Data-Parallelism and GPUs for Lattice Gas Fluid Simulations

M.G.B. Johnson , D. P. Playne and K. A. Hawick

mitchelljohnsonnz@gmail.com, d.p.playne@massey.ac.nz, k.a.hawick@massey.ac.nz
Complex Systems & Simulation Group, Institute of Information and Mathematical Sciences, Massey University, Albany

Introduction

Computational fluid dynamics is a powerful tool for exploring highly non-linear behaviour in complex fluid systems. Solving the Navier-Stokes equations, even in their simplest form is still a computationally intensive task and requires sophisticated numerical solver techniques. The lattice gas [1] cellular automaton [2] approximation can be used to explore some qualitative and quantitative aspects of certain fluid flows. LGCA models and various sophisticated refinements such a the Lattice Boltzmann approach [3] can be formulated on more regular structures that is sometimes possible for full-field Navier-Stokes equation formulation. LGCA models are also highly data-parallelisable and offer the potential of simulating large model systems. Many physical systems such as fluid-flow and particularly turbulent fluid flow [4] reveal interesting properties and behaviours on many different length scales. A large simulation that can effectively capture several powers-of-ten of different length scales can therefore provide a useful simulation tool.
Fluid flow simulations are normally based upon a numerical solution or time-integration of the relevant partial differential equations. These are derived from the continuity equation:

$$\frac{\partial \rho}{\partial t} + \nabla \cdot (\rho \mathbf{u}) = 0 \tag{1}$$

which is written here in terms of the fluid density ρ and the time t and position dependent $\mathbf{u}$ velocity, which is a vector field. For our purposes we restrict ourselves to cases of constant density and this "incompressible fluid" case gives: $\nabla \cdot \mathbf{u} = 0$. We use this to obtain a derivation of the Navier-Stokes equation is often expressed in terms of the substantive derivative – the derivative following the fluid motion, which is defined as:

$$\frac{D\mathbf{u}}{Dt} \equiv \frac{\partial \mathbf{u}}{\partial t} + \mathbf{u} \cdot \nabla \mathbf{u} \tag{2}$$

Which leads to the form of the general Navier-Stokes form in D dimensions (typically 2 or 3) as:

$$\frac{D(\rho\mathbf{u})}{Dt} + \rho(\mathbf{u}\cdot\nabla)\mathbf{u} = -\nabla p + \eta^2\mathbf{u} + \left(\zeta + \frac{1}{D}\eta\right)\nabla(\nabla\cdot\mathbf{u}) + \mathbf{F} \tag{3}$$

including a general applied force term $\mathbf{F}$ and the bulk viscosity ζ.. In the case of a fluid of constant density ρ this yields the incompressible form:

$$\rho\frac{D\mathbf{u}}{Dt} = -\nabla p + \eta\nabla^2\mathbf{u} + \mathbf{F} \tag{4}$$

It is useful to focus for our purposes on the two dimensional case in simple Cartesian coordinates where $\mathbf{u} \equiv (u, v)$ and:

$$\begin{aligned} \frac{\partial u}{\partial x} + \frac{\partial v}{\partial y} &= 0 \qquad (5) \\ \frac{\partial u}{\partial t} + u\frac{\partial u}{\partial x} + v\frac{\partial u}{\partial y} &= -\frac{1}{\rho}\frac{\partial p}{\partial x} + \nu\left(\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2}\right) + \frac{F_x}{\rho} \\ \frac{\partial v}{\partial t} + u\frac{\partial v}{\partial x} + v\frac{\partial v}{\partial y} &= -\frac{1}{\rho}\frac{\partial p}{\partial y} + \nu\left(\frac{\partial^2 v}{\partial x^2} + \frac{\partial^2 v}{\partial y^2}\right) + \frac{F_y}{\rho} \end{aligned}$$

where we have written the kinematic shear viscosity $\nu \equiv \eta/\rho$.

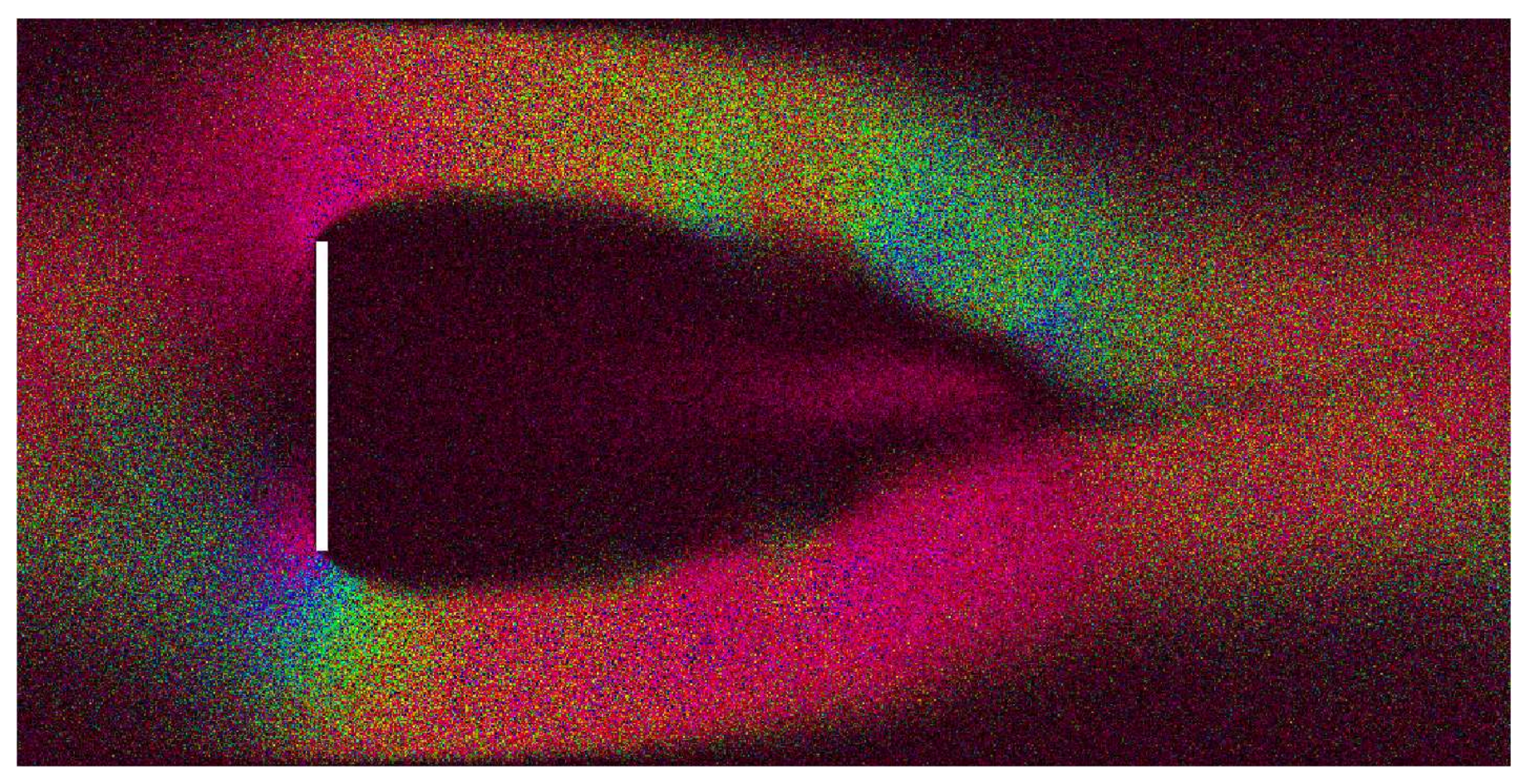

Figure 1: Lattice Gas Cellular Automaton flowing past a plane barrier. System size is 4096×2048 with rainbow colour wheel cells representing flow direction averaged over 5 bit cells

Lattice Gas Cellular Automata

As previously mentioned the LGCA method provides an efficient way to simulate large scale fluid-flow by moving particles within a triangle mesh (see Figure 2) and applying simple collision rules conserving mass, momentum and energy. A triangular lattice is used to satisfy the required interaction constraints established by the Navier-Stokes equation. The resulting system does not correctly model physics at a microscopic level but produces satisfactory bulk properties at a macroscopic level. Each site has six neighbours that it can interact with based on the collision rules identified by the velocity vectors within the site (see Figure 3). There are 256 possible particle configurations with applicable rules to be fixed and stored in a lookup table. Fortunately the rules for the presence or absence of a particle are opposite which allows us to specify only half of the rules and flip the bits on both the input and output to produce the other half based on the status of the particle in question.

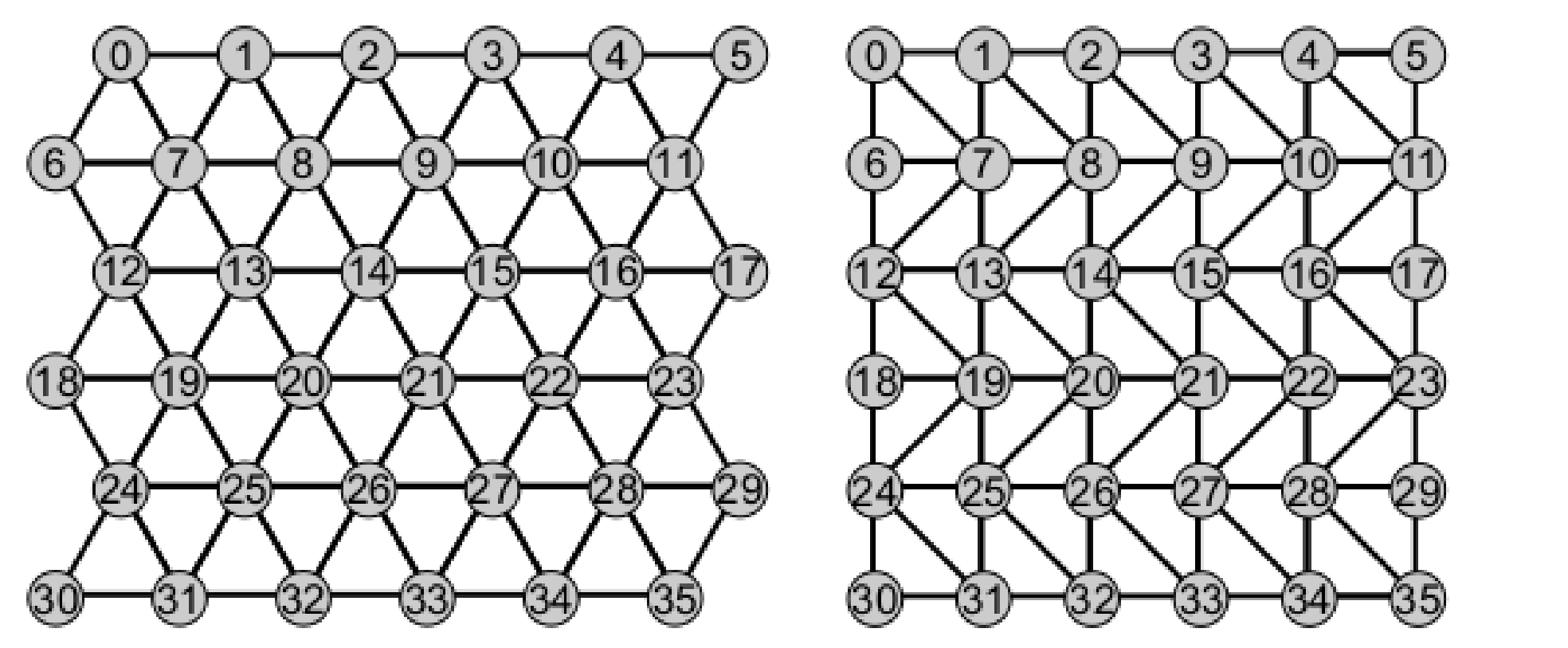

Figure 2: A Triangular mesh as it is used conceptually (left) and the same mesh distorted for ease of storage and computation (right).

Each particle can be represented in one byte, the first six bits from zero to five represent the six velocity vectors. Because mass, momentum and energy are conserved the interaction between the particles is solely based on the collision rules thus allowing the velocity and direction to be identified using only one bit. Bit six represents a rest particle while bit seven, when set allows a particle to act as a barrier that blocks all incoming particles. To improve memory management we have compacted four lattice sites into one 32bit int allowing the size of the array storing the data to be one quarter of the size it would be if a full int was used for each site.

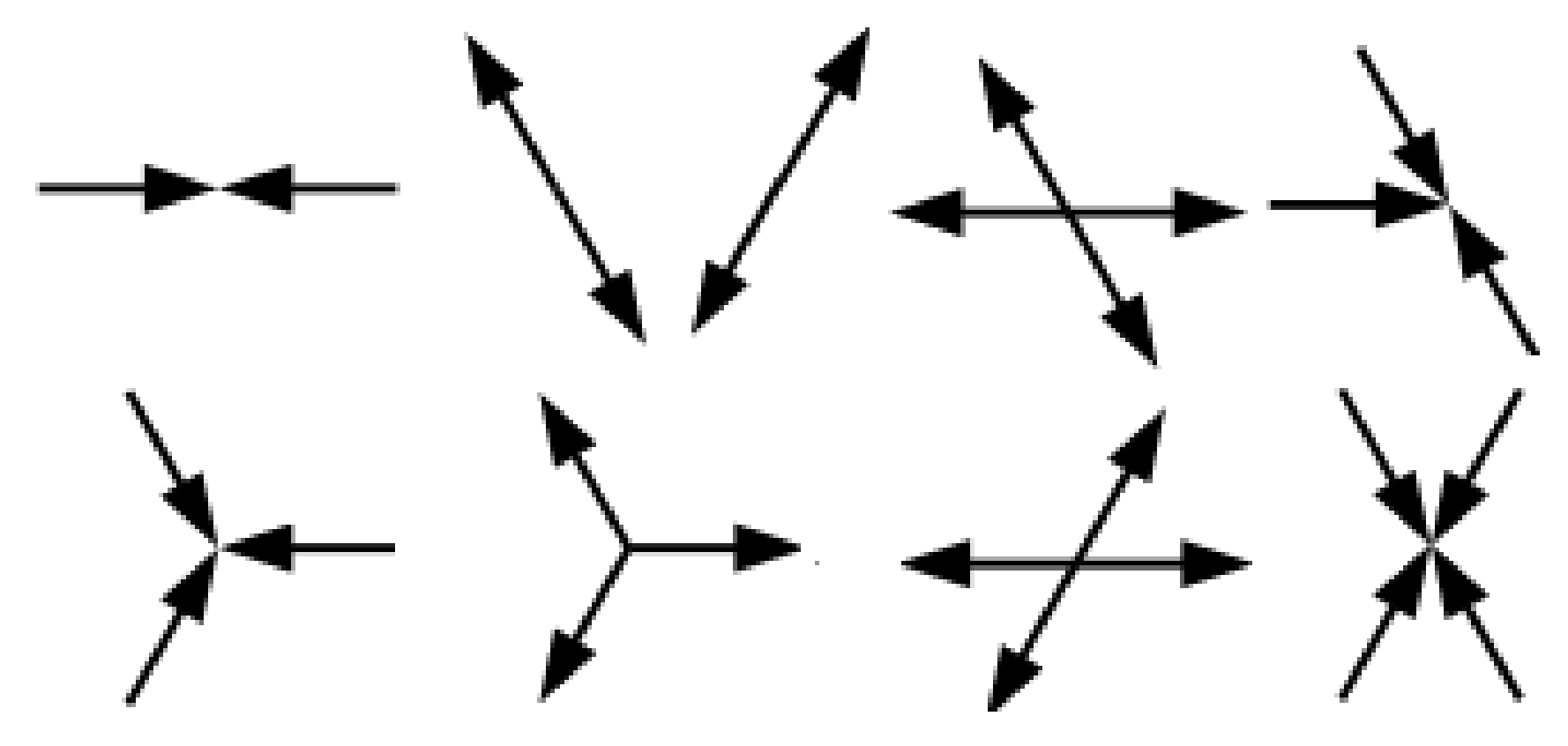

Figure 3: The collision rules of the Lattice Gas Cellular Automata.

LGCA on the GPU

Two types of GPU memory for accessing the triangular mesh were tested - Global and Texture. In previous neighbour-based, square lattice GPU simulations it was found that texture memory provided the best performance [14]. However, both types were tested as the access patterns are different for triangular lattices. The first three kernels (A,B,C) read the cell and neighbouring values directly from global memory whereas the last three kernels (D,E,F) make use of the texture cache when accessing these values.
While the spatial caching of texture memory is advantageous for accessing neighbouring values, it does require a additional memory copy from the output array into the texture-bound array. This overhead must be weighed against the performance benefits the spatial caching provides.
To test the performance of the LGCA implementations, six kernels (A-F) on four different NVIDIA GeForce 200 series graphics cards were executed. For GPU comparisons Kernel B (Global memory for lattice and Shared memory for look-up table)are used for admissible results, The cards tested are the GeForce 210, GT 220, GTX 260+ (factory overlocked to 666MHz) and a GTX 295. These implementations are also compared to two CPU simulations of the LGCA in Java and C++ running on an Intel Core i7 920 2.66GHz with 6 GB of DDR3-1600 memory and on an Intel Core 2 Quad 2.66 GHz with 8GB of DDR2-800 memory.

Implementation	Time (seconds)	Million hits per second
Java (i7 920)	73.6 ± 1.3	233 ± 4
C++ (Core 2 Quad)	101.3 ± 0.7	169 ± 1
C++ (i7 920)	51.7 ± 0.6	332 ± 4
GeForce 210	46.6 ± 3.4	369 ± 27
GT 220	11.1 ± 0.04	1,548 ± 6
GTX 260+	2.8 ± 0.005	6,136 ± 11
GTX 295	2.9 ± 0.005	5,924 ± 10

Figure 4: Performance results for the different implementations of the LGCA simulating 1024 time-steps on a 4096x4096 triangular lattice. All CUDA implementations use kernel B (Global memory for lattice and Shared memory for look-up table).

The performance for the GPUs is very positive (the GTX 260+ provides a 18x speedup over the i7), yet the speed-up factors are not as high as seen in previous square-lattice simulations [14]. The performance between the GTX 260+ and the GTX 295 are very similar (note that only one of the GPUs in the GTX 295 was used) with the GTX 260+ performing slightly faster. As seen previously [13] the higher clock speed of the GTX 260+ (666MHz vs 576 MHz) has more impact the the slighter higher number of cores in the GTX 295 GPU (240 vs 216). The performance results of the i7 920 is particularly impressive compared to the Core 2 Quad. Both processors have the same clock-speed yet the i7 can compute the simulation almost twice as fast.This performance improvement may be attributed to the processor design and also the memory speed. The Core 2 Quad is using DDR2-800 memory while the i7 has DDR3-1600. This increased memory frequency also contributes to the increased performance.

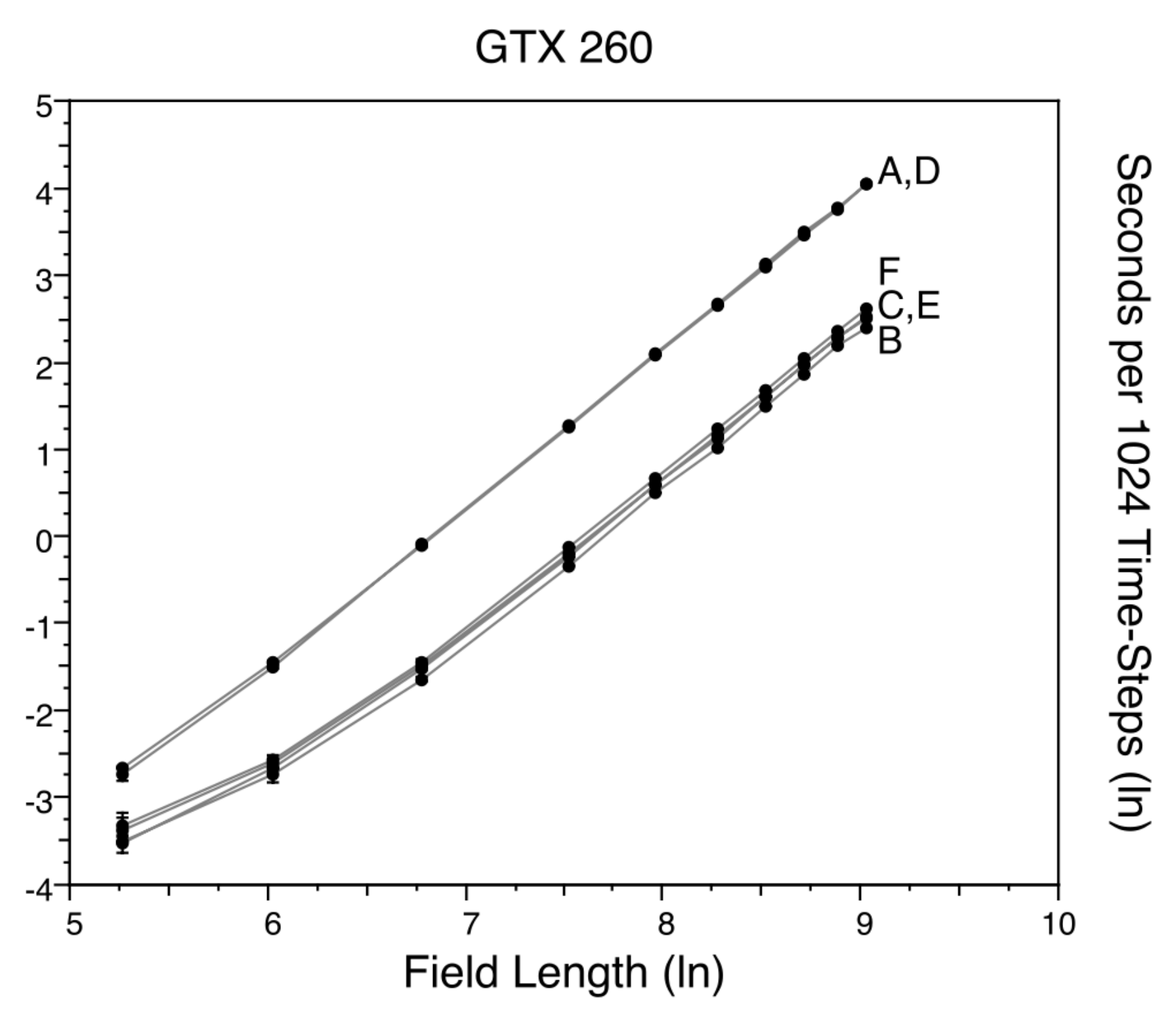

Figure 5: Performance results of the size LGCA kernels (A-F) on a GTX 260+ in ln-ln scale.

Summary

The *CUDA* version scales well with the number of cores in the GPU architecture. However, while the GPU is capable of significantly accelerating the performance of the algorithm, the triangular lattice requires some unexpected choices for GPU memory usage to attain optimal speed. Normal rectilinear simulations that exploit data-parallelism of the GPU performs best using **Texture** memory for lattice access whereas the triangular lattice of the LGCA simulation performs best with simple **Global** memory.
This appears to be due to alternating access patterns required for the triangular mesh, the differing access patterns restrict the GPU from making full use of the GPU texture cache.
An interactive-time graphical rendering of the simulated system with OpenGL graphics code was implemented and this has proved invaluable for debugging purposes. Although, the performance analysis has been done without graphics overheads. An area for future work is to combine the power of multiple GPUs to accelerate both the graphics **and** the simulation itself.
Planned extensions to this work include using suitable hyper-cubic face-centred cubic lattices to yield an effective 3-d simulation system with appropriate symmetries. It is expected this will also scale well on the GPU architecture although will quite likely also present some unexpected memory choices due to the symmetry constraints.

References

[1] O.Penrose, A.Buhagiar: Kinetics of nucleation in a lattice gas model: Microscopic theory and simulation compared. J.Stat.Phys **30** (1983) 219–241

[2] Dab, D., Boon, J.P., Li, Y.X.: Lattice-Gas automata for coupled reaction diffusion equations. Phys.Rev.Lett. **66** (1991) 2535–2538

[3] R.Benzi, S.Succi, M.Vergassola: The lattice boltzmann equation: theory and applications. Rome Preprint ROM 2F-92-10 (1992)

[4] Advisory Group for Aerospace Research and Development: Special Course on Modern Theoretical and Experimental approaches to Turbulent Flow Structure and its Modelling, 7 Rue Ancelle, 92200 Neuilly Sur Seine, France, Advisory Group for Aerospace Research and Development, NATO (1987) AGARD Report No 755.

[5] Kalland, K.M.: A Navier-Stokes Solver for Single and Two-Phase Flow. Master's thesis, Faculty of Mathematics and Natural Sciences, University of Oslo (2008)

[6] Simon, H., Golub, G.H., Huang, L.C., Tang, W.P.: A Fast Poisson Solver for the Finite Difference Solution of the Incompressible Navier-Stokes Equations. SIAM Journal on Scientific Computing **19** (1998) 1606–1624

[7] d'Humieres, D., Lallemand, P.: Lattice gas automata for fluid mechanics. Physica A: Stat. Mech and its Applications **140** (1986) 326–335

[8] Toral, R., Marro, J.: Cluster kinetics in the lattice gas model: The Becker-Doring type of equations. J. Phys. C: Solid State Physics **20** (1987) 2491–2500

[9] Wylie, B.J.: Application of Two-Dimensional Cellular Automaton Lattice-Gas Models to the Simulation of Hydrodynamics. PhD thesis, Physics Department, Edinburgh University (1990)

[10] Gould, H., Tobochnik, J., Christian, W.: An Introduction to Computer Simulation Methods. 3rd edn. Addison-Wesley (2006) ISBN: 0-8053-7758-1.

[11] Boghosian, B.M.: A survey of techniques for simulating partial differential equations with lattice gases. TMC Technical Note CA89-1, Thinking Machines Corporation (1989)

[12] Hawick, K.A., Leist, A., Playne, D.P.: Mixing Multi-Core CPUs and GPUs for Scientific Simulation Software. Technical Report CSTN-091, Computer Science, Massey University (2009)

[13] Playne, D.P., Johnson, M.G.B., Hawick, K.A.: Benchmarking GPU Devices with N-Body Simulations. In: Proc. 2009 International Conference on Computer Design (CDES 09) July, Las Vegas, USA. Number CSTN-077 (2009)

[14] Leist, A., Playne, D., Hawick, K.: Exploiting Graphical Processing Units for Data-Parallel Scientific Applications. Concurrency and Computation: Practice and Experience **21** (2009) 2400–2437 CSTN-065.

[15] Hawick, K., Leist, A., Playne, D.: Damaged Lattice and Small-World Spin Model Simulations using Monte-Carlo Cluster Algorithms, CUDA and GPUs. Technical Report CSTN-093, Computer Science, Massey University (2009)

[16] Leist, A., Playne, D., Hawick, K.: Visualising Spins and Clusters in Regular and Small-World Ising Models with GPUs. Technical Report CSTN-108, Computer Science, Massey University (2009)

[17] d'Humieres, D., Lallemand, P., Frisch, U.: Lattice gas models for 3d hydrodynamics. Europhys. Lett. **2** (1986) 291–297