Modelling and Simulation
Practical Assignment 2: Percolation

Rick van Veen (s1883933)* Laura Baakman (s1869140)*

October 11, 2015

1. Introduction

Beter inleiding, misschien beginnen met bacterie in petri schaaltje

Kenzel et al. [1] give the following interpretation of the percolation model; it describes the geometry of the randomly generated pores in a porous material through which only certain particles can percolate if the pores form continuous paths. We model this material using a finite lattice, although different lattices are possible, we consider only a square lattice.

The exact percolation model is describes in section 2, this section also presents an implementation of the model in pseudo code. In section 3 we discuss some of the experiments we have performed with the model and their results. Section 4 presents a summary of the findings of our experiment.

Algorithm 1: `percolation`$(mask, N, p)$

```
input  : $N$ size
         $p$ probability
         $mask$ $r \times c$ binary matrix.
output : $grid$ $(N+1) \times (N+1)$ matrix
1  $center := (N+1, N+1)$
2  push($queue$, $center$)
3  $grid :=$ initGrid($N$, $N$)
4  while not isEmpty($queue$) do
5      $site =$ pop($queue$)
6      $sites =$ grow($grid$, $site$, $mask$, $p$)
7      if $onBorder(site)$ then
8          break
9      push($queue$, $sites$)
10
```

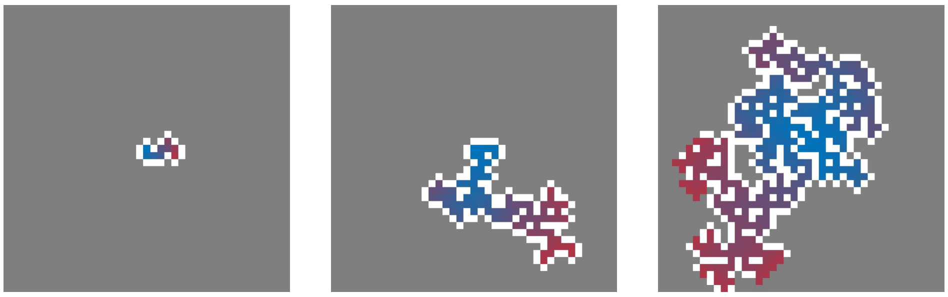
2. Method

Algorithm 1 presents our iterative growth process, the method `percolation` expects three arguments `N`, `probability` and `mask`. Given the size parameter `N`, the grid used for the percolation is $(N+1) \times (N+1)$, since this causes the grid to have an uneven number of rows and columns its center is always clearly defined as (N, N). The parameter $p \in [0, 1]$ is the probability that a given site in the cluster becomes occupied. The `mask` is a binary matrix with r rows and c columns that determines the used connectivity, until section 3.4 we only consider four-connected clusters, which use the mask presented in figure 6a.

Initially the only site we have to consider is the center site, which is consequently the only site in the queue at the first iteration.

Each iteration we pop the next `site` from the queue. We grow this point, using the function `grow`. This method considers all neighbors that are connected to `site` according to `mask`.

*These authors contributed equally to this work.



(a) *$N = 20$ and $p = 0.3$* **(b)** *$N = 20$ and $p = 0.5$* **(c)** *$N = 20$ and $p = 0.6$*

Figure 1: *Examples of (a) a small finite cluster, (b) a larger finite cluster and (c) a percolating cluster using four-connected neighbours. The colours of the elements in the cluster indicate when that point was added to the cluster, the 'colder' the color the earlier in the percolation it was added to the cluster. White cells are empty and gray cells are undetermined.*

For each of these neighbors we determine the value z, which is randomly sampled from an uniform distribution with the range $[0, 1]$. If $z \leq p$ we mark the neighbor site as occupied, otherwise it is marked empty. The method `grow` returns the neighbor sites that are occupied, these are added to the queue, as these sites are should also be allowed to grow.

The growth of the clusters stops if it cannot grow anymore or if it has reached one of the borders of the grid. In the first case the cluster is finite, which means that all neighbor sites of the cluster, according to the connectivity defined by the `mask`, are marked as empty. In algorithm 1 we test for this condition via the guard of the loop; if the queue is empty there are no more neighbors to consider, consequently the cluster must be finite.

A percolating cluster is a cluster that has reached the border of the grid, i.e. if there is a occupied site with row or column number 1 or $2N + 1$. We test for this condition with the method `onBorder`. It should be noted that we only check if a site is on the border of the grid after we have already grown the site.

Figure 1 presents three clusters grown with the algorithm. We can clearly see that the finite clusters are completely surround by a white border, which indicates that these sites are empty. The illustration of the percolation cluster, figure 1c, shows that although there are still sites that can grow indicate by the lack of white neighbors, there is one site on the border near the bottom left corner of the image that has stopped the percolation.

3. Experiments

This section presents our exploration of the parameter space of the percolation model. In section 3.1 we discuss the influence of the probability parameter p. Section 3.2 explores the effect of the size of the system on the generated clusters. In section 3.3 we attempt to determine the fractal dimension of the finite clusters as a function of p. Finally, section 3.4 presents a short analysis of the impact of the used connectivity.

3.1. Probability

One important property of clusters is their size, and how that size depends on the parameter p. Kenzel et al. [1] describe this relation as follows: for small values of p we get a large number of small clusters. As p increases we find positive correlation between p and the average cluster size until p reaches some threshold value p_c. For $p > p_c$ we get either a small finite cluster or a percolating

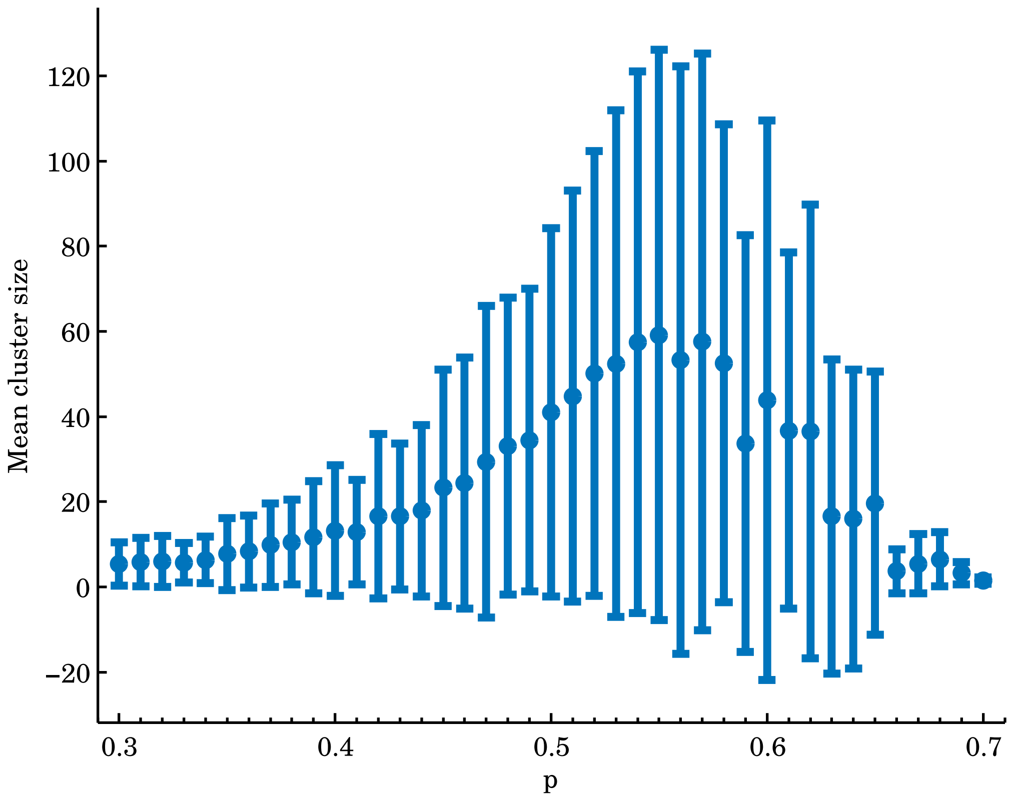


Figure 2: *Mean cluster sizes μ, represented as points, and standard deviations σ, indicated by the vertical error bars, computed as a function of p, with a step size of 0.1. Values μ and σ were calculated over 200 runs on a 41×41 grid.*

cluster. As $p > p_c$ increases the probability of ending with a finite cluster decreases, until we always get the percolating cluster for $p = 1$. Note that although in theory this cluster should cover the fill grid, this is not necessarily the case in our model, since it stops growing as soon as one border site is occupied.

Code heeft het wel over $p = 0.301, 0.302, \ldots, 0.7$, maar de plaatjes hebben de range uti de tekst.

To find an indication of the value of p_c with our model we have let it generate a cluster on a 41×41 grid for $p = 0.31, 0.32, \ldots, 0.7$. For each value of p we grow $r_{max} = 200$ clusters.

Figure 2 presents the mean and standard deviation of the size of the finite clusters as a function of p. In this figure we observe the effect of p on the mean cluster size described by Kenzel et al. Furthermore, based on these data one would estimate p_c to be approximately 0.55.

Kenzel et al. also predicted that the number of percolating clusters relative to the number of finite clusters would grow for $p > p_c$ until $p = 1$, where the only possibility would be a percolating cluster. To observe this effect figure 3 shows p_∞, which is the ratio of the number of percolating clusters to the number of finite clusters. This graph is based on the same data as figure 2. Based on this graph we would say that $p_c \approx 4\text{e}^{-1}$. This number is lower than the value for p_c based on figure 2, this is probably caused by the relatively small grid sizes, which causes us to classify some clusters as percolating, that are actually finite.

Stauffer [3] has found p_c to be approximately 5.93e^{-1} for a square lattice. As stated earlier our lower estimation of p_c is quite likely caused by our small grid.

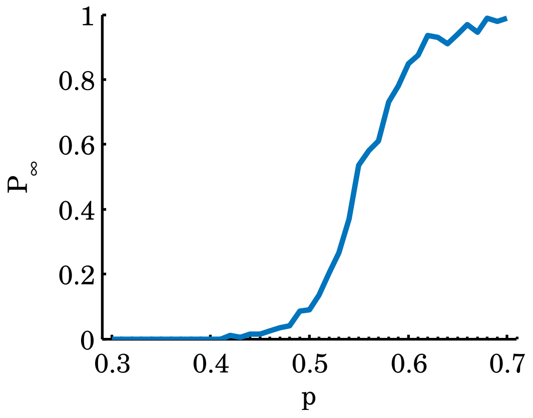


Figure 3: *Ratio of percolating clusters, P_∞, as a function of p. Ratios calculated over $r_{max} = 200$ runs on a grid size of 41×41.*

3.2. System Size

How do the results change when the system size changes. Experiment with different latice sizes

Wat could the behavior be in the limit of infinite lattice sizes

3.3. Fractal Dimension

Definitie van fractal dimension

Welke fractal dimension hebben anderen gevonden?

Boxcounting = MinkowskiâĂŞBouligand dimension uitleggen met source

$$(1) \quad \dim_{\text{box}} = \lim_{\epsilon \to 0} \frac{\log N(\epsilon)}{\log \left[\frac{1}{\epsilon}\right]}.$$

We have used the function `boxcount` by Moisy [2] to determine the fractal dimension of different clusters. This method uses boxsizes that are power of two consequently $\varepsilon = 1, 2, 3, \ldots 2^Q$ where Q is the smallest integer such that $Q \leq (2N + 1)$. We have used the boxcounting algorithm on a cluster gener-

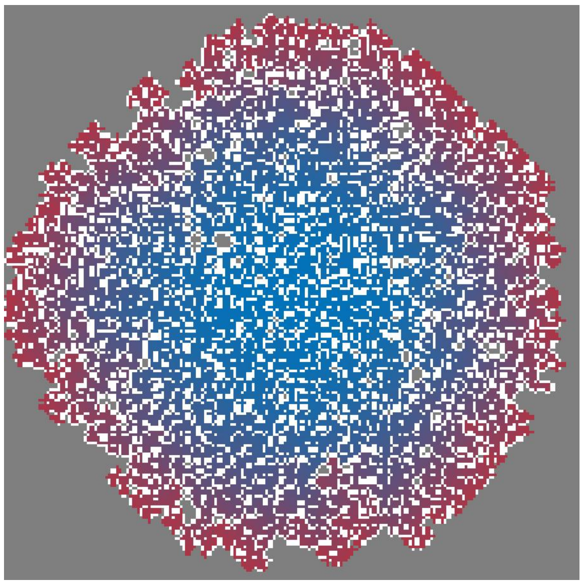


Figure 4: *The cluster used to determine the Minkowski-Bouligand dimension of clusters generated with percolation, the clusters was generated with $N = 80$, $p = 0.7$.*

ated with $N = 80$, $p = 0.7$, the used cluster is shown in figure 4.

Figure 5 presents the number of boxes as a function of the size of the boxes.

Observatie

Past de gevonden fractal dimension met de theorie?

3.4. Connectivity

Plaatjes opnieuw maken met matlab, zodat de kleuren fatsoenlijk overeen komen!

How does the connectivity influence the final cluster

4. Conclusion

Vat bevindingen van experiment samen

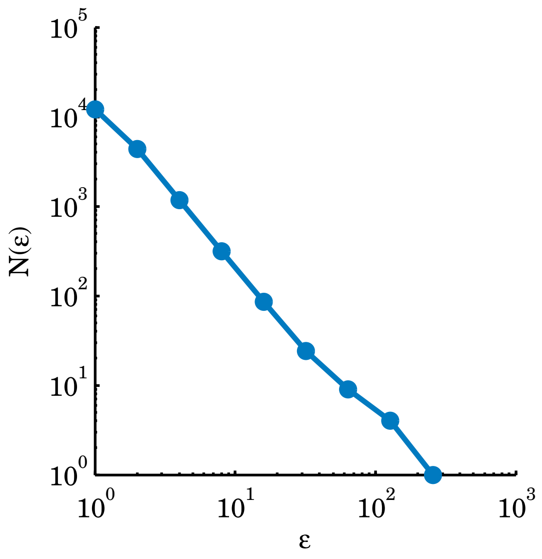
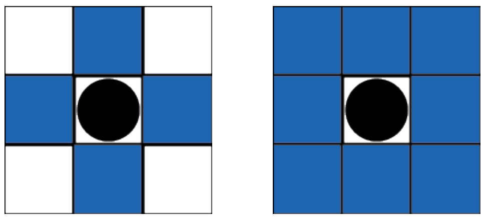


Figure 5: *The number of boxes used to cover a cluster ($N = 80, p = 0.7$) as a function of the box size for the cluster presented in figure 4.*



(a) *4-connectivity* **(b)** *8-connectivity*

Figure 6: *Connectivity masks for (a) four-connectivity and (b) eight-connectivity. The filled-in squares are considered neighbors of the center square, which is indicated with a dot.*

References

[1] Wolfgang Kenzel et al. *Physics by computer*. Springer-Verlag New York, Inc., 1997.

[2] F. Moisy. *Computing a fractal dimension with Matlab: 1D, 2D and 3D Box-counting*. 2008. URL: `http://www.fast.u-psud.fr/~moisy/ml/boxcount/html/demo.html`.

[3] Dietrich Stauffer. *Introduction to percolation theory*. Taylor & Francis, 1985. Chap. 2, pp. 15–58.