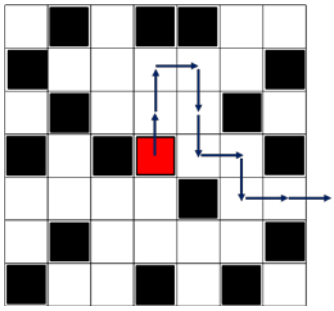
Chem 160/260 Project 2

You must log on to biopathway to do this Project. For full credit complete this Project before 12:00AM (midnight) on Monday, November 23, 2015. You can turn this in up to one week late (by midnight Monday, November 30, 2015) for a 25% penalty**. If you choose to turn your Project in late, email me so I know to look for it the following week.**

Project overview: Choose **one** of the following project assignments and complete it in your **`~/projects/project2`** directory. **Extra credit:** Do both of the project assignments.

Project Problem #1

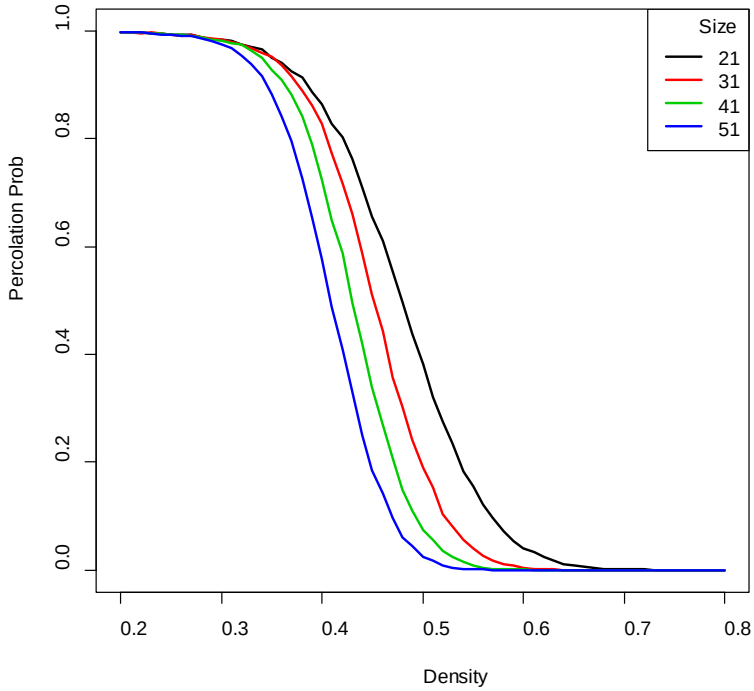


Percolation is the motion of a liquid through a porous solid phase. Interestingly, percolation is a form of *critical phenomenon* which means that there is a specific percolation threshold for the density of the porous medium. For densities below that threshold the liquid will make its way through the media, for densities above that threshold it will not. In this project we will use the random motion of a square on a 2D matrix to determine the percolation threshold density. An example of this system is shown at right. The black squares are randomly placed on the lattice and don't move. The red square starts at the center of the system and randomly moves left, right, up, or down, traveling only through open squares until it reaches the edge of the system.

The starting point for this simulation program will be the diffusion program, **`diffuse.py`**, we discussed in class 16 and is provided in your project2 directory. You need to make the following modifications to the program to complete this project:

- Use **`argv`** to read in a variable called **`density`** which will have a value between 0.0 and 1.0 that specifies the fraction of cells that are occupied by the non-moving squares.
- Set a variable **`maxsteps`** that will be the maximum number of steps the particle can move as it tries to percolate out of the grid. Set this to 10000 (but you may want to set this to a lower value while debugging your program).
- Set the **`npart`** variable to 500 (but you may want to set this to a lower value while debugging your program).
- Initialize a counter called **`perc`** to zero which will count the number of times the particle successfully moves from the center of the system to an edge.
- Just inside the loop over particles, loop over all cells in the grid (requires 2 nested loops) and set each cell to "1" with a probability given by **`density`** which will require you to use the **`random()`** function.
- Convert the "**`while 1`**" loop to a for loop over **`maxsteps`**
- After selecting a random step, test to see if the cell the particle would move into is already occupied (i.e. use an if statement to see if the new cell already has a value equal to 1). If it is occupied then do not move there and use a **`continue`** command to go to the next iteration of the for loop over **`maxsteps`**.
- If the particle reaches the edge of the system (as determined by the same if statements used in diffuse.py) then increment **`perc`** by one and go to the next particle.
- At the end of the program print the probability of the particle percolating out of the system (i.e. ratio **`perc/nparts`**).

To check your answers you can compare your results with the graph on the following page that gives the percolation probability vs density for systems with side=21, 31, 41, and 51.



Project Problem #2

The Ising model has a different critical temperature in different numbers of dimensions. For this project, you will modify your Ising model program from Class 17 to run in three dimensions (and provided in the project2 directory. In three dimensions the energy of each spin will depend on its six nearest neighbors—the original 4 neighbors, left, right, up, down, as well as front and back. The energy equation will be the same, +1 for each neighbor with an opposite spin and -1 for each neighbor with the same spin, which will give you a minimum energy of -6 per spin. Name your new program **`Ising3D.py`**. Once completed, you can compare your results at different temperatures to the following graph of average energy and heat capacity vs. temperature for a 20x20 3D Ising model.

Hint 1: this will require using the 3D list structure (**`list3d`**) that we talked about in Class 15.

Hint 2: Be sure to change the energy equation in both the calculation of the total energy and the ΔE.