Homework 01: Matrix Multiplication

Roberto Corti

July 8, 2020

Clone the Strassen's project template from.

`https://github.com/albertocasagrande/AD_strassen_template`

and solve the following exercises.

Exercise 1

Generalize the implementation to deal with non-square matrices.

The implementation is written in the file `src/rectangular.c` by the function `strassen_rectangular _multiplication`.

In order to deal with rectangular matrices, the idea of this implementation is to embed squared matrices whose size is not a power of two into squared matrices having size equal to the smaller power of two bigger than the original size through a *padding* operation.
Once having this generalization to square matrices, if the input matrices are rectangular they are then divided to square blocks having size a power of two. Given this operation, the rectangular matrix multiplication will be given by a block-wise Strassen's multiplication, followed by a sum of the partial results

Exercise 2

Improve the implementation of the Strassen's algorithm by reducing the memory allocations and test the effects on the execution time.

The implementation is written in the file `src/strassen.c` by the function `strassen_matrix_multiplication_opt`.

In order to have a more efficient use of memory, instead of allocating 17 matrices (10 for S, 7 for P) as the original code of Strassen do, I decided to allocate only 6 of these (2 for S and 4 for P) and use a sequential calculation that updates multiple times the value of these matrices in order to compute the blocks

$C_{11}, C_{12}, C_{21}, C_{22}$.
Once written this optimized version, I performed a test with the original implementation and the results are shown by the graph in Figure 1.

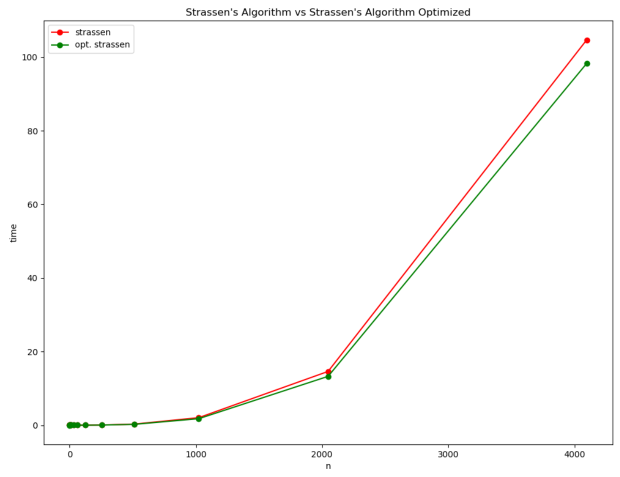

Figure 1: Time performance of the two implementations of the Strassen's Algorithm