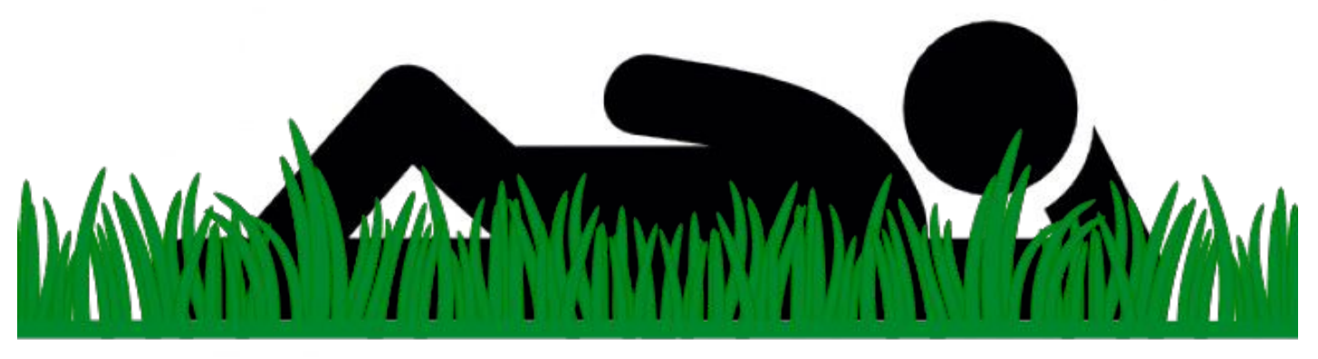
The Patch of Grass Problem



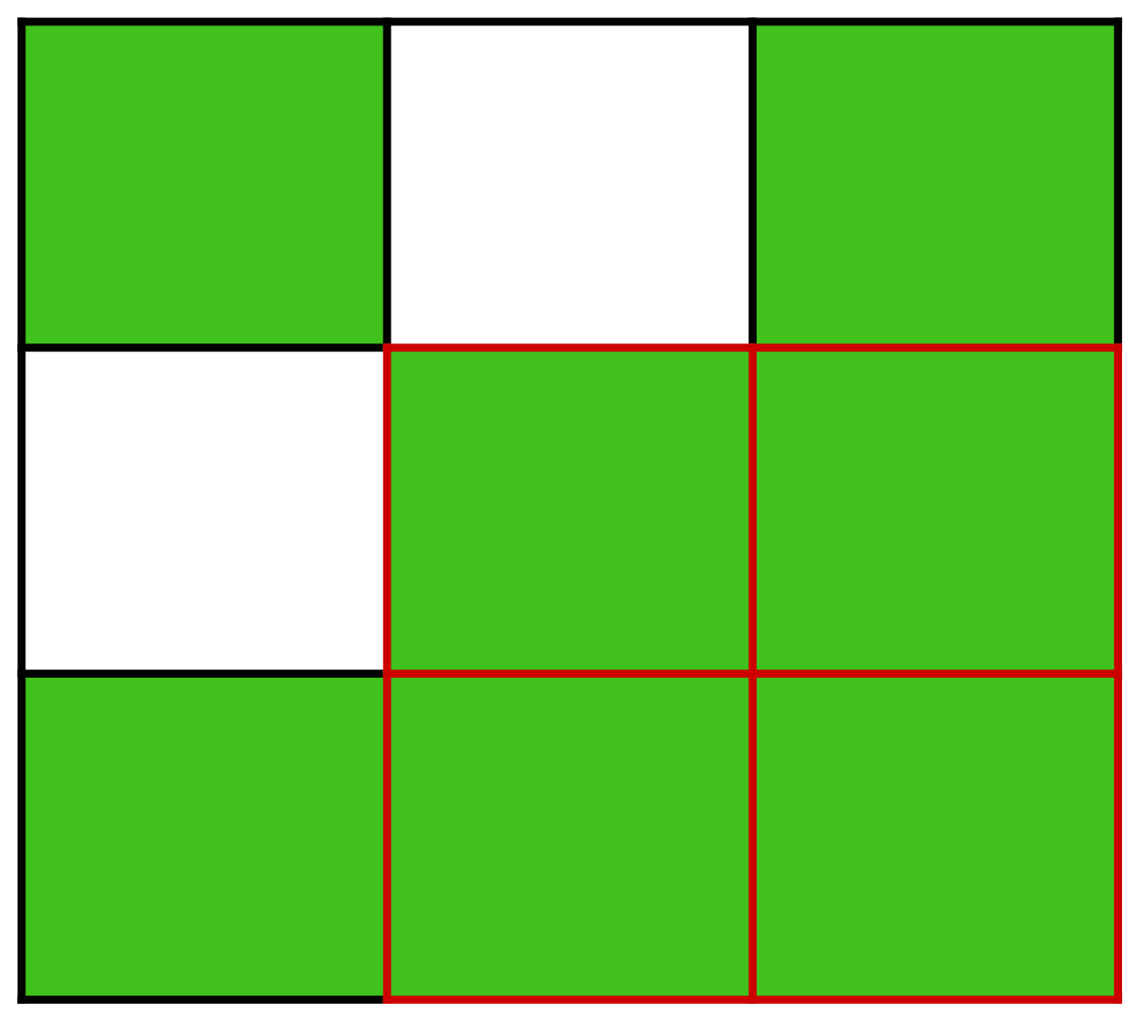
Macalester College, St. Paul MN
Matt Hagen '18 & Steven Roach '18

The Problem

Given a field of grass consisting of adequate and inadequate blades of grass, what is the largest rectangle composed of only adequate blades of grass?

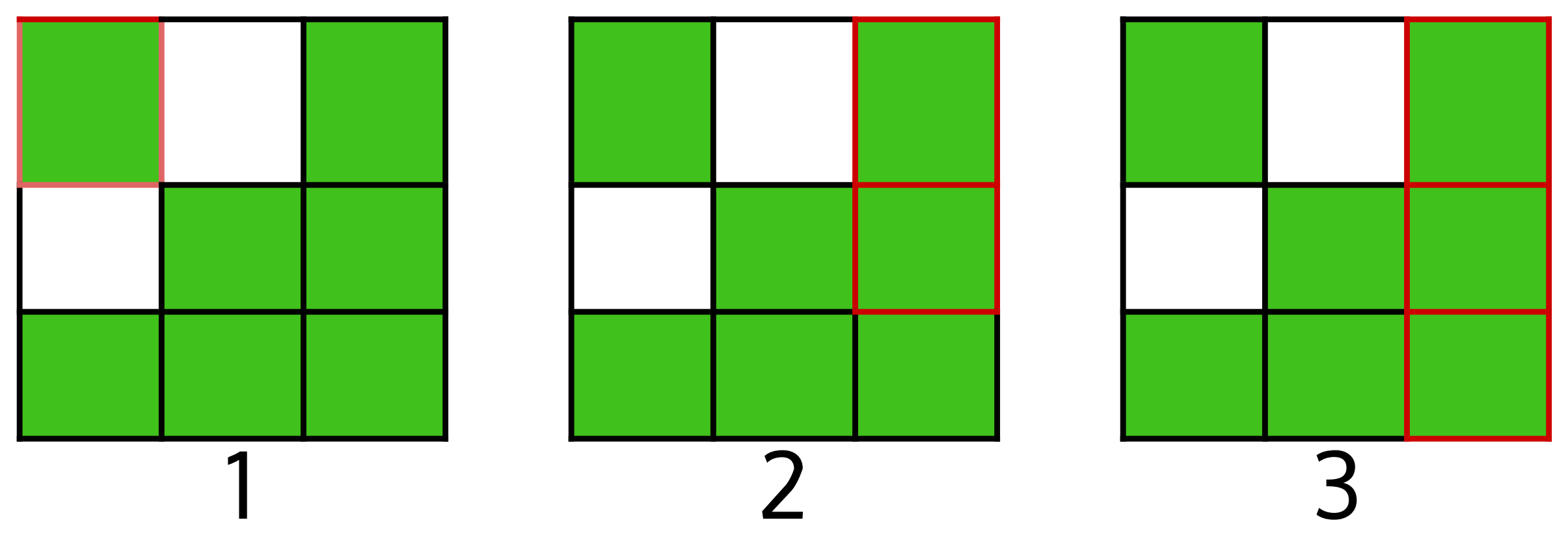
Computation Introduction

The field can be represented as an **M x N** matrix of 1's and 0's. In our visualization of the field matrix, 1's will be represented as 🟩 and the 0's as ⬜. Given a 3 x 3 matrix, we can then visualize the field, with the greatest area rectangle being the 2 x 2 in the lower right corner.



Brute Force Approach

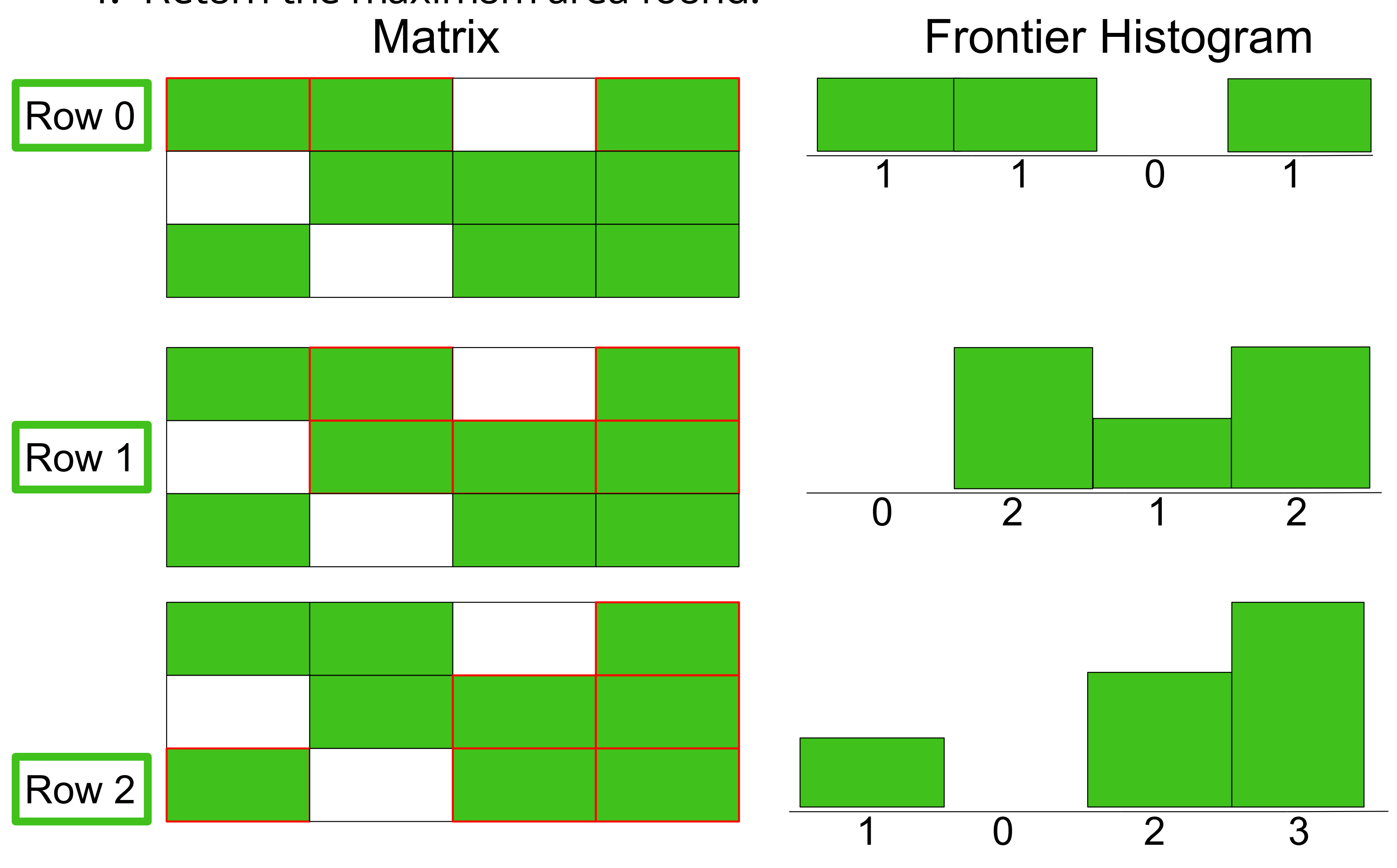
A brute force approach requires that all possible sub-matrices are checked for validity and size against the current largest rectangle. Our implementation is only guaranteed to find the greatest rectangle after 6 nested loops! This is given by θ(M^3 * N^3). As the algorithm progresses through the different sizes, we are able to watch the progression of its largest found rectangle before it finds the optimal 2x2 submatrix.



Dynamic Programming

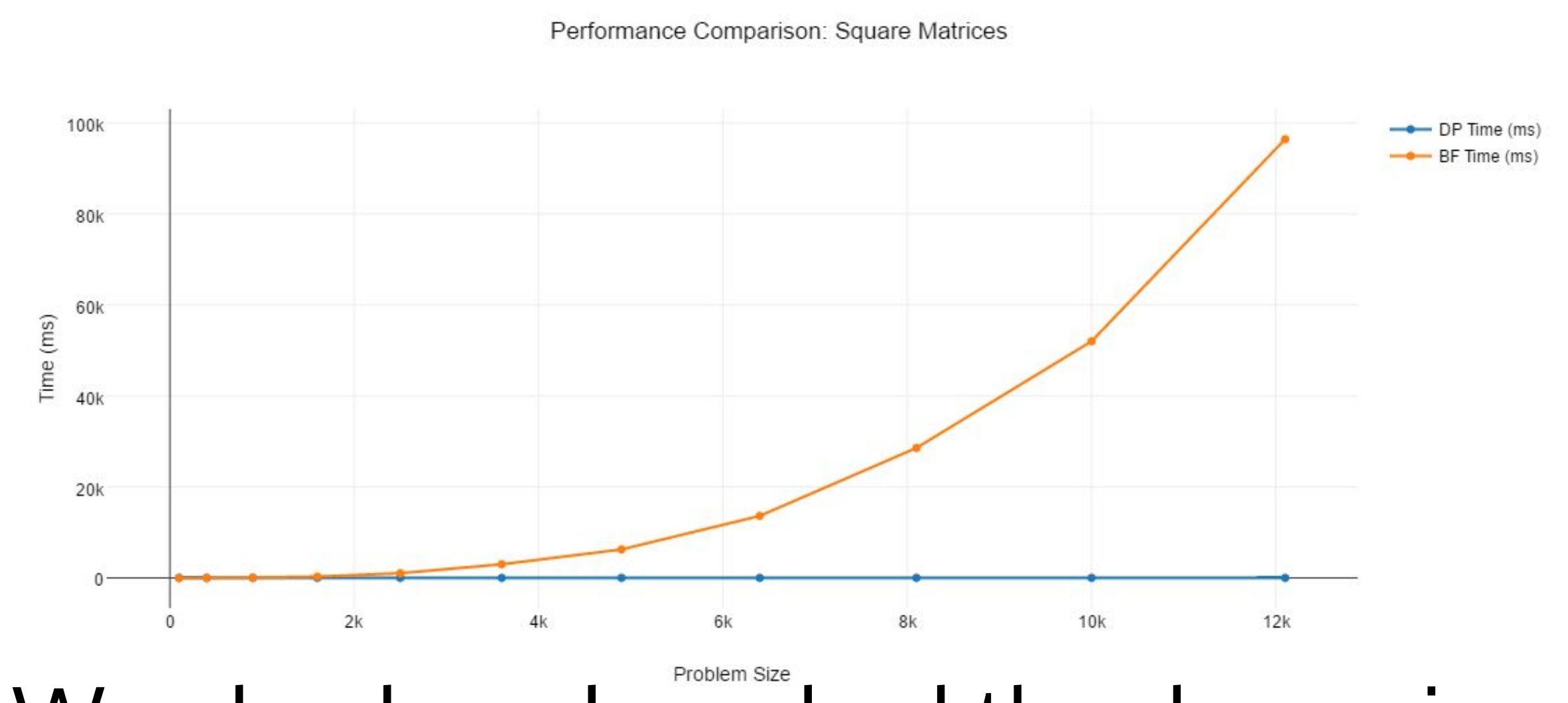
A more efficient algorithm for this problem can be found by using the basic algorithms principles of dynamic programming and transform and conquer. This algorithm works by building a frontier of heights that it updates as it moves down the rows of the matrix. After each row, the algorithm transforms the frontier into a histogram. We then use a known algorithm to find the largest rectangle in the histogram. This histogram algorithm runs in θ (n) time where n is equal to the number bars in the histogram. We call this method once per row of our matrix so the time complexity of the full algorithm is given by θ(M*N).

```
Algorithm: Find the area of the maximum area submatrix consisting of all 1's in a binary matrix.
Input: A matrix represented as a two dimensional array.
Output: The maximum area found.
  1. Initialize frontier as an array with its length equal to the number of columns of the matrix.
  2. Initialize maximum area found to be 0.
  3. For each row in matrix:
       a. For each entry in row:
            i. If the entry is equal to 0, set the corresponding entry in the frontier to be 0.
           ii. If the entry is equal to 1, increment the value of the corresponding entry in the frontier.
       b. Calculate the maximum area rectangle in the histogram of the frontier. This area is equal to the maximum area of a submatrix consisting of all 1's. If necessary, update the maximum area found.
  4. Return the maximum area found.
```



Time Comparison

We implemented both algorithms in Java and benchmarked them to compare their running times. The dynamic programming algorithm is significantly faster for all input sizes.



We also benchmarked the dynamic programming algorithm for inputs of nonsquare matrices to test if it is more efficient on tall or wide matrices.

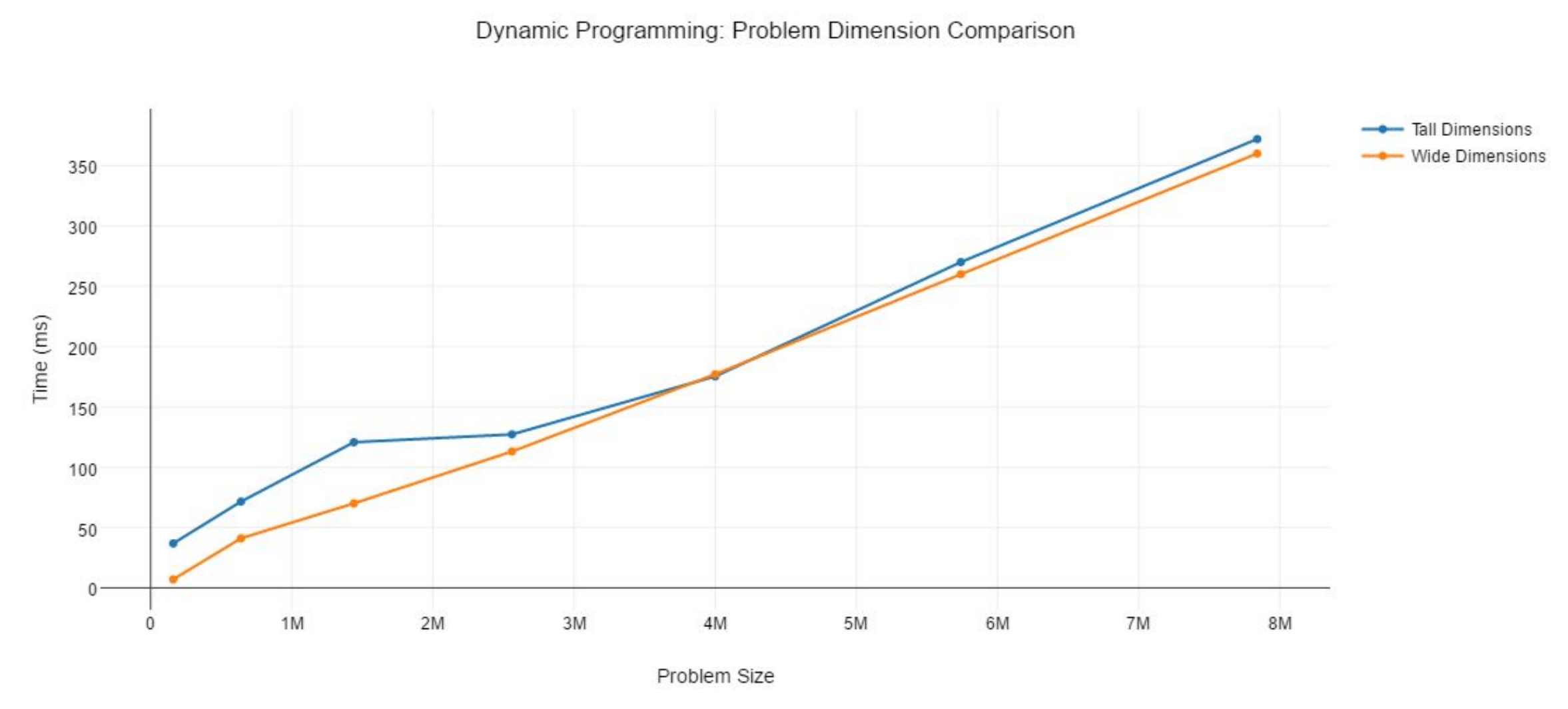


Image Segmentation

We can use variations of the algorithm described previously to segment objects within a picture. First, the picture can be converted to a black and white image represented by a binary matrix. Then we can look for contiguous patches of the same color in the matrix to identify and segment distinct objects. A picture may contain thousands of pixels, which make brute force algorithms unfeasible and dynamic programming solutions well suited for this task.