PHY566 - Percolation

Tianyu Dai, Xiaoqing Li, Yuheng Liao, Xuejian Ma

4/12/2017

Percolation theory describes the behavior of connected cluster in a random graph. It usually concerns the movement and filtering of fluids through porous materials. In this project, we simulate the percolation transition and try to analyze its properties.

1 Determine the critical probability p_c

The basic idea to simulate the percolation transition is to take a $N \times N$ lattice, and then occupy the lattice sites at random according to a certain probability p. If a cluster of occupied sites spans the entire lattice from edge to edge, it's called a spanning cluster. The probability of the appearance of a spanning cluster rises with the rise of p. When p is relatively small, it is highly unlikely to get a spanning cluster. However, for p large enough, the existence of spanning cluster is guaranteed. The transition from one regime to the other is rather sharp and occurs at a critical concentration, which we call p_c. For $N \times N$ lattice, our basic task is to simulate the percolation transitions with different probability p, and figure out the p_c. Doing those for different N, and then plot $p_c - N^{-1}$ graph. From $p_c(N^{-1})$ graph, we can extrapolate to the infinite size limit $p_c(0)$.

1.1 Basic Algorithm

The program details we should follow for $N \times N$ lattice is as below:

1. Begin with an empty $N \times N$ lattice and initialize all sites to zero, i.e. unoccupied.
2. Select and occupy a random site, labeling it with a cluster number.
3. Select another site at random and check neighboring sites to see if any belongs to a cluster created before. If not, the new site should be labeled with another cluster number. If so, we need to discuss among the following two situations:
   (a) If there's only one cluster number among all neighboring sites, give this number to the new site.

(b) If there are several cluster number among all occupied neighbors, the new site is considered as a site bridge which connects multiple clusters into one. We should choose a common unique cluster number for resulting cluster, and then also give this number to the new site and relabel all sites in merged cluster to that number.

4. Repeat step 3 until a common cluster number appears on all edges of the lattice.
5. The common cluster number is the number of the spanning cluster, and the fraction of all occupied sites among all lattice sites is p_c

1.2 Revised Algorithm

To simplify the model and save computation time, we revised the original algorithm.
The original $N \times N$ lattice is expanded to $(N+1) \times (N+1)$ first. Each site on the boundary is labeled a unique number, as shown in Fig. (1.2)

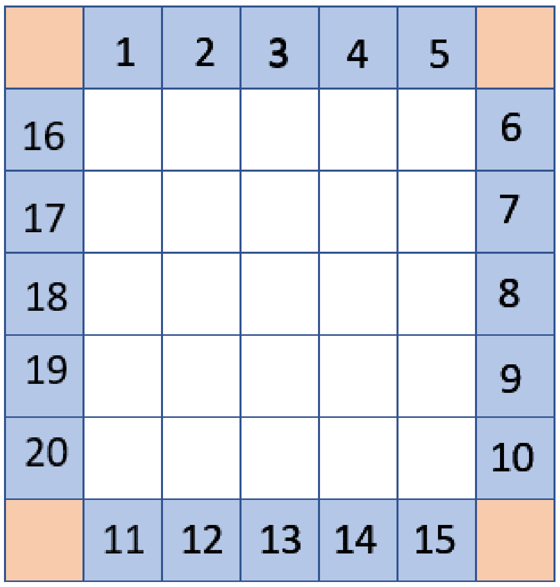


Figure 1: Boundary Setting

Then, follow below steps:

1. Select and occupy a random site from the lattice and label it as $4N+1+n$, which can distinguish it from the boundary. n is the step number.
2. Store the cluster number of four nearest sites.
3. Transverse all the occupied sites(including boundary sites). Relabel all sites which has the same cluster number as nearest sites to be the newest cluster number $4N+1+n$. The lattice before transverse and after transverse is shown in Fig. (2).

4. If the maximum label on each boundary is the same as the newest cluster number, the spanning cluster shows up. The appearance of spanning cluster in the lattice and the spanning cluster is Fig. (3).

5. Repeat steps from the beginning.

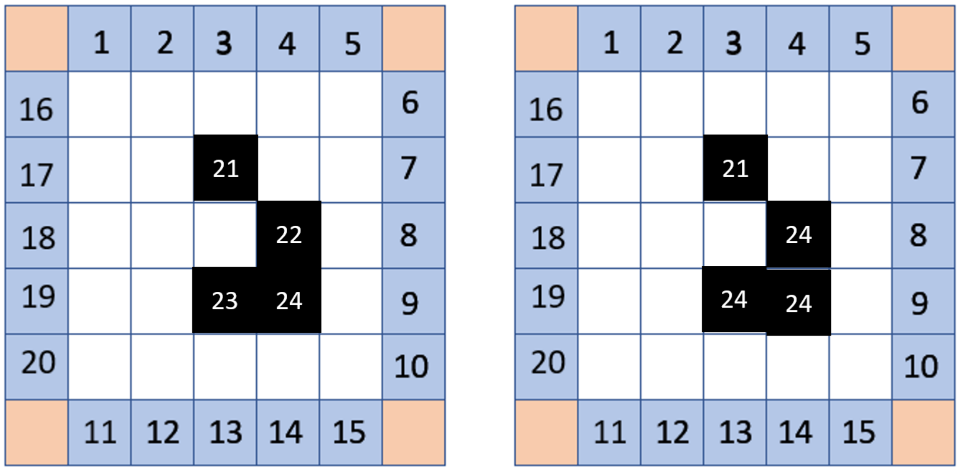


Figure 2: Cluster labels before and after the transverse

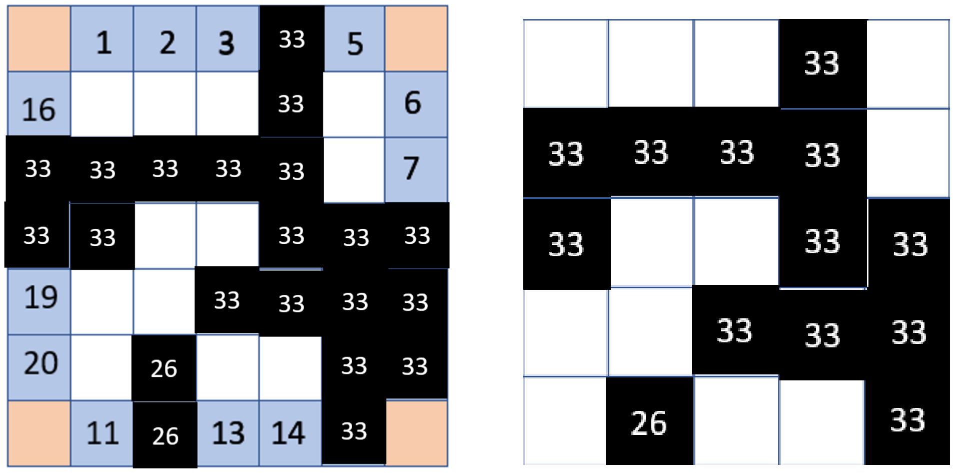


Figure 3: The spanning cluster

The revised algorithm avoid dealing with boundary conditions. When the newest added site reaches the boundary, the lattice before and after transverse is shown in Fig. (4).
The judge of how many nearest sites are occupied is also avoided. After each transverse, all cluster connected to the newest added site will be updated in each step.
Following the steps above, we can get the spanning clusters for different $N \in \{5, 10, 15, 20, 30, 50, 80\}$ as below:

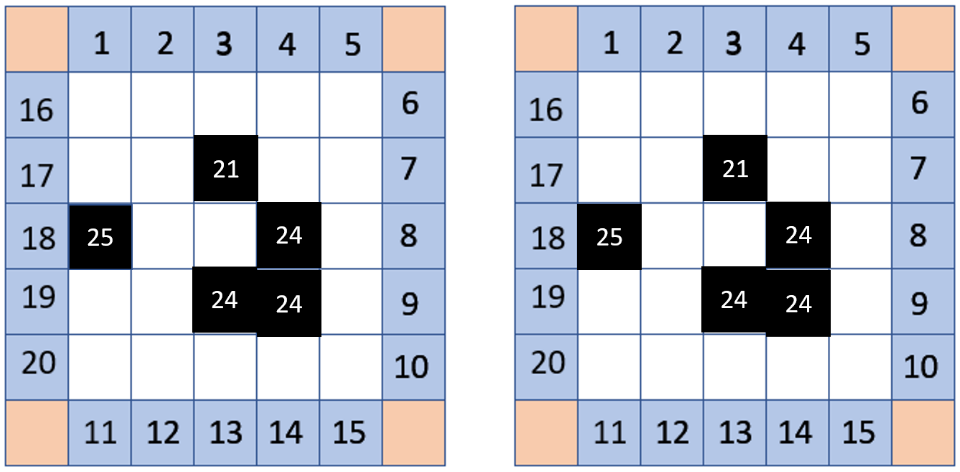
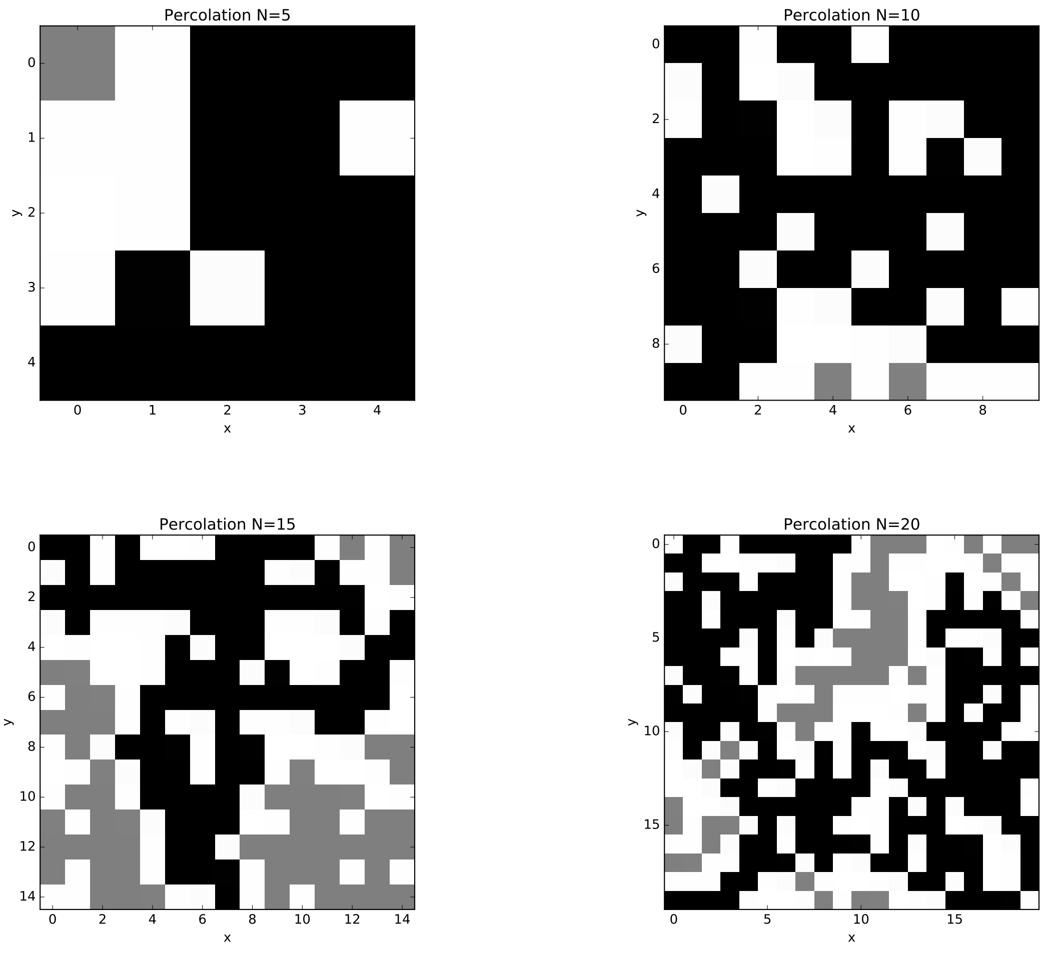


Figure 4: Newest added site reaches the boundary, before and after the transverse



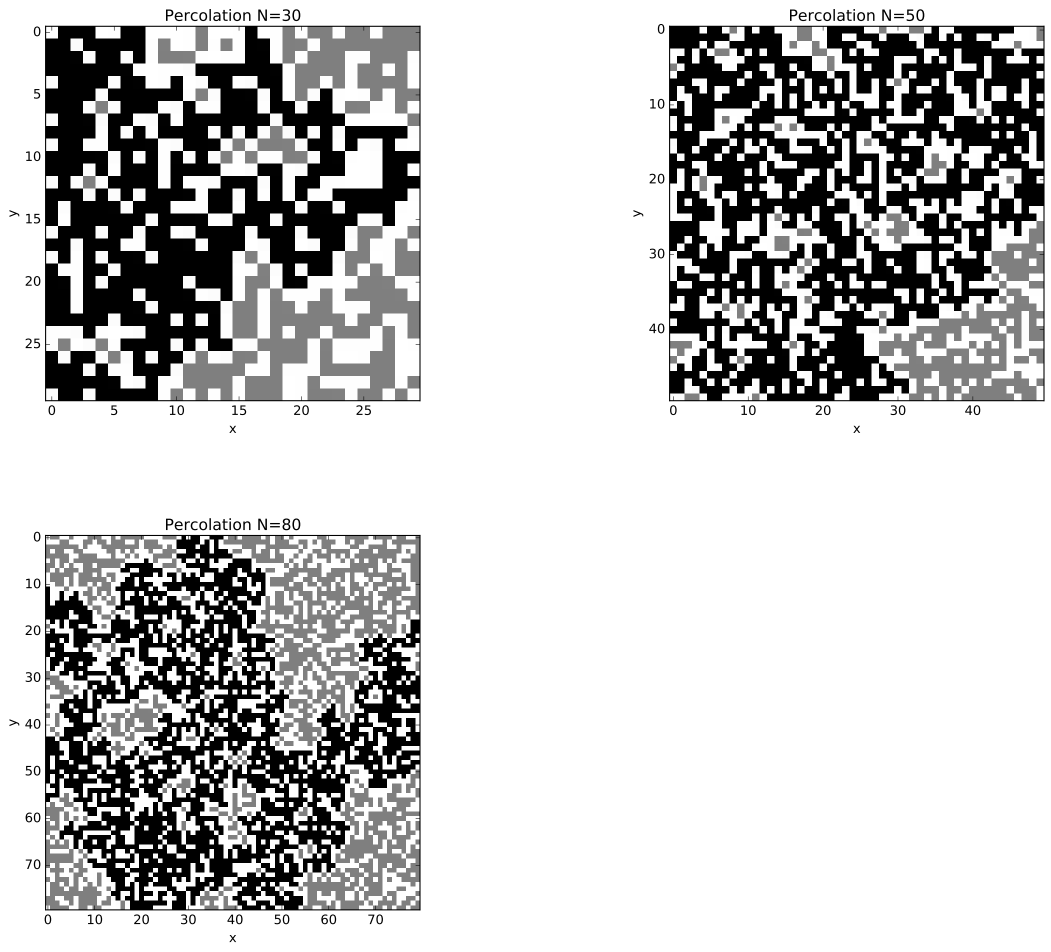


Figure 5: First Appearance of Spanning Clusters for Different Lattices

In the pictures above, black sites represent sites of spanning clusters, gray ones represent sites occupied while not belonging to spanning clusters, and white ones represent sites unoccupied.

By repeating the procedures over 50 times for each N, we can calculate the average $p_c(N^{-1})$. After doing so for all $N \in \{5, 10, 15, 20, 30, 50, 80\}$, we could plot the graph of $p_c(N^{-1})$ as Fig. (6):

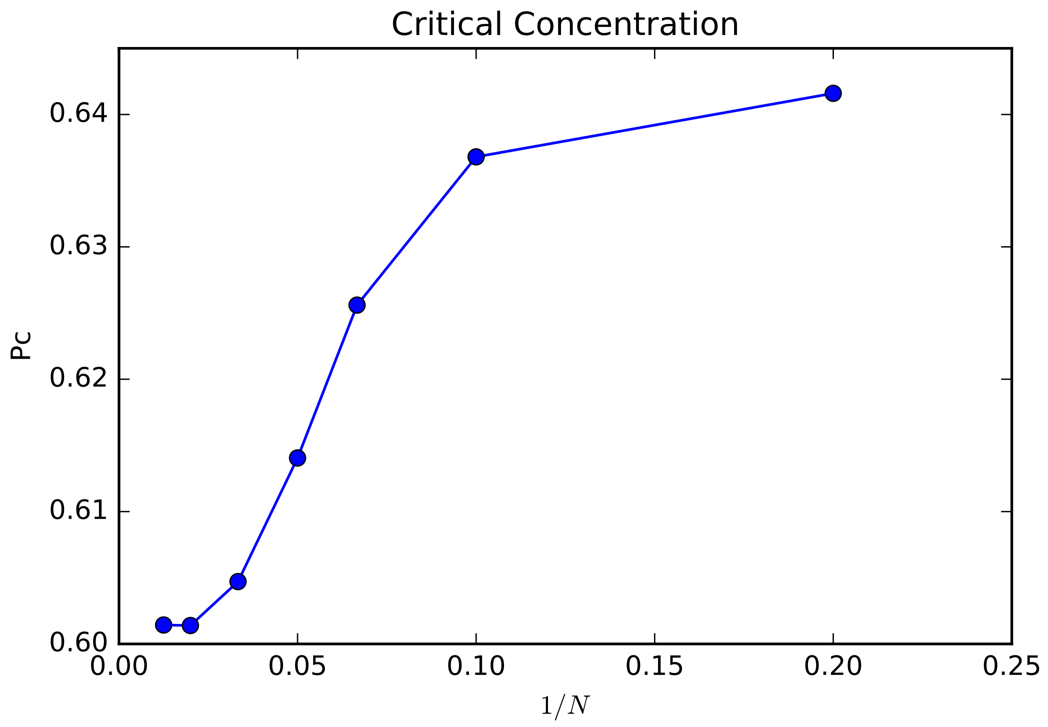


Figure 6: Critical Concentration

According to the graph, we could extrapolate to the infinite size limit $p_c(0) \approx 0.602$.

2 Spanning Cluster Size for Different Lattice

We define the fraction F of a as

$$F\left(p > p_c\right) = \frac{no.\,of\,sites\,in\,spanning\,cluster}{no.\,of\,occupied\,sites}$$

The procedure of finding the fraction for a certain p is similar to the procedure of finding the critical probability p_c. However, the condition to break the loop is not the appearance of the spanning cluster, but the number of occupied sites over the number of all the sites is p. When the spanning cluster first appears, we store the cluster number of the spanning cluster. After the iteration, calculate the number of spanning cluster. The occupied number is calculated by $p \times N \times N$.
The whole procedure is repeated for 50 times and then calculate average fraction F. In a lattice with fixed size (100×100), we plot the fraction F for different probability p in Fig. (2). The power law ansatz is

$$F = F_0\left(p - p_c\right)^{\beta}$$

By plotting the logarithm for each side and extracting the slope of a straight-line fit, the index β can be calculated as in Fig. (2). The calculated β is 0.1305, which is close to the theoretical value $\frac{5}{36}$.

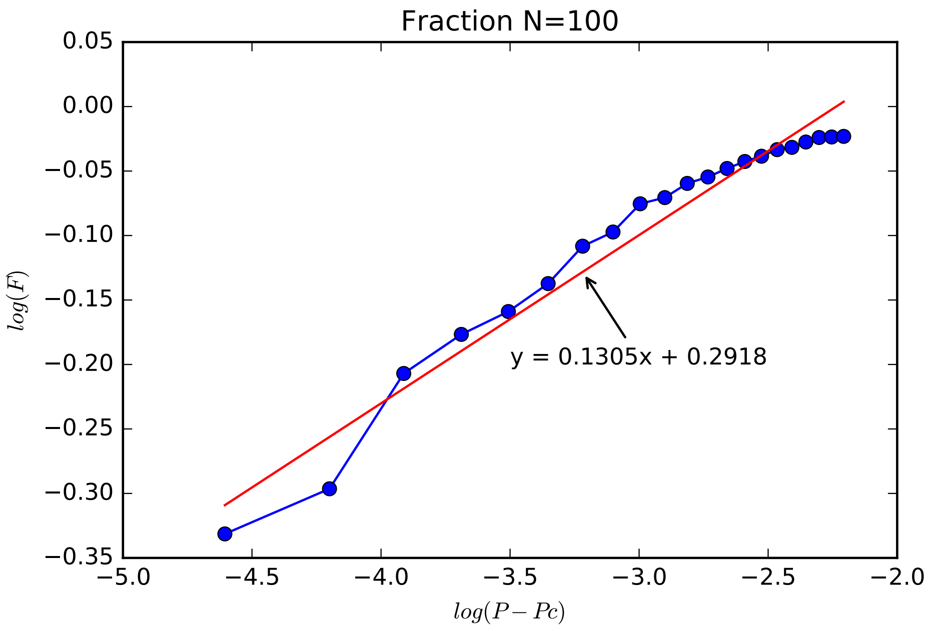


Figure 7: Fraction Plot and Linear Fitting