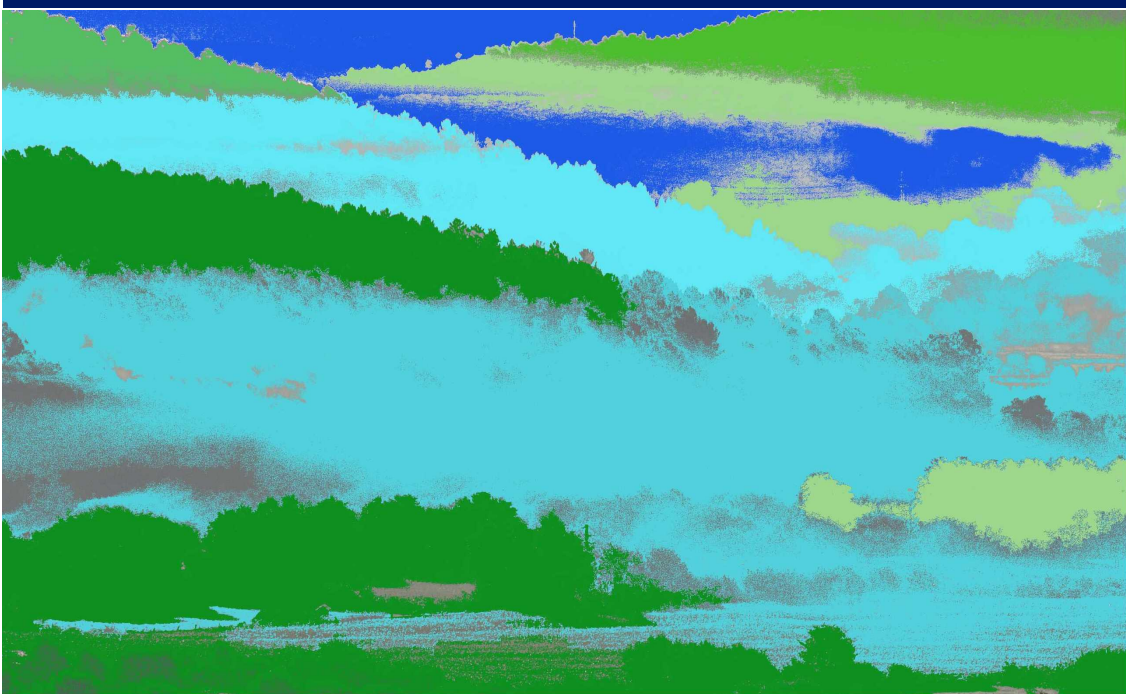


---

# Tópicos Avançados em Algoritmos

Armando Matos



2008

Departamento de Ciência de Computadores  
Faculdade de Ciências da Universidade do Porto



# Conteúdo

<b>0</b>	<b>Introdução</b>	<b>9</b>
<b>1</b>	<b>Preliminares: fundamentos da análise de algoritmos</b>	<b>13</b>
1.1	Eficiência dos algoritmos . . . . .	13
1.1.1	Eficiência dos algoritmos: duas análises . . . . .	13
1.1.2	Recursos e modelos de computação . . . . .	14
1.1.3	Pior caso e caso médio . . . . .	14
1.1.4	Recorrências, pior caso e caso médio, um exemplo . . . . .	15
1.2	Ordens de grandeza . . . . .	17
1.2.1	Majoração, minoração e ordem exacta . . . . .	17
1.2.2	Tempo de execução em algoritmos de ordenação e pesquisa . . . . .	20
1.3	Solução de recorrências . . . . .	21
1.3.1	Exemplos de funções definidas através de recorrências – definições indutivas . . . . .	21
1.3.2	O que é uma recorrência . . . . .	22
1.3.3	Método Tabelar $\rightarrow$ suspeitar $\rightarrow$ demonstrar . . . . .	24
1.3.4	Método das diferenças finitas constantes . . . . .	26
1.3.5	Método da mudança de variável . . . . .	28
1.3.6	Método da equação característica homogénea . . . . .	28
1.3.7	Equação característica homogénea – raízes distintas . . . . .	29
1.3.8	Equação característica homogénea, caso geral: existência de raízes múltiplas . . . . .	30
1.3.9	Método da equação característica não homogénea . . . . .	30
1.4	Um exemplo de análise: tempo médio do “quicksort” . . . . .	32
<b>2</b>	<b>Tempo de execução dos algoritmos – complementos</b>	<b>35</b>
2.1	Tempo de execução . . . . .	35
2.2	Sobre os modelos de computação . . . . .	36

2.2.1	O modelo externo dos dados . . . . .	37
2.2.2	Monotonia de $t(n)$ . . . . .	38
2.3	Análise amortizada de algoritmos . . . . .	39
2.3.1	“Stack” com gestão de memória . . . . .	40
2.3.2	A função potencial . . . . .	42
2.3.3	Outro exemplo, um contador binário . . . . .	44
2.3.4	Contador binário com custo exponencial na ordem do bit . . . . .	45
<b>3</b>	<b>Sobre o esquema “Dividir para Conquistar”</b>	<b>49</b>
3.1	Uma recorrência associada ao esquema “dividir para conquistar” . . . . .	49
3.2	Multiplicar mais rapidamente. . . . .	51
3.2.1	Multiplicação de inteiros . . . . .	51
3.2.2	Multiplicação de matrizes . . . . .	54
<b>4</b>	<b>Algoritmos aleatorizados e classes de complexidade</b>	<b>57</b>
4.1	Um problema de coloração . . . . .	57
4.2	O produto de duas matrizes iguala uma terceira? . . . . .	59
4.3	O “quick sort” . . . . .	60
4.3.1	Esquema básico do “quick sort” . . . . .	60
4.3.2	Análise no pior caso do “quick sort” clássico. . . . .	61
4.3.3	Análise do tempo médio do “quick sort” clássico . . . . .	61
4.3.4	O “quick sort” aleatorizado . . . . .	63
4.4	Técnica de redução da probabilidade de erro . . . . .	65
4.5	Outro algoritmo aleatorizado: o algoritmo de Rabin-Miller . . . . .	66
4.5.1	Ineficiência dos algoritmos elementares de primalidade . . . . .	66
4.5.2	Existem algoritmos polinomiais para a primalidade . . . . .	67
4.5.3	Testemunhos rarefeitos da não primalidade . . . . .	67
4.5.4	Testemunhos frequentes da não primalidade . . . . .	68
4.6	Computação aleatorizada: classes de complexidade . . . . .	72
4.6.1	Panorama geral das classes de complexidade . . . . .	72
4.6.2	Classes de complexidade aleatorizadas . . . . .	73
<b>5</b>	<b>Sobre a ordenação e a selecção</b>	<b>77</b>
5.1	Quando o universo é pequeno: indexação nos valores . . . . .	79
5.1.1	Vector sem elementos repetidos . . . . .	79

5.1.2	Comentário: uma representação de conjuntos . . . . .	80
5.1.3	Vector com elementos repetidos . . . . .	81
5.1.4	Notas sobre as tentativas de generalização do universo . . . . .	83
5.1.5	Ordenação de reais no intervalo $[0, 1)$ . . . . .	83
5.2	Métodos de ordenação baseados na representação dos valores . . . . .	85
5.2.1	“Radix sort”: começando pelo símbolo mais significativo . . . . .	85
5.2.2	“Radix sort”: começando pelo símbolo menos significativo . . . . .	86
5.3	Mediana; selecção . . . . .	88
5.3.1	Mediana em tempo médio $O(n)$ . . . . .	89
5.3.2	Mediana em tempo $O(n)$ (pior caso) . . . . .	91
<b>6</b>	<b>Circuitos e redes de ordenação</b>	<b>95</b>
6.1	Circuitos . . . . .	95
6.1.1	Classes de complexidade associadas ao modelo dos circuitos . . . . .	98
6.2	Redes de comparação e redes de ordenação . . . . .	98
6.2.1	Introdução e conceitos fundamentais . . . . .	98
6.2.2	Princípio 0/1 . . . . .	102
6.2.3	Ordenadores bitónicos . . . . .	103
6.2.4	Rede de ordenação baseada no “merge sort” . . . . .	105
6.2.5	Sumário, complexidade e minorantes . . . . .	107
<b>7</b>	<b>“Hash” universal e perfeito</b>	<b>111</b>
7.1	Considerações gerais sobre os métodos de “hash” . . . . .	111
7.1.1	Universos grandes, funções de “hash” . . . . .	111
7.1.2	Variantes do método de “hash” . . . . .	114
7.2	“Hash” universal: aleatorização do “hash” . . . . .	115
7.2.1	O método matricial de construção . . . . .	116
7.3	“Hash” perfeito . . . . .	118
7.3.1	Construção com espaço $O(n^2)$ . . . . .	118
7.3.2	Construção com espaço $O(n)$ . . . . .	119
7.4	Contar o número de elementos distintos . . . . .	121
<b>8</b>	<b>Programação Dinâmica: complementos</b>	<b>123</b>
8.1	Introdução . . . . .	123
8.2	Alguns exemplos . . . . .	124

8.2.1	Parentização óptima de um produto matricial . . . . .	125
8.2.2	Máxima sub-sequência comum . . . . .	129
8.2.3	Problema da mochila (“knapsack problem”) . . . . .	134
8.3	Comentário final . . . . .	136
<b>9</b>	<b>Sobre o algoritmo FFT</b>	<b>137</b>
9.1	Transformações de representação, generalidades . . . . .	137
9.2	Polinómios em corpos. Representações . . . . .	137
9.2.1	Cálculo de um polinómio num ponto . . . . .	137
9.2.2	Dois modos de representar um polinómio . . . . .	138
9.2.3	Multiplicação de 2 polinómios . . . . .	139
9.2.4	Corpos . . . . .	140
9.2.5	Raízes primitivas da unidade . . . . .	141
9.3	A DFT: dos coeficientes para os valores . . . . .	142
9.3.1	O que é a transformada discreta de Fourier, DFT? . . . . .	142
9.3.2	A inversa da transformada discreta de Fourier . . . . .	143
9.4	O algoritmo FFT . . . . .	145
9.4.1	Análise da eficiência do algoritmo FFT . . . . .	146
9.5	Aplicações . . . . .	147
9.5.1	Multiplicação eficiente de matrizes . . . . .	150
9.5.2	Transformadas tempo $\leftrightarrow$ frequência . . . . .	150
<b>10</b>	<b>Notas sobre minorantes de complexidade</b>	<b>153</b>
10.1	Exemplos introdutórios . . . . .	154
10.1.1	Um problema simples . . . . .	154
10.1.2	O problema das 12 bolas . . . . .	156
10.2	Entropia, informação e minorantes de complexidade . . . . .	157
10.2.1	Introdução . . . . .	157
10.2.2	Informação e os problemas de pesagens . . . . .	158
10.3	Minorantes de algoritmos de ordenação . . . . .	159
10.4	Algoritmos de ordenação em que o custo é o número de trocas . . . . .	161
10.5	Minorantes de algoritmos de determinação do maior elemento . . . . .	163
10.6	Determinação do segundo maior elemento . . . . .	164
10.7	Minorantes do problema de “Merge” . . . . .	166
10.8	Conectividade de grafos . . . . .	167

---

<b>11 Apêndices</b>	<b>169</b>
11.1 Apêndice: Alguns resultados úteis . . . . .	169
11.2 Apêndice: Implementações do “quick sort” em Haskell, Prolog e Python . . . . .	170
11.3 Apêndice: Algoritmo eficiente de potenciação modular . . . . .	171
11.4 Apêndice: Algoritmo de Rabin-Miller (teste de primalidade) . . . . .	172
11.5 Apêndice: Algoritmo do “radix sort” em <code>python</code> . . . . .	173





# Capítulo 0

## Introdução

Nesta publicação reunimos um conjunto de notas informais das aulas teóricas da disciplina de “Tópicos Avançados em Algoritmos”<sup>1</sup>. Incluímos também diversos exercícios propostos nas aulas teóricas.

Uma grande parte da Ciência de Computadores consiste no estudo dos algoritmos – projecto (“design”), prova da correcção, análise da eficiência e implementação<sup>2</sup>. Há muitos, muitos, algoritmos nas mais diversas áreas: algoritmos de ordenação e pesquisa, algoritmos para procura de padrões em textos (“pattern matching”) algoritmos de compressão de ficheiros, algoritmos numéricos, algoritmos geométricos, algoritmos para análise léxica ou sintática, algoritmos para “garbage collection”, algoritmos para problemas relacionados com grafos, algoritmos usados na Teoria dos Grupos... a lista quase não tem fim. Contudo, vale a pena estudarmos algoritmos específicos: quando se compreende bem ou se inventa um algoritmo concreto está-se muitas vezes a utilizar uma ideia ou um conjunto de ideias que são aplicáveis com muito mais generalidade.

Nesta publicação estudaremos os seguintes temas relacionados com algoritmos:

- Algumas técnicas de análise da eficiência de algoritmos.
- Algoritmos específicos.
- Algumas técnicas genéricas aplicáveis em muitos problemas: “dividir para conquistar” e Programação Dinâmica.
- Minorantes de complexidade.

---

<sup>1</sup>Anos de 2008 e 2009, Departamento de Ciência de Computadores, Faculdade de Ciências da Universidade do Porto.

<sup>2</sup>O projecto, a prova da correcção e a análise da eficiência estão intimamente relacionados; devem efectuar-se “em paralelo”.

Este livro está organizado da seguinte forma. No próximo capítulo apresentam-se alguns elementos de análise de algoritmos, sendo especialmente estudadas as ordens de grandeza das funções e alguns métodos de solução de recorrências. O Capítulo 2 trata dos modelos de computação e do tempo de execução dos algoritmos, estudando-se em especial a análise amortizada do tempo de execução. O esquema “dividir para conquistar”, bem como a solução das recorrências que lhe estão associadas são estudados no Capítulo 3; esse esquema é aplicado a algoritmos de multiplicação de inteiros e de matrizes grandes. No capítulo seguinte são considerados os algoritmos aleatorizados, isto é, que têm acesso a uma fonte de números aleatórios (ou pseudo-aleatórios); em particular é estudado o “quick-sort” aleatorizado e o algoritmo de primalidade Rabin-Miller. As classes de complexidade “aleatorizadas” são também estudadas. O Capítulo 5 trata de alguns algoritmos relacionados com o problema da ordenação, sendo considerados métodos de ordenação aplicáveis a universos “pequenos”, o “radix-sort” e um algoritmo eficiente de determinação da mediana. Os “circuitos” como modelos de computação são mencionados no Capítulo 6, sendo estudadas em algum pormenor as redes de ordenação. No Capítulo 7 são consideradas 2 questões relacionadas com os métodos de “hash”: o “hash” universal e o “hash” perfeito. Algumas aplicações típicas da Programação Dinâmica – parentização ótima de um produto matricial, máxima sub-sequência comum e o problema da mochila (“knapsack problem”) – são considerados em algum pormenor no Capítulo 8. O capítulo seguinte trata do importante algoritmo FFT (“Fast Fourier Transform”); este algoritmo é estudado do ponto de vista da conversão entre 2 representações dos polinómios – representação por coeficientes e representação por valores em determinados pontos. A Teoria da Informação é aplicada no Capítulo 10 à determinação de minorantes de complexidade de algoritmos de ordenação e pesquisa.

**Pré-requisitos.** Para a boa compreensão destes apontamentos é necessário : (i) ter alguma maturidade matemática, em particular na área da matemática discreta, (ii) ter conhecimentos mínimos da teoria das probabilidades, (iii) conhecer os principais algoritmos de ordenação e respectiva eficiência, (iv) ter alguma experiência de programação numa linguagem como, por exemplo, o C, o C++, o Python ou até o Java, e ter capacidade de implementar nessa linguagem algoritmos descritos numa linguagem informal, (v) conhecer os fundamentos da teoria dos problemas completos em NP.

Os exercícios são parte integrante deste curso.  
O leitor deverá fazer um esforço sério para os resolver!