Universidade de São Paulo

Escola de Artes, Ciências e Humanidades

Curso de Sistemas da Informação

Relátorio

Análise de algoritmos de busca sobre arquivos

Daniel Augusto de Melo Moreira – 8122477

SÚMARIO

INTRODUÇÃO

Muitos algoritmos foram e ainda são desenvolvidos, afim de eliminar/diminuir o gargalo no desempenho causado pela diferença de velocidade entre os diversos dispositivos em um computador.

Aqui será analisado algoritmos que têm por objetivo diminuir essa perda de desempenho quando há necessidade de se procurar dados armazenados em disco(memória secundária).

Com esse objetivo nas seções seguintes analisaremos os algoritmos de busca sequencial, busca binária e busca por índice, assim como algoritmos de ordenação externa e de geração de índices requeridos para a busca binária e por índice.

DESCRIÇÃO DOS ALGORITMOS

Busca Sequencial

A busca sequencial consiste em procurar o elemento desejado de forma linear, ou seja, testar todos os elementos do conjuto, um por um até que se encontre um elemento compatível ou que se tenha percorrido todos os elementos.

Essa abordagem permite que seja o algoritmo mais fácil de se implementar e enteder.

Busca Binária

A busca binária consiste em procurar o elemento desejado dividindo todos os elementos do conjunto no meio, então o elemento do meio é comparado, se for o elemento desejado a busca é encerrada. Senão continuamos a busca na parte anterior ou posterior do elemento mediano, conforme a posição do elemento procurado seja relativamente anterior ou posterior ao elemento mediano.

É fácil perceber que essa abordagem gera a necessidade que os dados estejam de alguma forma ordenados, caso contrário a busca poderia gerar resultados errados.

Ordenação Externa

A ordenação externa têm como objetivo ordenar grandes quantidades de dados. Como a quantidade de dados a ser ordenada é muito grande, então eles devem ser armazenados na memória secundária. Por isso o algoritmo deve diminuir ao máximo o número de acessos ao disco para que consiga o melhor desempenho possível.

O algoritmo atinge esse objetivo dividindo o arquivo em blocos do tamanho da memória principal dispónivel. Depois cada bloco é lido e ordenado utilizando a memória principal e por fim escrito novamente em um arquivo tempórario.

Quando todo o arquivo estiver ordenado em blocos menores, o algoritmo lê um pedaço de cada e os une em um arquivo de saída, utilizando uma estratégia semelhante ao

merge-sort. Quando todos os arquivos tiverem sido completamente lidos o arquivo de saída terá todos os dados ordenados.

Busca por Índice

A busca por índice consiste em procurar o elemento desejado em um arquivo menor, onde apenas as informações relevantes a busca são armazenadas, geralmente uma chave e a posição no arquivo original.

A busca por índice implementada possuí dois níveis: o primeiro ou base armazena a posição de cada dado no arquivo original e um campo de indentificação(ID), o segundo ou topo armazena o ID do primeiro dado de cada bloco(4096 bytes) do arquivo base e sua posição no mesmo.

A busca de um elemento ocorre do seguinte modo:

1. Busca binária no arquivo de indices topo para encontrar a posição no arquivo de indices base.
2. Busca binária no arquivo de indices base no bloco encontrado no passo 1.
3. Se um elemento compátivel foi encontrado acessamos o arquivo original na posição armazenada no bloco base.
4. Senão o elemento não pertence ao conjunto.

Geração de Índice

A geração de indices consiste em gerar dois arquivos, um arquivo de índices base que possuem uma relação um para um com o arquivo original e um arquivo de índices topo que armazena a primeira entrada de cada bloco do arquivo anterior.

A geração dos índices começa com a leitura do arquivo original e a escrita do campo identificador dos dados lidos, e suas respectivas posições relativas no arquivo original, no arquivo de índices base.

Em seguida o arquivo gerado é ordenado. E finalmente, o arquivo de indices topo é gerado inserindo as entradas referentes ao começo de cada bloco.

ANÁLISE DOS ALGORITMOS

Os algoritmos foram testados com três arquivos de dados de tamanhos de 62500, 125000 e 250000 blocos de 4096 bytes ou 32 registros cada. Os arquivos foram gerados visando o pior caso possível para a ordenação(dados completamente na ordem inversa) e para as buscas(sem repetição de chave).

Cada algoritmo de busca foi testado com 10 chaves, sendo que 8 dessas foram geradas aleatoriamente pelo site www.random.org/ dentro do espaço de chaves dos arquivos e 2 foram escolhidas fora do espaço de chaves, afim de analisar o custo de uma falha na busca.

A cada término de busca, bem sucedida ou não, o tempo em segundos e a quantidade de blocos acessados são registrados. Para cada algoritmo foi tirado o valor minimo, máximo e médio dessas duas medidas.

Por fim, os algoritmos serão avaliados conforme o dados coletados anteriormente.

ANÁLISE DOS DADOS OBTIDOS

Os dados coletados foram organizados em tabelas. Na tabela 1 foi registrado os dados coletados após 10 buscas em um arquivo de 62500 blocos.

Algoritmos	Busca Sequencial			Busca Binária			Busca por Índice		
Valores	Mín.	Máx.	Méd.	Mín.	Máx.	Méd.	Mín.	Máx.	Méd.
Blocos	1505	62500	30909,5	15	18	16,8	5	7	5,5
Tempo	0.047	1.907	0.9329	0.0	0.0	0.0	0.0	0.015	0.0015

Tabela 1 – Resultado de 10 buscas em um arquivo de 62500 blocos.

Ao analisar a tabela 1 podemos verificar que os algoritmos de busca binária e por índice têm um desempenho, no geral, melhor que a busca sequencial. Além disso, o algoritmo de busca por índice tem um desempenho geral 3 vezes melhor que a busca binária se analisarmos a quantidade de blocos acessados necessários para as realizações das buscas. Porém se compararmos esses dois últimos algoritmos pelo tempo gasto seus desempenhos são muito similares com uma ligeira vantagem para a busca binária que não precisa fazer requisições de E/S para 3 arquivos diferentes para realizar uma busca, o que pode gerar uma queda no desempenho.

Na tabela 2 foi registrado os dados coletados após 10 buscas em um arquivo de 125000 blocos.

Algoritmos	Busca Sequencial			Busca Binária			Busca por Índice		
Valores	Mín.	Máx.	Méd.	Mín.	Máx.	Méd.	Mín.	Máx.	Méd.
Blocos	2771	125000	67243,4	17	19	17,7	6	10	7
Tempo	0.078	3.547	1.9095	0.0	0.063	0.0422	0.0	0.016	0.0031

Tabela 2 – Resultado de 10 buscas em um arquivo de 125000 blocos.

Ao analisar a tabela 2 podemos verificar que os algoritmos de busca binária e por índice têm um desempenho muito melhor que a busca sequencial tanto em quantidade de blocos acessados quanto em tempo de execução.

O algoritmo de busca por índice dessa vez supera o de busca binária em ambas as unidades de desempenho, porém ainda ambos possuem um tempo de execução, no geral, muito perto de zero.

Na tabela 3 foi registrado os dados coletados após 10 buscas em um arquivo de 125000 blocos.

Algoritmos	Busca Sequencial			Busca Binária			Busca por Índice		
Valores	**Mín.**	**Máx.**	**Méd.**	**Mín.**	**Máx.**	**Méd.**	**Mín.**	**Máx.**	**Méd.**
Blocos	16558	250000	156550,1	18	19	18,3	7	9	7,9
Tempo	0.5	7.469	4.6738	0.0	0.015	0.0015	0.0	0.015	0.0015

Tabela 3 – Resultado de 10 buscas em um arquivo de 250000 blocos.

Ao analisar a tabela 3 podemos verificar que os algoritmos de busca binária e por índice, mais uma vez, superam o algoritmo de busca sequencial tanto em quantidade de blocos acessados quanto em tempo de execução.

Os algoritmos de busca por índice e busca binária nesse terceiro teste ficaram empatados ao analisar o tempo de execução requerido, porém pela terceira vez a busca por índice teve melhor desempenho na quantidade de blocos lidos.

CONCLUSÃO

As análises demonstraram que as unidades de desempenho do algoritmo de busca sequencial seguem uma relação linear com o tamanho do arquivo, enquanto para os algoritmos de busca binária e por índice o tamanho do arquivo parece ser pouco influenciável em suas respectivas variações de desempenho.

Além disso, em todos os três testes o algoritmo de busca sequencial teve o pior desempenho entre os três algoritmos testados e o algoritmo de busca por índice teve o melhor desempenho, analisando a quantidade de blocos acessados, e um desempenho praticamente igual ao de busca binária se analisarmos o tempo de execução de cada.

Por fim, se levarmos em conta apenas as buscas(descartando custos adicionais de ordenação e/ou geração de índices) o algoritmo de busca por índice com certeza possuí o melhor desempenho no geral.