Visual Simulation of a Multi-Species Coloured Lattice Gas Model
T. S. Lyes and M. G. B. Johnson and K. A. Hawick
2012

Understanding complex fluid flow patterns is still a computationally challenging problem and visualising computer simulations in this area is an important tool in investigating emergent complexity in fluid systems. Mixing and unmixing of complex multi-species fluid systems is particularly difficult to tackle using conventional field equational methods. We describe our multi-species lattice gas software for simulating and visualising multi-species fluid systems. We describe how a coloured lattice gas model was developed and run on a graphical processing unit using NVIDIA’s compute unified device architecture (CUDA) to yield a speed up over a typical CPU performance of one hundred fold. This then supports simulation of system sizes large enough to reveal interesting emergent complexity, and we present some initial scientific observations.

Keywords: lattice gas; coloured; FHP-3; CFHP; OpenGL; cellular automata; simulation

BiBTeX reference:
@INPROCEEDINGS{CSTN-147,
  author = {T. S. Lyes and M. G. B. Johnson and K. A. Hawick},
  title = {Visual Simulation of a Multi-Species Coloured Lattice Gas Model},
  booktitle = {Proc. Int. Conf. on Scientific Computing (CSC’12)},
  year = {2012},
  pages = {115-124},
  address = {Las Vegas, USA},
  month = {16-19 July},
  publisher = {CSREA},
  institution = {Computer Science, Massey University},
  keywords = {lattice gas; coloured; FHP-3; CFHP; OpenGL; cellular automata; simulation},
  owner = {kahawick},
  timestamp = {2012.05.03}
}

This is a early preprint of a Technical Note that may have been published elsewhere. Please cite using the information provided. Comments or queries to:
Prof Ken Hawick, Computer Science, Massey University, Albany, North Shore 102-904, Auckland, New Zealand.
Complete List available at: http://www.massey.ac.nz/~kahawick/cstn
Visual Simulation of a Multi-Species Coloured Lattice Gas Model

T.S. Lyes, M.G.B. Johnson and K.A. Hawick

Computer Science, Institute for Information and Mathematical Sciences,
Massey University, North Shore 102-904, Auckland, New Zealand
email: { t.s.lyes, m.johnson, k.a.hawick }@massey.ac.nz
Tel: +64 9 414 0800 Fax: +64 9 441 8181

March 2012

ABSTRACT
Understanding complex fluid flow patterns is still a computationally challenging problem and visualising computer simulations in this area is an important tool in investigating emergent complexity in fluid systems. Mixing and unmixing of complex multi-species fluid systems is particularly difficult to tackle using conventional field equational methods. We describe our multi-species lattice gas software for simulating and visualising multi-species fluid systems. We describe how a coloured lattice gas model was developed and run on a graphical processing unit using NVIDIA’s compute unified device architecture (CUDA) to yield a speed up over a typical CPU performance of one hundred fold. This then supports simulation of system sizes large enough to reveal interesting emergent complexity, and we present some initial scientific observations.

KEY WORDS
lattice gas; coloured; FHP-3; CFHP; OpenGL; cellular automata; simulation;

1 Introduction
Simulating the dynamics of fluid flow can often be a very complicated task. Lattice gas cellular automatons (LGCA’s) [1–3] can be used to approximate the behavior of fluid flows with a certain degree of accuracy—they may fail to model all physical dynamics of fluids at a microscopic level, however they can produce satisfactory results on a macroscopic level. Many variations of LGCA approaches, including using a lattice Boltzmann approach [4–6], have already been developed and improved upon to produce faster, more complex and more accurate results. More complex geometric arrangements such as reflecting boundaries, barriers, monolayer deposition [7] and the imposition of gravitational bias [8] are also possible.

Simulation work with large scale lattice gas models [9, 10] has important implications for studies of complexity and emergence on logarithmic length scales. Work is reported in the literature on the stability limits of lattice gas models [11] and lattice gas models are also of use in modelling crowd dynamics [12]. Our particular interest is in modelling multi-species systems lattice gas systems [13–15] where a fluid or some other modelled microscopic constituents either separate out from a random initial state or mix and co-dissolve under the right parametric circumstances.

In a previous study [16], a LGCA was used to simulate...
a lattice gas on a graphics processing unit (GPU), as LGCA models are highly parallelisable [17–21]. This allowed for better levels of optimization and execution speed of the LGCA, as well enabling the model to be simulated at a larger scale. A method of visualizing the lattice gas behavior on the fly, however, was not developed, and thus has become one of the goals of this paper.

Most standard LGCA models apply to a lattice containing only one type of gas (commonly referred to as a 'species'). There do exist, however, some models which take into account the option of having multiple species present in the same lattice. Such models are good for simulating diffusion effects between multiple gases, and can produce some interesting results depending on the properties of each species of gas. Models which deal with multiple species of gases are often called “coloured” models, as different particles can be represented visually using different colours. While accounting for a second species of particle will affect the overall performance of the model, there are ways to reduce this impact.

In this article we present a coloured LGCA based on the code from [16], which may be visually monitored on-the-fly while also being able to be parallelized easily for performance investigation. Firstly, a more in-depth explanation of the LGCA models used will be given in Section 2 as well as a description of the OpenGL [22] and Compute Unified Device Architecture (CUDA) [23] technologies used in Section 3. In Section 4 some screenshots of the resulting lattice gases will be shown, along with descriptions and additional information. In Section 5 some the performance results are put forth, and some coding issues and difficulties will also be discussed. Finally, conclusions and future work will be offered in Section 6.

2 The LGCA Models

While many models exist for using LGCAs to simulate fluid flow, the FHP-3 model [1] (named after Frisch, Hasslacher and Poumeau, who first introduced the model) allows for superior efficiency in simulation. Firstly, a more in-depth explanation of the LGCA models used will be given in Section 2 as well as a description of the OpenGL [22] and Compute Unified Device Architecture (CUDA) [23] technologies used in Section 3. In Section 4 some screenshots of the resulting lattice gases will be shown, along with descriptions and additional information. In Section 5 some the performance results are put forth, and some coding issues and difficulties will also be discussed. Finally, conclusions and future work will be offered in Section 6.

2.1 The FHP-3 Model

The FHP-3 model [1], or known simply as a "coloured" lattice gas, is a variation of the regular FHP-3 LGCA, in which particles may belong to two different species (colours). In a standard LGCA, every particle is designed to be mechanically identical. This remains true in a coloured LGCA - all particles remain identical in all ways except colour. This means the CFHP model need not create a larger, more complex set of collision rules to compensate for two different particles in the lattice. Indeed, the CFHP model uses exactly the same set of rules as the FHP-3 model. How, then, does one keep track of the colour of each particle in the lattice? The number of coloured particles need only be conserved in each collision. Before any collision rules are applied to each site, the number N of each coloured particle can be counted. Once the collision rules have been applied using the standard FHP-3 collision ruleset, the colour attribute can be randomly assigned to N number of channels using a lookup table. This need only be done with one of the two colours, as the second colour will automatically be assigned to those channels not randomly assigned the first colour attribute. Put simply, if the particle is not red, it is blue.

In working with the code for the coloured lattice gas, it was decided that we would develop an OpenGL rendering system that would allow us to visualize the lattice gas in real time, with the ability to monitor the different colours separate from each other. The resulting program allowed for visualization of the lattice in five
Figure 3: An CFHP model lattice setup - notice each direction has two channels, one for red particles and one for blue particles

Figure 4: Sample screen dump of the lattice rendered using arrows

3 OpenGL Lattice Rendering and CUDA

The code to simulate the coloured lattice gas was written both as a sequential OpenGL program for visualization purposes, and in CUDA for performance monitoring. The coloured lattice would be expected to run slower than the single-species lattice, as there is extra work in counting coloured channels and assigning random channels coloured attributes.

As opposed to the single-species lattice, the coloured lattice used three different arrays to store the lattice data - a "master" array on which the collision rule set was applied, and two arrays containing the information needed for the red and the blue lattices. This allowed for easy access by OpenGL when rendering only the red or only the blue lattice, while additionally removing the need to add extra rules to the rule set to be applied to the master lattice in order to account for the different coloured particles.
pendently for each time step. Cells in the lattice were also bit-packed to improve performance. The CUDA version of the code was used solely for performance checking, as any rendering would have slowed it down. The kernel itself followed an algorithm roughly resembling the following algorithm (Algorithm 1).

Algorithm 1: a typical coloured lattice gas kernel

```plaintext
for all cells in lattice do
    count red particles
    determine input channels
    perform bit-packing (unpack)
    apply collision rule
    randomize colour channels
    conserve red particles
    apply colour channels
    perform bit-packing (pack)
end for
update main lat, red lat, blue lat
```

The coloured lattice was simulated twice, once using the standard rule set, and once using a biased rule set. Similar to [16], a barrier was added to allow for observation of the behavior around impassible cells. The barrier was set up as a vertical line of cells centered horizontally in the lattice, roughly one fifth of the width from the left hand side. To allow for the most efficient demonstration of the gas interaction with the barrier, initial particle velocities were set to be perpendicular to the barrier, towards the right. Additionally, the lattice particles would wrap around right-to-left.

4 Visualisation Results

The particles in each screenshot are coloured in relation to their velocity. Darker areas of the lattice indicate an absence or lower concentration of particles in that particular area - this usually occurs around barriers. For each of the red and blue lattices a separate colour spectrum was used to represent the velocities of the particles. While visualisation of each coloured lattice on its own presented no problems, rendering both red and blue lattices together presented some visualisation problems as it is difficult in some places to distinguish the red lattice’s green particles from the blue lattice’s teal particles (and several other similar colour match ups as well). For reference and comparison, a visualization of the single-species lattice gas was also shown.

Each lattice configuration was simulated for 4096 time steps, while being observed and captured at the start, middle and end points of the simulation. The resulting screenshots were produced.

Both the standard and biased versions of the lattice were initialized exactly the same. Figure 5 shows this initial configuration. All particles above half the height of the lattice were initialized red, and all below were initialized as blue. In the early stages of the project, the red and blue particles were assigned random initial positions in the lattice - this configuration, however, produced no meaningful results, as interactions between red and blue particles were almost imperceptible, and diffusion behavior could not be observed (as both species were already fully diffused). The barrier cannot be seen yet in this example, as initial particle velocities have yet to take effect.

Figure 6 shows the state of the lattice at the half-way point of the simulation. This version uses the standard rule set. In these figures the barrier to the left has become clearly visible and the particles have begun to move around it. It is interesting to watch the behavior of the red and the blue lattices on their own. Although the majority of each species’ fellow particles remain concentrated at their respective sides, some particles are still forced in the opposite direction by the barrier and other incoming particles from the left. This can be shown by a "hook" like shape created as the particles are pushed down and around the edge of the barrier (this is more clear in the red lattice as the colour is lighter). This type of behavior occurs because the particles have no cohesion between others of the same species, and instead operate independently of each other. Both lattices display some slight diffusion as some blue particles begin to slowly move upward, and in turn some red particles slowly move downward.
Figure 6: State of the lattice at time step 2048 (half-way point of the simulation) using standard rules. From top-left to bottom-right: the standard single colour lattice, both coloured lattices, red only and blue only

Figure 7: Final state of the lattice (after 4096 time steps) using standard rules. From top-left to bottom-right: the standard single colour lattice, both coloured lattices, red only and blue only

Figure 8: Half-way state (time step 2048) of the lattice using a biased rule set. From top-left to bottom-right: the standard single colour lattice, both coloured lattices, red only and blue only

Figure 7 shows the final product of the lattice gas simulation after 4096 steps. The flow of the gas has now become much more apparent due to the colour change of the particles and the shape of the gas flow behind (to the right) of the barrier. The behavior of the red and blue lattices is also more apparent, as they form stronger curves around the barrier and the diffusion between the two species has increased.

Figure 8 shows the half-way state of the biased lattice. Notice how the shape of the single-species lattice is identical to that in Figure 6, however the shapes of the red and blue lattices are radically different. Red particles, while initially starting at the top (see Figure 5) have moved towards the bottom of the lattice, being completely replaced by blue particles at the extreme top of the lattice. Consequently, blue particles appear to have been “pushed” upwards by the red - a large area at the bottom of the lattice now contains solely red par-
particles. This area has a curious "hump" shape, as while the red particles are biased downwards, they are still effected by the right-ward velocity and barrier and are thus collecting mostly at the bottom just to the right of the barrier. Both these behaviors form what seems to look like "layers" when both lattices are rendered together.

Finally, Figure 9 shows the final result of the biased coloured lattice gas simulation. The red particles have almost completely migrated to the bottom of the lattice, while the blue have almost completely moved to the top - essentially the positions of the two species have been inverted. The effect of the velocity and barrier is still apparent, however the particles seem to be less inclined to separate - for example, the curve of the red particles moving up over the barrier is much sharper than the lattice using non-biased rules (Figure 7) and once the particles move over the edge of the barrier, they curve sharply back down towards the bottom.

![Figure 9: Final state of the lattice (after 4096 time steps) using a biased rule set. From top-left to bottom-right: the standard single colour lattice, both coloured lattices, red only and blue only.](image)

### 5 Discussion

The program running the coloured lattice when noticeably slower than the single-species lattice when they were both run sequentially using OpenGL to display the lattice after every time step. Due to the fact that the OpenGL rendering only used simple GL points with no 3d graphics or costly effects, the slowdown was probably due to the extra work being performed by the kernel during the update phase (as opposed to the display phase). To mitigate the overhead of copying to, and displaying the lattice on the CPU we could implement the OpenGL rendering directly on the GPU.

Any attempt to increase the lattice size would make the simulation run too slow to be viewed in real-time speed - while running nicely at a lattice size of 256x256, an increase in lattice size to 512x512 would display at around two frames (time steps) per second which is watchable but slow, while increasing the size to 1024x1024 would display one frame every five seconds. It is clear that increasing the size of the lattice needs parallelization to make the simulation run smoothly in real time.

The following table shows the results of the performance testing using the CUDA version of the coloured lattice gas. The graphics card used for the performance test was the NVidia Quadro 4000. It was tested using different sizes of the lattice (256x256 up to 4096x4096), recording the average time taken in seconds to execute each kernel once. As expected, the coloured lattice took longer than the standard lattice in all lattice sizes, although it is almost unnoticeable in the smaller sizes (256 and 512). As the lattice gets larger, the time gap between the two lattices becomes exponentially larger until the coloured lattice is almost half a second slower than the single species lattice for each kernel execution.

<table>
<thead>
<tr>
<th>Lattice size</th>
<th>Standard Model seconds per frame</th>
<th>Coloured Model seconds per frame</th>
</tr>
</thead>
<tbody>
<tr>
<td>256</td>
<td>0.0084</td>
<td>0.0101</td>
</tr>
<tr>
<td>512</td>
<td>0.0131</td>
<td>0.0189</td>
</tr>
<tr>
<td>1024</td>
<td>0.0315</td>
<td>0.0537</td>
</tr>
<tr>
<td>2048</td>
<td>0.1044</td>
<td>0.1963</td>
</tr>
<tr>
<td>3072</td>
<td>0.2181</td>
<td>0.4290</td>
</tr>
<tr>
<td>4096</td>
<td>0.3996</td>
<td>0.7895</td>
</tr>
</tbody>
</table>

Table 1: Performance results of standard lattice gas vs. coloured lattice gas models, parallelized using CUDA with global memory. Times accurate to approximately ±0.00005.

From these results we can also see the power and speed-up gained by using CUDA and the graphics card - as mentioned earlier, if we assume negligible time cost from rendering, a single kernel execution on a 1024x1024 lattice run sequentially would take approximately five seconds. Compared to the 0.0537 seconds for a 1024x1024 lattice on the GPU, the parallelized lattice is almost 100 times faster. The coloured model adds only a manageable overhead cost.
6 Conclusions

We have developed a two-dimensional coloured lattice gas model, using OpenGL to render the model and observe it’s behaviour in real-time. The model is able to easily be parallelized in CUDA for improved performance and scaling. For additional performance improvement, four cells in the lattice have been bit-packed into one 32-bit integer. The program allows up to two species of particle to exist in the lattice and each species may be viewed separately as needed. The behaviours of different species can be biased as required by generating a separate colour randomization lookup table with an increased chance to assign the colour to the desired direction channel or channels.

This encoding approach to the multi specied lattice gas makes good use of memory and hence aids cache-locality. Preliminary observations suggest the model is large enough to support measurements of thermodynamic mixing properties and experimental comparisons with realistic complex fluids.

We plan on extending this work into creating a three-dimensional lattice gas model and visualizing it in the same way, as well as models with varied densities. This would be expected to work well on a GPU architecture as well, although it is unsure whether the memory constraints which may arise in a three dimensional model will be manageable.

References