

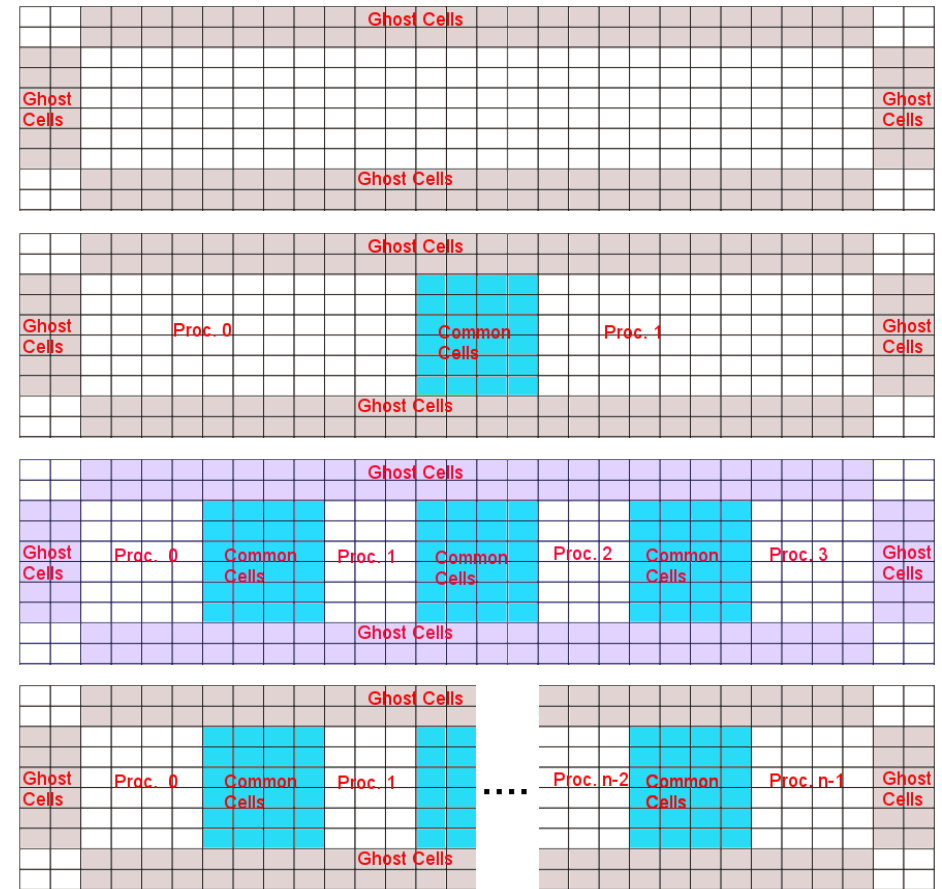
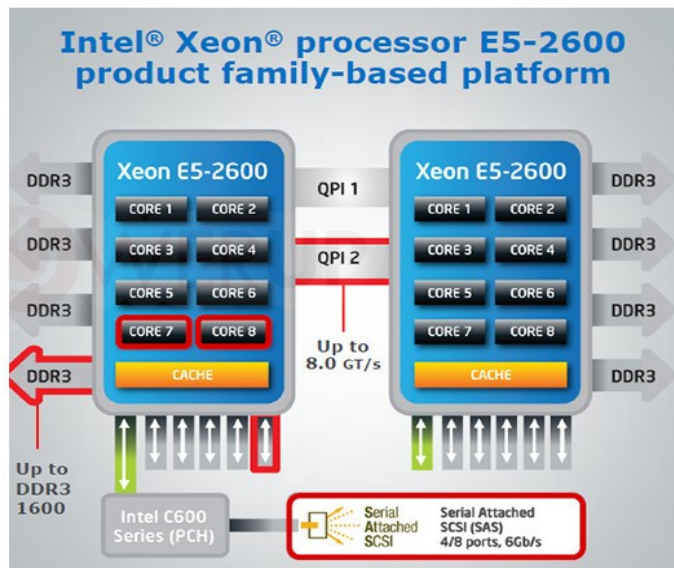
Introduction

Parallelization of a 2D Hydro code using MPI

Course: ESC401 - HPC1b: Parallel Computing

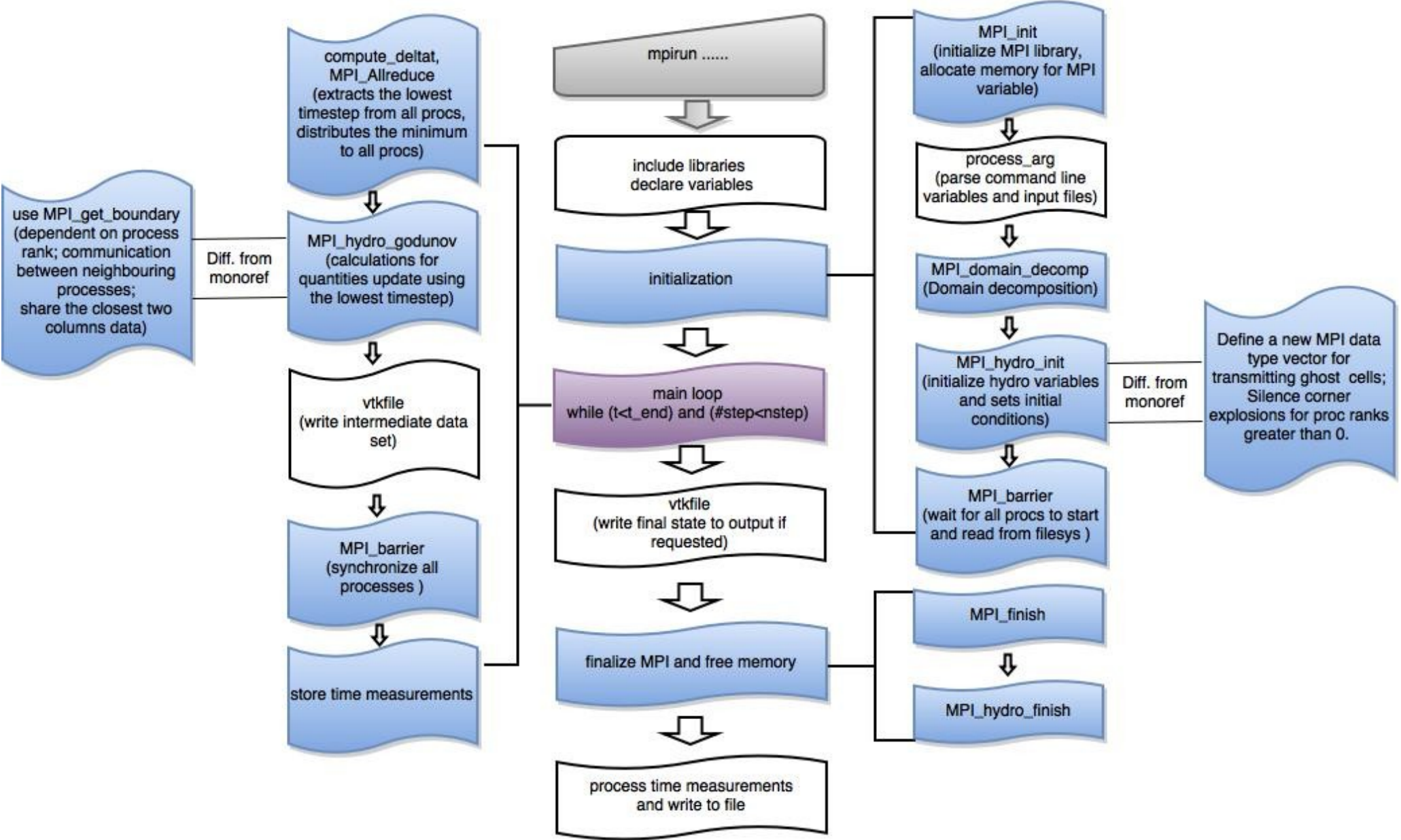
Authors: Rafael Küng, Christian Reinhardt and Mihai Tomozeiu

- The original code simulates an explosion in the lower left corner (right images) of a 2D wind tunnel using a single process
- Number of individual proper cells $n_x(\text{rows}) * n_y(\text{columns})$ and $2 * (n_x + n_y)$ ghost cells for reflective boundaries
- Approach:
 - general idea: modify as little code as possible
 - slicing the domain along the x-axis
- Domain distribution in right figures panels:
 - 2nd: 2 cores; 3rd: 4 cores; 4th: n cores
- Each neighboring processes need to communicate the closest two columns of proper cells to each other
- Minimum time step found among processes will become the time step of all processes

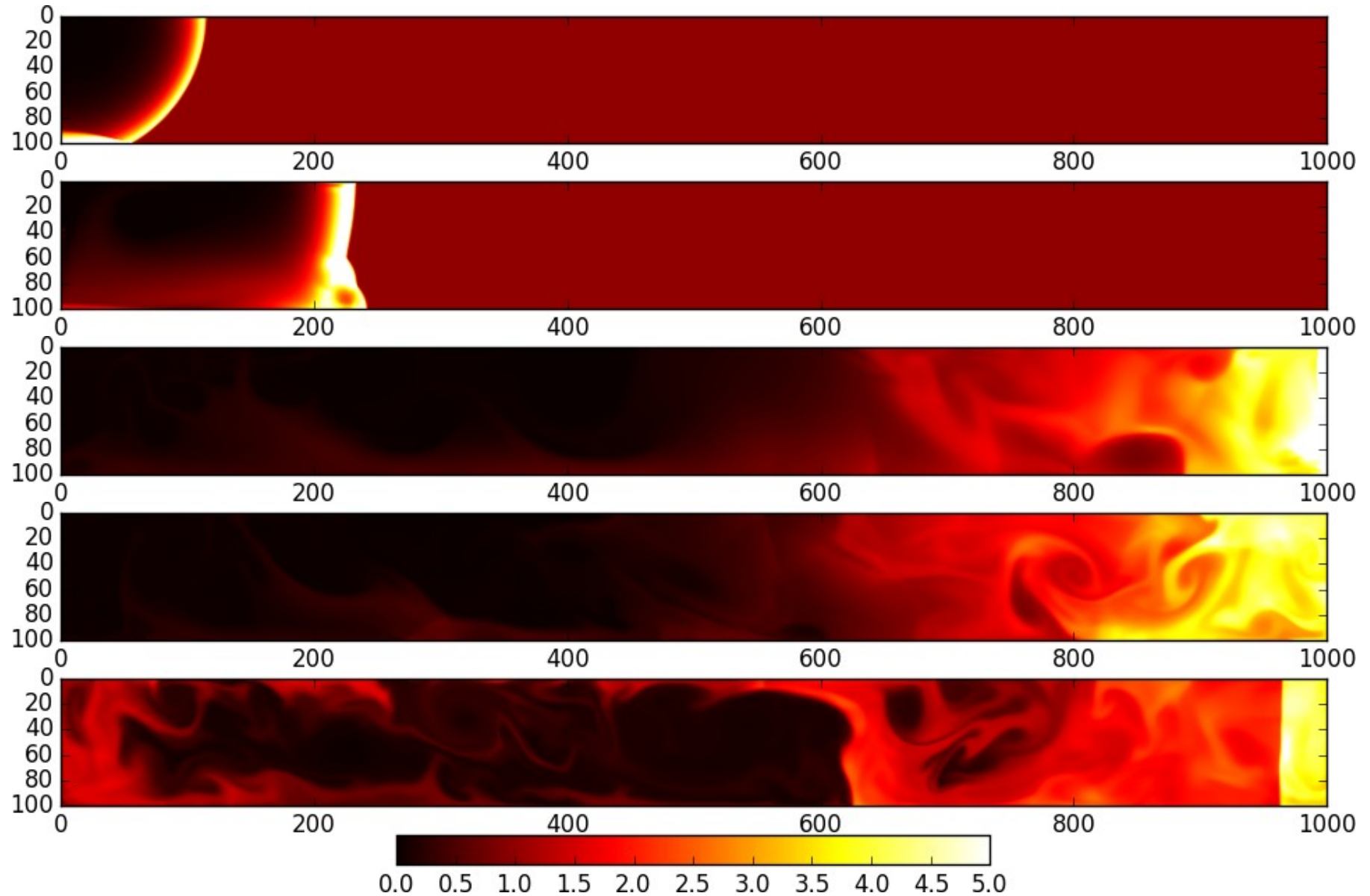


- The simulations were run on the local zbox4
- Intel Xeon E5-2660 (family platform sketch in the left image)
- 8 cores per socket
- 2 sockets per node
- Internode communication with QDR Infiniband

Code Description



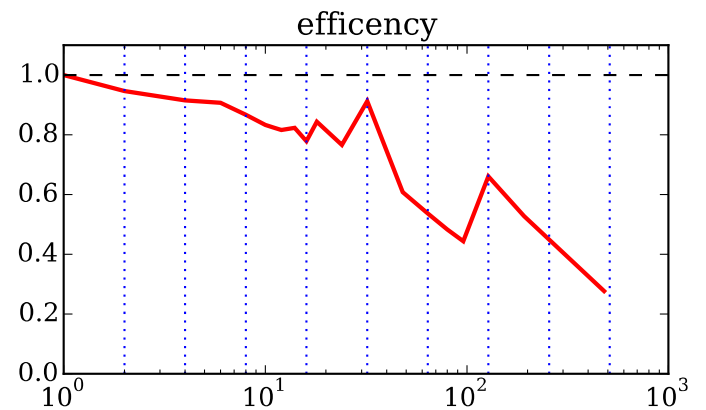
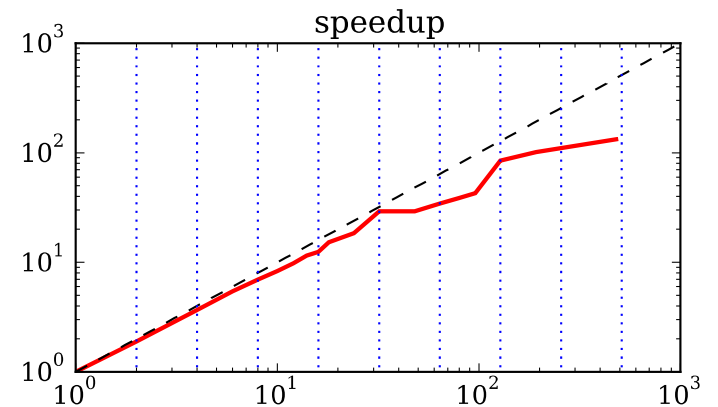
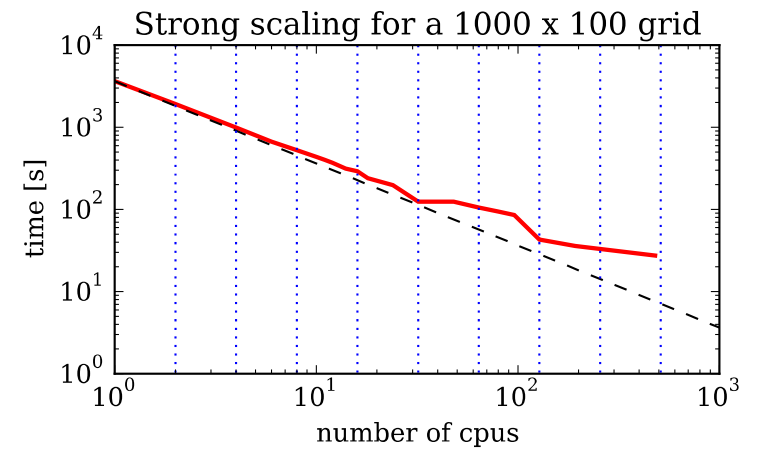
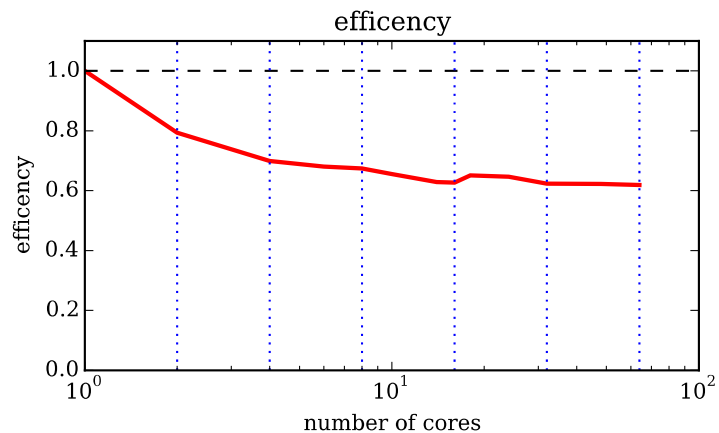
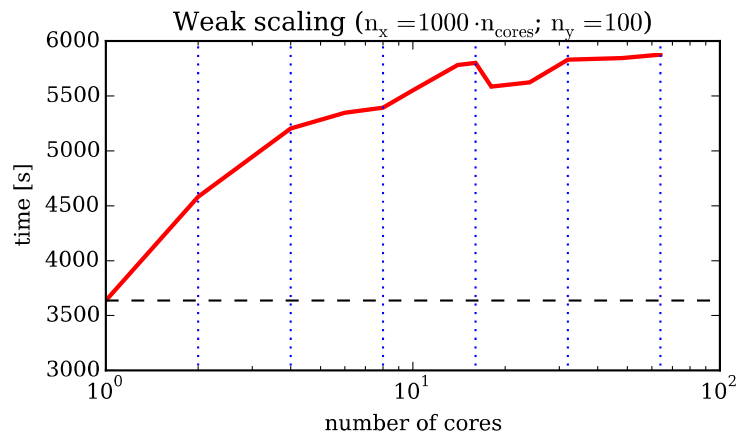
Results / Output



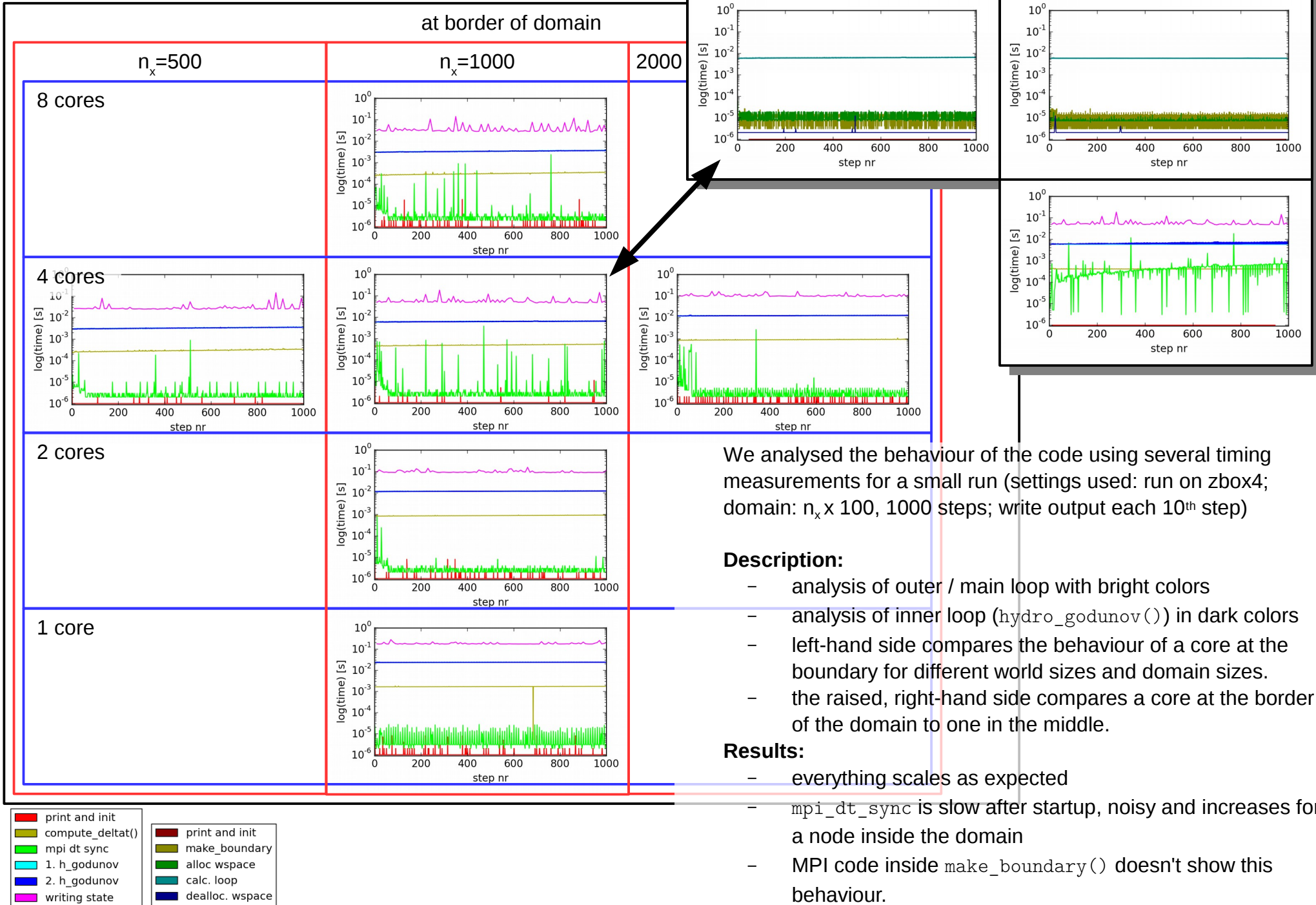
Example of a run (1000x100 grid): An initial overpressure in the top left cell causes a blast wave to propagate through the computational domain. The walls are reflecting so the shock bounces back at the domain boundaries and interferes with the original wave. This causes an oscillating bulk motion with beautiful and quite complex wave patterns in the inner part of the domain. The colors represent the fluid density where black is zero and white is at least five.

Scaling Behaviour

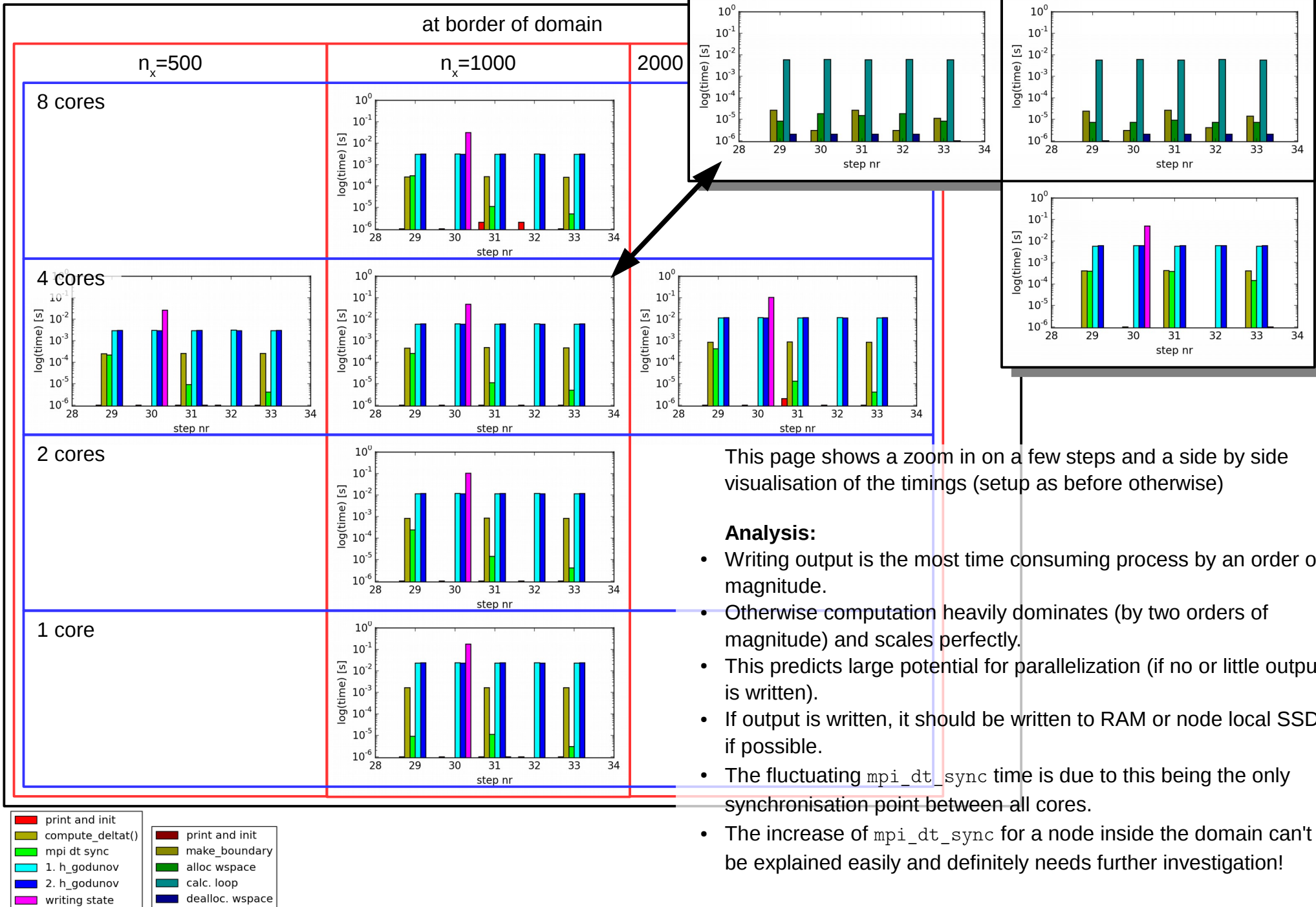
- Strong Scaling (right figures)
 - performed on a 1000x100 ($n_x \cdot n_y$) grid
 - results from 1 to 96 cores
- Weak Scaling (left figures)
 - n_x scaled with the number of cores, n_y constant
 - the 16 cores point marks the transition to internode communication showing a core performance worsening after 4 cores and no



Code Diagnostics – Overview



Code Diagnostics – Details



Conclusions

- We successfully parallelized the existing serial code with MPI.
- Using our code we ran several simulations with different grid sizes on a varying number of CPU cores. Both the hard and the weak scaling plots show a clear performance improvement when run on multiple cores.
- For the hard scaling the simulation time is linearly decreasing with the number of CPUs up to 10 tasks. Our timing analysis shows that after that writing outputs to the hard drive is the main bottleneck and causes a problem if too many outputs are written.