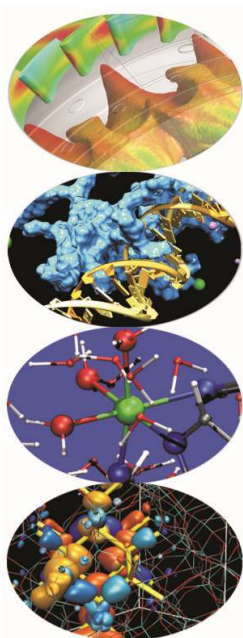


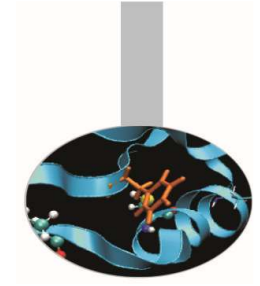
Advanced MPI

- *exercises* -

M.Cremonesi

May 2015





Startup notes

To access the server:

```
ssh a08tra??@login.eurora.cineca.it
```

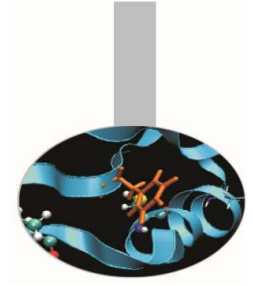
To reserve space on the server:

```
qsub -I -A train_cp1m2015 -l select=1:ncpus=4:mpiprocs=4 \  
-l walltime=3:00:00
```

To configure the MPI environment:

```
module load autoload openmpi
```

Compiling notes



To compile programs that make use of MPI library:

```
mpif90 | mpicc | mpiCC -o <executable> <file 1> <file 2> ... <file n>
```

Where: <file n> - program source files

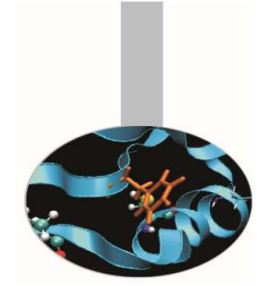
<executable> - executable file

To start parallel execution on one node only:

```
mpirun -np <processor_number> <executable> <exe_params>
```

To start parallel execution on many nodes:

```
mpirun -np <processor_number> -machinefile <node_list_file> \  
    <executable> <exe_params>
```



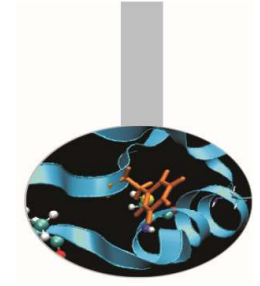
MPI exercises

You can find a list of useful exercises at the address:

<http://www.hpc.cineca.it/content/training-mpi>

Exercise 10: *create a cartesian topology and try using Cart_shift for neighborhood communications*

Exercise 11: *MPI data types are useful for managing matrix data whenever not-contiguous data are involved. To solve the exercise define a data type that include two columns/rows*



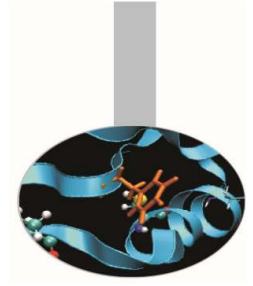
MPI exercises

Exercise 12: *again on matrix data; send and receive buffers do not need to be of the same type. The flag -std=c99 should be used to compile the C solution program with gcc*

Exercise 13: *some I/O functions are used to store and retrieve data; the function File_set_view is used to get File_write_all store data in the specified order*

Exercise 14: *use of MPI_Dims_create is not mandatory; remember to call File_set_view before using File_write_all and File_read_all. Use the flag -std=c99 to compile with gcc*

MPI exercises



Exercise 15: the function Alltoall redistribute an array in an ordered fashion; remember that sendcount and recvcount are the dimensions of the sub-blocks to be sent/received to/from single processes

Exercise 16: the function Allgather performs collection of distributed data to all processes