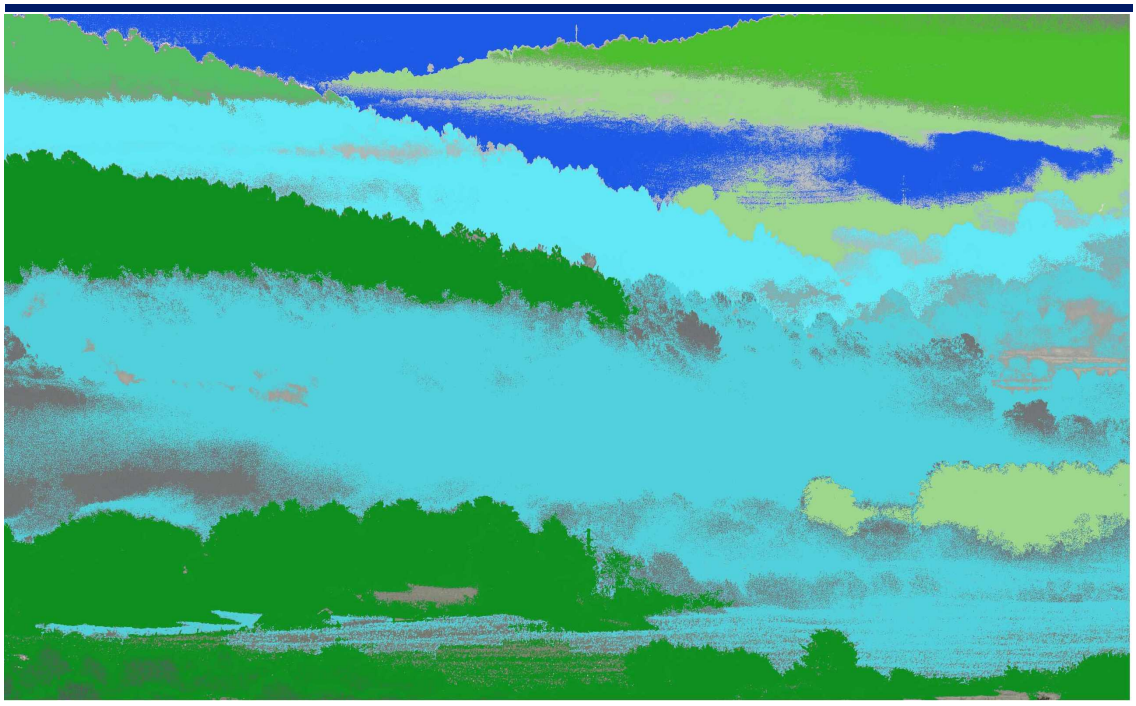
Tópicos Avançados em Algoritmos

Armando Matos



2008

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

Conteúdo

Capítulo 0

Introdução

Nesta publicação reunimos um conjunto de notas informais das aulas teóricas da disciplina de "Tópicos Avançados em Algoritmos"[1]. Incluimos 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[2]. 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.

[1] Anos de 2008 e 2009, Departamento de Ciência de Computadores, Faculdade de Ciências da Universidade do Porto.

[2] O projecto, a prova da correcção e a análise da eficiência estão intimamente relacionados; devem efectuar-se "em paralelo".

Este livro está organizado daseguinte 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 óptima 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-requesitos. 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!