquivo tem uma única chave (por exemplo).

Um caso prático real Suponha um colégio com milhares de alunos, que têm seus dados guardados em um banco de dados. Por hipótese, os dados de cada aluno vão sendo recolhidos quando eles fazem a matrícula, pelo que a ordem física dos dados no arquivo é a data da matrícula. Parece óbvio assinalar que um determinado conjunto de dados (um arquivo) só pode ter uma única ordem física em um determinado instante. Mas nada impede que existam muitas ordens lógicas diferentes para o mesmo arquivo. Tais ordens lógicas são providenciadas por estruturas muito parecidas a uma árvore-b. Neste exemplo, poderíamos ter índices por nome (ordem alfabética), curso, turma, endereço etc.

Vai-se descrever aqui uma árvore-b que dará acesso aos dados por código do aluno.

Suponha uma árvore-b de nome M que providenciará este acesso. Vamos descrevê-la com $k=4$, que é muito pouco para aplicações reais, mas fica razoável para trabalhar com este exemplo. De acordo com a definição, os nodos de M têm entre 3 e 7 códigos de aluno. Para esta dimensão, o nodo de M terá 17 colunas:

coluna 0 um asterisco nesta posição indica que este é o nodo raiz da árvore. Conta como coluna 0 porque na verdade esta informação não está fisicamente aqui, está em outro lugar.

coluna 1 indicativo de folha (1=sim; 0=não)

coluna 2 quantidade de chaves válidas neste nodo (nem sempre o valor inválido é zero)

cols 3 a 9 chaves, sempre em ordem crescente

cols 10 a 17 endereços dos filhos associados a cada chave. Só estão preenchidas para nodos não folha.

Veja um exemplo:

```
    1 2    3    4    5    6    7    8    9 10 11 12 13 14 15 16 17
  +-------------------------------------------------------------------
 1| 1 5    3    6    7    8   10    0    0
 2| 0 4   11   20   32   44    0    0    0  1 20 15  7 12
 3| 1 4  128  131  132  134    0    0    0
 4| 1 5  149  150  151  153  160    0    0
 5| 1 6   65   73   76   79   80   83    0
 6| 1 5  182  183  188  189  191    0    0
 7| 1 3   34   35   37    0    0    0    0
 8| 1 4  106  109  111  114    0    0    0
 9| 1 6  136  137  138  140  143  145    0
*10| 0 2   64  127    0    0    0    0    0  2 17 11
 11| 0 6  135  146  162  171  181  192    0  3  9  4 14 21  6 18
 12| 1 7   46   49   50   55   56   57   63
 13| 1 4   88   89   91   94    0    0    0
 14| 1 5  164  166  167  168  169    0    0
 15| 1 6   21   22   23   28   29   30    0
 16| 1 7  117  118  120  122  123  125  126
 17| 0 4   84   95  105  116    0    0    0  5 13 19  8 16
 18| 1 5  193  195  197  199  200    0    0
 19| 1 5   96   98  100  102  103    0    0
 20| 1 4   14   15   17   19    0    0    0
 21| 1 3  174  175  177    0    0    0    0
```

Antes de continuar no exercício, você deve tentar entender o funcionamento da árvore-b.

- Desenhe a árvore
- identifique a necessidade da coluna 1. Idem para a coluna 2
- Qual a altura da árvore ?
- Quando e como a altura da árvore cresce ?

Agoram suponha que o elemento 187 é incluído. Qual a linha;coluna da matriz acima que é modificada, ou seja onde o elemento recém incluido entra ? Localize-o e veja:

```
    1 2    3    4    5    6    7    8    9 10 11 12 13 14 15 16 17
  +-------------------------------------------------------------------
 1| 1 5    3    6    7    8   10    0    0
 2| 0 4   11   20   32   44    0    0    0  1 20 15  7 12
 3| 1 4  128  131  132  134    0    0    0
 4| 1 5  149  150  151  153  160    0    0
 5| 1 6   65   73   76   79   80   83    0
 6| 1 6  182  183(187)188  189  191    0
 7| 1 3   34   35   37    0    0    0    0
 8| 1 4  106  109  111  114    0    0    0
 9| 1 6  136  137  138  140  143  145    0
*10| 0 2   64  127    0    0    0    0    0  2 17 11
 11| 0 6  135  146  162  171  181  192    0  3  9  4 14 21  6 18
 12| 1 7   46   49   50   55   56   57   63
 13| 1 4   88   89   91   94    0    0    0
 14| 1 5  164  166  167  168  169    0    0
 15| 1 6   21   22   23   28   29   30    0
 16| 1 7  117  118  120  122  123  125  126
 17| 0 4   84   95  105  116    0    0    0  5 13 19  8 16
 18| 1 5  193  195  197  199  200    0    0
 19| 1 5   96   98  100  102  103    0    0
 20| 1 4   14   15   17   19    0    0    0
 21| 1 3  174  175  177    0    0    0    0
```

Como voce pode ver ele está na linha ____ e coluna ____. Algumas características notáveis da árvore-b

- Figurinha carimbada no mundo do software: usam B-trees: SYBASE e do Harbour (antigo Clipper). Também são usados no Oracle, MySQL, PostgreSQL, Microsoft SQL e DB2, da IBM.
- muitas estruturas usam a árvore-b+. O + significa que há uma ligação adicional entre as folhas, para permitir o processamento sequencial dos dados.
- Existem nodos folha e não-folha. Os endereços dos filhos (colunas 10 a 17) só existem para os nodos que os têm (nodos não folhas).
- Os códigos de aluno (colunas 3 a 9 não mostram, para diminuir o ruído), mas ao lado de cada código está o endereço físico real dos dados deste aluno, que afinal é o dado que nos interessa nesta busca.
- Os códigos (colunas 3 a 9) estão em ordem crescente. Isto é fundamental e não pode ser modificado.
- Perceba como, em média, apenas a metade dos códigos está preenchida. Isto é necessário para permitir inclusão de novos dados. Sinaliza também um certo desperdício (overhead) de área em disco que qualquer método de acesso consome.
- Quando um nodo tem todas as chaves preenchidas e chega uma nova, que deveria (deverá) entrar nele, o nodo sofre um processo chamado split e ele é transformado em 2 nodos (cada um com metade dos espaços vazios). Note que o número máximo de chaves em qualquer nodo é sempre ímpar ($2n-1$) o que implica em que o elemento do meio sobe para o *pai* do nodo que está sendo quebrado. A metade anterior vai para um nodo *splitado* e a metade posterior vai para o outro nodo *splitado*.
- Quando as chaves vão sendo excluídas, e um nodo passa a ter menos do que a metade das chaves (o que é proibido) ele é juntado com um nodo irmão e ambos passam a ser um único nodo. Este processo recebe o nome de merge. Não é tão simples assim, há diversos casos particulares, mas a ideia é essa.

Mais um exemplo. Suponha a inclusão do número 26. Em que linha;coluna ele entra ?

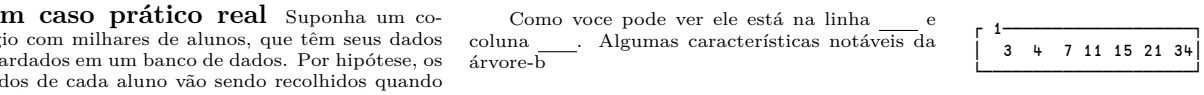
Algoritmo de Inclusão em árvore B

```
1: função INCLUA-BTREE (K:inteiro)
2: X ← raiz da árvore B
3: enquanto nodo X não é folha
4:     X ← filho correto onde K deve ser incluído
5: fim{enquanto}
6: se nodo X está cheio
7:     faça split do nodo X
8:     INCLUA-BTREE (K)
9:     abandone
10: senão
11:     inclua K no nodo X mantendo as chaves ordenadas
12: fim{se}
```

Algoritmos Estão disseminados pela comunidade. Deixo de apresentá-los aqui, pois são bastante específicos, complexos até, e mais importante do que implementá-los é entender-lhes o funcionamento. Até porque se você tiver que usar uma árvore b que não seja para aprender como ela funciona, recomendo fortemente que você use pacotes prontos, já testados e confiáveis.

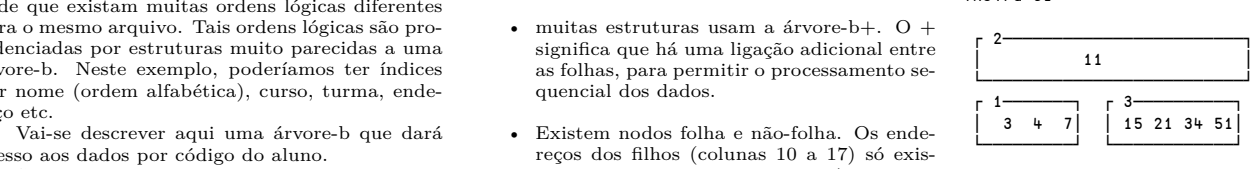
Um exemplo, passo a passo Seja uma árvore-b com $k=4$. Pela definição os nodos terão entre 3 e 7 chaves.

```
criabtree
insira 4, 7, 21, 11, 34, 15, 3
```

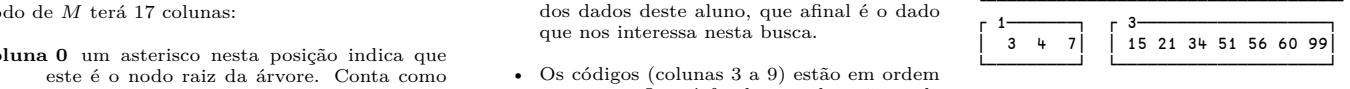


Agora, vamos inserir uma nova chave, o que vai provocar o split

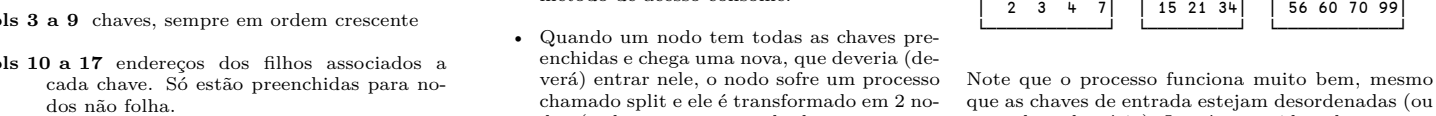
```
insira 51
```



Inserindo 60, 99, 56, fica



Inserindo 2 e 70, o 2 vai na boa para o nodo 1 enquanto o 70 força o split do nodo 3 e fica



Note que o processo funciona muito bem, mesmo que as chaves de entrada estejam desordenadas (ou em ordem aleatória). Isso é garantido pelos espaços em branco que são reservados nos nodos.

Um caso quasi-real: SYBASE

O SYBASE usa árvores B para indexar as tabelas. Existem 2 tipos de árvores B: as granuladas e as não granuladas (clustered). As granuladas são árvores esparsas (nem todo registro de dados tem entrada nos índices) nas quais os registros (na tabela original) são mantidos em ordem dos valores de índice e apenas o primeiro registro em cada página de dados tem uma entrada de índice. Árvores não granuladas são densas e cada um dos registros da tabela tem 1 entrada de índice.

Uma tabela SYBASE com índice granulado exige que seus elementos sejam mantidos em ordem dentro da tabela. Assim, inclusões podem se dar em qualquer página, a depender do conteúdo do campo de índice. Tabelas SYBASE que tem apenas índices não granulados fazem as inclusões de novos elementos sempre ao final das tabelas (na última página).

Tabelas SYBASE podem ter apenas 1 índice granulado, mas podem ter até 250 índices não granulados.

Na criação de um índice granulado, os dados da tabela devem estar colocados nas páginas em ordem.

Cada entrada de índice em sybase pode ser composta por até 16 campos e ter até 256 bytes. Estes campos não precisam estar contiguos. Só não podem ser partes de campos e resultados de cálculos.

O tamanho da página varia, mas o tamanho mais comum é 2K. Existem 5 tipos de páginas em sybase, a saber:

Páginas de Dados contém registros de dados ou registros de log. Ambos são estruturalmente diferentes, mas são construídos do mesmo jeito.

Páginas de índices contém registros de índice

Páginas de texto ou imagem contém BLOBS

Páginas de alocação contém estruturas de dados usadas para gerenciar o processo de alocacão de páginas

páginas de Estatísticas contém estatística de distribuição e uso para os índices.

Página de Dados Se os registros tem tamanho fixo, não haverá tabela de deslocamento na página e o endereço de cada registro é obtido mediante simples cálculo. Neste caso, o tamanho do registro (fixo) é guardado no campo TAMANHO MÍNIMO DE REGISTRO do header.

O header da página é um conjunto de 32 bytes contendo (entre outros): número lógico da página, anterior e próxima página lógica, identificação do objeto ao qual esta página pertence, primeira entrada de registro disponível nesta página, deslocamento do espaço livre, tamanho mínimo do registro

Depois vem a área de registros, que contém um número inteiro de slots para guardar registros. Um registro não atravessa limites de páginas

Tabela de deslocamento: Um conjunto de deslocamentos desde o início da página dando o início de cada um dos registros. Este conjunto cresce do fim da página para o início e só existe quando o tamanho do registro é variável.

Cada registro na página recebe um número de registro. É um campo de 1 byte e assim restringe-se o número de registros em uma página a 256. O número do registro usado nos índices não granulados (densos) é uma combinação do número da página com o número do registro na página.

O registro pode ocupar a página inteira (2048-32=2016). Não há tamanho mínimo para o registro, mas só pode haver 256 registros na página.

Os índices SYBASE são organizados na forma de arvores B + (o + significa que há uma ligação adicional entre as folhas, para permitir o processamento sequencial dos dados).

No exemplo aqui estudado, vão-se fazer algumas considerações/simplificações descritas a seguir.

Suponha um arquivo de pessoas contendo nome, cor preferida, bicho de estimação, alimento que gosta e idade.

Construindo um arquivo SYBASE contendo estes dados e que esteja sujeito às seguintes características:

- Registros de tamanho fixo igual a 40 bytes (9+9+9+9+4).
- Um índice granulado ou esparso (clustered) por um dos campos. No exemplo que é igual para todos, o índice esparso é por cor.
- Quatro índices não granulados (densos) pelos demais atributos.
- Páginas de dados com espaço para 4 registros e fator de preenchimento de 4:1 (usar apenas 3 espaços na carga original).
- Páginas de índices com espaço para 5 chaves (nome, cor, bicho, alimento ou idade). Lembrar que este número é impar, o que indica que no *split* vai subir a terceira chave (das 5 que existem).
- A identificação de cada registro deve ser página:número do registro na página. Por exemplo, a identificação 3:2, indica o segundo registro da terceira página.
- Nas páginas de índice (árvores-B) despreze as informações de endereço dos filhos. No exemplo, elas são irrelevantes. Faça como no exemplo.
- Os dados de entrada estão ordenados pela coluna do índice granulado.

Exemplo: Seja o conjunto de dados, no qual se fará o granulado por cor:

NOME	COR	ANIMAL	COMIDA	id.
josé	dourado	chopim	feijão	34
gabi	laranja	perú	aipim	22
sueli	lilas	gato	alho	40
antônio	musgo	vaca	batata	41
tatiana	petróleo	cavalo	arroz	37
juca	preto	águia	goiaba	36
romeu	roxo	dragão	maracujá	27
paulo	verde	urubú	rabanete	20

Com este conjunto de dados, ter-se-á a seguintes disposição nos arquivos

Páginas de Dados					
Pág1	josé	dourado	chopim	feijão	34
	gabi	laranja	perú	aipim	22
	sueli	lilas	gato	alho	40
Pág2	antônio	musgo	vaca	batata	41
	tatiana	petróleo	cavalo	arroz	37
	juca	preto	águia	goiaba	36
Pág3	romeu	roxo	dragão	maracujá	27
	paulo	verde	urubú	rabanete	20

A seguir, as páginas de índices

Páginas de índices	
Índice I1	granulado por COR
Página 1 ←raiz	dourado(1:1), musgo(2:1), roxo(3:1)
Índice I2	por NOME
Página 1	antônio(2:1), gabi(1:2)
Página 2 ←raiz	jose(1:1)
Página 3	juca(2:3), paulo(3:2), romeu(3:1), sueli(1:3), tatiana(2:2)
Índice I3	por BICHO
P1	águia(2:3), cavalo(2:2), chopim(1:1), dragão(3:1)
P2 ←raiz	gato(1:3)
P3	perú(1:2),urubú(3:2), vaca(2:1)
Índice I4	por COMIDA
P1	aipim(1:2),alho(1:3)
P2 ←raiz	arroz(2:2)
P3	batata(2:1), feijão(1:1), goiaba(2:3), maracujá(3:1), rabanete(3:2)
Índice I5	por IDADE
P1	20(3:2), 22(1:2), 27(3:1), 34(1:1), 36(2:3)
P2 ←raiz	37(2:2)
P3	40(1:3), 41(2:1)

☞ Para você fazer

Desenhe a árvore B a seguir:

```
     1 2    3    4    5    6    7    8    9 10 11 12 13 14 15 16 17
   +-------------------------------------------------------------------
 1| 1 4   32   33   34   35    0    0    0
 2| 0 5   37   49   60   70   78    0    0  1 18 12 20  4 14
 3| 1 5  103 108 109 110 111    0    0
 4| 1 4   71   73   74   77    0    0    0
 5| 1 4  125 126 127 128    0    0    0
 6| 1 5  146 148 149 150 151    0    0
 7| 1 6  193 194 195 196 199 200    0
 8| 1 7  167 168 170 171 172 173 175
 9| 1 5   85   86   88   89   90    0    0
*10| 0 2   84 124    0    0    0    0    0  2 17 11
 11| 0 6  129 145 152 166 176 191    0  5 21  6 19  8 13  7
 12| 1 3   53   54   55    0    0    0    0
 13| 1 7  178 180 181 186 188 189 190
 14| 1 4   79   80   82   83    0    0    0
 15| 1 7  114 115 118 119 120 121 123
 16| 1 3   94   95   99    0    0    0    0
 17| 0 3   91 101 113    0    0    0    0  9 16  3 15
 18| 1 6   38   40   42   43   44   48    0
 19| 1 4  153 158 162 164    0    0    0
 20| 1 4   61   62   64   69    0    0    0
 21| 1 6  131 132 135 137 141 144    0
```

Responda agora:

Qual a altura da árvore B ?	Se o 117 for incluido, qual linha,coluna na matriz?	Se o 140 for incluído, qual linha,coluna na matriz?

Mais uma Instância do mesmo exercício Desenhe a árvore B a seguir:

```
     1 2    3    4    5    6    7    8    9 10 11 12 13 14 15 16 17
   +-------------------------------------------------------------------
 1| 1 3   31   34   35    0    0    0    0
 2| 0 5   36   45   52   62   68    0    0  1 20 12 18  4 22
 3| 1 6  115 118 120 121 124 125    0
 4| 1 4   63   65   66   67    0    0    0
 5| 1 6  180 181 182 183 184 186    0
 6| 1 4  148 151 153 154    0    0    0
 7| 1 6   90   91   92   93   94   98    0
 8| 1 7   77   79   80   81   84   86   88
 9| 1 5  191 192 193 194 195    0    0
*10| 0 3   76 114 155    0    0    0    0  2 19 11 23
 11| 0 3  128 136 147    0    0    0    0  3 21 13  6
 12| 1 4   46   47   49   51    0    0    0
 13| 1 5  138 140 142 144 145    0    0
 14| 1 3  101 102 103    0    0    0    0
 15| 1 3  156 157 158    0    0    0    0
 16| 1 3  197 198 200    0    0    0    0
 17| 1 4  106 107 112 113    0    0    0
 18| 1 3   54   58   60    0    0    0    0
 19| 0 3   89 100 105    0    0    0    0  8  7 14 17
 20| 1 5   37   38   39   42   44    0    0
 21| 1 4  129 132 134 135    0    0    0
 22| 1 3   69   70   73    0    0    0    0
 23| 0 4  160 178 187 196    0    0    0 15 24  5  9 16
 24| 1 4  165 167 175 177    0    0    0
```

Responda agora:

Qual a altura da árvore B ?	Se o 126 for incluido, qual linha,coluna na matriz?	Se o 171 for incluído, qual linha,coluna na matriz?

CEP-UFPR-UTFPR-PUC/Pr-UP Sistemas de Informação 12/08/2024 - 11:35:52.4
Matemática aplicada Prof Dr P Kantek (pkantek@gmail.com)
Evolução de árvores: Árvore-B (e B+) VIVXq92a V: 1.02
76551 GABRIELA BERNARDO PASQUALOTTO
24CC2401 - 7 primeiro exercicio do T3 _____ / / _____

Árvore B

O conceito de árvore B foi desenvolvido por Rudolf Bayer e Edward McCreight em 1972.

Eles publicaram seu trabalho no artigo *"Organization and Maintenance of Large Ordered Indices"*, que descreve as propriedades e operações básicas das árvores B.

O trabalho de Bayer e McCreight teve um impacto significativo na área de ciência da computação, e as árvores B são agora amplamente utilizadas em bancos de dados e outros sistemas de software.

As árvores B são um tipo de árvore de busca balanceada que pode armazenar um grande número de chaves e valores de maneira eficiente. Elas são mais eficientes do que as árvores de busca binárias tradicionais, pois podem ter mais filhos por nó. As árvores B são usadas em uma variedade de aplicações, incluindo bancos de dados, sistemas de arquivos e sistemas de roteamento.

Eles não explicaram o significado do nome, só nos resta especular que esse B seja de Balanceada. Se você se lembrar das árvores binárias de pesquisa, deve recordar o problema que existe quando as chaves vem (quasi) ordenadas na entrada: a árvore gerada é desbalanceada e pode chegar a ter um desempenho terrível.

Outra qualidade da árvore-B é que ela tem grau k (k filhos por nodo). Em geral este valor é escolhido de modo a otimizar a blocagem de dados no arquivo de memória secundária. De qualquer modo este k modifica o desempenho, que lembrando é proporcional a $\log_k n$, onde n é a quantidade de chaves.

Como toda árvore, ela tem uma raiz que tem de 0 a $2k-1$ chaves e consequentemente tem de 0 a $2k$ filhos. Cada um dos filhos, tem de $k-1$ até $2k-1$ chaves de 0 até $2k$ filhos. Note que a raiz é o único nodo que sofre processamento especial, necessário para quando o arquivo tem uma única chave (por exemplo).

Um caso prático real Suponha um colégio com milhares de alunos, que têm seus dados guardados em um banco de dados. Por hipótese, os dados de cada aluno vão sendo recolhidos quando eles fazem a matrícula, pelo que a ordem física dos dados no arquivo é a data da matrícula. Parece óbvio assinalar que um determinado conjunto de dados (um arquivo) só pode ter uma única ordem física em um determinado instante. Mas nada impede que existam muitas ordens lógicas diferentes para o mesmo arquivo. Tais ordens lógicas são providenciadas por estruturas muito parecidas a uma árvore-b. Neste exemplo, poderíamos ter índices por nome (ordem alfabética), curso, turma, endereço etc.

Vai-se descrever aqui uma árvore-b que dará acesso aos dados por código do aluno.

Suponha uma árvore-b de nome M que providenciará este acesso. Vamos descrevê-la com $k=4$, que é muito pouco para aplicações reais, mas fica razoável para trabalhar com este exemplo. De acordo com a definição, os nodos de M têm entre 3 e 7 códigos de aluno. Para esta dimensão, o nodo de M terá 17 colunas:

coluna 0 um asterisco nesta posição indica que este é o nodo raiz da árvore. Conta como coluna 0 porque na verdade esta informação não está fisicamente aqui, está em outro lugar.

coluna 1 indicativo de folha (1=sim; 0=não)

coluna 2 quantidade de chaves válidas neste nodo (nem sempre o valor inválido é zero)

cols 3 a 9 chaves, sempre em ordem crescente

cols 10 a 17 endereços dos filhos associados a cada chave. Só estão preenchidas para nodos não folha.

Veja um exemplo:

```
    1 2    3    4    5    6    7    8    9 10 11 12 13 14 15 16 17
  +-----------------------------------------------------------------
 1| 1 5    3    6    7    8   10    0    0
 2| 0 4   11   20   32   44    0    0    0  1 20 15  7 12
 3| 1 4  128  131  132  134    0    0    0
 4| 1 5  149  150  151  153  160    0    0
 5| 1 6   65   73   76   79   80   83    0
 6| 1 5  182  183  188  189  191    0    0
 7| 1 3   34   35   37    0    0    0    0
 8| 1 4  106  109  111  114    0    0    0
 9| 1 6  136  137  138  140  143  145    0
*10| 0 2   64  127    0    0    0    0    0  2 17 11
 11| 0 6  135  146  162  171  181  192    0  3  9  4 14 21  6 18
 12| 1 7   46   49   50   55   56   57   63
 13| 1 4   88   89   91   94    0    0    0
 14| 1 5  164  166  167  168  169    0    0
 15| 1 6   21   22   23   28   29   30    0
 16| 1 7  117  118  120  122  123  125  126
 17| 0 4   84   95  105  116    0    0    0  5 13 19  8 16
 18| 1 5  193  195  197  199  200    0    0
 19| 1 5   96   98  100  102  103    0    0
 20| 1 4   14   15   17   19    0    0    0
 21| 1 3  174  175  177    0    0    0    0
```

Antes de continuar no exercício, você deve tentar entender o funcionamento da árvore-b.

- Desenhe a árvore
- identifique a necessidade da coluna 1. Idem para a coluna 2
- Qual a altura da árvore ?
- Quando e como a altura da árvore cresce ?

Agoram suponha que o elemento 187 é incluído. Qual a linha;coluna da matriz acima que é modificada, ou seja onde o elemento recém incluido entra ? Localize-o e veja:

```
    1 2    3    4    5    6    7    8    9 10 11 12 13 14 15 16 17
  +-----------------------------------------------------------------
 1| 1 5    3    6    7    8   10    0    0
 2| 0 4   11   20   32   44    0    0    0  1 20 15  7 12
 3| 1 4  128  131  132  134    0    0    0
 4| 1 5  149  150  151  153  160    0    0
 5| 1 6   65   73   76   79   80   83    0
 6| 1 6  182  183(187)188  189  191    0
 7| 1 3   34   35   37    0    0    0    0
 8| 1 4  106  109  111  114    0    0    0
 9| 1 6  136  137  138  140  143  145    0
*10| 0 2   64  127    0    0    0    0    0  2 17 11
 11| 0 6  135  146  162  171  181  192    0  3  9  4 14 21  6 18
 12| 1 7   46   49   50   55   56   57   63
 13| 1 4   88   89   91   94    0    0    0
 14| 1 5  164  166  167  168  169    0    0
 15| 1 6   21   22   23   28   29   30    0
 16| 1 7  117  118  120  122  123  125  126
 17| 0 4   84   95  105  116    0    0    0  5 13 19  8 16
 18| 1 5  193  195  197  199  200    0    0
 19| 1 5   96   98  100  102  103    0    0
 20| 1 4   14   15   17   19    0    0    0
 21| 1 3  174  175  177    0    0    0    0
```

Como voce pode ver ele está na linha ___ e coluna ____. Algumas características notáveis da árvore-b

- Figurinha carimbada no mundo do software: usam B-trees: SYBASE e do Harbour (antigo Clipper). Também são usados no Oracle, MySQL, PostgreSQL, Microsoft SQL e DB2, da IBM.
- muitas estruturas usam a árvore-b+. O + significa que há uma ligação adicional entre as folhas, para permitir o processamento sequencial dos dados.
- Existem nodos folha e não-folha. Os endereços dos filhos (colunas 10 a 17) só existem para os nodos que os têm (nodos não folhas).
- Os códigos de aluno (colunas 3 a 9 não mostram, para diminuir o ruído), mas ao lado de cada código está o endereço físico real dos dados deste aluno, que afinal é o dado que nos interessa nesta busca.
- Os códigos (colunas 3 a 9) estão em ordem crescente. Isto é fundamental e não pode ser modificado.
- Perceba como, em média, apenas a metade dos códigos está preenchida. Isto é necessário para permitir inclusão de novos dados. Sinaliza também um certo desperdício (overhead) de área em disco que qualquer método de acesso consome.
- Quando um nodo tem todas as chaves preenchidas e chega uma nova, que deveria (deverá) entrar nele, o nodo sofre um processo chamado split e ele é transformado em 2 nodos (cada um com metade dos espaços vazios). Note que o número máximo de chaves em qualquer nodo é sempre ímpar ($2n-1$) o que implica em que o elemento do meio sobe para o *pai* do nodo que está sendo quebrado. A metade anterior vai para um nodo *splitado* e a metade posterior vai para o outro nodo *splitado*.
- Quando as chaves vão sendo excluídas, e um nodo passa a ter menos do que a metade das chaves (o que é proibido) ele é juntado com um nodo irmão e ambos passam a ser um único nodo. Este processo recebe o nome de merge. Não é tão simples assim, há diversos casos particulares, mas a ideia é essa.

Mais um exemplo. Suponha a inclusão do número 26. Em que linha;coluna ele entra ?

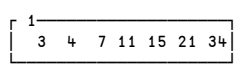
Algoritmo de Inclusão em árvore B

```
1: função INCLUA-BTREE (K:inteiro)
2: X ← raiz da árvore B
3: enquanto nodo X não é folha
4:     X ← filho correto onde K deve ser incluído
5: fim{enquanto}
6: se nodo X está cheio
7:     faça split do nodo X
8:     INCLUA-BTREE (K)
9:     abandone
10: senão
11:     inclua K no nodo X mantendo as chaves ordenadas
12: fim{se}
```

Algoritmos Estão disseminados pela comunidade. Deixo de apresentá-los aqui, pois são bastante específicos, complexos até, e mais importante do que implementá-los é entender-lhes o funcionamento. Até porque se você tiver que usar uma árvore b que não seja para aprender como ela funciona, recomendo fortemente que você use pacotes prontos, já testados e confiáveis.

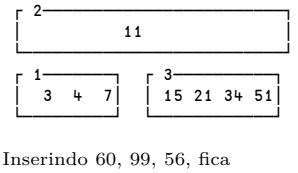
Um exemplo, passo a passo Seja uma árvore-b com $k=4$. Pela definição os nodos terão entre 3 e 7 chaves.

```
criabtree
insira 4, 7, 21, 11, 34, 15, 3
```

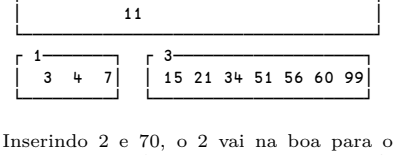


Agora, vamos inserir uma nova chave, o que vai provocar o split

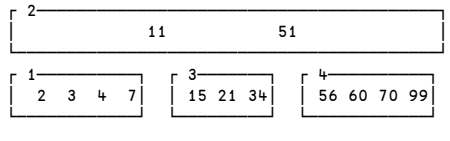
```
insira 51
```



Inserindo 60, 99, 56, fica



Inserindo 2 e 70, o 2 vai na boa para o nodo 1 enquanto o 70 força o split do nodo 3 e fica



Note que o processo funciona muito bem, mesmo que as chaves de entrada estejam desordenadas (ou em ordem aleatória). Isso é garantido pelos espaços em branco que são reservados nos nodos.

Um caso quasi-real: SYBASE

O SYBASE usa árvores B para indexar as tabelas. Existem 2 tipos de árvores B: as granuladas e as não granuladas (clustered). As granuladas são árvores esparsas (nem todo registro de dados tem entrada nos índices) nas quais os registros (na tabela original) são mantidos em ordem dos valores de índice e apenas o primeiro registro em cada página de dados tem uma entrada de índice. Árvores não granuladas são densas e cada um dos registros da tabela tem 1 entrada de índice.

Uma tabela SYBASE com índice granulado exige que seus elementos sejam mantidos em ordem dentro da tabela. Assim, inclusões podem se dar em qualquer página, a depender do conteúdo do campo de índice. Tabelas SYBASE que tem apenas índices não granulados fazem as inclusões de novos elementos sempre ao final das tabelas (na última página).

Tabelas SYBASE podem ter apenas 1 índice granulado, mas podem ter até 250 índices não granulados.

Na criação de um índice granulado, os dados da tabela devem estar colocados nas páginas em ordem.

Cada entrada de índice em sybase pode ser composta por até 16 campos e ter até 256 bytes. Estes campos não precisam estar contiguos. Só não podem ser partes de campos e resultados de cálculos.

O tamanho da página varia, mas o tamanho mais comum é 2K. Existem 5 tipos de páginas em sybase, a saber:

Páginas de Dados contém registros de dados ou registros de log. Ambos são estruturalmente diferentes, mas são construídos do mesmo jeito.

Páginas de índices contém registros de índice

Páginas de texto ou imagem contém BLOBS

Páginas de alocação contém estruturas de dados usadas para gerenciar o processo de alocacão de páginas

páginas de Estatísticas contém estatística de distribuição e uso para os índices.

Página de Dados Se os registros tem tamanho fixo, não haverá tabela de deslocamento na página e o endereço de cada registro é obtido mediante simples cálculo. Neste caso, o tamanho do registro (fixo) é guardado no campo TAMANHO MÍNIMO DE REGISTRO do header.

O header da página é um conjunto de 32 bytes contendo (entre outros): número lógico da página, anterior e próxima página lógica, identificação do objeto ao qual esta página pertence, primeira entrada de registro disponível nesta página, deslocamento do espaço livre, tamanho mínimo do registro

Depois vem a área de registros, que contém um número inteiro de slots para guardar registros. Um registro não atravessa limites de páginas

Tabela de deslocamento: Um conjunto de deslocamentos desde o início da página dando o início de cada um dos registros. Este conjunto cresce do fim da página para o início e só existe quando o tamanho do registro é variável.

Cada registro na página recebe um número de registro. É um campo de 1 byte e assim restringe-se o número de registros em uma página a 256. O número do registro usado nos índices não granulados (densos) é uma combinação do número da página com o número do registro na página.

O registro pode ocupar a página inteira (2048-32=2016). Não há tamanho mínimo para o registro, mas só pode haver 256 registros na página.

Os índices SYBASE são organizados na forma de arvores B + (o + significa que há uma ligação adicional entre as folhas, para permitir o processamento sequencial dos dados).

No exemplo aqui estudado, vão-se fazer algumas considerações/simplificações descritas a seguir.

Suponha um arquivo de pessoas contendo nome, cor preferida, bicho de estimação, alimento que gosta e idade.

Construindo um arquivo SYBASE contendo estes dados e que esteja sujeito às seguintes características:

- Registros de tamanho fixo igual a 40 bytes (9+9+9+9+4).
- Um índice granulado ou esparso (clustered) por um dos campos. No exemplo que é igual para todos, o índice esparso é por cor.
- Quatro índices não granulados (densos) pelos demais atributos.
- Páginas de dados com espaço para 4 registros e fator de preenchimento de 4:1 (usar apenas 3 espaços na carga original).
- Páginas de índices com espaço para 5 chaves (nome, cor, bicho, alimento ou idade). Lembrar que este número é impar, o que indica que no *split* vai subir a terceira chave (das 5 que existem).
- A identificação de cada registro deve ser página:número do registro na página. Por exemplo, a identificação 3:2, indica o segundo registro da terceira página.
- Nas páginas de índice (árvores-B) despreze as informações de endereço dos filhos. No exemplo, elas são irrelevantes. Faça como no exemplo.
- Os dados de entrada estão ordenados pela coluna do índice granulado.

Exemplo: Seja o conjunto de dados, no qual se fará o granulado por cor:

NOME	COR	ANIMAL	COMIDA	id.
josé	dourado	chopim	feijão	34
gabi	laranja	perú	aipim	22
sueli	lilas	gato	alho	40
antônio	musgo	vaca	batata	41
tatiana	petróleo	cavalo	arroz	37
juca	preto	águia	goiaba	36
romeu	roxo	dragão	maracujá	27
paulo	verde	urubú	rabanete	20

Com este conjunto de dados, ter-se-á a seguintes disposição nos arquivos

Páginas de Dados					
Pág1	josé	dourado	chopim	feijão	34
	gabi	laranja	perú	aipim	22
	sueli	lilas	gato	alho	40
Pág2	antônio	musgo	vaca	batata	41
	tatiana	petróleo	cavalo	arroz	37
	juca	preto	águia	goiaba	36
Pág3	romeu	roxo	dragão	maracujá	27
	paulo	verde	urubú	rabanete	20

A seguir, as páginas de índices

Páginas de índices	
Índice I1	granulado por COR
Página 1 ←raiz	dourado(1:1), musgo(2:1), roxo(3:1)
Índice I2	por NOME
Página 1	antônio(2:1), gabi(1:2)
Página 2 ←raiz	jose(1:1)
Página 3	juca(2:3), paulo(3:2), romeu(3:1), sueli(1:3), tatiana(2:2)
Índice I3	por BICHO
P1	águia(2:3), cavalo(2:2), chopim(1:1), dragão(3:1)
P2 ←raiz	gato(1:3)
P3	perú(1:2),urubú(3:2), vaca(2:1)
Índice I4	por COMIDA
P1	aipim(1:2),alho(1:3)
P2 ←raiz	arroz(2:2)
P3	batata(2:1), feijão(1:1), goiaba(2:3), maracujá(3:1), rabanete(3:2)
Índice I5	por IDADE
P1	20(3:2), 22(1:2), 27(3:1), 34(1:1), 36(2:3)
P2 ←raiz	37(2:2)
P3	40(1:3), 41(2:1)

☞ Para você fazer

Desenhe a árvore B a seguir:

```
     1  2    3    4    5    6    7    8    9  10  11  12  13  14  15  16  17
   +-------------------------------------------------------------------------
 1 | 1  5   31   33   37   38   39    0    0
 2 | 0  6   41   48   52   62   68   79    0   1  21   7  18   9   4  13
 3 | 1  4  119  121  122  123    0    0    0
 4 | 1  4   69   74   76   78    0    0    0
 5 | 1  4  153  156  158  160    0    0    0
 6 | 1  4   89   90   92   94    0    0    0
 7 | 1  3   49   50   51    0    0    0    0
 8 | 1  6  173  174  176  177  178  181    0
 9 | 1  5   63   64   65   66   67    0    0
*10| 0  2   88  151    0    0    0    0    0   2  11  17
11 | 0  5   95  103  117  124  139    0    0   6  15  19   3  12  20
12 | 1  4  125  129  137  138    0    0    0
13 | 1  6   81   82   83   84   85   86    0
14 | 1  6  183  185  187  189  191  192    0
15 | 1  3   99  101  102    0    0    0    0
16 | 1  5  164  167  169  170  171    0    0
17 | 0  4  161  172  182  194    0    0    0   5  16   8  14  22
18 | 1  4   55   56   59   61    0    0    0
19 | 1  7  105  106  108  109  111  112  113
20 | 1  6  142  143  144  147  148  150    0
21 | 1  4   42   45   46   47    0    0    0
22 | 1  3  195  197  199    0    0    0    0
```

Responda agora:

Qual a altura da árvore B ?	Se o 180 for incluido, qual linha,coluna na matriz?	Se o 127 for incluído, qual linha,coluna na matriz?

Mais uma Instância do mesmo exercício Desenhe a árvore B a seguir:

```
     1  2    3    4    5    6    7    8    9  10  11  12  13  14  15  16  17
   +-------------------------------------------------------------------------
 1 | 1  3   33   34   36    0    0    0    0
 2 | 0  6   39   48   52   63   71   86    0   1  20  15   6  18   4   8
 3 | 1  7  101  104  105  107  108  109  110
 4 | 1  7   72   74   76   79   80   83   84
 5 | 1  6  168  170  172  174  175  178    0
 6 | 1  5   53   54   57   59   61    0    0
 7 | 1  7  116  118  120  121  122  129  130
 8 | 1  6   88   90   92   93   95   99    0
 9 | 1  5  192  193  197  198  199    0    0
*10| 0  2  100  143    0    0    0    0    0   2  11  17
11 | 0  3  111  115  131    0    0    0    0   3  14   7  16
12 | 1  6  144  146  147  150  151  156    0
13 | 1  5  161  162  164  165  166    0    0
14 | 1  3  112  113  114    0    0    0    0
15 | 1  3   49   50   51    0    0    0    0
16 | 1  7  134  135  136  137  139  141  142
17 | 0  4  158  167  180  191    0    0    0  12  13   5  19   9
18 | 1  6   64   65   66   67   68   70    0
19 | 1  5  184  185  187  188  189    0    0
20 | 1  4   41   44   45   47    0    0    0
```

Responda agora:

Qual a altura da árvore B ?	Se o 173 for incluido, qual linha,coluna na matriz?	Se o 123 for incluído, qual linha,coluna na matriz?

CEP-UFPR-UTFPR-PUC/Pr-UP Sistemas de Informação 12/08/2024 - 11:35:52.4
Matemática aplicada Prof Dr P Kantek (pkantek@gmail.com)
Evolução de árvores: Árvore-B (e B+) VIVXq92a V: 1.02
76568 GUILHERME VITOR DA SILVA
24CC2401 - 8 primeiro exercicio do T3 _____ / / _____

Árvore B

O conceito de árvore B foi desenvolvido por Rudolf Bayer e Edward McCreight em 1972.

Eles publicaram seu trabalho no artigo *"Organization and Maintenance of Large Ordered Indices"*, que descreve as propriedades e operações básicas das árvores B.

O trabalho de Bayer e McCreight teve um impacto significativo na área de ciência da computação, e as árvores B são agora amplamente utilizadas em bancos de dados e outros sistemas de software.

As árvores B são um tipo de árvore de busca balanceada que pode armazenar um grande número de chaves e valores de maneira eficiente. Elas são mais eficientes do que as árvores de busca binárias tradicionais, pois podem ter mais filhos por nó. As árvores B são usadas em uma variedade de aplicações, incluindo bancos de dados, sistemas de arquivos e sistemas de roteamento.

Eles não explicaram o significado do nome, só nos resta especular que esse B seja de Balanceada. Se você se lembrar das árvores binárias de pesquisa, deve recordar o problema que existe quando as chaves vem (quasi) ordenadas na entrada: a árvore gerada é desbalanceada e pode chegar a ter um desempenho terrível.

Outra qualidade da árvore-B é que ela tem grau k (k filhos por nodo). Em geral este valor é escolhido de modo a otimizar a blocagem de dados no arquivo de memória secundária. De qualquer modo este k modifica o desempenho, que lembrando é proporcional a $\log_k n$, onde n é a quantidade de chaves.

Como toda árvore, ela tem uma raiz que tem de 0 a $2k-1$ chaves e consequentemente tem de 0 a $2k$ filhos. Cada um dos filhos, tem de $k-1$ até $2k-1$ chaves de 0 até $2k$ filhos. Note que a raiz é o único nodo que sofre processamento especial, necessário para quando o arquivo tem uma única chave (por exemplo).

Um caso prático real Suponha um colégio com milhares de alunos, que têm seus dados guardados em um banco de dados. Por hipótese, os dados de cada aluno vão sendo recolhidos quando eles fazem a matrícula, pelo que a ordem física dos dados no arquivo é a data da matrícula. Parece óbvio assinalar que um determinado conjunto de dados (um arquivo) só pode ter uma única ordem física em um determinado instante. Mas nada impede que existam muitas ordens lógicas diferentes para o mesmo arquivo. Tais ordens lógicas são providenciadas por estruturas muito parecidas a uma árvore-b. Neste exemplo, poderíamos ter índices por nome (ordem alfabética), curso, turma, endereço etc.

Vai-se descrever aqui uma árvore-b que dará acesso aos dados por código do aluno.

Suponha uma árvore-b de nome M que providenciará este acesso. Vamos descrevê-la com $k=4$, que é muito pouco para aplicações reais, mas fica razoável para trabalhar com este exemplo. De acordo com a definição, os nodos de M têm entre 3 e 7 códigos de aluno. Para esta dimensão, o nodo de M terá 17 colunas:

coluna 0 um asterisco nesta posição indica que este é o nodo raiz da árvore. Conta como coluna 0 porque na verdade esta informação não está fisicamente aqui, está em outro lugar.

coluna 1 indicativo de folha (1=sim; 0=não)

coluna 2 quantidade de chaves válidas neste nodo (nem sempre o valor inválido é zero)

cols 3 a 9 chaves, sempre em ordem crescente

cols 10 a 17 endereços dos filhos associados a cada chave. Só estão preenchidas para nodos não folha.

Veja um exemplo:

```
    1 2    3    4    5    6    7    8    9 10 11 12 13 14 15 16 17
  +-------------------------------------------------------------------
 1| 1 5    3    6    7    8   10    0    0
 2| 0 4   11   20   32   44    0    0    0  1 20 15  7 12
 3| 1 4  128  131  132  134    0    0    0
 4| 1 5  149  150  151  153  160    0    0
 5| 1 6   65   73   76   79   80   83    0
 6| 1 5  182  183  188  189  191    0    0
 7| 1 3   34   35   37    0    0    0    0
 8| 1 4  106  109  111  114    0    0    0
 9| 1 6  136  137  138  140  143  145    0
*10| 0 2   64  127    0    0    0    0    0  2 17 11
 11| 0 6  135  146  162  171  181  192    0  3  9  4 14 21  6 18
 12| 1 7   46   49   50   55   56   57   63
 13| 1 4   88   89   91   94    0    0    0
 14| 1 5  164  166  167  168  169    0    0
 15| 1 6   21   22   23   28   29   30    0
 16| 1 7  117  118  120  122  123  125  126
 17| 0 4   84   95  105  116    0    0    0  5 13 19  8 16
 18| 1 5  193  195  197  199  200    0    0
 19| 1 5   96   98  100  102  103    0    0
 20| 1 4   14   15   17   19    0    0    0
 21| 1 3  174  175  177    0    0    0    0
```

Antes de continuar no exercício, você deve tentar entender o funcionamento da árvore-b.

- Desenhe a árvore
- identifique a necessidade da coluna 1. Idem para a coluna 2
- Qual a altura da árvore ?
- Quando e como a altura da árvore cresce ?

Agoram suponha que o elemento 187 é incluído. Qual a linha;coluna da matriz acima que é modificada, ou seja onde o elemento recém incluido entra ? Localize-o e veja:

```
    1 2    3    4    5    6    7    8    9 10 11 12 13 14 15 16 17
  +-------------------------------------------------------------------
 1| 1 5    3    6    7    8   10    0    0
 2| 0 4   11   20   32   44    0    0    0  1 20 15  7 12
 3| 1 4  128  131  132  134    0    0    0
 4| 1 5  149  150  151  153  160    0    0
 5| 1 6   65   73   76   79   80   83    0
 6| 1 6  182  183(187)188  189  191    0
 7| 1 3   34   35   37    0    0    0    0
 8| 1 4  106  109  111  114    0    0    0
 9| 1 6  136  137  138  140  143  145    0
*10| 0 2   64  127    0    0    0    0    0  2 17 11
 11| 0 6  135  146  162  171  181  192    0  3  9  4 14 21  6 18
 12| 1 7   46   49   50   55   56   57   63
 13| 1 4   88   89   91   94    0    0    0
 14| 1 5  164  166  167  168  169    0    0
 15| 1 6   21   22   23   28   29   30    0
 16| 1 7  117  118  120  122  123  125  126
 17| 0 4   84   95  105  116    0    0    0  5 13 19  8 16
 18| 1 5  193  195  197  199  200    0    0
 19| 1 5   96   98  100  102  103    0    0
 20| 1 4   14   15   17   19    0    0    0
 21| 1 3  174  175  177    0    0    0    0
```

Como voce pode ver ele está na linha ____ e coluna ____. Algumas características notáveis da árvore-b

- Figurinha carimbada no mundo do software: usam B-trees: SYBASE e do Harbour (antigo Clipper). Também são usados no Oracle, MySQL, PostgreSQL, Microsoft SQL e DB2, da IBM.
- muitas estruturas usam a árvore-b+. O + significa que há uma ligação adicional entre as folhas, para permitir o processamento sequencial dos dados.
- Existem nodos folha e não-folha. Os endereços dos filhos (colunas 10 a 17) só existem para os nodos que os têm (nodos não folhas).
- Os códigos de aluno (colunas 3 a 9 não mostram, para diminuir o ruído), mas ao lado de cada código está o endereço físico real dos dados deste aluno, que afinal é o dado que nos interessa nesta busca.
- Os códigos (colunas 3 a 9) estão em ordem crescente. Isto é fundamental e não pode ser modificado.
- Perceba como, em média, apenas a metade dos códigos está preenchida. Isto é necessário para permitir inclusão de novos dados. Sinaliza também um certo desperdício (overhead) de área em disco que qualquer método de acesso consome.
- Quando um nodo tem todas as chaves preenchidas e chega uma nova, que deveria (deverá) entrar nele, o nodo sofre um processo chamado split e ele é transformado em 2 nodos (cada um com metade dos espaços vazios). Note que o número máximo de chaves em qualquer nodo é sempre ímpar ($2n-1$) o que implica em que o elemento do meio sobe para o *pai* do nodo que está sendo quebrado. A metade anterior vai para um nodo *splitado* e a metade posterior vai para o outro nodo *splitado*.
- Quando as chaves vão sendo excluídas, e um nodo passa a ter menos do que a metade das chaves (o que é proibido) ele é juntado com um nodo irmão e ambos passam a ser um único nodo. Este processo recebe o nome de merge. Não é tão simples assim, há diversos casos particulares, mas a ideia é essa.

Mais um exemplo. Suponha a inclusão do número 26. Em que linha;coluna ele entra ?

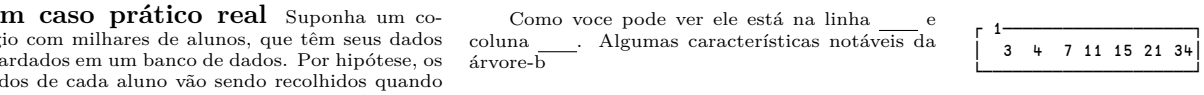
Algoritmo de Inclusão em árvore B

```
1: função INCLUA-BTREE (K:inteiro)
2: X ← raiz da árvore B
3: enquanto nodo X não é folha
4:     X ← filho correto onde K deve ser incluído
5: fim{enquanto}
6: se nodo X está cheio
7:     faça split do nodo X
8:     INCLUA-BTREE (K)
9:     abandone
10: senão
11:     inclua K no nodo X mantendo as chaves ordenadas
12: fim{se}
```

Algoritmos Estão disseminados pela comunidade. Deixo de apresentá-los aqui, pois são bastante específicos, complexos até, e mais importante do que implementá-los é entender-lhes o funcionamento. Até porque se você tiver que usar uma árvore b que não seja para aprender como ela funciona, recomendo fortemente que você use pacotes prontos, já testados e confiáveis.

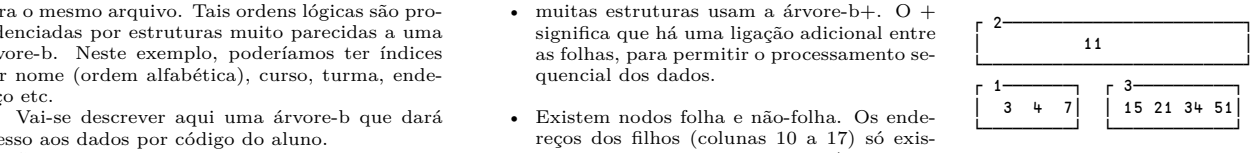
Um exemplo, passo a passo Seja uma árvore-b com $k=4$. Pela definição os nodos terão entre 3 e 7 chaves.

```
criabtree
insira 4, 7, 21, 11, 34, 15, 3
```

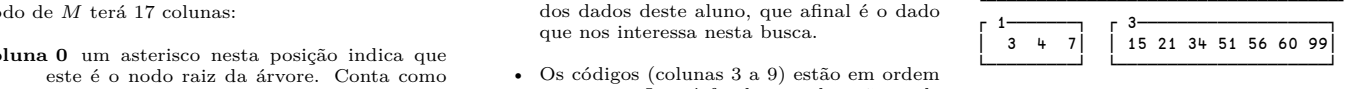


Agora, vamos inserir uma nova chave, o que vai provocar o split

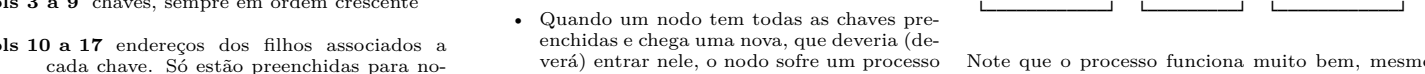
```
insira 51
```



Inserindo 60, 99, 56, fica



Inserindo 2 e 70, o 2 vai na boa para o nodo 1 enquanto o 70 força o split do nodo 3 e fica



Note que o processo funciona muito bem, mesmo que as chaves de entrada estejam desordenadas (ou em ordem aleatória). Isso é garantido pelos espaços em branco que são reservados nos nodos.

Um caso quasi-real: SYBASE

O SYBASE usa árvores B para indexar as tabelas. Existem 2 tipos de árvores B: as granuladas e as não granuladas (clustered). As granuladas são árvores esparsas (nem todo registro de dados tem entrada nos índices) nas quais os registros (na tabela original) são mantidos em ordem dos valores de índice e apenas o primeiro registro em cada página de dados tem uma entrada de índice. Árvores não granuladas são densas e cada um dos registros da tabela tem 1 entrada de índice.

Uma tabela SYBASE com índice granulado exige que seus elementos sejam mantidos em ordem dentro da tabela. Assim, inclusões podem se dar em qualquer página, a depender do conteúdo do campo de índice. Tabelas SYBASE que tem apenas índices não granulados fazem as inclusões de novos elementos sempre ao final das tabelas (na última página).

Tabelas SYBASE podem ter apenas 1 índice granulado, mas podem ter até 250 índices não granulados.

Na criação de um índice granulado, os dados da tabela devem estar colocados nas páginas em ordem.

Cada entrada de índice em sybase pode ser composta por até 16 campos e ter até 256 bytes. Estes campos não precisam estar contiguos. Só não podem ser partes de campos e resultados de cálculos.

O tamanho da página varia, mas o tamanho mais comum é 2K. Existem 5 tipos de páginas em sybase, a saber:

Páginas de Dados contém registros de dados ou registros de log. Ambos são estruturalmente diferentes, mas são construídos do mesmo jeito.

Páginas de índices contém registros de índice

Páginas de texto ou imagem contém BLOBS

Páginas de alocação contém estruturas de dados usadas para gerenciar o processo de alocacão de páginas

páginas de Estatísticas contém estatística de distribuição e uso para os índices.

Página de Dados Se os registros tem tamanho fixo, não haverá tabela de deslocamento na página e o endereço de cada registro é obtido mediante simples cálculo. Neste caso, o tamanho do registro (fixo) é guardado no campo TAMANHO MÍNIMO DE REGISTRO do header.

O header da página é um conjunto de 32 bytes contendo (entre outros): número lógico da página, anterior e próxima página lógica, identificação do objeto ao qual esta página pertence, primeira entrada de registro disponível nesta página, deslocamento do espaço livre, tamanho mínimo do registro

Depois vem a área de registros, que contém um número inteiro de slots para guardar registros. Um registro não atravessa limites de páginas

Tabela de deslocamento: Um conjunto de deslocamentos desde o início da página dando o início de cada um dos registros. Este conjunto cresce do fim da página para o início e só existe quando o tamanho do registro é variável.

Cada registro na página recebe um número de registro. É um campo de 1 byte e assim restringe-se o número de registros em uma página a 256. O número do registro usado nos índices não granulados (densos) é uma combinação do número da página com o número do registro na página.

O registro pode ocupar a página inteira (2048-32=2016). Não há tamanho mínimo para o registro, mas só pode haver 256 registros na página.

Os índices SYBASE são organizados na forma de arvores B + (o + significa que há uma ligação adicional entre as folhas, para permitir o processamento sequencial dos dados).

No exemplo aqui estudado, vão-se fazer algumas considerações/simplificações descritas a seguir.

Suponha um arquivo de pessoas contendo nome, cor preferida, bicho de estimação, alimento que gosta e idade.

Construindo um arquivo SYBASE contendo estes dados e que esteja sujeito às seguintes características:

- Registros de tamanho fixo igual a 40 bytes (9+9+9+9+4).
- Um índice granulado ou esparso (clustered) por um dos campos. No exemplo que é igual para todos, o índice esparso é por cor.
- Quatro índices não granulados (densos) pelos demais atributos.
- Páginas de dados com espaço para 4 registros e fator de preenchimento de 4:1 (usar apenas 3 espaços na carga original).
- Páginas de índices com espaço para 5 chaves (nome, cor, bicho, alimento ou idade). Lembrar que este número é impar, o que indica que no *split* vai subir a terceira chave (das 5 que existem).
- A identificação de cada registro deve ser página:número do registro na página. Por exemplo, a identificação 3:2, indica o segundo registro da terceira página.
- Nas páginas de índice (árvores-B) despreze as informações de endereço dos filhos. No exemplo, elas são irrelevantes. Faça como no exemplo.
- Os dados de entrada estão ordenados pela coluna do índice granulado.

Exemplo: Seja o conjunto de dados, no qual se fará o granulado por cor:

NOME	COR	ANIMAL	COMIDA	id.
josé	dourado	chopim	feijão	34
gabi	laranja	perú	aipim	22
sueli	lilas	gato	alho	40
antônio	musgo	vaca	batata	41
tatiana	petróleo	cavalo	arroz	37
juca	preto	águia	goiaba	36
romeu	roxo	dragão	maracujá	27
paulo	verde	urubú	rabanete	20

Com este conjunto de dados, ter-se-á a seguintes disposição nos arquivos

Páginas de Dados					
Pág1	josé	dourado	chopim	feijão	34
	gabi	laranja	perú	aipim	22
	sueli	lilas	gato	alho	40
Pág2	antônio	musgo	vaca	batata	41
	tatiana	petróleo	cavalo	arroz	37
	juca	preto	águia	goiaba	36
Pág3	romeu	roxo	dragão	maracujá	27
	paulo	verde	urubú	rabanete	20

A seguir, as páginas de índices

Páginas de índices	
Índice I1	granulado por COR
Página 1 ←raiz	dourado(1:1), musgo(2:1), roxo(3:1)
Índice I2	por NOME
Página 1	antônio(2:1), gabi(1:2)
Página 2 ←raiz	jose(1:1)
Página 3	juca(2:3), paulo(3:2), romeu(3:1), sueli(1:3), tatiana(2:2)
Índice I3	por BICHO
P1	águia(2:3), cavalo(2:2), chopim(1:1), dragão(3:1)
P2 ←raiz	gato(1:3)
P3	perú(1:2),urubú(3:2), vaca(2:1)
Índice I4	por COMIDA
P1	aipim(1:2),alho(1:3)
P2 ←raiz	arroz(2:2)
P3	batata(2:1), feijão(1:1), goiaba(2:3), maracujá(3:1), rabanete(3:2)
Índice I5	por IDADE
P1	20(3:2), 22(1:2), 27(3:1), 34(1:1), 36(2:3)
P2 ←raiz	37(2:2)
P3	40(1:3), 41(2:1)

☞ Para você fazer

Desenhe a árvore B a seguir:

```
     1 2    3    4    5    6    7    8    9 10 11 12 13 14 15 16 17
   +-----------------------------------------------------------------
 1| 1 3    32   33   34    0    0    0    0
 2| 0 4    36   43   49   62    0    0    0  1 23   9 16   6
 3| 1 5    84   85   86   88   90    0    0
 4| 1 7   145  146  149  150  156  160  162
 5| 1 6   122  125  128  129  130  132    0
 6| 1 4    64   65   66   67    0    0    0
 7| 1 5   179  184  185  188  190    0    0
 8| 1 3   112  115  116    0    0    0    0
 9| 1 4    45   46   47   48    0    0    0
*10| 0 3    71  111  163    0    0    0    0  2 19 11 21
 11| 0 3   118  133  144    0    0    0    0  8  5 13  4
 12| 1 7    99  101  102  103  106  108  109
 13| 1 6   135  137  138  139  140  143    0
 14| 1 4   165  166  167  168    0    0    0
 15| 1 5    72   74   75   76   77    0    0
 16| 1 7    51   52   54   55   58   59   60
 17| 1 4   193  196  199  200    0    0    0
 18| 1 3    80   81   82    0    0    0    0
 19| 0 4    79   83   91   98    0    0    0 15 18   3 22 12
 20| 1 3   171  175  176    0    0    0    0
 21| 0 3   169  178  191    0    0    0    0 14 20   7 17
 22| 1 3    93   95   96    0    0    0    0
 23| 1 4    38   40   41   42    0    0    0
```

Responda agora:

Qual a altura da árvore B ?	Se o 31 for incluido, qual linha,coluna na matriz?	Se o 189 for incluído, qual linha,coluna na matriz?

Mais uma Instância do mesmo exercício Desenhe a árvore B a seguir:

```
     1 2    3    4    5    6    7    8    9 10 11 12 13 14 15 16 17
   +-----------------------------------------------------------------
 1| 1 4    31   34   36   38    0    0    0
 2| 0 4    40   50   60   67    0    0    0  1 18   7 20 13
 3| 1 4   137  138  139  140    0    0    0
 4| 1 7    79   80   81   83   85   86   87
 5| 1 3   159  160  161    0    0    0    0
 6| 1 5    97   98  100  101  103    0    0
 7| 1 3    52   54   57    0    0    0    0
 8| 1 7   169  171  172  173  174  177  178
 9| 1 4   124  126  127  128    0    0    0
*10| 0 3    78  122  158    0    0    0    0  2 19 11 22
 11| 0 3   129  136  141    0    0    0    0  9 17   3 16
 12| 1 5   181  183  188  189  190    0    0
 13| 1 6    68   71   72   73   74   76    0
 14| 1 6   109  110  111  116  117  119    0
 15| 1 4    90   92   93   94    0    0    0
 16| 1 3   146  152  156    0    0    0    0
 17| 1 4   130  132  133  135    0    0    0
 18| 1 6    41   42   44   46   48   49    0
 19| 0 3    89   95  107    0    0    0    0  4 15   6 14
 20| 1 4    62   63   64   66    0    0    0
 21| 1 4   192  194  195  199    0    0    0
 22| 0 4   162  168  179  191    0    0    0  5 23   8 12 21
 23| 1 4   164  165  166  167    0    0    0
```

Responda agora:

Qual a altura da árvore B ?	Se o 58 for incluido, qual linha,coluna na matriz?	Se o 144 for incluído, qual linha,coluna na matriz?

CEP-UFPR-UTFPR-PUC/Pr-UP Sistemas de Informação 12/08/2024 - 11:35:52.4
Matemática aplicada Prof Dr P Kantek (pkantek@gmail.com)
Evolução de árvores: Árvore-B (e B+) VIVXq92a V: 1.02
76575 LAURA PACHECO DELATTRE
24CC2401 - 9 primeiro exercicio do T3 _____ / / _____

Árvore B

O conceito de árvore B foi desenvolvido por Rudolf Bayer e Edward McCreight em 1972.

Eles publicaram seu trabalho no artigo *"Organization and Maintenance of Large Ordered Indices"*, que descreve as propriedades e operações básicas das árvores B.

O trabalho de Bayer e McCreight teve um impacto significativo na área de ciência da computação, e as árvores B são agora amplamente utilizadas em bancos de dados e outros sistemas de software.

As árvores B são um tipo de árvore de busca balanceada que pode armazenar um grande número de chaves e valores de maneira eficiente. Elas são mais eficientes do que as árvores de busca binárias tradicionais, pois podem ter mais filhos por nó. As árvores B são usadas em uma variedade de aplicações, incluindo bancos de dados, sistemas de arquivos e sistemas de roteamento.

Eles não explicaram o significado do nome, só nos resta especular que esse B seja de Balanceada. Se você se lembrar das árvores binárias de pesquisa, deve recordar o problema que existe quando as chaves vem (quasi) ordenadas na entrada: a árvore gerada é desbalanceada e pode chegar a ter um desempenho terrível.

Outra qualidade da árvore-B é que ela tem grau k (k filhos por nodo). Em geral este valor é escolhido de modo a otimizar a blocagem de dados no arquivo de memória secundária. De qualquer modo este k modifica o desempenho, que lembrando é proporcional a $\log_k n$, onde n é a quantidade de chaves.

Como toda árvore, ela tem uma raiz que tem de 0 a $2k-1$ chaves e consequentemente tem de 0 a $2k$ filhos. Cada um dos filhos, tem de $k-1$ até $2k-1$ chaves de 0 até $2k$ filhos. Note que a raiz é o único nodo que sofre processamento especial, necessário para quando o arquivo tem uma única chave (por exemplo).

Um caso prático real Suponha um colégio com milhares de alunos, que têm seus dados guardados em um banco de dados. Por hipótese, os dados de cada aluno vão sendo recolhidos quando eles fazem a matrícula, pelo que a ordem física dos dados no arquivo é a data da matrícula. Parece óbvio assinalar que um determinado conjunto de dados (um arquivo) só pode ter uma única ordem física em um determinado instante. Mas nada impede que existam muitas ordens lógicas diferentes para o mesmo arquivo. Tais ordens lógicas são providenciadas por estruturas muito parecidas a uma árvore-b. Neste exemplo, poderíamos ter índices por nome (ordem alfabética), curso, turma, endereço etc.

Vai-se descrever aqui uma árvore-b que dará acesso aos dados por código do aluno.

Suponha uma árvore-b de nome M que providenciará este acesso. Vamos descrevê-la com $k=4$, que é muito pouco para aplicações reais, mas fica razoável para trabalhar com este exemplo. De acordo com a definição, os nodos de M têm entre 3 e 7 códigos de aluno. Para esta dimensão, o nodo de M terá 17 colunas:

coluna 0 um asterisco nesta posição indica que este é o nodo raiz da árvore. Conta como coluna 0 porque na verdade esta informação não está fisicamente aqui, está em outro lugar.

coluna 1 indicativo de folha (1=sim; 0=não)

coluna 2 quantidade de chaves válidas neste nodo (nem sempre o valor inválido é zero)

cols 3 a 9 chaves, sempre em ordem crescente

cols 10 a 17 endereços dos filhos associados a cada chave. Só estão preenchidas para nodos não folha.

Veja um exemplo:

```
    1 2    3    4    5    6    7    8    9 10 11 12 13 14 15 16 17
  +-------------------------------------------------------------------
 1| 1 5    3    6    7    8   10    0    0
 2| 0 4   11   20   32   44    0    0    0  1 20 15  7 12
 3| 1 4  128  131  132  134    0    0    0
 4| 1 5  149  150  151  153  160    0    0
 5| 1 6   65   73   76   79   80   83    0
 6| 1 5  182  183  188  189  191    0    0
 7| 1 3   34   35   37    0    0    0    0
 8| 1 4  106  109  111  114    0    0    0
 9| 1 6  136  137  138  140  143  145    0
*10| 0 2   64  127    0    0    0    0    0  2 17 11
 11| 0 6  135  146  162  171  181  192    0  3  9  4 14 21  6 18
 12| 1 7   46   49   50   55   56   57   63
 13| 1 4   88   89   91   94    0    0    0
 14| 1 5  164  166  167  168  169    0    0
 15| 1 6   21   22   23   28   29   30    0
 16| 1 7  117  118  120  122  123  125  126
 17| 0 4   84   95  105  116    0    0    0  5 13 19  8 16
 18| 1 5  193  195  197  199  200    0    0
 19| 1 5   96   98  100  102  103    0    0
 20| 1 4   14   15   17   19    0    0    0
 21| 1 3  174  175  177    0    0    0    0
```

Antes de continuar no exercício, você deve tentar entender o funcionamento da árvore-b.

- Desenhe a árvore
- identifique a necessidade da coluna 1. Idem para a coluna 2
- Qual a altura da árvore ?
- Quando e como a altura da árvore cresce ?

Agoram suponha que o elemento 187 é incluído. Qual a linha;coluna da matriz acima que é modificada, ou seja onde o elemento recém incluido entra ? Localize-o e veja:

```
    1 2    3    4    5    6    7    8    9 10 11 12 13 14 15 16 17
  +-------------------------------------------------------------------
 1| 1 5    3    6    7    8   10    0    0
 2| 0 4   11   20   32   44    0    0    0  1 20 15  7 12
 3| 1 4  128  131  132  134    0    0    0
 4| 1 5  149  150  151  153  160    0    0
 5| 1 6   65   73   76   79   80   83    0
 6| 1 6  182  183(187)188  189  191    0
 7| 1 3   34   35   37    0    0    0    0
 8| 1 4  106  109  111  114    0    0    0
 9| 1 6  136  137  138  140  143  145    0
*10| 0 2   64  127    0    0    0    0    0  2 17 11
 11| 0 6  135  146  162  171  181  192    0  3  9  4 14 21  6 18
 12| 1 7   46   49   50   55   56   57   63
 13| 1 4   88   89   91   94    0    0    0
 14| 1 5  164  166  167  168  169    0    0
 15| 1 6   21   22   23   28   29   30    0
 16| 1 7  117  118  120  122  123  125  126
 17| 0 4   84   95  105  116    0    0    0  5 13 19  8 16
 18| 1 5  193  195  197  199  200    0    0
 19| 1 5   96   98  100  102  103    0    0
 20| 1 4   14   15   17   19    0    0    0
 21| 1 3  174  175  177    0    0    0    0
```

Como voce pode ver ele está na linha ____ e coluna ____. Algumas características notáveis da árvore-b

- Figurinha carimbada no mundo do software: usam B-trees: SYBASE e do Harbour (antigo Clipper). Também são usados no Oracle, MySQL, PostgreSQL, Microsoft SQL e DB2, da IBM.
- muitas estruturas usam a árvore-b+. O + significa que há uma ligação adicional entre as folhas, para permitir o processamento sequencial dos dados.
- Existem nodos folha e não-folha. Os endereços dos filhos (colunas 10 a 17) só existem para os nodos que os têm (nodos não folhas).
- Os códigos de aluno (colunas 3 a 9 não mostram, para diminuir o ruído), mas ao lado de cada código está o endereço físico real dos dados deste aluno, que afinal é o dado que nos interessa nesta busca.
- Os códigos (colunas 3 a 9) estão em ordem crescente. Isto é fundamental e não pode ser modificado.
- Perceba como, em média, apenas a metade dos códigos está preenchida. Isto é necessário para permitir inclusão de novos dados. Sinaliza também um certo desperdício (overhead) de área em disco que qualquer método de acesso consome.
- Quando um nodo tem todas as chaves preenchidas e chega uma nova, que deveria (deverá) entrar nele, o nodo sofre um processo chamado split e ele é transformado em 2 nodos (cada um com metade dos espaços vazios). Note que o número máximo de chaves em qualquer nodo é sempre ímpar ($2n-1$) o que implica em que o elemento do meio sobe para o *pai* do nodo que está sendo quebrado. A metade anterior vai para um nodo *splitado* e a metade posterior vai para o outro nodo *splitado*.
- Quando as chaves vão sendo excluídas, e um nodo passa a ter menos do que a metade das chaves (o que é proibido) ele é juntado com um nodo irmão e ambos passam a ser um único nodo. Este processo recebe o nome de merge. Não é tão simples assim, há diversos casos particulares, mas a ideia é essa.

Mais um exemplo. Suponha a inclusão do número 26. Em que linha;coluna ele entra ?

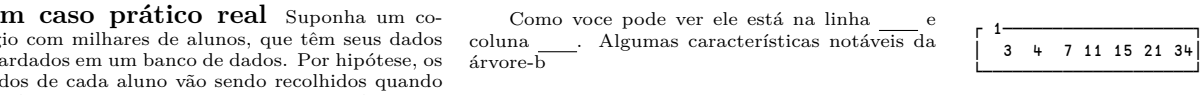
Algoritmo de Inclusão em árvore B

```
1: função INCLUA-BTREE (K:inteiro)
2: X ← raiz da árvore B
3: enquanto nodo X não é folha
4:     X ← filho correto onde K deve ser incluído
5: fim{enquanto}
6: se nodo X está cheio
7:     faça split do nodo X
8:     INCLUA-BTREE (K)
9:     abandone
10: senão
11:     inclua K no nodo X mantendo as chaves ordenadas
12: fim{se}
```

Algoritmos Estão disseminados pela comunidade. Deixo de apresentá-los aqui, pois são bastante específicos, complexos até, e mais importante do que implementá-los é entender-lhes o funcionamento. Até porque se você tiver que usar uma árvore b que não seja para aprender como ela funciona, recomendo fortemente que você use pacotes prontos, já testados e confiáveis.

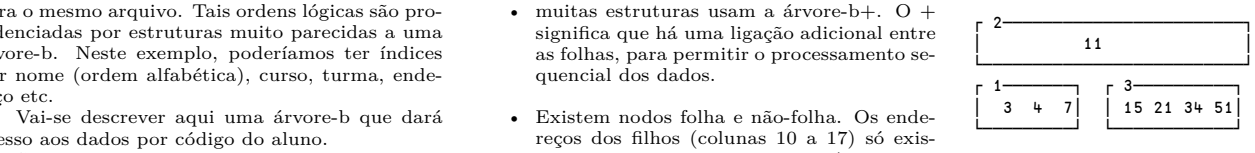
Um exemplo, passo a passo Seja uma árvore-b com $k=4$. Pela definição os nodos terão entre 3 e 7 chaves.

```
criabtree
insira 4, 7, 21, 11, 34, 15, 3
```

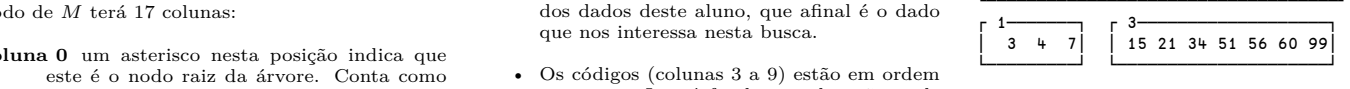


Agora, vamos inserir uma nova chave, o que vai provocar o split

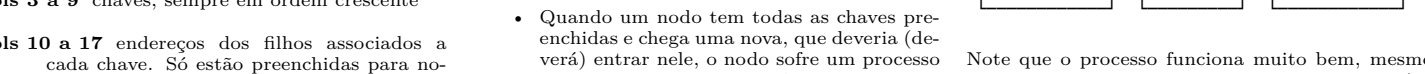
```
insira 51
```



Inserindo 60, 99, 56, fica



Inserindo 2 e 70, o 2 vai na boa para o nodo 1 enquanto o 70 força o split do nodo 3 e fica



Note que o processo funciona muito bem, mesmo que as chaves de entrada estejam desordenadas (ou em ordem aleatória). Isso é garantido pelos espaços em branco que são reservados nos nodos.

Um caso quasi-real: SYBASE

O SYBASE usa árvores B para indexar as tabelas. Existem 2 tipos de árvores B: as granuladas e as não granuladas (clustered). As granuladas são árvores esparsas (nem todo registro de dados tem entrada nos índices) nas quais os registros (na tabela original) são mantidos em ordem dos valores de índice e apenas o primeiro registro em cada página de dados tem uma entrada de índice. Árvores não granuladas são densas e cada um dos registros da tabela tem 1 entrada de índice.

Uma tabela SYBASE com índice granulado exige que seus elementos sejam mantidos em ordem dentro da tabela. Assim, inclusões podem se dar em qualquer página, a depender do conteúdo do campo de índice. Tabelas SYBASE que tem apenas índices não granulados fazem as inclusões de novos elementos sempre ao final das tabelas (na última página).

Tabelas SYBASE podem ter apenas 1 índice granulado, mas podem ter até 250 índices não granulados.

Na criação de um índice granulado, os dados da tabela devem estar colocados nas páginas em ordem.

Cada entrada de índice em sybase pode ser composta por até 16 campos e ter até 256 bytes. Estes campos não precisam estar contiguos. Só não podem ser partes de campos e resultados de cálculos.

O tamanho da página varia, mas o tamanho mais comum é 2K. Existem 5 tipos de páginas em sybase, a saber:

Páginas de Dados contém registros de dados ou registros de log. Ambos são estruturalmente diferentes, mas são construídos do mesmo jeito.

Páginas de índices contém registros de índice

Páginas de texto ou imagem contém BLOBS

Páginas de alocação contém estruturas de dados usadas para gerenciar o processo de alocacão de páginas

páginas de Estatísticas contém estatística de distribuição e uso para os índices.

Página de Dados Se os registros tem tamanho fixo, não haverá tabela de deslocamento na página e o endereço de cada registro é obtido mediante simples cálculo. Neste caso, o tamanho do registro (fixo) é guardado no campo TAMANHO MÍNIMO DE REGISTRO do header.

O header da página é um conjunto de 32 bytes contendo (entre outros): número lógico da página, anterior e próxima página lógica, identificação do objeto ao qual esta página pertence, primeira entrada de registro disponível nesta página, deslocamento do espaço livre, tamanho mínimo do registro

Depois vem a área de registros, que contém um número inteiro de slots para guardar registros. Um registro não atravessa limites de páginas

Tabela de deslocamento: Um conjunto de deslocamentos desde o início da página dando o início de cada um dos registros. Este conjunto cresce do fim da página para o início e só existe quando o tamanho do registro é variável.

Cada registro na página recebe um número de registro. É um campo de 1 byte e assim restringe-se o número de registros em uma página a 256. O número do registro usado nos índices não granulados (densos) é uma combinação do número da página com o número do registro na página.

O registro pode ocupar a página inteira (2048-32=2016). Não há tamanho mínimo para o registro, mas só pode haver 256 registros na página.

Os índices SYBASE são organizados na forma de arvores B + (o + significa que há uma ligação adicional entre as folhas, para permitir o processamento sequencial dos dados).

No exemplo aqui estudado, vão-se fazer algumas considerações/simplificações descritas a seguir.

Suponha um arquivo de pessoas contendo nome, cor preferida, bicho de estimação, alimento que gosta e idade.

Construindo um arquivo SYBASE contendo estes dados e que esteja sujeito às seguintes características:

- Registros de tamanho fixo igual a 40 bytes (9+9+9+9+4).
- Um índice granulado ou esparso (clustered) por um dos campos. No exemplo que é igual para todos, o índice esparso é por cor.
- Quatro índices não granulados (densos) pelos demais atributos.
- Páginas de dados com espaço para 4 registros e fator de preenchimento de 4:1 (usar apenas 3 espaços na carga original).
- Páginas de índices com espaço para 5 chaves (nome, cor, bicho, alimento ou idade). Lembrar que este número é impar, o que indica que no *split* vai subir a terceira chave (das 5 que existem).
- A identificação de cada registro deve ser página:número do registro na página. Por exemplo, a identificação 3:2, indica o segundo registro da terceira página.
- Nas páginas de índice (árvores-B) despreze as informações de endereço dos filhos. No exemplo, elas são irrelevantes. Faça como no exemplo.
- Os dados de entrada estão ordenados pela coluna do índice granulado.

Exemplo: Seja o conjunto de dados, no qual se fará o granulado por cor:

NOME	COR	ANIMAL	COMIDA	id.
josé	dourado	chopim	feijão	34
gabi	laranja	perú	aipim	22
sueli	lilas	gato	alho	40
antônio	musgo	vaca	batata	41
tatiana	petróleo	cavalo	arroz	37
juca	preto	águia	goiaba	36
romeu	roxo	dragão	maracujá	27
paulo	verde	urubú	rabanete	20

Com este conjunto de dados, ter-se-á a seguintes disposição nos arquivos

Páginas de Dados					
Pág1	josé	dourado	chopim	feijão	34
	gabi	laranja	perú	aipim	22
	sueli	lilas	gato	alho	40
Pág2	antônio	musgo	vaca	batata	41
	tatiana	petróleo	cavalo	arroz	37
	juca	preto	águia	goiaba	36
Pág3	romeu	roxo	dragão	maracujá	27
	paulo	verde	urubú	rabanete	20

A seguir, as páginas de índices

Páginas de índices	
Índice I1	granulado por COR
Página 1 ←raiz	dourado(1:1), musgo(2:1), roxo(3:1)
Índice I2	por NOME
Página 1	antônio(2:1), gabi(1:2)
Página 2 ←raiz	jose(1:1)
Página 3	juca(2:3), paulo(3:2), romeu(3:1), sueli(1:3), tatiana(2:2)
Índice I3	por BICHO
P1	águia(2:3), cavalo(2:2), chopim(1:1), dragão(3:1)
P2 ←raiz	gato(1:3)
P3	perú(1:2),urubú(3:2), vaca(2:1)
Índice I4	por COMIDA
P1	aipim(1:2),alho(1:3)
P2 ←raiz	arroz(2:2)
P3	batata(2:1), feijão(1:1), goiaba(2:3), maracujá(3:1), rabanete(3:2)
Índice I5	por IDADE
P1	20(3:2), 22(1:2), 27(3:1), 34(1:1), 36(2:3)
P2 ←raiz	37(2:2)
P3	40(1:3), 41(2:1)

☞ Para você fazer

Desenhe a árvore B a seguir:

```
     1 2    3    4    5    6    7    8    9 10 11 12 13 14 15 16 17
   +-------------------------------------------------------------------
 1| 1 7    31   32   33   34   35   36   37
 2| 0 4    39   46   58   65    0    0    0  1 12  8 13 19
 3| 1 4   133  134  135  136    0    0    0
 4| 1 7    75   76   78   82   83   84   88
 5| 1 5   174  175  176  177  178    0    0
 6| 1 4   119  122  123  124    0    0    0
 7| 1 4   166  168  169  172    0    0    0
 8| 1 4    47   50   55   57    0    0    0
 9| 1 6   194  196  197  198  199  200    0
*10| 0 2    71  132    0    0    0    0    0  2 18 11
 11| 0 5   139  161  173  179  191    0    0  3 15  7  5 16  9
 12| 1 4    41   43   44   45    0    0    0
 13| 1 4    59   61   62   64    0    0    0
 14| 1 4    90   92   93   95    0    0    0
 15| 1 7   140  141  143  144  149  151  155
 16| 1 6   181  182  183  186  189  190    0
 17| 1 7   104  106  107  108  111  112  114
 18| 0 5    89   96  102  117  125    0    0  4 14 20 17  6 21
 19| 1 4    66   67   68   70    0    0    0
 20| 1 4    97   98   99  100    0    0    0
 21| 1 3   126  127  130    0    0    0    0
```

Responda agora:

Qual a altura da árvore B ?	Se o 48 for incluido, qual linha,coluna na matriz?	Se o 154 for incluído, qual linha,coluna na matriz?

Mais uma Instância do mesmo exercício Desenhe a árvore B a seguir:

```
     1 2    3    4    5    6    7    8    9 10 11 12 13 14 15 16 17
   +-------------------------------------------------------------------
 1| 1 5    31   32   35   36   38    0    0
 2| 0 4    39   49   60   68    0    0    0  1 16 13  5 20
 3| 1 5   137  138  139  142  144    0    0
 4| 1 5    90   92   93   94   96    0    0
 5| 1 5    61   63   65   66   67    0    0
 6| 1 3   163  167  168    0    0    0    0
 7| 1 6   129  130  131  132  133  134    0
 8| 1 5   184  187  188  192  193    0    0
 9| 1 7    79   81   82   84   85   86   87
*10| 0 3    78  128  162    0    0    0    0  2 18 11 22
 11| 0 3   136  146  153    0    0    0    0  7  3 14 21
 12| 1 4   109  110  111  112    0    0    0
 13| 1 6    52   53   55   56   58   59    0
 14| 1 3   147  148  152    0    0    0    0
 15| 1 5   196  197  198  199  200    0    0
 16| 1 3    42   43   47    0    0    0    0
 17| 1 5   117  120  123  125  126    0    0
 18| 0 4    88   98  107  116    0    0    0  9  4 23 12 17
 19| 1 5   175  176  177  178  180    0    0
 20| 1 3    69   71   77    0    0    0    0
 21| 1 5   154  155  156  157  159    0    0
 22| 0 3   170  183  194    0    0    0    0  6 19  8 15
 23| 1 3   102  103  104    0    0    0    0
```

Responda agora:

Qual a altura da árvore B ?	Se o 151 for incluido, qual linha,coluna na matriz?	Se o 101 for incluído, qual linha,coluna na matriz?

CEP-UFPR-UTFPR-PUC/Pr-UP Sistemas de Informação 12/08/2024 - 11:35:52.4
Matemática aplicada Prof Dr P Kantek (pkantek@gmail.com)
Evolução de árvores: Árvore-B (e B+) VIVXq92a V: 1.02
76694 LAURO HENRIK RUTZ
24CC2401 - 10 primeiro exercicio do T3 _____ / / _____

Árvore B

O conceito de árvore B foi desenvolvido por Rudolf Bayer e Edward McCreight em 1972.

Eles publicaram seu trabalho no artigo *"Organization and Maintenance of Large Ordered Indices"*, que descreve as propriedades e operações básicas das árvores B.

O trabalho de Bayer e McCreight teve um impacto significativo na área de ciência da computação, e as árvores B são agora amplamente utilizadas em bancos de dados e outros sistemas de software.

As árvores B são um tipo de árvore de busca balanceada que pode armazenar um grande número de chaves e valores de maneira eficiente. Elas são mais eficientes do que as árvores de busca binárias tradicionais, pois podem ter mais filhos por nó. As árvores B são usadas em uma variedade de aplicações, incluindo bancos de dados, sistemas de arquivos e sistemas de roteamento.

Eles não explicaram o significado do nome, só nos resta especular que esse B seja de Balanceada. Se você se lembrar das árvores binárias de pesquisa, deve recordar o problema que existe quando as chaves vem (quasi) ordenadas na entrada: a árvore gerada é desbalanceada e pode chegar a ter um desempenho terrível.

Outra qualidade da árvore-B é que ela tem grau k (k filhos por nodo). Em geral este valor é escolhido de modo a otimizar a blocagem de dados no arquivo de memória secundária. De qualquer modo este k modifica o desempenho, que lembrando é proporcional a $\log_k n$, onde n é a quantidade de chaves.

Como toda árvore, ela tem uma raiz que tem de 0 a $2k-1$ chaves e consequentemente tem de 0 a $2k$ filhos. Cada um dos filhos, tem de $k-1$ até $2k-1$ chaves de 0 até $2k$ filhos. Note que a raiz é o único nodo que sofre processamento especial, necessário para quando o arquivo tem uma única chave (por exemplo).

Um caso prático real Suponha um colégio com milhares de alunos, que têm seus dados guardados em um banco de dados. Por hipótese, os dados de cada aluno vão sendo recolhidos quando eles fazem a matrícula, pelo que a ordem física dos dados no arquivo é a data da matrícula. Parece óbvio assinalar que um determinado conjunto de dados (um arquivo) só pode ter uma única ordem física em um determinado instante. Mas nada impede que existam muitas ordens lógicas diferentes para o mesmo arquivo. Tais ordens lógicas são providenciadas por estruturas muito parecidas a uma árvore-b. Neste exemplo, poderíamos ter índices por nome (ordem alfabética), curso, turma, endereço etc.

Vai-se descrever aqui uma árvore-b que dará acesso aos dados por código do aluno.

Suponha uma árvore-b de nome M que providenciará este acesso. Vamos descrevê-la com $k=4$, que é muito pouco para aplicações reais, mas fica razoável para trabalhar com este exemplo. De acordo com a definição, os nodos de M têm entre 3 e 7 códigos de aluno. Para esta dimensão, o nodo de M terá 17 colunas:

coluna 0 um asterisco nesta posição indica que este é o nodo raiz da árvore. Conta como coluna 0 porque na verdade esta informação não está fisicamente aqui, está em outro lugar.

coluna 1 indicativo de folha (1=sim; 0=não)

coluna 2 quantidade de chaves válidas neste nodo (nem sempre o valor inválido é zero)

cols 3 a 9 chaves, sempre em ordem crescente

cols 10 a 17 endereços dos filhos associados a cada chave. Só estão preenchidas para nodos não folha.

Veja um exemplo:

```
    1 2    3    4    5    6    7    8    9 10 11 12 13 14 15 16 17
  +-------------------------------------------------------------------
 1| 1 5    3    6    7    8   10    0    0
 2| 0 4   11   20   32   44    0    0    0  1 20 15  7 12
 3| 1 4  128  131  132  134    0    0    0
 4| 1 5  149  150  151  153  160    0    0
 5| 1 6   65   73   76   79   80   83    0
 6| 1 5  182  183  188  189  191    0    0
 7| 1 3   34   35   37    0    0    0    0
 8| 1 4  106  109  111  114    0    0    0
 9| 1 6  136  137  138  140  143  145    0
*10| 0 2   64  127    0    0    0    0    0  2 17 11
 11| 0 6  135  146  162  171  181  192    0  3  9  4 14 21  6 18
 12| 1 7   46   49   50   55   56   57   63
 13| 1 4   88   89   91   94    0    0    0
 14| 1 5  164  166  167  168  169    0    0
 15| 1 6   21   22   23   28   29   30    0
 16| 1 7  117  118  120  122  123  125  126
 17| 0 4   84   95  105  116    0    0    0  5 13 19  8 16
 18| 1 5  193  195  197  199  200    0    0
 19| 1 5   96   98  100  102  103    0    0
 20| 1 4   14   15   17   19    0    0    0
 21| 1 3  174  175  177    0    0    0    0
```

Antes de continuar no exercício, você deve tentar entender o funcionamento da árvore-b.

- Desenhe a árvore
- identifique a necessidade da coluna 1. Idem para a coluna 2
- Qual a altura da árvore ?
- Quando e como a altura da árvore cresce ?

Agoram suponha que o elemento 187 é incluído. Qual a linha;coluna da matriz acima que é modificada, ou seja onde o elemento recém incluido entra ? Localize-o e veja:

```
    1 2    3    4    5    6    7    8    9 10 11 12 13 14 15 16 17
  +-------------------------------------------------------------------
 1| 1 5    3    6    7    8   10    0    0
 2| 0 4   11   20   32   44    0    0    0  1 20 15  7 12
 3| 1 4  128  131  132  134    0    0    0
 4| 1 5  149  150  151  153  160    0    0
 5| 1 6   65   73   76   79   80   83    0
 6| 1 6  182  183(187)188  189  191    0
 7| 1 3   34   35   37    0    0    0    0
 8| 1 4  106  109  111  114    0    0    0
 9| 1 6  136  137  138  140  143  145    0
*10| 0 2   64  127    0    0    0    0    0  2 17 11
 11| 0 6  135  146  162  171  181  192    0  3  9  4 14 21  6 18
 12| 1 7   46   49   50   55   56   57   63
 13| 1 4   88   89   91   94    0    0    0
 14| 1 5  164  166  167  168  169    0    0
 15| 1 6   21   22   23   28   29   30    0
 16| 1 7  117  118  120  122  123  125  126
 17| 0 4   84   95  105  116    0    0    0  5 13 19  8 16
 18| 1 5  193  195  197  199  200    0    0
 19| 1 5   96   98  100  102  103    0    0
 20| 1 4   14   15   17   19    0    0    0
 21| 1 3  174  175  177    0    0    0    0
```

Como voce pode ver ele está na linha ____ e coluna ____. Algumas características notáveis da árvore-b

- Figurinha carimbada no mundo do software: usam B-trees: SYBASE e do Harbour (antigo Clipper). Também são usados no Oracle, MySQL, PostgreSQL, Microsoft SQL e DB2, da IBM.
- muitas estruturas usam a árvore-b+. O + significa que há uma ligação adicional entre as folhas, para permitir o processamento sequencial dos dados.
- Existem nodos folha e não-folha. Os endereços dos filhos (colunas 10 a 17) só existem para os nodos que os têm (nodos não folhas).
- Os códigos de aluno (colunas 3 a 9 não mostram, para diminuir o ruído), mas ao lado de cada código está o endereço físico real dos dados deste aluno, que afinal é o dado que nos interessa nesta busca.
- Os códigos (colunas 3 a 9) estão em ordem crescente. Isto é fundamental e não pode ser modificado.
- Perceba como, em média, apenas a metade dos códigos está preenchida. Isto é necessário para permitir inclusão de novos dados. Sinaliza também um certo desperdício (overhead) de área em disco que qualquer método de acesso consome.
- Quando um nodo tem todas as chaves preenchidas e chega uma nova, que deveria (deverá) entrar nele, o nodo sofre um processo chamado split e ele é transformado em 2 nodos (cada um com metade dos espaços vazios). Note que o número máximo de chaves em qualquer nodo é sempre ímpar ($2n-1$) o que implica em que o elemento do meio sobe para o *pai* do nodo que está sendo quebrado. A metade anterior vai para um nodo *splitado* e a metade posterior vai para o outro nodo *splitado*.
- Quando as chaves vão sendo excluídas, e um nodo passa a ter menos do que a metade das chaves (o que é proibido) ele é juntado com um nodo irmão e ambos passam a ser um único nodo. Este processo recebe o nome de merge. Não é tão simples assim, há diversos casos particulares, mas a ideia é essa.

Mais um exemplo. Suponha a inclusão do número 26. Em que linha;coluna ele entra ?

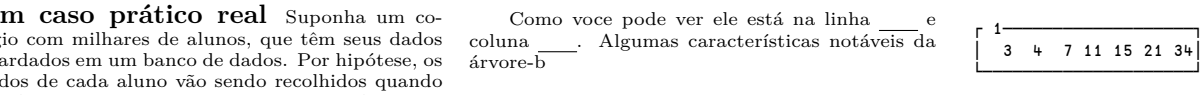
Algoritmo de Inclusão em árvore B

```
1: função INCLUA-BTREE (K:inteiro)
2: X ← raiz da árvore B
3: enquanto nodo X não é folha
4:     X ← filho correto onde K deve ser incluído
5: fim{enquanto}
6: se nodo X está cheio
7:     faça split do nodo X
8:     INCLUA-BTREE (K)
9:     abandone
10: senão
11:     inclua K no nodo X mantendo as chaves ordenadas
12: fim{se}
```

Algoritmos Estão disseminados pela comunidade. Deixo de apresentá-los aqui, pois são bastante específicos, complexos até, e mais importante do que implementá-los é entender-lhes o funcionamento. Até porque se você tiver que usar uma árvore b que não seja para aprender como ela funciona, recomendo fortemente que você use pacotes prontos, já testados e confiáveis.

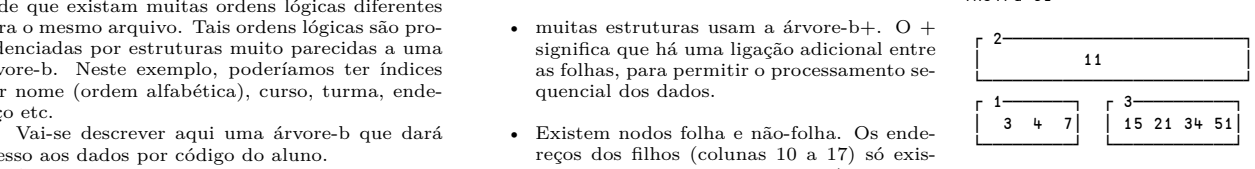
Um exemplo, passo a passo Seja uma árvore-b com $k=4$. Pela definição os nodos terão entre 3 e 7 chaves.

```
criabtree
insira 4, 7, 21, 11, 34, 15, 3
```

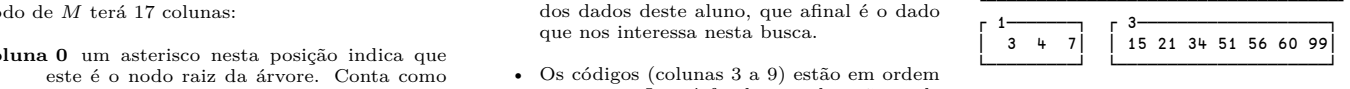


Agora, vamos inserir uma nova chave, o que vai provocar o split

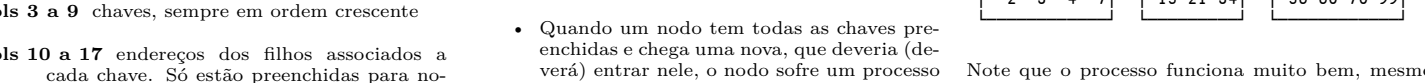
```
insira 51
```



Inserindo 60, 99, 56, fica



Inserindo 2 e 70, o 2 vai na boa para o nodo 1 enquanto o 70 força o split do nodo 3 e fica



Note que o processo funciona muito bem, mesmo que as chaves de entrada estejam desordenadas (ou em ordem aleatória). Isso é garantido pelos espaços em branco que são reservados nos nodos.

Um caso quasi-real: SYBASE

O SYBASE usa árvores B para indexar as tabelas. Existem 2 tipos de árvores B: as granuladas e as não granuladas (clustered). As granuladas são árvores esparsas (nem todo registro de dados tem entrada nos índices) nas quais os registros (na tabela original) são mantidos em ordem dos valores de índice e apenas o primeiro registro em cada página de dados tem uma entrada de índice. Árvores não granuladas são densas e cada um dos registros da tabela tem 1 entrada de índice.

Uma tabela SYBASE com índice granulado exige que seus elementos sejam mantidos em ordem dentro da tabela. Assim, inclusões podem se dar em qualquer página, a depender do conteúdo do campo de índice. Tabelas SYBASE que tem apenas índices não granulados fazem as inclusões de novos elementos sempre ao final das tabelas (na última página).

Tabelas SYBASE podem ter apenas 1 índice granulado, mas podem ter até 250 índices não granulados.

Na criação de um índice granulado, os dados da tabela devem estar colocados nas páginas em ordem.

Cada entrada de índice em sybase pode ser composta por até 16 campos e ter até 256 bytes. Estes campos não precisam estar contiguos. Só não podem ser partes de campos e resultados de cálculos.

O tamanho da página varia, mas o tamanho mais comum é 2K. Existem 5 tipos de páginas em sybase, a saber:

Páginas de Dados contém registros de dados ou registros de log. Ambos são estruturalmente diferentes, mas são construídos do mesmo jeito.

Páginas de índices contém registros de índice

Páginas de texto ou imagem contém BLOBS

Páginas de alocação contém estruturas de dados usadas para gerenciar o processo de alocacão de páginas

páginas de Estatísticas contém estatística de distribuição e uso para os índices.

Página de Dados Se os registros tem tamanho fixo, não haverá tabela de deslocamento na página e o endereço de cada registro é obtido mediante simples cálculo. Neste caso, o tamanho do registro (fixo) é guardado no campo TAMANHO MÍNIMO DE REGISTRO do header.

O header da página é um conjunto de 32 bytes contendo (entre outros): número lógico da página, anterior e próxima página lógica, identificação do objeto ao qual esta página pertence, primeira entrada de registro disponível nesta página, deslocamento do espaço livre, tamanho mínimo do registro

Depois vem a área de registros, que contém um número inteiro de slots para guardar registros. Um registro não atravessa limites de páginas

Tabela de deslocamento: Um conjunto de deslocamentos desde o início da página dando o início de cada um dos registros. Este conjunto cresce do fim da página para o início e só existe quando o tamanho do registro é variável.

Cada registro na página recebe um número de registro. É um campo de 1 byte e assim restringe-se o número de registros em uma página a 256. O número do registro usado nos índices não granulados (densos) é uma combinação do número da página com o número do registro na página.

O registro pode ocupar a página inteira (2048-32=2016). Não há tamanho mínimo para o registro, mas só pode haver 256 registros na página.

Os índices SYBASE são organizados na forma de arvores B + (o + significa que há uma ligação adicional entre as folhas, para permitir o processamento sequencial dos dados).

No exemplo aqui estudado, vão-se fazer algumas considerações/simplificações descritas a seguir.

Suponha um arquivo de pessoas contendo nome, cor preferida, bicho de estimação, alimento que gosta e idade.

Construindo um arquivo SYBASE contendo estes dados e que esteja sujeito às seguintes características:

- Registros de tamanho fixo igual a 40 bytes (9+9+9+9+4).
- Um índice granulado ou esparso (clustered) por um dos campos. No exemplo que é igual para todos, o índice esparso é por cor.
- Quatro índices não granulados (densos) pelos demais atributos.
- Páginas de dados com espaço para 4 registros e fator de preenchimento de 4:1 (usar apenas 3 espaços na carga original).
- Páginas de índices com espaço para 5 chaves (nome, cor, bicho, alimento ou idade). Lembrar que este número é impar, o que indica que no *split* vai subir a terceira chave (das 5 que existem).
- A identificação de cada registro deve ser página:número do registro na página. Por exemplo, a identificação 3:2, indica o segundo registro da terceira página.
- Nas páginas de índice (árvores-B) despreze as informações de endereço dos filhos. No exemplo, elas são irrelevantes. Faça como no exemplo.
- Os dados de entrada estão ordenados pela coluna do índice granulado.

Exemplo: Seja o conjunto de dados, no qual se fará o granulado por cor:

NOME	COR	ANIMAL	COMIDA	id.
josé	dourado	chopim	feijão	34
gabi	laranja	perú	aipim	22
sueli	lilas	gato	alho	40
antônio	musgo	vaca	batata	41
tatiana	petróleo	cavalo	arroz	37
juca	preto	águia	goiaba	36
romeu	roxo	dragão	maracujá	27
paulo	verde	urubú	rabanete	20

Com este conjunto de dados, ter-se-á a seguintes disposição nos arquivos

Páginas de Dados					
Pág1	josé	dourado	chopim	feijão	34
	gabi	laranja	perú	aipim	22
	sueli	lilas	gato	alho	40
Pág2	antônio	musgo	vaca	batata	41
	tatiana	petróleo	cavalo	arroz	37
	juca	preto	águia	goiaba	36
Pág3	romeu	roxo	dragão	maracujá	27
	paulo	verde	urubú	rabanete	20

A seguir, as páginas de índices

Páginas de índices	
Índice I1	granulado por COR
Página 1 ←raiz	dourado(1:1), musgo(2:1), roxo(3:1)
Índice I2	por NOME
Página 1	antônio(2:1), gabi(1:2)
Página 2 ←raiz	jose(1:1)
Página 3	juca(2:3), paulo(3:2), romeu(3:1), sueli(1:3), tatiana(2:2)
Índice I3	por BICHO
P1	águia(2:3), cavalo(2:2), chopim(1:1), dragão(3:1)
P2 ←raiz	gato(1:3)
P3	perú(1:2),urubú(3:2), vaca(2:1)
Índice I4	por COMIDA
P1	aipim(1:2),alho(1:3)
P2 ←raiz	arroz(2:2)
P3	batata(2:1), feijão(1:1), goiaba(2:3), maracujá(3:1), rabanete(3:2)
Índice I5	por IDADE
P1	20(3:2), 22(1:2), 27(3:1), 34(1:1), 36(2:3)
P2 ←raiz	37(2:2)
P3	40(1:3), 41(2:1)

☞ Para você fazer

Desenhe a árvore B a seguir:

```
     1 2    3    4    5    6    7    8    9 10 11 12 13 14 15 16 17
   +-------------------------------------------------------------------
 1| 1 5    33   34   35   39   40    0    0
 2| 0 3    41   57   69    0    0    0    0  1  5 14  3
 3| 1 6    70   71   72   74   76   77    0
 4| 1 4   172  175  176  177    0    0    0
 5| 1 5    42   43   44   54   56    0    0
 6| 1 5   131  132  133  134  135    0    0
 7| 1 3   137  138  141    0    0    0    0
 8| 1 5    93   95   96  100  101    0    0
 9| 1 5   194  195  197  198  200    0    0
*10| 0 3    82  128  169    0    0    0    0  2 18 11 22
 11| 0 4   136  142  149  157    0    0    0  6  7 23 13 20
 12| 1 5   113  115  116  118  119    0    0
 13| 1 5   151  152  153  154  155    0    0
 14| 1 6    59   64   65   66   67   68    0
 15| 1 7    83   84   85   86   88   89   91
 16| 1 4   185  186  187  189    0    0    0
 17| 1 3   122  126  127    0    0    0    0
 18| 0 4    92  103  110  120    0    0    0  0 15  8 19 12 17
 19| 1 4   105  106  107  109    0    0    0
 20| 1 4   158  160  164  166    0    0    0
 21| 1 3   179  180  183    0    0    0    0
 22| 0 3   178  184  192    0    0    0    0  4 21 16  9
 23| 1 4   143  144  145  147    0    0    0
```

Responda agora:

Qual a altura da árvore B ?	Se o 193 for incluido, qual linha,coluna na matriz?	Se o 61 for incluído, qual linha,coluna na matriz?

Mais uma Instância do mesmo exercício Desenhe a árvore B a seguir:

```
     1 2    3    4    5    6    7    8    9 10 11 12 13 14 15 16 17
   +-------------------------------------------------------------------
 1| 1 5    32   33   36   37   39    0    0
 2| 0 3    40   47   55    0    0    0    0  1 19  8 12
 3| 1 4    94   95   97   98    0    0    0
 4| 1 7   154  155  157  158  160  161  162
 5| 1 7   119  123  125  126  127  128  129
 6| 1 7    78   79   80   85   88   89   92
 7| 1 7   177  178  180  181  182  183  187
 8| 1 4    50   51   52   54    0    0    0
 9| 1 5   111  114  115  116  117    0    0
*10| 0 3    67  109  151    0    0    0    0  2 21 11 17
 11| 0 3   118  130  142    0    0    0    0  9  5 14 16
 12| 1 7    56   58   59   60   62   63   65
 13| 1 5   192  194  197  198  199    0    0
 14| 1 3   134  136  140    0    0    0    0
 15| 1 6   164  166  168  170  173  174    0
 16| 1 6   143  144  145  146  147  150    0
 17| 0 3   163  175  191    0    0    0    0  4 15  7 13
 18| 1 4    70   72   74   76    0    0    0
 19| 1 4    41   42   43   44    0    0    0
 20| 1 4   101  102  105  107    0    0    0
 21| 0 3    77   93   99    0    0    0    0 18  6  3 20
```

Responda agora:

Qual a altura da árvore B ?	Se o 73 for incluido, qual linha,coluna na matriz?	Se o 167 for incluído, qual linha,coluna na matriz?

CEP-UFPR-UTFPR-PUC/Pr-UP Sistemas de Informação 12/08/2024 - 11:35:52.4
Matemática aplicada Prof Dr P Kantek (pkantek@gmail.com)
Evolução de árvores: Árvore-B (e B+) VIVXq92a V: 1.02
76582 LUIZ MIGUEL OLINEK
24CC2401 - 11 primeiro exercicio do T3 _____ / / _____

Árvore B

O conceito de árvore B foi desenvolvido por Rudolf Bayer e Edward McCreight em 1972.

Eles publicaram seu trabalho no artigo *"Organization and Maintenance of Large Ordered Indices"*, que descreve as propriedades e operações básicas das árvores B.

O trabalho de Bayer e McCreight teve um impacto significativo na área de ciência da computação, e as árvores B são agora amplamente utilizadas em bancos de dados e outros sistemas de software.

As árvores B são um tipo de árvore de busca balanceada que pode armazenar um grande número de chaves e valores de maneira eficiente. Elas são mais eficientes do que as árvores de busca binárias tradicionais, pois podem ter mais filhos por nó. As árvores B são usadas em uma variedade de aplicações, incluindo bancos de dados, sistemas de arquivos e sistemas de roteamento.

Eles não explicaram o significado do nome, só nos resta especular que esse B seja de Balanceada. Se você se lembrar das árvores binárias de pesquisa, deve recordar o problema que existe quando as chaves vem (quasi) ordenadas na entrada: a árvore gerada é desbalanceada e pode chegar a ter um desempenho terrível.

Outra qualidade da árvore-B é que ela tem grau k (k filhos por nodo). Em geral este valor é escolhido de modo a otimizar a blocagem de dados no arquivo de memória secundária. De qualquer modo este k modifica o desempenho, que lembrando é proporcional a $\log_k n$, onde n é a quantidade de chaves.

Como toda árvore, ela tem uma raiz que tem de 0 a $2k-1$ chaves e consequentemente tem de 0 a $2k$ filhos. Cada um dos filhos, tem de $k-1$ até $2k-1$ chaves de 0 até $2k$ filhos. Note que a raiz é o único nodo que sofre processamento especial, necessário para quando o arquivo tem uma única chave (por exemplo).

Um caso prático real Suponha um colégio com milhares de alunos, que têm seus dados guardados em um banco de dados. Por hipótese, os dados de cada aluno vão sendo recolhidos quando eles fazem a matrícula, pelo que a ordem física dos dados no arquivo é a data da matrícula. Parece óbvio assinalar que um determinado conjunto de dados (um arquivo) só pode ter uma única ordem física em um determinado instante. Mas nada impede que existam muitas ordens lógicas diferentes para o mesmo arquivo. Tais ordens lógicas são providenciadas por estruturas muito parecidas a uma árvore-b. Neste exemplo, poderíamos ter índices por nome (ordem alfabética), curso, turma, endereço etc.

Vai-se descrever aqui uma árvore-b que dará acesso aos dados por código do aluno.

Suponha uma árvore-b de nome M que providenciará este acesso. Vamos descrevê-la com $k=4$, que é muito pouco para aplicações reais, mas fica razoável para trabalhar com este exemplo. De acordo com a definição, os nodos de M têm entre 3 e 7 códigos de aluno. Para esta dimensão, o nodo de M terá 17 colunas:

coluna 0 um asterisco nesta posição indica que este é o nodo raiz da árvore. Conta como coluna 0 porque na verdade esta informação não está fisicamente aqui, está em outro lugar.

coluna 1 indicativo de folha (1=sim; 0=não)

coluna 2 quantidade de chaves válidas neste nodo (nem sempre o valor inválido é zero)

cols 3 a 9 chaves, sempre em ordem crescente

cols 10 a 17 endereços dos filhos associados a cada chave. Só estão preenchidas para nodos não folha.

Veja um exemplo:

```
    1 2    3    4    5    6    7    8    9 10 11 12 13 14 15 16 17
  +-------------------------------------------------------------------
 1| 1 5    3    6    7    8   10    0    0
 2| 0 4   11   20   32   44    0    0    0  1 20 15  7 12
 3| 1 4  128  131  132  134    0    0    0
 4| 1 5  149  150  151  153  160    0    0
 5| 1 6   65   73   76   79   80   83    0
 6| 1 5  182  183  188  189  191    0    0
 7| 1 3   34   35   37    0    0    0    0
 8| 1 4  106  109  111  114    0    0    0
 9| 1 6  136  137  138  140  143  145    0
*10| 0 2   64  127    0    0    0    0    0  2 17 11
 11| 0 6  135  146  162  171  181  192    0  3  9  4 14 21  6 18
 12| 1 7   46   49   50   55   56   57   63
 13| 1 4   88   89   91   94    0    0    0
 14| 1 5  164  166  167  168  169    0    0
 15| 1 6   21   22   23   28   29   30    0
 16| 1 7  117  118  120  122  123  125  126
 17| 0 4   84   95  105  116    0    0    0  5 13 19  8 16
 18| 1 5  193  195  197  199  200    0    0
 19| 1 5   96   98  100  102  103    0    0
 20| 1 4   14   15   17   19    0    0    0
 21| 1 3  174  175  177    0    0    0    0
```

Antes de continuar no exercício, você deve tentar entender o funcionamento da árvore-b.

- Desenhe a árvore
- identifique a necessidade da coluna 1. Idem para a coluna 2
- Qual a altura da árvore ?
- Quando e como a altura da árvore cresce ?

Agoram suponha que o elemento 187 é incluído. Qual a linha;coluna da matriz acima que é modificada, ou seja onde o elemento recém incluido entra ? Localize-o e veja:

```
    1 2    3    4    5    6    7    8    9 10 11 12 13 14 15 16 17
  +-------------------------------------------------------------------
 1| 1 5    3    6    7    8   10    0    0
 2| 0 4   11   20   32   44    0    0    0  1 20 15  7 12
 3| 1 4  128  131  132  134    0    0    0
 4| 1 5  149  150  151  153  160    0    0
 5| 1 6   65   73   76   79   80   83    0
 6| 1 6  182  183(187)188  189  191    0
 7| 1 3   34   35   37    0    0    0    0
 8| 1 4  106  109  111  114    0    0    0
 9| 1 6  136  137  138  140  143  145    0
*10| 0 2   64  127    0    0    0    0    0  2 17 11
 11| 0 6  135  146  162  171  181  192    0  3  9  4 14 21  6 18
 12| 1 7   46   49   50   55   56   57   63
 13| 1 4   88   89   91   94    0    0    0
 14| 1 5  164  166  167  168  169    0    0
 15| 1 6   21   22   23   28   29   30    0
 16| 1 7  117  118  120  122  123  125  126
 17| 0 4   84   95  105  116    0    0    0  5 13 19  8 16
 18| 1 5  193  195  197  199  200    0    0
 19| 1 5   96   98  100  102  103    0    0
 20| 1 4   14   15   17   19    0    0    0
 21| 1 3  174  175  177    0    0    0    0
```

Como voce pode ver ele está na linha ____ e coluna ____. Algumas características notáveis da árvore-b

- Figurinha carimbada no mundo do software: usam B-trees: SYBASE e do Harbour (antigo Clipper). Também são usados no Oracle, MySQL, PostgreSQL, Microsoft SQL e DB2, da IBM.
- muitas estruturas usam a árvore-b+. O + significa que há uma ligação adicional entre as folhas, para permitir o processamento sequencial dos dados.
- Existem nodos folha e não-folha. Os endereços dos filhos (colunas 10 a 17) só existem para os nodos que os têm (nodos não folhas).
- Os códigos de aluno (colunas 3 a 9 não mostram, para diminuir o ruído), mas ao lado de cada código está o endereço físico real dos dados deste aluno, que afinal é o dado que nos interessa nesta busca.
- Os códigos (colunas 3 a 9) estão em ordem crescente. Isto é fundamental e não pode ser modificado.
- Perceba como, em média, apenas a metade dos códigos está preenchida. Isto é necessário para permitir inclusão de novos dados. Sinaliza também um certo desperdício (overhead) de área em disco que qualquer método de acesso consome.
- Quando um nodo tem todas as chaves preenchidas e chega uma nova, que deveria (deverá) entrar nele, o nodo sofre um processo chamado split e ele é transformado em 2 nodos (cada um com metade dos espaços vazios). Note que o número máximo de chaves em qualquer nodo é sempre ímpar ($2n-1$) o que implica em que o elemento do meio sobe para o *pai* do nodo que está sendo quebrado. A metade anterior vai para um nodo *splitado* e a metade posterior vai para o outro nodo *splitado*.
- Quando as chaves vão sendo excluídas, e um nodo passa a ter menos do que a metade das chaves (o que é proibido) ele é juntado com um nodo irmão e ambos passam a ser um único nodo. Este processo recebe o nome de merge. Não é tão simples assim, há diversos casos particulares, mas a ideia é essa.

Mais um exemplo. Suponha a inclusão do número 26. Em que linha;coluna ele entra ?

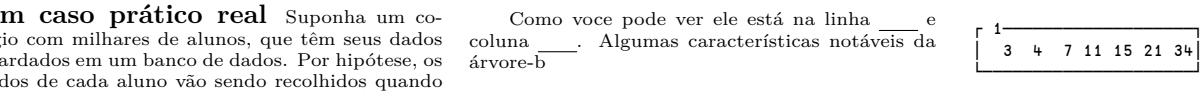
Algoritmo de Inclusão em árvore B

```
1: função INCLUA-BTREE (K:inteiro)
2: X ← raiz da árvore B
3: enquanto nodo X não é folha
4:     X ← filho correto onde K deve ser incluído
5: fim{enquanto}
6: se nodo X está cheio
7:     faça split do nodo X
8:     INCLUA-BTREE (K)
9:     abandone
10: senão
11:     inclua K no nodo X mantendo as chaves ordenadas
12: fim{se}
```

Algoritmos Estão disseminados pela comunidade. Deixo de apresentá-los aqui, pois são bastante específicos, complexos até, e mais importante do que implementá-los é entender-lhes o funcionamento. Até porque se você tiver que usar uma árvore b que não seja para aprender como ela funciona, recomendo fortemente que você use pacotes prontos, já testados e confiáveis.

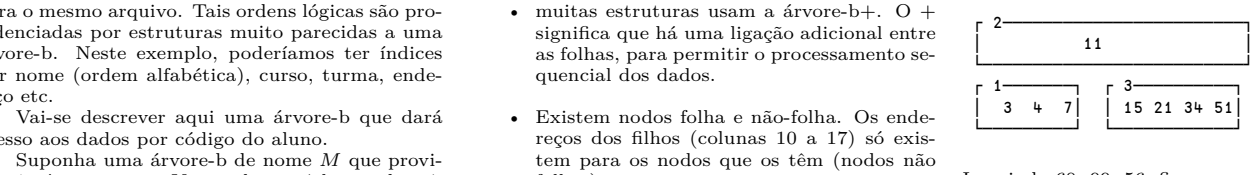
Um exemplo, passo a passo Seja uma árvore-b com $k=4$. Pela definição os nodos terão entre 3 e 7 chaves.

```
criabtree
insira 4, 7, 21, 11, 34, 15, 3
```

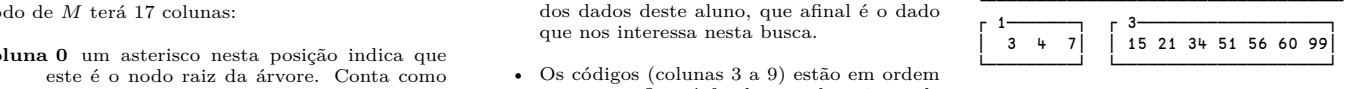


Agora, vamos inserir uma nova chave, o que vai provocar o split

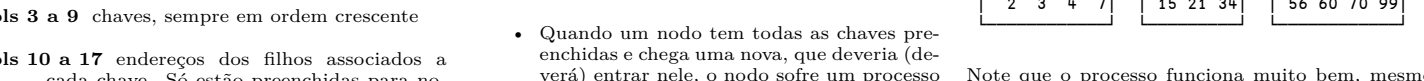
```
insira 51
```



Inserindo 60, 99, 56, fica



Inserindo 2 e 70, o 2 vai na boa para o nodo 1 enquanto o 70 força o split do nodo 3 e fica



Note que o processo funciona muito bem, mesmo que as chaves de entrada estejam desordenadas (ou em ordem aleatória). Isso é garantido pelos espaços em branco que são reservados nos nodos.

Um caso quasi-real: SYBASE

O SYBASE usa árvores B para indexar as tabelas. Existem 2 tipos de árvores B: as granuladas e as não granuladas (clustered). As granuladas são árvores esparsas (nem todo registro de dados tem entrada nos índices) nas quais os registros (na tabela original) são mantidos em ordem dos valores de índice e apenas o primeiro registro em cada página de dados tem uma entrada de índice. Árvores não granuladas são densas e cada um dos registros da tabela tem 1 entrada de índice.

Uma tabela SYBASE com índice granulado exige que seus elementos sejam mantidos em ordem dentro da tabela. Assim, inclusões podem se dar em qualquer página, a depender do conteúdo do campo de índice. Tabelas SYBASE que tem apenas índices não granulados fazem as inclusões de novos elementos sempre ao final das tabelas (na última página).

Tabelas SYBASE podem ter apenas 1 índice granulado, mas podem ter até 250 índices não granulados.

Na criação de um índice granulado, os dados da tabela devem estar colocados nas páginas em ordem.

Cada entrada de índice em sybase pode ser composta por até 16 campos e ter até 256 bytes. Estes campos não precisam estar contiguos. Só não podem ser partes de campos e resultados de cálculos.

O tamanho da página varia, mas o tamanho mais comum é 2K. Existem 5 tipos de páginas em sybase, a saber:

Páginas de Dados contém registros de dados ou registros de log. Ambos são estruturalmente diferentes, mas são construídos do mesmo jeito.

Páginas de índices contém registros de índice

Páginas de texto ou imagem contém BLOBS

Páginas de alocação contém estruturas de dados usadas para gerenciar o processo de alocacão de páginas

páginas de Estatísticas contém estatística de distribuição e uso para os índices.

Página de Dados Se os registros tem tamanho fixo, não haverá tabela de deslocamento na página e o endereço de cada registro é obtido mediante simples cálculo. Neste caso, o tamanho do registro (fixo) é guardado no campo TAMANHO MÍNIMO DE REGISTRO do header.

O header da página é um conjunto de 32 bytes contendo (entre outros): número lógico da página, anterior e próxima página lógica, identificação do objeto ao qual esta página pertence, primeira entrada de registro disponível nesta página, deslocamento do espaço livre, tamanho mínimo do registro

Depois vem a área de registros, que contém um número inteiro de slots para guardar registros. Um registro não atravessa limites de páginas

Tabela de deslocamento: Um conjunto de deslocamentos desde o início da página dando o início de cada um dos registros. Este conjunto cresce do fim da página para o início e só existe quando o tamanho do registro é variável.

Cada registro na página recebe um número de registro. É um campo de 1 byte e assim restringe-se o número de registros em uma página a 256. O número do registro usado nos índices não granulados (densos) é uma combinação do número da página com o número do registro na página.

O registro pode ocupar a página inteira (2048-32=2016). Não há tamanho mínimo para o registro, mas só pode haver 256 registros na página.

Os índices SYBASE são organizados na forma de arvores B + (o + significa que há uma ligação adicional entre as folhas, para permitir o processamento sequencial dos dados).

No exemplo aqui estudado, vão-se fazer algumas considerações/simplificações descritas a seguir.

Suponha um arquivo de pessoas contendo nome, cor preferida, bicho de estimação, alimento que gosta e idade.

Construindo um arquivo SYBASE contendo estes dados e que esteja sujeito às seguintes características:

- Registros de tamanho fixo igual a 40 bytes (9+9+9+9+4).
- Um índice granulado ou esparso (clustered) por um dos campos. No exemplo que é igual para todos, o índice esparso é por cor.
- Quatro índices não granulados (densos) pelos demais atributos.
- Páginas de dados com espaço para 4 registros e fator de preenchimento de 4:1 (usar apenas 3 espaços na carga original).
- Páginas de índices com espaço para 5 chaves (nome, cor, bicho, alimento ou idade). Lembrar que este número é impar, o que indica que no *split* vai subir a terceira chave (das 5 que existem).
- A identificação de cada registro deve ser página:número do registro na página. Por exemplo, a identificação 3:2, indica o segundo registro da terceira página.
- Nas páginas de índice (árvores-B) despreze as informações de endereço dos filhos. No exemplo, elas são irrelevantes. Faça como no exemplo.
- Os dados de entrada estão ordenados pela coluna do índice granulado.

Exemplo: Seja o conjunto de dados, no qual se fará o granulado por cor:

NOME	COR	ANIMAL	COMIDA	id.
josé	dourado	chopim	feijão	34
gabi	laranja	perú	aipim	22
sueli	lilas	gato	alho	40
antônio	musgo	vaca	batata	41
tatiana	petróleo	cavalo	arroz	37
juca	preto	águia	goiaba	36
romeu	roxo	dragão	maracujá	27
paulo	verde	urubú	rabanete	20

Com este conjunto de dados, ter-se-á a seguintes disposição nos arquivos

Páginas de Dados					
Pág1	josé	dourado	chopim	feijão	34
	gabi	laranja	perú	aipim	22
	sueli	lilas	gato	alho	40
Pág2	antônio	musgo	vaca	batata	41
	tatiana	petróleo	cavalo	arroz	37
	juca	preto	águia	goiaba	36
Pág3	romeu	roxo	dragão	maracujá	27
	paulo	verde	urubú	rabanete	20

A seguir, as páginas de índices

Páginas de índices	
Índice I1	granulado por COR
Página 1 ←raiz	dourado(1:1), musgo(2:1), roxo(3:1)
Índice I2	por NOME
Página 1	antônio(2:1), gabi(1:2)
Página 2 ←raiz	jose(1:1)
Página 3	juca(2:3), paulo(3:2), romeu(3:1), sueli(1:3), tatiana(2:2)
Índice I3	por BICHO
P1	águia(2:3), cavalo(2:2), chopim(1:1), dragão(3:1)
P2 ←raiz	gato(1:3)
P3	perú(1:2),urubú(3:2), vaca(2:1)
Índice I4	por COMIDA
P1	aipim(1:2),alho(1:3)
P2 ←raiz	arroz(2:2)
P3	batata(2:1), feijão(1:1), goiaba(2:3), maracujá(3:1), rabanete(3:2)
Índice I5	por IDADE
P1	20(3:2), 22(1:2), 27(3:1), 34(1:1), 36(2:3)
P2 ←raiz	37(2:2)
P3	40(1:3), 41(2:1)

☞ Para você fazer

Desenhe a árvore B a seguir:

```
    1 2    3    4    5    6    7    8    9 10 11 12 13 14 15 16 17
   +-------------------------------------------------------------------
 1| 1 7   31   33   35   36   38   40   41
 2| 0 5   42   50   61   68   72    0    0  1 15 19  5 23 12
 3| 1 4  129  131  132  135    0    0    0
 4| 1 6   97   98  101  102  103  104    0
 5| 1 4   64   65   66   67    0    0    0
 6| 1 7  158  159  160  164  166  168  169
 7| 1 7   86   87   88   89   90   93   95
 8| 1 6  172  175  178  179  180  184    0
 9| 1 4  137  138  139  142    0    0    0
*10| 0 3   84  126  157    0    0    0    0  2 18 11 22
 11| 0 3  136  143  147    0    0    0    0  3  9 13 20
 12| 1 6   76   77   78   80   81   83    0
 13| 1 3  144  145  146    0    0    0    0
 14| 1 5  120  121  122  123  125    0    0
 15| 1 3   43   44   49    0    0    0    0
 16| 1 4  187  189  191  194    0    0    0
 17| 1 3  110  111  113    0    0    0    0
 18| 0 3   96  107  119    0    0    0    0  7  4 17 14
 19| 1 4   51   55   58   59    0    0    0
 20| 1 4  149  152  153  156    0    0    0
 21| 1 3  197  199  200    0    0    0    0
 22| 0 3  171  185  195    0    0    0    0  6  8 16 21
 23| 1 3   69   70   71    0    0    0    0
```

Responda agora:

Qual a altura da árvore B ?	Se o 163 for incluido, qual linha,coluna na matriz?	Se o 34 for incluído, qual linha,coluna na matriz?

Mais uma Instância do mesmo exercício Desenhe a árvore B a seguir:

```
    1 2    3    4    5    6    7    8    9 10 11 12 13 14 15 16 17
   +-------------------------------------------------------------------
 1| 1 6   33   35   40   41   48   49    0
 2| 0 5   50   55   65   73   79    0    0  1 15  6 21  8 19
 3| 1 4  136  137  138  141    0    0    0
 4| 1 4  102  103  105  106    0    0    0
 5| 1 3  162  165  166    0    0    0    0
 6| 1 4   60   61   63   64    0    0    0
 7| 1 7  146  149  150  151  156  157  159
 8| 1 4   74   76   77   78    0    0    0
 9| 1 3  185  186  187    0    0    0    0
*10| 0 2   88  135    0    0    0    0    0  2 16 11
 11| 0 6  145  161  169  178  182  188    0  3  7  5 18 23  9 22
 12| 1 6  109  110  111  112  113  114    0
 13| 1 5   89   90   91   92   93    0    0
 14| 1 5  120  121  123  125  127    0    0
 15| 1 4   51   52   53   54    0    0    0
 16| 0 5   94  100  108  117  128    0    0 13 20  4 12 14 17
 17| 1 5  129  130  131  132  133    0    0
 18| 1 4  173  175  176  177    0    0    0
 19| 1 4   82   83   85   86    0    0    0
 20| 1 3   95   96   99    0    0    0    0
 21| 1 3   67   70   71    0    0    0    0
 22| 1 5  190  191  194  198  200    0    0
 23| 1 3  179  180  181    0    0    0    0
```

Responda agora:

Qual a altura da árvore B ?	Se o 118 for incluido, qual linha,coluna na matriz?	Se o 184 for incluído, qual linha,coluna na matriz?

401-76582 - iro e

CEP-UFPR-UTFPR-PUC/Pr-UP Sistemas de Informação 12/08/2024 - 11:35:52.4
Matemática aplicada Prof Dr P Kantek (pkantek@gmail.com)
Evolução de árvores: Árvore-B (e B+) VIVXq92a V: 1.02
76599 NICOLE DA SILVA DE AZEVEDO
24CC2401 - 12 primeiro exercicio do T3 _____ / / _____

Árvore B

O conceito de árvore B foi desenvolvido por Rudolf Bayer e Edward McCreight em 1972.

Eles publicaram seu trabalho no artigo *"Organization and Maintenance of Large Ordered Indices"*, que descreve as propriedades e operações básicas das árvores B.

O trabalho de Bayer e McCreight teve um impacto significativo na área de ciência da computação, e as árvores B são agora amplamente utilizadas em bancos de dados e outros sistemas de software.

As árvores B são um tipo de árvore de busca balanceada que pode armazenar um grande número de chaves e valores de maneira eficiente. Elas são mais eficientes do que as árvores de busca binárias tradicionais, pois podem ter mais filhos por nó. As árvores B são usadas em uma variedade de aplicações, incluindo bancos de dados, sistemas de arquivos e sistemas de roteamento.

Eles não explicaram o significado do nome, só nos resta especular que esse B seja de Balanceada. Se você se lembrar das árvores binárias de pesquisa, deve recordar o problema que existe quando as chaves vem (quasi) ordenadas na entrada: a árvore gerada é desbalanceada e pode chegar a ter um desempenho terrível.

Outra qualidade da árvore-B é que ela tem grau k (k filhos por nodo). Em geral este valor é escolhido de modo a otimizar a blocagem de dados no arquivo de memória secundária. De qualquer modo este k modifica o desempenho, que lembrando é proporcional a $\log_k n$, onde n é a quantidade de chaves.

Como toda árvore, ela tem uma raiz que tem de 0 a $2k-1$ chaves e consequentemente tem de 0 a $2k$ filhos. Cada um dos filhos, tem de $k-1$ até $2k-1$ chaves de 0 até $2k$ filhos. Note que a raiz é o único nodo que sofre processamento especial, necessário para quando o arquivo tem uma única chave (por exemplo).

Um caso prático real Suponha um colégio com milhares de alunos, que têm seus dados guardados em um banco de dados. Por hipótese, os dados de cada aluno vão sendo recolhidos quando eles fazem a matrícula, pelo que a ordem física dos dados no arquivo é a data da matrícula. Parece óbvio assinalar que um determinado conjunto de dados (um arquivo) só pode ter uma única ordem física em um determinado instante. Mas nada impede que existam muitas ordens lógicas diferentes para o mesmo arquivo. Tais ordens lógicas são providenciadas por estruturas muito parecidas a uma árvore-b. Neste exemplo, poderíamos ter índices por nome (ordem alfabética), curso, turma, endereço etc.

Vai-se descrever aqui uma árvore-b que dará acesso aos dados por código do aluno.

Suponha uma árvore-b de nome M que providenciará este acesso. Vamos descrevê-la com $k=4$, que é muito pouco para aplicações reais, mas fica razoável para trabalhar com este exemplo. De acordo com a definição, os nodos de M têm entre 3 e 7 códigos de aluno. Para esta dimensão, o nodo de M terá 17 colunas:

coluna 0 um asterisco nesta posição indica que este é o nodo raiz da árvore. Conta como coluna 0 porque na verdade esta informação não está fisicamente aqui, está em outro lugar.

coluna 1 indicativo de folha (1=sim; 0=não)

coluna 2 quantidade de chaves válidas neste nodo (nem sempre o valor inválido é zero)

cols 3 a 9 chaves, sempre em ordem crescente

cols 10 a 17 endereços dos filhos associados a cada chave. Só estão preenchidas para nodos não folha.

Veja um exemplo:

```
    1 2    3    4    5    6    7    8    9 10 11 12 13 14 15 16 17
  +-------------------------------------------------------------------
 1| 1 5    3    6    7    8   10    0    0
 2| 0 4   11   20   32   44    0    0    0  1 20 15  7 12
 3| 1 4  128  131  132  134    0    0    0
 4| 1 5  149  150  151  153  160    0    0
 5| 1 6   65   73   76   79   80   83    0
 6| 1 5  182  183  188  189  191    0    0
 7| 1 3   34   35   37    0    0    0    0
 8| 1 4  106  109  111  114    0    0    0
 9| 1 6  136  137  138  140  143  145    0
*10| 0 2   64  127    0    0    0    0    0  2 17 11
 11| 0 6  135  146  162  171  181  192    0  3  9  4 14 21  6 18
 12| 1 7   46   49   50   55   56   57   63
 13| 1 4   88   89   91   94    0    0    0
 14| 1 5  164  166  167  168  169    0    0
 15| 1 6   21   22   23   28   29   30    0
 16| 1 7  117  118  120  122  123  125  126
 17| 0 4   84   95  105  116    0    0    0  5 13 19  8 16
 18| 1 5  193  195  197  199  200    0    0
 19| 1 5   96   98  100  102  103    0    0
 20| 1 4   14   15   17   19    0    0    0
 21| 1 3  174  175  177    0    0    0    0
```

Antes de continuar no exercício, você deve tentar entender o funcionamento da árvore-b.

- Desenhe a árvore
- identifique a necessidade da coluna 1. Idem para a coluna 2
- Qual a altura da árvore ?
- Quando e como a altura da árvore cresce ?

Agoram suponha que o elemento 187 é incluído. Qual a linha;coluna da matriz acima que é modificada, ou seja onde o elemento recém incluido entra ? Localize-o e veja:

```
    1 2    3    4    5    6    7    8    9 10 11 12 13 14 15 16 17
  +-------------------------------------------------------------------
 1| 1 5    3    6    7    8   10    0    0
 2| 0 4   11   20   32   44    0    0    0  1 20 15  7 12
 3| 1 4  128  131  132  134    0    0    0
 4| 1 5  149  150  151  153  160    0    0
 5| 1 6   65   73   76   79   80   83    0
 6| 1 6  182  183(187)188  189  191    0
 7| 1 3   34   35   37    0    0    0    0
 8| 1 4  106  109  111  114    0    0    0
 9| 1 6  136  137  138  140  143  145    0
*10| 0 2   64  127    0    0    0    0    0  2 17 11
 11| 0 6  135  146  162  171  181  192    0  3  9  4 14 21  6 18
 12| 1 7   46   49   50   55   56   57   63
 13| 1 4   88   89   91   94    0    0    0
 14| 1 5  164  166  167  168  169    0    0
 15| 1 6   21   22   23   28   29   30    0
 16| 1 7  117  118  120  122  123  125  126
 17| 0 4   84   95  105  116    0    0    0  5 13 19  8 16
 18| 1 5  193  195  197  199  200    0    0
 19| 1 5   96   98  100  102  103    0    0
 20| 1 4   14   15   17   19    0    0    0
 21| 1 3  174  175  177    0    0    0    0
```

Como voce pode ver ele está na linha ____ e coluna ____. Algumas características notáveis da árvore-b

- Figurinha carimbada no mundo do software: usam B-trees: SYBASE e do Harbour (antigo Clipper). Também são usados no Oracle, MySQL, PostgreSQL, Microsoft SQL e DB2, da IBM.
- muitas estruturas usam a árvore-b+. O + significa que há uma ligação adicional entre as folhas, para permitir o processamento sequencial dos dados.
- Existem nodos folha e não-folha. Os endereços dos filhos (colunas 10 a 17) só existem para os nodos que os têm (nodos não folhas).
- Os códigos de aluno (colunas 3 a 9 não mostram, para diminuir o ruído), mas ao lado de cada código está o endereço físico real dos dados deste aluno, que afinal é o dado que nos interessa nesta busca.
- Os códigos (colunas 3 a 9) estão em ordem crescente. Isto é fundamental e não pode ser modificado.
- Perceba como, em média, apenas a metade dos códigos está preenchida. Isto é necessário para permitir inclusão de novos dados. Sinaliza também um certo desperdício (overhead) de área em disco que qualquer método de acesso consome.
- Quando um nodo tem todas as chaves preenchidas e chega uma nova, que deveria (deverá) entrar nele, o nodo sofre um processo chamado split e ele é transformado em 2 nodos (cada um com metade dos espaços vazios). Note que o número máximo de chaves em qualquer nodo é sempre ímpar ($2n-1$) o que implica em que o elemento do meio sobe para o *pai* do nodo que está sendo quebrado. A metade anterior vai para um nodo *splitado* e a metade posterior vai para o outro nodo *splitado*.
- Quando as chaves vão sendo excluídas, e um nodo passa a ter menos do que a metade das chaves (o que é proibido) ele é juntado com um nodo irmão e ambos passam a ser um único nodo. Este processo recebe o nome de merge. Não é tão simples assim, há diversos casos particulares, mas a ideia é essa.

Mais um exemplo. Suponha a inclusão do número 26. Em que linha;coluna ele entra ?

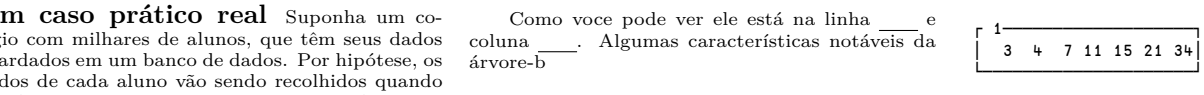
Algoritmo de Inclusão em árvore B

```
1: função INCLUA-BTREE (K:inteiro)
2: X ← raiz da árvore B
3: enquanto nodo X não é folha
4:     X ← filho correto onde K deve ser incluído
5: fim{enquanto}
6: se nodo X está cheio
7:     faça split do nodo X
8:     INCLUA-BTREE (K)
9:     abandone
10: senão
11:     inclua K no nodo X mantendo as chaves ordenadas
12: fim{se}
```

Algoritmos Estão disseminados pela comunidade. Deixo de apresentá-los aqui, pois são bastante específicos, complexos até, e mais importante do que implementá-los é entender-lhes o funcionamento. Até porque se você tiver que usar uma árvore b que não seja para aprender como ela funciona, recomendo fortemente que você use pacotes prontos, já testados e confiáveis.

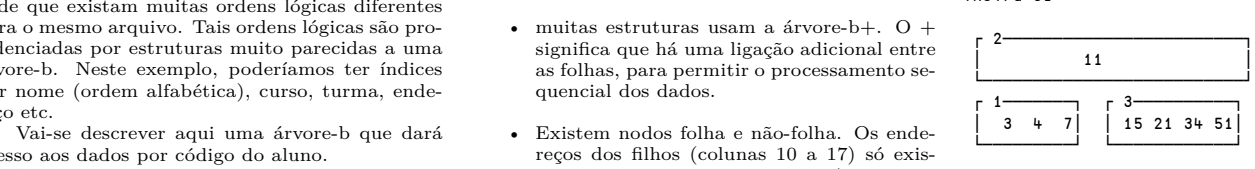
Um exemplo, passo a passo Seja uma árvore-b com $k=4$. Pela definição os nodos terão entre 3 e 7 chaves.

```
criabtree
insira 4, 7, 21, 11, 34, 15, 3
```

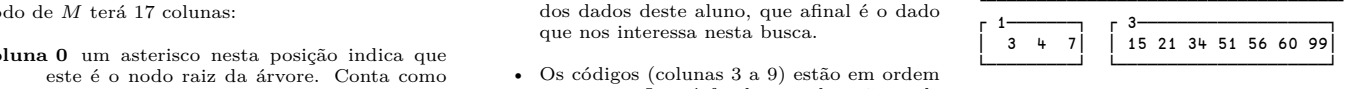


Agora, vamos inserir uma nova chave, o que vai provocar o split

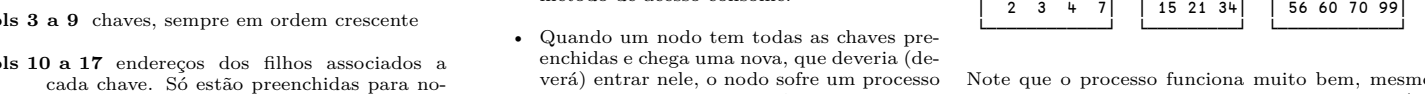
```
insira 51
```



Inserindo 60, 99, 56, fica



Inserindo 2 e 70, o 2 vai na boa para o nodo 1 enquanto o 70 força o split do nodo 3 e fica



Note que o processo funciona muito bem, mesmo que as chaves de entrada estejam desordenadas (ou em ordem aleatória). Isso é garantido pelos espaços em branco que são reservados nos nodos.

Um caso quasi-real: SYBASE

O SYBASE usa árvores B para indexar as tabelas. Existem 2 tipos de árvores B: as granuladas e as não granuladas (clustered). As granuladas são árvores esparsas (nem todo registro de dados tem entrada nos índices) nas quais os registros (na tabela original) são mantidos em ordem dos valores de índice e apenas o primeiro registro em cada página de dados tem uma entrada de índice. Árvores não granuladas são densas e cada um dos registros da tabela tem 1 entrada de índice.

Uma tabela SYBASE com índice granulado exige que seus elementos sejam mantidos em ordem dentro da tabela. Assim, inclusões podem se dar em qualquer página, a depender do conteúdo do campo de índice. Tabelas SYBASE que tem apenas índices não granulados fazem as inclusões de novos elementos sempre ao final das tabelas (na última página).

Tabelas SYBASE podem ter apenas 1 índice granulado, mas podem ter até 250 índices não granulados.

Na criação de um índice granulado, os dados da tabela devem estar colocados nas páginas em ordem.

Cada entrada de índice em sybase pode ser composta por até 16 campos e ter até 256 bytes. Estes campos não precisam estar contiguos. Só não podem ser partes de campos e resultados de cálculos.

O tamanho da página varia, mas o tamanho mais comum é 2K. Existem 5 tipos de páginas em sybase, a saber:

Páginas de Dados contém registros de dados ou registros de log. Ambos são estruturalmente diferentes, mas são construídos do mesmo jeito.

Páginas de índices contém registros de índice

Páginas de texto ou imagem contém BLOBS

Páginas de alocação contém estruturas de dados usadas para gerenciar o processo de alocacão de páginas

páginas de Estatísticas contém estatística de distribuição e uso para os índices.

Página de Dados Se os registros tem tamanho fixo, não haverá tabela de deslocamento na página e o endereço de cada registro é obtido mediante simples cálculo. Neste caso, o tamanho do registro (fixo) é guardado no campo TAMANHO MÍNIMO DE REGISTRO do header.

O header da página é um conjunto de 32 bytes contendo (entre outros): número lógico da página, anterior e próxima página lógica, identificação do objeto ao qual esta página pertence, primeira entrada de registro disponível nesta página, deslocamento do espaço livre, tamanho mínimo do registro

Depois vem a área de registros, que contém um número inteiro de slots para guardar registros. Um registro não atravessa limites de páginas

Tabela de deslocamento: Um conjunto de deslocamentos desde o início da página dando o início de cada um dos registros. Este conjunto cresce do fim da página para o início e só existe quando o tamanho do registro é variável.

Cada registro na página recebe um número de registro. É um campo de 1 byte e assim restringe-se o número de registros em uma página a 256. O número do registro usado nos índices não granulados (densos) é uma combinação do número da página com o número do registro na página.

O registro pode ocupar a página inteira (2048-32=2016). Não há tamanho mínimo para o registro, mas só pode haver 256 registros na página.

Os índices SYBASE são organizados na forma de arvores B + (o + significa que há uma ligação adicional entre as folhas, para permitir o processamento sequencial dos dados).

No exemplo aqui estudado, vão-se fazer algumas considerações/simplificações descritas a seguir.

Suponha um arquivo de pessoas contendo nome, cor preferida, bicho de estimação, alimento que gosta e idade.

Construindo um arquivo SYBASE contendo estes dados e que esteja sujeito às seguintes características:

- Registros de tamanho fixo igual a 40 bytes (9+9+9+9+4).
- Um índice granulado ou esparso (clustered) por um dos campos. No exemplo que é igual para todos, o índice esparso é por cor.
- Quatro índices não granulados (densos) pelos demais atributos.
- Páginas de dados com espaço para 4 registros e fator de preenchimento de 4:1 (usar apenas 3 espaços na carga original).
- Páginas de índices com espaço para 5 chaves (nome, cor, bicho, alimento ou idade). Lembrar que este número é impar, o que indica que no *split* vai subir a terceira chave (das 5 que existem).
- A identificação de cada registro deve ser página:número do registro na página. Por exemplo, a identificação 3:2, indica o segundo registro da terceira página.
- Nas páginas de índice (árvores-B) despreze as informações de endereço dos filhos. No exemplo, elas são irrelevantes. Faça como no exemplo.
- Os dados de entrada estão ordenados pela coluna do índice granulado.

Exemplo: Seja o conjunto de dados, no qual se fará o granulado por cor:

NOME	COR	ANIMAL	COMIDA	id.
josé	dourado	chopim	feijão	34
gabi	laranja	perú	aipim	22
sueli	lilas	gato	alho	40
antônio	musgo	vaca	batata	41
tatiana	petróleo	cavalo	arroz	37
juca	preto	águia	goiaba	36
romeu	roxo	dragão	maracujá	27
paulo	verde	urubú	rabanete	20

Com este conjunto de dados, ter-se-á a seguintes disposição nos arquivos

Páginas de Dados					
Pág1	josé	dourado	chopim	feijão	34
	gabi	laranja	perú	aipim	22
	sueli	lilas	gato	alho	40
Pág2	antônio	musgo	vaca	batata	41
	tatiana	petróleo	cavalo	arroz	37
	juca	preto	águia	goiaba	36
Pág3	romeu	roxo	dragão	maracujá	27
	paulo	verde	urubú	rabanete	20

A seguir, as páginas de índices

Páginas de índices	
Índice I1	granulado por COR
Página 1 ←raiz	dourado(1:1), musgo(2:1), roxo(3:1)
Índice I2	por NOME
Página 1	antônio(2:1), gabi(1:2)
Página 2 ←raiz	jose(1:1)
Página 3	juca(2:3), paulo(3:2), romeu(3:1), sueli(1:3), tatiana(2:2)
Índice I3	por BICHO
P1	águia(2:3), cavalo(2:2), chopim(1:1), dragão(3:1)
P2 ←raiz	gato(1:3)
P3	perú(1:2),urubú(3:2), vaca(2:1)
Índice I4	por COMIDA
P1	aipim(1:2),alho(1:3)
P2 ←raiz	arroz(2:2)
P3	batata(2:1), feijão(1:1), goiaba(2:3), maracujá(3:1), rabanete(3:2)
Índice I5	por IDADE
P1	20(3:2), 22(1:2), 27(3:1), 34(1:1), 36(2:3)
P2 ←raiz	37(2:2)
P3	40(1:3), 41(2:1)

☞ Para você fazer

Desenhe a árvore B a seguir:

```
    1 2    3    4    5    6    7    8    9 10 11 12 13 14 15 16 17
  +-------------------------------------------------------------------
 1| 1 7    34   35   38   39   41   42   43
 2| 0 7    44   51   62   71   76   84   93   1   9 17   4 12 22   6 16
 3| 1 3   159 161 163    0    0    0    0
 4| 1 5    63   65   66   68   70    0    0
 5| 1 5   104 107 108 109 110    0    0
 6| 1 4    85   87   89   91    0    0    0
 7| 1 7   176 177 179 181 183 184 188
 8| 1 4   123 126 131 132    0    0    0
 9| 1 3    45   46   49    0    0    0    0
*10| 0 2   103 140    0    0    0    0    0   2 11 19
 11| 0 3   111 120 135    0    0    0    0   5 15   8 18
 12| 1 3    72   74   75    0    0    0    0
 13| 1 6   193 194 195 197 198 199    0
 14| 1 3   142 144 146    0    0    0    0
 15| 1 5   113 114 115 117 118    0    0
 16| 1 7    94   95   98   99 100 101 102
 17| 1 5    53   54   55   59   61    0    0
 18| 1 3   137 138 139    0    0    0    0
 19| 0 5   148 157 164 173 190    0    0  14 20   3 21   7 13
 20| 1 5   150 152 154 155 156    0    0
 21| 1 4   165 166 169 171    0    0    0
 22| 1 4    78   79   81   82    0    0    0
```

Responda agora:

Qual a altura da árvore B ?	Se o 174 for incluido, qual linha,coluna na matriz?	Se o 178 for incluído, qual linha,coluna na matriz?

Mais uma Instância do mesmo exercício Desenhe a árvore B a seguir:

```
    1 2    3    4    5    6    7    8    9 10 11 12 13 14 15 16 17
  +-------------------------------------------------------------------
 1| 1 5    32   33   34   35   37    0    0
 2| 0 3    38   50   56    0    0    0    0   1   7 18   5
 3| 1 3   108 109 110    0    0    0    0
 4| 1 6   150 151 152 154 158 162    0
 5| 1 6    57   58   59   61   63   64    0
 6| 1 3   181 184 188    0    0    0    0
 7| 1 5    43   45   46   48   49    0    0
 8| 1 4    88   90   92   93    0    0    0
 9| 1 3   132 133 135    0    0    0    0
*10| 0 3    66 107 149    0    0    0    0   2 19 11 22
 11| 0 4   111 118 129 136    0    0    0   3 21 14   9 23
 12| 1 5   101 102 104 105 106    0    0
 13| 1 3    67   68   69    0    0    0    0
 14| 1 5   119 122 123 127 128    0    0
 15| 1 6   191 192 193 194 196 198    0
 16| 1 5    74   76   77   80   81    0    0
 17| 1 6   165 168 169 170 171 175    0
 18| 1 4    51   52   53   54    0    0    0
 19| 0 4    72   83   94 100    0    0    0  13 16   8 20 12
 20| 1 4    95   96   97   98    0    0    0
 21| 1 5   112 113 114 115 116    0    0
 22| 0 3   164 179 190    0    0    0    0   4 17   6 15
 23| 1 5   140 141 143 144 145    0    0
```

Responda agora:

Qual a altura da árvore B ?	Se o 131 for incluido, qual linha,coluna na matriz?	Se o 153 for incluído, qual linha,coluna na matriz?

CEP-UFPR-UTFPR-PUC/Pr-UP Sistemas de Informação 12/08/2024 - 11:35:52.4
Matemática aplicada Prof Dr P Kantek (pkantek@gmail.com)
Evolução de árvores: Árvore-B (e B+) VIVXq92a V: 1.02
76601 PAULO JOSE FREIRE CORREA
24CC2401 - 13 primeiro exercicio do T3 _____ / / _____

Árvore B

O conceito de árvore B foi desenvolvido por Rudolf Bayer e Edward McCreight em 1972.

Eles publicaram seu trabalho no artigo *"Organization and Maintenance of Large Ordered Indices"*, que descreve as propriedades e operações básicas das árvores B.

O trabalho de Bayer e McCreight teve um impacto significativo na área de ciência da computação, e as árvores B são agora amplamente utilizadas em bancos de dados e outros sistemas de software.

As árvores B são um tipo de árvore de busca balanceada que pode armazenar um grande número de chaves e valores de maneira eficiente. Elas são mais eficientes do que as árvores de busca binárias tradicionais, pois podem ter mais filhos por nó. As árvores B são usadas em uma variedade de aplicações, incluindo bancos de dados, sistemas de arquivos e sistemas de roteamento.

Eles não explicaram o significado do nome, só nos resta especular que esse B seja de Balanceada. Se você se lembrar das árvores binárias de pesquisa, deve recordar o problema que existe quando as chaves vem (quasi) ordenadas na entrada: a árvore gerada é desbalanceada e pode chegar a ter um desempenho terrível.

Outra qualidade da árvore-B é que ela tem grau k (k filhos por nodo). Em geral este valor é escolhido de modo a otimizar a blocagem de dados no arquivo de memória secundária. De qualquer modo este k modifica o desempenho, que lembrando é proporcional a $\log_k n$, onde n é a quantidade de chaves.

Como toda árvore, ela tem uma raiz que tem de 0 a $2k-1$ chaves e consequentemente tem de 0 a $2k$ filhos. Cada um dos filhos, tem de $k-1$ até $2k-1$ chaves de 0 até $2k$ filhos. Note que a raiz é o único nodo que sofre processamento especial, necessário para quando o arquivo tem uma única chave (por exemplo).

Um caso prático real Suponha um colégio com milhares de alunos, que têm seus dados guardados em um banco de dados. Por hipótese, os dados de cada aluno vão sendo recolhidos quando eles fazem a matrícula, pelo que a ordem física dos dados no arquivo é a data da matrícula. Parece óbvio assinalar que um determinado conjunto de dados (um arquivo) só pode ter uma única ordem física em um determinado instante. Mas nada impede que existam muitas ordens lógicas diferentes para o mesmo arquivo. Tais ordens lógicas são providenciadas por estruturas muito parecidas a uma árvore-b. Neste exemplo, poderíamos ter índices por nome (ordem alfabética), curso, turma, endereço etc.

Vai-se descrever aqui uma árvore-b que dará acesso aos dados por código do aluno.

Suponha uma árvore-b de nome M que providenciará este acesso. Vamos descrevê-la com $k=4$, que é muito pouco para aplicações reais, mas fica razoável para trabalhar com este exemplo. De acordo com a definição, os nodos de M têm entre 3 e 7 códigos de aluno. Para esta dimensão, o nodo de M terá 17 colunas:

coluna 0 um asterisco nesta posição indica que este é o nodo raiz da árvore. Conta como coluna 0 porque na verdade esta informação não está fisicamente aqui, está em outro lugar.

coluna 1 indicativo de folha (1=sim; 0=não)

coluna 2 quantidade de chaves válidas neste nodo (nem sempre o valor inválido é zero)

cols 3 a 9 chaves, sempre em ordem crescente

cols 10 a 17 endereços dos filhos associados a cada chave. Só estão preenchidas para nodos não folha.

Veja um exemplo:

```
    1 2    3    4    5    6    7    8    9 10 11 12 13 14 15 16 17
  +-------------------------------------------------------------------
 1| 1 5    3    6    7    8   10    0    0
 2| 0 4   11   20   32   44    0    0    0  1 20 15  7 12
 3| 1 4  128  131  132  134    0    0    0
 4| 1 5  149  150  151  153  160    0    0
 5| 1 6   65   73   76   79   80   83    0
 6| 1 5  182  183  188  189  191    0    0
 7| 1 3   34   35   37    0    0    0    0
 8| 1 4  106  109  111  114    0    0    0
 9| 1 6  136  137  138  140  143  145    0
*10| 0 2   64  127    0    0    0    0    0  2 17 11
 11| 0 6  135  146  162  171  181  192    0  3  9  4 14 21  6 18
 12| 1 7   46   49   50   55   56   57   63
 13| 1 4   88   89   91   94    0    0    0
 14| 1 5  164  166  167  168  169    0    0
 15| 1 6   21   22   23   28   29   30    0
 16| 1 7  117  118  120  122  123  125  126
 17| 0 4   84   95  105  116    0    0    0  5 13 19  8 16
 18| 1 5  193  195  197  199  200    0    0
 19| 1 5   96   98  100  102  103    0    0
 20| 1 4   14   15   17   19    0    0    0
 21| 1 3  174  175  177    0    0    0    0
```

Antes de continuar no exercício, você deve tentar entender o funcionamento da árvore-b.

- Desenhe a árvore
- identifique a necessidade da coluna 1. Idem para a coluna 2
- Qual a altura da árvore ?
- Quando e como a altura da árvore cresce ?

Agoram suponha que o elemento 187 é incluído. Qual a linha;coluna da matriz acima que é modificada, ou seja onde o elemento recém incluido entra ? Localize-o e veja:

```
    1 2    3    4    5    6    7    8    9 10 11 12 13 14 15 16 17
  +-------------------------------------------------------------------
 1| 1 5    3    6    7    8   10    0    0
 2| 0 4   11   20   32   44    0    0    0  1 20 15  7 12
 3| 1 4  128  131  132  134    0    0    0
 4| 1 5  149  150  151  153  160    0    0
 5| 1 6   65   73   76   79   80   83    0
 6| 1 6  182  183(187)188  189  191    0
 7| 1 3   34   35   37    0    0    0    0
 8| 1 4  106  109  111  114    0    0    0
 9| 1 6  136  137  138  140  143  145    0
*10| 0 2   64  127    0    0    0    0    0  2 17 11
 11| 0 6  135  146  162  171  181  192    0  3  9  4 14 21  6 18
 12| 1 7   46   49   50   55   56   57   63
 13| 1 4   88   89   91   94    0    0    0
 14| 1 5  164  166  167  168  169    0    0
 15| 1 6   21   22   23   28   29   30    0
 16| 1 7  117  118  120  122  123  125  126
 17| 0 4   84   95  105  116    0    0    0  5 13 19  8 16
 18| 1 5  193  195  197  199  200    0    0
 19| 1 5   96   98  100  102  103    0    0
 20| 1 4   14   15   17   19    0    0    0
 21| 1 3  174  175  177    0    0    0    0
```

Como voce pode ver ele está na linha ____ e coluna ____. Algumas características notáveis da árvore-b

- Figurinha carimbada no mundo do software: usam B-trees: SYBASE e do Harbour (antigo Clipper). Também são usados no Oracle, MySQL, PostgreSQL, Microsoft SQL e DB2, da IBM.
- muitas estruturas usam a árvore-b+. O + significa que há uma ligação adicional entre as folhas, para permitir o processamento sequencial dos dados.
- Existem nodos folha e não-folha. Os endereços dos filhos (colunas 10 a 17) só existem para os nodos que os têm (nodos não folhas).
- Os códigos de aluno (colunas 3 a 9 não mostram, para diminuir o ruído), mas ao lado de cada código está o endereço físico real dos dados deste aluno, que afinal é o dado que nos interessa nesta busca.
- Os códigos (colunas 3 a 9) estão em ordem crescente. Isto é fundamental e não pode ser modificado.
- Perceba como, em média, apenas a metade dos códigos está preenchida. Isto é necessário para permitir inclusão de novos dados. Sinaliza também um certo desperdício (overhead) de área em disco que qualquer método de acesso consome.
- Quando um nodo tem todas as chaves preenchidas e chega uma nova, que deveria (deverá) entrar nele, o nodo sofre um processo chamado split e ele é transformado em 2 nodos (cada um com metade dos espaços vazios). Note que o número máximo de chaves em qualquer nodo é sempre ímpar ($2n-1$) o que implica em que o elemento do meio sobe para o *pai* do nodo que está sendo quebrado. A metade anterior vai para um nodo *splitado* e a metade posterior vai para o outro nodo *splitado*.
- Quando as chaves vão sendo excluídas, e um nodo passa a ter menos do que a metade das chaves (o que é proibido) ele é juntado com um nodo irmão e ambos passam a ser um único nodo. Este processo recebe o nome de merge. Não é tão simples assim, há diversos casos particulares, mas a ideia é essa.

Mais um exemplo. Suponha a inclusão do número 26. Em que linha;coluna ele entra ?

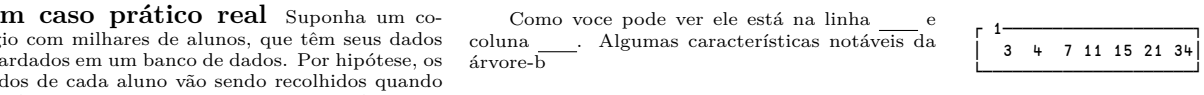
Algoritmo de Inclusão em árvore B

```
1: função INCLUA-BTREE (K:inteiro)
2: X ← raiz da árvore B
3: enquanto nodo X não é folha
4:     X ← filho correto onde K deve ser incluído
5: fim{enquanto}
6: se nodo X está cheio
7:     faça split do nodo X
8:     INCLUA-BTREE (K)
9:     abandone
10: senão
11:     inclua K no nodo X mantendo as chaves ordenadas
12: fim{se}
```

Algoritmos Estão disseminados pela comunidade. Deixo de apresentá-los aqui, pois são bastante específicos, complexos até, e mais importante do que implementá-los é entender-lhes o funcionamento. Até porque se você tiver que usar uma árvore b que não seja para aprender como ela funciona, recomendo fortemente que você use pacotes prontos, já testados e confiáveis.

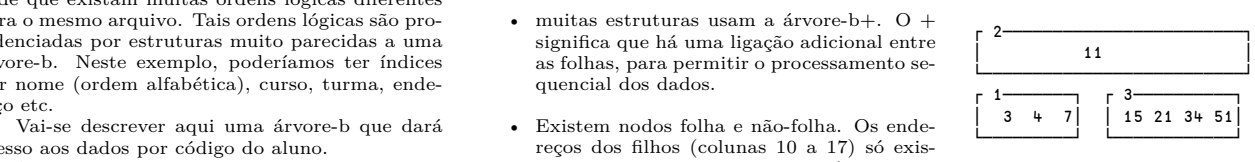
Um exemplo, passo a passo Seja uma árvore-b com $k=4$. Pela definição os nodos terão entre 3 e 7 chaves.

```
criabtree
insira 4, 7, 21, 11, 34, 15, 3
```

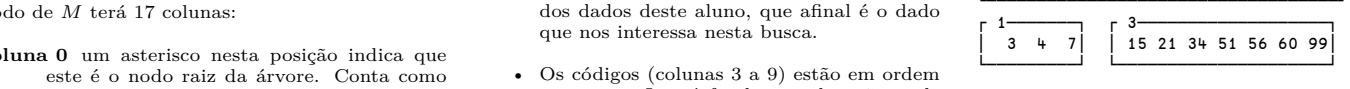


Agora, vamos inserir uma nova chave, o que vai provocar o split

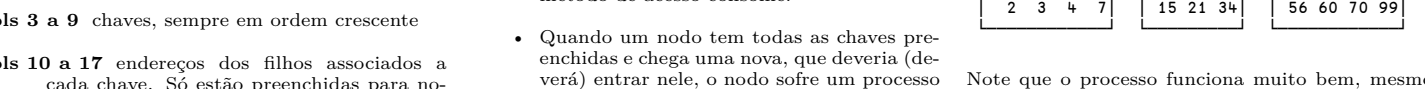
```
insira 51
```



Inserindo 60, 99, 56, fica



Inserindo 2 e 70, o 2 vai na boa para o nodo 1 enquanto o 70 força o split do nodo 3 e fica



Note que o processo funciona muito bem, mesmo que as chaves de entrada estejam desordenadas (ou em ordem aleatória). Isso é garantido pelos espaços em branco que são reservados nos nodos.

Um caso quasi-real: SYBASE

O SYBASE usa árvores B para indexar as tabelas. Existem 2 tipos de árvores B: as granuladas e as não granuladas (clustered). As granuladas são árvores esparsas (nem todo registro de dados tem entrada nos índices) nas quais os registros (na tabela original) são mantidos em ordem dos valores de índice e apenas o primeiro registro em cada página de dados tem uma entrada de índice. Árvores não granuladas são densas e cada um dos registros da tabela tem 1 entrada de índice.

Uma tabela SYBASE com índice granulado exige que seus elementos sejam mantidos em ordem dentro da tabela. Assim, inclusões podem se dar em qualquer página, a depender do conteúdo do campo de índice. Tabelas SYBASE que tem apenas índices não granulados fazem as inclusões de novos elementos sempre ao final das tabelas (na última página).

Tabelas SYBASE podem ter apenas 1 índice granulado, mas podem ter até 250 índices não granulados.

Na criação de um índice granulado, os dados da tabela devem estar colocados nas páginas em ordem.

Cada entrada de índice em sybase pode ser composta por até 16 campos e ter até 256 bytes. Estes campos não precisam estar contiguos. Só não podem ser partes de campos e resultados de cálculos.

O tamanho da página varia, mas o tamanho mais comum é 2K. Existem 5 tipos de páginas em sybase, a saber:

Páginas de Dados contém registros de dados ou registros de log. Ambos são estruturalmente diferentes, mas são construídos do mesmo jeito.

Páginas de índices contém registros de índice

Páginas de texto ou imagem contém BLOBS

Páginas de alocação contém estruturas de dados usadas para gerenciar o processo de alocacão de páginas

páginas de Estatísticas contém estatística de distribuição e uso para os índices.

Página de Dados Se os registros tem tamanho fixo, não haverá tabela de deslocamento na página e o endereço de cada registro é obtido mediante simples cálculo. Neste caso, o tamanho do registro (fixo) é guardado no campo TAMANHO MÍNIMO DE REGISTRO do header.

O header da página é um conjunto de 32 bytes contendo (entre outros): número lógico da página, anterior e próxima página lógica, identificação do objeto ao qual esta página pertence, primeira entrada de registro disponível nesta página, deslocamento do espaço livre, tamanho mínimo do registro

Depois vem a área de registros, que contém um número inteiro de slots para guardar registros. Um registro não atravessa limites de páginas

Tabela de deslocamento: Um conjunto de deslocamentos desde o início da página dando o início de cada um dos registros. Este conjunto cresce do fim da página para o início e só existe quando o tamanho do registro é variável.

Cada registro na página recebe um número de registro. É um campo de 1 byte e assim restringe-se o número de registros em uma página a 256. O número do registro usado nos índices não granulados (densos) é uma combinação do número da página com o número do registro na página.

O registro pode ocupar a página inteira (2048-32=2016). Não há tamanho mínimo para o registro, mas só pode haver 256 registros na página.

Os índices SYBASE são organizados na forma de arvores B + (o + significa que há uma ligação adicional entre as folhas, para permitir o processamento sequencial dos dados).

No exemplo aqui estudado, vão-se fazer algumas considerações/simplificações descritas a seguir.

Suponha um arquivo de pessoas contendo nome, cor preferida, bicho de estimação, alimento que gosta e idade.

Construindo um arquivo SYBASE contendo estes dados e que esteja sujeito às seguintes características:

- Registros de tamanho fixo igual a 40 bytes (9+9+9+9+4).
- Um índice granulado ou esparso (clustered) por um dos campos. No exemplo que é igual para todos, o índice esparso é por cor.
- Quatro índices não granulados (densos) pelos demais atributos.
- Páginas de dados com espaço para 4 registros e fator de preenchimento de 4:1 (usar apenas 3 espaços na carga original).
- Páginas de índices com espaço para 5 chaves (nome, cor, bicho, alimento ou idade). Lembrar que este número é impar, o que indica que no *split* vai subir a terceira chave (das 5 que existem).
- A identificação de cada registro deve ser página:número do registro na página. Por exemplo, a identificação 3:2, indica o segundo registro da terceira página.
- Nas páginas de índice (árvores-B) despreze as informações de endereço dos filhos. No exemplo, elas são irrelevantes. Faça como no exemplo.
- Os dados de entrada estão ordenados pela coluna do índice granulado.

Exemplo: Seja o conjunto de dados, no qual se fará o granulado por cor:

NOME	COR	ANIMAL	COMIDA	id.
josé	dourado	chopim	feijão	34
gabi	laranja	perú	aipim	22
sueli	lilas	gato	alho	40
antônio	musgo	vaca	batata	41
tatiana	petróleo	cavalo	arroz	37
juca	preto	águia	goiaba	36
romeu	roxo	dragão	maracujá	27
paulo	verde	urubú	rabanete	20

Com este conjunto de dados, ter-se-á a seguintes disposição nos arquivos

Páginas de Dados					
Pág1	josé	dourado	chopim	feijão	34
	gabi	laranja	perú	aipim	22
	sueli	lilas	gato	alho	40
Pág2	antônio	musgo	vaca	batata	41
	tatiana	petróleo	cavalo	arroz	37
	juca	preto	águia	goiaba	36
Pág3	romeu	roxo	dragão	maracujá	27
	paulo	verde	urubú	rabanete	20

A seguir, as páginas de índices

Páginas de índices	
Índice I1	granulado por COR
Página 1 ←raiz	dourado(1:1), musgo(2:1), roxo(3:1)
Índice I2	por NOME
Página 1	antônio(2:1), gabi(1:2)
Página 2 ←raiz	jose(1:1)
Página 3	juca(2:3), paulo(3:2), romeu(3:1), sueli(1:3), tatiana(2:2)
Índice I3	por BICHO
P1	águia(2:3), cavalo(2:2), chopim(1:1), dragão(3:1)
P2 ←raiz	gato(1:3)
P3	perú(1:2),urubú(3:2), vaca(2:1)
Índice I4	por COMIDA
P1	aipim(1:2),alho(1:3)
P2 ←raiz	arroz(2:2)
P3	batata(2:1), feijão(1:1), goiaba(2:3), maracujá(3:1), rabanete(3:2)
Índice I5	por IDADE
P1	20(3:2), 22(1:2), 27(3:1), 34(1:1), 36(2:3)
P2 ←raiz	37(2:2)
P3	40(1:3), 41(2:1)

☞ Para você fazer

Desenhe a árvore B a seguir:

```
    1 2    3    4    5    6    7    8    9 10 11 12 13 14 15 16 17
  +-------------------------------------------------------------------
 1| 1 7    32   33   34   38   39   41   42
 2| 0 4    45   51   60   69    0    0    0  1  9 22  7 24
 3| 1 6   116  117  119  120  122  123    0
 4| 1 4   160  161  162  166    0    0    0
 5| 1 4   145  146  147  148    0    0    0
 6| 1 3    94   98  100    0    0    0    0
 7| 1 4    61   65   67   68    0    0    0
 8| 1 5   177  178  179  181  182    0    0
 9| 1 3    47   48   49    0    0    0    0
*10| 0 3    85  115  151    0    0    0    0  2 23 11 18
 11| 0 3   125  130  143    0    0    0    0  3 13 15  5
 12| 1 4   110  111  112  113    0    0    0
 13| 1 4   126  127  128  129    0    0    0
 14| 1 3   195  198  199    0    0    0    0
 15| 1 7   132  134  135  136  138  139  140
 16| 1 6    86   87   88   89   90   92    0
 17| 1 4   152  153  155  157    0    0    0
 18| 0 5   158  167  174  183  192    0    0 17  4 21  8 20 14
 19| 1 4   103  104  105  106    0    0    0
 20| 1 4   184  187  189  190    0    0    0
 21| 1 3   168  171  172    0    0    0    0
 22| 1 4    52   54   56   58    0    0    0
 23| 0 3    93  102  109    0    0    0    0 16  6 19 12
 24| 1 3    72   74   77    0    0    0    0
```

Responda agora:

Qual a altura da árvore B ?	Se o 144 for incluido, qual linha,coluna na matriz?	Se o 81 for incluído, qual linha,coluna na matriz?

Mais uma Instância do mesmo exercício Desenhe a árvore B a seguir:

```
    1 2    3    4    5    6    7    8    9 10 11 12 13 14 15 16 17
  +-------------------------------------------------------------------
 1| 1 4    33   34   35   37    0    0    0
 2| 0 5    42   49   58   65   74    0    0  1 20 13 21  7 18
 3| 1 4   160  163  164  165    0    0    0
 4| 1 4    84   85   87   89    0    0    0
 5| 1 5   180  181  182  183  187    0    0
 6| 1 5   130  132  133  134  135    0    0
 7| 1 4    66   67   70   73    0    0    0
 8| 1 4   146  148  149  153    0    0    0
 9| 1 3   103  105  106    0    0    0    0
*10| 0 3    82  129  159    0    0    0    0  2 19 11 23
 11| 0 3   137  144  154    0    0    0    0  6 15  8 22
 12| 1 7   191  194  195  197  198  199  200
 13| 1 5    50   51   52   53   55    0    0
 14| 1 7   167  170  171  172  173  175  177
 15| 1 3   139  141  143    0    0    0    0
 16| 1 7   114  115  116  123  125  127  128
 17| 1 6    92   94   95   96   97   99    0
 18| 1 4    76   77   79   80    0    0    0
 19| 0 3    91  101  107    0    0    0    0  4 17  9 16
 20| 1 4    43   45   46   47    0    0    0
 21| 1 4    59   60   62   63    0    0    0
 22| 1 3   155  156  157    0    0    0    0
 23| 0 3   166  178  188    0    0    0    0  3 14  5 12
```

Responda agora:

Qual a altura da árvore B ?	Se o 131 for incluido, qual linha,coluna na matriz?	Se o 72 for incluído, qual linha,coluna na matriz?

CEP-UFPR-UTFPR-PUC/Pr-UP Sistemas de Informação 12/08/2024 - 11:35:52.4
Matemática aplicada Prof Dr P Kantek (pkantek@gmail.com)
Evolução de árvores: Árvore-B (e B+) VIVXq92a V: 1.02
76618 PEDRO HENRIQUE PIEKARSKI DE OL 24CC2401 - 14 primeiro exercicio do T3 _____ / / _____

Árvore B

O conceito de árvore B foi desenvolvido por Rudolf Bayer e Edward McCreight em 1972.

Eles publicaram seu trabalho no artigo *"Organization and Maintenance of Large Ordered Indices"*, que descreve as propriedades e operações básicas das árvores B.

O trabalho de Bayer e McCreight teve um impacto significativo na área de ciência da computação, e as árvores B são agora amplamente utilizadas em bancos de dados e outros sistemas de software.

As árvores B são um tipo de árvore de busca balanceada que pode armazenar um grande número de chaves e valores de maneira eficiente. Elas são mais eficientes do que as árvores de busca binárias tradicionais, pois podem ter mais filhos por nó. As árvores B são usadas em uma variedade de aplicações, incluindo bancos de dados, sistemas de arquivos e sistemas de roteamento.

Eles não explicaram o significado do nome, só nos resta especular que esse B seja de Balanceada. Se você se lembrar das árvores binárias de pesquisa, deve recordar o problema que existe quando as chaves vem (quasi) ordenadas na entrada: a árvore gerada é desbalanceada e pode chegar a ter um desempenho terrível.

Outra qualidade da árvore-B é que ela tem grau k (k filhos por nodo). Em geral este valor é escolhido de modo a otimizar a blocagem de dados no arquivo de memória secundária. De qualquer modo este k modifica o desempenho, que lembrando é proporcional a $\log_k n$, onde n é a quantidade de chaves.

Como toda árvore, ela tem uma raiz que tem de 0 a $2k-1$ chaves e consequentemente tem de 0 a $2k$ filhos. Cada um dos filhos, tem de $k-1$ até $2k-1$ chaves de 0 até $2k$ filhos. Note que a raiz é o único nodo que sofre processamento especial, necessário para quando o arquivo tem uma única chave (por exemplo).

Um caso prático real Suponha um colégio com milhares de alunos, que têm seus dados guardados em um banco de dados. Por hipótese, os dados de cada aluno vão sendo recolhidos quando eles fazem a matrícula, pelo que a ordem física dos dados no arquivo é a data da matrícula. Parece óbvio assinalar que um determinado conjunto de dados (um arquivo) só pode ter uma única ordem física em um determinado instante. Mas nada impede que existam muitas ordens lógicas diferentes para o mesmo arquivo. Tais ordens lógicas são providenciadas por estruturas muito parecidas a uma árvore-b. Neste exemplo, poderíamos ter índices por nome (ordem alfabética), curso, turma, endereço etc.

Vai-se descrever aqui uma árvore-b que dará acesso aos dados por código do aluno.

Suponha uma árvore-b de nome M que providenciará este acesso. Vamos descrevê-la com $k=4$, que é muito pouco para aplicações reais, mas fica razoável para trabalhar com este exemplo. De acordo com a definição, os nodos de M têm entre 3 e 7 códigos de aluno. Para esta dimensão, o nodo de M terá 17 colunas:

coluna 0 um asterisco nesta posição indica que este é o nodo raiz da árvore. Conta como coluna 0 porque na verdade esta informação não está fisicamente aqui, está em outro lugar.

coluna 1 indicativo de folha (1=sim; 0=não)

coluna 2 quantidade de chaves válidas neste nodo (nem sempre o valor inválido é zero)

cols 3 a 9 chaves, sempre em ordem crescente

cols 10 a 17 endereços dos filhos associados a cada chave. Só estão preenchidas para nodos não folha.

Veja um exemplo:

```
    1 2    3    4    5    6    7    8    9 10 11 12 13 14 15 16 17
  +-------------------------------------------------------------------
 1| 1 5    3    6    7    8   10    0    0
 2| 0 4   11   20   32   44    0    0    0  1 20 15  7 12
 3| 1 4  128  131  132  134    0    0    0
 4| 1 5  149  150  151  153  160    0    0
 5| 1 6   65   73   76   79   80   83    0
 6| 1 5  182  183  188  189  191    0    0
 7| 1 3   34   35   37    0    0    0    0
 8| 1 4  106  109  111  114    0    0    0
 9| 1 6  136  137  138  140  143  145    0
*10| 0 2   64  127    0    0    0    0    0  2 17 11
 11| 0 6  135  146  162  171  181  192    0  3  9  4 14 21  6 18
 12| 1 7   46   49   50   55   56   57   63
 13| 1 4   88   89   91   94    0    0    0
 14| 1 5  164  166  167  168  169    0    0
 15| 1 6   21   22   23   28   29   30    0
 16| 1 7  117  118  120  122  123  125  126
 17| 0 4   84   95  105  116    0    0    0  5 13 19  8 16
 18| 1 5  193  195  197  199  200    0    0
 19| 1 5   96   98  100  102  103    0    0
 20| 1 4   14   15   17   19    0    0    0
 21| 1 3  174  175  177    0    0    0    0
```

Antes de continuar no exercício, você deve tentar entender o funcionamento da árvore-b.

- Desenhe a árvore
- identifique a necessidade da coluna 1. Idem para a coluna 2
- Qual a altura da árvore ?
- Quando e como a altura da árvore cresce ?

Agoram suponha que o elemento 187 é incluído. Qual a linha;coluna da matriz acima que é modificada, ou seja onde o elemento recém incluido entra ? Localize-o e veja:

```
    1 2    3    4    5    6    7    8    9 10 11 12 13 14 15 16 17
  +-------------------------------------------------------------------
 1| 1 5    3    6    7    8   10    0    0
 2| 0 4   11   20   32   44    0    0    0  1 20 15  7 12
 3| 1 4  128  131  132  134    0    0    0
 4| 1 5  149  150  151  153  160    0    0
 5| 1 6   65   73   76   79   80   83    0
 6| 1 6  182  183(187)188  189  191    0
 7| 1 3   34   35   37    0    0    0    0
 8| 1 4  106  109  111  114    0    0    0
 9| 1 6  136  137  138  140  143  145    0
*10| 0 2   64  127    0    0    0    0    0  2 17 11
 11| 0 6  135  146  162  171  181  192    0  3  9  4 14 21  6 18
 12| 1 7   46   49   50   55   56   57   63
 13| 1 4   88   89   91   94    0    0    0
 14| 1 5  164  166  167  168  169    0    0
 15| 1 6   21   22   23   28   29   30    0
 16| 1 7  117  118  120  122  123  125  126
 17| 0 4   84   95  105  116    0    0    0  5 13 19  8 16
 18| 1 5  193  195  197  199  200    0    0
 19| 1 5   96   98  100  102  103    0    0
 20| 1 4   14   15   17   19    0    0    0
 21| 1 3  174  175  177    0    0    0    0
```

Como voce pode ver ele está na linha ____ e coluna ____. Algumas características notáveis da árvore-b

- Figurinha carimbada no mundo do software: usam B-trees: SYBASE e do Harbour (antigo Clipper). Também são usados no Oracle, MySQL, PostgreSQL, Microsoft SQL e DB2, da IBM.
- muitas estruturas usam a árvore-b+. O + significa que há uma ligação adicional entre as folhas, para permitir o processamento sequencial dos dados.
- Existem nodos folha e não-folha. Os endereços dos filhos (colunas 10 a 17) só existem para os nodos que os têm (nodos não folhas).
- Os códigos de aluno (colunas 3 a 9 não mostram, para diminuir o ruído), mas ao lado de cada código está o endereço físico real dos dados deste aluno, que afinal é o dado que nos interessa nesta busca.
- Os códigos (colunas 3 a 9) estão em ordem crescente. Isto é fundamental e não pode ser modificado.
- Perceba como, em média, apenas a metade dos códigos está preenchida. Isto é necessário para permitir inclusão de novos dados. Sinaliza também um certo desperdício (overhead) de área em disco que qualquer método de acesso consome.
- Quando um nodo tem todas as chaves preenchidas e chega uma nova, que deveria (deverá) entrar nele, o nodo sofre um processo chamado split e ele é transformado em 2 nodos (cada um com metade dos espaços vazios). Note que o número máximo de chaves em qualquer nodo é sempre ímpar ($2n-1$) o que implica em que o elemento do meio sobe para o *pai* do nodo que está sendo quebrado. A metade anterior vai para um nodo *splitado* e a metade posterior vai para o outro nodo *splitado*.
- Quando as chaves vão sendo excluídas, e um nodo passa a ter menos do que a metade das chaves (o que é proibido) ele é juntado com um nodo irmão e ambos passam a ser um único nodo. Este processo recebe o nome de merge. Não é tão simples assim, há diversos casos particulares, mas a ideia é essa.

Mais um exemplo. Suponha a inclusão do número 26. Em que linha;coluna ele entra ?

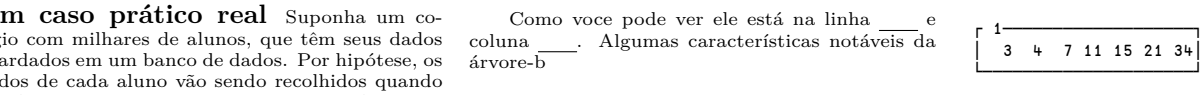
Algoritmo de Inclusão em árvore B

```
1: função INCLUA-BTREE (K:inteiro)
2: X ← raiz da árvore B
3: enquanto nodo X não é folha
4:     X ← filho correto onde K deve ser incluído
5: fim{enquanto}
6: se nodo X está cheio
7:     faça split do nodo X
8:     INCLUA-BTREE (K)
9:     abandone
10: senão
11:     inclua K no nodo X mantendo as chaves ordenadas
12: fim{se}
```

Algoritmos Estão disseminados pela comunidade. Deixo de apresentá-los aqui, pois são bastante específicos, complexos até, e mais importante do que implementá-los é entender-lhes o funcionamento. Até porque se você tiver que usar uma árvore b que não seja para aprender como ela funciona, recomendo fortemente que você use pacotes prontos, já testados e confiáveis.

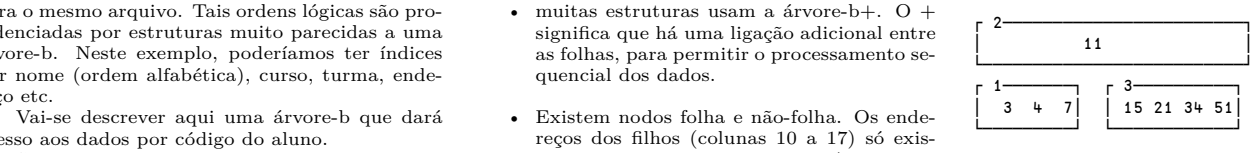
Um exemplo, passo a passo Seja uma árvore-b com $k=4$. Pela definição os nodos terão entre 3 e 7 chaves.

```
criabtree
insira 4, 7, 21, 11, 34, 15, 3
```

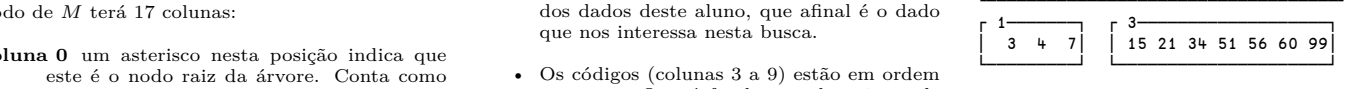


Agora, vamos inserir uma nova chave, o que vai provocar o split

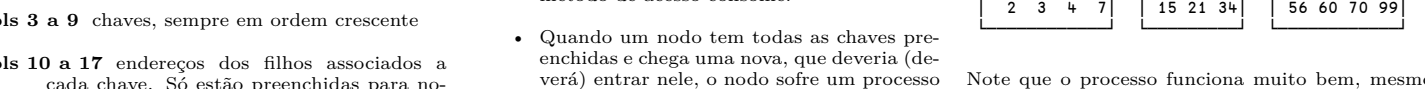
```
insira 51
```



Inserindo 60, 99, 56, fica



Inserindo 2 e 70, o 2 vai na boa para o nodo 1 enquanto o 70 força o split do nodo 3 e fica



Note que o processo funciona muito bem, mesmo que as chaves de entrada estejam desordenadas (ou em ordem aleatória). Isso é garantido pelos espaços em branco que são reservados nos nodos.

Um caso quasi-real: SYBASE

O SYBASE usa árvores B para indexar as tabelas. Existem 2 tipos de árvores B: as granuladas e as não granuladas (clustered). As granuladas são árvores esparsas (nem todo registro de dados tem entrada nos índices) nas quais os registros (na tabela original) são mantidos em ordem dos valores de índice e apenas o primeiro registro em cada página de dados tem uma entrada de índice. Árvores não granuladas são densas e cada um dos registros da tabela tem 1 entrada de índice.

Uma tabela SYBASE com índice granulado exige que seus elementos sejam mantidos em ordem dentro da tabela. Assim, inclusões podem se dar em qualquer página, a depender do conteúdo do campo de índice. Tabelas SYBASE que tem apenas índices não granulados fazem as inclusões de novos elementos sempre ao final das tabelas (na última página).

Tabelas SYBASE podem ter apenas 1 índice granulado, mas podem ter até 250 índices não granulados.

Na criação de um índice granulado, os dados da tabela devem estar colocados nas páginas em ordem.

Cada entrada de índice em sybase pode ser composta por até 16 campos e ter até 256 bytes. Estes campos não precisam estar contiguos. Só não podem ser partes de campos e resultados de cálculos.

O tamanho da página varia, mas o tamanho mais comum é 2K. Existem 5 tipos de páginas em sybase, a saber:

Páginas de Dados contém registros de dados ou registros de log. Ambos são estruturalmente diferentes, mas são construídos do mesmo jeito.

Páginas de índices contém registros de índice

Páginas de texto ou imagem contém BLOBS

Páginas de alocação contém estruturas de dados usadas para gerenciar o processo de alocacão de páginas

páginas de Estatísticas contém estatística de distribuição e uso para os índices.

Página de Dados Se os registros tem tamanho fixo, não haverá tabela de deslocamento na página e o endereço de cada registro é obtido mediante simples cálculo. Neste caso, o tamanho do registro (fixo) é guardado no campo TAMANHO MÍNIMO DE REGISTRO do header.

O header da página é um conjunto de 32 bytes contendo (entre outros): número lógico da página, anterior e próxima página lógica, identificação do objeto ao qual esta página pertence, primeira entrada de registro disponível nesta página, deslocamento do espaço livre, tamanho mínimo do registro

Depois vem a área de registros, que contém um número inteiro de slots para guardar registros. Um registro não atravessa limites de páginas

Tabela de deslocamento: Um conjunto de deslocamentos desde o início da página dando o início de cada um dos registros. Este conjunto cresce do fim da página para o início e só existe quando o tamanho do registro é variável.

Cada registro na página recebe um número de registro. É um campo de 1 byte e assim restringe-se o número de registros em uma página a 256. O número do registro usado nos índices não granulados (densos) é uma combinação do número da página com o número do registro na página.

O registro pode ocupar a página inteira (2048-32=2016). Não há tamanho mínimo para o registro, mas só pode haver 256 registros na página.

Os índices SYBASE são organizados na forma de arvores B + (o + significa que há uma ligação adicional entre as folhas, para permitir o processamento sequencial dos dados).

No exemplo aqui estudado, vão-se fazer algumas considerações/simplificações descritas a seguir.

Suponha um arquivo de pessoas contendo nome, cor preferida, bicho de estimação, alimento que gosta e idade.

Construindo um arquivo SYBASE contendo estes dados e que esteja sujeito às seguintes características:

- Registros de tamanho fixo igual a 40 bytes (9+9+9+9+4).
- Um índice granulado ou esparso (clustered) por um dos campos. No exemplo que é igual para todos, o índice esparso é por cor.
- Quatro índices não granulados (densos) pelos demais atributos.
- Páginas de dados com espaço para 4 registros e fator de preenchimento de 4:1 (usar apenas 3 espaços na carga original).
- Páginas de índices com espaço para 5 chaves (nome, cor, bicho, alimento ou idade). Lembrar que este número é impar, o que indica que no *split* vai subir a terceira chave (das 5 que existem).
- A identificação de cada registro deve ser página:número do registro na página. Por exemplo, a identificação 3:2, indica o segundo registro da terceira página.
- Nas páginas de índice (árvores-B) despreze as informações de endereço dos filhos. No exemplo, elas são irrelevantes. Faça como no exemplo.
- Os dados de entrada estão ordenados pela coluna do índice granulado.

Exemplo: Seja o conjunto de dados, no qual se fará o granulado por cor:

NOME	COR	ANIMAL	COMIDA	id.
josé	dourado	chopim	feijão	34
gabi	laranja	perú	aipim	22
sueli	lilas	gato	alho	40
antônio	musgo	vaca	batata	41
tatiana	petróleo	cavalo	arroz	37
juca	preto	águia	goiaba	36
romeu	roxo	dragão	maracujá	27
paulo	verde	urubú	rabanete	20

Com este conjunto de dados, ter-se-á a seguintes disposição nos arquivos

Páginas de Dados					
Pág1	josé	dourado	chopim	feijão	34
	gabi	laranja	perú	aipim	22
	sueli	lilas	gato	alho	40
Pág2	antônio	musgo	vaca	batata	41
	tatiana	petróleo	cavalo	arroz	37
	juca	preto	águia	goiaba	36
Pág3	romeu	roxo	dragão	maracujá	27
	paulo	verde	urubú	rabanete	20

A seguir, as páginas de índices

Páginas de índices	
Índice I1	granulado por COR
Página 1 ←raiz	dourado(1:1), musgo(2:1), roxo(3:1)
Índice I2	por NOME
Página 1	antônio(2:1), gabi(1:2)
Página 2 ←raiz	jose(1:1)
Página 3	juca(2:3), paulo(3:2), romeu(3:1), sueli(1:3), tatiana(2:2)
Índice I3	por BICHO
P1	águia(2:3), cavalo(2:2), chopim(1:1), dragão(3:1)
P2 ←raiz	gato(1:3)
P3	perú(1:2),urubú(3:2), vaca(2:1)
Índice I4	por COMIDA
P1	aipim(1:2),alho(1:3)
P2 ←raiz	arroz(2:2)
P3	batata(2:1), feijão(1:1), goiaba(2:3), maracujá(3:1), rabanete(3:2)
Índice I5	por IDADE
P1	20(3:2), 22(1:2), 27(3:1), 34(1:1), 36(2:3)
P2 ←raiz	37(2:2)
P3	40(1:3), 41(2:1)

☞ Para você fazer

Desenhe a árvore B a seguir:

```
    1 2    3    4    5    6    7    8    9 10 11 12 13 14 15 16 17
  +-----------------------------------------------------------------
 1| 1 6    31   33   34   35   36   39    0
 2| 0 3    40   49   59    0    0    0    0  1 12 16  4
 3| 1 3   153  154  156    0    0    0    0
 4| 1 5    60   61   63   64   65    0    0
 5| 1 7   181  182  183  184  185  186  188
 6| 1 6   100  102  103  110  113  115    0
 7| 1 6   131  135  136  138  139  143    0
 8| 1 7   170  171  172  173  176  177  178
 9| 1 4    81   82   84   85    0    0    0
*10| 0 2    66  130    0    0    0    0    0  2 17 11
 11| 0 6   144  150  157  169  180  189    0  7 14  3 20  8  5 19
 12| 1 3    41   42   43    0    0    0    0
 13| 1 6    89   90   93   95   96   97    0
 14| 1 5   145  146  147  148  149    0    0
 15| 1 6    67   68   70   74   76   77    0
 16| 1 5    50   52   53   54   55    0    0
 17| 0 4    79   88   98  116    0    0    0 15  9 13  6 18
 18| 1 4   117  119  124  127    0    0    0
 19| 1 6   190  191  193  194  195  200    0
 20| 1 6   159  162  164  165  166  168    0
```

Responda agora:

Qual a altura da árvore B ?	Se o 133 for incluido, qual linha,coluna na matriz?	Se o 142 for incluído, qual linha,coluna na matriz?

Mais uma Instância do mesmo exercício Desenhe a árvore B a seguir:

```
    1 2    3    4    5    6    7    8    9 10 11 12 13 14 15 16 17
  +-----------------------------------------------------------------
 1| 1 5    32   34   39   40   42    0    0
 2| 0 3    43   55   61    0    0    0    0  1 17 15  6
 3| 1 7   111  113  114  117  118  120  122
 4| 1 7   139  140  141  142  143  144  145
 5| 1 6   171  173  174  178  181  183    0
 6| 1 7    62   63   65   66   67   68   69
 7| 1 4    72   73   75   80    0    0    0
 8| 1 3   187  189  190    0    0    0    0
 9| 1 6    92   93   95   97   99  100    0
*10| 0 3    70  110  161    0    0    0    0  2 21 11 19
 11| 0 4   123  135  146  154    0    0    0  3 12  4 13 22
 12| 1 6   124  125  127  129  131  134    0
 13| 1 4   148  149  150  151    0    0    0
 14| 1 4    83   84   86   87    0    0    0
 15| 1 4    56   57   58   59    0    0    0
 16| 1 4   192  193  194  197    0    0    0
 17| 1 5    44   45   51   52   54    0    0
 18| 1 3   162  164  165    0    0    0    0
 19| 0 3   169  186  191    0    0    0    0 18  5  8 16
 20| 1 6   103  105  106  107  108  109    0
 21| 0 3    82   91  102    0    0    0    0  7 14  9 20
 22| 1 3   158  159  160    0    0    0    0
```

Responda agora:

Qual a altura da árvore B ?	Se o 156 for incluido, qual linha,coluna na matriz?	Se o 74 for incluído, qual linha,coluna na matriz?

CEP-UFPR-UTFPR-PUC/Pr-UP Sistemas de Informação 12/08/2024 - 11:35:52.4
Matemática aplicada Prof Dr P Kantek (pkantek@gmail.com)
Evolução de árvores: Árvore-B (e B+) VIVXq92a V: 1.02
76625 RAUL KNAPKI CUNHA
24CC2401 - 15 primeiro exercicio do T3 _____ / / _____

Árvore B

O conceito de árvore B foi desenvolvido por Rudolf Bayer e Edward McCreight em 1972.

Eles publicaram seu trabalho no artigo *"Organization and Maintenance of Large Ordered Indices"*, que descreve as propriedades e operações básicas das árvores B.

O trabalho de Bayer e McCreight teve um impacto significativo na área de ciência da computação, e as árvores B são agora amplamente utilizadas em bancos de dados e outros sistemas de software.

As árvores B são um tipo de árvore de busca balanceada que pode armazenar um grande número de chaves e valores de maneira eficiente. Elas são mais eficientes do que as árvores de busca binárias tradicionais, pois podem ter mais filhos por nó. As árvores B são usadas em uma variedade de aplicações, incluindo bancos de dados, sistemas de arquivos e sistemas de roteamento.

Eles não explicaram o significado do nome, só nos resta especular que esse B seja de Balanceada. Se você se lembrar das árvores binárias de pesquisa, deve recordar o problema que existe quando as chaves vem (quasi) ordenadas na entrada: a árvore gerada é desbalanceada e pode chegar a ter um desempenho terrível.

Outra qualidade da árvore-B é que ela tem grau k (k filhos por nodo). Em geral este valor é escolhido de modo a otimizar a blocagem de dados no arquivo de memória secundária. De qualquer modo este k modifica o desempenho, que lembrando é proporcional a $\log_k n$, onde n é a quantidade de chaves.

Como toda árvore, ela tem uma raiz que tem de 0 a $2k-1$ chaves e consequentemente tem de 0 a $2k$ filhos. Cada um dos filhos, tem de $k-1$ até $2k-1$ chaves de 0 até $2k$ filhos. Note que a raiz é o único nodo que sofre processamento especial, necessário para quando o arquivo tem uma única chave (por exemplo).

Um caso prático real Suponha um colégio com milhares de alunos, que têm seus dados guardados em um banco de dados. Por hipótese, os dados de cada aluno vão sendo recolhidos quando eles fazem a matrícula, pelo que a ordem física dos dados no arquivo é a data da matrícula. Parece óbvio assinalar que um determinado conjunto de dados (um arquivo) só pode ter uma única ordem física em um determinado instante. Mas nada impede que existam muitas ordens lógicas diferentes para o mesmo arquivo. Tais ordens lógicas são providenciadas por estruturas muito parecidas a uma árvore-b. Neste exemplo, poderíamos ter índices por nome (ordem alfabética), curso, turma, endereço etc.

Vai-se descrever aqui uma árvore-b que dará acesso aos dados por código do aluno.

Suponha uma árvore-b de nome M que providenciará este acesso. Vamos descrevê-la com $k=4$, que é muito pouco para aplicações reais, mas fica razoável para trabalhar com este exemplo. De acordo com a definição, os nodos de M têm entre 3 e 7 códigos de aluno. Para esta dimensão, o nodo de M terá 17 colunas:

coluna 0 um asterisco nesta posição indica que este é o nodo raiz da árvore. Conta como coluna 0 porque na verdade esta informação não está fisicamente aqui, está em outro lugar.

coluna 1 indicativo de folha (1=sim; 0=não)

coluna 2 quantidade de chaves válidas neste nodo (nem sempre o valor inválido é zero)

cols 3 a 9 chaves, sempre em ordem crescente

cols 10 a 17 endereços dos filhos associados a cada chave. Só estão preenchidas para nodos não folha.

Veja um exemplo:

```
    1 2    3    4    5    6    7    8    9 10 11 12 13 14 15 16 17
  +-------------------------------------------------------------------
 1| 1 5    3    6    7    8   10    0    0
 2| 0 4   11   20   32   44    0    0    0  1 20 15  7 12
 3| 1 4  128  131  132  134    0    0    0
 4| 1 5  149  150  151  153  160    0    0
 5| 1 6   65   73   76   79   80   83    0
 6| 1 5  182  183  188  189  191    0    0
 7| 1 3   34   35   37    0    0    0    0
 8| 1 4  106  109  111  114    0    0    0
 9| 1 6  136  137  138  140  143  145    0
*10| 0 2   64  127    0    0    0    0    0  2 17 11
 11| 0 6  135  146  162  171  181  192    0  3  9  4 14 21  6 18
 12| 1 7   46   49   50   55   56   57   63
 13| 1 4   88   89   91   94    0    0    0
 14| 1 5  164  166  167  168  169    0    0
 15| 1 6   21   22   23   28   29   30    0
 16| 1 7  117  118  120  122  123  125  126
 17| 0 4   84   95  105  116    0    0    0  5 13 19  8 16
 18| 1 5  193  195  197  199  200    0    0
 19| 1 5   96   98  100  102  103    0    0
 20| 1 4   14   15   17   19    0    0    0
 21| 1 3  174  175  177    0    0    0    0
```

Antes de continuar no exercício, você deve tentar entender o funcionamento da árvore-b.

- Desenhe a árvore
- identifique a necessidade da coluna 1. Idem para a coluna 2
- Qual a altura da árvore ?
- Quando e como a altura da árvore cresce ?

Agoram suponha que o elemento 187 é incluído. Qual a linha;coluna da matriz acima que é modificada, ou seja onde o elemento recém incluido entra ? Localize-o e veja:

```
    1 2    3    4    5    6    7    8    9 10 11 12 13 14 15 16 17
  +-------------------------------------------------------------------
 1| 1 5    3    6    7    8   10    0    0
 2| 0 4   11   20   32   44    0    0    0  1 20 15  7 12
 3| 1 4  128  131  132  134    0    0    0
 4| 1 5  149  150  151  153  160    0    0
 5| 1 6   65   73   76   79   80   83    0
 6| 1 6  182  183(187)188  189  191    0
 7| 1 3   34   35   37    0    0    0    0
 8| 1 4  106  109  111  114    0    0    0
 9| 1 6  136  137  138  140  143  145    0
*10| 0 2   64  127    0    0    0    0    0  2 17 11
 11| 0 6  135  146  162  171  181  192    0  3  9  4 14 21  6 18
 12| 1 7   46   49   50   55   56   57   63
 13| 1 4   88   89   91   94    0    0    0
 14| 1 5  164  166  167  168  169    0    0
 15| 1 6   21   22   23   28   29   30    0
 16| 1 7  117  118  120  122  123  125  126
 17| 0 4   84   95  105  116    0    0    0  5 13 19  8 16
 18| 1 5  193  195  197  199  200    0    0
 19| 1 5   96   98  100  102  103    0    0
 20| 1 4   14   15   17   19    0    0    0
 21| 1 3  174  175  177    0    0    0    0
```

Como voce pode ver ele está na linha ____ e coluna ____. Algumas características notáveis da árvore-b

- Figurinha carimbada no mundo do software: usam B-trees: SYBASE e do Harbour (antigo Clipper). Também são usados no Oracle, MySQL, PostgreSQL, Microsoft SQL e DB2, da IBM.
- muitas estruturas usam a árvore-b+. O + significa que há uma ligação adicional entre as folhas, para permitir o processamento sequencial dos dados.
- Existem nodos folha e não-folha. Os endereços dos filhos (colunas 10 a 17) só existem para os nodos que os têm (nodos não folhas).
- Os códigos de aluno (colunas 3 a 9 não mostram, para diminuir o ruído), mas ao lado de cada código está o endereço físico real dos dados deste aluno, que afinal é o dado que nos interessa nesta busca.
- Os códigos (colunas 3 a 9) estão em ordem crescente. Isto é fundamental e não pode ser modificado.
- Perceba como, em média, apenas a metade dos códigos está preenchida. Isto é necessário para permitir inclusão de novos dados. Sinaliza também um certo desperdício (overhead) de área em disco que qualquer método de acesso consome.
- Quando um nodo tem todas as chaves preenchidas e chega uma nova, que deveria (deverá) entrar nele, o nodo sofre um processo chamado split e ele é transformado em 2 nodos (cada um com metade dos espaços vazios). Note que o número máximo de chaves em qualquer nodo é sempre ímpar ($2n-1$) o que implica em que o elemento do meio sobe para o *pai* do nodo que está sendo quebrado. A metade anterior vai para um nodo *splitado* e a metade posterior vai para o outro nodo *splitado*.
- Quando as chaves vão sendo excluídas, e um nodo passa a ter menos do que a metade das chaves (o que é proibido) ele é juntado com um nodo irmão e ambos passam a ser um único nodo. Este processo recebe o nome de merge. Não é tão simples assim, há diversos casos particulares, mas a ideia é essa.

Mais um exemplo. Suponha a inclusão do número 26. Em que linha;coluna ele entra ?

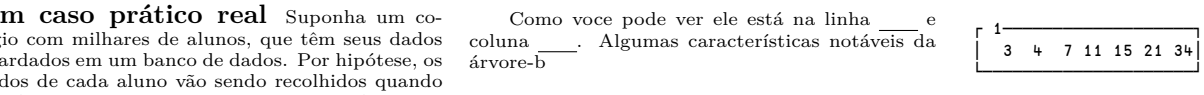
Algoritmo de Inclusão em árvore B

```
1: função INCLUA-BTREE (K:inteiro)
2: X ← raiz da árvore B
3: enquanto nodo X não é folha
4:     X ← filho correto onde K deve ser incluído
5: fim{enquanto}
6: se nodo X está cheio
7:     faça split do nodo X
8:     INCLUA-BTREE (K)
9:     abandone
10: senão
11:     inclua K no nodo X mantendo as chaves ordenadas
12: fim{se}
```

Algoritmos Estão disseminados pela comunidade. Deixo de apresentá-los aqui, pois são bastante específicos, complexos até, e mais importante do que implementá-los é entender-lhes o funcionamento. Até porque se você tiver que usar uma árvore b que não seja para aprender como ela funciona, recomendo fortemente que você use pacotes prontos, já testados e confiáveis.

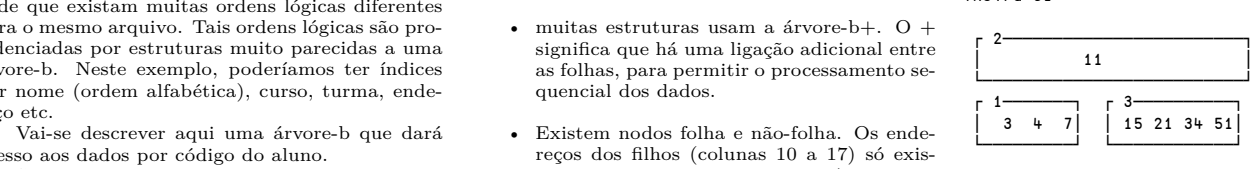
Um exemplo, passo a passo Seja uma árvore-b com $k=4$. Pela definição os nodos terão entre 3 e 7 chaves.

```
criabtree
insira 4, 7, 21, 11, 34, 15, 3
```

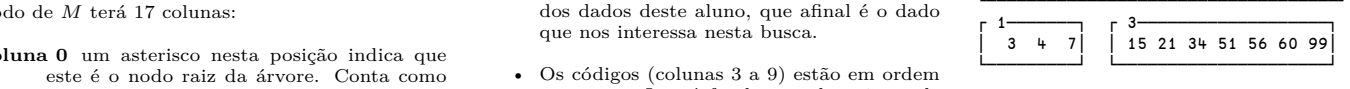


Agora, vamos inserir uma nova chave, o que vai provocar o split

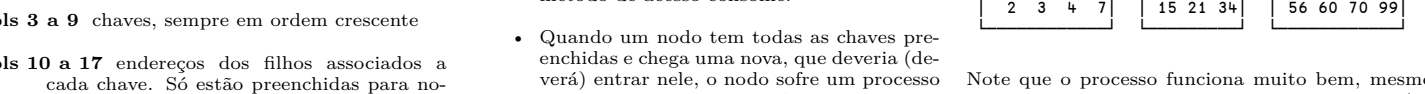
```
insira 51
```



Inserindo 60, 99, 56, fica



Inserindo 2 e 70, o 2 vai na boa para o nodo 1 enquanto o 70 força o split do nodo 3 e fica



Note que o processo funciona muito bem, mesmo que as chaves de entrada estejam desordenadas (ou em ordem aleatória). Isso é garantido pelos espaços em branco que são reservados nos nodos.

Um caso quasi-real: SYBASE

O SYBASE usa árvores B para indexar as tabelas. Existem 2 tipos de árvores B: as granuladas e as não granuladas (clustered). As granuladas são árvores esparsas (nem todo registro de dados tem entrada nos índices) nas quais os registros (na tabela original) são mantidos em ordem dos valores de índice e apenas o primeiro registro em cada página de dados tem uma entrada de índice. Árvores não granuladas são densas e cada um dos registros da tabela tem 1 entrada de índice.

Uma tabela SYBASE com índice granulado exige que seus elementos sejam mantidos em ordem dentro da tabela. Assim, inclusões podem se dar em qualquer página, a depender do conteúdo do campo de índice. Tabelas SYBASE que tem apenas índices não granulados fazem as inclusões de novos elementos sempre ao final das tabelas (na última página).

Tabelas SYBASE podem ter apenas 1 índice granulado, mas podem ter até 250 índices não granulados.

Na criação de um índice granulado, os dados da tabela devem estar colocados nas páginas em ordem.

Cada entrada de índice em sybase pode ser composta por até 16 campos e ter até 256 bytes. Estes campos não precisam estar contiguos. Só não podem ser partes de campos e resultados de cálculos.

O tamanho da página varia, mas o tamanho mais comum é 2K. Existem 5 tipos de páginas em sybase, a saber:

Páginas de Dados contém registros de dados ou registros de log. Ambos são estruturalmente diferentes, mas são construídos do mesmo jeito.

Páginas de índices contém registros de índice

Páginas de texto ou imagem contém BLOBS

Páginas de alocação contém estruturas de dados usadas para gerenciar o processo de alocacão de páginas

páginas de Estatísticas contém estatística de distribuição e uso para os índices.

Página de Dados Se os registros tem tamanho fixo, não haverá tabela de deslocamento na página e o endereço de cada registro é obtido mediante simples cálculo. Neste caso, o tamanho do registro (fixo) é guardado no campo TAMANHO MÍNIMO DE REGISTRO do header.

O header da página é um conjunto de 32 bytes contendo (entre outros): número lógico da página, anterior e próxima página lógica, identificação do objeto ao qual esta página pertence, primeira entrada de registro disponível nesta página, deslocamento do espaço livre, tamanho mínimo do registro

Depois vem a área de registros, que contém um número inteiro de slots para guardar registros. Um registro não atravessa limites de páginas

Tabela de deslocamento: Um conjunto de deslocamentos desde o início da página dando o início de cada um dos registros. Este conjunto cresce do fim da página para o início e só existe quando o tamanho do registro é variável.

Cada registro na página recebe um número de registro. É um campo de 1 byte e assim restringe-se o número de registros em uma página a 256. O número do registro usado nos índices não granulados (densos) é uma combinação do número da página com o número do registro na página.

O registro pode ocupar a página inteira (2048-32=2016). Não há tamanho mínimo para o registro, mas só pode haver 256 registros na página.

Os índices SYBASE são organizados na forma de arvores B + (o + significa que há uma ligação adicional entre as folhas, para permitir o processamento sequencial dos dados).

No exemplo aqui estudado, vão-se fazer algumas considerações/simplificações descritas a seguir.

Suponha um arquivo de pessoas contendo nome, cor preferida, bicho de estimação, alimento que gosta e idade.

Construindo um arquivo SYBASE contendo estes dados e que esteja sujeito às seguintes características:

- Registros de tamanho fixo igual a 40 bytes (9+9+9+9+4).
- Um índice granulado ou esparso (clustered) por um dos campos. No exemplo que é igual para todos, o índice esparso é por cor.
- Quatro índices não granulados (densos) pelos demais atributos.
- Páginas de dados com espaço para 4 registros e fator de preenchimento de 4:1 (usar apenas 3 espaços na carga original).
- Páginas de índices com espaço para 5 chaves (nome, cor, bicho, alimento ou idade). Lembrar que este número é impar, o que indica que no *split* vai subir a terceira chave (das 5 que existem).
- A identificação de cada registro deve ser página:número do registro na página. Por exemplo, a identificação 3:2, indica o segundo registro da terceira página.
- Nas páginas de índice (árvores-B) despreze as informações de endereço dos filhos. No exemplo, elas são irrelevantes. Faça como no exemplo.
- Os dados de entrada estão ordenados pela coluna do índice granulado.

Exemplo: Seja o conjunto de dados, no qual se fará o granulado por cor:

NOME	COR	ANIMAL	COMIDA	id.
josé	dourado	chopim	feijão	34
gabi	laranja	perú	aipim	22
sueli	lilas	gato	alho	40
antônio	musgo	vaca	batata	41
tatiana	petróleo	cavalo	arroz	37
juca	preto	águia	goiaba	36
romeu	roxo	dragão	maracujá	27
paulo	verde	urubú	rabanete	20

Com este conjunto de dados, ter-se-á a seguintes disposição nos arquivos

Páginas de Dados					
Pág1	josé	dourado	chopim	feijão	34
	gabi	laranja	perú	aipim	22
	sueli	lilas	gato	alho	40
Pág2	antônio	musgo	vaca	batata	41
	tatiana	petróleo	cavalo	arroz	37
	juca	preto	águia	goiaba	36
Pág3	romeu	roxo	dragão	maracujá	27
	paulo	verde	urubú	rabanete	20

A seguir, as páginas de índices

Páginas de índices	
Índice I1	granulado por COR
Página 1 ←raiz	dourado(1:1), musgo(2:1), roxo(3:1)
Índice I2	por NOME
Página 1	antônio(2:1), gabi(1:2)
Página 2 ←raiz	jose(1:1)
Página 3	juca(2:3), paulo(3:2), romeu(3:1), sueli(1:3), tatiana(2:2)
Índice I3	por BICHO
P1	águia(2:3), cavalo(2:2), chopim(1:1), dragão(3:1)
P2 ←raiz	gato(1:3)
P3	perú(1:2),urubú(3:2), vaca(2:1)
Índice I4	por COMIDA
P1	aipim(1:2),alho(1:3)
P2 ←raiz	arroz(2:2)
P3	batata(2:1), feijão(1:1), goiaba(2:3), maracujá(3:1), rabanete(3:2)
Índice I5	por IDADE
P1	20(3:2), 22(1:2), 27(3:1), 34(1:1), 36(2:3)
P2 ←raiz	37(2:2)
P3	40(1:3), 41(2:1)

☞ Para você fazer

Desenhe a árvore B a seguir:

```
     1 2    3    4    5    6    7    8    9 10 11 12 13 14 15 16 17
   +-------------------------------------------------------------------
 1| 1 7    31   32   33   34   35   36   37
 2| 0 7    39   45   54   68   75   82   87   1   8 20   5   6 21 13 19
 3| 1 3   114 117 118    0    0    0    0
 4| 1 6   161 164 165 167 168 169    0
 5| 1 7    56   59   61   63   64   65   66
 6| 1 4    70   71   73   74    0    0    0
 7| 1 7    98   99 100 101 105 107 109
 8| 1 4    40   41   42   43    0    0    0
 9| 1 4   137 138 139 140    0    0    0
*10| 0 2    97 144    0    0    0    0    0   2 11 17
 11| 0 4   110 119 127 134    0    0    0   7   3 15 22   9
 12| 1 6   189 190 191 194 195 200    0
 13| 1 3    84   85   86    0    0    0    0
 14| 1 5   146 147 149 151 152    0    0
 15| 1 4   121 123 125 126    0    0    0
 16| 1 6   178 180 181 182 183 184    0
 17| 0 4   153 158 176 187    0    0    0  14 18   4 16 12
 18| 1 3   154 155 156    0    0    0    0
 19| 1 4    88   90   91   92    0    0    0
 20| 1 4    46   47   49   51    0    0    0
 21| 1 3    76   78   80    0    0    0    0
 22| 1 3   129 131 133    0    0    0    0
```

Responda agora:

Qual a altura da árvore B ?	Se o 130 for incluido, qual linha,coluna na matriz?	Se o 159 for incluído, qual linha,coluna na matriz?

Mais uma Instância do mesmo exercício Desenhe a árvore B a seguir:

```
     1 2    3    4    5    6    7    8    9 10 11 12 13 14 15 16 17
   +-------------------------------------------------------------------
 1| 1 7    31   32   34   35   37   40   43
 2| 0 4    44   55   62   67    0    0    0   1 13 22   9 18
 3| 1 7    73   75   76   77   78   79   80
 4| 1 5   140 144 147 149 153    0    0
 5| 1 6   117 118 121 122 124 125    0
 6| 1 4   162 164 165 169    0    0    0
 7| 1 7    91   92   95   98   99 108 109
 8| 1 6   177 178 180 182 183 187    0
 9| 1 3    63   64   66    0    0    0    0
*10| 0 3    72 115 161    0    0    0    0   2 21 11 19
 11| 0 3   129 139 154    0    0    0    0   5 12   4 16
 12| 1 4   130 131 133 138    0    0    0
 13| 1 5    45   46   48   52   53    0    0
 14| 1 6   190 191 193 194 195 198    0
 15| 1 4   171 172 174 175    0    0    0
 16| 1 4   155 156 159 160    0    0    0
 17| 1 5    82   83   85   87   88    0    0
 18| 1 4    68   69   70   71    0    0    0
 19| 0 3   170 176 188    0    0    0    0   6 15   8 14
 20| 1 3   111 112 114    0    0    0    0
 21| 0 3    81   90 110    0    0    0    0   3 17   7 20
 22| 1 4    57   58   59   60    0    0    0
```

Responda agora:

Qual a altura da árvore B ?	Se o 123 for incluido, qual linha,coluna na matriz?	Se o 106 for incluído, qual linha,coluna na matriz?

CEP-UFPR-UTFPR-PUC/Pr-UP Sistemas de Informação 12/08/2024 - 11:35:52.4
Matemática aplicada Prof Dr P Kantek (pkantek@gmail.com)
Evolução de árvores: Árvore-B (e B+) VIVXq92a V: 1.02
76632 VICTOR BRONDANI DOS SANTOS
24CC2401 - 16 primeiro exercicio do T3 _____ / / _____

Árvore B

O conceito de árvore B foi desenvolvido por Rudolf Bayer e Edward McCreight em 1972.

Eles publicaram seu trabalho no artigo *"Organization and Maintenance of Large Ordered Indices"*, que descreve as propriedades e operações básicas das árvores B.

O trabalho de Bayer e McCreight teve um impacto significativo na área de ciência da computação, e as árvores B são agora amplamente utilizadas em bancos de dados e outros sistemas de software.

As árvores B são um tipo de árvore de busca balanceada que pode armazenar um grande número de chaves e valores de maneira eficiente. Elas são mais eficientes do que as árvores de busca binárias tradicionais, pois podem ter mais filhos por nó. As árvores B são usadas em uma variedade de aplicações, incluindo bancos de dados, sistemas de arquivos e sistemas de roteamento.

Eles não explicaram o significado do nome, só nos resta especular que esse B seja de Balanceada. Se você se lembrar das árvores binárias de pesquisa, deve recordar o problema que existe quando as chaves vem (quasi) ordenadas na entrada: a árvore gerada é desbalanceada e pode chegar a ter um desempenho terrível.

Outra qualidade da árvore-B é que ela tem grau k (k filhos por nodo). Em geral este valor é escolhido de modo a otimizar a blocagem de dados no arquivo de memória secundária. De qualquer modo este k modifica o desempenho, que lembrando é proporcional a $\log_k n$, onde n é a quantidade de chaves.

Como toda árvore, ela tem uma raiz que tem de 0 a $2k-1$ chaves e consequentemente tem de 0 a $2k$ filhos. Cada um dos filhos, tem de $k-1$ até $2k-1$ chaves de 0 até $2k$ filhos. Note que a raiz é o único nodo que sofre processamento especial, necessário para quando o arquivo tem uma única chave (por exemplo).

Um caso prático real Suponha um colégio com milhares de alunos, que têm seus dados guardados em um banco de dados. Por hipótese, os dados de cada aluno vão sendo recolhidos quando eles fazem a matrícula, pelo que a ordem física dos dados no arquivo é a data da matrícula. Parece óbvio assinalar que um determinado conjunto de dados (um arquivo) só pode ter uma única ordem física em um determinado instante. Mas nada impede que existam muitas ordens lógicas diferentes para o mesmo arquivo. Tais ordens lógicas são providenciadas por estruturas muito parecidas a uma árvore-b. Neste exemplo, poderíamos ter índices por nome (ordem alfabética), curso, turma, endereço etc.

Vai-se descrever aqui uma árvore-b que dará acesso aos dados por código do aluno.

Suponha uma árvore-b de nome M que providenciará este acesso. Vamos descrevê-la com $k=4$, que é muito pouco para aplicações reais, mas fica razoável para trabalhar com este exemplo. De acordo com a definição, os nodos de M têm entre 3 e 7 códigos de aluno. Para esta dimensão, o nodo de M terá 17 colunas:

coluna 0 um asterisco nesta posição indica que este é o nodo raiz da árvore. Conta como coluna 0 porque na verdade esta informação não está fisicamente aqui, está em outro lugar.

coluna 1 indicativo de folha (1=sim; 0=não)

coluna 2 quantidade de chaves válidas neste nodo (nem sempre o valor inválido é zero)

cols 3 a 9 chaves, sempre em ordem crescente

cols 10 a 17 endereços dos filhos associados a cada chave. Só estão preenchidas para nodos não folha.

Veja um exemplo:

```
    1 2    3    4    5    6    7    8    9 10 11 12 13 14 15 16 17
  +-------------------------------------------------------------------
 1| 1 5    3    6    7    8   10    0    0
 2| 0 4   11   20   32   44    0    0    0  1 20 15  7 12
 3| 1 4  128  131  132  134    0    0    0
 4| 1 5  149  150  151  153  160    0    0
 5| 1 6   65   73   76   79   80   83    0
 6| 1 5  182  183  188  189  191    0    0
 7| 1 3   34   35   37    0    0    0    0
 8| 1 4  106  109  111  114    0    0    0
 9| 1 6  136  137  138  140  143  145    0
*10| 0 2   64  127    0    0    0    0    0  2 17 11
 11| 0 6  135  146  162  171  181  192    0  3  9  4 14 21  6 18
 12| 1 7   46   49   50   55   56   57   63
 13| 1 4   88   89   91   94    0    0    0
 14| 1 5  164  166  167  168  169    0    0
 15| 1 6   21   22   23   28   29   30    0
 16| 1 7  117  118  120  122  123  125  126
 17| 0 4   84   95  105  116    0    0    0  5 13 19  8 16
 18| 1 5  193  195  197  199  200    0    0
 19| 1 5   96   98  100  102  103    0    0
 20| 1 4   14   15   17   19    0    0    0
 21| 1 3  174  175  177    0    0    0    0
```

Antes de continuar no exercício, você deve tentar entender o funcionamento da árvore-b.

- Desenhe a árvore
- identifique a necessidade da coluna 1. Idem para a coluna 2
- Qual a altura da árvore ?
- Quando e como a altura da árvore cresce ?

Agoram suponha que o elemento 187 é incluído. Qual a linha;coluna da matriz acima que é modificada, ou seja onde o elemento recém incluido entra ? Localize-o e veja:

```
    1 2    3    4    5    6    7    8    9 10 11 12 13 14 15 16 17
  +-------------------------------------------------------------------
 1| 1 5    3    6    7    8   10    0    0
 2| 0 4   11   20   32   44    0    0    0  1 20 15  7 12
 3| 1 4  128  131  132  134    0    0    0
 4| 1 5  149  150  151  153  160    0    0
 5| 1 6   65   73   76   79   80   83    0
 6| 1 6  182  183(187)188  189  191    0
 7| 1 3   34   35   37    0    0    0    0
 8| 1 4  106  109  111  114    0    0    0
 9| 1 6  136  137  138  140  143  145    0
*10| 0 2   64  127    0    0    0    0    0  2 17 11
 11| 0 6  135  146  162  171  181  192    0  3  9  4 14 21  6 18
 12| 1 7   46   49   50   55   56   57   63
 13| 1 4   88   89   91   94    0    0    0
 14| 1 5  164  166  167  168  169    0    0
 15| 1 6   21   22   23   28   29   30    0
 16| 1 7  117  118  120  122  123  125  126
 17| 0 4   84   95  105  116    0    0    0  5 13 19  8 16
 18| 1 5  193  195  197  199  200    0    0
 19| 1 5   96   98  100  102  103    0    0
 20| 1 4   14   15   17   19    0    0    0
 21| 1 3  174  175  177    0    0    0    0
```

Como voce pode ver ele está na linha ____ e coluna ____. Algumas características notáveis da árvore-b

- Figurinha carimbada no mundo do software: usam B-trees: SYBASE e do Harbour (antigo Clipper). Também são usados no Oracle, MySQL, PostgreSQL, Microsoft SQL e DB2, da IBM.
- muitas estruturas usam a árvore-b+. O + significa que há uma ligação adicional entre as folhas, para permitir o processamento sequencial dos dados.
- Existem nodos folha e não-folha. Os endereços dos filhos (colunas 10 a 17) só existem para os nodos que os têm (nodos não folhas).
- Os códigos de aluno (colunas 3 a 9 não mostram, para diminuir o ruído), mas ao lado de cada código está o endereço físico real dos dados deste aluno, que afinal é o dado que nos interessa nesta busca.
- Os códigos (colunas 3 a 9) estão em ordem crescente. Isto é fundamental e não pode ser modificado.
- Perceba como, em média, apenas a metade dos códigos está preenchida. Isto é necessário para permitir inclusão de novos dados. Sinaliza também um certo desperdício (overhead) de área em disco que qualquer método de acesso consome.
- Quando um nodo tem todas as chaves preenchidas e chega uma nova, que deveria (deverá) entrar nele, o nodo sofre um processo chamado split e ele é transformado em 2 nodos (cada um com metade dos espaços vazios). Note que o número máximo de chaves em qualquer nodo é sempre ímpar ($2n-1$) o que implica em que o elemento do meio sobe para o *pai* do nodo que está sendo quebrado. A metade anterior vai para um nodo *splitado* e a metade posterior vai para o outro nodo *splitado*.
- Quando as chaves vão sendo excluídas, e um nodo passa a ter menos do que a metade das chaves (o que é proibido) ele é juntado com um nodo irmão e ambos passam a ser um único nodo. Este processo recebe o nome de merge. Não é tão simples assim, há diversos casos particulares, mas a ideia é essa.

Mais um exemplo. Suponha a inclusão do número 26. Em que linha;coluna ele entra ?

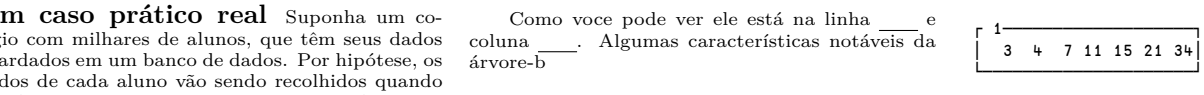
Algoritmo de Inclusão em árvore B

```
1: função INCLUA-BTREE (K:inteiro)
2: X ← raiz da árvore B
3: enquanto nodo X não é folha
4:     X ← filho correto onde K deve ser incluído
5: fim{enquanto}
6: se nodo X está cheio
7:     faça split do nodo X
8:     INCLUA-BTREE (K)
9:     abandone
10: senão
11:     inclua K no nodo X mantendo as chaves ordenadas
12: fim{se}
```

Algoritmos Estão disseminados pela comunidade. Deixo de apresentá-los aqui, pois são bastante específicos, complexos até, e mais importante do que implementá-los é entender-lhes o funcionamento. Até porque se você tiver que usar uma árvore b que não seja para aprender como ela funciona, recomendo fortemente que você use pacotes prontos, já testados e confiáveis.

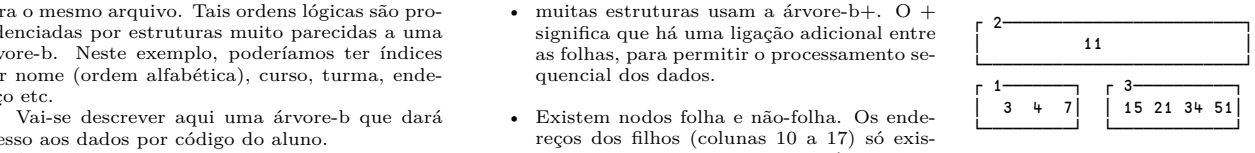
Um exemplo, passo a passo Seja uma árvore-b com $k=4$. Pela definição os nodos terão entre 3 e 7 chaves.

```
criabtree
insira 4, 7, 21, 11, 34, 15, 3
```

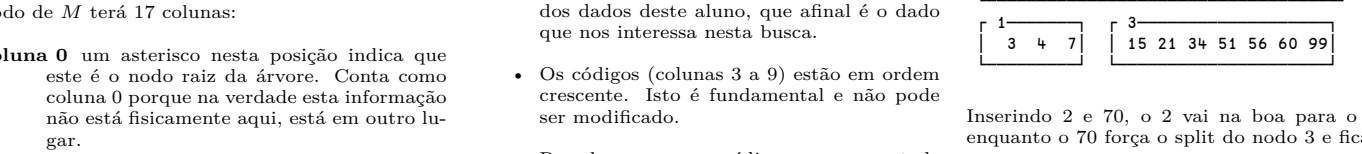


Agora, vamos inserir uma nova chave, o que vai provocar o split

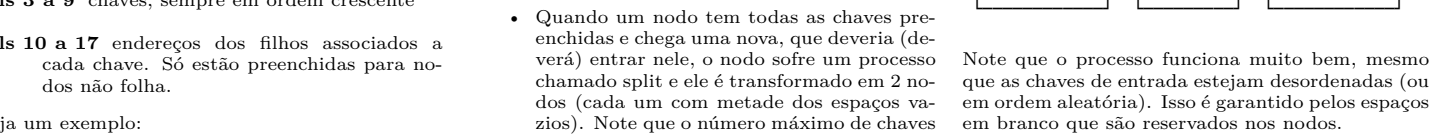
```
insira 51
```



Inserindo 60, 99, 56, fica



Inserindo 2 e 70, o 2 vai na boa para o nodo 1 enquanto o 70 força o split do nodo 3 e fica



Note que o processo funciona muito bem, mesmo que as chaves de entrada estejam desordenadas (ou em ordem aleatória). Isso é garantido pelos espaços em branco que são reservados nos nodos.

Um caso quasi-real: SYBASE

O SYBASE usa árvores B para indexar as tabelas. Existem 2 tipos de árvores B: as granuladas e as não granuladas (clustered). As granuladas são árvores esparsas (nem todo registro de dados tem entrada nos índices) nas quais os registros (na tabela original) são mantidos em ordem dos valores de índice e apenas o primeiro registro em cada página de dados tem uma entrada de índice. Árvores não granuladas são densas e cada um dos registros da tabela tem 1 entrada de índice.

Uma tabela SYBASE com índice granulado exige que seus elementos sejam mantidos em ordem dentro da tabela. Assim, inclusões podem se dar em qualquer página, a depender do conteúdo do campo de índice. Tabelas SYBASE que tem apenas índices não granulados fazem as inclusões de novos elementos sempre ao final das tabelas (na última página).

Tabelas SYBASE podem ter apenas 1 índice granulado, mas podem ter até 250 índices não granulados.

Na criação de um índice granulado, os dados da tabela devem estar colocados nas páginas em ordem.

Cada entrada de índice em sybase pode ser composta por até 16 campos e ter até 256 bytes. Estes campos não precisam estar contiguos. Só não podem ser partes de campos e resultados de cálculos.

O tamanho da página varia, mas o tamanho mais comum é 2K. Existem 5 tipos de páginas em sybase, a saber:

Páginas de Dados contém registros de dados ou registros de log. Ambos são estruturalmente diferentes, mas são construídos do mesmo jeito.

Páginas de índices contém registros de índice

Páginas de texto ou imagem contém BLOBS

Páginas de alocação contém estruturas de dados usadas para gerenciar o processo de alocacão de páginas

páginas de Estatísticas contém estatística de distribuição e uso para os índices.

Página de Dados Se os registros tem tamanho fixo, não haverá tabela de deslocamento na página e o endereço de cada registro é obtido mediante simples cálculo. Neste caso, o tamanho do registro (fixo) é guardado no campo TAMANHO MÍNIMO DE REGISTRO do header.

O header da página é um conjunto de 32 bytes contendo (entre outros): número lógico da página, anterior e próxima página lógica, identificação do objeto ao qual esta página pertence, primeira entrada de registro disponível nesta página, deslocamento do espaço livre, tamanho mínimo do registro

Depois vem a área de registros, que contém um número inteiro de slots para guardar registros. Um registro não atravessa limites de páginas

Tabela de deslocamento: Um conjunto de deslocamentos desde o início da página dando o início de cada um dos registros. Este conjunto cresce do fim da página para o início e só existe quando o tamanho do registro é variável.

Cada registro na página recebe um número de registro. É um campo de 1 byte e assim restringe-se o número de registros em uma página a 256. O número do registro usado nos índices não granulados (densos) é uma combinação do número da página com o número do registro na página.

O registro pode ocupar a página inteira (2048-32=2016). Não há tamanho mínimo para o registro, mas só pode haver 256 registros na página.

Os índices SYBASE são organizados na forma de arvores B + (o + significa que há uma ligação adicional entre as folhas, para permitir o processamento sequencial dos dados).

No exemplo aqui estudado, vão-se fazer algumas considerações/simplificações descritas a seguir.

Suponha um arquivo de pessoas contendo nome, cor preferida, bicho de estimação, alimento que gosta e idade.

Construindo um arquivo SYBASE contendo estes dados e que esteja sujeito às seguintes características:

- Registros de tamanho fixo igual a 40 bytes (9+9+9+9+4).
- Um índice granulado ou esparso (clustered) por um dos campos. No exemplo que é igual para todos, o índice esparso é por cor.
- Quatro índices não granulados (densos) pelos demais atributos.
- Páginas de dados com espaço para 4 registros e fator de preenchimento de 4:1 (usar apenas 3 espaços na carga original).
- Páginas de índices com espaço para 5 chaves (nome, cor, bicho, alimento ou idade). Lembrar que este número é impar, o que indica que no *split* vai subir a terceira chave (das 5 que existem).
- A identificação de cada registro deve ser página:número do registro na página. Por exemplo, a identificação 3:2, indica o segundo registro da terceira página.
- Nas páginas de índice (árvores-B) despreze as informações de endereço dos filhos. No exemplo, elas são irrelevantes. Faça como no exemplo.
- Os dados de entrada estão ordenados pela coluna do índice granulado.

Exemplo: Seja o conjunto de dados, no qual se fará o granulado por cor:

NOME	COR	ANIMAL	COMIDA	id.
josé	dourado	chopim	feijão	34
gabi	laranja	perú	aipim	22
sueli	lilas	gato	alho	40
antônio	musgo	vaca	batata	41
tatiana	petróleo	cavalo	arroz	37
juca	preto	águia	goiaba	36
romeu	roxo	dragão	maracujá	27
paulo	verde	urubú	rabanete	20

Com este conjunto de dados, ter-se-á a seguintes disposição nos arquivos

Páginas de Dados					
Pág1	josé	dourado	chopim	feijão	34
	gabi	laranja	perú	aipim	22
	sueli	lilas	gato	alho	40
Pág2	antônio	musgo	vaca	batata	41
	tatiana	petróleo	cavalo	arroz	37
	juca	preto	águia	goiaba	36
Pág3	romeu	roxo	dragão	maracujá	27
	paulo	verde	urubú	rabanete	20

A seguir, as páginas de índices

Páginas de índices	
Índice I1	granulado por COR
Página 1 ←raiz	dourado(1:1), musgo(2:1), roxo(3:1)
Índice I2	por NOME
Página 1	antônio(2:1), gabi(1:2)
Página 2 ←raiz	jose(1:1)
Página 3	juca(2:3), paulo(3:2), romeu(3:1), sueli(1:3), tatiana(2:2)
Índice I3	por BICHO
P1	águia(2:3), cavalo(2:2), chopim(1:1), dragão(3:1)
P2 ←raiz	gato(1:3)
P3	perú(1:2),urubú(3:2), vaca(2:1)
Índice I4	por COMIDA
P1	aipim(1:2),alho(1:3)
P2 ←raiz	arroz(2:2)
P3	batata(2:1), feijão(1:1), goiaba(2:3), maracujá(3:1), rabanete(3:2)
Índice I5	por IDADE
P1	20(3:2), 22(1:2), 27(3:1), 34(1:1), 36(2:3)
P2 ←raiz	37(2:2)
P3	40(1:3), 41(2:1)

☞ Para você fazer

Desenhe a árvore B a seguir:

```
     1  2    3    4    5    6    7    8    9 10 11 12 13 14 15 16 17
   +-------------------------------------------------------------------
 1| 1  3   32   34   35    0    0    0    0
 2| 0  4   36   43   57   65    0    0    0  1 23  5 14  7
 3| 1  3   85   87   88    0    0    0    0
 4| 1  5  169  170  171  172  173    0    0
 5| 1  6   47   48   50   53   54   55    0
 6| 1  6  124  127  128  129  131  132    0
 7| 1  4   68   69   72   73    0    0    0
 8| 1  4  184  187  188  192    0    0    0
 9| 1  6  175  176  177  178  179  180    0
*10| 0  3   77  123  168    0    0    0    0  2 18 11 22
11| 0  3  134  145  158    0    0    0    0  6 19 15 13
12| 1  5  111  112  114  115  116    0    0
13| 1  3  161  166  167    0    0    0    0
14| 1  5   58   60   61   63   64    0    0
15| 1  6  147  148  152  153  154  157    0
16| 1  6   99  103  105  107  108  109    0
17| 1  3   81   82   83    0    0    0    0
18| 0  5   84   90   96  110  117    0    0 17  3 24 16 12 21
19| 1  3  137  140  144    0    0    0    0
20| 1  3  194  195  198    0    0    0    0
21| 1  3  118  119  122    0    0    0    0
22| 0  3  174  183  193    0    0    0    0  4  9  8 20
23| 1  4   37   38   39   40    0    0    0
24| 1  4   91   92   93   94    0    0    0
```

Responda agora:

Qual a altura da árvore B ?	Se o 141 for incluido, qual linha,coluna na matriz?	Se o 186 for incluído, qual linha,coluna na matriz?

Mais uma Instância do mesmo exercício Desenhe a árvore B a seguir:

```
     1  2    3    4    5    6    7    8    9 10 11 12 13 14 15 16 17
   +-------------------------------------------------------------------
 1| 1  4   31   32   33   34    0    0    0
 2| 0  6   35   42   65   78   87   93    0  1 20 12  3  8 14  5
 3| 1  7   66   70   71   72   74   75   77
 4| 1  3  132  135  136    0    0    0    0
 5| 1  7   95   96   99  100  101  102  104
 6| 1  3  178  180  181    0    0    0    0
 7| 1  3  147  148  149    0    0    0    0
 8| 1  4   79   81   83   84    0    0    0
 9| 1  6  106  107  108  109  113  116    0
*10| 0  2  105  163    0    0    0    0    0  2 11 18
11| 0  5  117  130  137  143  150    0    0  9 16  4 19  7 22
12| 1  7   43   49   53   54   56   59   61
13| 1  4  197  198  199  200    0    0    0
14| 1  4   88   89   91   92    0    0    0
15| 1  3  165  166  167    0    0    0    0
16| 1  4  118  120  122  128    0    0    0
17| 1  7  185  187  188  189  190  191  193
18| 0  4  169  177  184  194    0    0    0 15 21  6 17 13
19| 1  5  138  139  140  141  142    0    0
20| 1  3   36   40   41    0    0    0    0
21| 1  4  170  171  175  176    0    0    0
22| 1  5  151  152  154  155  161    0    0
```

Responda agora:

Qual a altura da árvore B ?	Se o 94 for incluido, qual linha,coluna na matriz?	Se o 124 for incluído, qual linha,coluna na matriz?

CEP-UFPR-UTFPR-PUC/Pr-UP Sistemas de Informação 12/08/2024 - 11:35:52.4
Matemática aplicada Prof Dr P Kantek (pkantek@gmail.com)
Evolução de árvores: Árvore-B (e B+) VIVXq92a V: 1.02
76649 VICTOR HUGO DOS SANTOS DE CAMA
24CC2401 - 17 primeiro exercicio do T3 _____ / / _____

Árvore B

O conceito de árvore B foi desenvolvido por Rudolf Bayer e Edward McCreight em 1972.

Eles publicaram seu trabalho no artigo *"Organization and Maintenance of Large Ordered Indices"*, que descreve as propriedades e operações básicas das árvores B.

O trabalho de Bayer e McCreight teve um impacto significativo na área de ciência da computação, e as árvores B são agora amplamente utilizadas em bancos de dados e outros sistemas de software.

As árvores B são um tipo de árvore de busca balanceada que pode armazenar um grande número de chaves e valores de maneira eficiente. Elas são mais eficientes do que as árvores de busca binárias tradicionais, pois podem ter mais filhos por nó. As árvores B são usadas em uma variedade de aplicações, incluindo bancos de dados, sistemas de arquivos e sistemas de roteamento.

Eles não explicaram o significado do nome, só nos resta especular que esse B seja de Balanceada. Se você se lembrar das árvores binárias de pesquisa, deve recordar o problema que existe quando as chaves vem (quasi) ordenadas na entrada: a árvore gerada é desbalanceada e pode chegar a ter um desempenho terrível.

Outra qualidade da árvore-B é que ela tem grau k (k filhos por nodo). Em geral este valor é escolhido de modo a otimizar a blocagem de dados no arquivo de memória secundária. De qualquer modo este k modifica o desempenho, que lembrando é proporcional a $\log_k n$, onde n é a quantidade de chaves.

Como toda árvore, ela tem uma raiz que tem de 0 a $2k-1$ chaves e consequentemente tem de 0 a $2k$ filhos. Cada um dos filhos, tem de $k-1$ até $2k-1$ chaves de 0 até $2k$ filhos. Note que a raiz é o único nodo que sofre processamento especial, necessário para quando o arquivo tem uma única chave (por exemplo).

Um caso prático real Suponha um colégio com milhares de alunos, que têm seus dados guardados em um banco de dados. Por hipótese, os dados de cada aluno vão sendo recolhidos quando eles fazem a matrícula, pelo que a ordem física dos dados no arquivo é a data da matrícula. Parece óbvio assinalar que um determinado conjunto de dados (um arquivo) só pode ter uma única ordem física em um determinado instante. Mas nada impede que existam muitas ordens lógicas diferentes para o mesmo arquivo. Tais ordens lógicas são providenciadas por estruturas muito parecidas a uma árvore-b. Neste exemplo, poderíamos ter índices por nome (ordem alfabética), curso, turma, endereço etc.

Vai-se descrever aqui uma árvore-b que dará acesso aos dados por código do aluno.

Suponha uma árvore-b de nome M que providenciará este acesso. Vamos descrevê-la com $k=4$, que é muito pouco para aplicações reais, mas fica razoável para trabalhar com este exemplo. De acordo com a definição, os nodos de M têm entre 3 e 7 códigos de aluno. Para esta dimensão, o nodo de M terá 17 colunas:

coluna 0 um asterisco nesta posição indica que este é o nodo raiz da árvore. Conta como coluna 0 porque na verdade esta informação não está fisicamente aqui, está em outro lugar.

coluna 1 indicativo de folha (1=sim; 0=não)

coluna 2 quantidade de chaves válidas neste nodo (nem sempre o valor inválido é zero)

cols 3 a 9 chaves, sempre em ordem crescente

cols 10 a 17 endereços dos filhos associados a cada chave. Só estão preenchidas para nodos não folha.

Veja um exemplo:

```
    1 2    3    4    5    6    7    8    9 10 11 12 13 14 15 16 17
  +-------------------------------------------------------------------
 1| 1 5    3    6    7    8   10    0    0
 2| 0 4   11   20   32   44    0    0    0  1 20 15  7 12
 3| 1 4  128  131  132  134    0    0    0
 4| 1 5  149  150  151  153  160    0    0
 5| 1 6   65   73   76   79   80   83    0
 6| 1 5  182  183  188  189  191    0    0
 7| 1 3   34   35   37    0    0    0    0
 8| 1 4  106  109  111  114    0    0    0
 9| 1 6  136  137  138  140  143  145    0
*10| 0 2   64  127    0    0    0    0    0  2 17 11
 11| 0 6  135  146  162  171  181  192    0  3  9  4 14 21  6 18
 12| 1 7   46   49   50   55   56   57   63
 13| 1 4   88   89   91   94    0    0    0
 14| 1 5  164  166  167  168  169    0    0
 15| 1 6   21   22   23   28   29   30    0
 16| 1 7  117  118  120  122  123  125  126
 17| 0 4   84   95  105  116    0    0    0  5 13 19  8 16
 18| 1 5  193  195  197  199  200    0    0
 19| 1 5   96   98  100  102  103    0    0
 20| 1 4   14   15   17   19    0    0    0
 21| 1 3  174  175  177    0    0    0    0
```

Antes de continuar no exercício, você deve tentar entender o funcionamento da árvore-b.

- Desenhe a árvore
- identifique a necessidade da coluna 1. Idem para a coluna 2
- Qual a altura da árvore ?
- Quando e como a altura da árvore cresce ?

Agoram suponha que o elemento 187 é incluído. Qual a linha;coluna da matriz acima que é modificada, ou seja onde o elemento recém incluido entra ? Localize-o e veja:

```
    1 2    3    4    5    6    7    8    9 10 11 12 13 14 15 16 17
  +-------------------------------------------------------------------
 1| 1 5    3    6    7    8   10    0    0
 2| 0 4   11   20   32   44    0    0    0  1 20 15  7 12
 3| 1 4  128  131  132  134    0    0    0
 4| 1 5  149  150  151  153  160    0    0
 5| 1 6   65   73   76   79   80   83    0
 6| 1 6  182  183(187)188  189  191    0
 7| 1 3   34   35   37    0    0    0    0
 8| 1 4  106  109  111  114    0    0    0
 9| 1 6  136  137  138  140  143  145    0
*10| 0 2   64  127    0    0    0    0    0  2 17 11
 11| 0 6  135  146  162  171  181  192    0  3  9  4 14 21  6 18
 12| 1 7   46   49   50   55   56   57   63
 13| 1 4   88   89   91   94    0    0    0
 14| 1 5  164  166  167  168  169    0    0
 15| 1 6   21   22   23   28   29   30    0
 16| 1 7  117  118  120  122  123  125  126
 17| 0 4   84   95  105  116    0    0    0  5 13 19  8 16
 18| 1 5  193  195  197  199  200    0    0
 19| 1 5   96   98  100  102  103    0    0
 20| 1 4   14   15   17   19    0    0    0
 21| 1 3  174  175  177    0    0    0    0
```

Como voce pode ver ele está na linha ____ e coluna ____. Algumas características notáveis da árvore-b

- Figurinha carimbada no mundo do software: usam B-trees: SYBASE e do Harbour (antigo Clipper). Também são usados no Oracle, MySQL, PostgreSQL, Microsoft SQL e DB2, da IBM.
- muitas estruturas usam a árvore-b+. O + significa que há uma ligação adicional entre as folhas, para permitir o processamento sequencial dos dados.
- Existem nodos folha e não-folha. Os endereços dos filhos (colunas 10 a 17) só existem para os nodos que os têm (nodos não folhas).
- Os códigos de aluno (colunas 3 a 9 não mostram, para diminuir o ruído), mas ao lado de cada código está o endereço físico real dos dados deste aluno, que afinal é o dado que nos interessa nesta busca.
- Os códigos (colunas 3 a 9) estão em ordem crescente. Isto é fundamental e não pode ser modificado.
- Perceba como, em média, apenas a metade dos códigos está preenchida. Isto é necessário para permitir inclusão de novos dados. Sinaliza também um certo desperdício (overhead) de área em disco que qualquer método de acesso consome.
- Quando um nodo tem todas as chaves preenchidas e chega uma nova, que deveria (deverá) entrar nele, o nodo sofre um processo chamado split e ele é transformado em 2 nodos (cada um com metade dos espaços vazios). Note que o número máximo de chaves em qualquer nodo é sempre ímpar ($2n-1$) o que implica em que o elemento do meio sobe para o *pai* do nodo que está sendo quebrado. A metade anterior vai para um nodo *splitado* e a metade posterior vai para o outro nodo *splitado*.
- Quando as chaves vão sendo excluídas, e um nodo passa a ter menos do que a metade das chaves (o que é proibido) ele é juntado com um nodo irmão e ambos passam a ser um único nodo. Este processo recebe o nome de merge. Não é tão simples assim, há diversos casos particulares, mas a ideia é essa.

Mais um exemplo. Suponha a inclusão do número 26. Em que linha;coluna ele entra ?

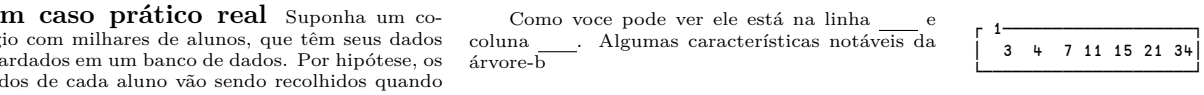
Algoritmo de Inclusão em árvore B

```
1: função INCLUA-BTREE (K:inteiro)
2: X ← raiz da árvore B
3: enquanto nodo X não é folha
4:     X ← filho correto onde K deve ser incluído
5: fim{enquanto}
6: se nodo X está cheio
7:     faça split do nodo X
8:     INCLUA-BTREE (K)
9:     abandone
10: senão
11:     inclua K no nodo X mantendo as chaves ordenadas
12: fim{se}
```

Algoritmos Estão disseminados pela comunidade. Deixo de apresentá-los aqui, pois são bastante específicos, complexos até, e mais importante do que implementá-los é entender-lhes o funcionamento. Até porque se você tiver que usar uma árvore b que não seja para aprender como ela funciona, recomendo fortemente que você use pacotes prontos, já testados e confiáveis.

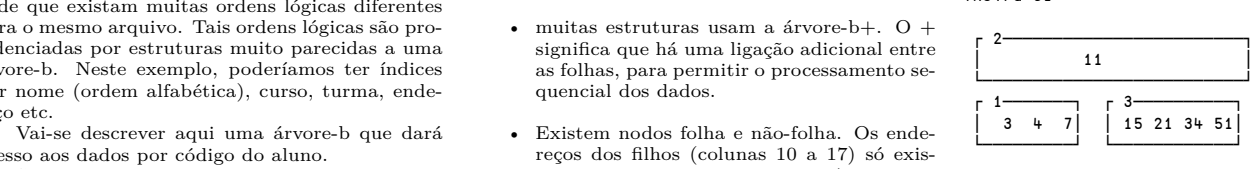
Um exemplo, passo a passo Seja uma árvore-b com $k=4$. Pela definição os nodos terão entre 3 e 7 chaves.

```
criabtree
insira 4, 7, 21, 11, 34, 15, 3
```

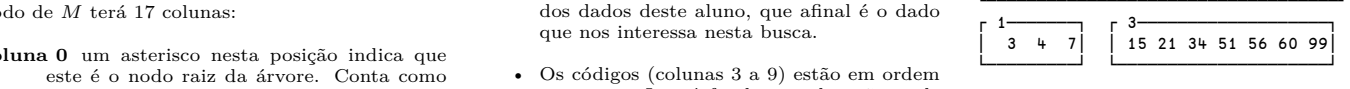


Agora, vamos inserir uma nova chave, o que vai provocar o split

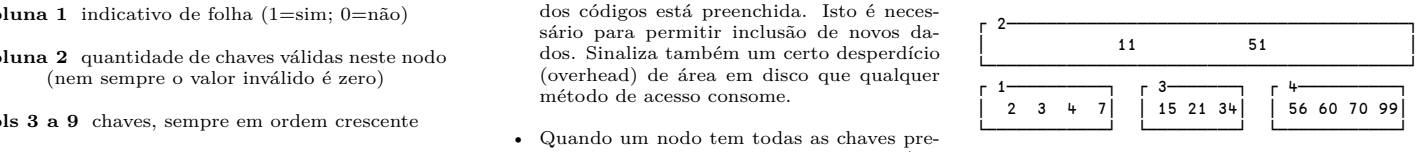
```
insira 51
```



Inserindo 60, 99, 56, fica



Inserindo 2 e 70, o 2 vai na boa para o nodo 1 enquanto o 70 força o split do nodo 3 e fica



Note que o processo funciona muito bem, mesmo que as chaves de entrada estejam desordenadas (ou em ordem aleatória). Isso é garantido pelos espaços em branco que são reservados nos nodos.

Um caso quasi-real: SYBASE

O SYBASE usa árvores B para indexar as tabelas. Existem 2 tipos de árvores B: as granuladas e as não granuladas (clustered). As granuladas são árvores esparsas (nem todo registro de dados tem entrada nos índices) nas quais os registros (na tabela original) são mantidos em ordem dos valores de índice e apenas o primeiro registro em cada página de dados tem uma entrada de índice. Árvores não granuladas são densas e cada um dos registros da tabela tem 1 entrada de índice.

Uma tabela SYBASE com índice granulado exige que seus elementos sejam mantidos em ordem dentro da tabela. Assim, inclusões podem se dar em qualquer página, a depender do conteúdo do campo de índice. Tabelas SYBASE que tem apenas índices não granulados fazem as inclusões de novos elementos sempre ao final das tabelas (na última página).

Tabelas SYBASE podem ter apenas 1 índice granulado, mas podem ter até 250 índices não granulados.

Na criação de um índice granulado, os dados da tabela devem estar colocados nas páginas em ordem.

Cada entrada de índice em sybase pode ser composta por até 16 campos e ter até 256 bytes. Estes campos não precisam estar contiguos. Só não podem ser partes de campos e resultados de cálculos.

O tamanho da página varia, mas o tamanho mais comum é 2K. Existem 5 tipos de páginas em sybase, a saber:

Páginas de Dados contém registros de dados ou registros de log. Ambos são estruturalmente diferentes, mas são construídos do mesmo jeito.

Páginas de índices contém registros de índice

Páginas de texto ou imagem contém BLOBS

Páginas de alocação contém estruturas de dados usadas para gerenciar o processo de alocacão de páginas

páginas de Estatísticas contém estatística de distribuição e uso para os índices.

Página de Dados Se os registros tem tamanho fixo, não haverá tabela de deslocamento na página e o endereço de cada registro é obtido mediante simples cálculo. Neste caso, o tamanho do registro (fixo) é guardado no campo TAMANHO MÍNIMO DE REGISTRO do header.

O header da página é um conjunto de 32 bytes contendo (entre outros): número lógico da página, anterior e próxima página lógica, identificação do objeto ao qual esta página pertence, primeira entrada de registro disponível nesta página, deslocamento do espaço livre, tamanho mínimo do registro

Depois vem a área de registros, que contém um número inteiro de slots para guardar registros. Um registro não atravessa limites de páginas

Tabela de deslocamento: Um conjunto de deslocamentos desde o início da página dando o início de cada um dos registros. Este conjunto cresce do fim da página para o início e só existe quando o tamanho do registro é variável.

Cada registro na página recebe um número de registro. É um campo de 1 byte e assim restringe-se o número de registros em uma página a 256. O número do registro usado nos índices não granulados (densos) é uma combinação do número da página com o número do registro na página.

O registro pode ocupar a página inteira (2048-32=2016). Não há tamanho mínimo para o registro, mas só pode haver 256 registros na página.

Os índices SYBASE são organizados na forma de arvores B + (o + significa que há uma ligação adicional entre as folhas, para permitir o processamento sequencial dos dados).

No exemplo aqui estudado, vão-se fazer algumas considerações/simplificações descritas a seguir.

Suponha um arquivo de pessoas contendo nome, cor preferida, bicho de estimação, alimento que gosta e idade.

Construindo um arquivo SYBASE contendo estes dados e que esteja sujeito às seguintes características:

- Registros de tamanho fixo igual a 40 bytes (9+9+9+9+4).
- Um índice granulado ou esparso (clustered) por um dos campos. No exemplo que é igual para todos, o índice esparso é por cor.
- Quatro índices não granulados (densos) pelos demais atributos.
- Páginas de dados com espaço para 4 registros e fator de preenchimento de 4:1 (usar apenas 3 espaços na carga original).
- Páginas de índices com espaço para 5 chaves (nome, cor, bicho, alimento ou idade). Lembrar que este número é impar, o que indica que no *split* vai subir a terceira chave (das 5 que existem).
- A identificação de cada registro deve ser página:número do registro na página. Por exemplo, a identificação 3:2, indica o segundo registro da terceira página.
- Nas páginas de índice (árvores-B) despreze as informações de endereço dos filhos. No exemplo, elas são irrelevantes. Faça como no exemplo.
- Os dados de entrada estão ordenados pela coluna do índice granulado.

Exemplo: Seja o conjunto de dados, no qual se fará o granulado por cor:

NOME	COR	ANIMAL	COMIDA	id.
josé	dourado	chopim	feijão	34
gabi	laranja	perú	aipim	22
sueli	lilas	gato	alho	40
antônio	musgo	vaca	batata	41
tatiana	petróleo	cavalo	arroz	37
juca	preto	águia	goiaba	36
romeu	roxo	dragão	maracujá	27
paulo	verde	urubú	rabanete	20

Com este conjunto de dados, ter-se-á a seguintes disposição nos arquivos

Páginas de Dados					
Pág1	josé	dourado	chopim	feijão	34
	gabi	laranja	perú	aipim	22
	sueli	lilas	gato	alho	40
Pág2	antônio	musgo	vaca	batata	41
	tatiana	petróleo	cavalo	arroz	37
	juca	preto	águia	goiaba	36
Pág3	romeu	roxo	dragão	maracujá	27
	paulo	verde	urubú	rabanete	20

A seguir, as páginas de índices

Páginas de índices	
Índice I1	granulado por COR
Página 1 ←raiz	dourado(1:1), musgo(2:1), roxo(3:1)
Índice I2	por NOME
Página 1	antônio(2:1), gabi(1:2)
Página 2 ←raiz	jose(1:1)
Página 3	juca(2:3), paulo(3:2), romeu(3:1), sueli(1:3), tatiana(2:2)
Índice I3	por BICHO
P1	águia(2:3), cavalo(2:2), chopim(1:1), dragão(3:1)
P2 ←raiz	gato(1:3)
P3	perú(1:2),urubú(3:2), vaca(2:1)
Índice I4	por COMIDA
P1	aipim(1:2),alho(1:3)
P2 ←raiz	arroz(2:2)
P3	batata(2:1), feijão(1:1), goiaba(2:3), maracujá(3:1), rabanete(3:2)
Índice I5	por IDADE
P1	20(3:2), 22(1:2), 27(3:1), 34(1:1), 36(2:3)
P2 ←raiz	37(2:2)
P3	40(1:3), 41(2:1)

☞ Para você fazer

Desenhe a árvore B a seguir:

```
    1 2    3   4   5   6   7   8   9 10 11 12 13 14 15 16 17
  +-----------------------------------------------------------
 1| 1 5   31  33  34  35  37   0   0
 2| 0 5   38  53  65  74  83   0   0  1 19  9  3  8 13
 3| 1 7   66  67  69  70  71  72  73
 4| 1 5  138 139 140 142 144   0   0
 5| 1 5   93  96  99 101 103   0   0
 6| 1 4  161 162 163 164   0   0   0
 7| 1 6  119 120 121 122 124 126   0
 8| 1 6   76  77  78  79  81  82   0
 9| 1 6   54  55  57  59  62  64   0
*10| 0 2   90 145   0   0   0   0   0  2 11 17
 11| 0 4  104 118 127 135   0   0   0  5 15  7 18  4
 12| 1 3  179 180 181   0   0   0   0
 13| 1 4   84  85  88  89   0   0   0
 14| 1 7  183 184 186 187 193 195 200
 15| 1 7  107 108 111 113 114 115 116
 16| 1 5  150 152 154 155 156   0   0
 17| 0 4  160 165 178 182   0   0   0 16  6 20 12 14
 18| 1 4  128 129 130 132   0   0   0
 19| 1 7   39  41  43  47  50  51  52
 20| 1 4  166 168 169 173   0   0   0
```

Responda agora:

Qual a altura da árvore B ?	Se o 97 for incluido, qual linha,coluna na matriz?	Se o 106 for incluído, qual linha,coluna na matriz?

Mais uma Instância do mesmo exercício Desenhe a árvore B a seguir:

```
    1 2    3   4   5   6   7   8   9 10 11 12 13 14 15 16 17
  +-----------------------------------------------------------
 1| 1 4   31  32  33  34   0   0   0
 2| 0 4   36  43  52  61   0   0   0  1 25  9 21 16
 3| 1 4  123 124 125 126   0   0   0
 4| 1 6  142 144 146 147 148 153   0
 5| 1 3   74  76  78   0   0   0   0
 6| 1 4   96  97  99 100   0   0   0
 7| 1 4  169 170 172 174   0   0   0
 8| 1 4  111 112 114 115   0   0   0
 9| 1 5   44  45  46  48  50   0   0
*10| 0 3   73 109 156   0   0   0   0  2 22 11 18
 11| 0 4  117 122 128 141   0   0   0  8 24  3 14  4
 12| 1 4  185 187 190 191   0   0   0
 13| 1 3   90  91  92   0   0   0   0
 14| 1 5  132 134 135 136 137   0   0
 15| 1 4  157 160 162 166   0   0   0
 16| 1 5   62  63  65  69  71   0   0
 17| 1 3  195 198 199   0   0   0   0
 18| 0 4  167 175 184 194   0   0   0 15  7 19 12 17
 19| 1 4  176 178 180 183   0   0   0
 20| 1 4   81  82  84  85   0   0   0
 21| 1 4   53  54  57  59   0   0   0
 22| 0 4   79  88  93 102   0   0   0  5 20 13  6 23
 23| 1 4  103 104 107 108   0   0   0
 24| 1 3  118 119 120   0   0   0   0
 25| 1 4   37  40  41  42   0   0   0
```

Responda agora:

Qual a altura da árvore B ?	Se o 51 for incluido, qual linha,coluna na matriz?	Se o 164 for incluído, qual linha,coluna na matriz?

CEP-UFPR-UTFPR-PUC/Pr-UP Sistemas de Informação 12/08/2024 - 11:35:52.4
Matemática aplicada Prof Dr P Kantek (pkantek@gmail.com)
Evolução de árvores: Árvore-B (e B+) VIVXq92a V: 1.02
76656 VITOR EDUARDO DE GOES FONTES
24CC2401 - 18 primeiro exercicio do T3 _____ / / _____

Árvore B

O conceito de árvore B foi desenvolvido por Rudolf Bayer e Edward McCreight em 1972.

Eles publicaram seu trabalho no artigo *"Organization and Maintenance of Large Ordered Indices"*, que descreve as propriedades e operações básicas das árvores B.

O trabalho de Bayer e McCreight teve um impacto significativo na área de ciência da computação, e as árvores B são agora amplamente utilizadas em bancos de dados e outros sistemas de software.

As árvores B são um tipo de árvore de busca balanceada que pode armazenar um grande número de chaves e valores de maneira eficiente. Elas são mais eficientes do que as árvores de busca binárias tradicionais, pois podem ter mais filhos por nó. As árvores B são usadas em uma variedade de aplicações, incluindo bancos de dados, sistemas de arquivos e sistemas de roteamento.

Eles não explicaram o significado do nome, só nos resta especular que esse B seja de Balanceada. Se você se lembrar das árvores binárias de pesquisa, deve recordar o problema que existe quando as chaves vem (quasi) ordenadas na entrada: a árvore gerada é desbalanceada e pode chegar a ter um desempenho terrível.

Outra qualidade da árvore-B é que ela tem grau k (k filhos por nodo). Em geral este valor é escolhido de modo a otimizar a blocagem de dados no arquivo de memória secundária. De qualquer modo este k modifica o desempenho, que lembrando é proporcional a $\log_k n$, onde n é a quantidade de chaves.

Como toda árvore, ela tem uma raiz que tem de 0 a $2k-1$ chaves e consequentemente tem de 0 a $2k$ filhos. Cada um dos filhos, tem de $k-1$ até $2k-1$ chaves de 0 até $2k$ filhos. Note que a raiz é o único nodo que sofre processamento especial, necessário para quando o arquivo tem uma única chave (por exemplo).

Um caso prático real Suponha um colégio com milhares de alunos, que têm seus dados guardados em um banco de dados. Por hipótese, os dados de cada aluno vão sendo recolhidos quando eles fazem a matrícula, pelo que a ordem física dos dados no arquivo é a data da matrícula. Parece óbvio assinalar que um determinado conjunto de dados (um arquivo) só pode ter uma única ordem física em um determinado instante. Mas nada impede que existam muitas ordens lógicas diferentes para o mesmo arquivo. Tais ordens lógicas são providenciadas por estruturas muito parecidas a uma árvore-b. Neste exemplo, poderíamos ter índices por nome (ordem alfabética), curso, turma, endereço etc.

Vai-se descrever aqui uma árvore-b que dará acesso aos dados por código do aluno.

Suponha uma árvore-b de nome M que providenciará este acesso. Vamos descrevê-la com $k=4$, que é muito pouco para aplicações reais, mas fica razoável para trabalhar com este exemplo. De acordo com a definição, os nodos de M têm entre 3 e 7 códigos de aluno. Para esta dimensão, o nodo de M terá 17 colunas:

coluna 0 um asterisco nesta posição indica que este é o nodo raiz da árvore. Conta como coluna 0 porque na verdade esta informação não está fisicamente aqui, está em outro lugar.

coluna 1 indicativo de folha (1=sim; 0=não)

coluna 2 quantidade de chaves válidas neste nodo (nem sempre o valor inválido é zero)

cols 3 a 9 chaves, sempre em ordem crescente

cols 10 a 17 endereços dos filhos associados a cada chave. Só estão preenchidas para nodos não folha.

Veja um exemplo:

```
    1 2    3    4    5    6    7    8    9 10 11 12 13 14 15 16 17
  +-------------------------------------------------------------------
 1| 1 5    3    6    7    8   10    0    0
 2| 0 4   11   20   32   44    0    0    0  1 20 15  7 12
 3| 1 4  128  131  132  134    0    0    0
 4| 1 5  149  150  151  153  160    0    0
 5| 1 6   65   73   76   79   80   83    0
 6| 1 5  182  183  188  189  191    0    0
 7| 1 3   34   35   37    0    0    0    0
 8| 1 4  106  109  111  114    0    0    0
 9| 1 6  136  137  138  140  143  145    0
*10| 0 2   64  127    0    0    0    0    0  2 17 11
 11| 0 6  135  146  162  171  181  192    0  3  9  4 14 21  6 18
 12| 1 7   46   49   50   55   56   57   63
 13| 1 4   88   89   91   94    0    0    0
 14| 1 5  164  166  167  168  169    0    0
 15| 1 6   21   22   23   28   29   30    0
 16| 1 7  117  118  120  122  123  125  126
 17| 0 4   84   95  105  116    0    0    0  5 13 19  8 16
 18| 1 5  193  195  197  199  200    0    0
 19| 1 5   96   98  100  102  103    0    0
 20| 1 4   14   15   17   19    0    0    0
 21| 1 3  174  175  177    0    0    0    0
```

Antes de continuar no exercício, você deve tentar entender o funcionamento da árvore-b.

- Desenhe a árvore
- identifique a necessidade da coluna 1. Idem para a coluna 2
- Qual a altura da árvore ?
- Quando e como a altura da árvore cresce ?

Agoram suponha que o elemento 187 é incluído. Qual a linha;coluna da matriz acima que é modificada, ou seja onde o elemento recém incluido entra ? Localize-o e veja:

```
    1 2    3    4    5    6    7    8    9 10 11 12 13 14 15 16 17
  +-------------------------------------------------------------------
 1| 1 5    3    6    7    8   10    0    0
 2| 0 4   11   20   32   44    0    0    0  1 20 15  7 12
 3| 1 4  128  131  132  134    0    0    0
 4| 1 5  149  150  151  153  160    0    0
 5| 1 6   65   73   76   79   80   83    0
 6| 1 6  182  183(187)188  189  191    0
 7| 1 3   34   35   37    0    0    0    0
 8| 1 4  106  109  111  114    0    0    0
 9| 1 6  136  137  138  140  143  145    0
*10| 0 2   64  127    0    0    0    0    0  2 17 11
 11| 0 6  135  146  162  171  181  192    0  3  9  4 14 21  6 18
 12| 1 7   46   49   50   55   56   57   63
 13| 1 4   88   89   91   94    0    0    0
 14| 1 5  164  166  167  168  169    0    0
 15| 1 6   21   22   23   28   29   30    0
 16| 1 7  117  118  120  122  123  125  126
 17| 0 4   84   95  105  116    0    0    0  5 13 19  8 16
 18| 1 5  193  195  197  199  200    0    0
 19| 1 5   96   98  100  102  103    0    0
 20| 1 4   14   15   17   19    0    0    0
 21| 1 3  174  175  177    0    0    0    0
```

Como voce pode ver ele está na linha ____ e coluna ____. Algumas características notáveis da árvore-b

- Figurinha carimbada no mundo do software: usam B-trees: SYBASE e do Harbour (antigo Clipper). Também são usados no Oracle, MySQL, PostgreSQL, Microsoft SQL e DB2, da IBM.
- muitas estruturas usam a árvore-b+. O + significa que há uma ligação adicional entre as folhas, para permitir o processamento sequencial dos dados.
- Existem nodos folha e não-folha. Os endereços dos filhos (colunas 10 a 17) só existem para os nodos que os têm (nodos não folhas).
- Os códigos de aluno (colunas 3 a 9 não mostram, para diminuir o ruído), mas ao lado de cada código está o endereço físico real dos dados deste aluno, que afinal é o dado que nos interessa nesta busca.
- Os códigos (colunas 3 a 9) estão em ordem crescente. Isto é fundamental e não pode ser modificado.
- Perceba como, em média, apenas a metade dos códigos está preenchida. Isto é necessário para permitir inclusão de novos dados. Sinaliza também um certo desperdício (overhead) de área em disco que qualquer método de acesso consome.
- Quando um nodo tem todas as chaves preenchidas e chega uma nova, que deveria (deverá) entrar nele, o nodo sofre um processo chamado split e ele é transformado em 2 nodos (cada um com metade dos espaços vazios). Note que o número máximo de chaves em qualquer nodo é sempre ímpar ($2n-1$) o que implica em que o elemento do meio sobe para o *pai* do nodo que está sendo quebrado. A metade anterior vai para um nodo *splitado* e a metade posterior vai para o outro nodo *splitado*.
- Quando as chaves vão sendo excluídas, e um nodo passa a ter menos do que a metade das chaves (o que é proibido) ele é juntado com um nodo irmão e ambos passam a ser um único nodo. Este processo recebe o nome de merge. Não é tão simples assim, há diversos casos particulares, mas a ideia é essa.

Mais um exemplo. Suponha a inclusão do número 26. Em que linha;coluna ele entra ?

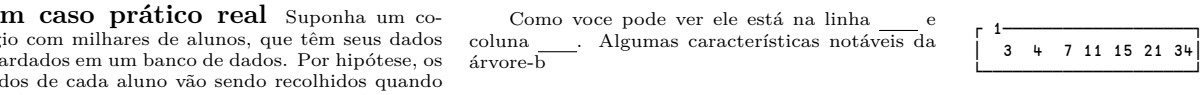
Algoritmo de Inclusão em árvore B

```
1: função INCLUA-BTREE (K:inteiro)
2: X ← raiz da árvore B
3: enquanto nodo X não é folha
4:     X ← filho correto onde K deve ser incluído
5: fim{enquanto}
6: se nodo X está cheio
7:     faça split do nodo X
8:     INCLUA-BTREE (K)
9:     abandone
10: senão
11:     inclua K no nodo X mantendo as chaves ordenadas
12: fim{se}
```

Algoritmos Estão disseminados pela comunidade. Deixo de apresentá-los aqui, pois são bastante específicos, complexos até, e mais importante do que implementá-los é entender-lhes o funcionamento. Até porque se você tiver que usar uma árvore b que não seja para aprender como ela funciona, recomendo fortemente que você use pacotes prontos, já testados e confiáveis.

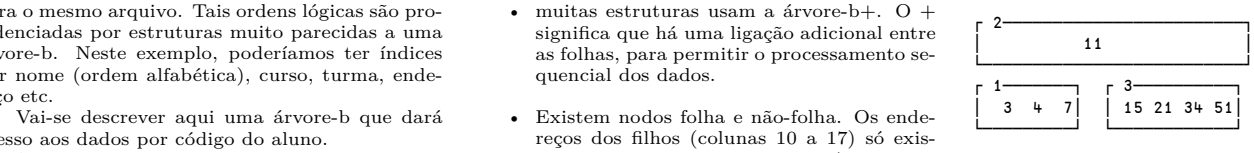
Um exemplo, passo a passo Seja uma árvore-b com $k=4$. Pela definição os nodos terão entre 3 e 7 chaves.

```
criabtree
insira 4, 7, 21, 11, 34, 15, 3
```

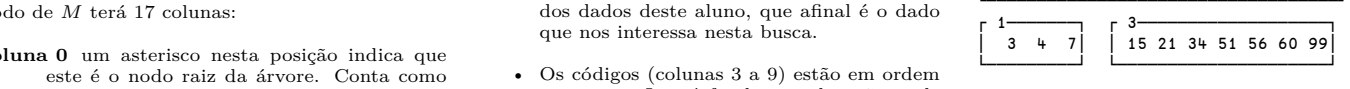


Agora, vamos inserir uma nova chave, o que vai provocar o split

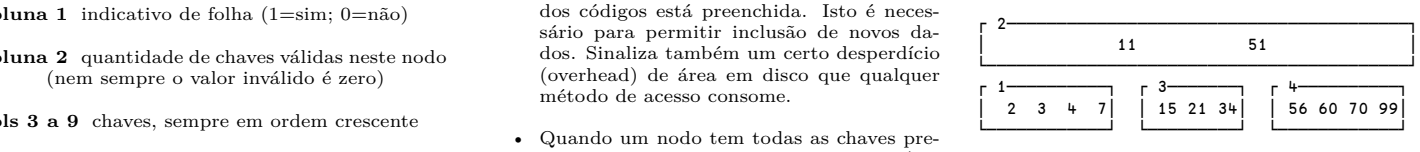
```
insira 51
```



Inserindo 60, 99, 56, fica



Inserindo 2 e 70, o 2 vai na boa para o nodo 1 enquanto o 70 força o split do nodo 3 e fica



Note que o processo funciona muito bem, mesmo que as chaves de entrada estejam desordenadas (ou em ordem aleatória). Isso é garantido pelos espaços em branco que são reservados nos nodos.

Um caso quasi-real: SYBASE

O SYBASE usa árvores B para indexar as tabelas. Existem 2 tipos de árvores B: as granuladas e as não granuladas (clustered). As granuladas são árvores esparsas (nem todo registro de dados tem entrada nos índices) nas quais os registros (na tabela original) são mantidos em ordem dos valores de índice e apenas o primeiro registro em cada página de dados tem uma entrada de índice. Árvores não granuladas são densas e cada um dos registros da tabela tem 1 entrada de índice.

Uma tabela SYBASE com índice granulado exige que seus elementos sejam mantidos em ordem dentro da tabela. Assim, inclusões podem se dar em qualquer página, a depender do conteúdo do campo de índice. Tabelas SYBASE que tem apenas índices não granulados fazem as inclusões de novos elementos sempre ao final das tabelas (na última página).

Tabelas SYBASE podem ter apenas 1 índice granulado, mas podem ter até 250 índices não granulados.

Na criação de um índice granulado, os dados da tabela devem estar colocados nas páginas em ordem.

Cada entrada de índice em sybase pode ser composta por até 16 campos e ter até 256 bytes. Estes campos não precisam estar contiguos. Só não podem ser partes de campos e resultados de cálculos.

O tamanho da página varia, mas o tamanho mais comum é 2K. Existem 5 tipos de páginas em sybase, a saber:

Páginas de Dados contém registros de dados ou registros de log. Ambos são estruturalmente diferentes, mas são construídos do mesmo jeito.

Páginas de índices contém registros de índice

Páginas de texto ou imagem contém BLOBS

Páginas de alocação contém estruturas de dados usadas para gerenciar o processo de alocacão de páginas

páginas de Estatísticas contém estatística de distribuição e uso para os índices.

Página de Dados Se os registros tem tamanho fixo, não haverá tabela de deslocamento na página e o endereço de cada registro é obtido mediante simples cálculo. Neste caso, o tamanho do registro (fixo) é guardado no campo TAMANHO MÍNIMO DE REGISTRO do header.

O header da página é um conjunto de 32 bytes contendo (entre outros): número lógico da página, anterior e próxima página lógica, identificação do objeto ao qual esta página pertence, primeira entrada de registro disponível nesta página, deslocamento do espaço livre, tamanho mínimo do registro

Depois vem a área de registros, que contém um número inteiro de slots para guardar registros. Um registro não atravessa limites de páginas

Tabela de deslocamento: Um conjunto de deslocamentos desde o início da página dando o início de cada um dos registros. Este conjunto cresce do fim da página para o início e só existe quando o tamanho do registro é variável.

Cada registro na página recebe um número de registro. É um campo de 1 byte e assim restringe-se o número de registros em uma página a 256. O número do registro usado nos índices não granulados (densos) é uma combinação do número da página com o número do registro na página.

O registro pode ocupar a página inteira (2048-32=2016). Não há tamanho mínimo para o registro, mas só pode haver 256 registros na página.

Os índices SYBASE são organizados na forma de arvores B + (o + significa que há uma ligação adicional entre as folhas, para permitir o processamento sequencial dos dados).

No exemplo aqui estudado, vão-se fazer algumas considerações/simplificações descritas a seguir.

Suponha um arquivo de pessoas contendo nome, cor preferida, bicho de estimação, alimento que gosta e idade.

Construindo um arquivo SYBASE contendo estes dados e que esteja sujeito às seguintes características:

- Registros de tamanho fixo igual a 40 bytes (9+9+9+9+4).
- Um índice granulado ou esparso (clustered) por um dos campos. No exemplo que é igual para todos, o índice esparso é por cor.
- Quatro índices não granulados (densos) pelos demais atributos.
- Páginas de dados com espaço para 4 registros e fator de preenchimento de 4:1 (usar apenas 3 espaços na carga original).
- Páginas de índices com espaço para 5 chaves (nome, cor, bicho, alimento ou idade). Lembrar que este número é impar, o que indica que no *split* vai subir a terceira chave (das 5 que existem).
- A identificação de cada registro deve ser página:número do registro na página. Por exemplo, a identificação 3:2, indica o segundo registro da terceira página.
- Nas páginas de índice (árvores-B) despreze as informações de endereço dos filhos. No exemplo, elas são irrelevantes. Faça como no exemplo.
- Os dados de entrada estão ordenados pela coluna do índice granulado.

Exemplo: Seja o conjunto de dados, no qual se fará o granulado por cor:

NOME	COR	ANIMAL	COMIDA	id.
josé	dourado	chopim	feijão	34
gabi	laranja	perú	aipim	22
sueli	lilas	gato	alho	40
antônio	musgo	vaca	batata	41
tatiana	petróleo	cavalo	arroz	37
juca	preto	águia	goiaba	36
romeu	roxo	dragão	maracujá	27
paulo	verde	urubú	rabanete	20

Com este conjunto de dados, ter-se-á a seguintes disposição nos arquivos

Páginas de Dados					
Pág1	josé	dourado	chopim	feijão	34
	gabi	laranja	perú	aipim	22
	sueli	lilas	gato	alho	40
Pág2	antônio	musgo	vaca	batata	41
	tatiana	petróleo	cavalo	arroz	37
	juca	preto	águia	goiaba	36
Pág3	romeu	roxo	dragão	maracujá	27
	paulo	verde	urubú	rabanete	20

A seguir, as páginas de índices

Páginas de índices	
Índice I1	granulado por COR
Página 1 ←raiz	dourado(1:1), musgo(2:1), roxo(3:1)
Índice I2	por NOME
Página 1	antônio(2:1), gabi(1:2)
Página 2 ←raiz	jose(1:1)
Página 3	juca(2:3), paulo(3:2), romeu(3:1), sueli(1:3), tatiana(2:2)
Índice I3	por BICHO
P1	águia(2:3), cavalo(2:2), chopim(1:1), dragão(3:1)
P2 ←raiz	gato(1:3)
P3	perú(1:2),urubú(3:2), vaca(2:1)
Índice I4	por COMIDA
P1	aipim(1:2),alho(1:3)
P2 ←raiz	arroz(2:2)
P3	batata(2:1), feijão(1:1), goiaba(2:3), maracujá(3:1), rabanete(3:2)
Índice I5	por IDADE
P1	20(3:2), 22(1:2), 27(3:1), 34(1:1), 36(2:3)
P2 ←raiz	37(2:2)
P3	40(1:3), 41(2:1)

☞ Para você fazer

Desenhe a árvore B a seguir:

```
     1 2    3    4    5    6    7    8    9 10 11 12 13 14 15 16 17
   +-------------------------------------------------------------------
 1| 1 6    31   32   33   35   38   39    0
 2| 0 5    41   54   65   82   97    0    0  1  9  4 14  3 20
 3| 1 6    84   87   90   92   94   96    0
 4| 1 5    55   56   58   59   64    0    0
 5| 1 7   156  157  158  159  160  161  162
 6| 1 6   164  165  168  169  170  173    0
 7| 1 5   125  127  130  131  132    0    0
 8| 1 6   108  109  110  112  113  115    0
 9| 1 7    42   43   44   47   48   50   53
*10| 0 2   107  147    0    0    0    0    0  2 11 17
 11| 0 4   116  122  133  139    0    0    0  8 16  7 19 12
 12| 1 5   141  143  144  145  146    0    0
 13| 1 6   189  190  191  194  195  197    0
 14| 1 6    70   72   73   74   76   79    0
 15| 1 5   148  149  151  152  154    0    0
 16| 1 3   118  119  121    0    0    0    0
 17| 0 4   155  163  175  185    0    0    0 15  5  6 18 13
 18| 1 3   176  178  181    0    0    0    0
 19| 1 4   134  135  136  137    0    0    0
 20| 1 5   101  102  103  104  106    0    0
```

Responda agora:

Qual a altura da árvore B ?	Se o 77 for incluido, qual linha,coluna na matriz?	Se o 186 for incluído, qual linha,coluna na matriz?

Mais uma Instância do mesmo exercício Desenhe a árvore B a seguir:

```
     1 2    3    4    5    6    7    8    9 10 11 12 13 14 15 16 17
   +-------------------------------------------------------------------
 1| 1 3    31   32   35    0    0    0    0
 2| 0 4    38   49   56   61    0    0    0  1 15  8 25  6
 3| 1 5   127  133  134  135  136    0    0
 4| 1 4    98  100  102  103    0    0    0
 5| 1 3   173  177  178    0    0    0    0
 6| 1 3    62   64   68    0    0    0    0
 7| 1 4   108  109  110  112    0    0    0
 8| 1 4    50   51   52   53    0    0    0
 9| 1 3   162  163  164    0    0    0    0
*10| 0 3    69  107  160    0    0    0    0  2 24 11 18
 11| 0 5   113  124  138  147  155    0    0  7 20  3 17 12 22
 12| 1 4   148  149  153  154    0    0    0
 13| 1 6   192  193  194  195  196  198    0
 14| 1 3    85   86   87    0    0    0    0
 15| 1 5    42   44   45   46   48    0    0
 16| 1 4   166  167  169  170    0    0    0
 17| 1 4   139  140  142  145    0    0    0
 18| 0 4   165  172  179  191    0    0    0  9 16  5 21 13
 19| 1 4    73   75   80   81    0    0    0
 20| 1 4   114  117  119  123    0    0    0
 21| 1 7   181  182  184  185  186  189  190
 22| 1 3   156  157  158    0    0    0    0
 23| 1 5    90   91   92   93   96    0    0
 24| 0 3    84   89   97    0    0    0    0 19 14 23  4
 25| 1 3    57   59   60    0    0    0    0
```

Responda agora:

Qual a altura da árvore B ?	Se o 187 for incluido, qual linha,coluna na matriz?	Se o 144 for incluído, qual linha,coluna na matriz?