UNIVERSIDADE FEDERAL DE OURO PRETO – UFOP

Ciência da Computação

ESTRUTURA DE DADOS II

RESUMO PESQUISA EXTERNA

Gabriel Mace do Santos Ferreira

Marcus Vinícius Souza Fernandes

Ouro Preto

2021

Pesquisa Externa, o que é?

Pesquisa Externa é um método de pesquisa em memória secundária. Ela é utilizada quando há dados demais para a memória principal, normalmente a eficácia (medida de complexidade) do método de pesquisa é comprovada por meio do número de transferências de dados entre memória principal e secundária, visto que os dados devem ser transferidos da memória secundária para a principal e também da principal para a secundária quando eles já não forem mais utilizados.

Sistema de paginação

Sistema de paginação consiste em uma estratégia que pode promover a implementação eficiente de métodos de pesquisa externa. Ela ocorre quando o número de itens excede a capacidade da memória principal, evitando transferir um item por vez para análise até encontrar este que satisfaz a condição desejada, no entanto, no pior caso desta situação, são necessárias **n** transferências.

Visando melhoria neste processo, a estratégia de paginação divide a memória secundária em blocos (páginas) que possuem tamanhos iguais e suficiente para que o bloco seja transferido para a memória principal e a devida pesquisa seja feita no mesmo, a complexidade do processo depende do número de blocos.

O mecanismo do sistema de paginação possui duas funções:

- **Mapeamento de endereços:** Determina se uma página da memória secundária já está na memória principal, evitando repetir a transferência.
- **Transferência das páginas:** Transferência da memória principal para a secundária e da memória secundária para a principal.

O endereço de um item em uma página é composto pelo número da página (p) e o número do byte do item dentro dela (b). Uma Tabela de páginas é utilizada para fazer o controle das páginas que se encontram na memória principal, fazendo assim, uma correspondência entre uma página na memória principal e uma página no arquivo binário.

Para acessar um item que não se encontra na memória principal é necessário carregar uma página da memória secundária para a memória principal, atualizando a tabela de páginas considerando apenas a página carregada. Caso não haja espaço na memória principal é necessário retirar uma página que não vá ser utilizada a longo prazo.

A escolha desta página emprega as seguintes estratégias:

1. **Menos recentemente utilizada (LRU):** Remove a página menos utilizada recentemente. As páginas são colocadas em uma fila e quando um item é utilizado, o mesmo é movido para o final da fila. O item no começo da fila é a página LRU e quando um novo item é adicionado à fila, ele assume a moldura que contém a página LRU.
2. **Menos frequentemente utilizada (LFU):** Remove a página acessada menos vezes. É necessário ter um contador associado a cada página, no entanto existe a possibilidade de eliminar uma página recentemente enviada para a memória principal, já que possui um número baixo de acessos. Uma possível solução seria elevar o contador de uma página recém adicionada para a média dos valores atuais.
3. **Ordem de chegada (FIFO):** Remove a página que se encontra na memória principal há mais tempo (fila tradicional), é o algoritmo mais simples e barato de se manter, no entanto ignora a importância da página para o código, devido a simplicidade do critério de desempate.

Acesso Sequencial Indexado

O acesso sequencial indexado faz uso do princípio da pesquisa sequencial, onde os itens são lidos sequencialmente até encontrar a chave de pesquisa desejada.

Os requisitos para a eficiência:

- O arquivo deve estar ordenado de acordo com a chave dos itens.
- A criação de um arquivo que contém um índice de páginas, na formatação <x,p>, sendo **x** a chave do item e **p** o endereço da página em que o primeiro item possui a chave x.

Para a sua execução, cria-se um índice de páginas, no índice é localizada a página que pode conter o item desejado ao fazer comparações da chave desejada com as chaves do primeiro item de cada uma das páginas, permitindo assim, inferir se o item desejado se encontra na lista ou não.

Exemplo de índice:

3	**14**	**25**	**41**
1	**2**	**3**	**4**

Determinada a página desejada, ela é levada para a memória principal onde será realizada uma pesquisa binária/sequencial (baseado na quantidade de itens existentes) para encontrar o item desejado.

Árvore binária de pesquisa externa

Árvores binárias de pesquisa são estruturas muito eficientes quando os dados se encontram na memória principal mas também podem ser utilizadas em memória secundária, de forma simplista.

Para simular uma árvore binária em um arquivo, armazena-se os nodos em disco e seus apontadores a esquerda e à direita com os respectivos endereços de disco, ao invés dos endereços da memória principal. Em suma, o arquivo binário possui a seguinte forma: cada posição possui o valor de um item e dois valores adicionais que representam o endereço (no arquivo) dos filhos a esquerda e direita respectivamente. Destacando que o valor **-1** representa um nó vazio (null) de uma árvore.

O número de transferências possui complexidade logarítmica, visto que para cada leitura é feita uma transferência, ou seja, cada vez que se deseja um item, lê-se o primeiro, determina o ramo da árvore em que o item pode se encontrar e o processo se repete até encontrar este valor desejado.

Torna-se possível melhorar a eficácia da árvore utilizando o método de páginas, o que permite a identificação do intervalo onde o valor pode ser encontrado, evitando ter que acessar a memória secundária constantemente para fazer comparações.

É importante ressaltar que após a inserção de um novo item, deve-se atualizar os valores adicionais para os filhos a esquerda ou direita e rebalancear a árvore.

Organização de nodos dentro de páginas:

- **Alocação sequencial:** Como o nome diz, é uma maneira de armazenar os nodos em posições consecutivas na página. Destaca-se como uma vantagem a utilização de todo o espaço da página e uma desvantagem seria a questão dos nodos estarem organizados pela ordem de entrada e não de acordo com o posicionamento na árvore.
- **Método de Muntz e Uzgalis:** Considera a proximidade dos nodos dentro da árvore, ou seja, coloca os nodos filhos na mesma página do pai, caso a página do pai esteja cheia o nodo é colocado no início de uma nova página criada. Destaca-se como uma vantagem, o número de acessos na pesquisa é próximo de ótimo e já como uma desvantagem, a questão da ocupação média das páginas ser baixa.
- **Árvore B:** Solução para o problema da árvore de Muntz Uzgalis, permitindo o crescimento equilibrado da árvore e permitindo inserções e retiradas.

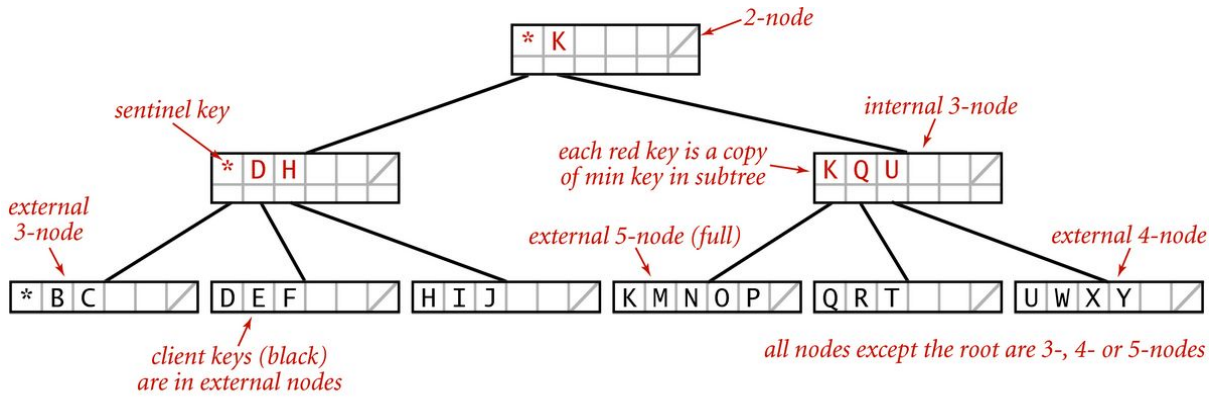
Árvore B:

Árvore B é uma técnica de organização e manutenção de arquivos, ela é uma árvore n-ária: possui mais de dois descendentes por nodo. A sua complexidade de pesquisa é logarítmica e é interessante destacar que é fundamental que a árvore esteja sempre balanceada.

Os elementos de uma árvore B de ordem **m**:

1. **Página raiz:** Contém de 1 a 2**m** itens;
2. **Outras páginas**: Contém pelo menos **m** itens e **m**+1 descendentes, no máximo 2**m** itens e 2**m**+1 descendentes.
3. **Páginas folhas (nodos que não possuem descendentes):** Aparecem todas no mesmo nível.
4. **Itens:** São armazenados em uma página em ordem crescente de acordo com as chaves, da esquerda para a direita.

Exemplo comentado da estrutura de uma árvore:



Anatomy of a B-tree set (M = 6)

Fonte: https://www.ime.usp.br/~pf/estruturas-de-dados/aulas/B-trees.html

Pesquisa em árvore B:

A pesquisa é semelhante à operação realizada na árvore binária de pesquisa, inicialmente compara-se a chave desejada com o valor da página raiz, a fim de determinar a “direção” que deve-se caminhar, o caminhamento ocorre seguindo o mesmo princípio anterior até que o intervalo desejado seja encontrado. Após isso, faz-se uma pesquisa sequencial para encontrar o item desejado.

Este processo se repete até a chave desejada ser encontrada, caso não seja possível localizá-la, a pesquisa retorna **null**.

Inserção em árvore B:

Para inserir um novo item na árvore B, Inicialmente é feita uma pesquisa até atingir uma página folha em que o valor desejado possa ser inserido, caso realmente haja uma posição disponível dentro da folha, o item é inserido e a folha é rearranjada de forma crescente. No entanto, se não houver espaço na página, devem ser adotadas estratégias para rearranjar os itens da página em questão.

A estratégia mais utilizada é criar uma nova página e inserir a ela metade dos elementos da página que se encontra cheia, após isso, o item central é movido para a página pai. No momento em que a árvore se encontra completamente cheia, mesmo a raiz, faz-se necessário criar uma nova raiz, dessa forma fazendo com que a árvore cresça para “cima”.

Remoção em árvore B:

Para realizar a remoção de um item na árvore B, inicialmente é necessário fazer uma pesquisa a fim de encontrar o item que se deseja, caso ele esteja em uma árvore folha é possível removê-lo imediatamente, caso contrário é necessário substituir o item por sua chave sucessora (subárvore mais a esquerda da árvore a direita) ou sua antecessora (subárvore mais a direita da árvore a esquerda).

As propriedades das páginas, quando utilizado o método de páginas, devem ser respeitadas, portanto quando ela chega a um número inferior de itens ao que deveria conter, é necessário que a árvore seja rearranjada. Ao receber um item “emprestado” de sua página irmã ocorre uma rotação de valores, quando não é possível a página irmã emprestar um valor, é feito uma fusão entre a página irmã e página que viola as propriedades. Além disso, o item “pai” desce para a página em que ocorreu a fusão.

Após o item pai descer é necessário verificar se a página pai viola a propriedade, se violar é necessário repetir o processo novamente. Se em algum momento a página raiz se tornar nula é necessário retirá-la e tornar a próxima página a raiz.

Durante a retirada, quando existe a possibilidade de escolha entre empréstimo ou fusão, a fusão é melhor no longo prazo, visto que reduz o tempo de pesquisa ao diminuir a altura da árvore, no entanto com o empréstimo não há alterações na árvore.

Observações:

- Páginas vizinhas com **m** itens sofrem fusão.
- Páginas com **m**+itens sofrem empréstimo.

Árvore B* de pesquisa externa

A árvore B* é uma das várias formas de se implementar a árvore B. Nesse caso os itens se encontram nas páginas folhas, enquanto as páginas superiores são organizadas no formato de uma árvore B e contêm os índices das páginas, ou seja, possuem apenas chaves que irão direcionar até o elemento desejado em uma página folha.

Devido ao fato que os registros se encontram apenas nas páginas folhas é possível fazer um apontador sequencial para que se possa caminhar dentro da árvore B com o índice e na árvore B* com o apontador. Caso um item for removido e fizer parte do índice, será necessário a remoção do mesmo apenas nas páginas folhas.

Pesquisa em árvore B*:

A pesquisa em árvore B* é semelhante à pesquisa na árvore B. Podemos destacar que a pesquisa sempre termina em uma árvore folha, visto que os registros se encontram sempre na parte inferior da árvore (folhas).

Inserção em árvore B*:

A inserção é igual à árvore B, no entanto quando a folha é dividida a chave do item do meio é encaminhada para uma página interna, enquanto o mesmo se mantém na página folha da direita.

Remoção em árvore B*:

Relativamente mais simples à árvore B, no entanto quando a remoção de um item viola as propriedades da página folha, faz-se necessário emprestar ou fundir as páginas.

- **Empréstimo:** Como os elementos da página pai são apenas chaves, eles não podem “descer” para a página folha, portanto o elemento da página vizinha deve ser transferido para a página que viola as propriedades diretamente e a chave da página pai é alterada para a chave do elemento movido.
- **Fusão:** Semelhante ao empréstimo, o item da página pai não pode “descer” para a página do filho visto que é uma chave, portanto as páginas são simplesmente fundidas e o item chave da página pai é removido. Caso ocorra a violação de propriedade nas páginas internas o processo para fusão e empréstimo é idêntico ao de uma árvore B.