Problem 8. Finding the Largest Black Square [20 points] 很有啟發性的dp問題

A black-and-white *image* can be represented as a square matrix of size n by n, where each element is either 1 (representing a black pixel) or 0 (representing a white pixel). Design an algorithm that takes such an image as input and returns the size of the largest black square, that is, the largest **square** sub-matrix that contains all 1's.

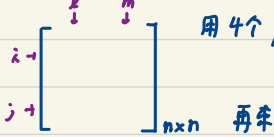
For example, given the following matrix, your algorithm should return 16.

0	1	1	0	1	0
0	1	1	1	1	0
1	1	1	1	1	1
1	1	1	1	1	1
1	1	1	1	1	1
0	1	1	0	1	0

Describe your algorithm, and provide a running time analysis. (You do not need to provide a correctness proof.) For full points, your algorithm should run in $O(n^2)$ time. A slower but correct algorithm will receive partial credit. 暴力解的話是n^4

① Brute force: 給定一 $n \times n$ matrix，窮舉所有 submatrix，判斷是否合法，計算 area

窮舉方式：



用4個 pointer 即可，$\Rightarrow C_2^n \times C_2^n = O(n^4)$

再來判斷是否合法 $\Rightarrow O(n^2)$

$\therefore$ 共為 $O(n^6)$

② DP解：定義 $d[i,j]$ 為 $A[1,\dots,i, 1,\dots,j]$ 的最大那個 square 的值且包含 square (i,j)

$\therefore$ 共有 n^2 個子問題

令 C_{ij} 為 square (i,j) 的值

則：

$$d[i,j] = \begin{cases} C_{i0} & \text{if } j=0 \\ C_{0j} & \text{if } i=0 \\ d[i-1,j-1]+1 & \text{if } d[i-1,j] \wedge d[i-1,j-1] \wedge d[i,j-1] \\ \min(d[i-1,j], d[i,j-1], d[i-1,j-1]) & \text{otherwise} \end{cases}$$

最後再計算 $\max_{i,j}\{d[i,j]\}$ 即可

想過用dp解可能是最佳的，但是不知道怎麼拆成子問題

Solution: Let c_{ij} denote the pixel value at square (i, j) (row i, column j).

Subproblem Definition: Let $DP[i][j]$ denote the side length of the largest all-one square matrix in the upper left corner (rows 1 to i, columns 1 to j) that includes the square (i, j). There are n^2 subproblems total.

之前也有想到類似概念

希望有多點例子能夠印證這個想法

這種dp就不能include當下的點

Recurrence: $DP[i][0] = c_{i0}$, $DP[0][j] = c_{0j}$. When $i, j \geq 1$, $DP[i][j] = min(DP[i-1][j], DP[i-1][j-1], DP[i][j-1]) + 1$ if $c_{ij} = 1$, $DP[i][j] = 0$ if $c_{ij} = 0$. Time per subproblem: $O(1)$.

為什麼是min

Order to solve subproblems: Start from square $(0, 0)$, following the increasing order of $i + j$.

Getting the final result: Find the maximum $DP[i][j]$, square it.

Total time: $O(n^2)$.