TO: *Alan Sprague (Professor)*

FROM: *Li Dong (BlazerID: ldong)*

DATE: *12/13/2012*

SUBJECT: *CS 455/555 Graduate Project: Forest fire modeling with Monte Carlo Simulation*

INTRODUCTION

Monte Carlo simulation relies on repeated random sampling to compute approximate results when it is infeasible to find an exact solution with a deterministic algorithm. More accurate results can be obtained by simulating a long run with the help of random number generators in computers [1].

In this project, the Monte Carlo Method was used to solve a forest fire spread problem. A forest consists of 1000 trees forming a perfect 50×20 rectangle. The northwestern (top-left) corner tree catches fire. Wind blows from the west, therefore, the probability that any tree catches fire from its burning left neighbor is 0.8. The probability to catch fire from a tree immediately to the right, above, or below is only 0.3. A Monte Carlo study was conducted to estimate the probability that more than 30% of the forest will eventually be burning. Also giving suggestions to the wooden house located in the northeastern corner of the forest.

METHODS

In this part, a detailed explanation about the code is presented.

For the sake of convenience, several variables were declared as global variables, including *N* and *M* which are the number of rows and columns of the rectangular forest, *percentage* which is described in the problem statement as 30%, *T* which is the number of runs of the simulation, *count[]* which tracks the number of burned trees in each simulation, and *danger[][]* which tracks the number of each tree that is burned during the total *T* times simulations.

In order to decide the number of runs of the simulation *T*, it is assumed that the accuracy is α=0.05 and ε=0.005 which means the error of the estimation does not exceed 0.005 with high probability of 95%. Similar to the example 5.14 [1], *T* was calculated as

$$T \geq 0.25 \times (z_{\alpha/2} / \varepsilon)^2 = 0.25 \times (1.96 / 0.005)^2 = 38416$$

In each simulation run, a *N*×*M* boolean array *open[][]* and two queues to track the row and column value of active burning trees in the *N by M* rectangle were generated. For the ease of programming and also to simulate the realistic behavior of the fire spread, an assumption was made here. Every time when a tree catches fire, the tree will burn out and do not spread any more immediately after it makes inquiries to its up, down, left and right neighbors.

Because the upper left corner started fire every time, the first cell of the array *open* was set *true* and *0* and *0* were also put in the two queues. Those information was transferred to the function *flow()* which determines the spread of the fire.

In the *flow()* function, a *for* loop was running until nothing left in the queue which means that the trees were burning out and no more trees would catch fire. The probability of a tree catching fire was one of its four neighbors spreads fire to it, which means $1-P_l \times P_r \times P_u \times P_d$. P_l was the probability that the tree to its left was not on fire, and the same went for P_r, P_u, and P_d. Then a random variable from [0, 1) uniform distribution was generated and compared to the probability of the tree catching fire. The tree would be catching fire if the random variable was less than the probability of the tree catching fire. The fire condition of the four neighbors was checked every time calculating the probability. The cell in the *open[][]* was set to *true* if the tree was determined to be on fire, and the row and column values were put in the queues waiting for the inquiry. At the same time, the global variables *count[]* and *danger[][]* were updating their values.

RESULTS AND DISCUSSIONS

After running the Monte Carlo simulation, the probability that more than 30% of the forest will eventually be burning is about **0.495** (number may change slightly for each run of the program), and the probability of tree at the top right corner catching fire and endangering the house is about **0.345** (number may change slightly for each run of the program). Based on the high probability that the tree at the top right corner catching fire, it is strongly recommended to alert the occupants in the house immediately the fire starts from the top left corner. A fire spreading map (*grid.xls*) was further developed based on this simulation. The probability was calculated for every tree on the 50×20 rectangle. This map reflects the fire spreading process in a realistic manner. For example, because the wind is blowing to the right or say east, this makes the immediate right cell of the first cell is much more prone to catching fire than the immediate down cell, and the bottom left of the rectangle even has probability of catching fire approaching to zero. Figure 1 is an example to show the fire spread in the forest. It is only one of the 38416 runs.

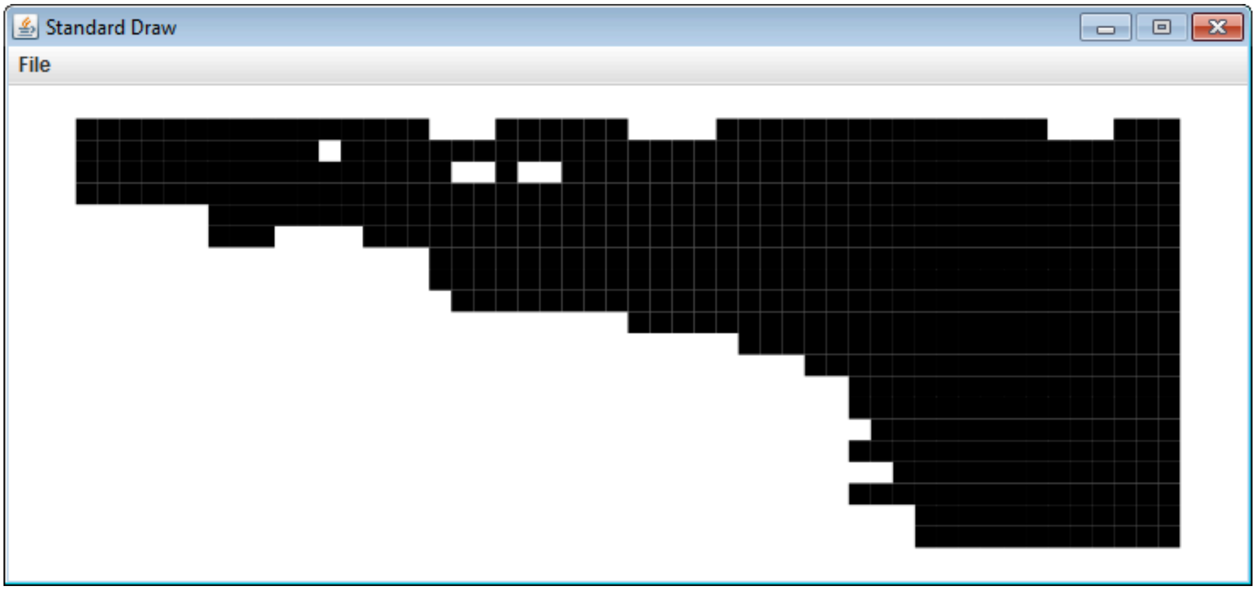

Figure 1 The fire spread for one simulation run

CONCLUSIONS

Through the simulation of a classical percolation problem – the forest fire spread, the Monte Carlo method is proved to be very effective after a long run. The probability value stays almost stable after a certain number of simulation runs.

REFERENCES

1. Baron, Michael. *Probability and statistics for computer scientists.* 1st edition. Florida: Chapman and Hall/CRC. December, 2006.