Summer 2010 – NSERC USRA Report

Simulations of Random Walks on Branching Processes

Lisa Zhang

This summer I worked with Dr. Martin Barlow, Dr. Gordon Slade, and Dr. Richard Liang of the UBC Math Department. Random walks have more freedom to move in higher dimensional space than in lower dimensional space. The behaviour of resistance networks in various dimensions is used to model the properties of branching structures related to changing the number of dimensions. These branching structures are formed by using a branching process to determine the number of nodes in the branching structure, and then conducting a random walk along the branching process to determine how the nodes intersect with one another based on collisions in the space coordinates of the random walk. The resistor network is given by assigning a resistance value of one to each edge along the branching structure.

The programming language C++ is used to write code that produces random resistor networks and calculates the effective resistance. This is done by first creating a branching process: starting from a single node, there is a Binomial(2, ½) distribution of the number of nodes in the next generation. That is, there is half a chance for a node to produce each of two offspring. This distribution is then applied to each of the nodes in the new generation, and this continues until the number of generations reaches the predefined number of generations. There is one added constraint that there must be at least one node at each level. Next, with the first node fixed at the origin, each node is assigned a location one step away from its parent. Depending on the number of dimensions of the system, there are more or fewer choices.

This information must be further organized to determine the structure of the network. Collisions between nodes of the same level need to be identified. Where there are collisions, a single new node is created to replace the colliding nodes which inherits all the parents and children of the colliding nodes. This is accomplished by putting all the nodes at each level into a hash table which makes the comparisons in O(1) time (only one comparison is made as opposed to pair-wise comparisons). In addition, multiple bonds between adjacent nodes are counted in the same way as a single bond, so the parents and children of each node are examined to remove duplicate entries.

This completes the structure of the resistor network and the effective resistance can be calculated. A potential difference of one is assigned between the first and last generations, and each parent-child bond is assigned a resistance value of one. The method of Gaussian Relaxation is used, where many trials, involving different branching structures and random walks, are carried out. The standard procedure involves iterating through all the nodes and making the potential at each node equal to the average of the potential of all its neighbours, and then repeating the process again and again until the potentials at all the nodes cease to change. The following equation is used to calculate the new potential of node x and step n+1, where x is the

node in question, y are neighbours of x and N is the total number of neighbours of x:

$$h_{n+1}(x)=\frac{\Sigma_y h_n(y)}{N_x} \quad . \qquad (1)$$

Some modifications of this method are employed to speed up the process. First, drawing from Kirchoff's laws and Ohm's laws, a bundle of resistors only connected to the main network at one point has the potential of that point. Moreover, no current flows through that resistor bundle. Excising these removes a significant portion of the network, which greatly simplifies the Gaussian Relaxation. Secondly, using the altered formula

$$h_{n+1}(x)=h_n(x)\times(1-\lambda)+\lambda\times\frac{\Sigma_y h_n(y)}{N_x} \quad , \qquad (2)$$

larger errors are subject to a large correction and small errors are subject to a smaller correction with a value of λ not equal to one.

We have been creating 10000 trees of dimensions 1, 2, 3, 4, 5, 6, 7, and infinity, and depth 4, 8, 16, 32... (in powers of 2) up to a maximum of 1024 for the infinite dimensional case. Larger branching processes are desired, but we have been constrained by the available memory on the computers, and the long run times of the program. The reason that larger branching processes are desired is because the values calculated by the program does not reflect what is known theoretically. Further analysis on the significance of the values and their standard deviations are being looked at to see if the theoretical value is within the error bounds.