COMPARAÇÃO DO TEMPO DE EXECUÇÃO DE ALGORITMOS EXECUTADOS EM COMPUTADORES COM MEMÓRIA CACHE

PRADELA, Izaura Pereira[1], OLIVEIRA, Lucineida Nara de Andrade[1], QUEIROZ, Marcela Dantas[1], PARREIRA JÚNIOR, Walteno Martins[2]

[1]Graduandas do Curso de Engenharia de Computação da UEMG - campus de Ituiutaba (UEMG-FEIT-ISEPI) email: izaurapradela@bol.com.br, lucineida_nara@hotmail.com ,cellysindy@hotmail.com

[2]Professor dos cursos de Engenharia da Computação, Engenharia Elétrica e Sistema de Informação da UEMG - campus de Ituiutaba (UEMG-FEIT-ISEPI) – email: walteno@ituiutaba.uemg.br.

Área FAPEMIG: Ciência da Computação – 1.03.03.00-6

Resumo: *No desenvolvimento de softwares, é importante avaliar sua correção temporal, principalmente em softwares de sistemas de tempo real. Alguns aspectos arquiteturais internos dos processadores, tais como memória cache e pipelines, influenciam no tempo de execução das instruções de máquina. Este artigo apresenta análises de dois algoritmos de ordenação com elevado tempo de execução (usando delay's) executados no mesmo computador.*

Palavras-chave: algoritmo, memória cachê, tempo de execução.

1. Introdução

Este trabalho relata a experiência de desenvolvimento de algoritmos de programação e a sua otimização para melhoria de performance. A experiência consiste em utilizar os mesmos algoritmos, no caso de ordenação interna, e observar a variação de tempo entre as execuções, ocasionados por recursos internos aos processadores.

A memória cache surgiu quando percebeu-se que as memórias já não eram capazes de acompanhar os processadores em termos de velocidade, fazendo com que ele ficasse esperando a memória RAM liberar os dados para poder concluir suas tarefas, perdendo muito tempo em desempenho.

Para solucionar este problema passou a ser utilizada a memória cache, pois o acesso é rápido a esse dispositivo e ela serve para armazenar dados que possuem uma grande probabilidade de serem utilizados novamente pelo processador. Com uso da memória da memória cache o tempo de execução pode cair em até 95% comparados com o uso somente da memória RAM.

A memória cache contém uma cópia de partes da memória principal. Assim, quando o processador deseja ler uma palavra da memória, uma verificação é efetuada para determinar se a palavra está na memória cache, otimizando o acesso. Caso ela esteja é imediatamente fornecida ao processador. Caso contrário, um bloco de dados da memória principal é lido para a memória cache e em seguida a palavra requerida é entregue ao processador.

Os algoritmos foram implementados na Linguagem C, compilados no TurboC e executados várias vezes no mesmo computador, permitindo que seus tempos de execução fossem comparados.

2. Desenvolvimento

Os algoritmos foram executados em um computador com as seguintes configurações: processador Intel (R) Pentium (R) 4, CPU 2.80GHz, 261.664 KB de RAM, CACHE L2 512KB.

Os métodos de ordenação constituem um bom exemplo de como resolver problemas utilizando computadores. As técnicas de ordenação permitem apresentar um conjunto amplo de algoritmos distintos para resolver uma mesma tarefa. Dependendo da aplicação, cada algoritmo considerado possui uma vantagem particular sobre os outros algoritmos. Segundo Ziviani (2002, p. 71), como são algoritmos de ordenação interna, significa que utilizam uma estrutura de vetor que armazenam na memória principal. Nestes casos, a medida de complexidade relevante contam o número de comparações entre as chaves e o número de movimentações de itens no arquivo.

Os algoritmos de ordenação implementados na linguagem C para comparação foram: a) por seleção e b) por inserção. São métodos de fácil compreensão e de implementação.

2.1 - Ordenação por seleção

A ordenação por seleção consiste, em cada etapa, em selecionar o maior (ou o menor) elemento e alocá-lo em sua posição correta dentro da futura lista ordenada. Durante a execução, a lista com **n** registros é decomposta em duas sublistas, uma contendo os itens já ordenados e a outra com os elementos ainda não ordenados. No início, a sublista ordenada está vazia e a desordenada contém todos os elementos, no final do processo a sublista ordenada apresentará (n-1) elementos e a desordenada terá um elemento. Nesta experiência foram usados vetores de tamanho: 10, 20, 30 e 40 elementos. Segundo Parreira Júnior (2006, p.2) a complexidade deste método é $O(n^2)$ para qualquer que seja os valores iniciais; e o número de comparações é $C(n) = ½ (n^2 – n)$.

2.2 - Ordenação por inserção

O método consiste em cada passo, a partir de i=2, o i-ésimo item da seqüência fonte é apanhado e transferido para a seqüência destino, sendo colocado na posição correta. A inserção do elemento no lugar de destino é efetuado movendo-se os itens com chave maiores para a direita e então inserindo-o na posição que ficou vazia. Neste processo de alternar comparações e movimentação de registros existem duas situações que podem causar o término do processo, a primeira é quando um item com chave menor que o item em consideração é encontrado e a segunda situação é quando o final da seqüência destino é atingida (à esquerda). A melhor solução para a situação de um anel com duas condições de terminação é a utilização de uma sentinela, para isto, na posição zero do vetor coloca o próprio registro analisado. Segundo Parreira Júnior (2006, p.4) a complexidade deste método é $O(n^2)$ para os casos médio (números distribuídos randomicamente) e pior caso (situação em que está inversamente ordenado) em relação aos valores iniciais; e o número de comparações para o caso médio é $C(n) = ¼ (n^2 + 11\ n) – 3$.

3. Resultados obtidos

3.1-Análise do algoritmo de ordenação por seleção:

Tamanho do vetor	Nro Comparações	Média (segundos)	Tempo 1	Tempo 2	Tempo 3
10	45	32,7s	30s	34s	34s
20	190	116s	112s	118s	118s
30	435	210,3s	239s	243s	149s
40	780	335s	434s	227s	344s

Tabela 1-número de comparações e tempo de execução

Pode-se observar na tabela 1 que o número de comparações determinado pela expressão encontrada na literatura segue a complexidade e então é crescente. A média de tempo determinada experimentalmente não segue a mesma razão que a complexidade, o que deve ser ocasionado por outras ações em tempo de execução. Uma das explicações é exatamente a utilização da memória cachê para o armazenamento dos dados que estão em execução.

3.2-Análise do algoritmo de ordenação por inserção:

Tamanho do vetor	Nro Comparações	Média (segundos)	Tempo 1	Tempo 2	Tempo 3
10	49,5	43,3s	43s	44s	43s
20	152	183,7s	187s	187s	177s
30	304,5	238,3s	312s	220s	183s
40	507	767s	761s	770s	770s

Tabela 2- número de comparações e tempo de execução

Observa-se que a diferença de tempo de execução é grande para os vetores de 30 e 40, isso se deve ao fato do computador estar usando a memória cache ao invés de acessar a memória RAM na execução do algoritmo. Como esse algoritmo faz muitas comparações, o bloco da memória RAM que o processador mais utilizava foi transferida para a memória cache a partir da segunda vez em que o algoritmo foi executado, diminuindo o tempo de acesso para leitura e assim o tempo de processamento (diminuição de até 51% do tempo) do algoritmo como um todo.

4. Conclusão

Durante a análise desses algoritmos, concluímos que para desenvolver um software qualquer não é necessário apenas ter conhecimento das linguagens de programação e uma ferramenta de implementação em mãos. É necessário entender como funcionam os componentes que compõem a arquitetura dos computadores, pois o software pode não atender os valores de limite de tempo que se deseja que ele execute.

5. Bibliografia

MORIMOTO, Carlos E. **Memória Cachê.** IN: Guia do Hardware. Disponível em: <http://www.guiadohardware.net/termos/memoria-cache> acesso em 08 set. 2008.

PARREIRA JÚNIOR, Walteno Martins. **Métodos de ordenação (Apostila).** Ituiutaba: FEIT-UEMG, 2006.

STALLINGS, W. **Arquitetura e Organização de Computadores**, 5ª Edição, São Paulo: Prentice Hall, 2002. (Cap. 4)

ZIVIANI, Nivio. **Projeto de Algoritmos: Com Implementação em Pascal e C**. São Paulo: Pioneira – Thonson Learning, 2002.

Para Referencia do Artigo:

PRADELA, Izaura Pereira, OLIVEIRA, Lucineida Nara de Andrade, QUEIROZ, Marcela Dantas & PARREIRA JÚNIOR, Walteno M. Comparação do tempo de execução de algoritmos executados em computadores com memória cache. IN: Seminário de Iniciação Científica e extensão da UEMG, 10, 2008, Divinópolis (MG). **Anais do 10 Seminário da UEMG**. Divinópolis: UEMG e FUNED. 2008. CD-ROM. ISSN: 1983-9693. Disponível em <www.waltenomartins.com.br/artigos>