With a maximum extent of 94 lattice constants, one obtains the estimate $D = \ln 3150 / \ln 94 \simeq 1.77$ for the fractal dimension.

Additionally the simulations show that, thanks to the introduction of the long jumps for $R > R_d$, the processing time does not increase significantly if the radius of the annihilation circle is enlarged.

Exercises

Improved dimensional analysis. In determining the fractal dimension we have, so far, neglected the fact that the cluster being generated is not complete yet. While the "arms" of the cluster continue to form, additional particles can still reach its interior regions. By arbitrarily stopping the simulation as soon as a certain maximum radius is reached, we neglect those particles. The actual cluster should be slightly denser, the fractal dimension slightly higher.

Calculate the mass of the cluster neglecting all particles whose distance from the origin is greater than r. Plot the mass as a function of r in a log–log diagram and attempt to interpret the behavior.

Varying the model parameters. It is interesting to observe in which way the form and fractal dimension of the cluster change if the original DLA model is varied slightly.

Introduce an additional parameter p_{stick} into the program `dla.c`, which represents the probability that a particle which arrives at a site adjacent to the cluster will actually stick to it.

If the particle is not attached, it shall continue the diffusion process until it hits the cluster again, at which point it will again have a probability p_{stick} of getting stuck. In continuing the diffusion process one has to make sure, of course, that the particle is prohibited from jumping to a site that is already occupied.

Literature

Gould H., Tobochnik J. (1996) An Introduction to Computer Simulation Methods: Applications to Physical Systems. Addison-Wesley, Reading, MA

Sander E., Sander L.M., Ziff R.M. (1994) Fractals and Fractal Correlations. Computers in Physics 8:420

5.3 Percolation

In an ideal crystal, all atoms are arranged on a lattice in a regular way. In this case we can use Fourier transformations to describe the periodic structures as

well as the crystal excitations, e.g., the phonons. Since nature rarely consists of ideal crystals, however, solid state physicists have been trying for decades to understand materials with irregular structures as well.

There is a model for disordered materials, which permits a rather simple description: the *percolation model.* This model is intended to describe a porous material through which certain particles can diffuse (or *percolate*) if there are continuous paths through the pores. The geometry of the randomly generated pores is described by percolation theory. The model can be explained to any beginning student, it is easily programmed on a computer, yet it contains interesting physics related to phase transitions, critical phenomena, universality, and self-similar fractals.

In this section, we want to present and simulate the percolation model.

Physics

Let us assume that we have an alloy consisting of two kinds of atoms, which we label by A and B respectively. The concentration of A atoms is denoted by p, and both kinds of particles are assumed to be distributed randomly and to form a regular crystal lattice together. Let us further assume that the A atoms are magnetic, with a short-range magnetic interaction which is only effective if two A atoms are located next to one another. If there is no coupling with the nonmagnetic B atoms, the crystal can only form a magnetic order if there is an infinite network of connected A atoms. While, strictly speaking, phase transitions can only occur in infinite systems with a sufficient degree of interconnectedness, we will be able to see clear indications of a phase transition at a critical concentration of A atoms even for finite systems. The size of the structures of A atoms which are all connected with each other plays an important role in this process. Such a set of A atoms that are connected by their couplings is called a *cluster.*

For small concentrations p, all we get is a large number of small clusters but no overall magnetic ordering. As p increases, the average size of the clusters of A atoms will increase as well until, starting at a threshold value p_c, a cluster extends through the entire crystal. The value p_c is called the percolation threshold or critical concentration. For $p > p_c$, the infinite cluster coexists with many small ones; not until $p = 1$ are all sites occupied by A atoms and there is only one (infinite) cluster.

For the infinite crystal there is a single, well-defined value for the percolation threshold p_c. For example on the square lattice, with four nearest neighbors, one finds the numerical value $p_c = 0.59275 \pm 0.00003$; on the triangular lattice, with six nearest neighbors, $p_c = 1/2$ is known exactly. While p_c depends on the lattice type and on the range of the couplings, it does not depend on the specific realization of the sample, if the size of the crystal approaches infinity. In the finite crystal, on the other hand, there is of course only a certain probability, depending on p, that a cluster connecting two opposite sides exists.

Now one can ask many questions concerning the properties of the clusters and work out mathematical relations. What is the value of p_c for different lattice structures and cluster definitions? How does the average size of the finite clusters increase as p is increased up to the threshold p_c? How does the density of the infinite cluster increase above p_c? What does the distribution of cluster sizes look like? Does the infinite cluster have a structure at p_c?

The governing laws in the vicinity of the percolation threshold p_c are particularly interesting. For example, we define an average size $R(s)$ of a cluster consisting of s particles by

$$R^2(s) = \frac{1}{s(s-1)} \sum_{i \neq j} (\boldsymbol{r}_i - \boldsymbol{r}_j)^2 \ , \tag{5.17}$$

where i and j number the particles of the cluster and $\boldsymbol{r}_i$ is the position of the ith atom on the lattice. Then the average size ξ of the finite clusters is defined as

$$\xi = \sqrt{\langle R^2(s) \rangle_{s<\infty}} \ . \tag{5.18}$$

Here, $\langle \ldots \rangle_{s<\infty}$ designates the average over all finite clusters.

At the percolation threshold ξ diverges, and close to p_c we have

$$\xi \sim |p - p_c|^{-\nu} \ . \tag{5.19}$$

One obtains this law for all percolation models. Surprisingly, the exponent ν is universal. This means that its value depends only on the spatial dimension. In the planar case, the value $\nu = 4/3$ is known exactly for different lattice types and ranges of the couplings, and even for the corresponding problem without a lattice, and for real two-dimensional alloys. For three dimensions, one numerically obtains $\nu \simeq 0.88$.

The probability $P(p)$ that an arbitrarily chosen A atom is part of the infinite cluster is zero for $p < p_c$ and increases continuously from $P(p_c) = 0$ to $P(1) = 1$. Close to p_c the rise is again described by a power law with a universal exponent β:

$$P(p) \sim (p - p_c)^{\beta} \qquad (p \gtrsim p_c) \ . \tag{5.20}$$

In two dimensions we have $\beta = 5/36 \simeq 0.14$; this means that $P(p)$ rises very steeply near p_c.

While the density of the infinite cluster vanishes right at the percolation threshold, $P(p_c) \cdot p_c = 0$, it does exist there and has an interesting structure: it is a fractal. If we consider a square section of the lattice with sides of length L and designate the number of particles in this section that belong to the infinite cluster by $M(L)$, then at p_c we obtain

$$\langle M(L) \rangle \propto L^D \ , \tag{5.21}$$

where $\langle \ldots \rangle$ designates the average over different sections. Thus the mass of the cluster increases as a power of the length, and according to Sect. 3.3 D

is its fractal dimension. In two dimensions, $D = 91/48 \simeq 1.89$. This shows that the critical percolation cluster is significantly more compact than the aggregate from the previous section, which was generated by diffusion.

So far, we have learned about three universal critical exponents ν, β, and D, but near p_c many more properties are described by power laws, with additional exponents. Singularities like these, however, are not independent of each other, but are connected by scaling laws. We only want to outline the idea here; for a deeper understanding, we refer the reader to textbooks about critical phase transitions and the theory of renormalization groups.

Let $M(p, L)$ be the average number of particles in a cluster that traverses a square with sides of length L. For each value of L, one obtains the number M as a function of p; this results in a whole set of curves. Scaling theory then tells us that close to the critical point this set of curves can, after suitable scaling, be represented by a single universal function f. The scales for the quantities M and L can be expressed by powers of either $p - p_c$ or the variable ξ defined in (5.19). In doing so, L is measured in units of ξ, and M in units of a power of ξ, e.g., ξ^x. This means

$$M(p, L) \sim \xi^x f\left(\frac{L}{\xi}\right) , \tag{5.22}$$

where f is a function which is not known initially and x is a critical exponent. If we set $L = k\xi$, with a constant k, then $x = D$ follows from (5.21). Thus, L is measured in units of ξ and M in units of ξ^D, hence the name "scaling theory."

Further, from $L = k\xi$ it follows that

$$\frac{M(p, L)}{L^d} \sim \xi^{D-d} f(k) , \tag{5.23}$$

where L^d is the number of lattice sites in the d-dimensional cube with edges of length L. M/L^d, though, is the probability that a lattice site belongs to the percolation cluster. Therefore, for large values of L we get

$$\frac{M(p, L)}{L^d} \simeq pP(p) \sim (p - p_c)^\beta . \tag{5.24}$$

Equations (5.19), (5.23), and (5.24) lead to

$$(p - p_c)^\beta \sim (p - p_c)^{-\nu(D-d)} . \tag{5.25}$$

From this, one can conclude

$$\beta = (d - D)\nu . \tag{5.26}$$

This means that the three critical exponents are coupled by a scaling law. Indeed, the knowledge of two exponents is sufficient for calculating all others.

To begin with, the critical exponents are defined only for infinite lattices. With the computer, though, we can only populate finite lattices with particles. In finite systems, however, there is no phase transition, no sharply

defined percolation threshold, and there are no divergences. How can one deduce the universal critical laws of the infinite lattice from the properties of a finite system?

The theory of *finite size scaling* answers this question. It is based on the scaling laws outlined above which, among other things, make statements about the dependence of the singularities on the lattice size L. Using the behavior of the correlation length described by (5.19) in the scaling relation (5.22) one obtains

$$M(p, L) \cdot |p - p_c|^{\nu D} \sim \tilde{f}\left((p - p_c) L^{1/\nu}\right) , \tag{5.27}$$

where $f(x) = \tilde{f}(x^{1/\nu})$. Consequently, the concentration p and the system size L are related to one another close to the critical point $p = p_c$, $L = \infty$.

The fractal dimension D is already obtainable from (5.21), since for $L \ll \xi$ the finite system looks just like the infinite one. Therefore, (5.21) yields $M(p_c, L) \sim L^D$. Thus one can numerically calculate the fractal dimension D from the increase of the number $M(p_c, L)$ of particles in the percolation cluster as a function of L. But how does one determine p_c and the exponent ν?

To do this, we establish a scaling law analogous to (5.27) for the probability $\pi(p, L)$ of finding a cluster connecting two opposite faces, which therefore percolates, in a sample of size L. Obviously,

$$\pi(p, \infty) = \begin{cases} 0 & \text{for } p < p_c , \\ 1 & \text{for } p > p_c . \end{cases} \tag{5.28}$$

For finite values of L, this step function is rounded off in the vicinity of p_c. Since π is constant for $p > p_c$ in the infinite system, the value of the corresponding scaling exponent is zero. Therefore, analogous to (5.27) for $M(p, L)$, the scaling equation for $\pi(p, L)$ is

$$\pi(p, L) = g\left((p - p_c) L^{1/\nu}\right) \tag{5.29}$$

with an unknown function g. If we now populate the sites of a lattice with atoms with a probability p, and then increase p, all clusters will grow. At a threshold value $p_c(L)$, a percolating cluster will appear.

The probability for $p_c(L)$ to lie in the interval $[p, p + \mathrm{d}p]$ is given by the value of $(\mathrm{d}\pi/\mathrm{d}p)\,\mathrm{d}p$ at the point p. The derivative $\mathrm{d}\pi/\mathrm{d}p$ exhibits a maximum that diverges for $L \to \infty$ and moves towards the percolation threshold p_c of the infinite lattice. Because of the large statistical fluctuations, the maximum of $\mathrm{d}\pi/\mathrm{d}p$ is not well determined numerically. It is better to determine the mean value and the fluctuations of $p_c(L)$. The mean value of the percolation threshold is given by

$$\langle p_c(L) \rangle = \int p \frac{\mathrm{d}\pi}{\mathrm{d}p} \mathrm{d}p . \tag{5.30}$$

From (5.29), one obtains for this the scaling law

$$\langle p_c(L)\rangle = \int p L^{1/\nu} g'\left((p-p_c)L^{1/\nu}\right) \mathrm{d}p \,. \tag{5.31}$$

With the substitution $z = (p-p_c)L^{1/\nu}$ and with $\int(\mathrm{d}\pi/\mathrm{d}p)\mathrm{d}p = \pi(1,L) - \pi(0,L) = 1$, this leads to

$$\langle p_c(L)\rangle - p_c \sim L^{-1/\nu} \,. \tag{5.32}$$

If we determine the value $p_c(L)$ for as many systems of size L as possible, and repeat this for many values of L, which should be as large as possible, we can fit the mean value $\langle p_c(L)\rangle$ according to (5.32) and thereby determine p_c and ν. At the same time, we can calculate the mean value of the square of $p_c(L)$ and obtain from this a measure Δ of the width of $\mathrm{d}\pi/\mathrm{d}p$. For if we perform calculations analogous to (5.30) and (5.31) for

$$\begin{aligned}\Delta^2 &= \left\langle \left(p_c(L) - \langle p_c(L)\rangle\right)^2 \right\rangle = \left\langle p_c(L)^2 \right\rangle - \langle p_c(L)\rangle^2 \\ &= \left\langle \left(p_c(L) - p_c\right)^2 \right\rangle - \langle p_c(L) - p_c\rangle^2 \,,\end{aligned} \tag{5.33}$$

we obtain

$$\Delta \sim L^{-1/\nu} \,. \tag{5.34}$$

This means that the parameter ν can be obtained directly from the standard deviation of $p_c(L)$

Thus, by using the scaling laws, we are able to calculate the critical values for the infinite lattice from the properties of finite systems. This *finite size scaling* is not just limited to the percolation problem, but is valid for other kinds of phase transitions as well, e.g., for the magnetic transition of the Ising ferromagnet, a topic with which we will deal in Sect. 5.5. With this method, we have learned about an important tool for calculating universal, or model-independent, quantities.

Finally, we want to mention that, in addition to the *site percolation* discussed here, there is also the so-called *bond percolation*. In this case, not the *sites* of a lattice are populated with a probability p, but the *bonds* between adjacent sites. Such systems have a different percolation threshold, but they have the same universal critical properties as the model discussed.

Algorithm

On a lattice, the percolation structure is easily generated numerically. We choose a square lattice with sites (i,j), $i = 0,\ldots,L-1$ and $j = 0,\ldots,L-1$, and use uniformly distributed random numbers $r \in [0,1]$. With this, the algorithm becomes:

1. Loop through all (i,j).
2. Draw a random number r.
3. If $r < p$, plot a point at the position (i,j).

The corresponding C program is:

```
double p = 0.59275;
int i, j, L = 500, pr;
pr = p*RAND_MAX;
for (i = 0; i < L; i++)
for (j = 0; j < L; j++)
    if (rand() < pr) putpixel(i,j,WHITE);
```

Figure 5.6 shows the result. One can see a pattern without any apparent structure. It is therefore surprising that one can extract so many mathematical regularities by quantitatively answering suitable questions. The analysis of the percolation structure, however, is not as easy to program as its generation. For example, we cannot easily answer the question of whether the structure in Fig. 5.6 percolates, i.e., whether there is a path connecting two opposite faces that only uses occupied sites. Even our visual sense is overtasked by this question.

We need an algorithm that can identify clusters of connected particles. A naive method would be to draw circles of increasing size around each occupied site and to mark all connected sites. This, however, takes a very large amount of processing time. A fast algorithm was developed in 1976 by Hoshen and Kopelman. It loops through the lattice only once and assigns cluster numbers to all occupied sites. Occasionally, though, parts of the same cluster are assigned different numbers. By using a bookkeeping trick, such conflicts are easily resolved at the end. The details of this algorithm are well described in the textbooks by Stauffer/Aharony and Gould/Tobochnik.

Here we want to pursue a different path, which was also suggested in 1976 by Leath. Clusters are to be generated directly by a growth process. To do

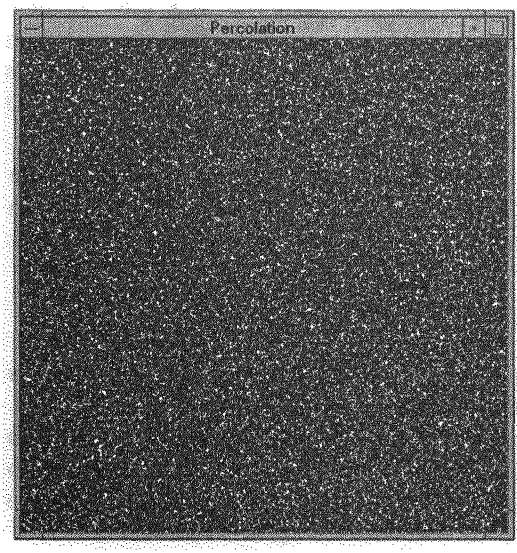

Fig. 5.6. Percolation structure on a square lattice at the percolation threshold

this, we start by occupying just the center of an empty lattice and then we define all adjacent sites as either occupied or free, with the probabilities p and $1-p$ respectively. This process is iterated, converting for each occupied site the adjacent undefined sites to either free or occupied ones. When no undefined sites adjacent to the cluster are left, the growth stops; if the resulting cluster connects two opposite sides of the lattice, it is a percolation cluster. Many repetitions of this process yield a distribution of clusters that can be analyzed using the methods of finite size scaling. Since each particle generated gets the statistical weight p, and each free site gets the weight $(1-p)$, and since the entire cluster is weighted with the product of all these factors, the growth process generates the clusters with the same probability as the previous algorithm.

Thus the growth algorithm for a cluster is:

1. Label all sites of a square lattice with sides of length L as "undefined" and occupy the center. Initially, therefore, the cluster consists of just one particle.
2. Pick an undefined site adjacent to the cluster.
3. Draw a uniformly distributed random number $r \in [0,1]$ and occupy the site if $r < p$. Otherwise, label the site as "vacant."
4. Iterate 2 and 3 until no undefined sites adjacent to the cluster are left.

To make sure that the growth process is terminated in a well-defined manner at the edge of the lattice, we label the sites around the lattice boundary as "vacant." For item 2, we will generate a list, from which we can obtain the positions of the remaining undefined sites next to the cluster. Of course, this list has to be updated constantly during the growth process.

Since we want to simulate the largest possible lattices again and directly observe the growing cluster on the screen, we have written the algorithm in C. First, we label the three different cases with the numbers 0, 1, and 2:

```
#define NOTDEFINED 0
#define OCCUPIED 1
#define VACANT 2
```

For the lattice, which only needs to take these three values, we use a data type that uses little space in memory, namely `char array[L][L]`; additionally, we define a list that will hold the positions `(i,j)` of the undefined sites adjacent to the cluster via

```
struct {int x; int y;} list[PD];
```

This list, which initially contains only the four sites next to the center, is processed sequentially and receives new entries while this is taking place. Since the processed sites are no longer needed, this list can be written to and looped through periodically. Therefore, its length `PD` does not have to be equal to L^2; in our tests the choice $\texttt{PD} = 4L$ proved to be sufficient. The initialization is done as follows:

```
for(i=0;i<L;i++)
   array[0][i]=array[i][0]=array[i][L-1]=
                array[L-1][i]=VACANT;
for(i=1;i<L-1;i++)
for(j=1;j<L-1;j++)
   array[i][j]=NOTDEFINED;
array[L/2][L/2]=OCCUPIED;
list[0].x=L/2+1; list[0].y=L/2;
list[1].x=L/2  ; list[1].y=L/2+1;
list[2].x=L/2-1; list[2].y=L/2;
list[3].x=L/2  ; list[3].y=L/2-1;
count=3;
putpixel(L/2, L/2, WHITE);
```

Here, count is a pure counting variable, whose value is incremented by 1 each time a new undefined boundary site of the cluster is generated. The main loop, item 4 above, is:

```
while(! done )
     {
      event();
      for(k=0; k<=count; k++)
        {
         i=list[k%PD].x; j=list[k%PD].y;
         definition(i,j);
        } /* for */
     } /* while */
```

Thus, the program loops through L and processes the list of still-undefined cluster boundary sites. The function definition(i,j) starts by checking whether the site (i,j) is already defined. This is necessary because the same boundary site may appear in the list more than once. If the site is not yet defined, step 3 is executed, and the site (i,j) is occupied if the random number r is smaller than the concentration p. If this newly occupied site has any undefined neighbors, their positions are entered in the list above, and the value of the variable count is increased accordingly.

In the program, this looks as follows:

```
void definition(int i, int j)
 {
  double r;
  if( array [i][j] != NOTDEFINED ) return;
  r = rand() / (double) RAND_MAX;
  if( r < p )
   {
    array[i][j] = OCCUPIED;
    putpixel (i, j, WHITE);
    if( array[i][j+1]==NOTDEFINED )
     {count++;list[count%PD].x=i;list[count%PD].y=j+1;}
    if( array[i][j-1]==NOTDEFINED )
     {count++;list[count%PD].x=i;list[count%PD].y=j-1;}
    if( array[i+1][j]==NOTDEFINED )
```

```
  {count++;list[count%PD].x=i+1;list[count%PD].y=j;}
 if( array[i-1][j]==NOTDEFINED )
  {count++;list[count%PD].x=i-1;list[count%PD].y=j;}
   }
 else array[i][j] = VACANT;
}
```

The program still needs to be supplied with the declarations, the graphics initialization, and the input handling routine `event()`.

Results

To begin, we use the first algorithm for the generation of the entire percolation structure, and populate a square lattice of 500×500 sites with a concentration $p = 0.59275$. The occupied sites (A atoms) are marked in black, the lattice and the vacant sites are not shown. Figure 5.6 shows the result. One can only see noise, which does not give any indication of the mathematical laws that can be obtained from a quantitative analysis of this picture. We have selected the concentration p right at the percolation threshold, but neither a percolation cluster nor critical properties are recognizable. Only a suitable computer program can extract the clusters from this structure and quantitatively analyze them.

Figure 5.7 shows a cluster that was generated by the growth algorithm. The concentration p and the lattice size are the same as those in Fig. 5.6. Now one sees a percolating cluster, which exhibits structures on all length scales, from the lattice constant up to the size of the entire lattice. It is a fractal self-similar object. Of course, the program not only generates percolating clusters;

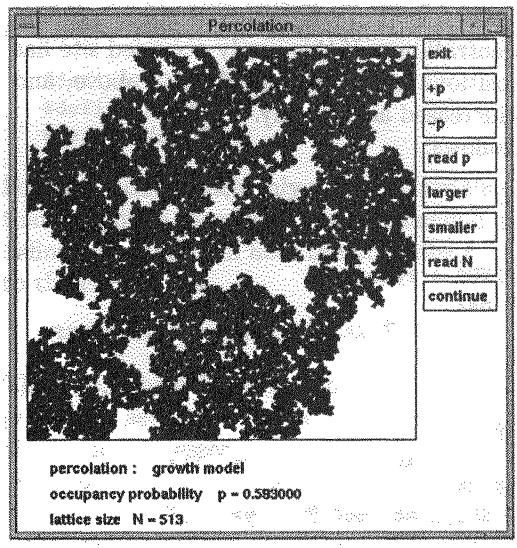

Fig. 5.7. Percolating cluster at the percolation threshold

very often the growth process stops before the cluster has reached the edge. For $p < p_c$ this occurs very frequently, whereas for $p > p_c$ percolating clusters are generated almost exclusively.

Exercise

Let s be the number of particles in a percolating cluster. In the infinitely large system, the average cluster mass $\langle s \rangle$ of the *finite* clusters diverges near the percolation threshold according to the power law

$$\langle s \rangle \sim |p - p_c|^{-\gamma} \ .$$

Calculate the mean value $\langle s \rangle$ for all numerically generated clusters that do not touch the edge. Plot these values as a function of the concentration p, and attempt to use the result to determine the critical concentration p_c. Calculate the maximum of this function for different values of the lattice size L. Try to calculate the critical exponent γ with the help of *finite size scaling*. In doing this, you may use the value of ν given above.

Literature

Gould H., Tobochnik J. (1996) An Introduction to Computer Simulation Methods: Applications to Physical Systems. Addison-Wesley, Reading, MA

Stauffer D., Aharony A. (1994) Introduction to Percolation Theory. Taylor & Francis, London, Bristol, PA

5.4 Polymer Chains

Polymers play an important role in chemistry as well as in biology and medicine. But physics, too, has long been interested in general mathematical laws concerning the properties of polymers. Even a single molecule which is made up of a long chain of identical units (*monomers*) has interesting properties. In order to be able to mathematically describe such chains of many thousands of monomers, one must attempt to model the essential properties and structures as simply as possible. One proven model for this is the randomly generated path, which is called the *random walk* in scientific literature. The path consists of many short line segments that are joined together in random directions.

In Sect. 3.3 we have already established that a random walk can be regarded as a model of a polymer molecule, all of whose configurations are equally probable in a heat reservoir. In that case, the mass of the chain increases as the square of its mean size. The mathematical theory for this is very well developed. This model neglects an important mechanism, however: