Strassen's algorithm

This repository contains the code which provides a generalized implementation of the Strassen's algorithm to deal with non square and odd matrices.
First, in order to work also with rectangular matrices, I modified the Strassen's algorithm by taking in input the rows and the columns of the first matrix and the columns of the second matrix. At this point, the algorithm correctly computes the multiplication between two rectangular matrices whose size is a power of two.

Then, I chose the `dynamic peeling` technique (in `strassen.c` file, function `strassen_matrix_multiplication`) to make the algorithm works also with odd matrices. After the peeling and a call to the recursive Strassen's function, I did the corrections needed to obtain the correct result, as illustrated in *"Huss-Lederman S, Jacobson EM, Johnson JR, Tsao A, Turnbull T. Implementation of Strassen's algorithm for matrix multiplication. Proceedings of Supercomputing '96, November 1996."* (https://pdfs.semanticscholar.org/09d0/0291fc1232f07fcdfbcb7467114bdbdf1e42.pdf) .

Finally, I reduced the number of memory allocations from 17 up to 5 for each recursive call (in `strassen.c` file, function `strassen_matrix_multiplication_imp`) .

A Makefile can be produced by using `cmake` as follows:

```
cmake -G "Unix Makefiles" CMakeLists.txt
```

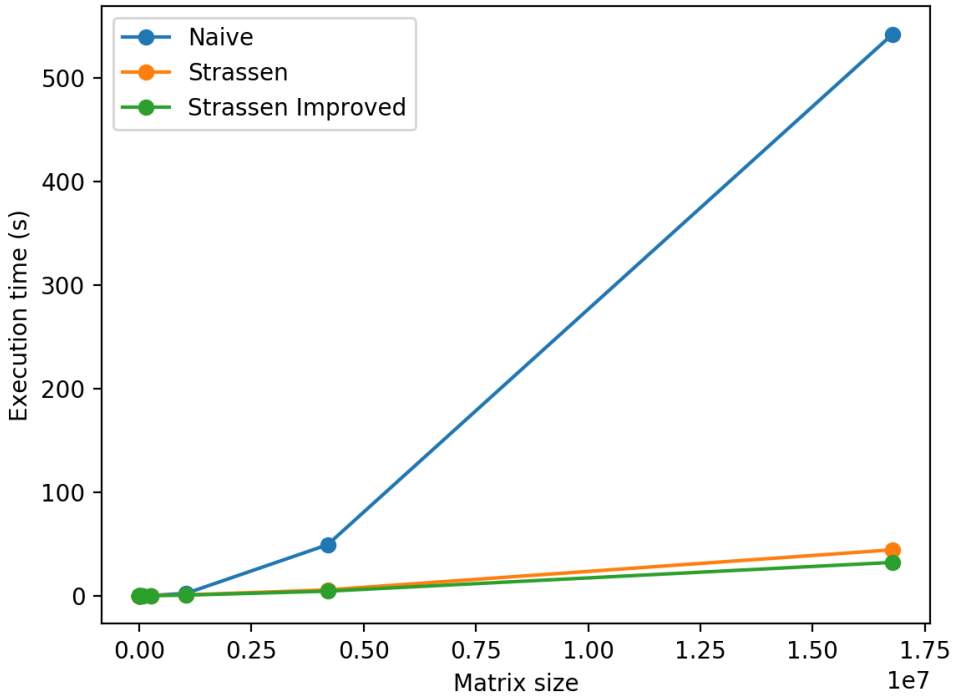
Afterwards the code can be compiled by executing `make` which produces an executable named `strassen_test`.

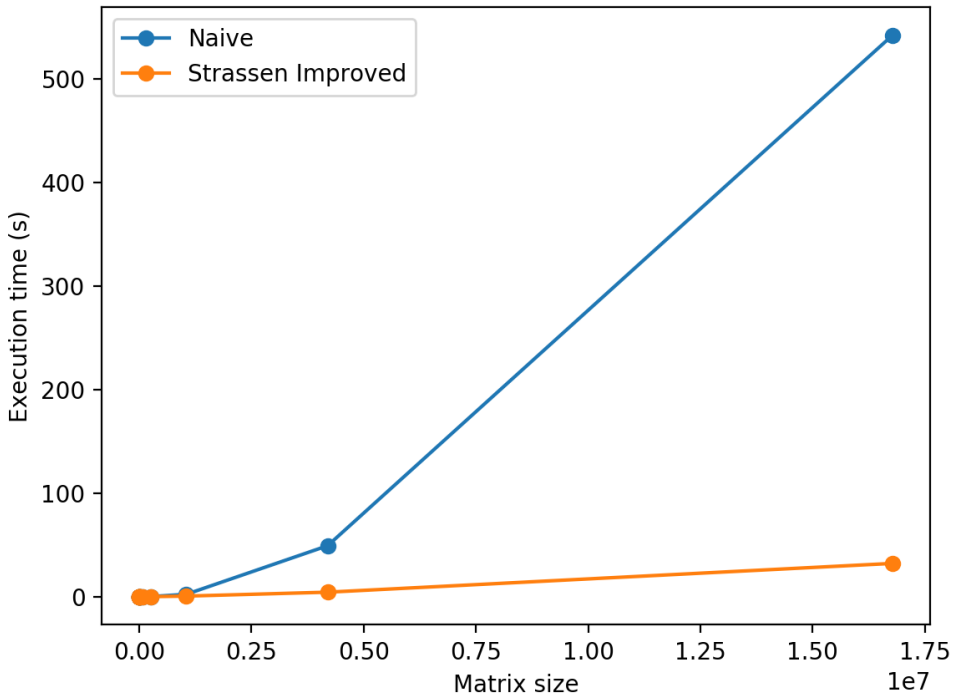
Testing

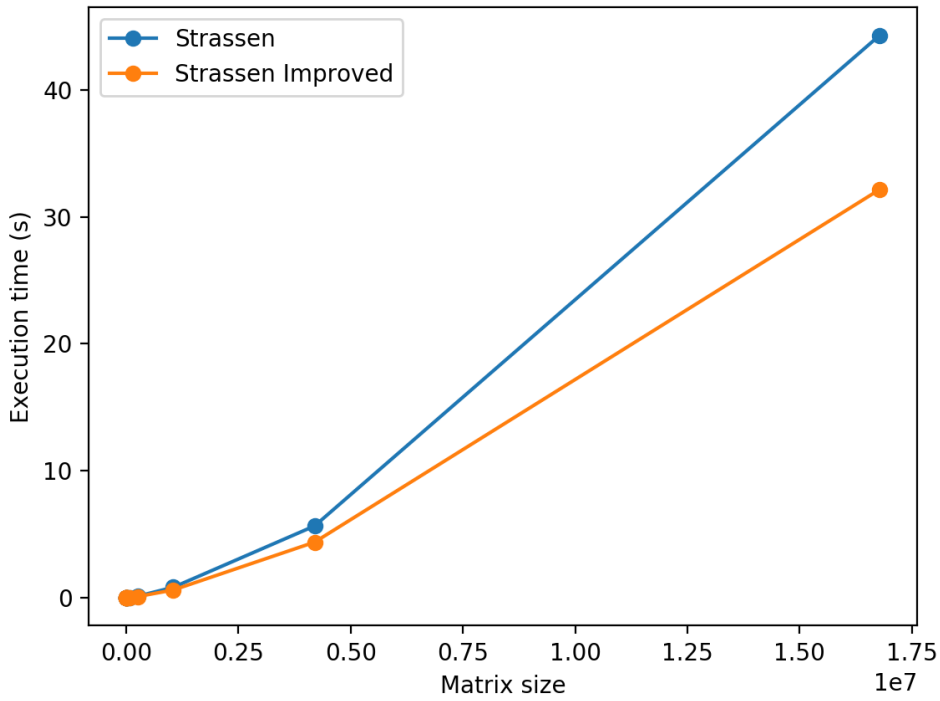
In this section, I reported some plots which underline the efficency of Strassen's algorithm with respect to the naive and the effects on the execution time due to the reduction of the memory allocation. The results shown below were obtained compiling the code with the -O4 option of `gcc` and the tests were performed using rectangular odd matrices.

The first graph shows the difference in the performance achieved with the naive algorithm, the Strassen's algorithm and the improved Strassen's algorithm; in the second one the naive algorithm and the improved version are compared and in the third one are shown the execution times of the improved version of the Strassen's algorithm and the Strassen's algorithm.

It is possible to notice that the Strassen's algorithm is much more efficient than the naive implementation and that the two versions of the Strassen's algorithm achieve almost the same performance. However, the improved version is always better than the standard one and for large matrix sizes it is even more efficient.


Naive
Strassen
Strassen Improved
Execution time (s)
0
100
200
300
400
500
0.00
0.25
0.50
0.75
1.00
1.25
1.50
1.75
Matrix size
1e7


Naive
Strassen Improved
Execution time (s)
0
100
200
300
400
500
0.00
0.25
0.50
0.75
1.00
1.25
1.50
1.75
Matrix size
1e7


Strassen
Strassen Improved
Execution time (s)
0
10
20
30
40
0.00
0.25
0.50
0.75
1.00
1.25
1.50
1.75
Matrix size
1e7