Universidade Federal de Goiás – Instituto de Informática

Análise e Projeto de Algoritmos

Introdução

Diogo Stelle
com slides do Prof. Fábio H. V. Martinez
diogostelle@inf.ufg.br / diogo.stelle@gmail.com

2019

Conteúdo da Aula

- Algoritmos
- Algoritmos como uma Tecnologia
- Exercícios

Algoritmos

Definição Inicial

- um **algoritmo** é um processo computacionalmente bem definido que recebe um conjunto de valores como **entrada** e produz um conjunto de valores como **saída**
- um algoritmo é uma ferramenta para solucionar um problema computacional bem definido
- a definição do problema especifica em termos gerais a relação desejada entre a entrada e a saída
- o algoritmo descreve um processo computacional específico para obter essa relação entre a entrada e a saída

Ordenação

- **PROBLEMA DA ORDENAÇÃO**
  - **Entrada:** uma sequência de **n** números $\{a_1; a_2;...; a_n\}$
  - **Saída:** uma permutação $\{a'_1; a'_2;...; a'_n\}$ da sequência de entrada de tal forma que $a'_1 \leq a'_2 \leq ... \leq a'_n$
- por exemplo, dada a sequência de entrada {82; 71; 63; 18; 44; 97; 56}, um algoritmo de ordenação devolve como saída a sequência {18; 44; 56; 63; 71; 82; 97}
- essa sequência de entrada é chamada uma **instância do problema** de ordenação
- uma instância de um problema consiste da entrada necessária para computar a solução do problema

Qual o melhor?

- ordenação é uma operação fundamental em Computação e, por isso, muitos bons algoritmos de ordenação foram desenvolvidos
- qual algoritmo é o melhor para uma dada aplicação?
- nesse problema, a resposta depende de quantos itens devem ser ordenados, de quantos já estão ordenados, do tamanho dos itens e “onde” serão ordenados

Correção

- um algoritmo é dito correto se, para toda instância de entrada, o algoritmo para com uma resposta **correta**
- neste caso, dizemos que o algoritmo **resolve** o problema computacional
- observe que um algoritmo incorreto pode não parar com algumas instâncias de entrada ou pode parar, mas com respostas diferentes daquelas desejadas
- um algoritmo pode ser especificado em sentenças de língua portuguesa, em sentenças de uma linguagem de programação ou mesmo como um dispositivo de hardware
- a única restrição é que tal especificação deve fornecer uma descrição precisa do processo computacional a ser seguido

Exemplos de problemas computacionais

- **Tipos de problemas que podem ser resolvidos com algoritmos**
    - Projeto Genoma Humano: sequenciamento do genoma humano, determinação dos genes do DNA, etc;
    - Internet: acesso rápido e recuperação, gerenciamento e manipulação de grandes quantidades de informação, etc;
    - Comércio eletrônico: manutenção de informações sigilosas, etc;
    - Empresas e indústrias: otimização de recursos e lucros, etc;
    - entre outros...

Organização da informação

- **Estruturas de dados**
  - os algoritmos que veremos trabalham com diversas estruturas de dados
  - uma estrutura de dados é uma forma de armazenar e organizar informações com objetivo de facilitar o acesso e as modificações
  - não é possível usar uma única estrutura de dados para todos os propósitos e, portanto, é importante conhecer as potencialidades
  - e limitações das diversas estruturas de dados que trabalharemos

Algoritmos eficientes X problemas intratáveis

- **Problemas difíceis**
  - a esmagadora maioria dos problemas que estudamos na graduação podem ser solucionados com algoritmos eficientes
  - para alguns problemas computacionais, no entanto, não são conhecidos algoritmos eficientes
  - um importante subconjunto desses problemas é conhecido como conjunto dos problemas NP-completos
  - problemas desta classe surgem muito frequentemente na prática

Algoritmos como uma Tecnologia

Computadores atuais e futuros

- e se os computadores fossem infinitamente rápidos e a memória dos computadores fosse um recurso sem custo?
- ainda teríamos de estudar os algoritmos pelo menos para mostrar que nossas soluções terminam e com a resposta correta
- computadores podem ser muito rápidos, mas não infinitamente rápidos; assim como memória pode ser barata, mas não grátis
- dessa forma, tempo e espaço computacional são recursos limitados

Eficiência de algoritmos

- **Eficiência**
  - algoritmos projetados para solucionar o mesmo problema computacional diferem dramaticamente em termos de eficiência
  - essas diferenças podem ser muito mais significativas que aquelas devido apenas ao hardware ou ao software
  - como exemplo, vamos lembrar de dois algoritmos de ordenação que já vimos: **ordenação por inserção** e **ordenação por intercalação**

Eficiência de algoritmos

- **Eficiência**
  - o algoritmo de ordenação por inserção tem, em termos gerais, tempo de execução igual a $\mathbf{c_1 n^2}$ para rearranjar um vetor com **n** itens, onde $\mathbf{c_1}$ é uma constante positiva que não depende de **n**
  - o algoritmo de ordenação por intercalação tem tempo de execução igual a $\mathbf{c_2\ n\ lg\ n}$, onde **lg n** é na verdade $\mathbf{\log_2 n}$ e $\mathbf{c_2}$ é outra constante que não depende de **n**

Eficiência de algoritmos

- **Eficiência**
  - um algoritmo de ordenação por inserção tem, em geral, menor fator constante que um algoritmo de ordenação por intercalação;
  - isto é, $c_1 < c_2$ em geral , mas os fatores constantes são frequentemente menos significativos em termos do tempo de execução que o tamanho da entrada **n**
  - para pequenas instâncias do problema, um algoritmo de ordenação por inserção pode ser mais rápido que um algoritmo de ordenação por intercalação, mas quando **n** torna-se suficientemente grande, o algoritmo de ordenação por intercalação se transforma no algoritmo mais rápido, independentemente das constantes c_1 e c_2

Comparação de algoritmos

- **Eficiência**
  - suponha que um computador A muito rápido esteja executando o algoritmo de ordenação por inserção e um computador B muito lento esteja executando o algoritmo de ordenação por intercalação
  - suponha que cada algoritmo tem de ordenar um milhão de números
  - suponha que o computador A execute um bilhão de instruções por segundo e que o computador B execute dez milhões de instruções por segundo; isso significa que o computador A é 100 vezes mais rápido que o computador B

Comparação de algoritmos

- **Eficiência**
  - suponha que um dos melhores programadores do mundo desenvolveu um programa para a ordenação por inserção, cujo código resultante necessita de $\mathbf{2n^2}$ instruções para ordenar **n** itens
  - a ordenação por intercalação, no entanto, foi implementada por um programador mediano em uma linguagem de programação que possui um compilador ineficiente, com código resultante usando **50n lg n** instruções para ordenar **n** itens

Comparação de algoritmos

- **Eficiência**
  - para ordenar um milhão de números o computador A leva

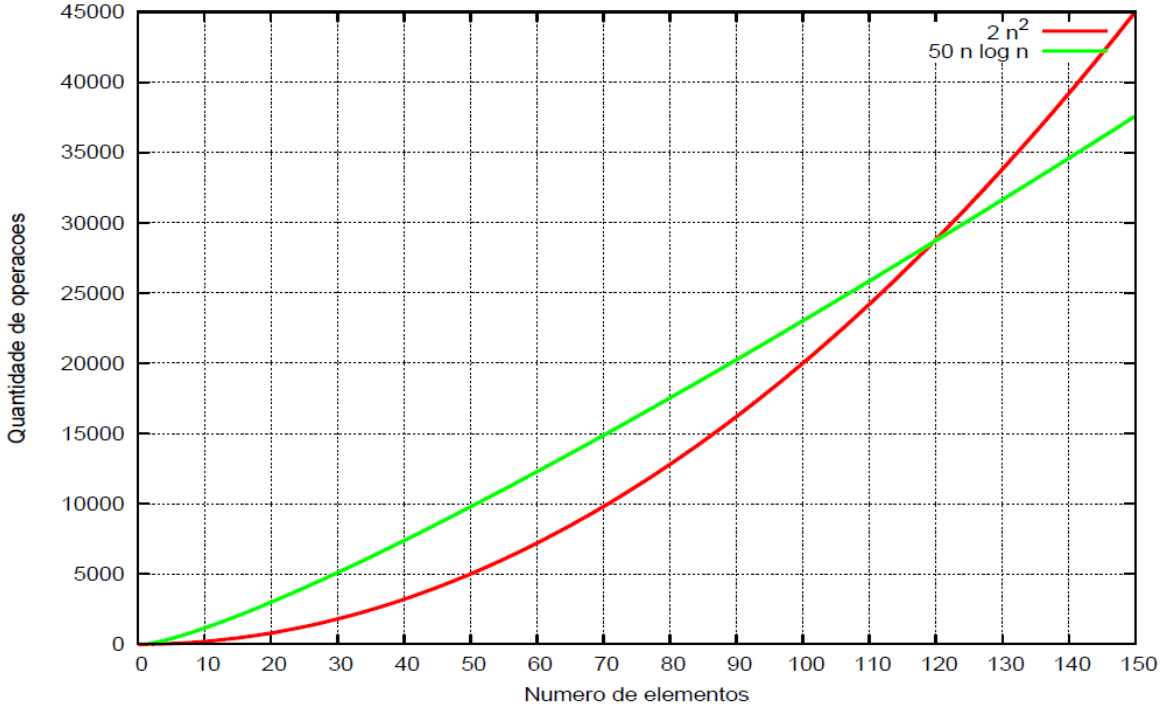
$$\frac{2 \cdot (10^6)^2 \ \ instruções}{10^9 \ instruções \ por \ segundo} = 2000 \text{ segundos}$$

  - enquanto que o computador B leva

$$\frac{50 \cdot (10^6) \log 10^6 \ \ instruções}{10^7 \ instruções \ por \ segundo} \approx 100 \text{ segundos}$$

  - usando um algoritmo cujo tempo de execução cresce mais lentamente, mesmo com um compilador ruim, a solução apresentada no computador B é **20 vezes** mais rápida que aquela apresentada no computador A!

Comparação de algoritmos



Pesquisa em algoritmos

- **Algoritmos e outras tecnologias**
  - algoritmos, assim como hardware, são uma tecnologia
  - o desempenho total de um sistema depende da escolha de algoritmos eficientes, da mesma forma que depende da escolha de computadores rápidos
  - assim como rápidos avanços são feitos em muitas tecnologias computacionais, também são feitos em algoritmos

Pesquisa em algoritmos

- **Algoritmos e outras tecnologias**
  - uma sólida base de conhecimento de algoritmos é uma característica que separa os programadores hábeis dos novatos
  - com um computador moderno você pode realizar algumas tarefas mesmo sem saber muito sobre algoritmos, mas com um bom conhecimento de algoritmos você pode fazer muito mais

Exercícios

Exercícios

- 1.1-2 Além da rapidez, quais outras medidas de eficiência podemos usar em aplicações computacionais reais?

- 1.1-3 Selecione uma estrutura de dados que você já viu anteriormente e discuta seus pontos fortes e fracos

- 1.2-2 Suponha que estamos comparando implementações do algoritmo de ordenação por inserção e do algoritmo de ordenação por intercalação implementados no mesmo computador. Para entradas de tamanho **n**, o algoritmo de ordenação por inserção usa $\mathbf{8n^2}$ passos enquanto que o algoritmo de ordenação por intercalação usa **64n lg n** passos. Para quais valores de **n** o algoritmo de ordenação por inserção é melhor que o algoritmo de ordenação por intercalação?

Exercícios

- 1.2-3 Qual o menor valor de **n** tal que um algoritmo cujo tempo de execução é $\mathbf{100n^2}$ é mais rápido que um algoritmo cujo tempo de execução é $\mathbf{2^n}$ no mesmo computador?

- 1-1 Comparação de tempos de execução Para cada função ***f (n)*** e tempo ***t*** na tabela a seguir, determine o maior tamanho ***n*** de um problema que pode ser resolvido em tempo ***t***, considerando que o algoritmo para resolver o problema gasta ***f (n)*** microssegundos.

Exercícios

	1 segundo	1 minuto	1 hora	1 dia	1 mês	1 ano	1 século
$\lg n$							
$\sqrt{n}$							
n							
$n \lg n$							
n^2							
n^3							
2^n							
$n!$							