UDESC UNIVERSIDADE DO ESTADO DE SANTA CATARINA | JOINVILLE CENTRO DE CIÊNCIAS TECNOLÓGICAS

DCC – Departamento de Ciência da Computação
EDA1001 – Estruturas de Dados I
Profa Everlin F. Costa Marques

Conteúdo: Experimento em Algoritmos de Sort (Ordenação Interna) (junho 2024)

Assume-se um algoritmo sort da bibliografia de referência. O experimento consiste em medir o tempo gasto para ordenar um arquivo. Para os algoritmos de ordenação internas, mede-se a complexidade através do número de comparações entre chaves e do número de trocas entre os itens do arquivo.
Escolha uma métrica, por exemplo, ao observar a tabela de [ZIV], pode-se escolher como métrica o número de comparações entre as chaves (na abordagem dividir para conquistar).

Considerações sobre métodos de ordenação, segundo [ZIV]		
Método de ordenação (classe)	**Algoritmo**	**Característica**
Dividir para conquisar	*Quicksort*, *mergesort* (intercalação), *bubble sort*, *selection sort*	Ordenação dos itens por comparação entre chaves. Custo na ordem de $O(n^2)$
Ordenação digital	*Radixsort/ bucketsort*	*Não usa comparação entre chaves. Demanda de memória pode ser proibitiva. Dificuldade em lidar com a distribuição dos grupos.* Custo na ordem de *O(n).*

Assim, selecionado o algoritmo e a métrica, precisa-se definir o tamanho da amostra.
Ex. 100, 200, 300, 400, ...2000.
Ou 100, 500, 1000, 1500, … 10 k.

É importante repetir o mesmo experimento 10 vezes para gerar uma média.
Por exemplo

Algoritmo Bubble sort

Tamanho da amostra	Rodada (métrica: comparações)										
	1	2	3	4	5	6	7	8	9	10	Média da rodada
100											9.200
200											18.000
300											
400											
500											
600											

Assim, os valores de “tamanho de amostra” geram os valores para o eixo x.
Os valores da “média de rodada”, com o log aplicado, são os valores de y.

Onde o tamanho da amostra significa a quantidade de dados de entrada.
Ex. vetor de 100 inteiros no *bubble sort*.

Por fim, gera-se um gráfico (de linhas) com o tamanho das amostras no eixo x, e no

eixo y plota-se a média das métrica selecionada. Porém, dependendo da mostra, pode ser necessário suavizar a curva para entregar uma leitura de dados informativa pela aplicação de uma função log em todo o y.

As amostras devem ser geradas de forma aleatória para melhor distribuição de probabilidades.

Cada rodada deve ter uma amostra diferente (ex. Rodar o bubble sort para amostras de 100 , deve ter 10 amostras diferentes). Exemplo do que enviei no moodle.

Há lugares para gerar as amostras, para ter uma ideia, como https://site112.com/gerador-numeros-aleatorios. Escolha o limite de números e o valor máximo.

Exemplo de código para o *bubble sort*:

```
#include <stdio.h>
#include <math.h>
#define A 100
int main()
{

    FILE * out;
    int i, j , tmp, ncomp=0, ntroca=0;
    int v[A]={-14220,-17549,-24234,-14872,-23890,19958,30480,-1432,20962,-21965,
-22636,-24804,-9233,-27427,-23112,28034,30386,17456,28149,-20003,
-10774,7367,-4373,-14597,29043,8941,28441,-4580,-17344,27709,
34600,18902,-15845,15472,-23947,-29639,5164,33238,-4762,12537,
-29191,28376,5052,34402,7045,-8568,11030,-8608,2005,-12731,
19791,30882,-23627,11262,34796,20299,-8524,-16574,-20610,-23680,
-7109,30328,-27049,-1952,-5553,-7766,21673,-26576,-8417,-28099,
30291,22992,9714,16708,13297,-6337,15205,-22415,21974,4409,
-6208,887,15142,31586,16360,21755,-25297,8220,-4715,25611,
34729,25796,-20386,-3316,-1141,-10505,-11415,18726,-23132,-1927};
//a amostra deve ser lida de um arquivo para facilitar a mudança de valores no experimento
    for(i=0; i<A;i++ )
      for( j=0; j < A-1; j++){
          ncomp++;
          if ( v[j] > v[j+1]){
               tmp =v[j];
               v[j] = v[j+1];
               v[j+1]= tmp;
               ntroca++;
          }
      }

    if (  (out=fopen("saida100.txt", "a"))!=NULL){
        fprintf(out,"%i comparacoes;%i trocas\n", ncomp, ntroca);
        for(i=0; i< A; i++)
            fprintf(out, "%i,", v[i]);
        fclose(out);
    }

    return 0;
}
```

Referências

[MOR] MORIN, P. Open Data Structures. 2013. Disponível em:<https://www.opendatastructures.org/ods-python-screen.pdf>. Acesso em 15/6/2024.(Tradução própria para esse texto).

[ZIV] ZIVIANI, N. Projeto de Algoritmos com implementações em PASCAL e C. Ed. Thomson, 2ª.ed. São Paulo: 2004.

CORMEN, Thomas, H. *et al*. Algoritmos. Disponível em: Minha Biblioteca, (4th edição). Grupo GEN, 2024.

HUANG, S. O que é a notação Big O: complexidade de tempo e de espaço. Disponível em:<https://www.freecodecamp.org/portuguese/news/o-que-e-a-notacao-big-o-complexidade-de-tempo-e-de-espaco/>. Tradução: Daniel Rosa. Acessado em 17/06/2024.

CORMEN, T.H. Desmistificando Algoritmos. Tradução Arlete Simille Marques. Rio de Janeiro: Elsevier, 2014.