



ソリティア

@sortreew

問題概要

- $3 \times N$ マスの盤面があり、いくつかのマ스에コマが置かれている。 ($N \leq 2000$)
- 左右両方もしくは上下両方にコマが置かれている、コマのおかれていないマ스에コマを置く。
- この操作をコマが置けなくなるまで繰り返す。
- すべてのコマを置くまでの手順は何通り存在するか。

例

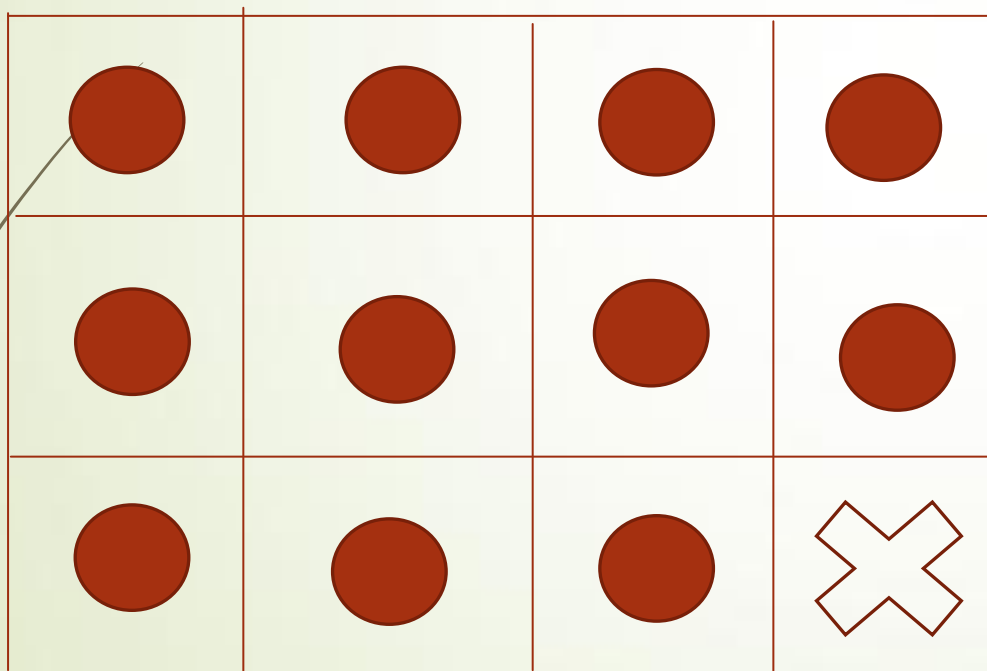
●	●	1	●	5	●	●	●
●	3	2	●	6	7	8	●
●	4	●	10	●	●	9	●

置ける場所

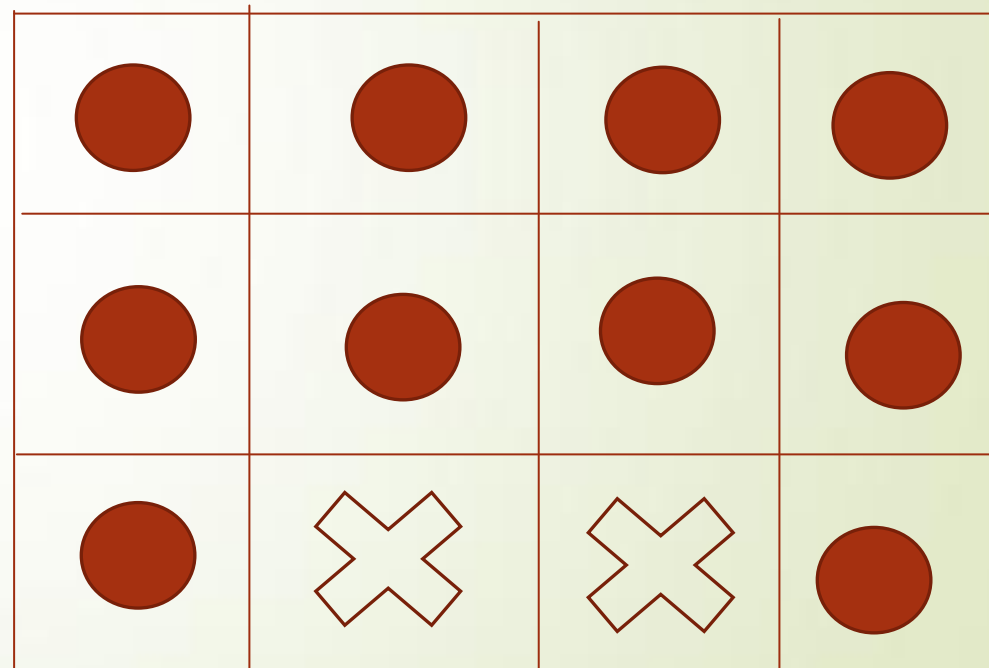
●	○	●	×
○	×	×	×
●	×	×	●

0通りの場合


→ すべての石を置けない場合が存在！



四隅に空白



1 or 3行目に横に連続した空白



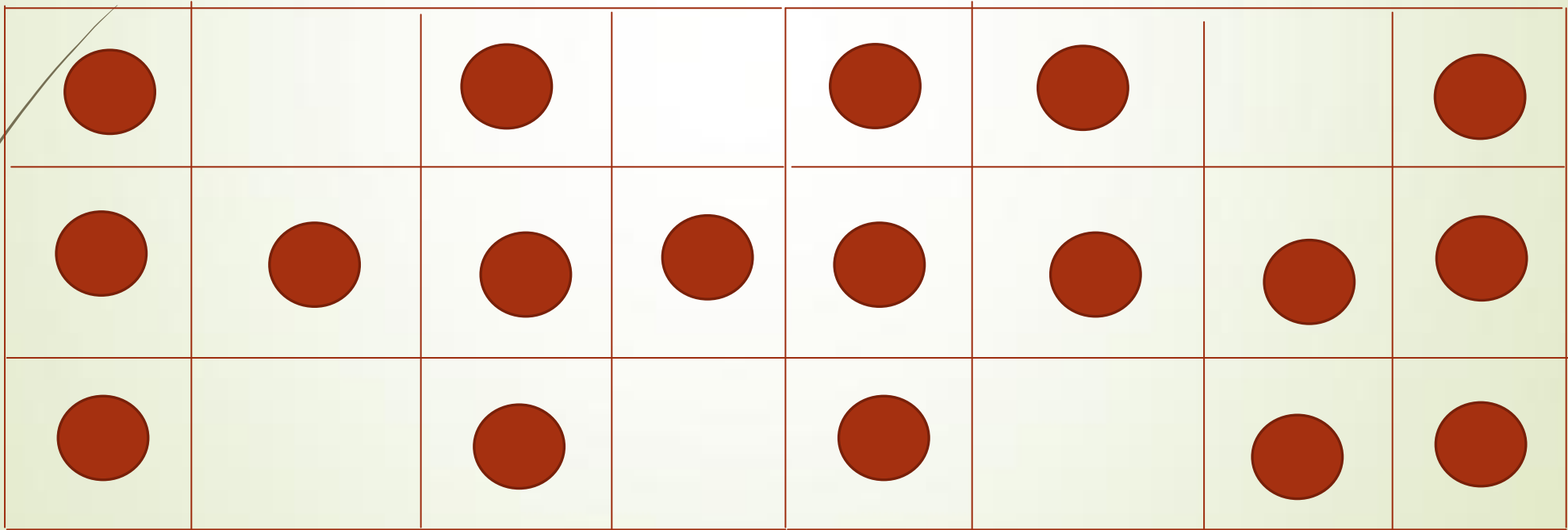
0通りの場合

これらの場合を取り除いて考える。



ちよい考察

0通りの場合でないとき、
1行目および3行目の空白の左右はコマが置かれている
(それはそう)



小課題1


▶ 空白が16個以下

盤面の状態は高々 $2^{16}=65536$ 通り

というわけでDP[盤面の状態]でbitDPすれば間に合う

$O(N \cdot 2^N)$ でも間に合うので余裕

全探索は $16!$ がきびしいのできびしい



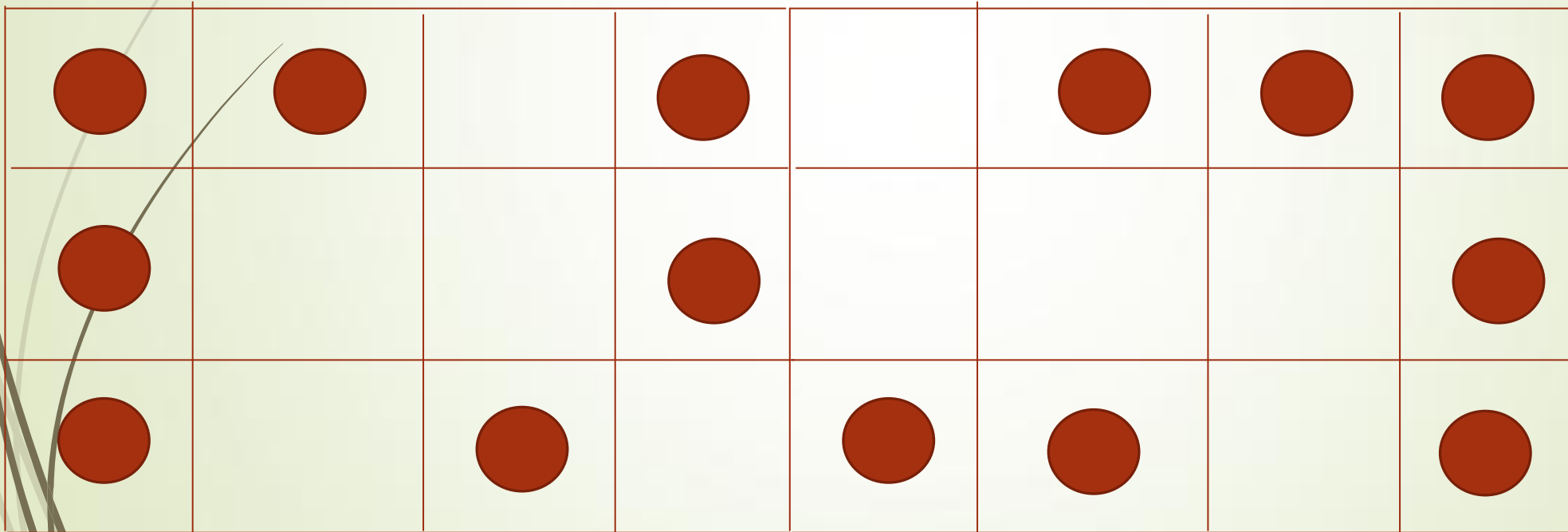
小課題2のための考察


連結でない空白は置けるかどうかに関与しない



小課題2のための考察

連結でない空白は置けるかどうかに関与しない





小課題2のための考察

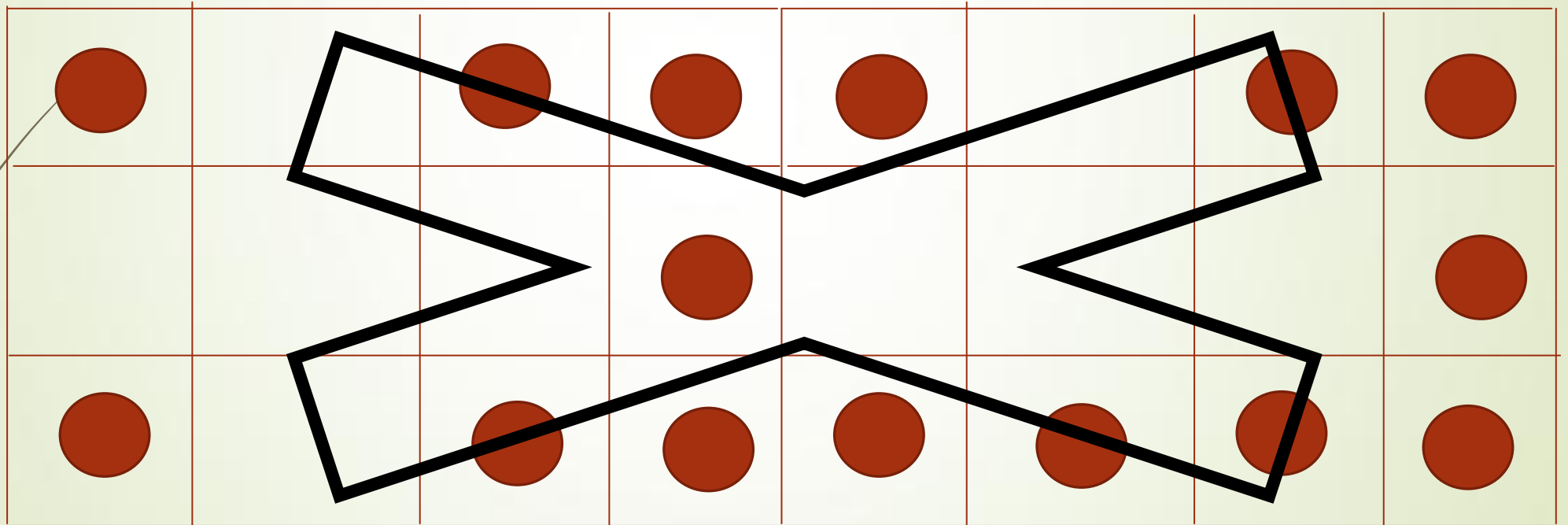
連結でない空白は置けるかどうかに関与しない

なので、それぞれの連結した空白について考えられる
(答えはさておき)

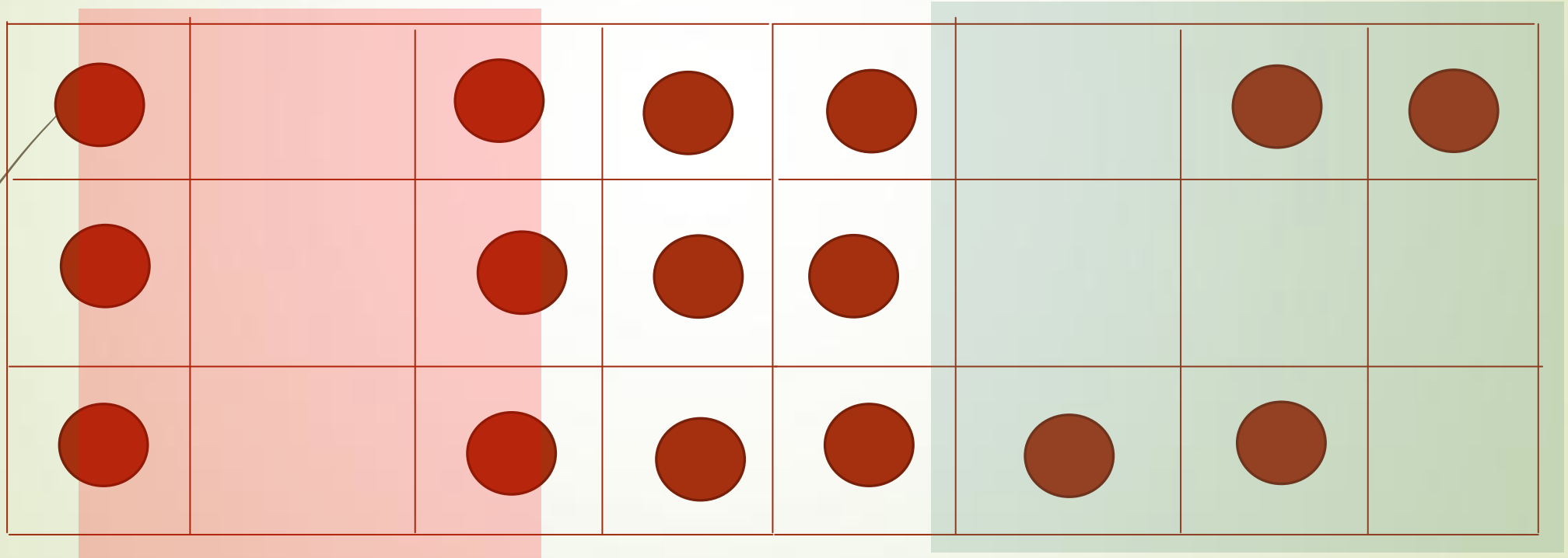
あと、石を置いても他に影響が出ない場所は連結して
いないものとする

小課題2

- ▶ 小課題2では空白に隣接する空白は高々2つ



小課題2でありうるパターン



小課題2でありうるパターン

すごい広いように見えるが・・・？

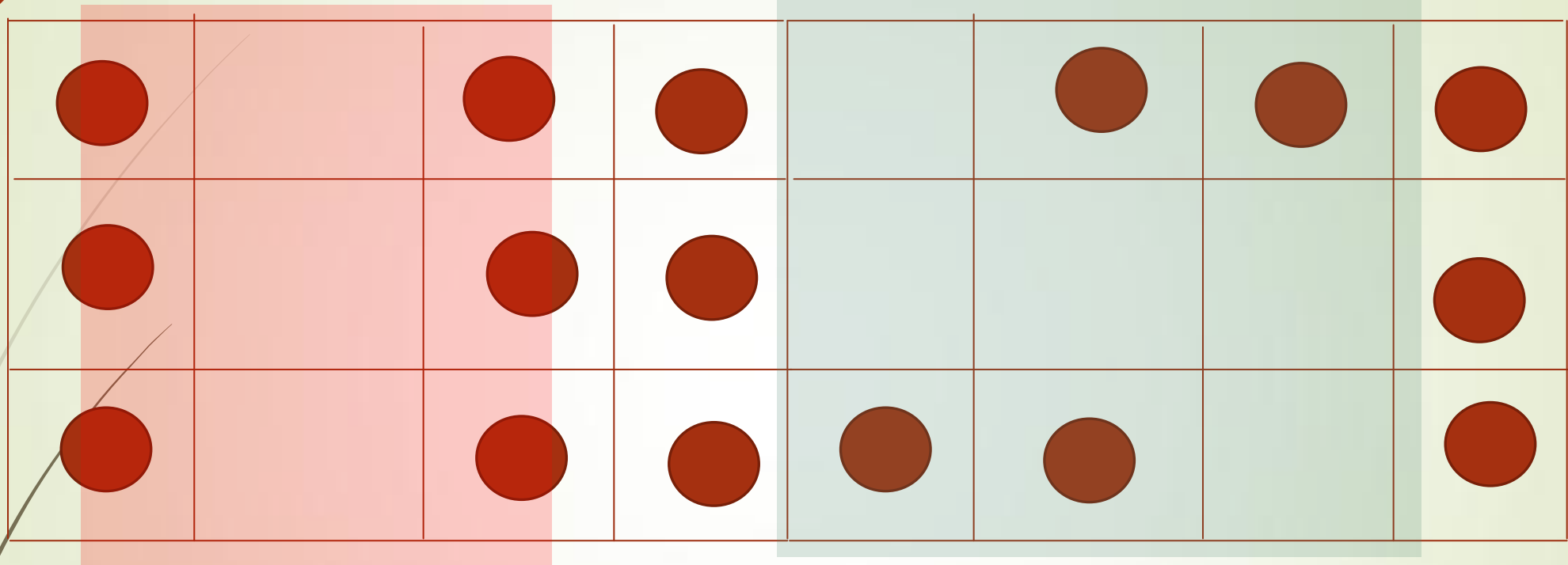
●		●	●	●	●	●	●
●							●
●	●	●	●	●	●		●



小課題2

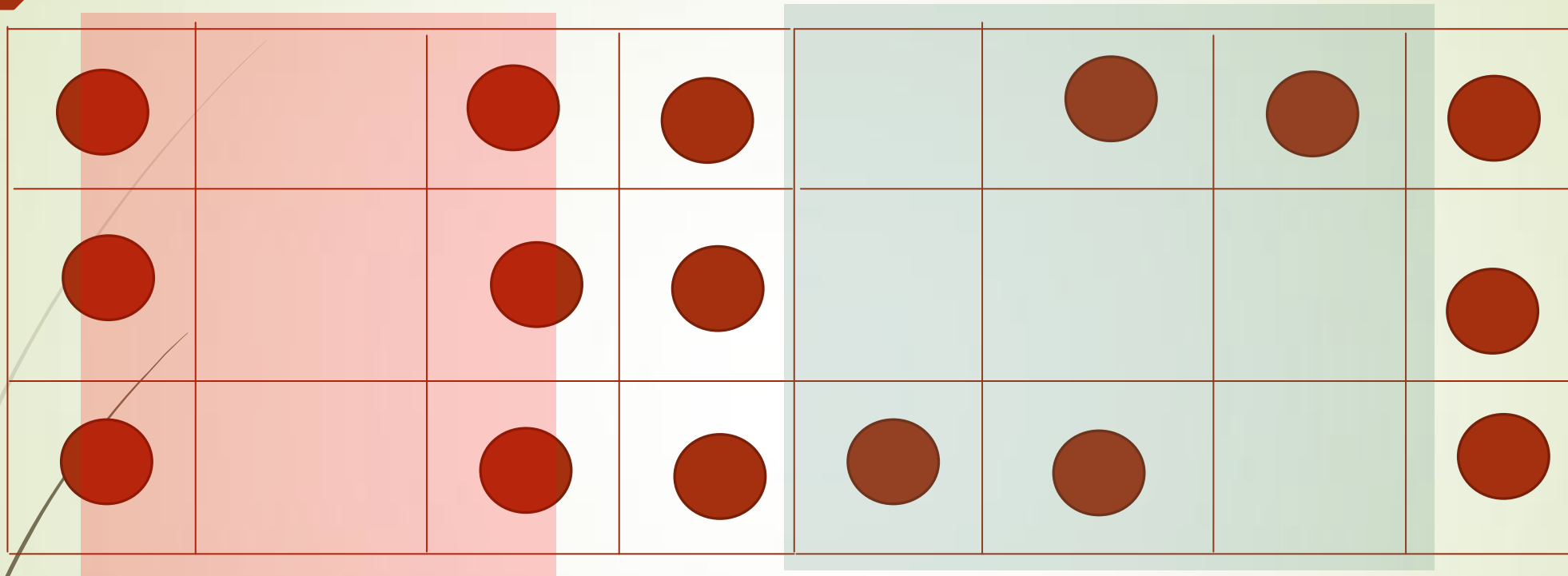
- 小課題2では、依存関係にある領域はそんなに大きくならない（高々5箇所）ので、それぞれ別個に考えてやればよい

計算

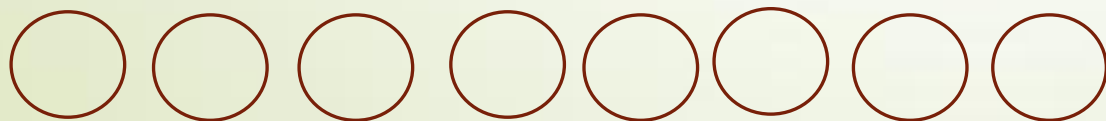


全体の操作回数は8回、赤い領域に石を置く回数は3回、青い領域には5回

計算




8回のうち赤い領域に置く番を決める。
赤い領域の中で順番を決める
青い領域で同様のことを行う




計算

- 実際にはもっと領域の数が多いが、一つの領域について手番の選び方を数える（組み合わせ）→領域の中での順番の数を数える（全探索とかbitDPとか）を繰り返して掛けあわせていけばよい
- 組み合わせはパスカルの三角形を用いたDPで問題ない

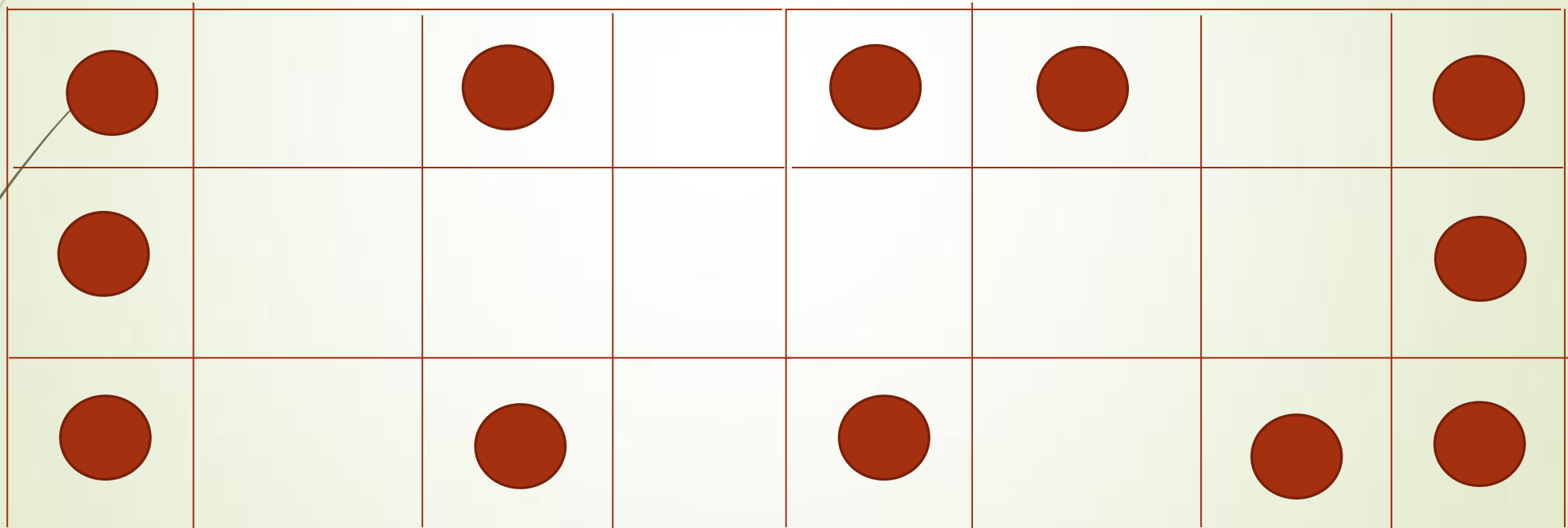



小課題4,5

- ▶ 小課題5では、連結領域が
- 

小課題4,5

▶ 小課題5では、連結領域がこんなものになる






ちよい考察

連結でない空白は置けるかどうかに関与しない
これはそのまま

しかし、連結した空白の数は小課題2と比べ膨大な大きさになる
(数千とか)

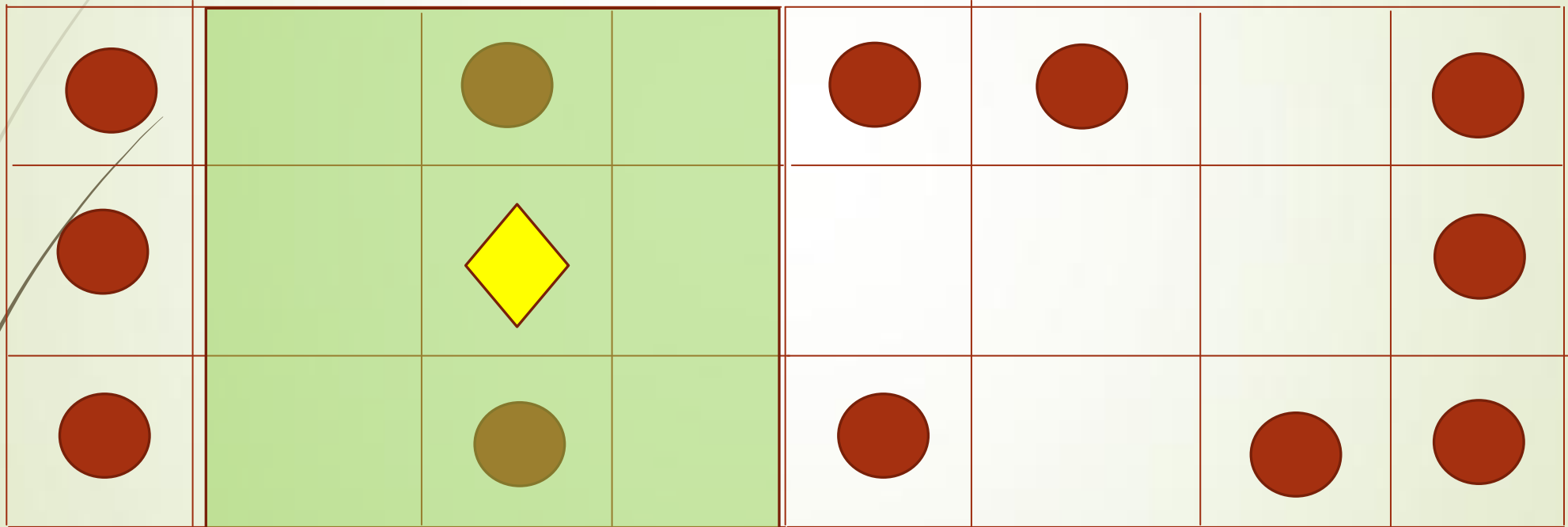
→全探索やbitDPではきびしい！



考察 1 2行目の空白について

2行目の空白はタテヨコ両方どちらでも挟められる
しかし、横に連続した2個以上の空白を両方ヨコで挟める場合は存在しない。

考察2 置く順の依存について




あるマスの置き方に影響する・されるマスは
高々そのマスの周囲1マス



■ 周囲の情報をうまく持てばDPできそう？



- 
- 周囲の情報をうまく持てばDPできそう？
→ 実際できる

実際の解法

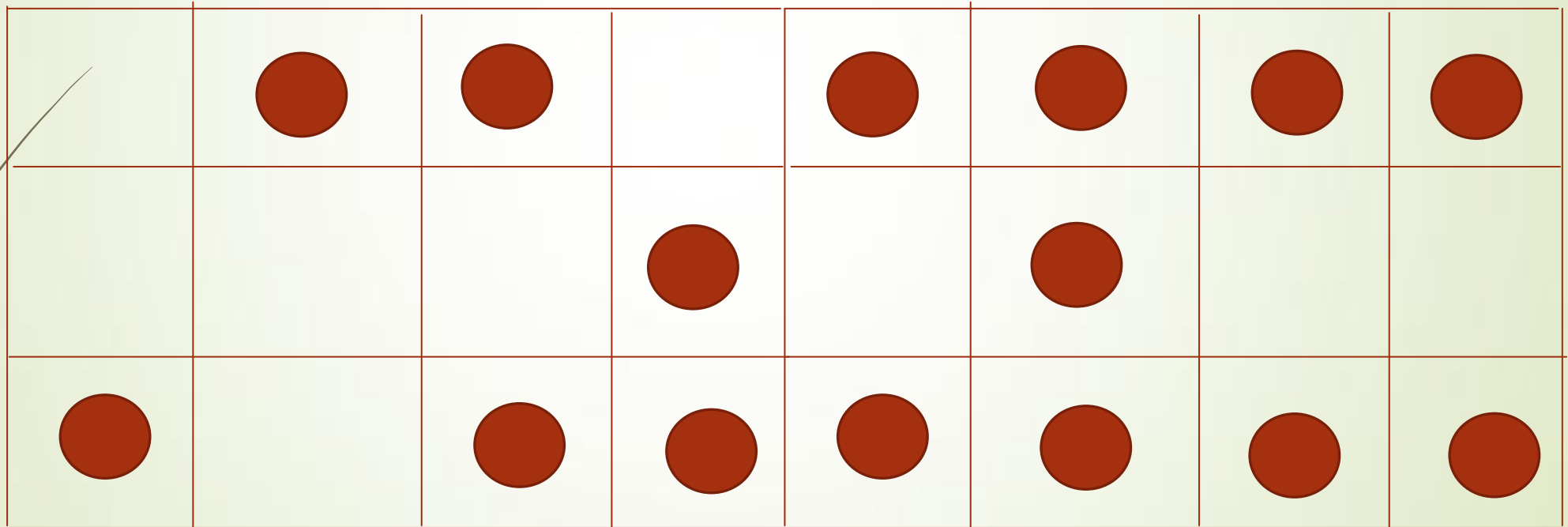
- 大まかな方針は小課題2と同じ
- 一つの領域について手番の選び方を数える（組み合わせ）→領域の中での順番の数を数える（全探索とかbitDPとか）を繰り返して掛けあわせる。



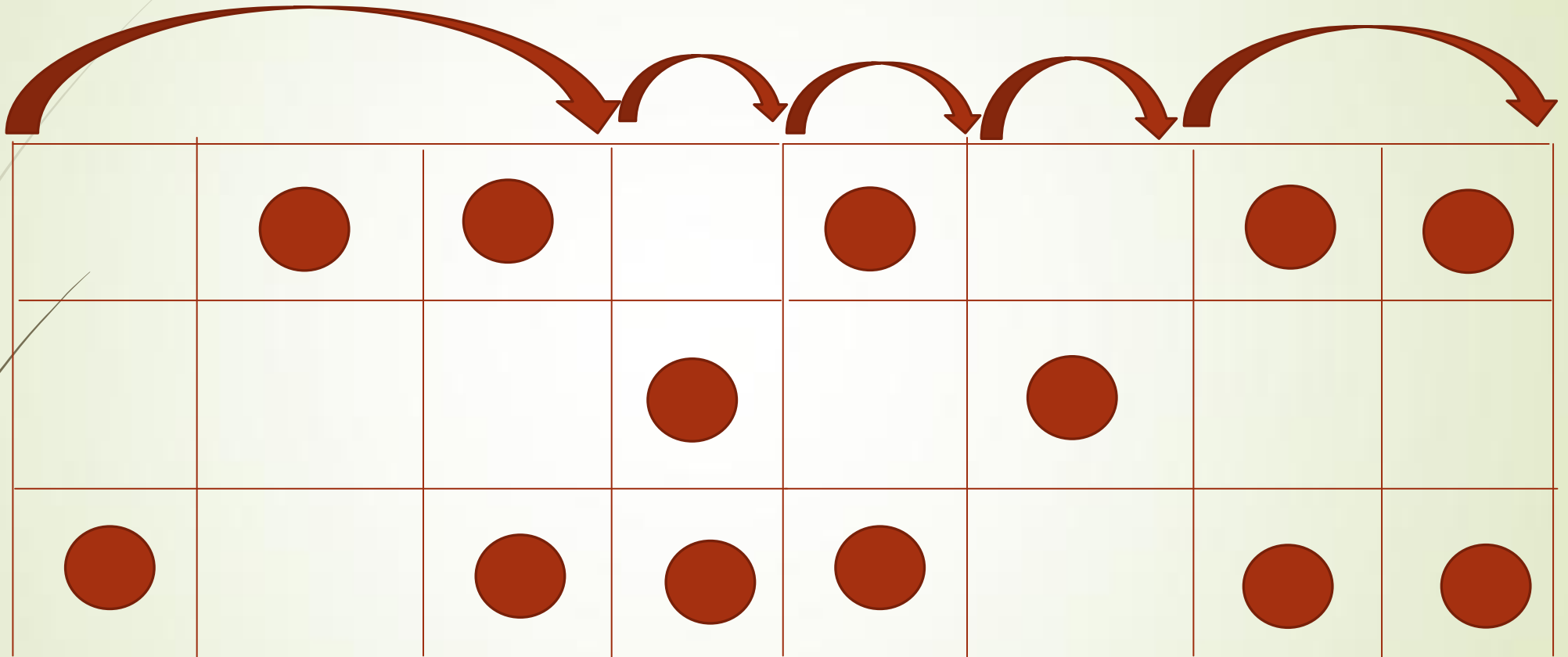
実際の解法

組み合わせがNが数千でも問題ない
順番の数については考える必要がある

領域の分け方



領域の分け方




2行目の空白が連続している列によって領域を区切る
2行目の空白が存在しない列は1列で独立している



2行目にコマがある列について

- ▶ 2行目にコマがある場合、その列単独で考えればよい。
- ▶ 列の空白が0,1つのとき、領域内での順番の数は1通り
- ▶ 列の空白が2つのとき、領域内での順番の数は2通り



2行目にコマがない列（領域）について

動的計画法を用いる

DP[i][j][k]

領域ははじめからi列目までの盤面としたとき、

i列目の2行目のコマが列の中で

(k?最後以外:最後)かつ全体j番目に置かれる場合の数(i+1列目に空白はないとする)

(最後以外・最後は横・縦で考えるとよい)

初期条件

- a列目からb列目までの領域について
a列目の空白がc個あるとき
DP[a][j][1]は $1 \leq c < j$ なら cPc 通り
DP[a][j][0]は $c=j$ なら cPc 通り
それ以外は0通り



遷移

2行目が最後のとき、

2行目 左右について特に制約はない

1,3行目 2行目よりも前にコマを置く必要がある

1つ前の2行目 最後以外なら今の2行目より後に置く必要がある



遷移

2行目が最後以外のとき、

2行目 左の2行目のコマよりも後である必要がある

1,3行目 両方とも2行目より前はだめ (片方だけならOK)

1つ前の2行目 最後以外の場合は不可

遷移

```
for (i, a...b) {  
    c = 領域内のi列目の空白の数;  
    d=領域内のi-1列目までの空白の数;  
    for (j, 0...d) {  
        for (k, 1...c) {  
            if (k == c)  
                DP[i][j + k][0] += nPr(c - 1, c - 1)*nCr(j + k - 1, k - 1)*(DP[i-1][0...d][0] + DP[i-1][j+1...d][1]);  
            else  
                DP[i][j + k][1] += nPr(c - 1, c - 1)*nCr(j + k - 1, k - 1)*nCr(c+d - (j + k), c - k)*DP[i-1][0...j][0];  
        }  
    }  
}
```



そのほか

modはちゃんと取ろう

DPの総和を取ってくる部分を高速化しないと $O(N^3)$ となってしまう、満点が取れなくなります。

小課題3はよくわかりません