Parallel Computing

2022/2023 1^{st} semester

Exercise Sheet #6 2022-11-16

Create a parallel processing code that performs the multiplication of two matrices A and B of dimension $n \times n$. Read n from the command line and use a random process to create the components of the matrices.

Note that, while testing the code, it is recommended to create matrices with systematic values for their components instead of a random process, in order to simplify debugging.

1. Distribute the lines of matrix A equally over $n_p = 4$ participating processes. This way, the master process has to send the corresponding part of matrix A and the whole matrix B to each of the slave processes, and the slaves send back the corresponding lines of the result matrix.

2. Distribute equal parts of both matrix A and matrix B over the processes. If the number of processes n_p involved is a square number so that $n_p = m^2$, you can divide the lines of matrix A and the columns of matrix B in m equally sized segments and thus have each slave process calculate one segment of the result with $\frac{n}{m}$ lines and $\frac{n}{m}$ columns. Make sure that each slave process receives only the parts of A and B that it needs for its part of the calculation. This reduces drastically the amount of data transferred through MPI.

   Note that for an efficient transmission of subsequent columns of matrix B, it might be useful to use the transposed matrix (`np.transpose`).

   In order for the master to recombine the resulting matrix from the segments sent by the slaves, you might want to have a look at the block method of numpy – `help(np.block)` .

   Begin with $n_p = n = 4$, and then make the code work for numbers of processes n_p that are powers of 4 (4, 16, 64, 256, . . .) and matrices with a dimension n such that n^2 is a multiple of n_p .

Submit the codes of your programs (1 to 2) at `https://trixi.coimbra.lip.pt/cap` until the beginning of the next lectures.