

Análise de Algoritmos  
Projeto 01 - Ordenação por Seleção de Raiz Quadrada  
Bacharelado em Ciência da Computação

Prof. Daniel Saad Nogueira Nunes



# 1 Ordenação por seleção de raiz quadrada

O método de ordenação por seleção de raiz quadrada,  $\text{SQRTSORT}(V)$ , funciona da seguinte maneira:

1. Divida, logicamente, o vetor de entrada em várias partes  $V_1, V_2, \dots, V_k$ , cada uma de tamanho  $\sqrt{n}$ . Caso o tamanho do vetor não seja múltiplo de  $\sqrt{n}$ , a última parte terá tamanho  $(n \bmod \sqrt{n})$ .
2. Encontre o maior elemento  $e_i$  de cada parte  $V_i$ , com  $1 \leq i \leq k$ .
3. Pegue o maior elemento dentre  $e_1 \dots e_k$  e o insira no vetor solução. Se o maior elemento for o  $e_j$ , então ele deve ser descartado da parte  $V_j$ .
4. Repita os passos de 2 a 3 até que todas as partes estejam vazias.

Claramente o tempo do  $\text{SQRTSORT}$  depende de como os passos 2 e 3 são executados a cada iteração. Este projeto visa comparar duas possíveis formas de implementar o  $\text{SQRTSORT}$ :

- Empregando um método quadrático de ordenação.
- Utilizando a estrutura de dados Heap.

## Utilizando um método quadrático de ordenação

Caso cada parte  $V_i$  seja ordenada com um método quadrático de ordenação, como o  $\text{BUBBLESORT}$  ou o  $\text{INSERTIONSORT}$ , podemos descobrir qual é o maior elemento de cada parte em tempo constante, restando apenas descobrir qual é o maior entre todas as partes a cada iteração, o que pode ser obtido através de uma simples comparação entre todos os elementos  $e_1, \dots, e_k$ .

## Utilizando uma Heap

A heap, estrutura central do  $\text{HEAPSORT}$ , é uma estrutura de dados que permite concluir qual o maior elemento, dentre uma sequência de  $m$  elementos, em tempo eficiente. Ela possui as seguintes operações:

- $\text{MAKEHEAP}(V)$ : transforma um vetor  $V[1, m]$  em uma heap. Esta operação custa  $\Theta(m)$  e é necessária para utilizar qualquer operação em uma heap.
- $\text{INSERTHEAP}(V, x)$ : insere um elemento  $x$  em uma heap de tamanho  $m$ . Esta operação tem custo  $\Theta(\lg m)$ .
- $\text{REMOVEHEAP}(V)$ : remove o maior elemento da heap e o retorna, além de reestruturar a heap. Em outras palavras, se  $x$  for o maior elemento da heap, ele é retirado da heap e o próximo elemento  $y \leq x$  é colocado no topo da heap, implicando que será o próximo a ser retirado caso nenhum outro elemento  $z > y$  seja inserido antes.

## 2 Análise

Deverão ser analisadas as duas implementações do SQRTSORT, de acordo com o feramental visto no curso e também através da análise empírica, a qual deverá corroborar com a análise em termos das notações assintóticas.

### Análise dos Algoritmos

A análise das duas implementações do SQRTSORT, em relação ao pior caso, deverá ser realizada, e uma cota, em termos da notação  $\Theta$ , deverá ser determinada para cada implementação.

### Análise Empírica

A fim de corroborar a análise dos algoritmos, cada um deles deverá ser implementado em uma linguagem de programação de livre escolha e o seu tempo para ordenar uma entrada aleatória de tamanho  $n \in [10^4, 10^5, 10^6, 10^7]$  deverá ser aferido. Sugere-se executar cada procedimento diversas vezes e tomar a média dos tempos de execução para uma medida estatisticamente mais relevante.

## 3 Relatório Técnico

Um relatório técnico com os resultados deverá ser escrito. Uma sugestão de organização do relatório é a seguinte:

- Introdução: contextualiza o problema a ser resolvido.
- Ordenação por seleção de raiz quadrada: descreve o método de seleção por raiz quadrada de maneira breve com exemplos próprios bem como as diferentes possibilidades de implementação do mesmo.
- Análise dos Algoritmos: analisa cada uma das duas implementações em termos da notação assintótica  $\Theta$ .
- Análise Empírica: apresenta os resultados empíricos em relação às implementações do SQRTSORT em formato de gráfico (um ponto para cada tamanho de entrada), além de descrever a metodologia para obtenção dos dados.
- Discussão: realiza uma discussão entre a análise dos algoritmos e os dados empíricos obtidos a fim de confrontar a cota estabelecida com a prática.
- Considerações finais: realiza a discussão sobre os resultados observados e apresenta as considerações finais de acordo com os objetivos do projeto.

O relatório deverá utilizar o modelo de artigos L<sup>A</sup>T<sub>E</sub>X da Sociedade Brasileira de Computação.

## 4 Considerações

- Este projeto pode ser executado individualmente ou em dupla.
- Detecção de plágio automaticamente acarretará nota 0 para os envolvidos. Medidas disciplinares também serão tomadas.
- O trabalho deve ser entregue dentro de uma pasta zipada com a devida identificação do(s) aluno(s) através da sala de aula virtual da disciplina.
- Os códigos utilizados para implementação dos métodos e execução dos experimentos devem ser anexados junto ao relatório.

### 4.1 Suporte Experimental

- Para automatização dos experimentos, podem ser utilizadas linguagens de *script*, como por exemplo: **Python**, **Perl** e **Bash**.
- A maioria das linguagens de programação modernas possuem mecanismos para mensurar o tempo de execução entre dois trechos de códigos. Estes mecanismos podem ser utilizados para mensurar o tempo. O mesmo pode ser dito sobre a geração aleatória de números.
- É interessante compilar (caso se aplique) os códigos-fonte com as *flags* de otimização, para que o programa possa executar mais rápido.