CS-E3190 Principles of Algorithmic Techniques

02. Recursive Algorithms – Graded Exercise

Please read the following **rules** very carefully.

- Do not consciously search for the solution on the internet.
- You are allowed to discuss the problems with your classmates but you should **write the solutions yourself**.
- Be aware that **if plagiarism is suspected**, you could be asked to have an interview with teaching staff.
- The teaching staff can assist with understanding the problem statements, but will **not be giving any hints** on how to solve the exercises.
- In order to ease grading, we want the solution of each problem and subproblem to start on a **new page**. If this requirement is not met, **points will be deduced**.

1. **Fibonacci sequence.** The Fibonacci sequence is defined in the following way:

$$\begin{cases} F(1) = & 1 \\ F(2) = & 1 \\ F(n) = & F(n-2) + F(n-1), \forall n > 2 \end{cases}$$

   (a) Write vector $\begin{pmatrix} F(n) \\ F(n-1) \end{pmatrix}$ as a multiplication between a matrix and a vector, depending only on n (the function F mustn't appear).

   (b) Give a recursive algorithm using a divide and conquer approach to compute $F(n)$, $\forall n \in \mathbb{N}$. The time complexity of this algorithm should be $o(n)$.
   Hint: Note this is o notation and not O, and that we are counting the number of arithmetic operations.

   (c) Analyse the time complexity of your algorithm.

2. **Binomial coefficients computation.** Recall that the binomial coefficient $\binom{n}{k}$ is the number of ways to choose an unordered subset of size k from a fixed set of size n. It can be computed with the following formula:

$$\binom{n}{k} = \frac{n!}{k!(n-k)!}.$$

   (a) Give a recursive algorithm for computing binomial coefficients that doesn't use the direct formula.
   Hint: search for relations between binomial coefficients with different values of n and k.

   (b) Give the recurrence relation verified by the time complexity of this algorithm.

   (c) Use it to compute the time complexity of the algorithm with respect to n. Then try to prove that the complexity is in $o(n!)$.