

Algoritmos de ordenação- Desempenho

Thiago Ramos de Souza¹

Faculdade Paraíso do Ceará (FAP)
Rua da Conceição, 1228 – Juazeiro do Norte – CE – Brasil

thiagosouzasi@hotmail.com

Abstract. *The operations of sorting data is of paramount importance to computer systems, choosing the most efficient method has great impact on the performance of a given application. This paper will discuss data sorting algorithms using two elementary sort, selection sort and insertion sort with a focus on performance presented in order to establish what is the most efficient in the ordering of elements in a vector. The performance calculation is performed by observing the number of iterations performed for total ordering of the elements on the basis the worst case, ie sorting elements incrementally from a set of data arranged in decreasing order.*

Keywords : Algorithms. Ordering. Performance.

Resumo. *As operações de ordenação de dados são de suma importância para sistemas computacionais, a escolha do método mais eficiente tem grande impacto no desempenho de uma determinada aplicação. O presente trabalho discorrerá sobre a ordenação de dados utilizando dois algoritmos de ordenação elementares, ordenação por seleção e ordenação por inserção com enfoque no desempenho apresentado buscando estabelecer qual o mais eficiente na ordenação de elementos em um vetor. O cálculo de desempenho será realizado através da observação do número de iterações realizadas para ordenação total dos elementos tomando como base o pior caso O, ou seja ordenar elementos de forma crescente a partir de um conjunto de dados ordenados de forma decrescente.*

Keywords: Algoritmos. Ordenação. Desempenho.

1 Introdução

Este artigo tem por finalidade abordar de maneira sucinta um tema que é bastante discutido entre estudantes e profissionais de informática, o desempenho de algoritmos utilizados para ordenação.

Para obtenção dos resultados utiliza-se dois dos mais simples e conhecidos algoritmos para ordenação de elementos em um vetor, os métodos de inserção e Seleção testando o seu desempenho com base no número de iterações que cada algoritmo realiza para conseguir ordenar o vetor completamente, ou seja quantas vezes o algoritmo troca os valores de lugar dentro do vetor para obtenção do resultado final.

A quantidade de posições do vetor bem como os valores das chaves são os mesmos para os dois algoritmos, ambos são implementados em linguagem C e os testes feitos no mesmo computador, garantindo assim que as condições sejam as mesmas para os dois algoritmos, o que permite que os resultados obtidos sejam mais confiáveis. Ainda com esse mesmo intuito o teste contempla sempre o pior caso ou seja a complexidade O , que nesse caso será o vetor inicialmente ordenado em ordem decrescente.

A escolha do tema deve-se principalmente ao fato de os métodos de ordenação apresentarem bastante relevância no campo da programação sendo de grande importância por apresentarem muitas aplicações em problemas reais do cotidiano. A distribuição dos dados de maneira organizada facilita o acesso e a recuperação de dados armazenados anteriormente em uma aplicação.

2 Objetivos

2.1 Objetivo Geral

Apontar qual entre os métodos de ordenação escolhidos qual apresenta melhor desempenho na ordenação dos elementos de um vetor utilizando como parâmetro o número de iterações realizadas.

2.2 Objetivos específicos

Determinar através de testes realizados com implementações em C dos métodos de ordenação por inserção e ordenação por seleção, o número de trocas das chaves entre as posições de um mesmo vetor passado para os dois métodos, afim de verificar qual dos dois apresenta melhor performance quanto ao número de iterações realizadas ou seja qual deles resolve o problema com menor número de trocas para ordenação completa do vetor.

3 Levantamento Bibliográfico

A ordenação de dados é de grande importância e amplamente discutida nos meios da informática, sendo bastante discutida e alvo de muitos estudos.

Um algoritmo de busca eficiente sem dúvidas é de grande valia em inúmeras aplicações pois são muito comuns os casos em que para termos acesso aos dados de maneira eficiente precisamos organizá-los. Corroborando com essa ideia Adam Drozdek (2002, p.404) escreve:

[...]A conveniência de usar dados ordenados é inquestionável e precisa ser aplicada também a ciência da computação. Embora um computador possa manipular um caderno de telefones não-ordenado mais fácil e rapidamente do que o ser humano, é extremamente ineficiente ter o computador processando um conjunto desordenado de dados. Frequentemente é necessário ordenar os dados antes do processamento. [...]

Seguindo a mesma linha de pensamento e concordando com as ideias expostas acima este trabalho propõe um breve estudo sobre dois métodos de ordenação de dados elementares.

4 Discussão

Ordenar corresponde a ação de organizar um conjunto de elementos podendo ser crescente ou decrescente, a ordenação torna-se necessária e indispensável para obtenção de desempenho em análise de dados, organização de elementos, acesso rápido aos elementos, relatórios gerenciais, comparações periódicas de elementos, etc. Uma decisão importante antes de trabalhar com algum algoritmo de ordenação é saber qual se adéqua melhor ao seu problema.

Levando em consideração apenas como principal fator de desempenho dos algoritmos para a ordenação, a atribuição de valores decorrentes das trocas de posições no vetor no instante da ordenação do mesmo. Os algoritmos ordenação por seleção e ordenação por inserção foram analisados e comparados destacando as suas peculiaridades, provando a suas eficiências em sua função de ordenação, ambos trabalhando com o pior caso particular entre os dois e utilizando as mesmas ferramentas propostas para análise de comparação.

Ordenação por seleção, baseando-se no pior caso e considerando ordenação crescente, o algoritmo realiza duas operações que compreendem a ordenação: percorre o vetor seleciona no vetor o menor elemento, efetua uma troca de posições com o primeiro elemento do vetor, repete o mesmo procedimento percorrendo o vetor, selecionando o elemento menor para trocar com o segundo, depois o terceiro, assim sucessivamente até o último elemento do vetor ser ordenado.

Ordenação por inserção considerando ordenação crescente, e ainda com base no pior caso o algoritmo percorre o vetor e a cada posição do vetor a partir da posição inicial mais um, é feita uma comparação do elemento do que está sendo inserido com o elemento anterior a ele no vetor caso o elemento a ser inserido seja menor que o elemento anterior a ele no vetor é feita a troca e é feita uma nova verificação para o caso de ainda existirem elementos maiores que o elemento a ser inserido em posição anterior a ele no vetor se houver efetua-se novas trocas isso se repete até que o elemento seja inserido na posição adequada e outro elemento entre no laço, assim sucessivamente até que o último elemento do vetor seja ordenado.

Os algoritmos foram comparados em seu desempenho com base no comportamento dos resultados dos testes com a meta de avaliar de modo geral a quantidade de iterações geradas pelo trabalho de cada algoritmo para ordenar vetores de elementos numéricos, os algoritmos foram submetidos a três testes, todos os testes são idênticos para os dois métodos de ordenação, ordenar: um primeiro vetor numérico com dez posições ordenado de forma decrescente com elementos de dez a um, um segundo vetor numérico de vinte posições ordenado de forma decrescente com elementos de vinte a um, um terceiro vetor numérico de quarenta posições ordenado de forma decrescente com elementos de quarenta a um.

O algoritmo de seleção obteve os seguintes resultados: O vetor de dez elementos foi ordenado com vinte e sete iterações; vetor de vinte elementos foi ordenado com cinquenta e sete iterações; vetor de quarenta elementos foi ordenado com cento e dezessete iterações. Foi detectada uma tendência que para a ordenação total do vetor se manteve o padrão constante de interações conforme a formula: $3(n-1)$, ou seja o número de iterações é constante e é sempre igual a três vezes $n-1$ onde “n” é o número do posições vetor (ADAM DROZDEK, 2009).

O algoritmo de inserção obteve os seguintes resultados: vetor de dez elementos foi ordenado com cinquenta e quatro iterações; vetor de vinte elementos foi ordenado com duzentas e nove iterações; vetor de quarenta elementos foi ordenado com oitocentos e dezenove iterações. Foi observado que para a ordenação total dos vetores decrescentes se manteve o padrão constante de interações, para o vetor já ordenado tem-se um padrão constante conforme a formula: $(n-1)$, onde “n” corresponde ao número de elementos do vetor.

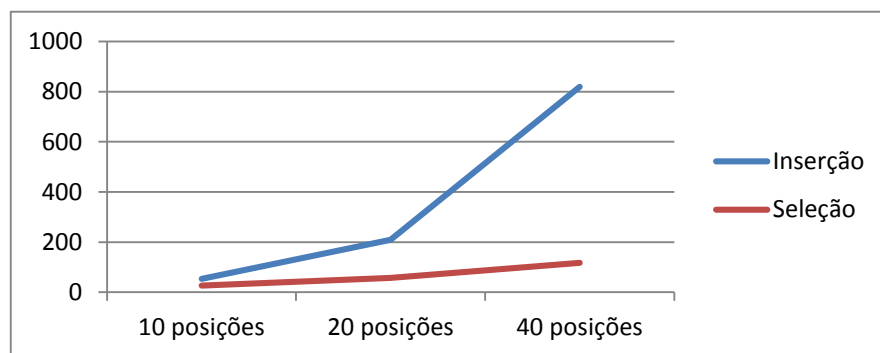


Gráfico 1 – Gráfico de desempenho dos métodos de ordenação

Os algoritmos de ordenação atingem o objetivo e ordenam com perfeição, ordenação por seleção foi bastante satisfatória nos três testes apresentando desempenho melhor que o de ordenação por inserção que obteve melhor resultado bem inferior. Fazendo uma análise dos resultados obtidos nos testes com os algoritmos podemos constatar que:

[...] O método [ordenação por seleção] é espetacular quanto ao número de movimentações de registros [...] Consequentemente, este é o algoritmo a ser utilizado para arquivos com registro muito grandes. (NIVIO ZIVIANI, p. 99, 2005) [...]

O método de inserção é o método a ser utilizado quando o arquivo está “quase” ordenado. [...] deseja adicionar uns poucos itens a um arquivo já ordenado e depois obter outro arquivo ordenado. (NIVIO ZIVIANI, p.101, 2005).

Com isso podemos verificar que os resultados dos testes refletem com precisão o que já fora escrito anteriormente por outros autores sobre o desempenho dos métodos estudados neste trabalho.

5 Considerações Finais

De acordo com os resultados obtidos nos testes realizados com os métodos de ordenação pode-se concluir que a ordenação por inserção não é um bom método de ordenação para ser aplicado quando o a quantidade de registros a serem ordenados é grande por apresentar um aumento muito grande no número de interações conforme o número de dados forem aumentando. Já o algoritmo de seleção apresenta um crescimento do número de interações que é sempre constante e atende a formula $3*(n-1)$ onde “n” é o número de elementos a serem ordenados.

6 Referências Bibliográficas

DROZDEK, A. **Estruturas de dados e algoritmos em C++**. 1ª ed. São Paulo-SP: Cenage Learning, 2009

MADEIRA, Thiago. **Algoritmos computacionais**. Disponível em <<http://algoritmos.tiagomadeira.net>>. Acessado em: 14 Jun. 2013

PUGA, S; RISSETTI, G. **Lógica de programação e estruturas de dados**. 2ª ed: São Paulo-SP: Pearson Prentice Hall, 2009.

.

ZIVIANI, N. **Projeto de algoritmos**: com implementação em Pascal e C. 2ª Ed: São Paulo-SP: Pioneira Thomson Learning, 2005.