CI1056: Algoritmos e Estruturas de Dados II

Prof. Dr. Marcos Castilho

Departamento de Informática/UFPR

8 de dezembro de 2020

Resumo

Um panorama sobre algoritmos de ordenação (parte 2)

Objetivos da aula

- Discutir algoritmos de ordenação de forma geral (parte 2)

Motivação

- Quais os aspectos teóricos ou práticos de destaque?
- O que mais pode ser estudado sobre este assunto?

Comentários iniciais

- Nesta aula não farei uma análise completa e correta dos temas
- Como o título diz é um panorama do assunto
- Então para que serve se não é formal o suficiente?

Algoritmos de ordenação

- O site `https://www.geeksforgeeks.org` hoje apresenta 38 algoritmos de ordenação
- Vimos até agora meia duzia deles...
- Esta meia dúzia são os algoritmos clássicos
- E os outros, vamos estudar todos? Não!
- Pelos motivos apresentados na aula anterior
- Vocês saberão se um algoritmo é ou não é bom

Sobre a importância de Algoritmos 2

- Muito do que se falará a seguir são dicas
- Mas *vocês têm condições de entender na literatura!*

Exemplos ou contra-exemplos?

- Na verdade, existem muita piadas sobre algoritmos de ordenação
- Até que ponto uma destas piadas é séria?
- Até que ponto você acha graça destas piadas?
  - Bozosort
  - Sleepsort
  - Miraclesort

Spoiler

- Dado um vetor qualquer, temos um resultado provado matematicamente:
  - Não é possível, *no caso geral* ordenar um conjunto de elementos com menos de $n.log_2(n)$ comparações
  - Mas, para situações absolutamente especiais é possível ordenar com custo linear (alguma aula próxima veremos alguns)

Os algoritmos estudados

- algoritmos "ingênuos"
  - selectionsort
  - insertionsort
  - bublesort (exercício, ele é quadrático, mas bem famoso)
- algoritmos "espertos"
  - mergesort
  - quicksort
  - heapsort

O que vamos ver?

- Custo de comparações (já vimos)
- Custo do número de trocas dos elementos
- Estabilidade
- Uso de memória
- Algum outro critério

O que é preciso saber antes de começar?

- Análise teórica é diferente da prática
- Tudo depende:
  - Do seu hardware
  - Do seu sistema operacional
  - Da linguagem de programação
  - Do seu compilador
- Mas a análise matemática é a base segura
- É possível, após análise matemática, *estimar* o tempo de execução

Custo de comparações

- Vimos dois tipos de algoritmos:
  - os "ingênuos" (n^2)
    - `insertionsort`
    - `selectionsort`
  - os "espertos" ($n.log_2(n)$)
    - `mergesort`
    - `quicksort`
    - `heapsort`
- Até quando a complexidade teórica é relevante?

Algoritmos derivados

- Vários outros algoritmos têm base nestes:
- Exemplos:
  - `shellsort`: aperfeiçoamento do `insertionsort`
  - `heapsort`: aperfeiçoamento do `selectionsort`
  - `3-way-quicksort`: aperfeiçoamento do `quicksort`
  - `3-way-mergesort`: aperfeiçoamento do `mergesort`
  - e por ai vai...

Observação

- Tudo o que for falado daqui para frente exige estudo!
- Vejam a literatura!!!

Variante do *insertionsort*

- Denominado `shellsort`
- Se preocupa com o pior caso do `insertionsort` que são os elementos "menores" nas últimas posições
- Faz uma iteração adicional controlando o "passo"
- O mais comum é iniciar o passo em $n/2$ depois em $n/4$ até 1
- Sedgewick afirma algo diferente, algo em torno de $2/3$

Fim do tópico

- O conteúdo desta aula está basicamente no livro do Sedgewick, capítulo 2 (base) e em vários outros. . . . Também está espalhado por diversos outros livros, tais quais o do Cormem, na "biblia" do Knuth, ou até mesmo em alguns sites confiáveis da Internet (quando se tratar de implementações). . .

Licença

- Slides feitos em LaTeX usando beamer
- Licença

  Creative Commons Atribuição-Uso Não-Comercial-Vedada a Criação de Obras Derivadas 2.5 Brasil License.`http://creativecommons.org/licenses/by-nc-nd/2.5/br/`

  Creative Commons Atribuição-Uso Não-Comercial-Vedada a Criação de Obras Derivadas 2.5 Brasil License.`http://creativecommons.org/licenses/by-nc-nd/2.5/br/`