Extract from Project 1 statement:

The names of the available machines are provided in the `machines.xt` file. Each name of machine is followed by the number of cores this machine is supposed to have. The way the program sees the infrastructure depends on how names of the machines are written in this file. Suppose you have **p** machines, each machine having **n** cores. Thus you have two ways to see the infrastructure:

- You consider that you have **pxn** machines, each machine having one core and the program launches **pxn** workers, each worker using one core

- You consider that you have **p** machines, each machine having **n** cores and the program launches **p** workers, each worker using n cores.
  This file `machines.txt` is read by the program and must contain the list of the names of available machines written in the format corresponding to the way you want to use the infrastructure. Thus, before starting an execution you have to copy in the `machines.txt` file, the correct description of the available machines.

The file `machines_cores.txt` contains the description for the multi-cores execution (**p** machines with **n** cores) when the file `machines_nocores.txt` contains the description for the single core execution (**pxn** machines of 1 cores). At the beginning of the script `scriptParallel.sh` you will find the line where this copy is done. You have to modify this line accordingly to what you want. Note that to exploit the fact that a machine is multi-core, the program uses the indication of the number of cores contained in the file `machines.txt` to launch several instances of the method `solveCore(...)` of the `parclass MatWorker`.

Workers	Initialisation (s)	Sending (s)	Computing (s)	Total Time (s)	Speedup
1	0.101	0.077	10.215	10.393	1.000
2	0.475	1.069	1.715	3.259	3.189
4	1.237	1.919	0.870	4.026	2.582
6	1.620	2.170	0.738	4.528	2.295
12	3.462	3.470	0.559	7.491	1.387
16	4.579	3.970	0.577	9.127	1.139

Table 2: N=[redacted] (4 Cores per Machine)

Attention:
How many cores are you using for each worker?

How does it compare with single-core environment? Does the speed-up make sense?

You have to check all the information: use initialisation, sending, computing and total time values to understand better the speed-up behavior.
Consider the machines, and total cores use in each environment to give your conclusions. That will give you the number of threads that the program is launching.
YOUR CONCLUSIONS ARE INDIVIDUAL

Extracted from class slides:

• To determine the “performance” of a parallel system, with use the notion of

Speedup • Definition:

- If T_1 is the time required by a sequential program to solve the problem A on a sequential computer and T_p the time required by a parallel program to solve the same problem A on a parallel computer having p processors, the speedup is the ratio:

$$S(p)=T_1 / T_p$$

- **Requires a <u>clear definition</u> of the following elements:**

  A sequential program that solves the same problem

  A sequential computer

  p processors

<u>Always specify how the speedup has been calculated</u> • What we have compared

Please define clearly what elements are you using and explain your conclusions according to that.

The important point is to clearly define what is "one processing unit (hardware)". For sequential time, we have provide the time for 1 machine with 1 core.