# SS-2001-01: Portable Lattice-Boltzmann in Java

Rubén Jesús García Hernández

EPCC, University of Edinburgh
University of Edinburgh
email ruben@epcc.ed.ac.uk

## Lattice-Boltzmann

The Lattice-Boltzmann (LB) is an algorithm which allows physicists to simulate the hydrodynamics of complex fluids in 3D.

## What is Ludwig?

Ludwig is a general purpose parallel Lattice-Boltzmann code. A C version was developed in 2000.[1] It divides the calculation into four routines:

- Set boundary conditions.
- Propagation.
- Collision.
- Bounce Back.

This project consists in the Java port of Ludwig. Three major versions have been developed: a serial version, a parallel version using shared memory, and a parallel version using message passing.
Here is one of the simulated results using Ludwig:

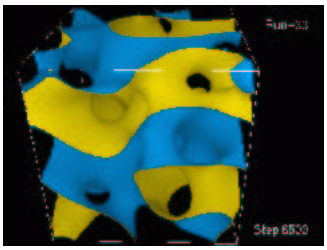

*Figure 1 : Evolution of the fluid-fluid interface*

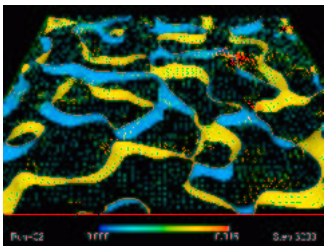

*Figure 2 : Time-resolved velocity maps (cropped for clarity to a thin section)*

## Comparison Sequential C versus Sequential Java

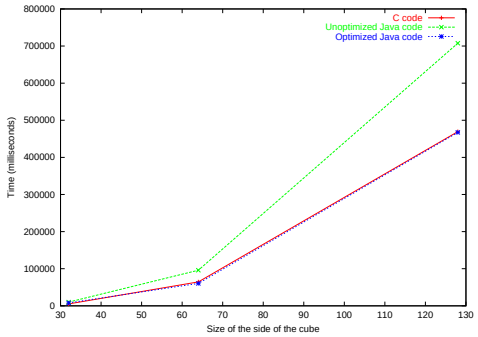


*Figure 3 : Graphic comparison C vs Java.*

## Comparison among different Parallel Java versions

## OMP

- First version
  The problem with the original propagation algorithm was that a copy had to be made of the boundary planes of each processor because the other processors needed to access them before they are overwritten. As the number of processors increased, so did the amount of copying done. The final result is no speedup in this zone of the program. When enough processors are used, Amdahl's Law causes the total speedup to freeze.

- Second version
  A close examination of the sequential loops showed that part of the calculation done dealt with obtaining the first and last elements each processor had. That calculation was independent of the loop index and was moved outside. This provided a small increment in speed.

- Third version
  The final approach was a new algorithm which consisted in using a double buffer for the sites array. This meant that no copying of the buffers was necessary, since the original array was not modified. This eliminated the non-scaling part of the algorithm at the cost of doubling the amount of memory needed. Since in High Performance the scaling is much more important than the memory needs (because what is wanted is to be able to use more processors as new machines are available and a new machine usually means more memory as well), this new algorithm is preferable.

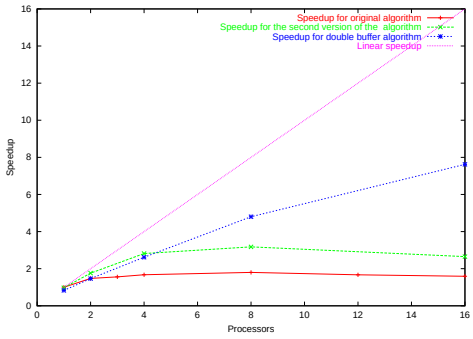


*Figure 4 : Comparison between the scaling of the original parallel version and the new double buffer version*

## MPI

This version is still under development. Most of the code has been ported, but validation, error correction and benchmarking have to be done.

## Acknowledgements

Supervisors in this project:
Dr J-C Desplat
Mark Bull
Thanks to Alexander Wagner and Lorna Smith for their help.

## References

1. J.-C. Desplat, I. Pagonabarraga, and P. Bladon, 'LUDWIG: A parallel Lattice-Boltzmann code for complex fluids', Computer Physics Communications 134, pp 273-290 (2001)