Matrix Multiplication: homework (12/03/2020)

Gaia Saveri

1.

The required code is implemented in the function `strassen_padding` (inside `strassen/src/padding.c`).

The idea of this function is to embed squared matrices whose size is not a power of two into squared matrices having size the smallest power of two bigger than the size of the input matrices.

If the the input matrices are rectangular, then they are divided in square blocks, these blocks are padded into square matrices having size a power of two (if necessary) with the embedding technique described above. Then the algorithms performs block-wise multiplication (using the Strassen algorithm) and sums the partial results.

2.

The required code is implemented by the function `strassen` (inside `strassen/src/strassen.c`).

The original version of the Strassen algorithm allocates 17 matrices for each recursive call. In the optimized version, I allocated only 6 auxiliary matrices for each recursive call. During the computations the algorithm exploits multiple times each of these auxiliary matrices, allowing us to reduce the memory allocation.

This results eventually in a reduced execution time, as shown in `Figure 1`

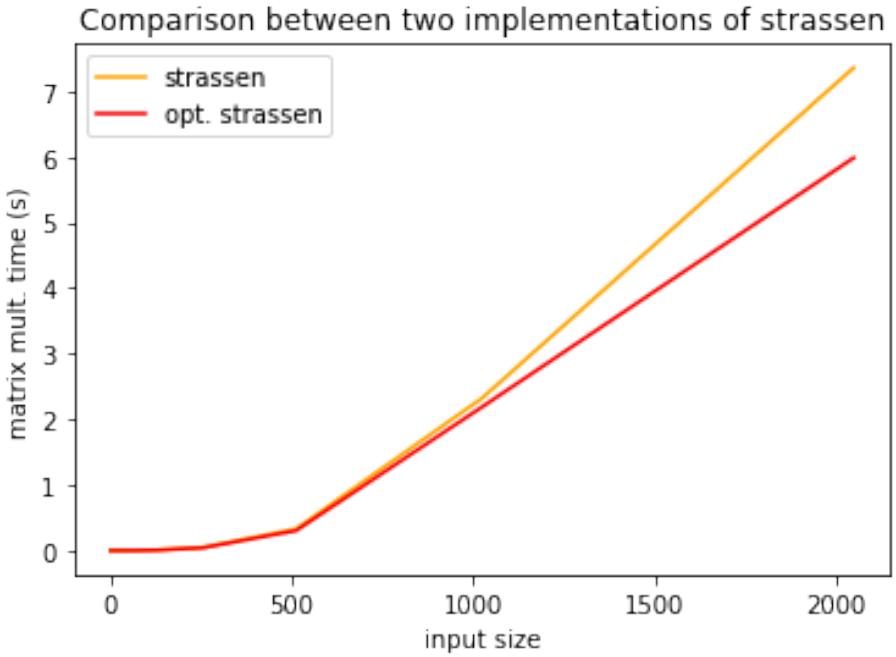

Figure 1: Strassen algorithm: execution time