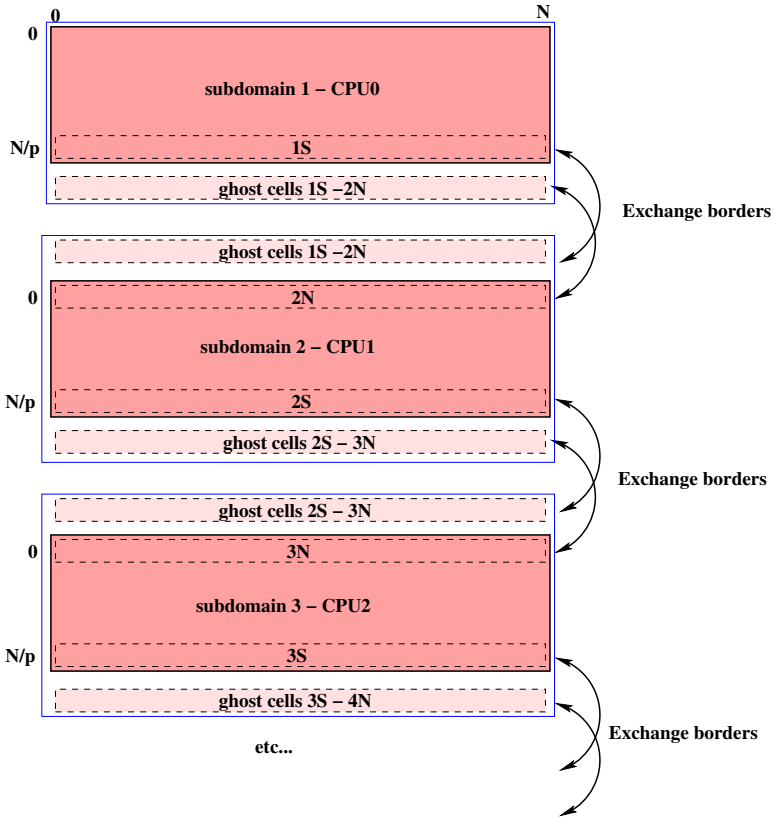
Serie 5

Exercise 1 (Parallelization with MPI)**.** Parallelize the Poisson 2D problem using the Messages Passing Interface (MPI). As a starting point, you can use the debugged, profiled and optimized serial version of the serie 4 or start from scratch.

Here follows some advises for your work :

- The memory allocation in C is done in "Row-Major Order" : make your domain decomposition by lines



  (In Fortran, the memory allocation is "Column-Major Order" (make the decomposition by columns))

- Try to keep the size of the MPI messages as large as possible (i.e. send/receive a full line instead of single elements). In order to avoid deadlocks, use $MPI\_Sendrecv$ first.
- Same problem as with OpenMP : the main bottleneck is the file writings : remove the call to write_to_file(). The verification is done by comparing the number of iterations to reach a given error ($L2$). To be sure your parallel implementation is correct : compare your results against the serial implementation.
- To increase the performance of your code, the communications can be hidden behind computation by using non-blocking communications (MPI_Isend)

Remember : each MPI process runs the same executable !

- Increase the size of the grid so that the total execution time on one node is close to 3-4 minutes
- Run your application on an increasing number of nodes by fixing the total size of the problem. Draw a log-log graph with the speedup ($S_p = t_1/t_p$) on the y axis, the number of nodes on the x axis (**strong scaling**)
- Run your application on an increasing number of nodes by fixing the size of the problem per processor. Draw a graph with the parallel efficiency ($E_p = S_p/p$) on the y axis, the number of nodes on the x axis (**weak scaling**)