

## A Case Study: Percolation

**Percolation.** Pour liquid on top of some porous material.  
Will liquid reach the bottom?

**Applications.** [ chemistry, materials science, ... ]

- Chromatography.
- Spread of forest fires.
- Natural gas through semi-porous rock.
- Flow of electricity through network of resistors.
- Permeation of gas in coal mine through a gas mask filter.
- ...

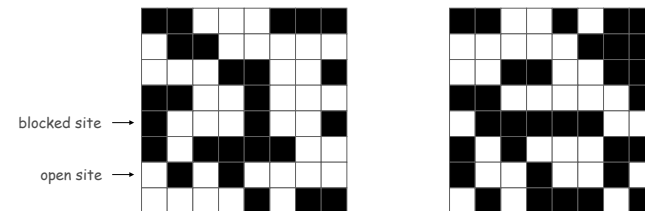
1

## A Case Study: Percolation

**Percolation.** Pour liquid on top of some porous material.  
Will liquid reach the bottom?

**Abstract model.**

- $N$ -by- $N$  grid of sites.
- Each site is either **blocked** or **open**.



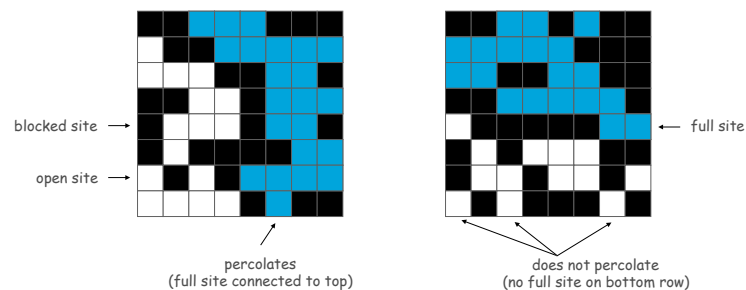
2

## A Case Study: Percolation

**Percolation.** Pour liquid on top of some porous material.  
Will liquid reach the bottom?

**Abstract model.**

- $N$ -by- $N$  grid of sites.
- Each site is either **blocked** or **open**.
- An open site is **full** if it is connected to the top via open sites.



3

## A Scientific Question

**Random percolation.** Given an  $N$ -by- $N$  system where each site is vacant with probability  $p$ , what is the probability that system percolates?



**Remark.** Famous open question in statistical physics.

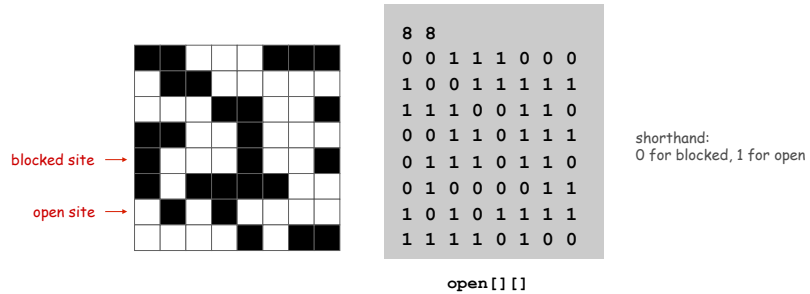
no known mathematical solution

**Recourse.** Take a computational approach: **Monte Carlo simulation.**

4

## Data Representation

**Data representation.** Use one  $N$ -by- $N$  boolean matrix to store which sites are open; use another to compute which sites are full.

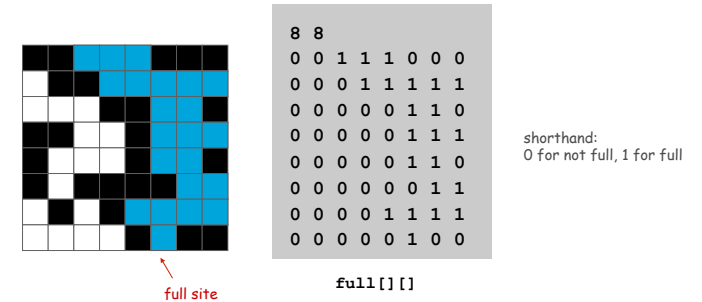


5

## Data Representation

**Data representation.** Use one  $N$ -by- $N$  boolean matrix to store which sites are open; use another to compute which sites are full.

**Standard array I/O library.** Library to support reading and printing 1- and 2-dimensional arrays.



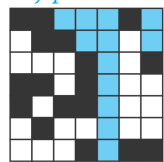
6

## Vertical Percolation

**Next step.** Start by solving an easier version of the problem.

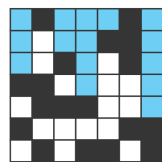
**Vertical percolation.** Is there a path of open sites from the top to the bottom that goes **straight down**?

*vertically percolates*



site connected to top with a vertical path

*does not vertically percolate*



no open site connected to top with a vertical path

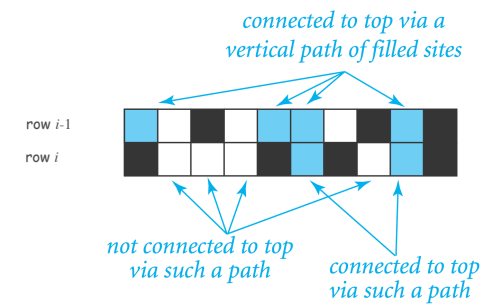
7

## Vertical Percolation

**Q.** How to determine if site  $(i, j)$  is full?

**A.** It's full if  $(i, j)$  is open and  $(i-1, j)$  is full.

**Algorithm.** Scan rows from top to bottom.



8

## Vertical Percolation

Q. How to determine if site  $(i, j)$  is full?

A. It's full if  $(i, j)$  is open and  $(i-1, j)$  is full.

Algorithm. Scan rows from top to bottom.

```
public static boolean[][] flow(boolean[][] open) {  
    int N = open.length;           ← initialize  
    boolean[][] full = new boolean[N][N];  
    for (int j = 0; j < N; j++)  
        full[0][j] = open[0][j];  
  
    for (int i = 1; i < N; i++)     ← find full sites  
        for (int j = 0; j < N; j++)  
            full[i][j] = open[i][j] && full[i-1][j];  
  
    return full;  
}
```

9

## Vertical Percolation: Testing

Testing. Add helper methods to generate random inputs and visualize using standard draw.

```
public class Percolation {  
    ...  
  
    // return a random N-by-N matrix; each cell true with prob p  
    public static boolean[][] random(int N, double p) {  
        boolean[][] a = new boolean[N][N];  
        for (int i = 0; i < N; i++)  
            for (int j = 0; j < N; j++)  
                a[i][j] = StdRandom.bernoulli(p);  
        return a;  
    }  
  
    // plot matrix to standard drawing  
    public static void show(boolean[][] a, boolean foreground)  
}
```

10

## General Percolation: Recursive Solution

Percolation. Given an  $N$ -by- $N$  system, is there **any** path of open sites from the top to the bottom.

not just straight down

Depth first search. To visit all sites reachable from  $i$ - $j$ :

- If  $i$ - $j$  already marked as reachable, return.
- If  $i$ - $j$  not open, return.
- Mark  $i$ - $j$  as reachable.
- Visit the 4 neighbors of  $i$ - $j$  recursively.

Percolation solution.

- Run DFS from each site on top row.
- Check if any site in bottom row is marked as reachable.



11

## Depth First Search: Java Implementation

```
public static boolean[][] flow(boolean[][] open) {  
    int N = open.length;  
    boolean[][] full = new boolean[N][N];  
    for (int j = 0; j < N; j++)  
        if (open[0][j]) flow(open, full, 0, j);  
    return full;  
}  
  
public static void flow(boolean[][] open,  
                        boolean[][] full, int i, int j) {  
    int N = full.length;  
    if (i < 0 || i >= N || j < 0 || j >= N) return;  
    if (!open[i][j]) return;  
    if (full[i][j]) return;  
  
    full[i][j] = true;           // mark  
    flow(open, full, i+1, j);   // down  
    flow(open, full, i, j+1);   // right  
    flow(open, full, i, j-1);   // left  
    flow(open, full, i-1, j);   // up  
}
```

12