

# Graphic Mode Representation of Random Distributions

*Isolation of Random Distributions Using the Method of Indexing of a Two-Dimensional Matrix*

**Michael Huynh**

*AP Computer Science, Upper Darby High School*

October 2003

Objective: To develop a solution in the C++ programming language to isolate random distributions of cells in a variably sized two dimension matrix.

## 1. BACKGROUND DISCUSSION

Biotechnology has become a very fast moving industry today with heavy research dedicated to formulating new drugs and improving existing ones. These drugs are put through many intricate and tedious tests to determine their effectiveness. In the past, experiments had to be manually conducted slowly for every variation of the drug. Today, researchers can take advantage of the versatility of the computer to simulate these tests. Although the computer simulations are not complete tests that can replace actual experimentation, it does narrow the best possibilities of a drug down to a smaller effective group for further testing. This can save researchers great deal of time and effort.

Antibiotics are substances that inhibit the growth of or destroy microorganisms<sup>1</sup>. The antibiotic affects the bacteria that are directly around its perimeter. More antibiotic does not necessarily mean an increased effectiveness. If the distribution of the antibiotic is clumped together tightly in a single region, the antibiotic will end up conflicting amongst each other and only the antibiotic on the outskirts of the group will fight the bacteria. In drug delivery, the antibiotics are distributed randomly throughout the target area—the technology is not advanced enough to evenly distribute the antibiotics. The random distribution may end with antibiotic clusters that are not favorable. The distribution also depends on the amount of antibiotics distributed within a given region. A 100% distribution of antibiotic throughout a region (fully covering it) will kill all of the bacteria in that region. However, a lot of the antibiotic will be wasted because of neighboring conflicts with adjacent antibiotics. Every other antibiotic area can be removed and the distribution will still have the same effectiveness in destroying bacteria. In practical applications, antibiotics are expensive and drug manufacturers want to use the least amount of antibiotics possible while

maximizing effectiveness. This is balancing cost with performance. This program simulates the population of antibiotics randomly in a region by using a two dimensional matrix. The simulation will have the potential to determine the lowest distribution of antibiotic possible for the highest bacteria elimination.

## 2. PROGRAMMING PROBLEM SOLUTION

In order to create the simulation, a number of factors in real life drug delivery had to be represented with computer code. The bacteria affected area was represented with a two dimensional matrix [p1 26]. This assumed that the bacteria affected a surface and not a volume. The population of antibiotic on the matrix was variably determined [p3 141-143]. A higher percentage of population would result in more antibiotic targets being placed on the matrix. A fifty percent population would ideally result in half of the matrix elements being populated by the antibiotic. Next, the distribution of the antibiotic was randomly generated to parallel the inconsistencies of drug delivery [p4 165-176]<sup>2</sup>. Once the antibiotic was distributed, the matrix was indexed searching for single antibiotic targets that did not touch any surrounding antibiotics [p4-6 195-330]. These single antibiotic targets were the most effective at destroying bacteria since it had the luxury of being surrounded by them. The surrounding area was marked as “cured” in the matrix [p7-8 332-353]. Antibiotic targets that had adjacent antibiotic targets were ignored since they lost their efficiency. After the simulation was completed, a ratio of cured regions to uncured regions was calculated [p8 367-399]. Higher ratios were desired since that would mean the distribution was effective in eliminating bacteria. By varying the population percentage of the antibiotic, it is possible to find a high ratio which keeping a low distribution.

### 3. PSEUDOCODE

```
//Define a matrix of 10 by 10
matrix[10][10];

//Get population percentage
get(populPercentage);

//Out of 10x10 total matrix
//elements find populPercentage of
//10x10
populated=populPercentage*(10*10);

//Fill the matrix randomly with a
//limit of populated
fillMatrix(populated, matrix);

//Loop through the whole matrix
//searching for elements that do
//not have any neighbors. If such
//an element is found, fill in the
//adjacent elements. They will be
//called the "cured" elements.
for(x=0;x<10;x++) {
for(y=0;y<10;y++)
{
if(noNeighbors(matrix, x, y)) {
fillAdj(matrix, x, y);
}
}}

//Count the number of cured or
//filled in elements
cured = countCured(matrix);
//Count the number of unaffected
//elements
unaff = countUnaffected(matrix);

//Calculate the ratio of cured
//elements to unaffected elements
ratio = cured/unaff;
```

### 4. CODE

Actual program code can be found attached to this document.

### 5. VALIDATION

The program was visually inspected for errors. The correct program would populate the matrix according to the defined percentage. Also, the distribution of the population would be random each time the program is executed. Neighbor checking and cured elements filling could be easily verified by determining if the program successfully surrounded the targeted elements with no adjacent

neighbors. This program is believed to be free from errors.

### 5. SUMMARY

The program was successful in simulating antibiotic drug delivery. It provided random scenarios for the distributions and successfully identified efficient elements with no adjacent neighbors and was able to mark those neighbors as cured. Determining the ratio of cured elements to unaffected elements was also accomplished. While this program would be too simple for real world applications, the principles used in this program, such as isolating random distributions, could be used in a commercial program for drug research.

### REFERENCES

1. "Antibiotic." The American Heritage® Dictionary of the English Language, Fourth Ed. 2000.
2. Borland Turbo C++ function help file. 1994.