HPC Lab: (0). Introductory MPI exercises

(0.a) Use the template program **pingPong.c** to implement a loop to send/receive messages with different sizes. How does the message size influence the time being spent in MPI_Send and MPI_Recv? You may use **MPI_Wtime()** to measure wall-clock time and go with array size from 1, 2^2, 2^4 , …, to 2^{20} elements to make the impact of the data size clearly visible.

Derive an approximation for the communication time: $t_{comm}(m) = \alpha + m\beta$, where m is the number of bytes as the message length, α is the start-up time and β the transmission bandwidth (bytes/s). (Be mind: the curve of measured time might have two or more segments of trends, in that case you should calculate the values α and β of each segment separately). (Note: it is noticed that the first time of sending a message in a MPI program takes an unexpected long time, probably due to the communication initialization, so please ignore the time of sending 1 element when you try to find a formula for $t_{comm}(m)$.)

Perform the same measurement, but now placing the sending and receiving processes on two different nodes (with srun option: --nodes=2).

(Extra) Optional: what about the communication when MPI_Send and MPI_Recv are replaced by MPI_SendRecv ?

(0.b) Make a parallel version of the matrix-matrix multiplication algorithm (see the serial program ***MM-product.c***). Describe how the matrices will be partitioned and the communication of the initial data and the results. Measure the speedups for several values of the matrix dimension, and under the number of processes P=1, 2, 8, 24, 48 and 64. Discuss the results.