**Cellular Automata and Lattice Gases**

The subject of lattice gases involves being able to stop between the microscopic and macroscopic levels without having to pass through a continuum description. Cellular automata (CA) are a more general concept of lattice gases. Any CA has a set of connected sites, states that are allowed on the sites, and a rule as to how they should be updated. An example of CA is Conway’s Game of Life, where sites on a grid can be born, survive, and die in future generations based on the number of occupied neighboring sites.

The FHP rule is an important example of a lattice gas. It is operated in 2D on a triangular lattice. There are six bits that specify the state of each site. Each bit represents a particle on one of the six links around the site. There are different update rules for handling with collisions. I think it is very helpful that the authors provided images (Figure 9.2) to illustrate how different collisions are handled. Reading the text made it somewhat clear, but looking at the graphs made it a lot clearer as to exactly how the collisions are handled.

HPP is a simpler rule related to FHP, except that it operates on a square lattice. Four bits are used to represent the sites, and direction-changing collisions are only allowed when two of the particles meet head-on). Even though HPP is a lot simpler than FHP, I think it would have been helpful if an image had been provided to illustrate how collisions are dealt with.

I like how the authors described a use for lattice gases in computing. I think it’s interesting how the bits at a site can be used as indices into a look-up table, simulating memory. I also think that it is very interesting that modest hardware can perhaps exceed the performance of supercomputers for CA problems. I would’ve liked it if the authors had elaborated a little bit on this concept.

I like how the authors provided an example of how the rendering of 3D graphics can be modeled by cellular automata rules. I also liked the example with the billiard balls colliding to generate the AND function while if one of the streams of balls is continuous it generates the NOT function. I think it was extremely helpful that an image was provided to illustrate the concepts with the billiard balls (Figure 9.4). I’m still a little confused as to how the delaying works, and the crossover image seems a little cluttered. I think that maybe each of those images could’ve been in their own “figure” and there could’ve been a better explanation for how those functions are represented with the billiard ball logic.