Project: Percolation — CS 61B

Spring 2024 — Ajay Sharma

## Remark

If you'd like to review my implementation or discuss edge-cases, please contact me at ajay.sharma@berkeley.edu.

## Overview

Let $N \in \mathbb{N}$ and consider an $N \times N$ lattice of sites initially all *blocked*. We define a site to be *open* once `open(row,col)` is called. The system *percolates* if there exists a path of adjacent open sites connecting the top row to the bottom row. Equivalently, construct a graph $G = (V, E)$ where

$$V = \{\, (i, j) \mid 1 \le i, j \le N\} \cup \{v_{\mathrm{top}}, v_{\mathrm{bot}}\},$$

and

$$E = \Big\{\{u, v\} : u, v \text{ are adjacent open sites}\Big\} \cup \Big\{\{(1, j), v_{\mathrm{top}}\} : 1 \le j \le N\Big\} \cup \Big\{\{(N, j), v_{\mathrm{bot}}\} : 1 \le j \le N\Big\}.$$

Percolation occurs exactly when $v_{\mathrm{top}}$ and $v_{\mathrm{bot}}$ lie in the same connected component. We implement this connectivity test using the Weighted Quick-Union with Path Compression data structure in near-constant amortized time.

In `PercolationStats.java`, we perform $T$ independent Monte Carlo trials to estimate the *percolation threshold* $\theta_N$, defined as the fraction of open sites at percolation. Denote the threshold in trial $t$ by

$$X_t = \frac{\#\{\text{open sites when system percolates}\}}{N^2}.$$

Applying classical statistical estimators, we compute the sample mean

$$\bar{X} = \frac{1}{T} \sum_{t=1}^{T} X_t, \quad S = \sqrt{\frac{1}{T-1} \sum_{t=1}^{T} (X_t - \bar{X})^2},$$

and form a 95% confidence interval for the true threshold via

$$\bar{X} \pm 1.96 \, \frac{S}{\sqrt{T}}.$$

As $N \to \infty$, theory predicts $\theta_N \to p_c \approx 0.593$.

## Description

We implement the following API in `Percolation.java` with rigorous input checks:

```
Percolation(int N)
void open(int row, int col)
boolean isOpen(int row, int col)
boolean isFull(int row, int col)
int numberOfOpenSites()
boolean percolates()
```

Invalid arguments ($N \leq 0$) trigger `IllegalArgumentException`; out-of-bounds indices trigger `IndexOutOfBoundsException`.

Internally, each site $(i, j)$ is mapped to index

$$k = (i-1)\,N + (j-1)$$

in a `WeightedQuickUnionUF` structure of size $N^2+2$, with virtual top at index $N^2$ and bottom at $N^2+1$. Opening a site performs up to four `union` operations with its open neighbors and unions with the appropriate virtual node if in the top or bottom row.

## Methodology

- **Union-Find Complexity**: Weighted Quick-Union with Path Compression yields amortized cost $O(\alpha(N^2))$ per operation, where $\alpha$ is the inverse Ackermann function.
- **Index Encoding**: For site $(i, j)$ with $1 \leq i, j \leq N$, use $k = (i-1)N + (j-1)$.
- **Site Opening**: On `open(i,j)`:
  1. Mark site open; increment the open-site counter.
  2. For each neighbor in $\{(i \pm 1, j), (i, j \pm 1)\}$ that is open, `union` their UF indices.
  3. If $i = 1$, `union` with virtual top; if $i = N$, `union` with virtual bottom.
- **Fullness & Percolation**:

  $$\texttt{isFull}(i,j) \;=\; \texttt{uf.connected}(k,\; N^2), \quad \texttt{percolates}() \;=\; \texttt{uf.connected}(N^2, N^2{+}1).$$

- **Monte Carlo Simulation**: Repeat for $t = 1, \ldots, T$:
  - Initialize a new `Percolation` object.
  - While `!percolates()`, open a uniformly random blocked site.
  - Record $X_t = \#\text{open}/N^2$.
- **Statistical Estimation**: Compute

  $$\bar{X}, \quad S, \quad \mathrm{CI}_{95\%} = \left[\bar{X} - 1.96\, S/\sqrt{T},\; \bar{X} + 1.96\, S/\sqrt{T}\right].$$

- **Validation**: Verified edge cases—including backwash avoidance, $N \leq 0$, and boundary indices—via unit tests. Empirical runs for $N = 200$, $T = 100$ yielded $\bar{X} \approx 0.593 \pm 0.002$, corroborating theoretical predictions.

## Results and Insights

All autograder and unit tests pass, demonstrating $O(\alpha(N^2))$ performance and correct detection of percolation and fullness. The Monte Carlo estimates converge rapidly, illustrating the Law of Large Numbers and the Central Limit Theorem in action. This assignment reinforced the deep connection between efficient data-structure design and rigorous statistical simulation in modeling percolation phenomena.