Forest Fires and Criticality

Aims

When a single tree in the middle of a forest catches fire, there is a very high chance that the whole forest would gradually burn down. The propagation of the fire through the forest depends on a variety of conditions such as the forest density and the wind, in addition to some negligible factors such as lightning, rain and trees growing rate. Besides all that, another important phenomenon is that under what conditions the fire would take the most time to burn the forest down. Literally this situation is called the critical point which mainly depends on the forest density.

Using computer programming, the spread of fire under different conditions can be analyzed. Moreover, the critical point can be found by two different methods. One is to find the largest cluster of trees with respect to the corresponding forest size and density and the other is to ignite the forest and record how long it would take the fire to stop under a variety of circumstances. The forest model is commenced by having each site as either a tree or empty. When the forest is ignited, the system will follow two rules. One is that a burning tree becomes empty site after a time step. The other is that a tree becomes a burning tree if at least one of its neighbors is burning [1].

Techniques

In order to simulate the forest fire, first a double array namely "*Grid*" had to be defined to represent the forest grid. Next it had to be ignited after the grid was filled with trees. Finally the spread of fire could be analyzed under the desired conditions.

To fill the grid with trees, each site had to be checked one by one. The task was performed from $i = 0$ to $i = m-1$ and $j = 0$ to $j = n-1$ where $\boldsymbol{m}$ was the grid length and $\boldsymbol{n}$ was the grid width, and $\boldsymbol{i}$ and $\boldsymbol{j}$ represented the position of a site. A random number "$\boldsymbol{a}$" between *0.00* and *1.00* was generated for each site and was compared with the value $\boldsymbol{p}$, the forest density or the chance of finding a tree anywhere in the forest. If "a" was less than or equal to p, the site would be filled with a tree otherwise it was left empty.

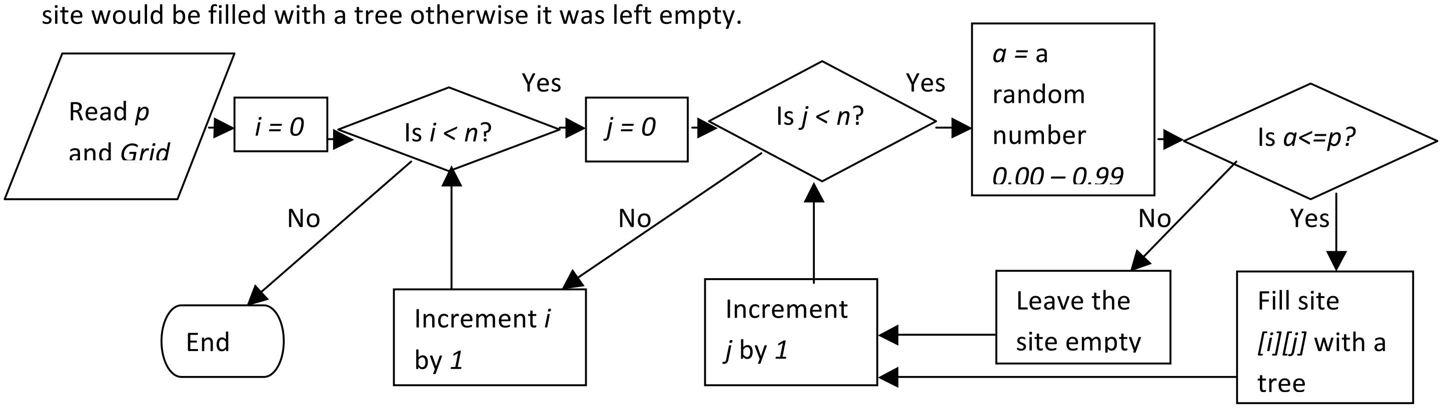

Figure 1.The Fill function which fills the forest (density p) with trees.

When the grid was filled with trees, there needed to be an ignition to start the fire. A random site was chosen and was set to fire.

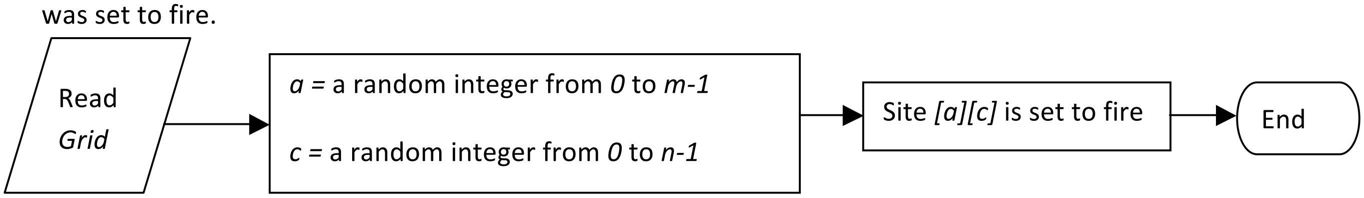

Figure 2.The Ignite function which sets a random site to fire.

The four possible phenomena wind, rain, trees growth and lightning had to be implemented in the simulation as well. Trees growth, lightning and rain followed a similar procedure. All the sites had to be checked one by one. For each site a random number was generated and was compared with the desired parameters ***g*** *(the growth probability),* ***s*** *(the strike probability) or* ***r*** *(the rain probability)* all being in the range *0* to *1*. In the case of trees growth, if $a < g$ and the site was empty, then the empty site would change to a tree site. For lightning If $a < s$ and the site was a tree, the site would become a burning tree and for rain if $a < r$ and the site was a fire then the site would extinguish and become an empty site. These three functions followed exactly the same logic and only the parameters and terms were altered.

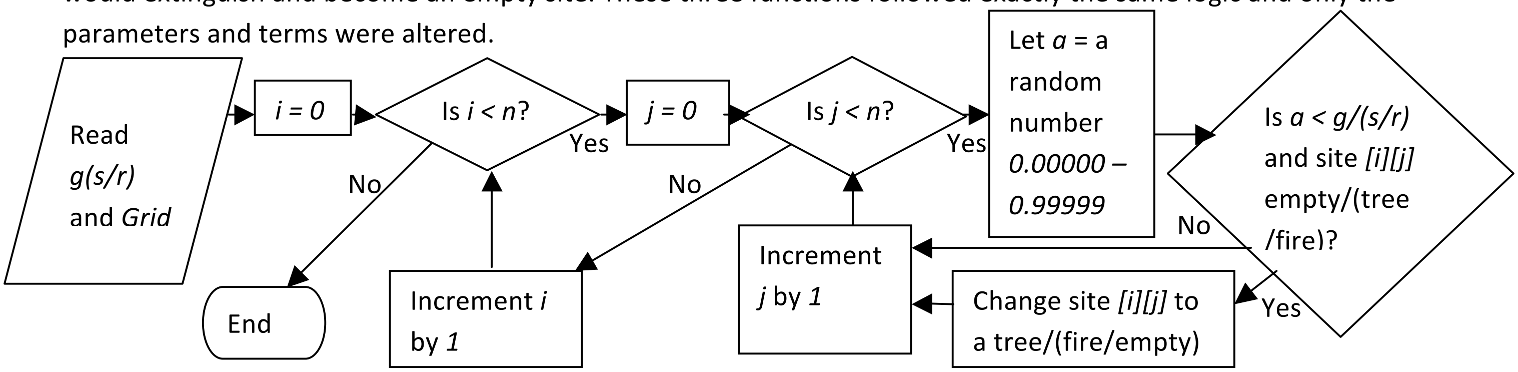

Figure 3.The Grow function which allows empty spaces to grow a tree with a probability of g. Values and parameters in bracket corresponds to that of the Strike and Rain function respectively.

To allow the fire to spread, there had to be new trees catching fire and old trees being extinguished. In order to implement that, two methods were used. One was to define a temporary grid and store values in that or define a new distinguishable item, burnt tree. The former was used for model display and latter used for results and analysis (compiled and ran faster).

When a new time step began, the burnt trees should become empty spaces and burning trees become burnt trees. Then, for every tree, if there was a neighboring burnt tree and no wind ($a < w$), that site would catch fire.

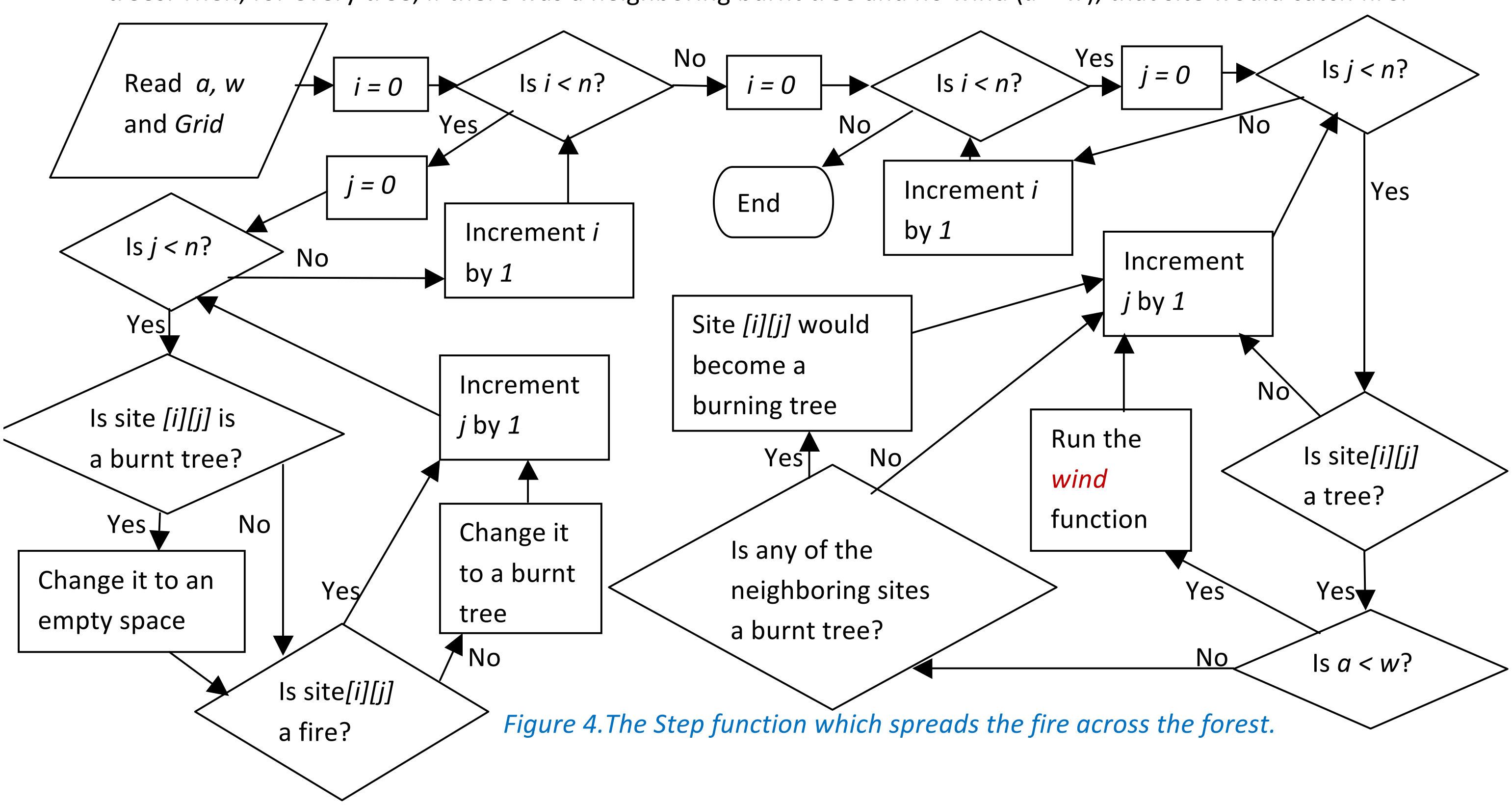

Figure 4.The Step function which spreads the fire across the forest.

Since the wind had a direct effect on the propagation of fire, it had to be implemented within the Step function. The parameters ***a*** and ***w*** were only specified for the wind. The variable *"a"* was a random number from 0 to 1 to be compared with wind probability *w*. Parameters ***dir*** and ***str*** represented the wind direction and strength respectively. When there would be wind blowing in a certain direction, then more trees would catch fire in that direction and fewer trees would catch fire in the opposite direction depending on the wind strength. If for instance the wind strength was 3, then 3 consecutive trees in that direction would catch fire. The trees in the opposite direction of wind would catch fire with a probability of $\boldsymbol{bw} = (6 - str)^3 \times 0.01$.

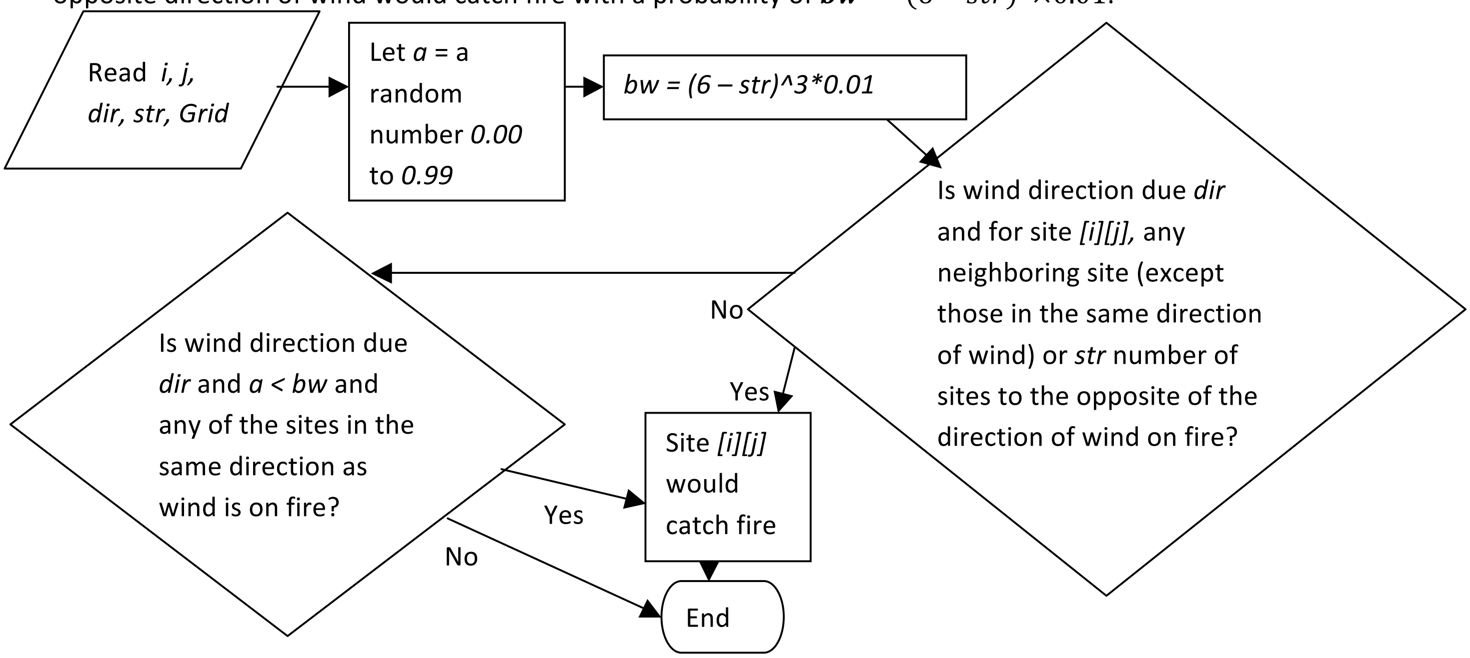

Figure 5.The Wind function which spreads the fire across the forest according to the wind parameters such as direction and strength.

One of the methods to analyze the effect of different circumstances on the forest was to check the number of trees left in the forest after every time step. Every site was checked and if a site was a tree, a counter namely *trees* would be incremented by *1*.

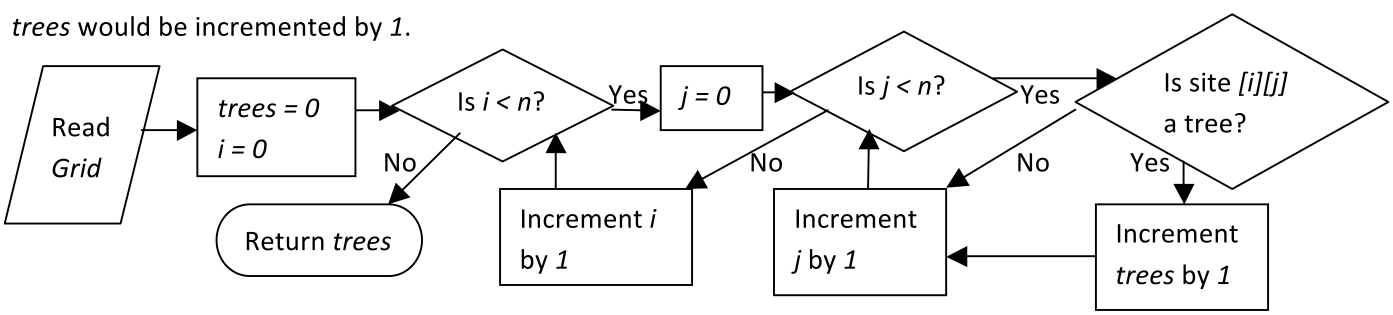

Figure 6.The Treesleft function which counts and returns the number of trees left on the grid at every time step.

Likewise, to print the whole grid, each site was checked one by one and printed. When a row was checked and printed completely, a line was skipped and next row was printed and so on.

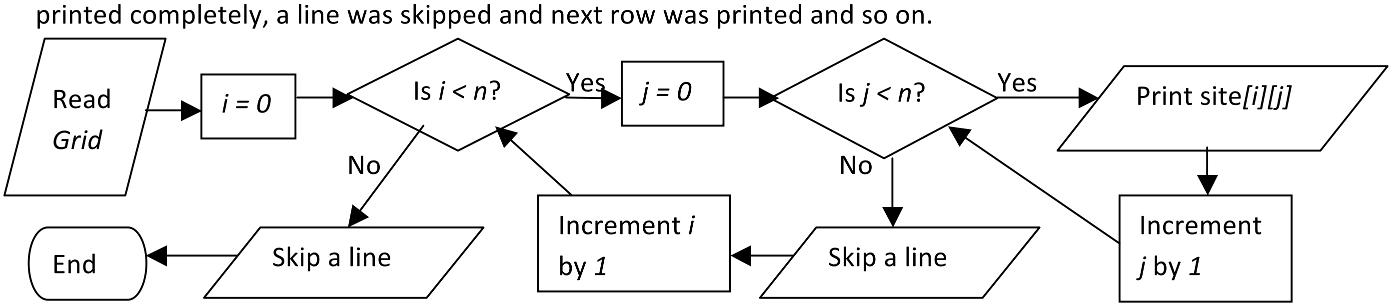

Figure 7.The PrintGrid function which prints the Grid.

When the fire was over, the simulation was supposed to stop. To manipulate such a function, the existence of burning trees was verified at each time step. The *condition* to continue simulation was assumed to be *false* at each time step and once a burning tree was detected, *condition* would change to *true* and simulation continued.

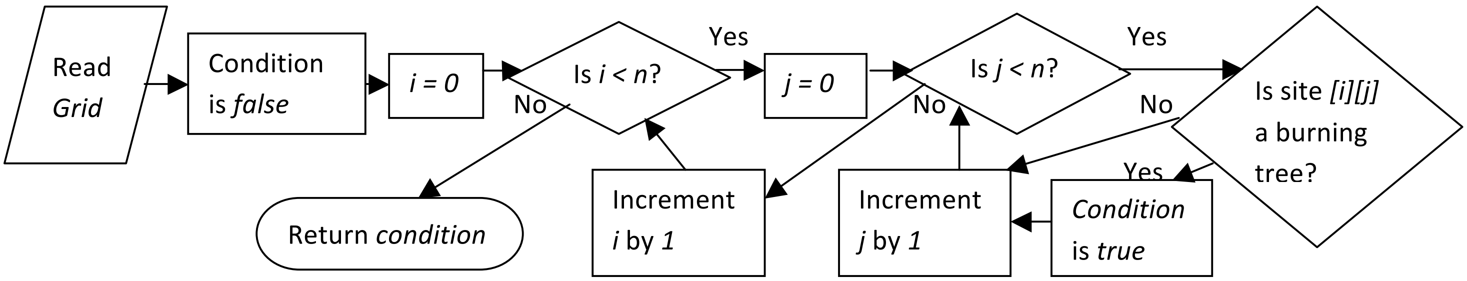

Figure 8.The End function which terminates the simulation when there is no fire on the grid.

Finally all the functions were brought together to perform variety of analysis on the behavior of the fire. The total number of time steps in a run, the number of trees left after each time step and a visual animation of the forest all at user defined conditions could be analyzed. First of all conditions such as rain, strike, growth rate, wind and forest density either at constant values or in a range of values were determined by the user. Next the grid was filled and ignited. Then the simulation would start the when end condition was true. If there was wind blowing, random wind strength, direction and duration were determined. Wind duration determined how long a wind with certain strength and direction would blow. For no wind duration would just be 1. Then there would be strikes, trees growing, raining and fire spreading. At the end, the time step count would increment by 1 and duration would decrease by 1. When duration gets to 0, new random values for wind were determined. Meanwhile, the grid could be printed and trees left could be recorded at each time step and the number of counts could be recorded at the end of simulation.

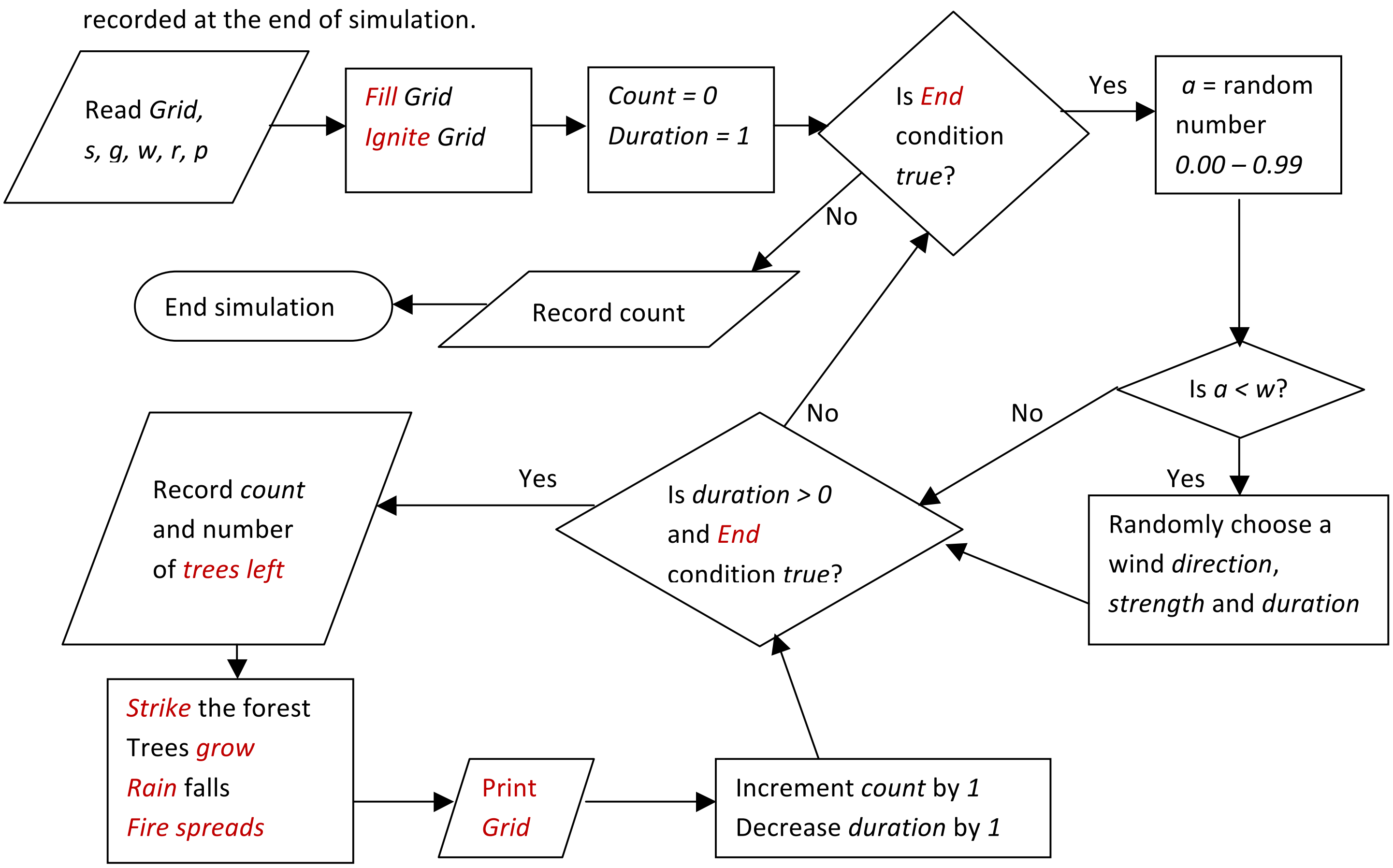

Figure 9.The Main program which brings together all the functions (in red) to run the simulation and record results.

Lastly, to find the size of the largest cluster on the grid, first the forest density ***p*** and maximum grid dimensions ***m*** were defined by user. Then starting the analysis from *1x1* grid and going up to $m \times m$ grid, the largest cluster was found and recorded correspondingly with the grid size ***L***. The first step was to fill the grid with density *p*. Then two variables ***temp*** and ***final*** were defined. *Temp* would store the size of the cluster being analyzed and *final* would store the size of the largest clusters analyzed. Going through each site one by one, if a tree was found, it was checked. The *Check* function would return the size of the cluster the tree was in and the value was stored in *temp*. If no surrounding tree was found, *temp* would simply be *1*. Doing the same procedure for all sites, the largest temp value was stored in *final* and recorded along with the grid size *L*. Then the whole process was done with different grid sizes. In the *Check* function, a site was checked for trees surrounding the site. For every neighboring tree, again recursively the neighboring trees were checked and so on. At each step the number of trees was recorded and sites were marked "checked" to avoid double counting. At the end the *Check* function would return the total number of counted trees in the cluster.

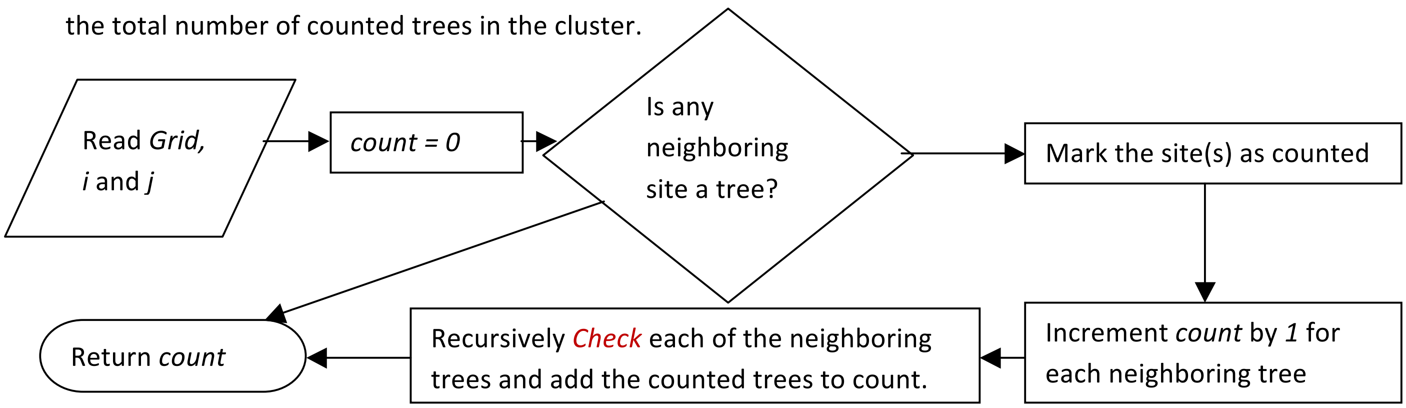

Figure 10.The Check function which checks a certain tree and returns the size of the tree cluster concerned with that tree.

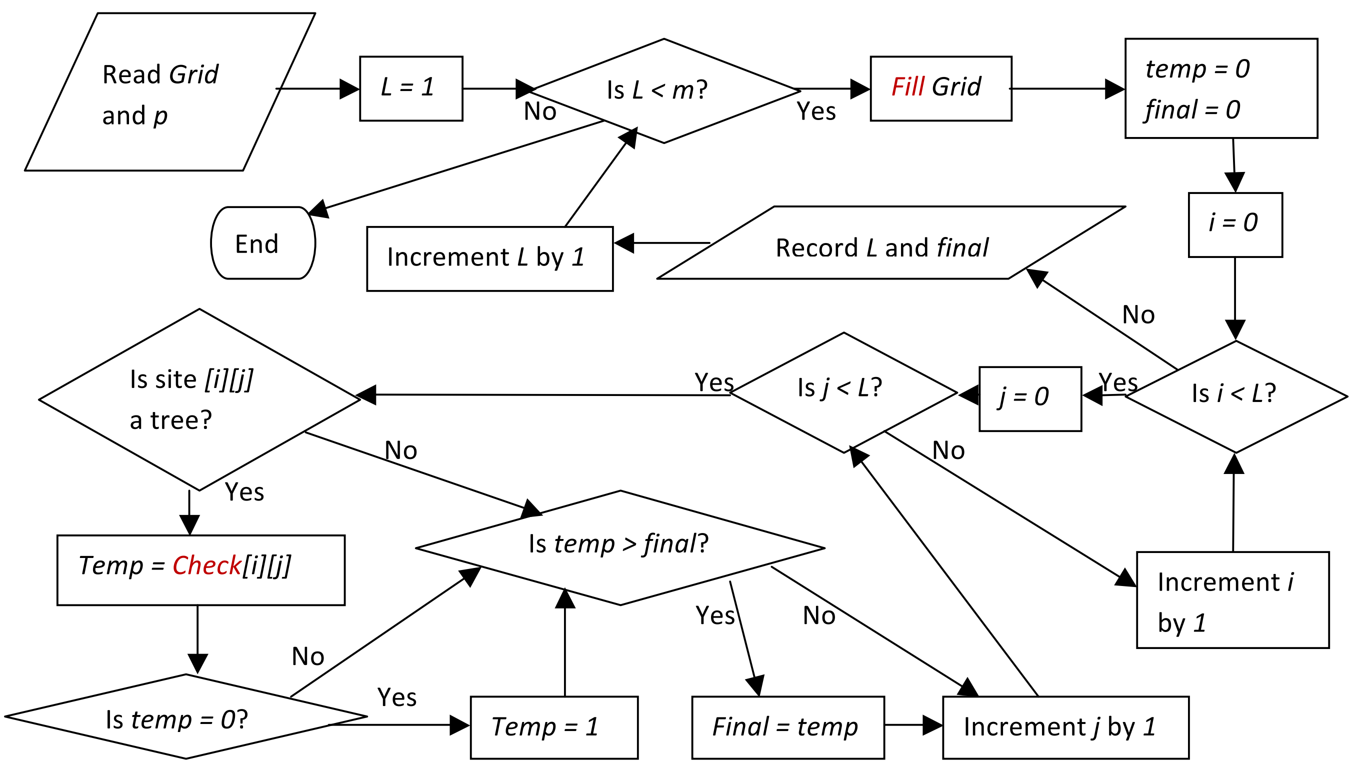

Figure 11.The Cluster program which finds the size of the largest cluster of trees correspondingly with the grid size over a range of different grid sizes from $1{\times}1$ to $m{\times}m$.

Results

The program was used to evaluate the effects of different factors on the forest. Each time, all factors were kept at *0%* and only one factor was kept at a constant value between *0%* and *100%.* To achieve more accurate results, the simulation was done on a 100×100 grid and for each factor, the simulation was run for *5* times and the average of *5* series of data was taken. Since the simulation was based on random numbers, taking the average would ensure a better analysis of the data.

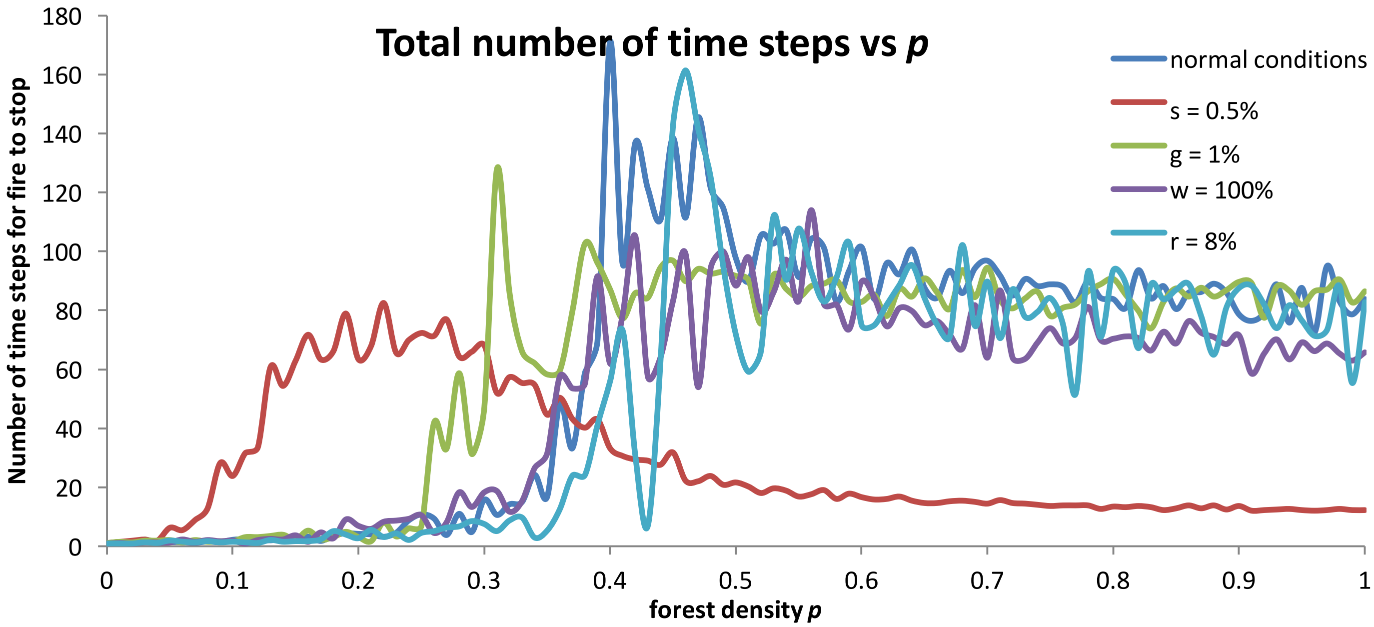

Figure 12. The number of total time steps against the forest density under different conditions 100×100 *grid.*

In figure *(12)* the number of steps for the fire to extinguish completely was recorded for different forest densities *p* and for different conditions. The highest peak corresponded to the critical point. In other words, the critical point was the *forest density p* for which the fire would burn longest. For normal conditions where all factors were kept at *0%* it was found that the critical point was at *p = 40%.* For a high strike rate *s (=0.5%)* it was at *p = 22%.* For a high growth rate *g (= 1%)* the critical point was at *p = 31%.* For windy conditions *(w=100%)* there was no highest peak but the shoot up was at *p =40%.* Finally for rainy conditions *(r = 8%),* the critical point was slightly shifted and was at about *p=46%.*

For a high strike rate, since many spots on the grid would immediately catch fire, the shape of the graph was sensible. For high *p*, the fire would stop incredibly fast because at each time step there were new sites catching fire and spreading, causing the fire to spread all over the grid in a few time steps. For a high growth rate, since there were new trees growing and changing empty sites to trees, then it could be concluded that at each time step, the forest density was increased. Since the forest density would get to *40%* in a few time steps, the critical point had to be lower than that of normal conditions. For windy conditions, the direction and strength of the wind were randomly generated. Therefore, it could be concluded that the fire only spread faster but had no other effects. Hence the only effect was that the fire would stop earlier than that of normal conditions which by observing figure (*12*) the curve corresponding to windy conditions was generally lower than the normal conditions curve. Lastly for rainy conditions, the rain could extinguish some of the fire sites, reducing the forest density slightly after some time steps relatively with that of normal conditions. This caused the critical point to be slightly higher than that of normal conditions since after a few time steps, the forest densities of rainy and normal conditions would equate.

In addition to analyzing the system using numbers, the system was also observed visually by printing the sites after each time step. It helped to realize the effects of different factors by observing real time progression of the system.

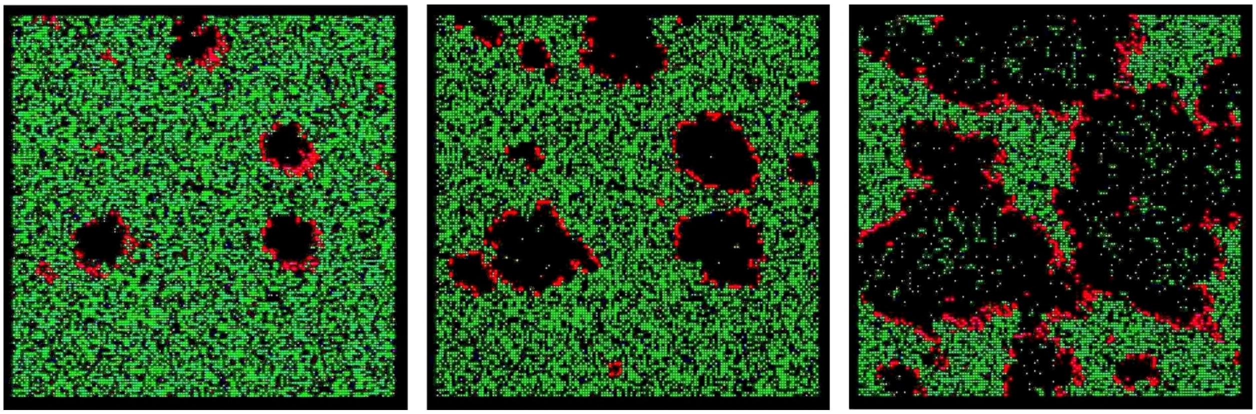

Figure 13. Visual real time model of the forest. P = 45%, w = 15%, r = 1%, g = 0.1%, s = 0.01% and 170 x 170 Grid

Another way to check the effects of different factors on the grid was to analyze how the number of trees declined under different circumstances. The simulation was performed and instead of recording the final number of time steps against forest densities, the number of trees left on the grid was recorded against time step. The simulation was ran on a 100×100 grid and at *p= 40%.* Again, all factors except one were kept at *0%.* The normal condition corresponded to all factors being at *0%.*

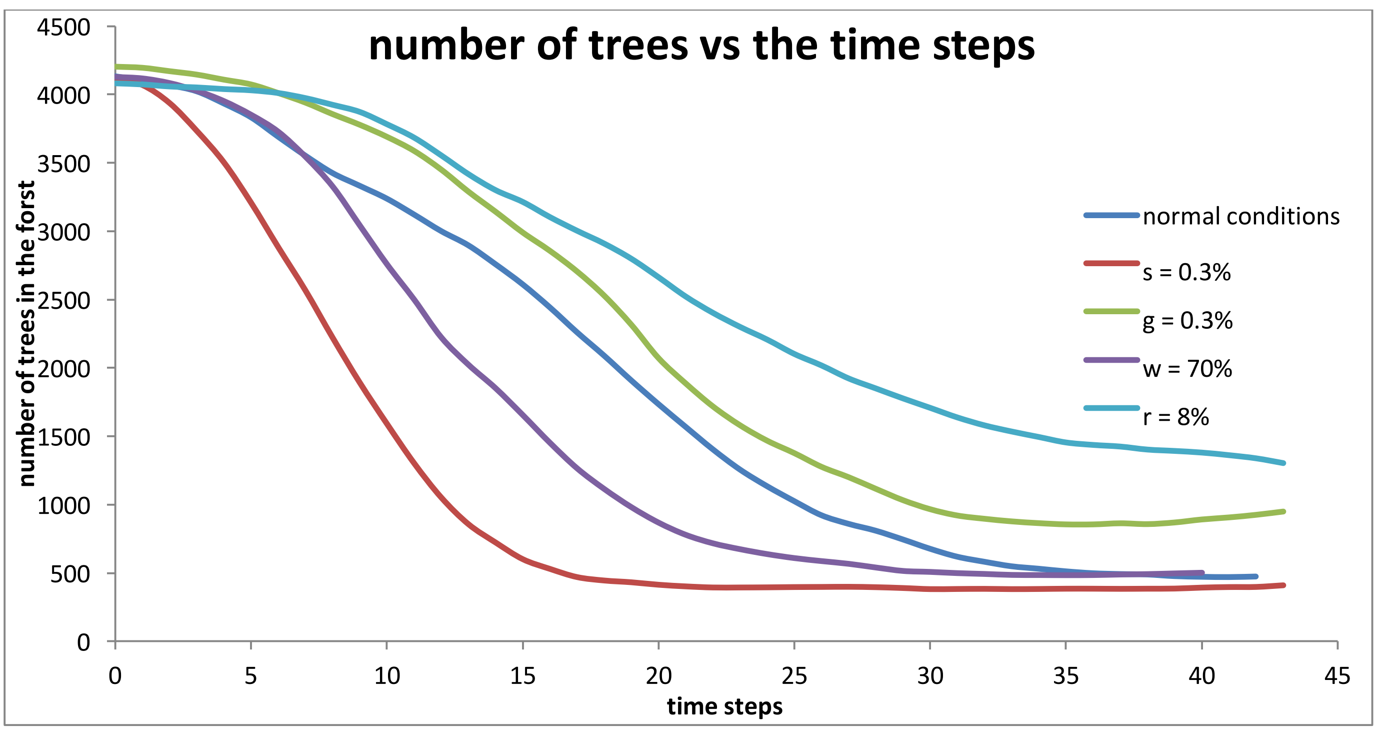

Figure 14.Number of trees on the grid versus the time steps under different conditions. P = 40% and 100 x 100 grid.

For a high strike rate *s (=0.3%)* and windy conditions *(w = 70%),* the graphs were steeper, indicating that the speed of the propagation of fire was higher than that of normal conditions. Moreover, the final number of trees left was slightly lower than that of normal conditions, showing that more trees were burnt under such conditions. On the other hand for a high growth rate *g (= 0.3%)* and rainy conditions *(r = 8%)*, the graphs were less steep and the final number of trees left was higher than that of normal conditions. It was due to the fact that rain extinguished fire and so slowed down its propagation and hence causing fewer trees to catch fire. For a high growth rate, the new trees being grown caused an increase in number of trees on the grid at each time step and causing the graph to be less steep. Also the slope of the graph became positive at a point, indicating that the rate of trees growing was faster than the rate of trees burning after that point.

The last analysis was to find the size of the largest cluster of trees at different forest densities over a range of grid sizes. In general many properties of the system would change drastically as the critical point approaches. For a *2* dimensional system and a grid size of *L*, it was found that

- For $p \ll$ critical point, largest cluster scales with L as $\log L$
- For $p \gg$ critical point, the largest cluster scales with L as L^2
- For $p \approx$ critical point, the largest cluster scales with L as L^D where $D < 2$. D is fractal dimension and it is said that largest cluster is fractal near the percolation threshold [2].

In graph *(13),* the size of the largest cluster was plotted against the grid size *L* at different forest densities. It was found that the drastic change in the shape of the graph occurred at *p = 40%* where the graph of *p = 40%* was the average of *5* sets of data to achieve a good approximation of *D*. the value of *D* was found to be *1.91* which was less than 2 as expected.

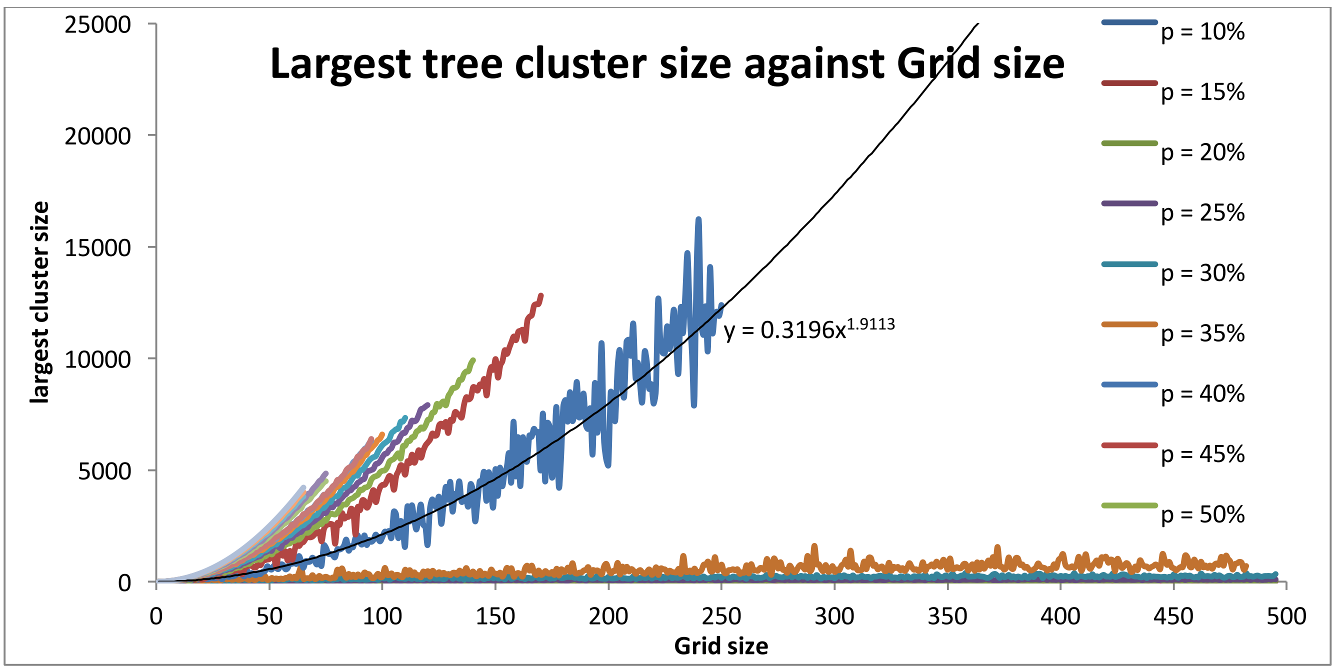

Figure 14.Size of largest cluster of trees against grid size L at different p's.

The shape of the graphs changed from logarithmic to algebraic, and the density where the sudden change occurred corresponded to the critical point. This result did agree with the critical point found earlier for normal conditions.

Bibliography

1. Self-Organized Critical Forest-Fire Model by B.Drossel and F.Schwabl , 14 sept 1992
2. Imperial College London Department of Physics, Introduction to Computational Physics by Dr Peter Haynes page 71, October 2008.