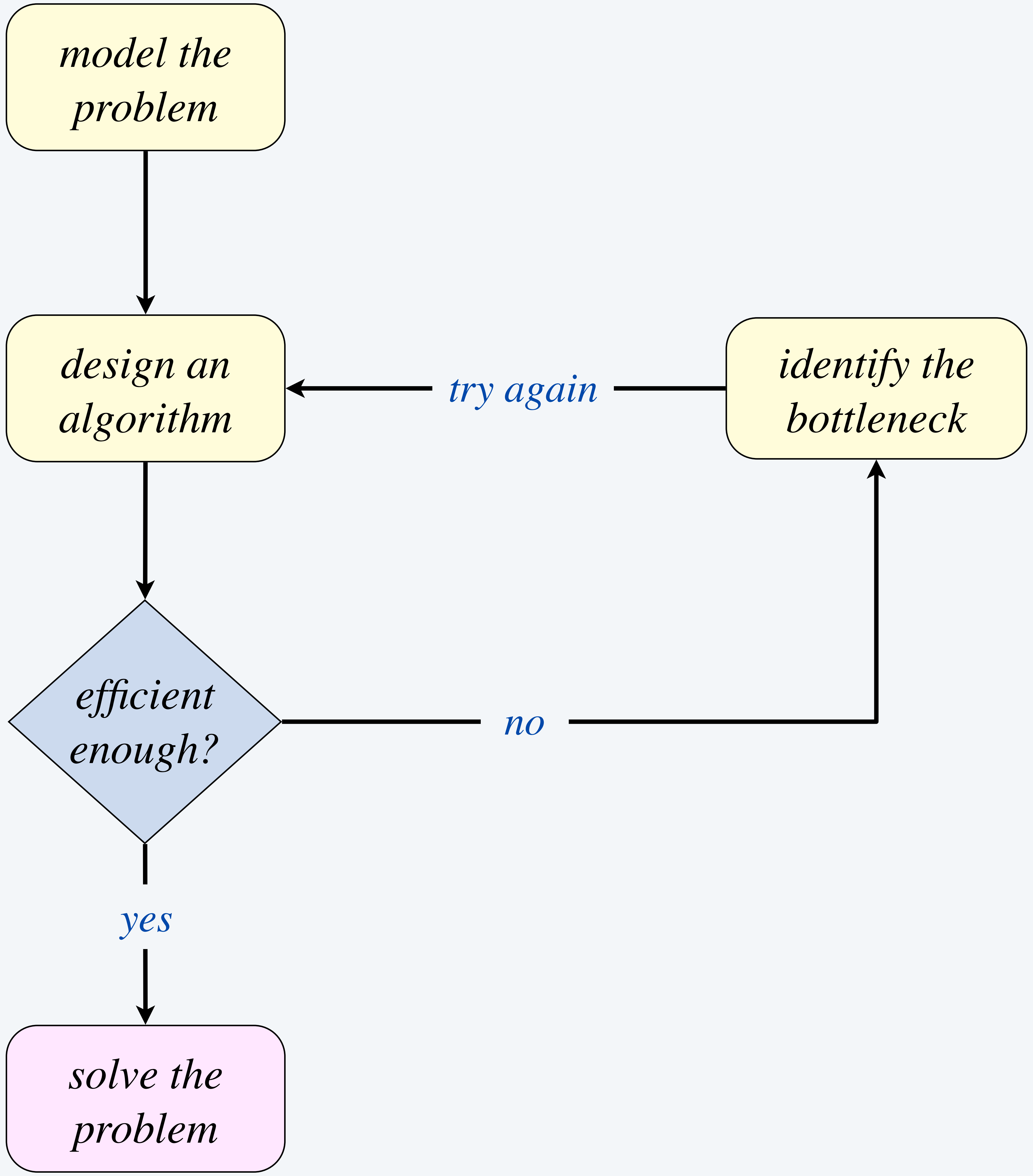

https://algs4.cs.princeton.edu

# Percolation

# Subtext of today's lecture (and this course)

Steps to develop a usable algorithm to solve a computational problem.



# Percolation

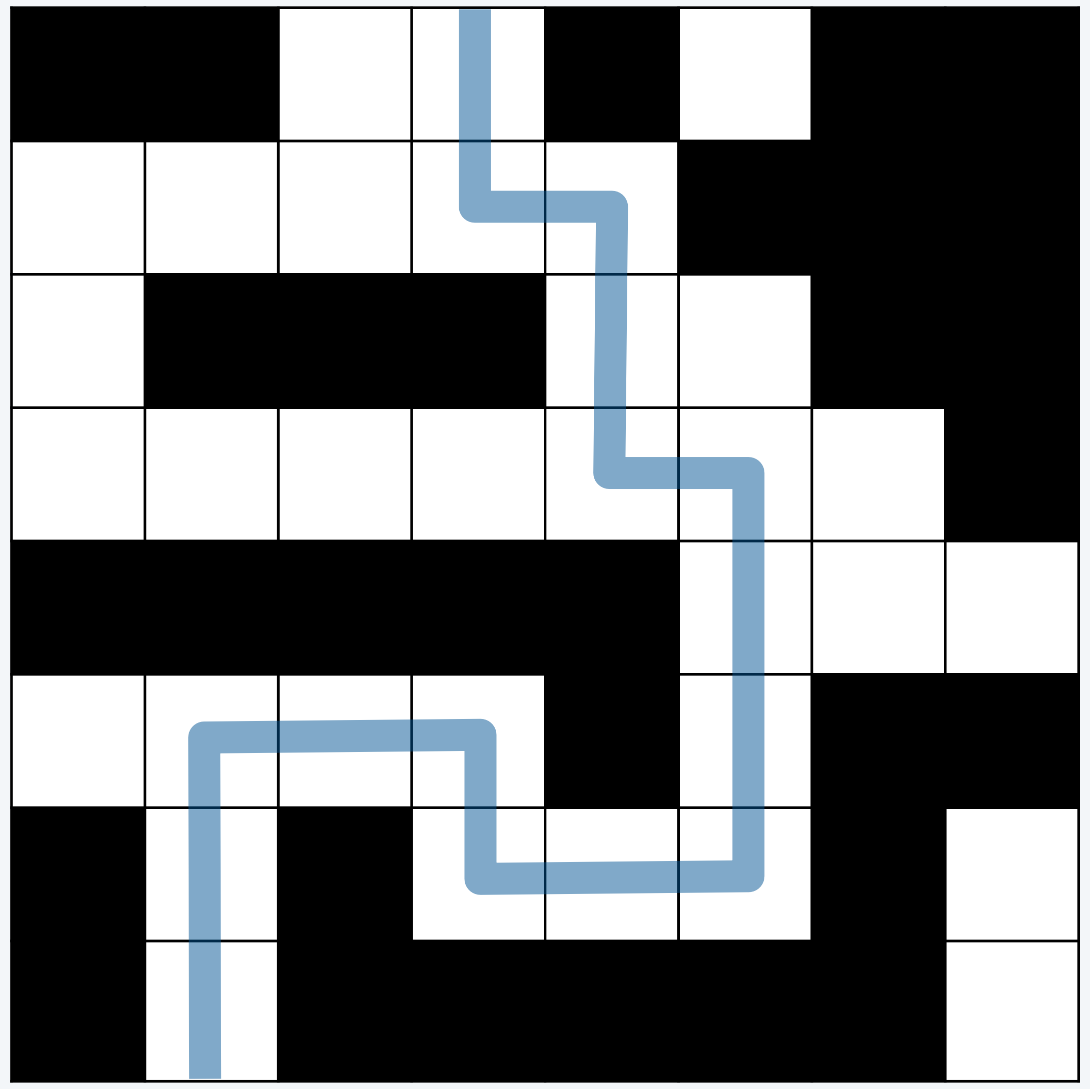
An abstract model of many physical systems:

- $n$-by-$n$ grid of sites.
- Each site is independently open with probability $p$ (and blocked otherwise).
- System **percolates** if there is a path of neighboring open sites from the top row to the bottom row.

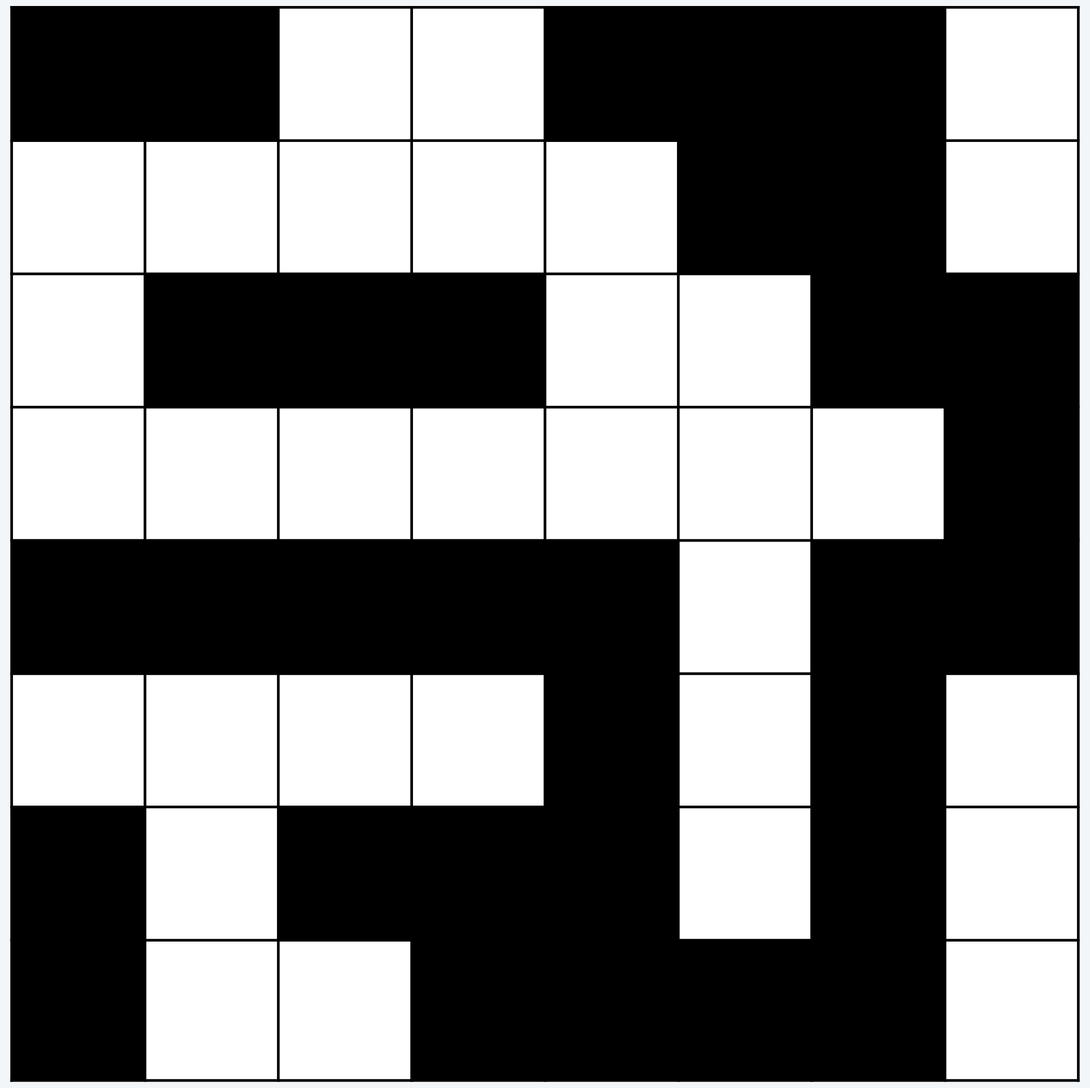
*horizontal or vertical*

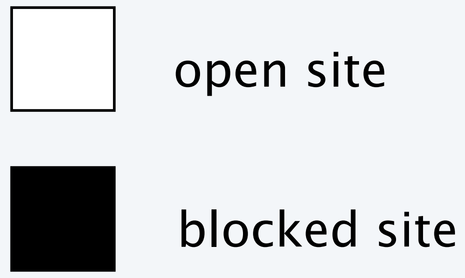
**Intuition.** Pour liquid onto a porous material. Will the liquid reach the bottom?

**percolates**



**does not percolate**



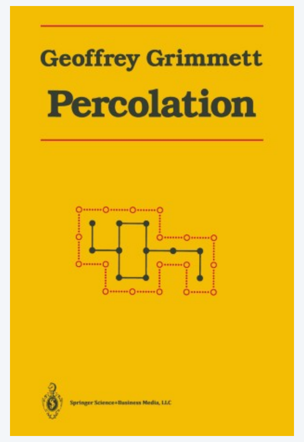


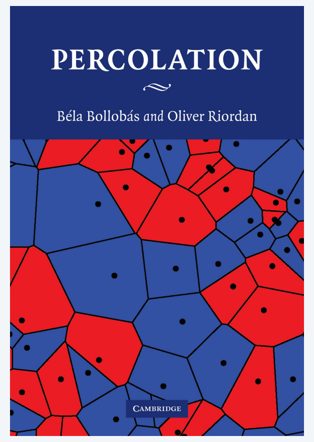
# Percolation

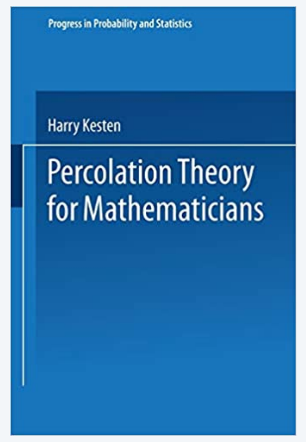
An abstract model of many physical systems:

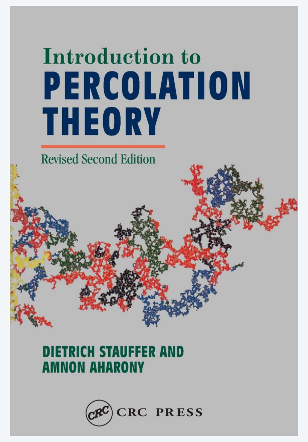
- $n$–by–$n$ grid of sites.
- Each site is independently open with probability $p$ (and blocked otherwise).
- System percolates if there is a path of neighboring open sites from the top row to the bottom row.

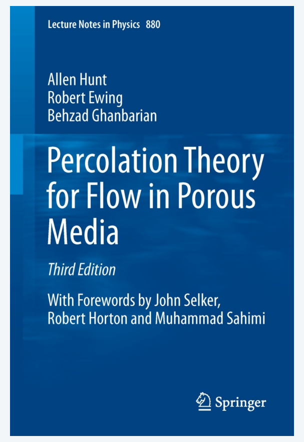
| system | open site | blocked site | if system percolates … |
|---|---|---|---|
| *geological* | cavity | rock | fluid can flow |
| *electrical* | metallic | dielectric | electric current can flow |
| *ecological* | tree | no tree | wildfire can propagate |
| *epidemiological* | susceptible | immune | infection can spread |

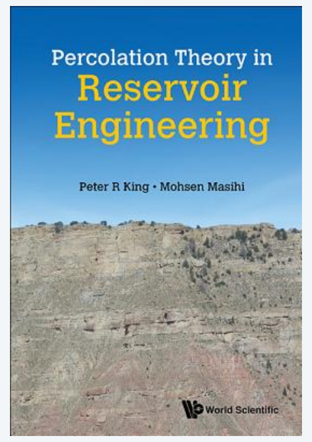


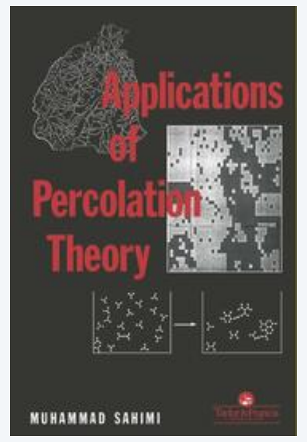


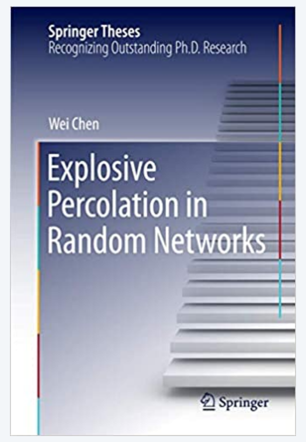












# Percolation poll 1

**Which of these systems percolate?**

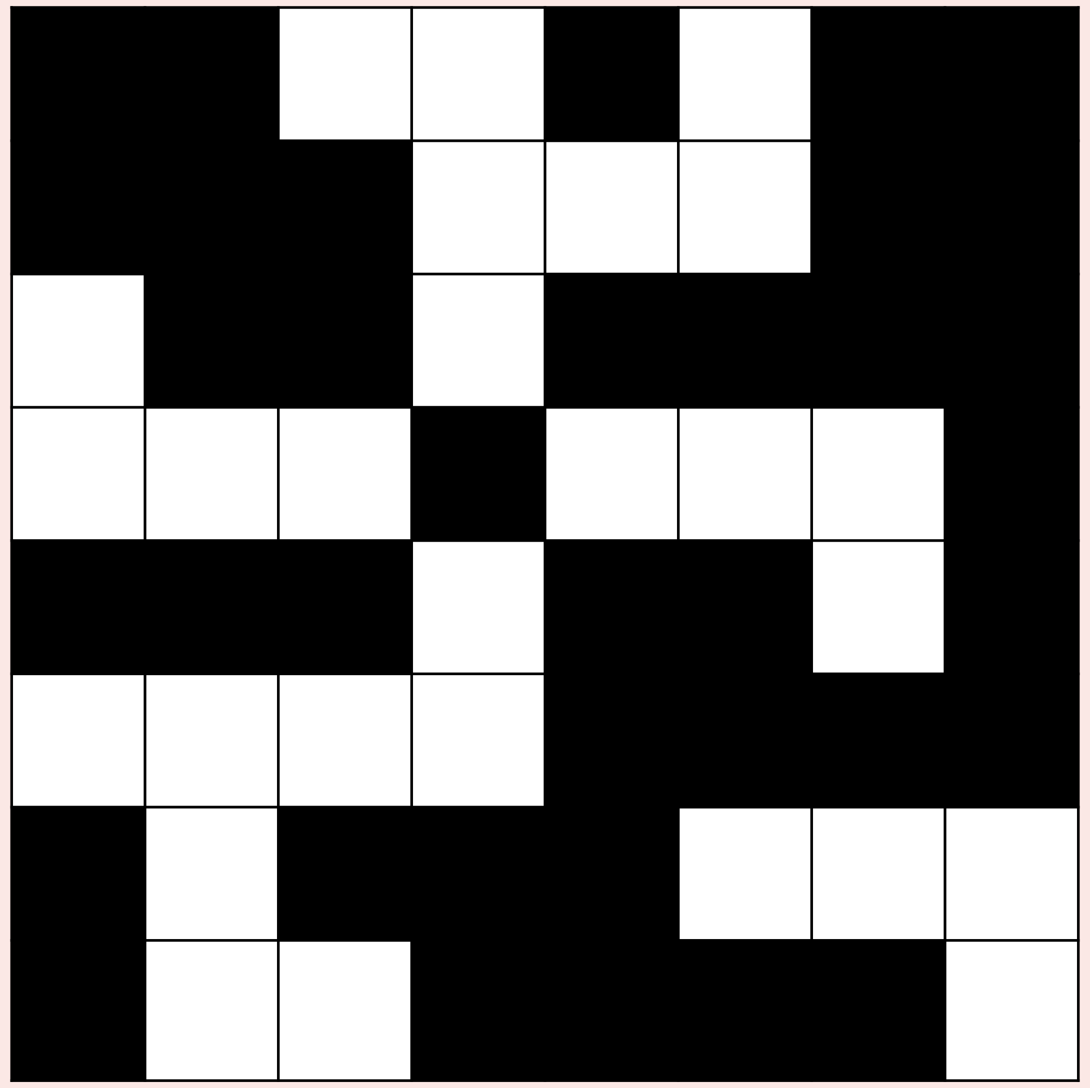
**A.** A only

**B.** B only

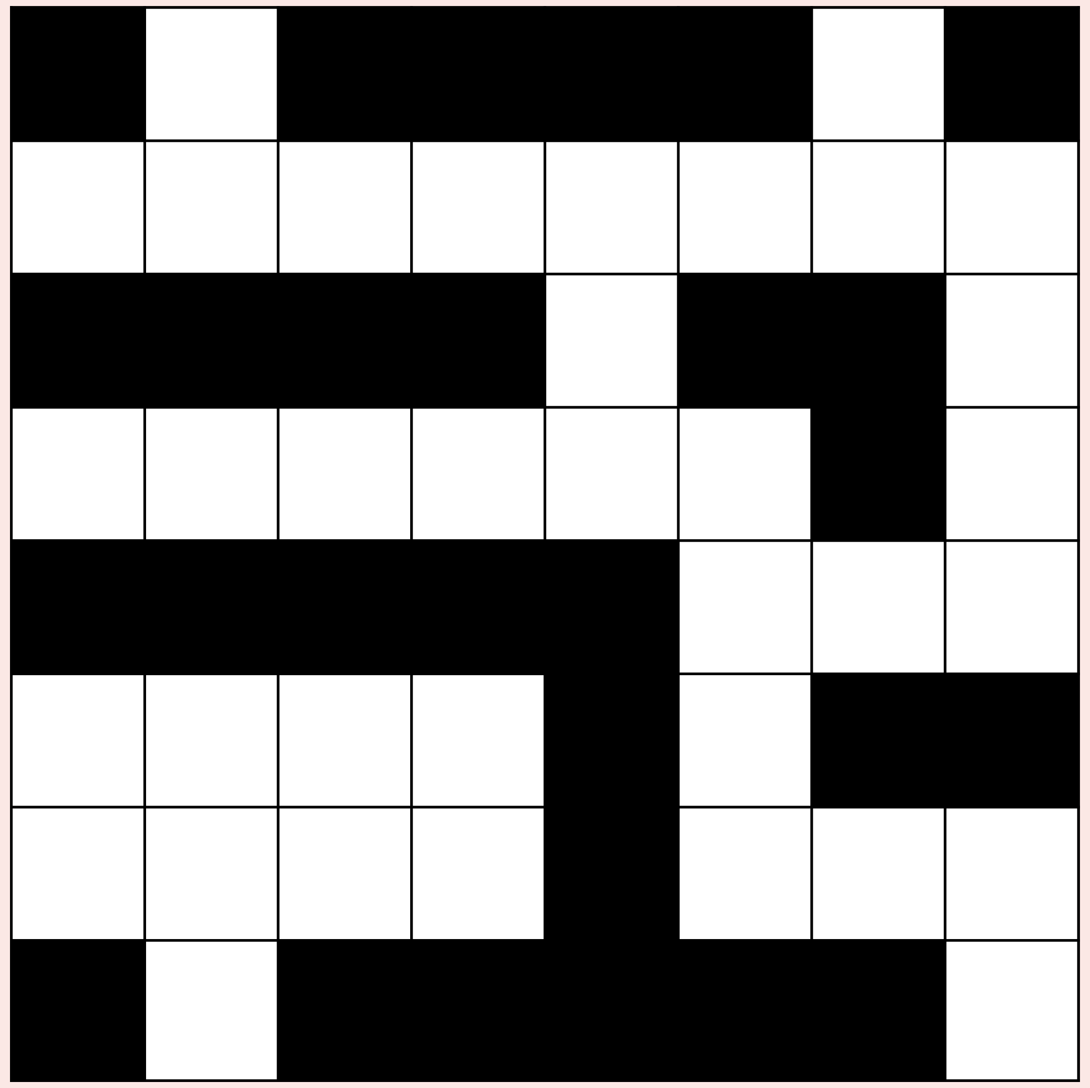
**C.** Both A and B

**D.** Neither A nor B

**A.**



**B.**

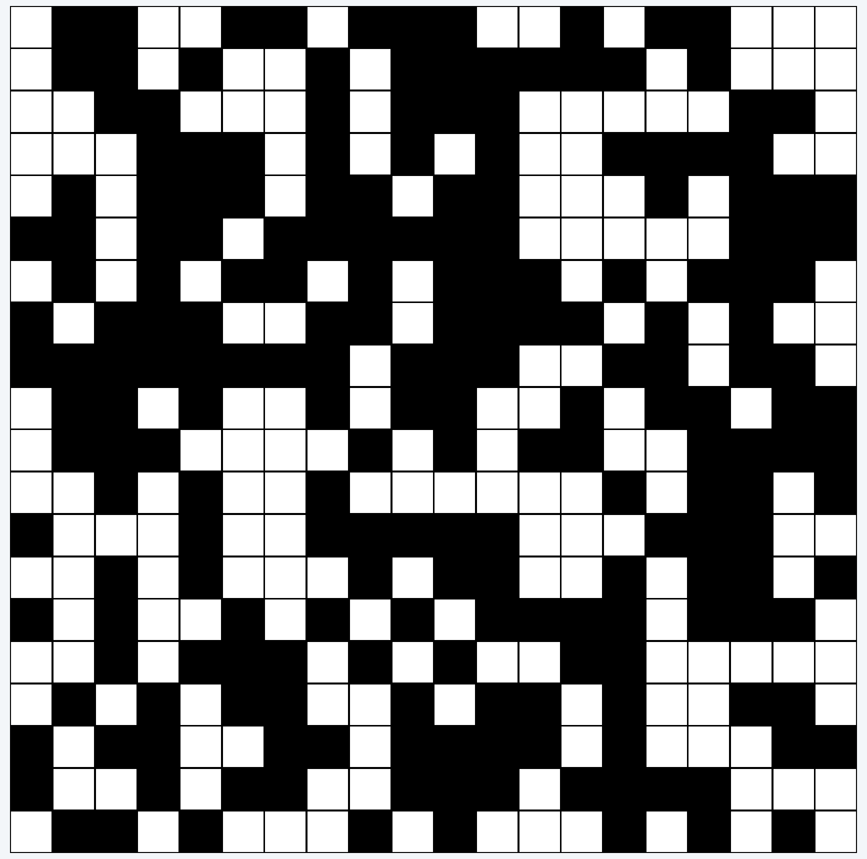
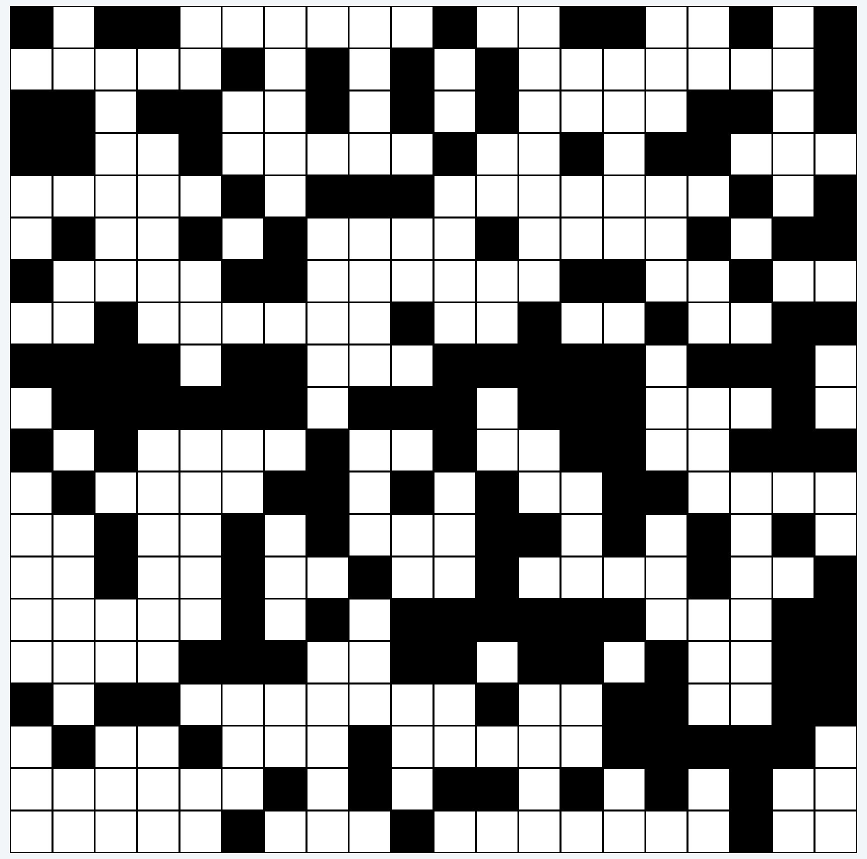
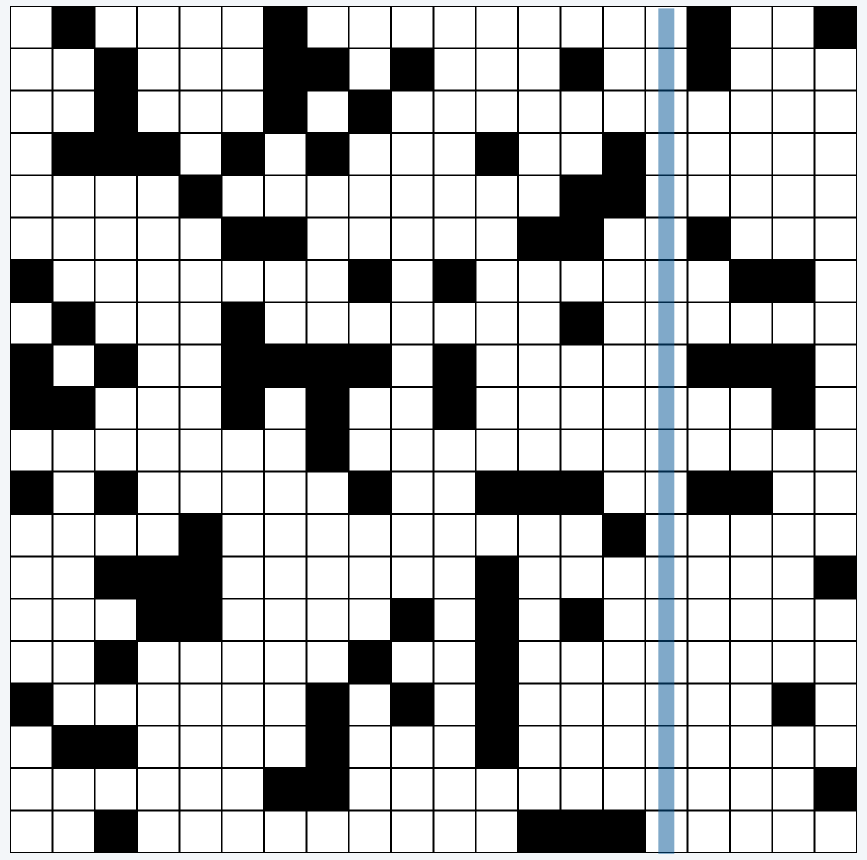


open site

blocked site

# Probability that system percolates

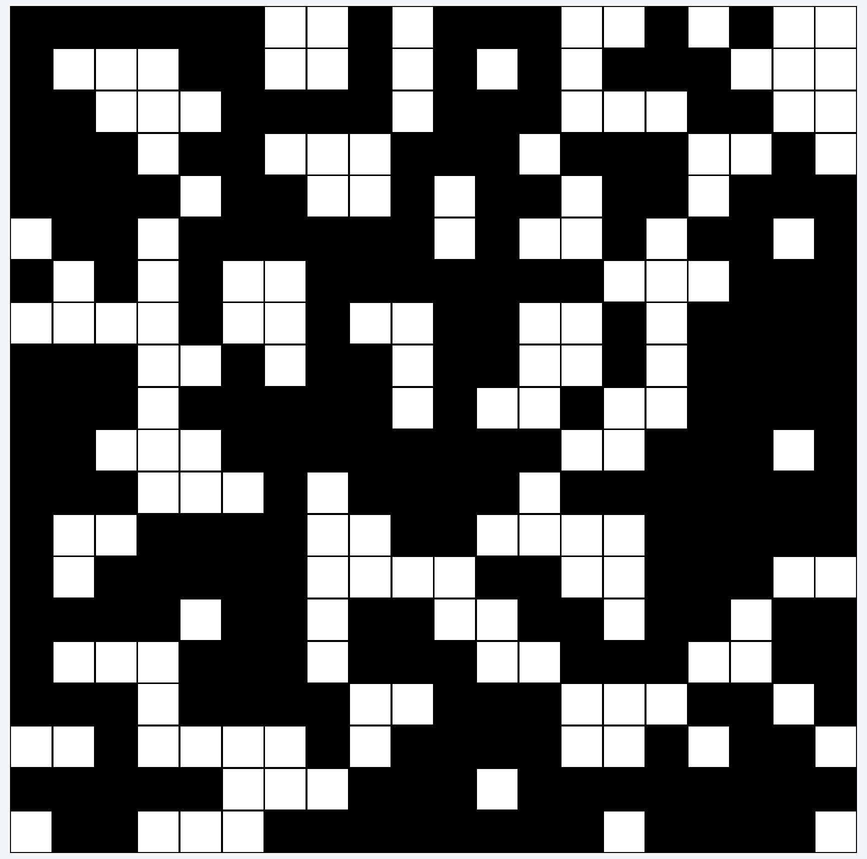
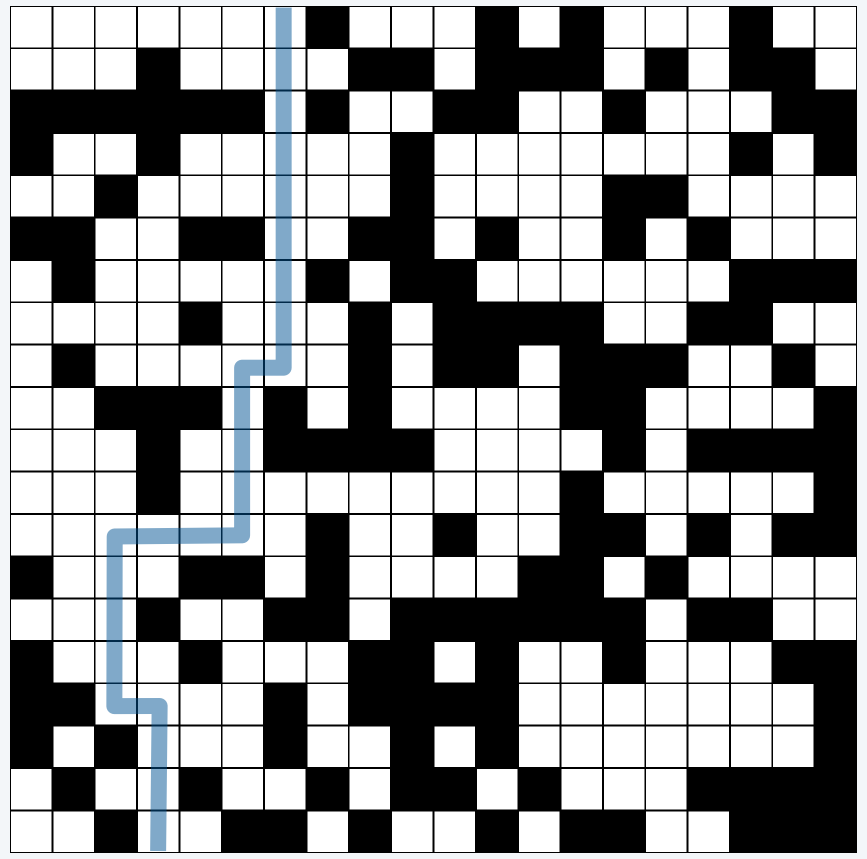
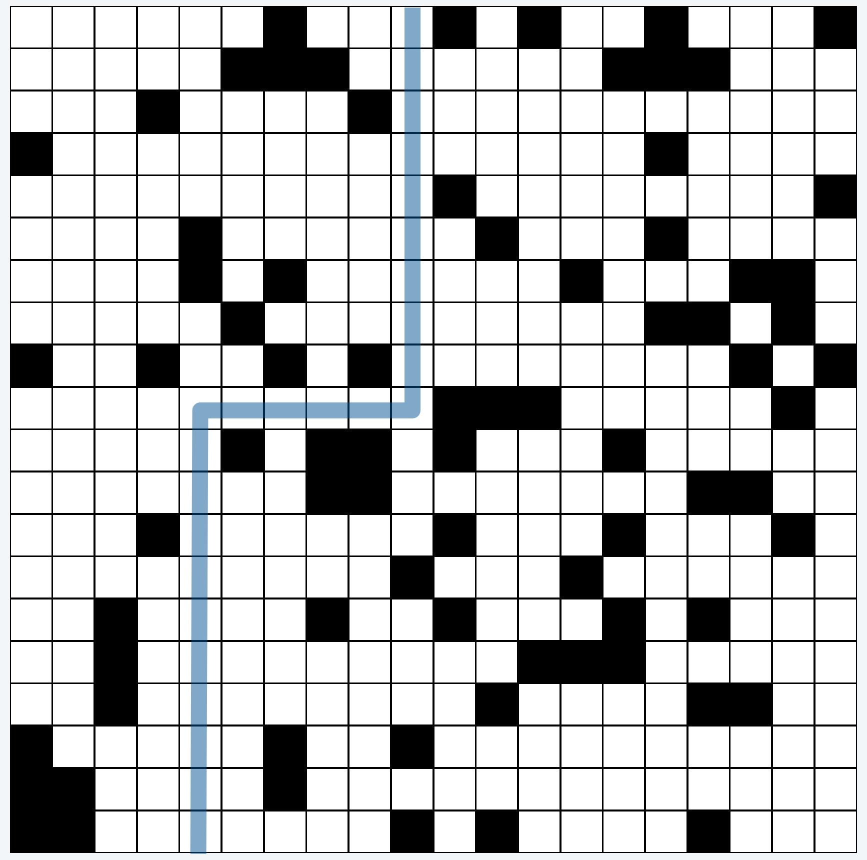
Depends on grid size $n$ and open-site probability $p$.

$p = 0.4$
*(rarely percolate)*

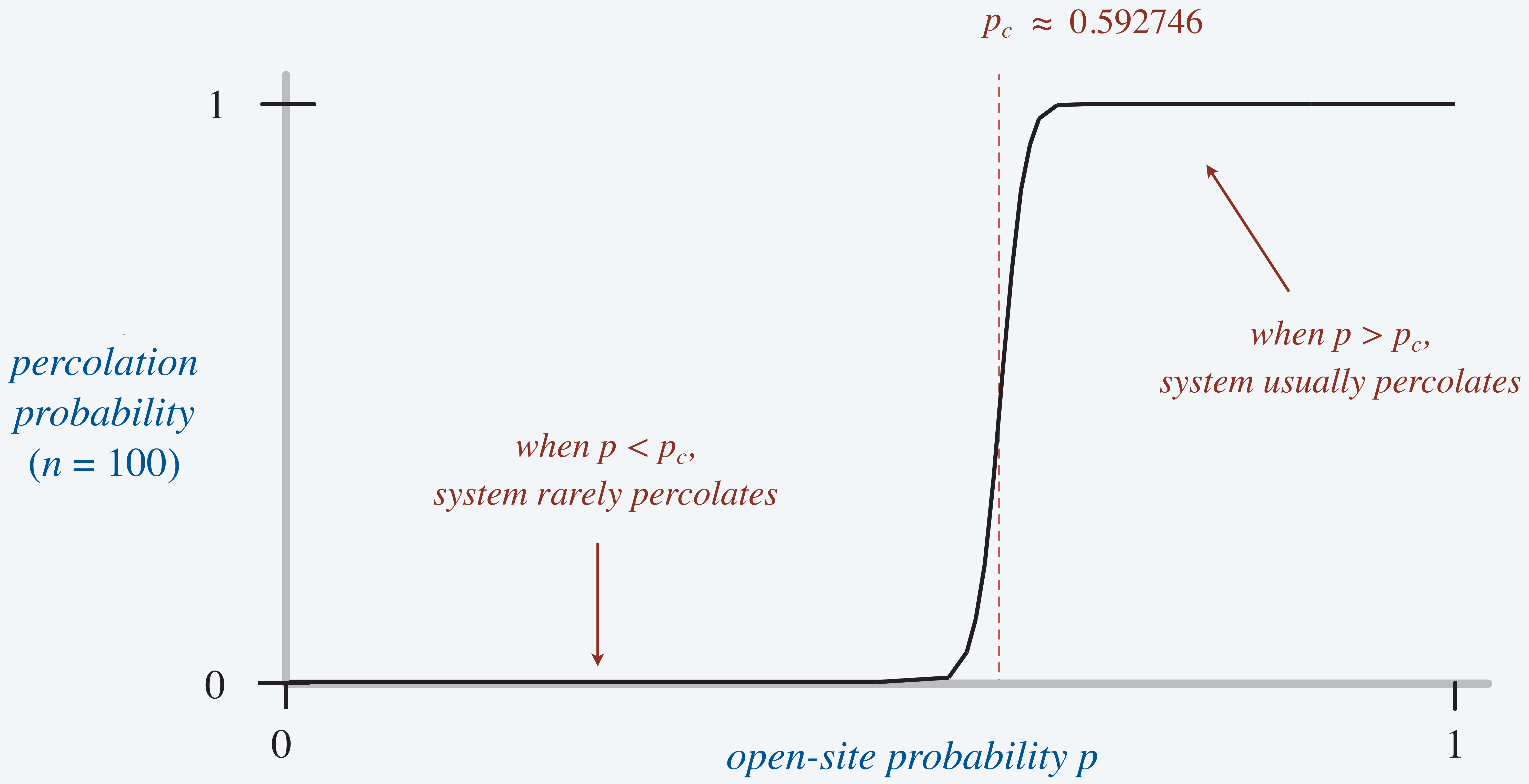
$p = 0.6$

$p = 0.8$
*(usually percolates)*

# Percolation phase transition

**Fact.** As $n \to \infty$, the system's behavior changes sharply at a critical value $p = p_c$.




**Q.** What is the value of $p_c$ ?

**A.** Its exact value is unknown. ⟵ *proven that* $0.556 < p_c < 0.666894$

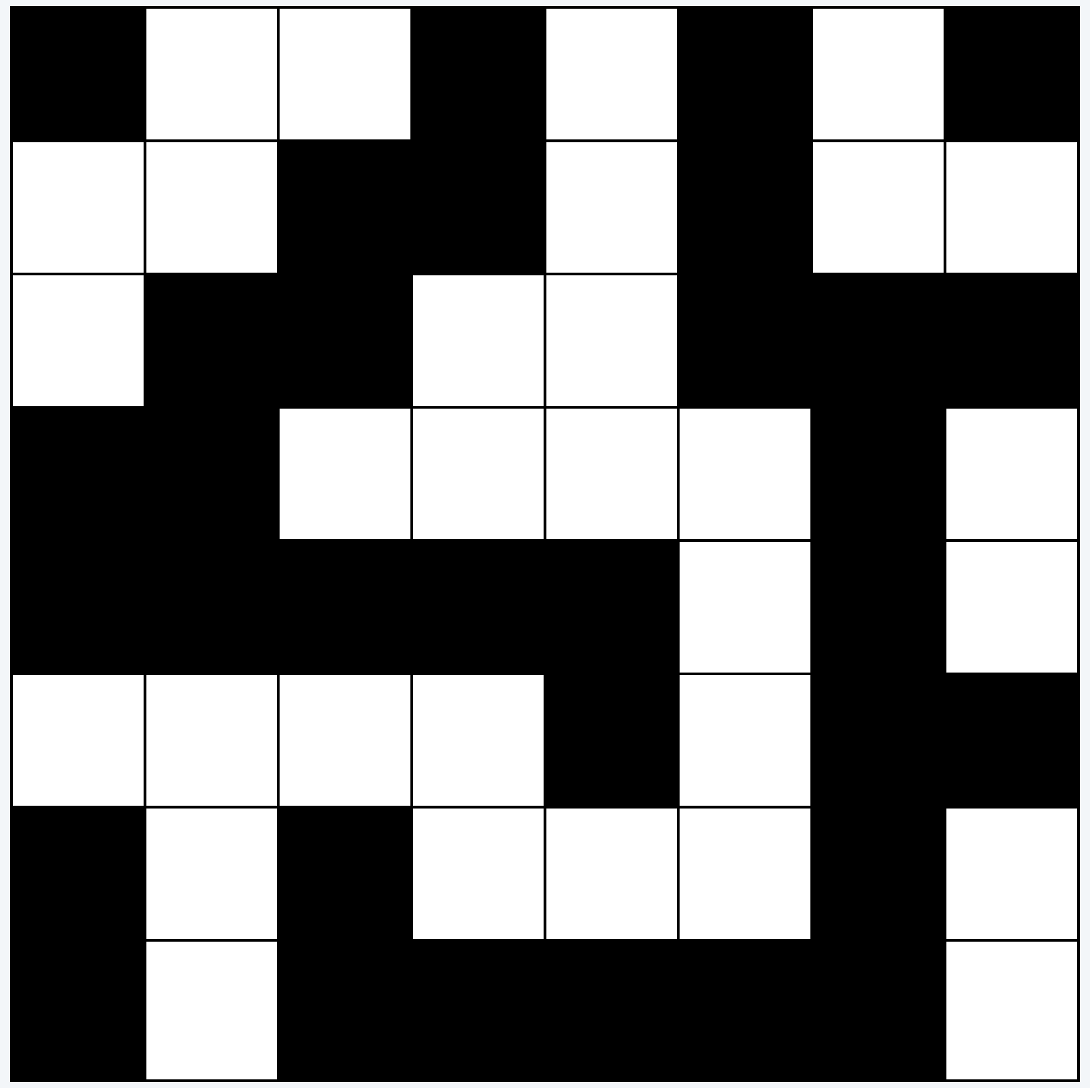
$\Longrightarrow$ Estimate $p_c$ through computational experiments. ⟵ *see Assignment 1*

# Monte Carlo simulation

A computational experiment.

- Initially, all sites in an $n$-by-$n$ grid are blocked.
- Open sites one at a time, uniformly at random, until the system percolates.
- Fraction of opened sites = estimate of percolation threshold $p_c$.

Monte Carlo simulation. Repeat the experiment many times and average the results.



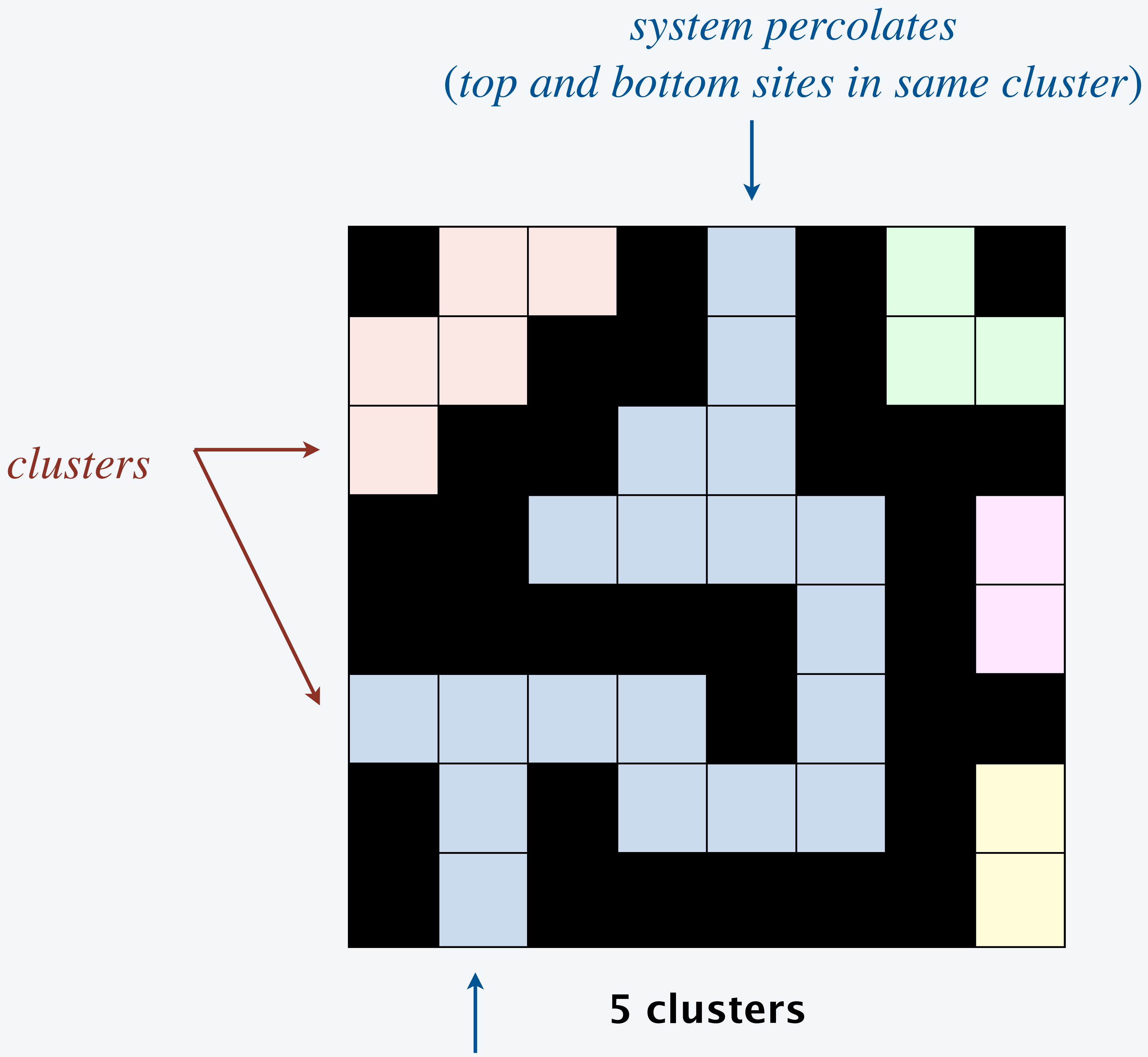
$p_c$ estimate = 31 / 64

Monte-Carlo Casino in Monaco

# Clusters

**Def.** A cluster is a maximal set of open sites such that any two are connected by a path of open sites.

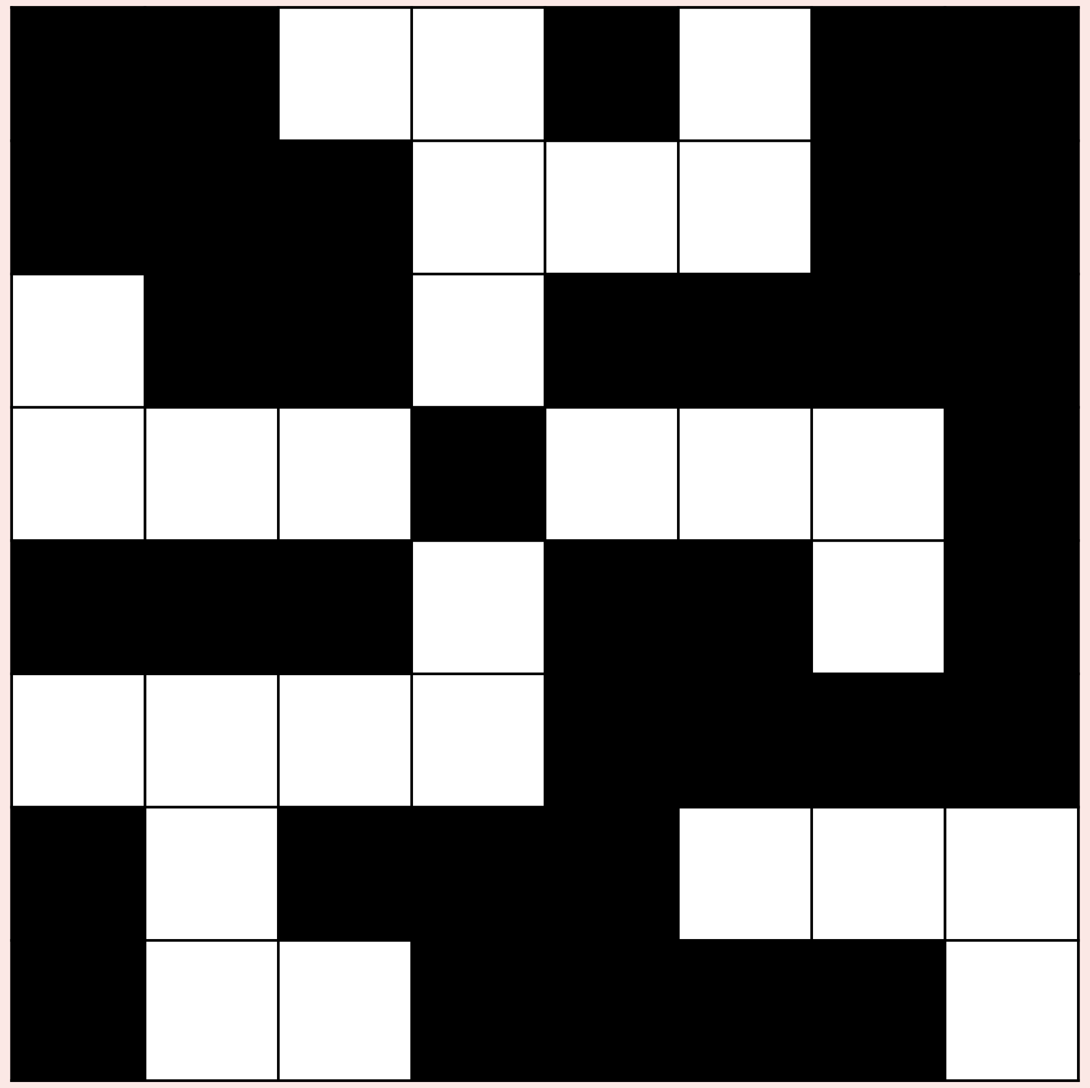
**Note.** System percolates if any site in top row is in the same cluster as any site in bottom row.



# Percolation poll 2

**How many clusters?**

- **A.** 2
- **B.** 3
- **C.** 4
- **D.** 5



# Computing clusters

Def. A cluster is a maximal set of open sites such that any two are connected by a path of open sites.

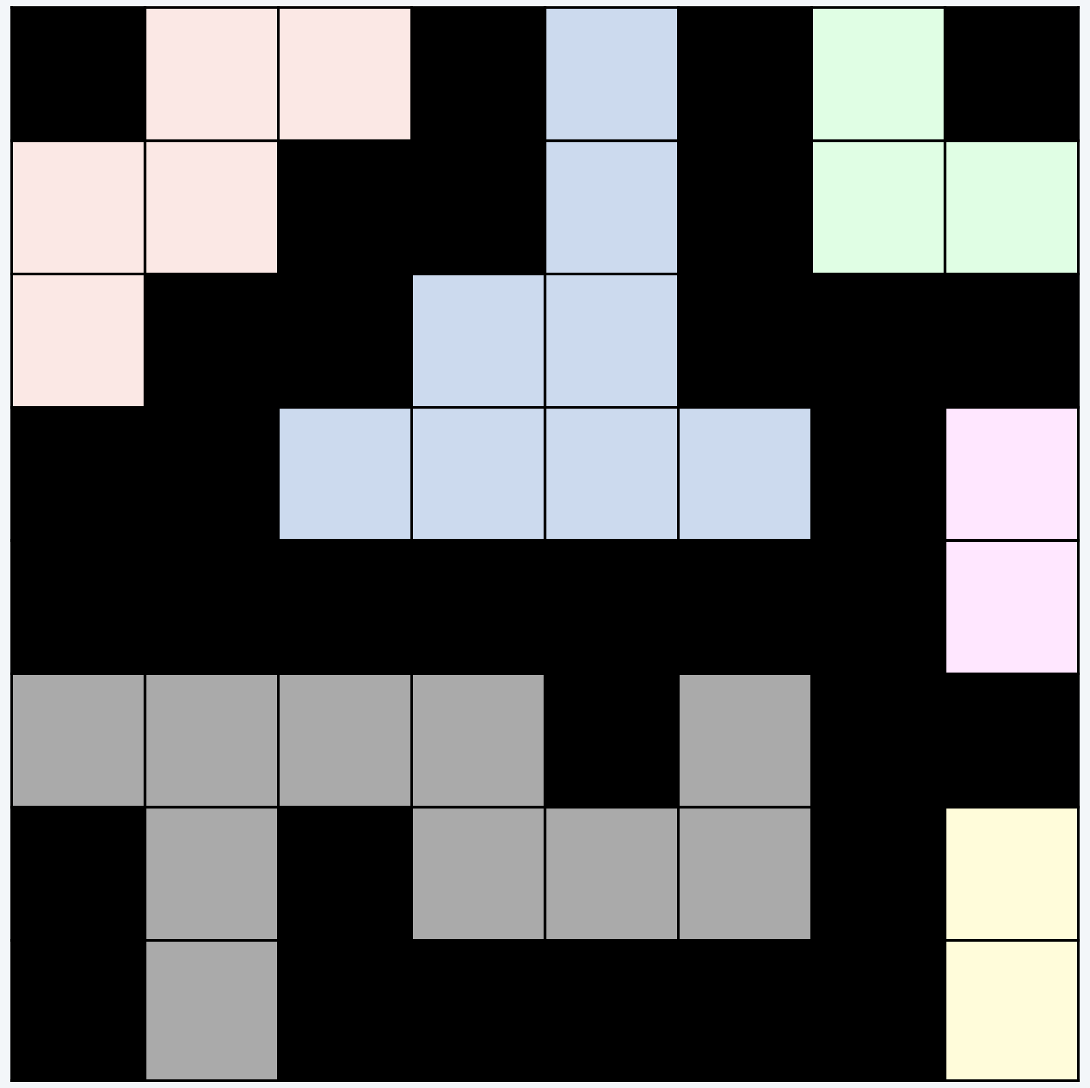
Q. How to identify the clusters?

A. Could run depth-first search.

*recall: blob finding in Atomic assignment*

[ but takes $\Theta(n^2)$ time and must be repeated after opening each site ]

*clusters*



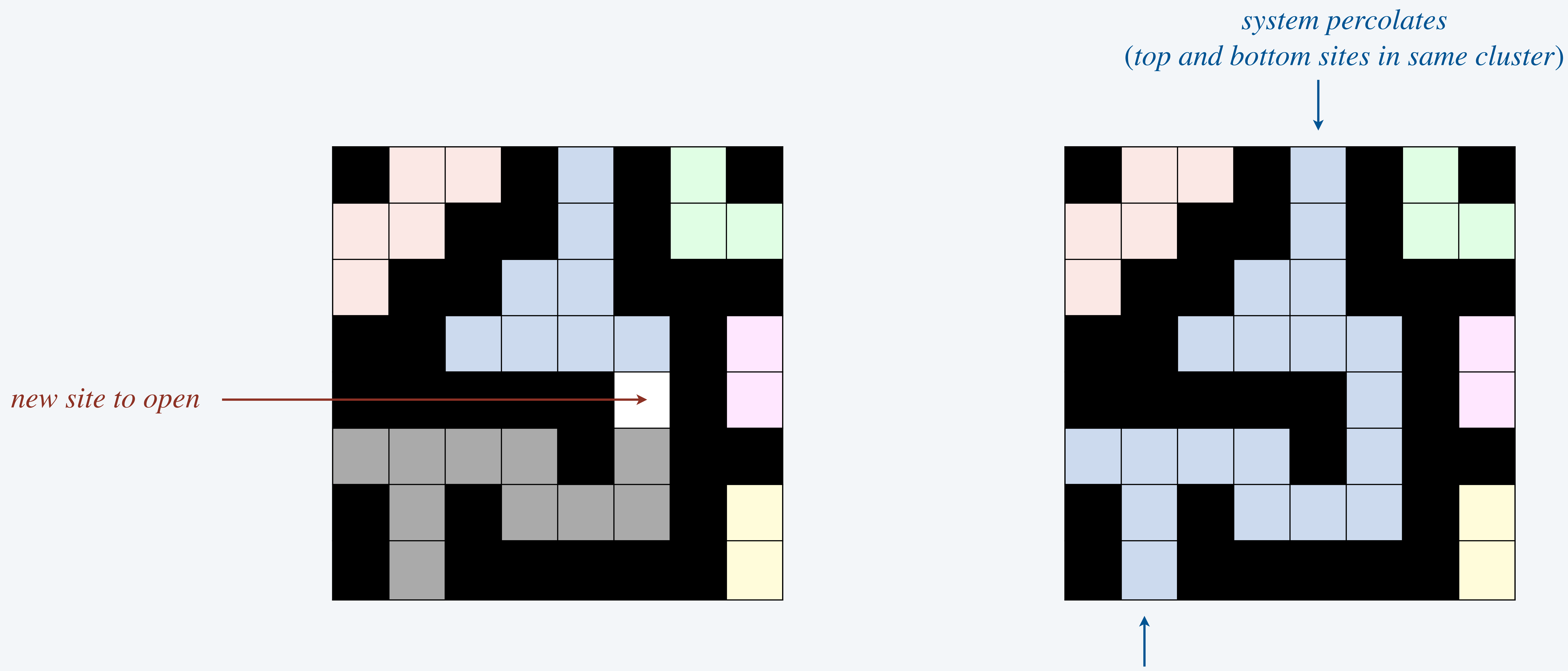
# Maintaining the clusters

Critical observation. When opening a site, only a few clusters can change.

Goal. Devise a data structure to maintain the clusters as sites are opened.

- When a site opens, it initially forms a singleton cluster.
- UNION: merge two clusters.
- FIND: determine the cluster containing a given site.

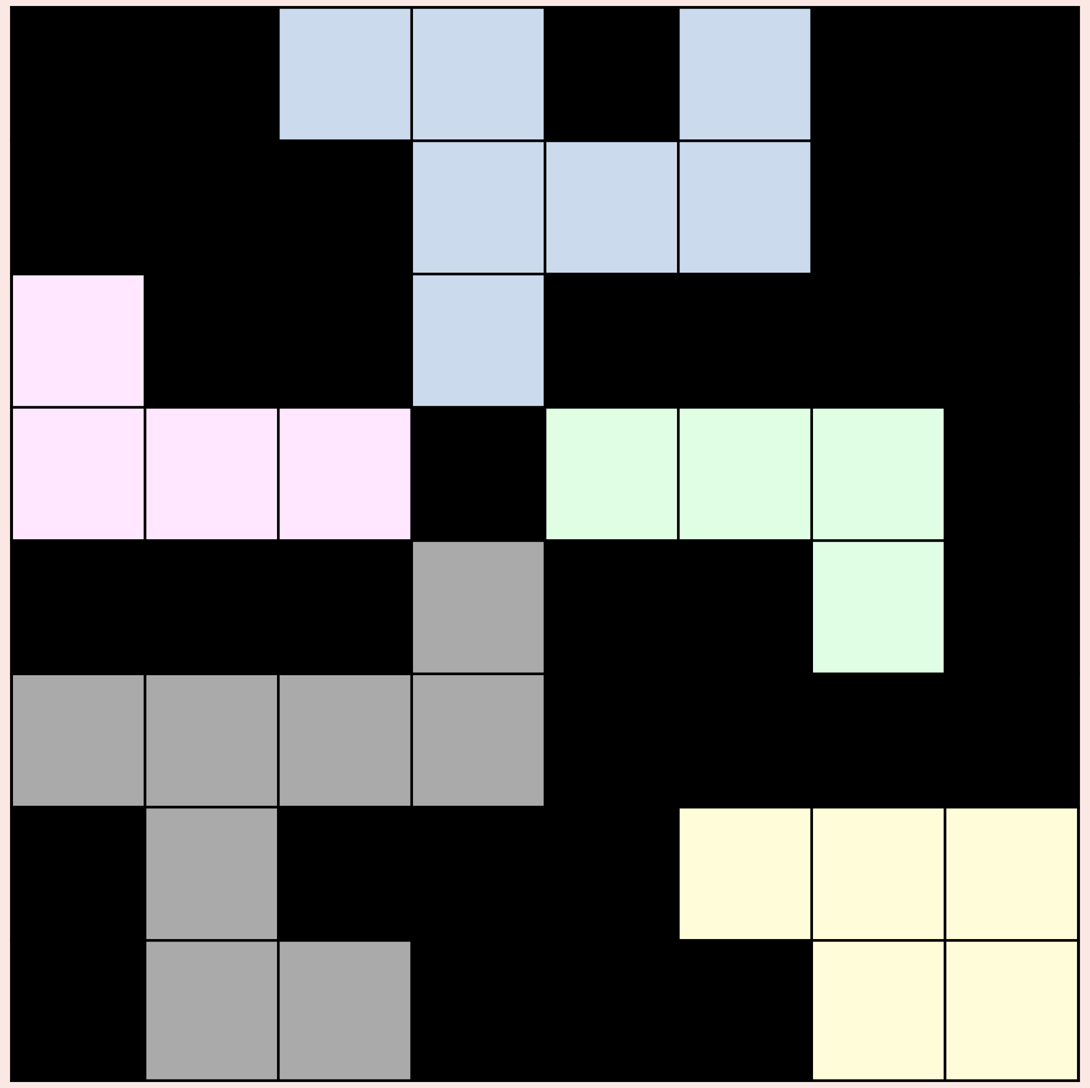
*return a unique id number (or color)*



## Percolation poll 3

**When opening a site in an n-by-n system, what is the maximum number of UNION operations needed to update the clusters?**

- **A.** 1
- **B.** 2
- **C.** 3
- **D.** 4



# Percolation poll 4

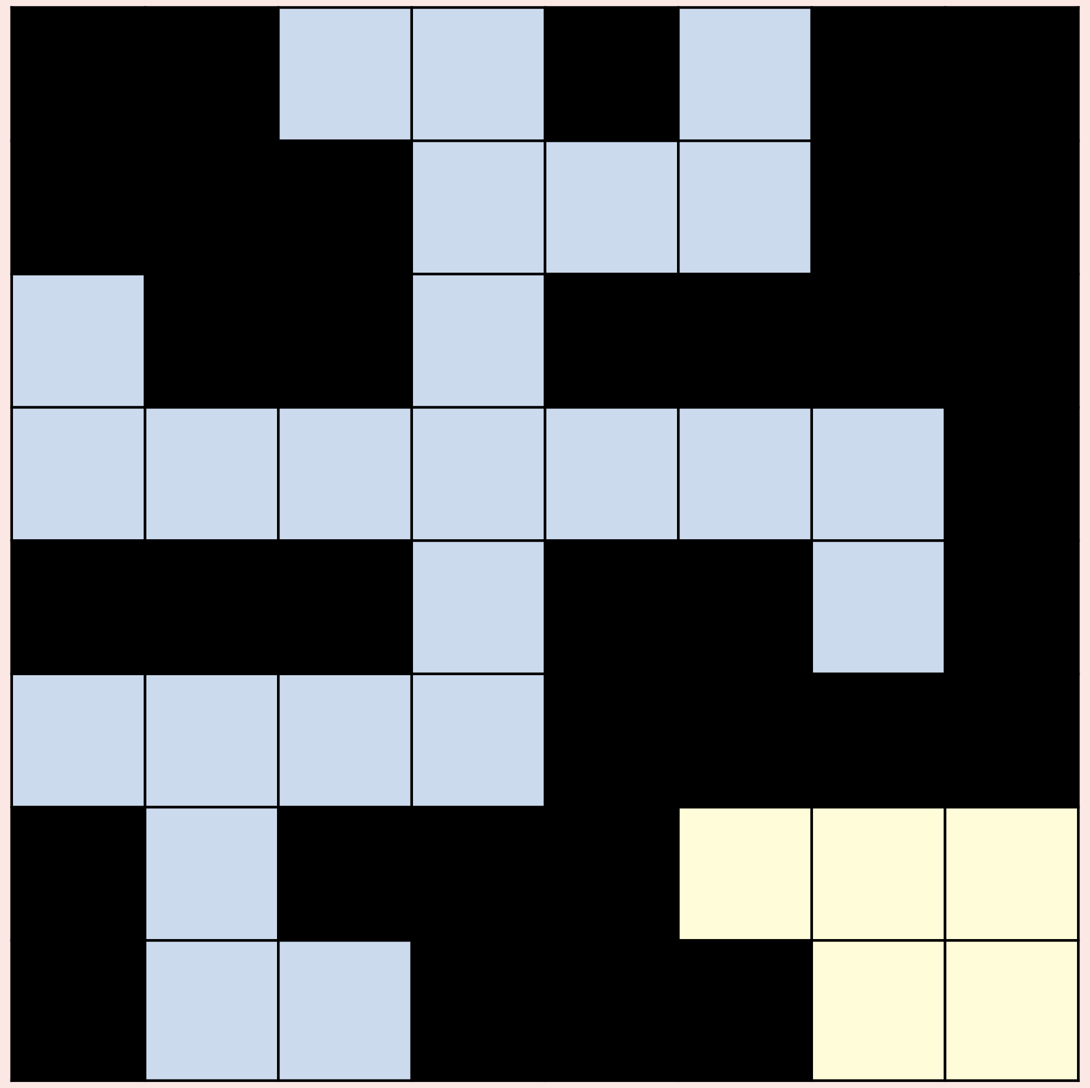
**After opening a new site in an n-by-n system and updating the clusters, how many FIND operations might we need in the worst case to check whether the system percolates?**

**A.** $\Theta(1)$

**B.** $\Theta(n)$

**C.** $\Theta(n \log n)$

**D.** $\Theta(n^2)$

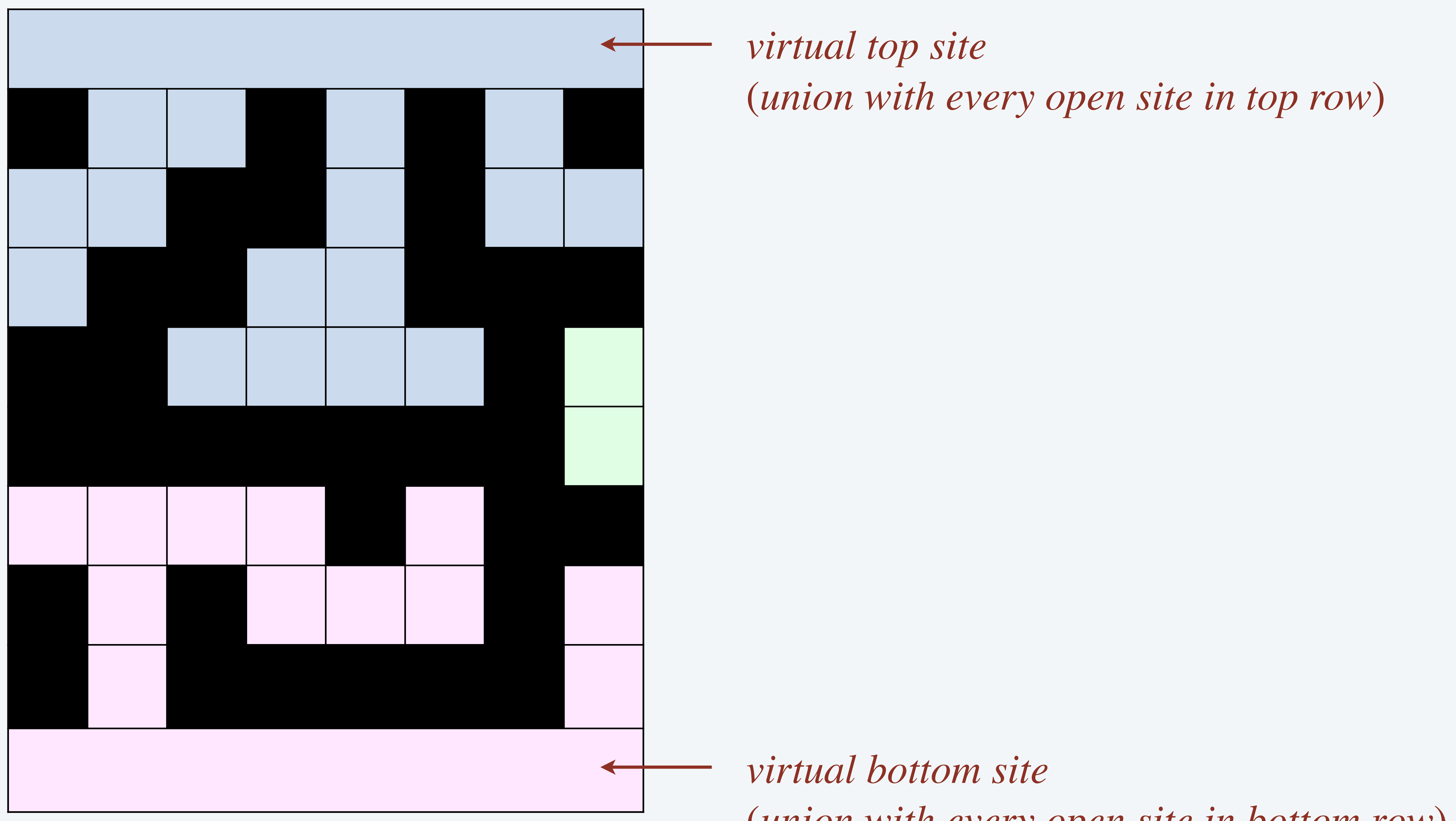


# A clever modeling trick

Key idea. Create "virtual" top and bottom sites.

Observation. The system percolates exactly when the two virtual sites are in the same cluster.

Impact. Now, only two calls to FIND suffice to detect percolation.



# From percolation to union–find

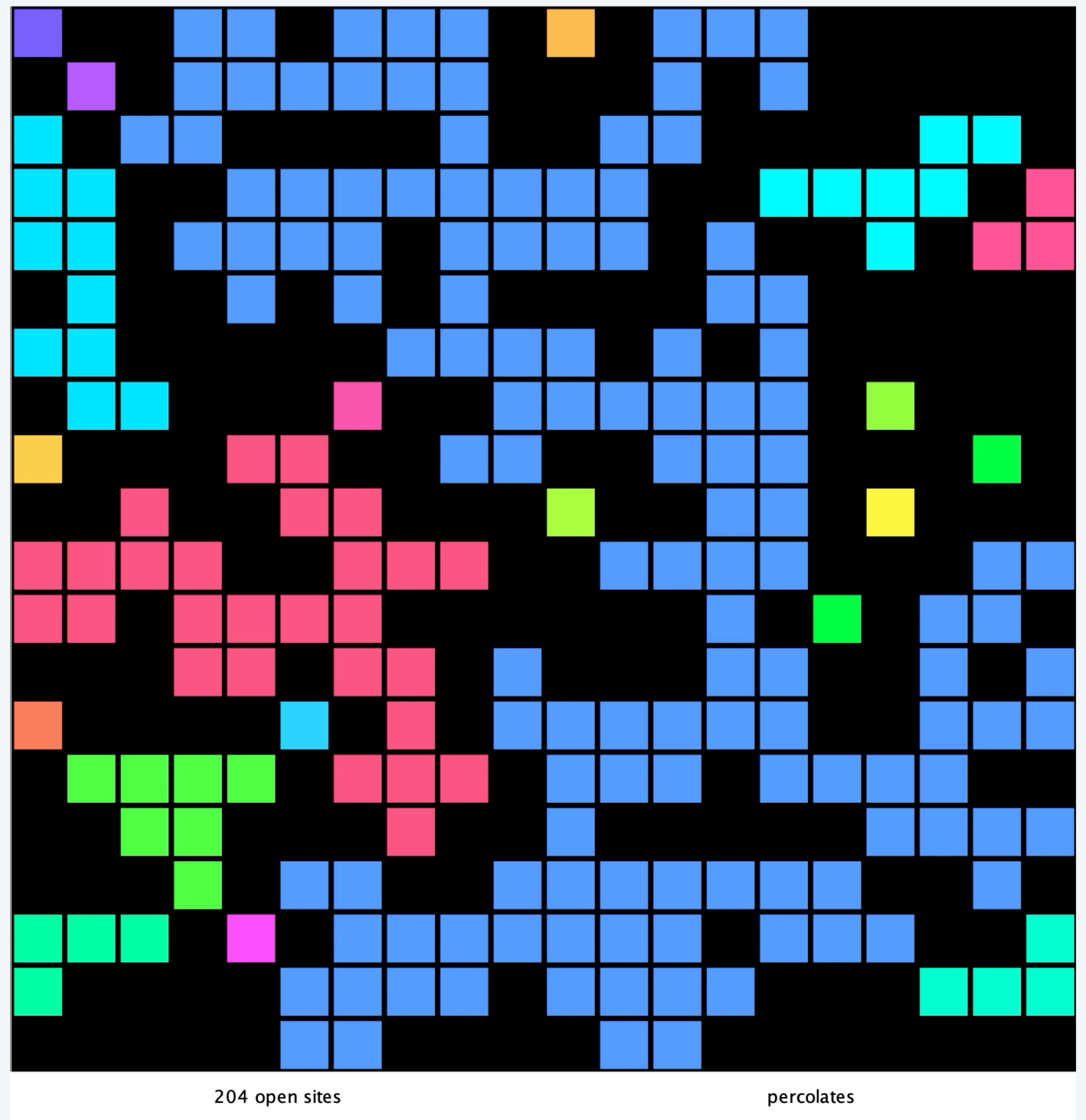
Estimating the percolation threshold. Open sites one at a time and maintain clusters.

With union–find, opening a site involves only $\Theta(1)$ union and find operations:

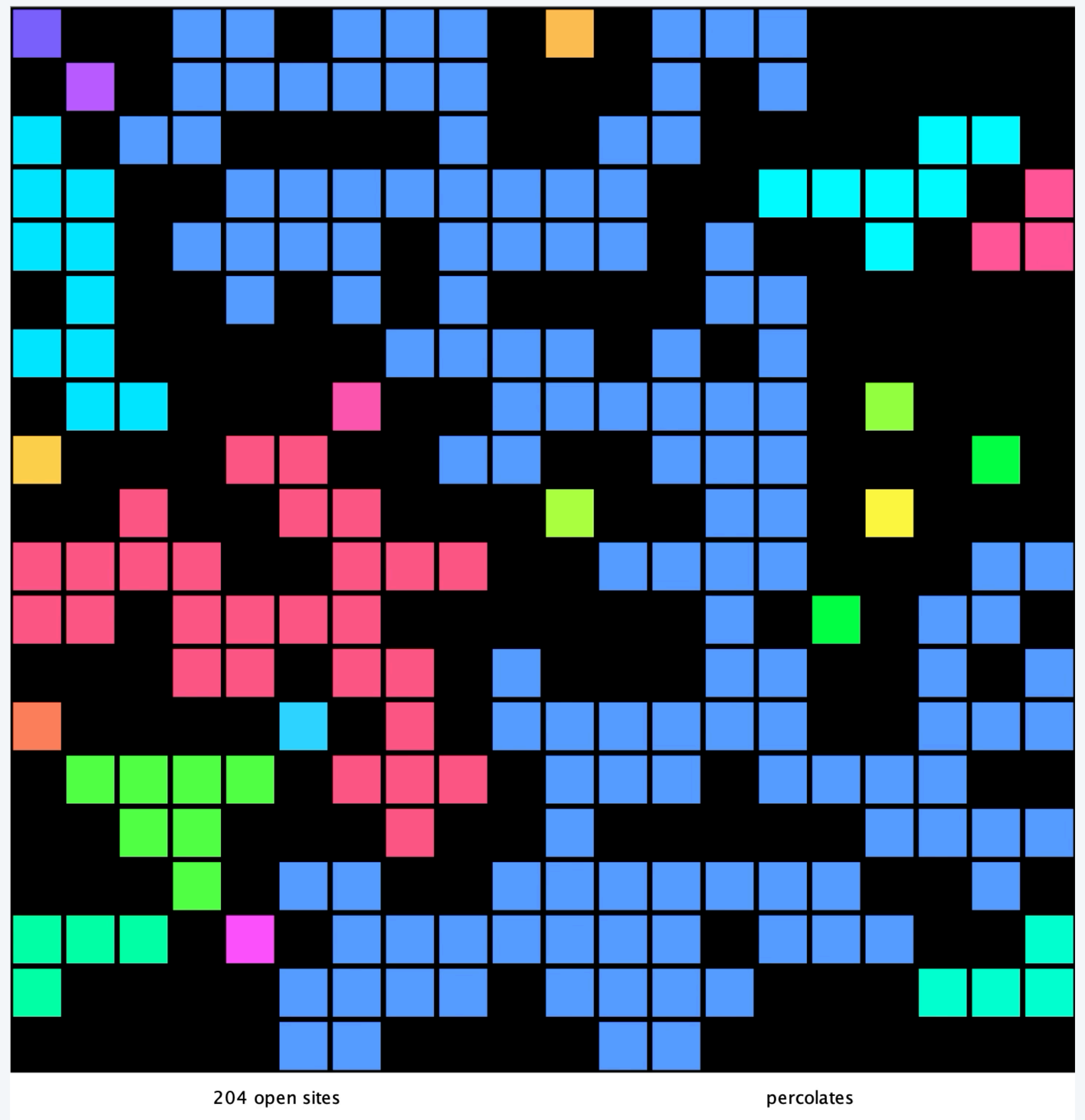
- UNION: merge two clusters.
- FIND: determine the cluster containing a given site.

Next lecture. Design an efficient union–find data structure. ⟵ $\Theta(\log n)$ *per operation, or faster*

Impact. Efficient algorithms enable scientific progress (and much more)!




**brute force**




**ingenious**

# Credits

| image | source | license |
| --- | --- | --- |
| *Monte Carlo Casino* | Wikimedia | CC BY-SA 4.0 |