求01矩阵中最大的全为0或1的矩形&正方形

悬线法dp

悬线法的用途

针对求给定矩阵中满足某条件的极大矩阵，比如“面积最大的长方形、正方形”“周长最长的矩形等等”。可以满足在时间复杂度为O(M*N)的要求，比一般的枚举高效的多，也易于理解。

悬线法的思路

悬线法，悬线的定义，就是一条竖线，这条竖线要满足上端点在整个矩形上边界或者是一个障碍点。然后以这条悬线进行左右移动，直到移至障碍点或者是矩阵边界，进而确定这条悬线所在的极大矩阵。也就是说，我们要针对矩阵中每个点进行求极大矩阵的操作，所以我们需要Left[]数组存每个点能到达的最右位置，Right[]数组存放每个点能到达的最左位置，Up[]数组位置。设置好这些数组之后，我们开始遍历矩阵中的每个点ves[i,j]，把每个点和上一个点(ves[i-1][j])的Left和Right进行比较，分别取最大和最小，Up则是上一个点的Up+1，进而求出面积进行比较。所以我们可以得到相关的递推公式。
递推公式：Up：Up[i][j] = Up[i-1][j] + 1
Right：min(Right[i][j]，RIght[i-1],[j])
Left:：max(Left[i][j]，Left[i-1][j])

在一个二维01矩阵中找到全为1的最大正方形：
以矩阵中每一个点作为正方形右下角点来处理，而以该点为右下角点的最大边长最多比以它的左方、上方和左上方为右下角的正方形边长多1，所以这时只能取另外三个正方形中最小的正方形边长+1。用d[i][j]表示以i，j坐标为右下角的正方形最大边。
则有状态转移方程：**dp[i][j] = min(dp[i-1][j-1], min(dp[i-1][j], dp[i][j-1])) + 1**
代码：https://paste.ubuntu.com/p/zS5Sn8zBvZ/

现在是矩形的情况：在一个二维01矩阵中找到全为1的面积最大矩形
首先,有一个很显然但很重要的结论,那就是求极大子矩阵肯定要贴着边或一个障碍点,否则就会浪费。根据这个结论,我们可以考虑一种做法我们可以枚举每一个可放置的点。预处理出它与它左边的障碍点(或边界)的距离,也可以得知它上面与下面能扩展到哪里(即无障碍点最多能到哪里)。那这个点能扩出的长方形的最大面积就是它左边的上面与下面能扩展出来的距离的最小值*它到左边障碍点的距离。
代码：https://paste.ubuntu.com/p/KQjNYc4NYx/

单调栈法

与悬线法有相同的地方，首先预处理出每个点可以向上延申多少。枚举矩形的底边所在行，问题转化为：在一个柱状图中求最大矩形面积。这是一个单调栈的经典问题，求解方法为通过单调栈维护某一个点作为区间最低点向两侧可以延申多远（这就是以该点高度为高的最大矩形的宽）。
代码：https://paste.ubuntu.com/p/N9sf955W37/

笛卡尔树法

与单调栈法的思路相同，考虑笛卡尔树的每个节点对应的区间就是该节点作为最小值的区间，那么这个节点对应矩形面积为(h[x]*siz[x])而使用笛卡尔树的优点在于：计算完一行的答案，继续计算下一行时，无

需重新建树，只需对于所有节点高度加1，对于新出现的0节点修改其高度为0。时间复杂度更小为O(R*logN)，R为矩阵高度，N为0的个数,可以完成更大的数据范围。

如：[ZJOI2012]小蓝的好友链接：https://www.luogu.com.cn/problem/P2611

数据范围：$1\le R,C\le 4\times 10^4$，$1\le N\le 10^5$

代码：