Matrix Multiplication: Homework

Clone the Strassen's project template from

https://github.com/albertocasagrande/AD_strassen_template

and solve the following exercises.

1. *Generalize the implementation to deal with non-square matrices.*

   The solution can be found in the branch `rectangular` of this repository, in the folder Strassen_alg.

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

   The solution can be found in the function `strassen_matrix_multiplication_best` (and consequently in the function `strassen_aux_best`), contained in the file `strassen.c` in the folder Strassen_alg.

   I compiled and run the code on Ulysses cluster in Sissa, for both square and rectangular matrices. The output was

   - for square matrices:

     ```
     ./strassen_test.x

         n   Naive Alg.   Strassen's Alg. Str. Alg. best  Same result
         1   0.000001     0.000020     0.000001     1 1
         2   0.000000     0.000000     0.000000     1 1
         4   0.000001     0.000000     0.000000     1 1
         8   0.000009     0.000001     0.000001     1 1
        16   0.000005     0.000004     0.000004     1 1
        32   0.000032     0.000034     0.000028     1 1
        64   0.000250     0.000244     0.000234     1 1
       128   0.001934     0.001884     0.001853     1 1
       256   0.100109     0.013079     0.012471     1 1
       512   0.905262     0.089764     0.086891     1 1
      1024   8.541473     0.627640     0.612996     1 1
      2048   66.494477    4.420142     4.326053     1 1
      4096   185.323830   31.036216    30.434407    1 1
     ```

   - for rectangular matrices:

     ```
     ./strassen_test.x

             dim   Naive Alg.   Strassen's Alg. Str. Alg. best  Same result
        1x   3x   4  0.000001     0.000022     0.000001     1 1
        2x   6x   8  0.000001     0.000001     0.000000     1 1
        4x  12x  16  0.000002     0.000001     0.000001     1 1
        8x  24x  32  0.000007     0.000006     0.000006     1 1
       16x  48x  64  0.000061     0.000049     0.000049     1 1
       32x  96x 128  0.000390     0.000368     0.000364     1 1
       64x 192x 256  0.022256     0.022456     0.022060     1 1
     ```

```
  128x 384x 512  0.172694     0.174371     0.174951     1 1
  256x 768x1024  1.452749     0.465220     0.463335     1 1
  512x1536x2048  12.976235    1.766585     1.750519     1 1
 1024x3072x4096  69.019623    8.170306     8.084943     1 1
```

While on the new partition `frontend-beta` we have the following results:

- for square matrices:

```
./strassen_test.x

     n   Naive Alg.  Strassen's Alg. Str. Alg. best  Same result
     1   0.000002    0.000006    0.000001    1 1
     2   0.000001    0.000001    0.000001    1 1
     4   0.000001    0.000001    0.000001    1 1
     8   0.000003    0.000002    0.000002    1 1
    16   0.000008    0.000006    0.000006    1 1
    32   0.000047    0.000041    0.000041    1 1
    64   0.000371    0.000347    0.000346    1 1
   128   0.003007    0.002964    0.002960    1 1
   256   0.025591    0.021378    0.020506    1 1
   512   0.211942    0.109676    0.100745    1 1
  1024   1.380608    0.741221    0.714642    1 1
  2048   19.931641   5.106519    5.021268    1 1
  4096   257.180293  35.810735   35.314176   1 1
```

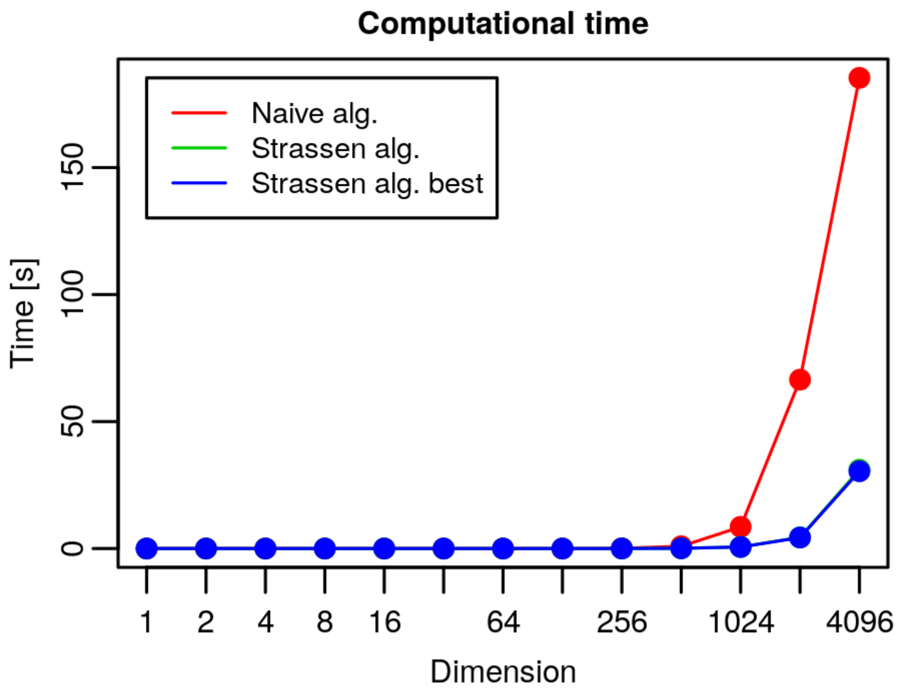
- for rectangular matrices:

```
./strassen_test.x

       dim   Naive Alg.  Strassen's Alg. Str. Alg. best  Same result
    1x   3x   4  0.000002     0.000006     0.000001     1 1
    2x   6x   8  0.000002     0.000001     0.000001     1 1
    4x  12x  16  0.000003     0.000002     0.000002     1 1
    8x  24x  32  0.000012     0.000009     0.000009     1 1
   16x  48x  64  0.000072     0.000066     0.000066     1 1
   32x  96x 128  0.000580     0.000552     0.000556     1 1
   64x 192x 256  0.004737     0.004684     0.004678     1 1
  128x 384x 512  0.039687     0.039087     0.039437     1 1
  256x 768x1024  0.294623     0.199306     0.196376     1 1
  512x1536x2048  2.336765     1.403819     1.380727     1 1
 1024x3072x4096  65.748678    9.800090     9.665999     1 1
```

We can see that the Strassen's algorithm is much better than the naive one, while the Strassen's algorithm with reduced memory allocations is only slightly better. Besides, it seems that on the new partition of Ulysses the times are a bit higher.

In the graph we can see that only in the last two pints the time is significantly different from 0. Unfortunately, we have too few significant points to establish with certainty that the complexity of the naive algorithm is $\Theta(n^3) = \Theta(n^{\log_2 8})$ and the one of the Strassen's algorithm is $\Theta(n^{\log_2 7})$, even though the graph is growing very quickly. The problem is that with high power of 2 in n the matrices become very very big and are impossible to store in memory.


Computational time
Naive alg.
Strassen alg.
Strassen alg. best
Time [s]
0
50
100
150
1
2
4
8
16
64
256
1024
4096
Dimension