Take home Exam #1

March 29, 2021

Abstract

This take home exam is due on Friday April 30. You should upload your work on Moodle. To present your work you should use a Jupyter notebook file. Projet can be turned back by group of two or three.

1 Choose your subject (7 points)!

Here are three subjects that we have covered in lecture and recitation:

- Strassen's algorithm
- RSA cryptosystem
- Multiplication of polynomials using the FFT

Choose one of the subject and propose a nice presentation of what you have understood. In particular your presentation should contains:

- Some Python codes
- Analysis of the correctness of the codes (not necessarly all the codes involved but at least one)
- Practical tests to illustrate the complexity of some algorithms
- Prolongation to related subjects

2 Sorting Algorithms (7 points)

In the lecture and recitation we have discussed two sorting algorithms:

- Selection Sort
- Merge Sort

Find a third type of algorithms in the literature (Quick Sort, Bubble sort, Heap sort etc...).

1. Explain the principle of this new sorting algorithm in you own words by emphasizing the differences with Selection sort and Merge sort.
2. Implement the three sorting algorithms.
3. Perfom numerical tests on the runing time of these algorithms and produce curves to plot the time complexity as the a function of the size of the array to sort.

3 Product of matrices of special shape (6 points)

T-matrices of size $(n+1)\times(n+1)$ are matrices of the following form:

$$\begin{pmatrix} a & b_1 & b_2 & \dots & \dots & b_n \\ c_1 & a & b_1 & b_2 & \dots & b_{n-1} \\ c_2 & c_1 & a & b_1 & \dots & b_{n-2} \\ c_3 & c_2 & c_1 & a & \dots & b_{n-3} \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ c_n & c_{n-1} & c_{n-2} & \dots & \dots & a \end{pmatrix}$$

1. Prove that the sum of two T-matrices is a T-matrix. What about the product ?

2. Propose a representation of a T-matrix such that the sum of two T-matrix can be calculated in $\mathcal{O}(n)$.

3. Let A be a T matrix. One considers the following representation of A as a polynomial of degree $2n+1$:

$$p_A(x) = c_n+c_{n-1}x+c_{n-2}x^2+\cdots+c_1x^{n-1}+ax^n+b_1x^{n+1}+b_2x^{n+2}+\cdots+b_nx^{2n+1} \tag{1}$$

Let us now consider $V = \begin{pmatrix} v_0 \\ v_1 \\ \vdots \\ v_n \end{pmatrix}$ a $n+1$ dimensional vector and let us represent V as a polynomial of degree n:

$$p_V(x) = v_n + v_{n-1}x + v_{n-2}x^2 + \dots v_0x^n. \tag{2}$$

(a) Explain why computing AV is equivalent to calculate $p_A(x).p_V(x)$.

(b) Deduce from the previous question that there exists an algorithm of complexity $\mathcal{O}(n\ln(n))$ that allows us to calculate $A.V$?

(c) Compare with the complexity of the naïve matrix-vector multiplication.

(d) Find an efficient way for multiplying two matrices of T-shape (justify the efficiency with a complexity argument).