*МЕТОДИЧЕСКИЕ МАТЕРИАЛЫ*

*Вестник Сыктывкарского университета.*
*Серия 1: Математика. Механика. Информатика.*
*Выпуск 3 (24). 2017*

**УДК 537.622**

# ЗАДАЧА О ПЕРКОЛЯЦИИ

***В. А. Устюгов, А. Е. Чуфырев***

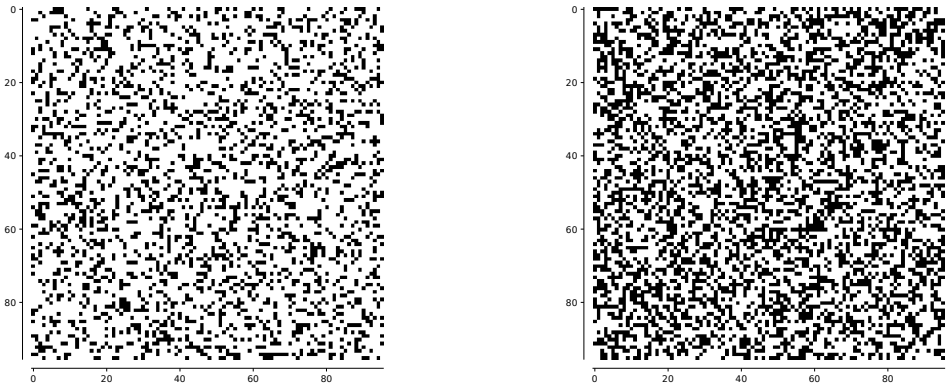
В статье дан обзор алгоритмов для решения задачи поиска стягивающего кластера на квадратной решетке. Описана методика определения порога перколяции, а также нахождения зависимости доли ячеек стягивающего кластера от общего числа занятых ячеек. Объяснено сингулярное поведение последней зависимости в окрестности критической концентрации.
*Ключевые слова:* перколяция, стягивающий кластер, алгоритм Хошена-Копельмана.

Задача о перколяции возникает в совершенно различных областях науки: при анализе зависимости проводимости тонких композитных пленок от соотношения концентраций металла и диэлектрика в составе пленки; при определении связности разветвленных компьютерных сетей; также в экономических и социальных науках (теория шести рукопожатий и др.).

В качестве модельной системы рассмотрим квадратную решетку, каждая из позиций которой занята с некоторой заданной вероятностью $p$. На рисунке занятые позиции обозначены цветом, а незанятые оставлены белыми. Если у клеток есть общая граница, будем считать их соединенными. Из цепочек соединенных между собой клеток образуются *кластеры*.

При малой вероятности заполнения $p$ практически все занятые клетки изолированы друг от друга (рис. 1a,b), т. е. большинство кластеров системы имеет размер 1. При незначительном увеличении концентрации до 0.4 почти все клетки объединяются в кластеры с характерным размером 5–10. При $p = 0.8$ (рис. 2b) с очень малой долей вероятности остаются одиночные клетки, практически все частицы объединены

© Устюгов В. А., Чуфырев А. Е., 2017.




a) b)

**Рис. 1.** Решетки, заполненные частицами с вероятностью a) 0.25; b) 0.4

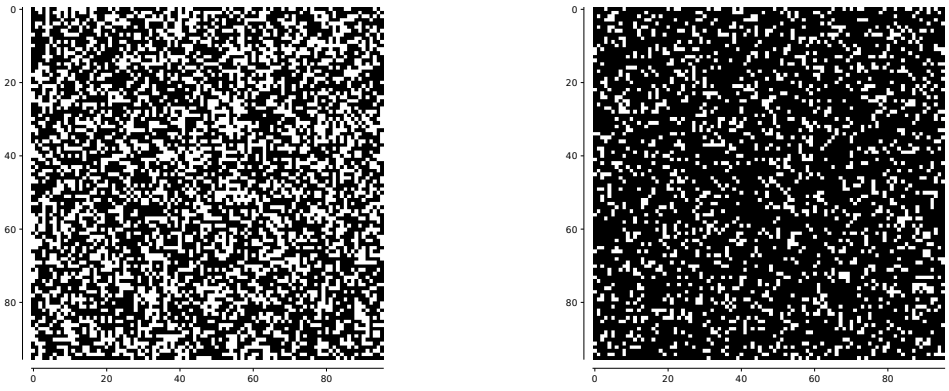
в кластеры большого размера. Фактически при высокой концентрации почти частицы системы являются частью одного большого кластера.

Теория перколяции изучает строение систем вблизи порога перколяции (порога протекания), т. е. при концентрациях $p$, при которых стабильно начинает появляться перколяционный кластер [3]. Это эквивалентно фазовому переходу второго рода, при котором происходит существенное изменение упорядоченности вещества. Для бесконечной квадратной решетки критическая концентрация составляет $p_c = 0.593$. Заметим, что вдали от порога протекания для описания системы достаточно модели эффективной среды.

## 1. Определение порога перколяции

Для ряда двумерных решеток критическая концентрация может быть рассчитана аналитически, однако мы воспользуемся методом моделирования. На пустую решетку будем помещать частицы, выбирая координаты случайным образом и анализируя состояние решетки после каждого шага. Когда на решетке появится стягивающий кластер, процесс будет остановлен. Очевидно, что в этот момент концентрация частиц на решетке близка к $p_c$. Для более точного вычисления $p_c$ эксперимент требуется провести многократно и усреднить результаты.

Основная проблема, возникающая при решении этой задачи, — определение наличия перколяционного кластера (рис. 3). Простой способ решения состоит в том, чтобы помечать частицы уникальными числовыми метками кластеров, к которым они принадлежат. При этом про-




a) b)

**Рис. 2.** Решетки, заполненные частицами с вероятностью a) 0.6; b) 0.8

верка наличия протекания сведется к поиску частиц, находящихся на противоположных границах решетки и имеющих одинаковые метки.

Прямолинейное решение может строиться по следующему алгоритму [2]:

1. Перед началом работы алгоритма решетка пуста. Инициализируем все ячейки нулями, означающими, что ячейки не заняты.

2. Выбираем случайную позицию в решетке и занимаем ее, занося в соответствующую ячейку массива номер кластера 1.

3. Случайным образом выбираем вторую позицию и проверяем соседние ячейки. Если они пусты (содержат нули), присваиваем текущей ячейке номер кластера 2. Если некоторой соседней ячейке ранее бы присвоен номер, то он присваивается и текущей.

4. При выборе следующей ячейки в случае отсутствия соседей ей присваивается очередной свободный номер кластера. Если же соседние ячейки заняты и принадлежат одному кластеру, то к нему подключается и выбранная. Однако, если ячейка становится точкой соединения двух и более кластеров, требуется выбрать новый номер (это может быть не использовавшееся ранее число либо, например, минимальный номер кластера среди соединяемых) и переобозначить все ячейки, принадлежащие сливающимся в единый блок кластерам.

5. Продолжаем процедуру занятия ячеек и анализа, пока в решетке не появится кластер, частицы которого занимают позиции на всех двух или четырех (в зависимости от формулировки задачи) границах. Доля занятых ячеек и есть искомая концентрация $p_c$.

Недостаток описанного алгоритма заключается в низкой эффективности из-за необходимости переобозначения ячеек кластеров при слиянии последних. Если $S$ — число ячеек решетки, то время работы алгоритма пропорционально $S^2$.

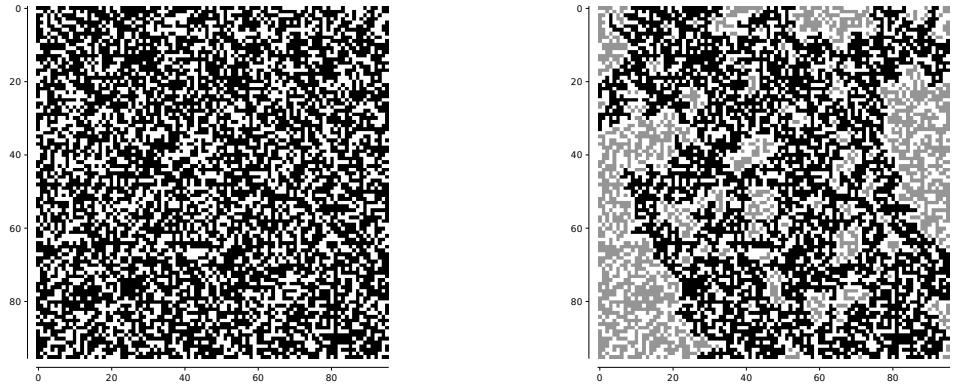



**Рис. 3.** Решетка, заполненная частицами с вероятностью 0.6 (a) и выделенный перколяционный кластер (b)

Алгоритм Хошена-Копельмана (алгоритм многократной маркировки кластеров) позволяет определить наличие протекания за время, пропорциональное $S$. Рассмотрим процесс работы с двумерной решеткой, заполненной частицами с вероятностью $p$. С этой же вероятностью массив `lattice[]`, хранящий состояние системы, заполнен единицами. Также для удобства (единообразия работы с ячейками) исследуемую решетку будем хранить в строках и столбцах массива, начиная с индексов, равных 1, а в нулевые строку и столбец запишем нули.

Суть алгоритма заключается в маркировке кластеров таким образом, что одному кластеру может соответствовать несколько различных меток [1]. Для хранения меток кластеров будем использовать одномерный массив `labels[]`, проинициализированный номерами собственных ячеек. Такие значения мы назовем *правильными*. В процессе прохода исследуемой решетки значения `labels[]` могут изменяться. Если число, хранящееся в ячейке, становится меньше индекса ячейки, то такое

значение мы будем называть *неправильным*. Алгоритм будет содержать следующие шаги:

1. Проход массива начинаем с элемента `lattice[1,1]` (с верхнего левого угла) и просмотр ведем, двигаясь вдоль строк. Если значение в очередной выбранной ячейке равно нулю, переходим к следующей.

2. Если текущее значение элемента равно 1 (позиция занята), выполняем проверку значений левой и верхней соседних ячеек `lattice[]`. Если они равны нулю, то делаем предположение, что текущий элемент входит в новый кластер и присваиваем ему свободный номер кластерной метки.

3. Если значение ячейки сверху равно 0, а слева не равно нулю, то текущая ячейка и левый сосед принадлежат одному кластеру. Текущему элементу присвоим номер кластерной метки соседа слева.

4. Аналогично, если значение ячейки слева равно 0, а сверху не равно нулю, то текущая ячейка и верхний сосед принадлежат одному кластеру. Текущему элементу присваиваем номер *правильной* кластерной метки соседа сверху. Это важный момент, поскольку у соседа сверху в результате слияний кластеров кластерная метка может стать неправильной.

5. Если и верхний, и левый сосед имеют ненулевые значения, то все три ячейки принадлежат одному кластеру. При этом текущему элементу присваивается наименьший из номеров *правильных* кластерных меток соседей сверху и слева. В этом случае требуется корректирование массива `labels[]`: в элемент массива, содержащий большее значение правильной кластерной метки, записываем также определенный выше минимальный номер.

Вернемся к вопросу о вычислении критической концентрации $p_c$. Строго говоря, $p_c$ — это наименьшая концентрация, при которой впервые появляется перколяционный кластер на решетке бесконечного размера. Для решеток конечного размера, используемых при моделировании, найденное значение концентрации может варьироваться от эксперимента к эксперименту, и для вычисления истинного значения требуется усреднение по результатам многократно проведенного моделирования. Также значение $p_c$ будет зависеть от принятого определения

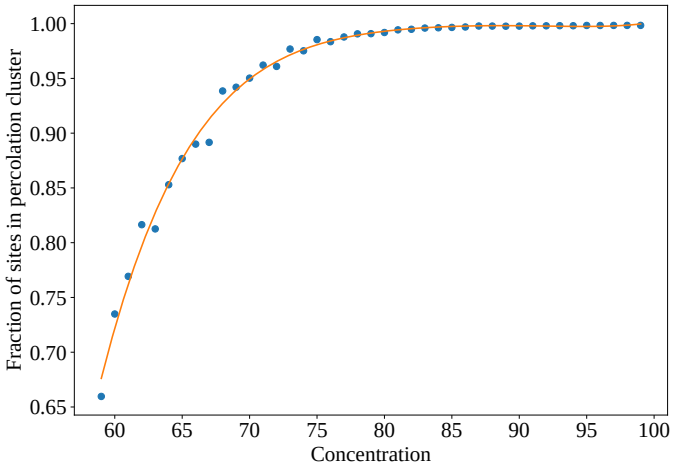



**Рис. 4.** Зависимость доли занятых ячеек, принадлежащих перколяционному кластеру, от концентрации

протекания (т. е. от того, какой кластер считается стягивающим: касающийся всех границ или только двух противоположных). Можно найти, что критическая концентрация решетки конечного размера тем ближе к 0.593, чем больше размер этой решетки.

Поведение системы вблизи порога перколяции — это пример фазового перехода второго рода: несвязное состояние системы, включающей множество мелких кластеров, сменяется связным, при котором в системе присутствует один крупный стягивающий кластер. Так, на границе фазового перехода различные величины, характеризующие свойства системы, имеют сингулярное поведение, причем эти сингулярности часто описываются степенными законами. В качестве примера рассмотрим стягивающий кластер при концентрации выше $p_c$. Из рис. 2 видно, что для высоких значений $p$ практически все занятые ячейки принадлежат одному кластеру, однако при приближении к критической концентрации система содержит множество небольших кластеров.

В качестве примера рассмотрим стягивающий кластер при концентрации выше $p_c$. Из рис. 2 видно, что для высоких значений $p$ практически все занятые ячейки принадлежат одному кластеру, однако при приближении к критической концентрации система содержит множество небольших кластеров.

Рассмотрим долю ячеек $F$, принадлежащих стягивающему кластеру, от общего числа *занятых* ячеек. Для этого будем генерировать ре-

шетки с заданной концентрацией частиц, производить маркировку кластеров с помощью алгоритма Хошена-Копельмана, определять наличие протекания и находить искомое значение $F$. График зависимости $F(p)$ представлен на рис. 4. Мы видим, что при уменьшении концентрации $p$ величина $F$ также уменьшается, и можно предположить, что при $p = p_c$ величина $F$ обращается в нуль. Оказывается, что вблизи критической концентрации $F$ изменяется согласно степенному закону

$$F = F_0(p - p_c)^\beta, \tag{1}$$

где $\beta$ — критический показатель. Несмотря на то что набор точек на рис. 4 не позволяет точно рассчитать $\beta$, очевидно, что функция $F(p)$ имеет сингулярное поведение: $dF/dp \to \infty$ при $p \to p_c$. Аналитически определенное точное значение $\beta$ составляет $5/36$ для двумерных решеток с квадратными, треугольными и шестиугольными (пчелиные соты) ячейками.

Обратим внимание на следующую особенность зависимости $F(p)$. При наличии критической концентрации занятых ячеек бесконечной решетки стягивающий кластер должен иметь бесконечный размер, однако при этом доля его ячеек среди общего числа занятых становится исчезающе мала: $F \to 0$! Если рассмотреть перколяционный кластер при относительно малой концентрации $p = 0.6$ (рис. 3b), видно, что во многих областях перемычки между составляющими его подкластерами являются тонкими, т. е. исключение из такой системы даже малого числа ячеек может привести к потере связности. Это говорит о фрактальном характере стягивающего кластера, т. е. данный объект обладает нетривиальной структурой как целое, и подобной структурой обладают и его составляющие.

## 2. Алгоритм решения задачи

Код для решения поставленной задачи написан на языке Python 3.6, для работы требуется библиотека математических процедур `numpy`. Массив, в пределах которого осуществляется поиск перколяционного кластера `lattice`, первоначально заполняется случайным образом. В процессе работы для нахождения правильной метки кластера используется функция `properize`, осуществляющая рекурсивный поиск необходимого значения. В последнем цикле `for` происходит поиск одинаковых меток кластеров на границах массива с переобозначенными в ходе работы алгоритма ячейками.

```
import numpy as np

def properize(labels, cell):
    if labels[cell] == cell:
        return cell
    while labels[cell] != cell:
        cell = labels[cell]
    return cell

SIZE = 96
CONC = 60

# Generate our lattice
lattice = np.random.randint(100, size=(SIZE+1, SIZE+1))
lattice[lattice<=CONC] = 1
lattice[lattice>CONC] = 0

lattice[0] = 0   # zero first array row
lattice[:,0] = 0 # zero first array column

labels = np.arange(SIZE*SIZE/2, dtype=np.int32)
currLabel = 1

# Iterate the array; x is column number, y is row
for (x,y), value in np.ndenumerate(lattice):

    if not value:
        continue        # If lattice[x,y] == 0

    if not lattice[x,y-1] and not lattice[x-1,y]:
        lattice[x,y] = currLabel
        currLabel += 1
        continue

    #  left neighbor
    if lattice[x, y-1] and not lattice[x-1,y]:
        lattice[x,y] = lattice[x, y-1]
        continue

    # upper neighbor
```

```
        if not lattice[x,y-1] and lattice[x-1,y]:
            lattice[x,y] = properize(labels, lattice[x-1,y])
            continue

        up_proper = properize(labels, lattice[x-1, y])
        left_proper = properize(labels, lattice[x, y-1])

        lattice[x,y] = min(up_proper, left_proper)
        labels[max(up_proper, left_proper)] = lattice[x,y]

    # Correction of the lattice
    for (x,y), value in np.ndenumerate(lattice):

        if not value:
            continue        # If lattice[x,y] == 0

        properLabel = value
        while labels[properLabel] != properLabel:
            properLabel = labels[properLabel]

        lattice[x,y] = properLabel

    for x, value in np.ndenumerate(lattice[:,1]):
        if value and value in lattice[:,SIZE]:
            print("Percolate!")
```

### 3. Заключение

Таким образом, метод Хошена-Копельмана является эффективным инструментом для определения наличия перколяционного кластера и маркировки ячеек, принадлежащих к нему. Реализация на языке Python позволяет успешно применять описанный метод для решения задач разных областей науки в образовательном процессе.

## Список литературы

1. **Giordano N. J.** Computational physics / N. J. Giordano, H. Nakanishi. Pearson/Prentice Hall, 2006. 544 p.

2. **Гулд Х., Тобочник Я.** Компьютерное моделирование в физике. М.: Мир, 1990. 400 с.

3. **Тарасевич Ю. Ю.** Перколяция: теория, приложения и алгоритмы: Учебное пособие. М.: Едиториал УРСС, 2002. 112 с.

**Summary**

**Ustyugov V. A., Chufyrev A. E.** The percolation problem

The article gives an overview of algorithms for solving the problem of finding a spanning cluster on a square lattice. A technique for determining the percolation threshold is described. The singular behavior of the last dependence in the vicinity of the critical concentration is explained. The solution of the problem in the form of a program in Python programming lagnuage is given.

*Keywords: percolation, spanning cluster, Hoshen-Kopelman algorithm.*

**References**

1. **Giordano N. J.** *Computational physics* / N. J. Giordano, H. Nakanishi, Pearson/Prentice Hall, 2006, 544 p.

2. **Gould H., Tobochnik J.** *Komp'yuternoe modelirovanie v fizike* (Computer modelling in physics), M.: Mir, 1990, 400 p.

3. **Tarasevich Yu. Yu.** *Perkolyaciya: teoriya, prilozheniya i algoritmy* (Percolation: theory, application, algorithms), M.: Editorial URSS, 2002, 112 p.

**Для цитирования:** Устюгов В. А.,Чуфырев А. Е. Задача о перколяции // *Вестник Сыктывкарского университета. Сер. 1: Математика. Механика. Информатика. 2017. Вып. 3 (24). С. 104–113.*

**For citation:** Ustyugov V. A., Chufyrev A. E. The percolation problem, *Bulletin of Syktyvkar University, Series 1: Mathematics. Mechanics. Informatics*, 2017, №3 (24), pp. 104–113.

*СГУ им. Питирима Сорокина* *Поступила 01.10.2017*