Discente: Ronaldo Ribeiro Porto Filho – 202410131
Data: 29/08/2025

Exercício classe assintótica do algoritmo Shakesort

Pior caso:

Vamos analisar o pior caso do algoritmo Shakesort, o qual representa a notação Big-O do mesmo. Ele ocorre quando o vetor está ordenado de forma decrescente, ou seja, executa trocas em todas as verificações, sendo esse o caso mais custoso computacionalmente, e assim, o seu limite superior de tempo.

```c
void shakesort(int *vet, int tamanho){              // Tempos:
    int aux;                                        // 1
    int esq = 0, dir = tamanho-1;                   // 1

    while(dir > esq){                               // A soma dos tempos dos dois laços 'for'
        for (int i = esq; i < dir; i++){            // Somatório 1
            if (vet[i] > vet[i+1]){                 // 1
                aux = vet[i+1];                     // 1
                vet[i+1] = vet[i];                  // 1
                vet[i] = aux;                       // 1
            }
        }

        for (int i = dir-1; i > esq; i--){          // Somatório 2
            if (vet[i] < vet[i-1]){                 // 1
                aux = vet[i-1];                     // 1
                vet[i-1] = vet[i];                  // 1
                vet[i] = aux;                       // 1
            }
        }

        esq++;                                      // 1
        dir--;                                      // 1
    }
    return;
}
```

Dentro do laço while temos dois laços for, eles podem ser expressos por dois somatórios, e com isso, por duas figuras, vamos destrinchá-los:

Ele verifica se o valor atual daquele índice do vetor é maior que o do próximo, se for, então ele realiza uma troca, e na próxima iteração aumenta o valor da variável local i, fazendo essa verificação ao longo de todo o vetor.

Após verificar todo o vetor ele entra no segundo laço for, que começa a verificar se o valor do índice atual é menor que o valor do índice anterior, se for, ele realiza uma troca, diminuindo o valor da variável local i a cada iteração, porém, esse laço começa desconsiderando o último índice do vetor, porque o mesmo já está com o valor correto por conta do último ‘for’.

Dessa forma, após essas duas etapas, as variáveis de controle, que indicam onde os laços for começarão nas próximas vezes, são atualizadas, sendo a que aponta pra esquerda do vetor incrementada, e a que aponta pra direita do vetor decrementada.

Assim, achamos um padrão no somatório desses laços:

Primeiro:

(n - 1) + (n - 3) + (n - 5) + ... + 1 =

Porque na primeira iteração do laço while, o primeiro for verificará um total de (n - 1) vezes, se o vetor possui n índices, então o total de comparações possíveis são n -1, e na próxima, ele verificará (n - 3), pois as variáveis de controle serão atualizadas, e assim por diante, até por fim verificar somente 1 vez.

Segundo:

(n - 2) + (n - 4) + (n - 6) + ... + 2 =

Porque assim como no primeiro laço esse também verificará (n - 1) vezes, porém, desconsiderando o último índice, que já está com o valor correto, ou seja, (n - 1 - 1) = (n - 2), e após a atualização das variáveis de controle, (n - 4), pois não verificamos mais dois índices, e assim por diante, até por fim verificar somente 2 vezes, sendo que na última iteração do while esse for não fará nenhuma verificação.

Agora, podemos representar esses somatórios como figuras, usando n = 8 como exemplo:

Primeiro:

x x x x x x x

x x x x x

x x x

x

Segundo:

x x x x x x

x x x x

x x

Somando ambos, achamos a figura completa do laço while, ou seja, a complexidade de tempo do algoritmo:

x x x x x x x

x x x x x x

x x x x x

x x x x

x x x

x x

x

Podemos descobrir a soma dos laços for a partir dessa representação, que pode ser escrita como: 1 + 2 + 3 + 4 + ... + (n - 1), uma progressão aritmética, e utilizando a fórmula de sua soma temos:

$$S = \frac{(n-1)*(1+n-1)}{2}$$

$$S = \frac{(n-1)*n}{2}$$

$$S = \frac{n^2-n}{2}$$

Assim, encontramos que a complexidade de tempo (classe assintótica) do pior caso do algoritmo Shakesort é $O(n^2)$.

Caso Médio:

Vamos analisar a complexidade de tempo do caso médio do algoritmo Shakesort. Nesse caso, não consideramos um vetor ordenado decrescentemente e nem o inverso, consideramos um vetor ordenado aleatoriamente.

Novamente, precisamos encontrar o tempo do laço while para descobrir sua classe assintótica, a partir da soma dos dois laços for, que podem ser expressos como aqueles mesmos somatórios do pior caso, pois o vetor está em ordem aleatória, e assim, terá a mesma complexidade de tempo da análise anterior.

Assim, encontramos que a complexidade de tempo (classe assintótica) do caso médio do algoritmo Shakesort é $O(n^2)$.

Melhor Caso:

Vamos analisar a complexidade de tempo do melhor caso do algoritmo Shakesort. Nesse caso, consideramos um vetor ordenado crescentemente, pois dessa forma, não realizamos nenhuma troca no código, a estrutura já está organizada da forma como queríamos.

Porém, o meu código não possui uma flag para indicar se houve ou não troca, pois dessa maneira, ele sairia do laço while logo após verificar o vetor pela primeira vez, e não entraria no segundo laço for. Ou seja, nesse caso, sua complexidade de tempo seria de $O(n)$, pois ele percorreria o vetor apenas uma vez fazendo $(n - 1)$ comparações.

Então, como meu código não possui essa flag, apesar de ele não realizar nenhuma troca, ele ainda percorrerá todo o vetor diversas vezes, fazendo suas comparações assim como o pior caso e o caso médio, podendo ser representado pelos mesmos somatórios, e assim, tendo a mesma complexidade de tempo $O(n^2)$.