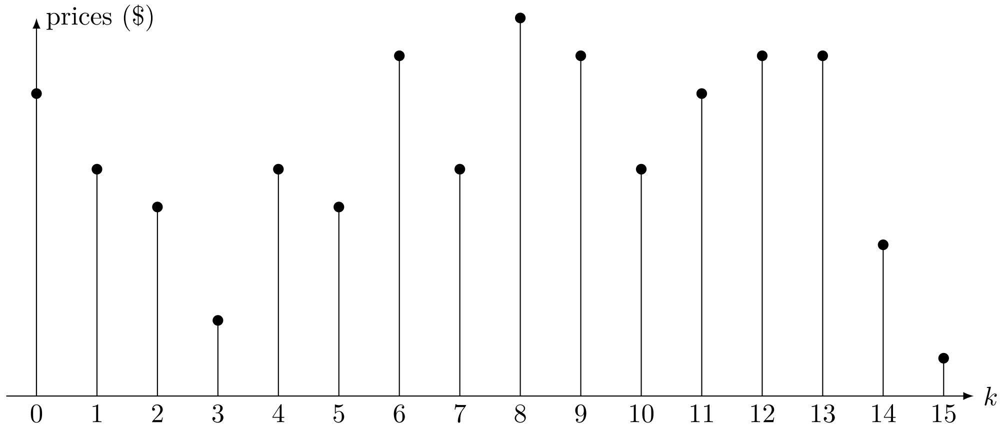
Divide and conquer – Exercises
ENSEA/FAME Computer Science

Exercise 1 – The buy and sell problem

We would like to solve the following problem: given a times series $(x_0, x_1, \dots, x_{n-1})$, find indexes i and j such that $0 \leqslant i < j \leqslant n-1$ and $x_j - x_i$ is maximal.

For example, in the following figure, the answer to the buy and sell problem is $(i, j) = (3, 8)$.



Question 1. What is the complexity of the brute-force algorithm solving this problem?

Question 2. Write an `def buyandsell(tab)`, based on the divide and conquer paradigm, which returns a solution to the buy and sell problem.

Question 3. What is the complexity of the function `buyandsell`?

Exercice 2 – Maximum Sum Subarray Problem

Given an array `tab` containing n (relative) integers, is it possible to find a subarray of `tab` with maximum sum? More precisely, we want to write a function `def find_max_sum(tab)` which returns a couple (i, j) such that $tab[i; j]$ is a maximal sum subarray of `tab`.

Question 1. What would be the complexity of the naive approach to this problem?

Question 2. Find a divide and conquer solution and compute its complexity using the master theorem.

Exercice 3 – Strassen algorithm for matrix multiplication

Question 1. Write a naive algorithm to compute the multiplication of two (possibly rectangular) matrices. What is its complexity?

Consider the following block decomposition of A and B (where A_i and B_i are matrices of size roughly $n/2$.

$$\begin{pmatrix} C_1 & C_2 \\ C_3 & C_4 \end{pmatrix} = \begin{pmatrix} A_1 & A_2 \\ A_3 & A_4 \end{pmatrix} \begin{pmatrix} B_1 & B_2 \\ B_3 & B_4 \end{pmatrix}.$$

Question 2. How many multiplications of submatrices do you need to compute C_1, C_2, C_3 and C_4? What would be the complexity of a divide and conquer algorithm based on the preceding block decomposition?

We can improve the last algorithm. We define the quantities:

$$\begin{aligned}
\pi_1 &= (A_1 + A_4)(B_1 + B_4)\\
\pi_2 &= (A_3 + A_4)B_1\\
\pi_3 &= A_1(B_2 - B_4)\\
\pi_4 &= A_4(B_3 - B_1)\\
\pi_5 &= (A_1 + A_2)B_4\\
\pi_6 &= (A_3 - A_1)(B_1 + B_2)\\
\pi_7 &= (A_2 - A_4)(B_3 + B_4).
\end{aligned}$$

Question 3. Check that the product matrix C can be computed in only 7 multiplications of submatrices via

$$\begin{aligned}
C_1 &= \pi_1 + \pi_4 - \pi_6 + \pi_7\\
C_2 &= \pi_3 + \pi_5\\
C_3 &= \pi_2 + \pi_4\\
C_4 &= \pi_1 - \pi_2 + \pi_3 + \pi_6
\end{aligned}$$

Question 4. Create a function `def strassenmultiply(A,B)` which takes as arguments two matrices `A` and `B` (implemented as a list of lists). What is its complexity?

Question 5. Using the block decomposition

$$A = \begin{pmatrix} A_1 & A_2 & cc \\ A_3 & A_4 \end{pmatrix},$$

show that one can compute A^2 (assuming A is a square matrix) in only 5 multiplications.

Question 6. Is it possible to compute the square of a matrix in $O(n^{\log_2 5})$ operations.