Physics 420/580: Assignment 1  
Due: Wednesday, September 23, 2009

In this assignment, you'll be exploring the FHP lattice gas introduced by Frisch, Hasslacher and Pomeau [Phys. Rev. Lett. 56, 1505 (1986)]. The automaton consists of a regular triangular array of sites holding identical particles of unit mass that are either stationary or moving with one of six discrete velocities of equal magnitude directed along the lattice links. No more than one particle per site can exist in any of these seven states. Particle updates are carried out in two passes. The first is a propagation step in which each non-stationary particle is made to move along its velocity vector to an adjacent site. The second is a collision step in which the following net-momentum-preserving reconfigurations are carried out.

In the figure above, black sites are occupied by a stationary particle, and white sites are unoccupied. The occupancy of the grey sites is unspecified and may be zero or one. When there are two possible reconfiguration pathways—as is the case for the two- and four-velocity collisions—one of the branches is chosen randomly with probability 1/2.

To start, download and unpack the Assignment1.tar.gz archive from the class website. The Assignment1 directory contains a program file lattice_gas.cpp that (partially) implements an extension of the FHP model that allows for reflection off fixed obstacles. In the code, the triangular lattice is sheared so that it can be represented as a square lattice (which is itself stored as a one-dimensional C array in row-major order). In this geometry, two of the nearest neighbours to each site are connected diagonally across a square plaquet. The six nearest-neighbour directions are described by an enumerated type ver_t and named for the corresponding compass points:

The program makes use of a compressed storage format. The state of each lattice site is stored in a single byte, with each of the eight bits flagging the presence of a velocity, stationary particle, or obstacle.
Issuing the commands

$ make
$ ./lattice_gas

Usage: lattice_gas (setup=0,1,...,4) (0 < concentration < 1)

from inside the Assignment1 directory will build and execute the program. Note that lattice_gas expects two command line parameters. The first selects one of five simulation modes: (0) uniform gas; (1) circular dilution wavefront; (2) linear dilution wavefront; (3) box with unequal pressures inside and outside; (4) fluid flow past an elliptical barrier. The second parameter controls the number density of the fluid. When both options are correctly specified, the program opens a new window and animates the resulting hydrodynamics.

1. (4 points) The lattice connectivity is defined through a collection of functions named indexN, indexE, ... indexNW. Implement these functions, making sure that the boundary conditions are properly defined for the triangular lattice.

2. (4 points) The function report writes the total particle number and the total x- and y-directed momenta to the terminal after each time step. The particle number at each site, computed with occupancy, is accumulated in an integer variable n. The momenta, stored as double-precision floating-point variables px and py, are incremented using the inc_momentum function. Write the body of the occupancy and inc_momentum functions. Verify that n, px, and py are conserved quantities.

   $ ./lattice_gas 0 0.3
   336170 -347 -211.31
   336170 -347 -211.31
   336170 -347 -211.31
   336170 -347 -211.31
   336170 -347 -211.31

3. (5 points) Extend the collision function to account for four-velocity collisions. Check that n, px, and py are still conserved.

4. (3 points) Run lattice_gas in modes 1 and 2 for various values of the concentration. How does the behaviour change when you turn off all collisions? Give me a brief physical explanation for your observations.

   void animate(int)
   {
     propagate();
     //collision();
     if (is_flowing) flow();
     report();
     glutPostRedisplay();
     glutTimerFunc(delay,animate,0);
   }

5. (6 points) Run the simulation in mode 3 and dump the output to a file:

   ./lattice_gas 3 0.65 > outfile.dat
Make a plot of the px and py values versus the simulation clock. Use this information to make a rough estimate of the pressure difference between the fluid inside and outside the box.

*6. (8 points) Run the simulation in mode 4 with concentration 0.4. Roughly determine the time needed for the flowing liquid to reach its steady state. Measure the steady-state velocity field, time-averaged and coarse-grained over several lattice sites. Produce a vector plot of the resulting smoothed velocity field.