CEFET-MG - Campus II
Departamento de Engenharia de Computação
Laboratório de Algoritmos e Estruturas de Dados II

Prática IV

Implementação em JAVA do TAD Heaps

Alunos: Antônio Augusto Diniz Sousa
Professor orientador: Thiago de Souza Rodrigues

10 de Setembro
2018

CEFET-MG - Campus II
Departamento de Engenharia de Computação
Laboratório de Algoritmos e Estruturas de Dados II

Prática IV

Implementação em JAVA do TAD Heaps

Relatório da prática IV apresentado à Disciplina de Laboratório de Algoritmos e Estruturas de Dados II do Curso de Engenharia de Computação do Centro Federal de Educação Tecnológica de Minas Gerais, como requisito parcial para conclusão da disciplina.

Alunos: Antônio Augusto Diniz Sousa

Professor orientador: Thiago de Souza Rodrigues

10 de Setembro
2018

Sumário

1 Apresentação

A tarefa dessa prática foi testar o comportamento do algoritmo *HeapSort* com diversos vetores, alterando tanto o tamanho quanto sua organização. Utilizou-se a implementação do Autor Ziviani, disponível nesse link ou acessado diretamente em `http://www2.dcc.ufmg.br/livros/algoritmos-java/implementacoes-05.php`.

Utilizando a linguagem *Java*, criou-se 30 vetores, sendo 10 deles utilizando ordem crescente, outros 10 utilizando ordem decrescente e os últimos 10, utilizando números aleatórios. Dentre os dez vetores de cada organização (crescente, decrescente ou aleatório), o primeiro possuía 10000 elementos, o segundo 20000, e assim por diante, até o 10°, com 100000 elementos.

Analisou-se diante desses vetores o número de comparações necessárias para ordená-los. Os resultados seguem adiante nesse relatório e os códigos fontes podem ser acessados clicando aqui ou acessando em `https://gitlab.com/antonioaads/LAEDII`.

2 Descrição de atividades

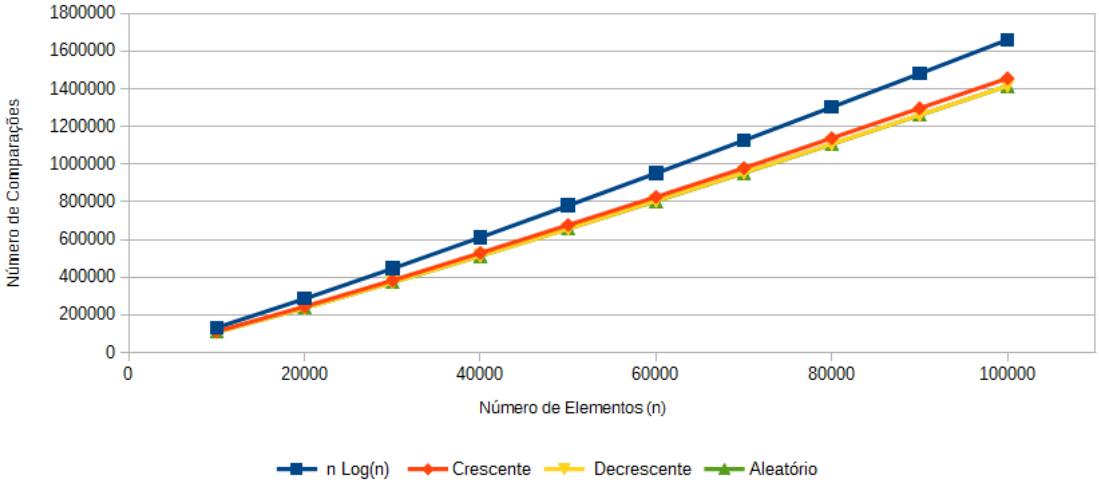
Após a execução do código, que não iremos tratar especificamente nesse relatório, pois o mesmo se encontra no repositório citado na introdução devidamente comentado e organizado, tirou-se os resultados da Tabela 1.

Tabela 1: Resultados

Itens	$nlog_2(n)$	Crescente	Decrescente	Aleatório
	Comparações	Comparações	Comparações	Comparações
10000	132877	112281	108373	108309
20000	285754	244543	236727	236524
30000	446180	383173	372735	372109
40000	611508	529069	513403	513105
50000	780482	677730	657449	657916
60000	952360	826393	805867	804212
70000	1126655	979636	955446	953622
80000	1303017	1138583	1107933	1106246
90000	1481187	1297508	1261544	1260405
100000	1660964	1456431	1415979	1415849

Utilizou-se a ferramenta *LibreOffice Calc* para plotar o gráfico ilustrados na Figura 1.

Figura 1: Representação do gráfico n x Número de comparações



2.1 Análise

O *HeapSort* é um algoritmo bastante eficiente quanto a complexidade, sendo $O(nlogn)$ para todos os casos. Nos testes executados, conforme gráfico ilustrado na Figura 1, o número de comparações realmente se aproximou de uma onda $nlog_2(n)$, mantendo sempre um pouco abaixo, o que comprova que sua complexidade é $O(nlogn)$ para quaisquer organização de dados.

Como a complexidade é a mesma para todos os casos, as ondas praticamente se sobrescreveram, pois o número de comparações é bem próximo.

3 Conclusão

O algoritmo *HeapSort* possui uma complexidade interessante e constante para todos os tipos de entrada, o que o torna um algoritmo muito estável para aplicações que possuí entradas variadas.

Por isso e pela fácil implementação, o *HeapSort* é um dos algoritmos de ordenação mais utilizados no mundo, o que torna fundamental esse estudo sobre o mesmo.