Algoritmos e Estruturas de Dados II

Diego Trevisan Lara (GRR20091968) e Vinícius André Massuchetto (GRR20063784)
Universidade Federal do Paraná

Análise de Desempenho dos Algoritmos 'Selection Sort' e 'Merge Sort'

Resumo: Algoritmos de ordenação são fundamentais para uma série de processos computacionais utilizados cotidianamente pela sociedade, e é de igual importância que saibamos em quais condições uns são mais adequados do que outros para o melhor tratamento dos dados. O presente artigo expõe um estudo a respeito dos algoritmos 'selection sort' e 'merge sort', procurando relacionar suas especificidades e custos computacionais.

Palavras-chave: selection sort, merge sort, ordenação, seleção, algoritmo, custo computacional

Introdução

Os processos de ordenação “selection sort” e “merge sort” compõem o conjunto básico de algoritmos que têm por finalidade a ordenação de dados. Ambos foram tomados como objeto de estudo para a análise de custo de algoritmos.

A análise de custos de algoritmos é feita de maneira ampla e diversificada de acordo com cada aplicação. Tomam-se parâmetros de medição como tempo de execução, tempo do processador, número de acessos à memória, atribuições, trocas, passagem de parâmetros, comparações e outros[1].

O Selection Sort possui complexidade quadrática por possuir duas instâncias de percorrimento da lista de dados: uma que permanece sobre o limiar da porção ordenada e desordenada do vetor, e outra que percorre a porção desordenada em busca do menor ou maior valor, dependendo do sentido da ordenação[2].

Já o Merge Sort possui complexidade logarítmica , e faz parte de um grupo de algoritmos do tipo “dividir para conquistar”, dividindo a sequência original em pares de dados ordenados, e depois juntando-os[3].

Métodos

Utilizou-se a linguagem C e o compilador GCC para implementação dos dois algoritmos e do gerador de números aleatórios do kernel do Unix através do Bash. Este último gerou a sequência de números aleatórios submetidos à um arquivo de entrada lido pelo programa e ordenado pelos dois algoritmos.

```
for i in {1..1000000}; do
    echo $RANDOM;
done > teste.in
```

Algoritmo 1: Gerando Números Aleatórios no Bash

O custo foi computado através do incremento de variáveis de acordo com o número de interações de cada algoritmo, o que no caso da versão interativa do *Selection Sort* é o número total de comparações como mostra o Algoritmo 2.

```
void SelectionSort(int *v,int tam) {
    int i, j, min, aux;
    for(i=0; i<tam-1; i++) {
        min = i;
        aux = v[i];
        for(j=i+1; j<tam; j++) {
            if (v[j] < aux) {
                min=j;
                aux=v[j];
            }
        }
        aux = v[i];
        v[i] = v[min];
        v[min] = aux;
    }
}
```

Algoritmo 2: Selection Sort

O Algoritmo 3 é o *Merge Sort* que teve seu custo contado a cada chamada recursiva do *Intercala.*

```
void MergeSort(int *v, int tam) {
    int meio;
    if (tam > 1) {
        meio = tam / 2;
        MergeSort(v, meio);
        custo_merge =
        MergeSort(v + meio, tam - meio);
        Intercala(v, tam);
    }
}
```

Algoritmo 3: Merge Sort

1 MUSSER, David - Measuring Computing Times and Operation Counts of Generic Algorithms

2 Selection Sort – Wikipedia

3 Merge Sort - Wikipedia

Resultados e Discussões

Com as descrições das complexidades dos algoritmos (quadrádica e logarítmica) das fórmulas 1 e 2, pode-se verificar que em algum momento haverá um cruzamento dos custos, pois um tenderá ao crescimento de custo muito rapidamente, enquanto o outro terá um crescimento linear.

$$C(merge) = nlogn$$

Fórmula 1: Custo Analítico do Merge Sort

$$C(merge) = nlogn$$

Fórmula 2: Custo Analítico do Selection Sort

O objetivo aqui seria verificar em que ponto um algoritmo torna-se mais custoso do que outro.

Com alguns procedimentos de ordenação chegou-se os seguintes valores que parecem ser suficientes para expor os resultados:

elementos	selection	merge
1	0	0
2	1	2
3	3	4
4	6	8
5	10	11
6	15	15
7	21	19
8	28	24
9	36	28
10	45	38

Tabela 1: Valores de Custo dos Algoritmos

Referenciando-se aos custos dos algoritmos pode-se traçar o Gráfico 1.

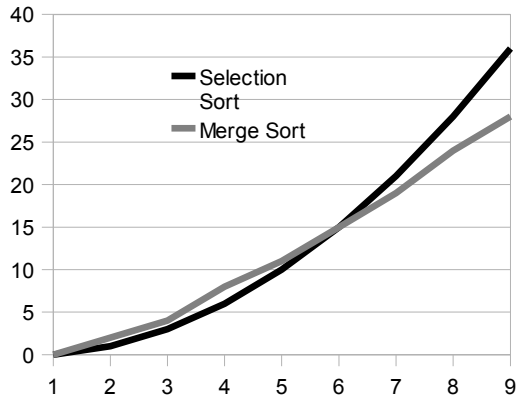

Gráfico 1: Selection Sort X Merge Sort Até 9 Elementos

Para fins de confirmação, gerou-se um arquivo com 10^6 números aleatório, o qual foi submetido à ordenação somente pelo Merge Sort, e em outra instância pelo algoritmo combinado juntamente com o comando *time* do Unix.

tipo	tempo real	tempo do usuário
somente merge	0. 784	0. 740
combinado	0. 699	0. 654

Tabela 2: Tempos de execução para o *Merge Sort* e para o Algoritmo Combinado

Conclusão

O custo de cerca de seis a sete elementos de um vetor passa a ser maior em um algoritmo de seleção do que em um algoritmo de mistura.

Por tais motivos pode-se construir um algoritmo combinado a fim de verificar o tamanho do fragmento de dados que o *Merge Sort* gera, e se este fragmento é maior ou melhor do que o limiar de custo entre os dois algoritmos.

```
void MergeSort(int *v, int tam) {
    int meio;
    if (tam >= 7) {
        meio = tam / 2;
        MergeSort(v, meio);
        MergeSort(v + meio, tam - meio);
        Intercala(v, tam);
    } else {
        SelectionSort(v,tam);
    }
}
```

Algoritmo 4: Merge Sort e Selection Sort Combinados

Desta maneira, em uma larga ordenação o custo dos fragmentos menores do que sete elementos podem ser ordenados com o *Selection Sort* para alguma economia de processamento.

Referências

CORMEN, T.H.; CHARLES, E.L.; RONALD, L.R; CLIFFORD S.
Algoritmos Teoria e Pratica
Editora: Campus, 2000

MUSSER, David - Measuring Computing Times and Operation Counts of Generic Algorithms
<http://www.cs.rpi.edu/~musser/gp/timing.html>
Acessado em 21/10/09

REBELSKY, Samuel – MergeSort – Fundamentals of CSc.
<http://www.cs.grinnell.edu/~rebelsky/Courses/CS151/2002F/Readings/mergesort.html>
Acessado em 19/10/09

SZWARCFITER, J.L.; MARKENZON, L
Estruturas de Dados e seus Algoritmos.
LTC-Livros Técnicos e Científicos, Rio de Janeiro, RJ, 1994.

WIKIPEDIA. Merge Sort
<http://en.wikipedia.org/wiki/Merge_sort>
Acessado em 23/10/09

WIKIPEDIA. Selection Sort
<http://en.wikipedia.org/wiki/Selection_sort>
Acessado em 23/10/09