BZOJ1018：线段树

这题需要维护连通性，看到有连接删除，很容易直接就想LCT了。然而这题点数20w操作10w，LCT卡常估计过不去。看到这个东西只有两行，考虑能否用魔改后的线性数据结构去维护。我想到了线段树。

考虑如果两个点相连，能有几种情况。有一种是两个点直接经过中间的路径相连，这个满足合并性，很容易维护。然后就是某一个点(或两个点)从两边绕了一下，由上到下或由下到上，然后走中间了路径相连的情况。

(借用官方的一张图)

对于第二种情况，考虑它应该是是什么样子的。注意这张图总共就两行，那么这个东西一定是从上面一行走横着的边到某一个位置，走一条竖着的边，然后再到下面连续走横着的边。

所以，我们对于某一个位置能否到达其对应位置，只需要维护其横向能到达的最远位置，以及这两个位置之间有没有纵向边即可。

确定位置只需要维护横向连通性，然后线段树二分即可。

横向连通性满足合并性，总向边可以用数量求和，均可以用线段树维护。

于是此题得解。

关于实现，我们定义每个区间保存一个Node，其中f[0/1][0/1]表示区间左边的上、下能否到区间右边的上下(0上1下)，维护linked[0/1]表示区间(0上1下)是否左右全部联通，同时维护sum表示这个区间纵向边数量的和。

对于每一个位置，维护ver表示是否有纵向边，hor[0/1]表示从位置i有没有到位置i+1的横向边(0上1下)。

合并的话节点直接用左右状态判，和很容易转移。

查询位置的线段树二分，无非就是先向上走再向下走，自行脑补一发即可(不会看代码)。

最终判定的时候用了一下状压，仅能判定上面的点用1，仅能判定下面用2，如果上下联通，则均可判定，用3来表示。

最后上代码：