Introduction to Modeling and Simulation, Spring 2002, MIT

1.992, 2.993, 3.04, 10.94, 18.996, 22.091

Lecture 3: Percolation

Excerpts from the following article will be handed out during Lecture 5: "Percolation" by A. Bunde and S. Havlin, in *Fractals and Disorderd Systems* (Springer 1996, second edition). Some images from this article were shown as transparencies during the lecture, and it provides a good introduction to the subject. The lecture, outlined below, emphasized a few basic points.

1. **What is percolation?** — Imagine flipping $N = L^2$ fair coins, arranging them in a square array, and then studying connected "clusters" of adjacent coins of the same type, e.g. heads. Interesting questions about this very simple statistical system include, "what is the expected size of the largest cluster?" and "what is the probability of finding a 'spanning cluster' which connects opposite boundaries?".

   These questions get much more interesting in the "thermodynamic limit" of infinite system size $N \to \infty$, especially when one considers unfair coins with a probability $0 < p < 1$ of showing heads ("occupied sites"). The parameter p is equal to the average fraction of occupied sites, and thus acts like a concentration[1] In mathematics and physics, the term "percolation" refers to the game of probability just described, i.e. indepedently occupying sites or bonds on a lattice with a probabiilty p and then studying the statistical properties of clusters of adjacent occupied sites. Remarkably, this simple model actually has something to do with the more familiar meaning of "percolation" as the flow of a fluid (e.g. coffee) through a disordered medium.

2. **The Percolation Phase Transition.** — As illustrated in various computer demonstrations during lecture (soon to be available on the web), there is a curious "phase transition" in cluster connectivity as a function of p in the thermodynamic limit: Below a certain critical value $p < p_c$, there is no infinite cluster, while above this value, $p > p_c$, there is one infinite cluster (with probability one). Exactly at the critical point $p = p_c$, there can be multiple infinite clusters, and they are fractal objects (see lecture 4). These clusters are very large in spatial extent, but take up hardly any space, as it were, just barely spanning the system.

   As shown clearly in the comptuer demonstrations, away from the critical point, there is a typical length scale for large clusters, called the correlation length $\xi(p)$, e.g. which can

[1] The first problem on problem set #2 is to estimate computationally whether an infinite spanning cluster has a finite probability of occuring in the example of fair coins, $p = 1/2$, on a square lattice.

be defined as the expected distance between two points on the same cluster. If you color the clusters, a given percolation system looks like an army camoflauge with a fixed length scale for the cluster shapes. This correlation length can be defined on both sides of the phase transition by excluding the infinite cluster (or excluding the largest cluster in a finite system). The correlation length grows as $p \to p_c$, and at the critical point it becomes infinite, $\xi(p_c) = \infty$.

3. **Applications.** — Percolation was first proposed by Flory in the 1940s as a model for polymer gelation, e.g. the boiling of an egg, the vulcanization of rubber, the sol-gel transition, or the making of cheese from milk. In these examples, the concentration of protein monomers is increased by the removal of solvent until purely by statistical effects, monomers start to connect and form long chains which "percolate" across the system. When percolation occurs, the clear liquid in an egg turns white, as clusters grow to the scale of the optical wavelengths (500nm) and start reflecting visible light.

   The model in the general form described above and the term "percolation" itself were invented by Broadbent and Hammersley in the 1950s in the context of modeling flow in porous media. This application continues today, as percolation is used to model the pathways of connected pores in a porous rock, e.g. from which secondary oil may be extracted.

   Another modern application is in amorphous computing, being developed at MIT in the AI lab. The idea is to endow a vast number of tiny micro-processors with the ability to communicate via radio waves, and embed these processors in a gel matrix, effectively creating a "smart paint". After being painted on a surface (and being scattered at random), the processors undergo a discovery phase where they find their neighbors and self-assemble into a giant parallel computer. The real challenge is to understand how to program such a computer. However, percolation is also an important aspect of the problem, because the concentration of processors must be chosen large enough to ensure significant interconnectivity, but low enough to avoid jamming of radio frequencies and inefficiency.

   Although percolation does not provide a complete description in most applications, it nevertheless provides a very useful universal paradigm to understand connectivity in disordered systems.

4. **Leath Algorithm.** — Clusters can be grown dynamically from a point (the origin) via the Leath algorithm. Initially, all sites on the lattice aside from the initial growth site are considered "virgin" sites. Then, recursively, the neighbors of the neighbors of the neighbors... of the initial site are labeled "occupied" with probability p and otherwise "unoccupied". This process continues until the cluster stops growing (due to the formation

of a boundary of unoccupied sites) or until an outer boundary of the system (e.g. a large circle) is reached. The latter method is useful to quickly determine spanning probabilities. If one wants the mass of clusters, it is best to instead view the outer boundary as a hard wall, which does not stop the growth altogether, but simply prohibits it from proceeding outside the boundary.

To implement the Leath algorithm, use an array `cluster[i,j]` to represent the lattice. This array has a value of -1 for a virgin site, 0 for an unoccipied site, and 1 for an occupied site. A "depth first" strategy to grow clusters then is based on the following recursively called subroutine (without checking for spanning):

```
void grow(int i0, int j0)
{
  for each virgin neighbor [i,j] of [i0,j0] inside the boundary {
    if(random_number < p) {
      cluster[i,j] = 1;
      grow[i,j];
    }
    else {
      cluster[i,j] = 0;
    }
  }
}
```

The main program starts the cluster growth from site [I,J] by simply calling `grow[I,J]`. Here, `random_number` is a random real number between 0 and 1.

The Leath algorithm also correponds to a basic model of epidemic spreading. The lattice represents the social network of a population (people are sites, and friendships are indicated by bonds). The initial growth site is the first infected individual. The epidemic grows as friends of infected people become infected with probability p, or remain resistant to infection or out of contact with probability $1 - p$. If $p < p_c$ the epidemic will enventually die out (although perhaps not before all the people are gone in a finite population!). Obviously, $p \geq p_c$ (or more precisely $\xi(p) \gg L$) is very bad...