TÉCNICO LISBOA

## PARALLEL AND DISTRIBUTED COMPUTING

2nd Semester | 2018/2019 | MEEC

# PROJECT ASSIGNMENT

## PARTICLES SIMULATION

*Group 39*

José Velez | Nº 81358

Vasco Amaral | Nº 81532

**April 16th 2019**

## Serial Implementation

For the serial implementation, our approach to solve the problem was to use an array structure for the particles, a matrix for the grid in order to represent the division of the average centres of mass of all the particles in that cell throughout the space. This way if the space is divided in 9 cells, we have a 3 by 3 matrix.

The first step is to initialize the first position for each particle, and while doing this create a list for each grid cell with all the particles in that cell.

Then for each iteration we calculate the centre of Mass of all the cells, with the average of all the particles in each cell for that iteration. After that we clear the list of particles since it was only used to calculate the centres of mass, and while calculating the forces, speeds and new positions of the new particles with the information of the centres of mass, we insert the particles in the cell list corresponding to the newly calculated position of the particle.

## OpenMP Implementation

For the OpenMp implementation, since we structured the serial implementation to support parallelism, our approach was to parallelize all the loops where each iteration was independent from each other. This was the case of the calculations of the centres of mass, of the several movements of particles (since its movement was independent of the position of the other particles for that iteration, it was only dependent of the current averages of the centres of mass for each cell) and for the several for loops going through the grid to update values and to clear the positions of the grid.

## MPI Implementation

For the MPI implementation, we had to make a few changes to the serial implementation in order to allow the passing of information correctly between the different machines in the most efficiently possible manner.

To achieve this, we decided the best way to approach the problem was to divide the particles evenly between the different machines, with the grouped approach of the Block Decomposition Algorithm to decide which particles were sent to each machine and keeping the entire Grid structure in each of them. We did this since all the calculations for each particle were independent, therefore this was the best way to distribute the particles evenly between the different machines, since most of the computation was regarding all the different particles.

Initially, only the main machine runs the init_particles() and after defining the first position for each of its particles, we start creating a buffer with all the initial information for each particle, and started filling a fixed size buffer that would not be bigger the machine's stack and would repeatedly send that buffer to the machine that was supposed to process that particle until we got to a particle that was assigned to a different machine. Each of the machines would then receive that buffer and use it to assign the first position of the particle throughout its local Particle structure array and calculating the initial centres of mass for each cell of the grid. A barrier was placed after the described process to assure all the machines had finished receiving and computing it. After this it would then calculate the partial contribution that the particles it was computing had for each cell of the grid, and would then broadcast this grid array to all the other machines, in order for them to add all of them together, so that each machine had the information of the centres of mass of the entire grid, and would not have to communicate with all the machines each time it needed information the others had.

Then for each step of movement, each machine would repeat this calculation of their contribution for the centres of mass and broadcast them to all the other machines, and then calculate the forces, speeds and position of each of the particles assigned to them, since they were independent from each other, the same way the serial implementation worked, since it was already supporting the division of particles between them. After the calculation of all the particles' new position there is a barrier, to make sure all the machines have finished it. Then they would broadcast the new sum of the masses of their particles in each cell, so all the machines would just add each of the machine's mass to get the total mass for each cell for the next step.

This implementation made it possible to use the previous OpenMP implementation in a very similar way, since each machine's computation was independent from the others, so we could just parallelize each of the loops' computation in different threads in each machine, as we did before.