Liquid or Gas

New Mexico Super Computing Challenge

Final Report

April 1, 2006

Team #74

Oñate High School

Team Members: Luke Murphy, Petrina Strader,

and Jason Li

Sponsors: Donald Downs and Josefina Dominguez

Table of Contents

Executive Summary

Our project's main objective is to model water molecules changing state from liquid to gas molecules. We are using the Ising Model. We started by doing simple systems that would be easy to solve by hand to learn about the Ising Model. We modified the Ising Model to express water and its phases because the Ising Model was originally used for Ferro-magnets, but the same principles applied well to thermal-equilibrium and phase shifting. After interpreting how the model works, we programmed the model in Java. In our program we started using very small arrays of molecules, and we are now using an array of 900 molecules. Using the program, we conducted several experiments by changing the initial concentration of liquid and gas molecules. The experiments were conducted at, below, and above boiling point, which is around 373 degrees Kelvin, by varying the starting ratio of liquid to gas. In our program, we are using the concept of pseudo-randomization to get various results and graphs. We used Java's built-in graphics to express our results, by displaying liquid as blue and gas as red. The results, of course, vary wildly which is perfectly representative of the vast possibilities for any particular system.

Problem

We are investigating the phase change of water at the temperature at, below, and above the boiling point (373 degrees Kelvin). At boiling point, the proportion of water molecules is around half in a liquid state and the other half in the gaseous state. Through the programming, we are testing the problem by trying to vary the percentage of liquid and gas at boiling point and examining the results and checking for the number of times that it goes through the process of changing a molecule's state through pseudo-randomization before it changes into one phase. Also, due to pseudo-randomizing in our program, every graph in Java looks a little different until it reaches one phase completely.

Ising Model and Our Method

To investigate our problem, we used the Ising Model. The Ising Model is a mathematical model in statistical mechanics. It was first used to determine the magnetization of a Ferro-magnet. The Ising Model is named by Ernst Ising who studied the model for his doctorate degree. It was actually created by William Lenz in 1920. The model is distinguished for being the simplest model that can express a large system of molecules and illustrate the behavior and actions of the molecules. It might be considered the simplest model, but it has been extremely difficult to compute the Ising Model even with advanced mathematics and computer science.

The Ising Model is based on the concept of "spins" in pairs that can easily be represented as arrows pointing up or down. The model uses negative one and positive one to represent the spin. In the problem of phase probability, negative one represents an arrow pointing down, and positive one represents an arrow pointing up. The probability of phase depends on the variables, energy and temperature, in degrees Kelvin. A molecule and its neighbor(s) make a pair. When the pair has the same direction, it is considered as being parallel, and when they have different directions, they are antiparallel. Energy is determined by the antiparallel pairs subtracting the parallel pairs. The temperature is a variable that is determined by the person conducting the project. All these calculations can be done by hand, but when there are many molecules in a system, it will be extremely difficult to conduct by hand and will be feverishly time consuming. With a large system, it is impossible even for a supercomputer to do it. For example, a one hundred by one hundred system has two to the ten-thousandth possible variations.

To calculate the probability with a massive system, we will pseudo-randomly pick a molecule in the system and change the spin on the molecule to figure out if it will affect its neighbors. This calculation of probability is the Monte Carlo Method. A molecule is selected pseudo-randomly, and depended on a pseudo-random generated number will either stay the same state or flip to the opposite. This operation is formed over and over on different molecules until all molecules are one state.

For our program, we modified the Ising Model by having every molecule be affected by its nearest neighbors and every molecule has four nearest neighbors (above, below, left, and right). We made all molecules have four neighbors by programming a "wrap around" that let molecules on the edge of the array be affected by the corresponding molecule on the other side, as if, the array was a sphere. By determining that every molecule had four neighbors, the only available amounts of energy (antiparallel subtract parallel) were -4, -2, 0, 2, and 4. The program pseudo-randomly picks a molecule and decides to flip it or not. Afterwards, we assigned probabilities for the molecules to change phase. The molecules would change constantly until completion. Completion is when the whole array reaches one state. We ran twenty trials by having the molecules in an array be half liquid and half gas, which is around boiling point of water, 373 degrees Kelvin. Furthermore we ran twenty trials starting with 60% liquid and 40% gas, which is probably a little below boiling point, and recorded results. We also ran twenty trials with 40% liquid and 60% gas, probably over boiling point, and got various results.

Results

Data for 60% Liquid and 40% Gas

Trials	Number of Repetitions	Completion State and Color
1	745000	Liquid(Blue)
2	566000	Liquid(Blue)
3	270000	Liquid(Blue)
4	640000	Liquid(Blue)
5	1311000	Gas(Red)
6	4900000	Liquid(Blue)
7	256000	Liquid(Blue)
8	806000	Gas(Red)
9	1523000	Liquid(Blue)
10	833000	Gas(Red)
11	1628000	Gas(Red)
12	1145000	Liquid(Blue)
13	834000	Liquid(Blue)
14	5231000	Liquid(Blue)
15	508000	Gas(Red)
16	1009000	Liquid(Blue)
17	602000	Liquid(Blue)
18	564000	Liquid(Blue)
19	275000	Liquid(Blue)
20	1406000	Liquid(Blue)

This data table expresses that when the systems starts out with more liquid than gaseous water molecules, there is a higher chance that the array of molecules goes to a complete liquid state than the gas state.

Data for 50% Liquid and 50% Gas

Trials	Number of Repetitions	Completion State and Color
1	1421000	Gas(Red)
2	303000	Gas(Red)
3	677000	Liquid(Blue)
4	3141000	Liquid(Blue)
5	426000	Liquid(Blue)
6	615000	Gas(Red)
7	1487000	Liquid(Blue)
8	1069000	Gas(Red)
9	1576000	Gas(Red)
10	613000	Gas(Red)
11	1032000	Gas(Red)
12	1597000	Gas(Red)
13	999000	Liquid(Blue)
14	750000	Liquid(Blue)
15	2039000	Liquid(Blue)
16	567000	Gas(Red)
17	2344000	Gas(Red)
18	1339000	Liquid(Blue)
19	451000	Liquid(Blue)
20	1039000	Liquid(Blue)

These sets of results and trials represents that there is around a fifty percent chance for an array at boiling point to become into a single state of liquid or gas evenly. In the twenty trials, the arrays of molecules went into the liquid state ten times and the gaseous state ten times.

Data for 40% Liquid and 60% Gas

Trials	Number of Repetitions	Completion State and Color
1	210000	Gas(Red)
2	282000	Gas(Red)
3	137000	Gas(Red)
4	630000	Gas(Red)
5	674000	Gas(Red)
6	669000	Liquid(Blue)
7	1058000	Gas(Red)
8	756000	Gas(Red)
9	843000	Liquid(Blue)
10	776000	Liquid(Blue)
11	967000	Gas(Red)
12	1227000	Gas(Red)
13	1599000	Liquid(Blue)
14	366000	Gas(Red)
15	415000	Gas(Red)
16	629000	Liquid(Blue)
17	454000	Liquid(Blue)
18	1128000	Gas(Red)
19	464000	Gas(Red)
20	1230000	Liquid(Blue)

The results clearly illustrate that when a system of molecules starts with more gas than liquid, the array usually goes into complete gas than liquid.

All these data tables express the amount of repetitions it needs to reach completion, going into one state. Repetitions is the number of times it goes through the process of pseudo-randomly choosing a molecule and determining to change the state of that molecule or not. For every trial, it went through the process many times before it reached one state.

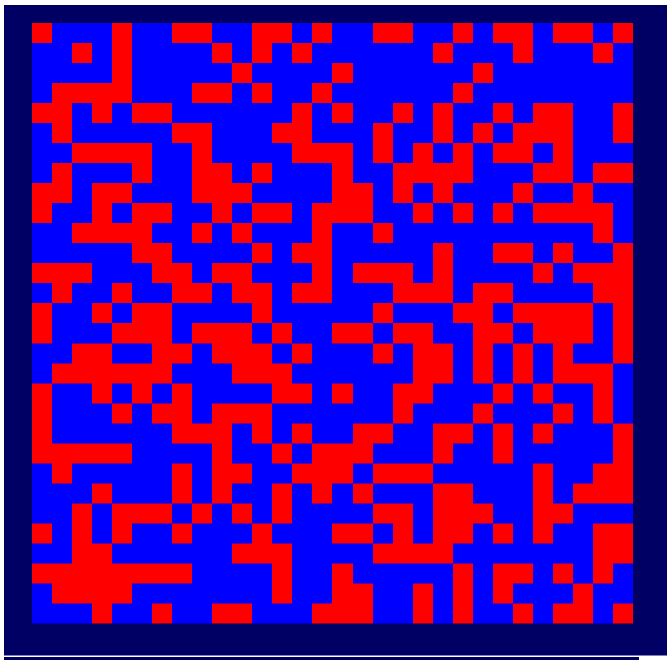
Conclusion

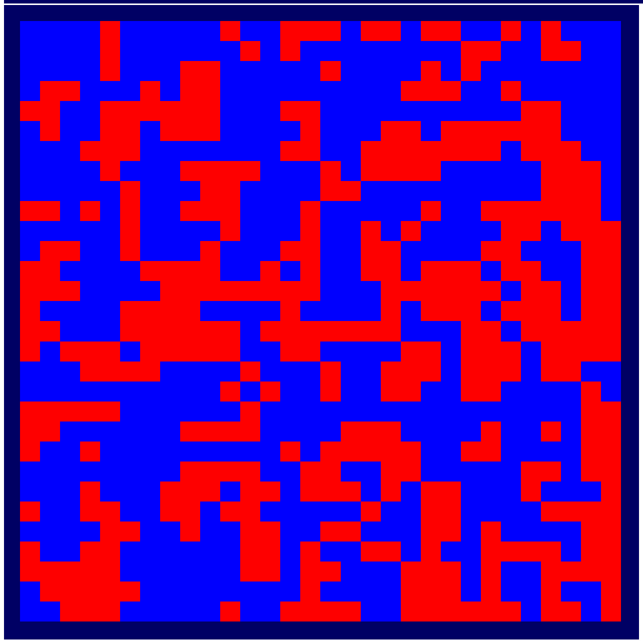
After running through twenty trials of three different starting situations, the results express that the amount of a molecule at a certain state affects the probability of the whole array changing into one complete state. When the array of water molecules start out having half liquid and the other half gas, there is around a fifty percent chance that the array will become liquid or gas. While when there is more of one molecule in one state than the other, the array will usually go completely into the state that having the higher initial concentration. For example, when the array starts off having more liquid water molecules than gas, the whole array goes into completely water than gas most of the time. Because it starts out with more of one molecule, there is a higher tendency to change into that molecule because of the effect from the molecule's neighbors, which is probably in the majority state.

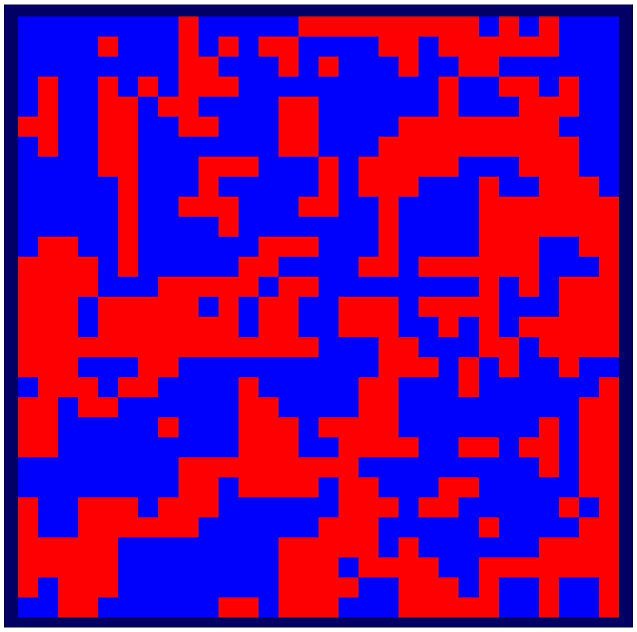
To expand our project, we could add variables such as volume and pressure to make the results seem more realistic, because volume and pressure affect phase change in the real world.

In conclusion, the program illustrates results that are affected by the beginning amount of molecules in one state. The results are displayed through the graphs that illustrate the states of the molecules and also express the change of phase for some molecules.

Graphs and Programming Code







These are graph displays of our results before its arrival to completion and these graphs are from the Java program that we are using. These three graphs are in order. The blue in these graphs is liquid and the red is gas. Each molecule in these graphs is displayed as a little square. As this array goes through repetition, the clusters of the molecules in the same state are more noticeable.

```
import javax.swing.JFrame;
import java.awt.Container;
import java.awt.GridLayout;
import java.awt.Color;
import java.awt.Graphics;
import java.awt.Graphics2D;
import java.awt.geom.Rectangle2D;
import java.awt.geom.Rectangle2D.Double;
import java.util.Random;
import javax.swing.Timer;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import java.lang.Math;

public class Ising extends JFrame implements ActionListener
 {
        final int NUMROWS=30;
        final int NUMCOLS=30;
        States s1=new States(NUMROWS,NUMCOLS);
        Random rand=new Random();
        Timer timer;
        int count;
        int row;
        int col;
        /***** Constructor *****/

        public Ising()
        {
                count =0;
                Container container = getContentPane();
                setBounds( 50, 50, 600, 600 );
                setBackground( new Color(0, 0, 100 ) );
                container.setLayout( new GridLayout(NUMROWS,NUMCOLS) );
                setVisible( true );
                int i,j;
//time per rep goes here
                timer = new Timer(5, this);
                timer.start();

                for(i=0; i<NUMROWS;i++)
                {
                        for(j=0; j<NUMCOLS;j++)
                        {
                                int z=rand.nextInt(1000000);
                                if(z>400000)
                                {
                                        s1.getMolecule(i,j).spin = -1;
                                }
                                else
                                {
                                        s1.getMolecule(i,j).spin = 1;
                                }
                        }
                }

        }


//Timed event...
        public void actionPerformed( ActionEvent event )
        {
        int NNE;
        double probSwitch;
        int rProb;
        int i,j;

        repaint();

        if( Math.abs(totalEn()) == NUMROWS*NUMCOLS )
        {
           System.out.println( "Final count =" +count);
           timer.stop();
        }
        else
        {
           count +=1000;
           System.out.println("Count =" + count);
                 for(i=0; i<1000;i++)
           {
//prob switching
              row = rand.nextInt(NUMROWS);
              col = rand.nextInt(NUMCOLS);
              NNE = s1.noseyNeighbor(row,col);
              if(NNE==4)
              {
                 probSwitch=1;
              }
              else if(NNE==2)
              {
                 probSwitch=.75;
```

```
                }
                else if(NNE==0)
                {
                   probSwitch=.50;
                }
                else if(NNE==-2)
                {
                            probSwitch=.25;
                }
                else
                {
                            probSwitch=0;
                }
                rProb=rand.nextInt(1000000);
                if(rProb < probSwitch*1000000)
                {
                   s1.getMolecule(row,col).spin *= -1;
                }
             }
          }
       }


//Fun with Paints
         public void paint( Graphics g )
         {
                  Graphics2D g2 = (Graphics2D)g;
                  Rectangle2D.Double rect2D;
                  int OldEnergy;
                  int NewEnergy;
                int i,j;


                for( row=0; row<NUMROWS; ++row)
                {
                        for( col=0; col<NUMCOLS; ++col )
                        {
                                if(s1.getMolecule(row,col).spin==1)
                                {
                                        g2.setColor( Color.red );
                                }
                                else
                                {
                                        g2.setColor( Color.blue );
                                }

                                rect2D = new Rectangle2D.Double(50+
col*10,50+row*10,10,10 );
                                g2.fill( rect2D );
                        }
                }
     }


     public int totalEn()
     {
        int energy = 0;
        for( row=0; row<NUMROWS; ++row)
                {
                        for( col=0; col<NUMCOLS; ++col )
                        {
                    energy += s1.getMolecule(row,col).spin;
                }
            }
        return energy;
    }

        /***** main *****/

        public static void main( String [] args )
        {
            Ising myIsing = new Ising();
                myIsing.setDefaultCloseOperation( JFrame.EXIT_ON_CLOSE );
    }
}
```

```
public class States
{
	// private numbers
	private Molecule[] [] A;

	// Constructors
	public States(int numRows, int numCols)
	{
		A= new Molecule[numRows][numCols];
		int i, j;
		for (i=0; i< numRows; ++ i)
		{
			for(j=0; j< numCols; ++ j)
			{
				A[i][j] = new Molecule();
			}
		}
	}

	public Molecule getMolecule(int i, int j)
	{

		return A[i][j];
	}

	public int calculateH()
	{

		int i,j;
		int antiParallels=0;
int parallels=0;
for(i=0;i<A.length;++i)
{

          for(j=0;j<A[i].length-1;++j)
          {

                    if(A[i][j].spin*A[i][j+1].spin==1)
                    {

                              parallels ++;
                    }

                    else
                    {

                              antiParallels ++;
                    }

          }

}




for(j=0;j<A[0].length;++j)
{

          for(i=0;i<A.length-1;++i)
```

```
                {

                        if(A[i][j].spin*A[i+1][j].spin==1)
                        {

                                parallels ++;
                        }

                        else
                        {

                                antiParallels ++;
                        }

                }

        }

        return antiParallels- parallels;
}



public int noseyNeighbor(int row,int col)
{

        int antiParallels=0;
        int parallels=0;
        int numRows=A.length;
        int numCols=A[0].length;
```

```
//Below
if(A[row][col].spin*A[(row+1)%numRows][col].spin==1)
{
        ++parallels;
}

else
{
        ++antiParallels;
}

//Above
if(A[row][col].spin*A[(row-1+numRows)%numRows][col].spin==1)
{
        ++parallels;
}

else
{
        ++antiParallels;
}

//Right
if(A[row][col].spin*A[row][(col+1)%numCols].spin==1)
{
        ++parallels;
}

else
```

```
                {
                        ++antiParallels;
                }

                //Left
                if(A[row][col].spin*A[row][(col-1+numCols)%numCols].spin==1)
                {
                        ++parallels;
                }

                else
                {
                        ++antiParallels;
                }


                return antiParallels- parallels;
        }
}
```

```
public class Molecule
{
	//member variables

	public int spin;
	public double boltzmanWeight;


	//constructors

	public Molecule()
	{
		spin=1;
		boltzmanWeight=0.0;
	}

	public Molecule(int spin)
	{
		this.spin=spin;
		boltzmanWeight=0.0;

	}
}
```

Significance

The most significant achievement in this competition is how much we learned in the process of completing this project. First of all, we learned how to program in Java and we learned about the Ising Model itself. We learned about the difficulties of programming and the concepts in computational chemistry. As a team, we have achieved a feeling of success, even if we do not win because we put a lot of effort in this rigorous project and just completing the project so far makes us feel very successful. Also, we learned that research and programming in anything never has a real end to it because even now, we are finding ways to make our program more and more precise and finding possible variables that can also affect the result of the project. If we had more time, we would have modified it more and we will try to modify it even after this competition.

Acknowledgements

First of all, we have to thank the New Mexico Supercomputing Challenge providing a chance for us to learn something new and for such a great experience. We thank our sponsor by informing us about the supercomputing challenge and aiding us in our project for the competition. Also, we have to thank the Las Cruces Public Schools and Oñate High School for providing a place to work on our project. The midterm judges helped us a lot on our project by critiquing our project and providing input and ideas for our project.

Bibliography

"Ising Model." Wikipedia Online. 5 Dec.2005.

http://en.wikipedia/wiki/Ising_model

"TheIsing Model."Revised: 24 Dec. 2004. Online Internet. 5 Dec. 2005

www.cecm.sfu.ca/~thalie/Phd/model6.html

"Introduction to the Ising Model."Revised: 6 June 1997. Online Internet. 5 Dec. 2005

www.physics.cornell.edu/sethna/teaching/sss/ising/intro.htm

"Phase Transition." Wikipedia Online. 5 Dec. 2005.

http://en.wikipedia/wiki/Phase_transition

"Phases in the Ising Model. Revised: 6 June 1997. Online Internet. 7 Dec. 2005

http://www.physics.cornell.edu/sethna/teaching/sss/ising/phases.htm