北京师范大学
系统科学学院
《复杂性思维》

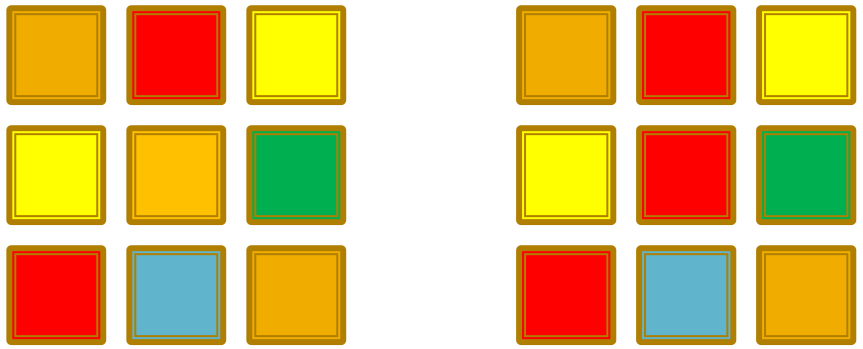
# 元胞自动机

# 投票模型

- 一个人工村落
- 每个人有8位邻居
- 这些人对总统竞选人进行投票
- 邻居的观点会对此人的选择实行影响
- 考察这些人的观点如何随时间演变

# 投票模型

- 每个格点有m种状态
- 随机选择一个格点拷贝它的状态
- 这是一个随机元胞自动机



# 投票模型

元胞自动机的状态value

```
patches-own[
  value
]
to setup
  ca
  ask patches
  [
    set value random opinions
    colorit
  ]
end
to go
  ask n-of number-changes patches[
    set value [value] of one-of neighbors
  ]
  ask patches
  [
    colorit
  ]
  do-plot
end
```

初始化

染色过程

```
to colorit
  let colors [black green blue yellow red
brown sky violet orange pink]
  set pcolor (item value colors)
End
```

画曲线的过程

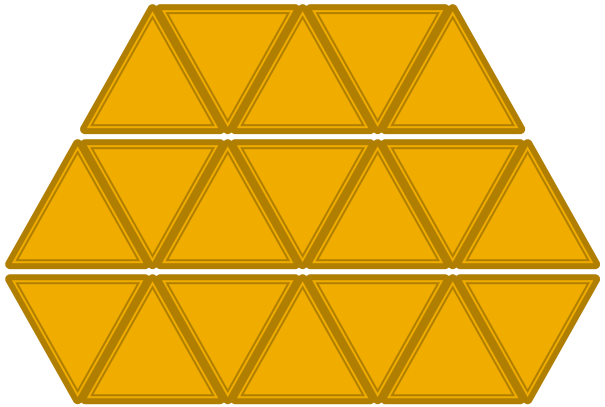
```
to do-plot
  set-current-plot "opinions"
  let i 1
  let species 0
  while [i < opinions]
  [
    let num (count patches with [value = i])
    if (num > 0)[set num 1]
    set species species + num
    set i i + 1
  ]
  plot species
end
```

统计某人得到的票数

随机选择number-changes个元胞
从所有patches选择

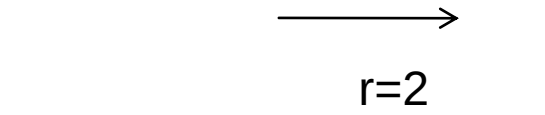
# 其它元胞自动机

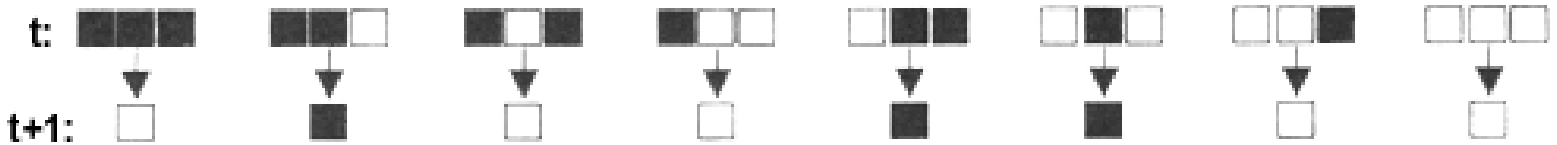
- 按照空间的排列方式
  - 四角格
  - 三角格空间
  - 六角格空间



# 一维元胞自动机

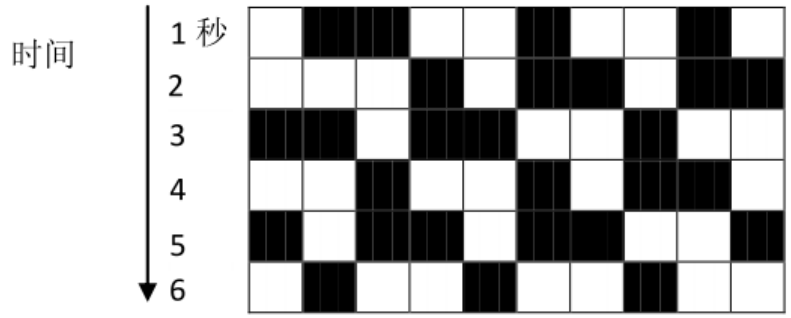
- 在一条一维的直线上
- 每个格子的左右两侧半径为r内的方格为其邻居
- 每个方格的状态可以用不同的颜色来表示
- 规则为一个列表



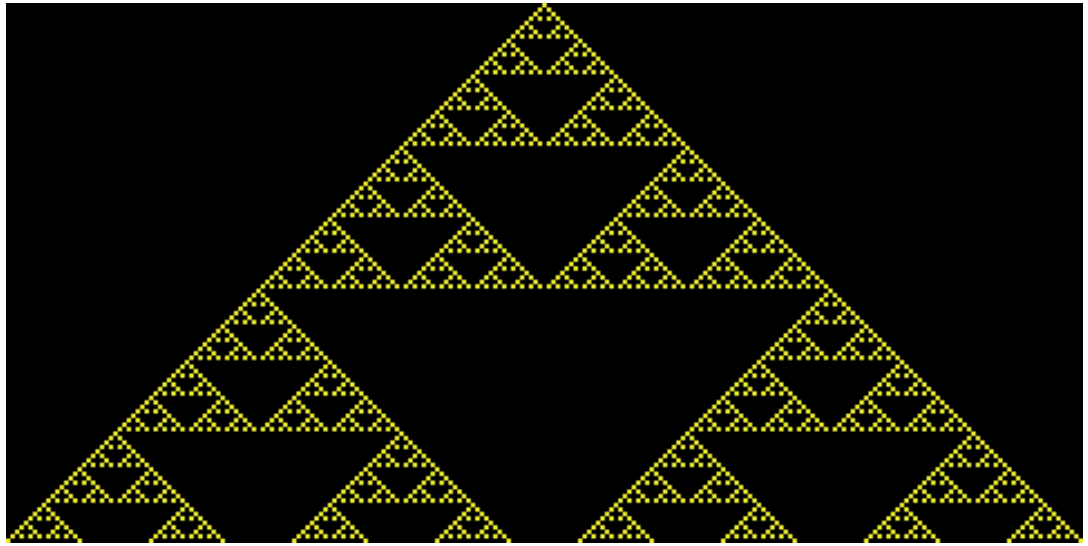
# 一维元胞自动机的运行

- 将某一个时刻所有格子排成一排
- 所有时刻的格子按照时间先后顺序从上到下进行排列，得到一张二维图



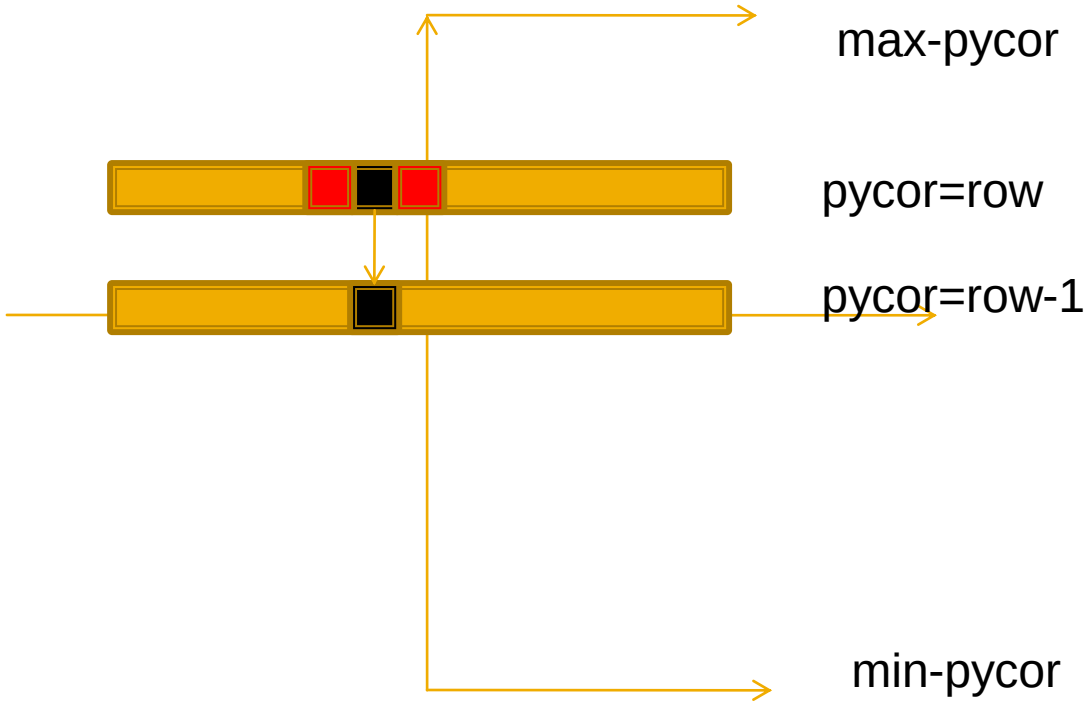
# 一维元胞自动机实例

- 一个一维元胞自动机，规则如下：
  - 如果自己或者邻居中有奇数个黄格，
    - 则下一时刻为黄色
  - 否则
    - 下一时刻为黑色



# 编程思路

- 设置变量row表示当前更新的行
- 只有那些pycor=row的方格才被更新，并且邻居状态也仅仅考虑pycor=row的方格



# 程序代码

```
globals [row]
patches-own [totalvalue]

to setup
  clear-all
  set row max-pycor
  ask patch 0 max-pycor [
     set pcolor yellow
  ]
  reset-ticks
end

to go
  if (row = min-pycor) [ stop ]
  ask patches with [pycor = row]
    [ do-rule ]
  set row (row - 1)
  tick
end

to do-rule
   set totalvalue count neighbors with [pcolor = yellow
         and pycor = row]
  ifelse (totalvalue mod 2 ) = 0
    [ ask patch-at 0 -1 [ set pcolor black ] ]
    [ ask patch-at 0 -1 [ set pcolor yellow ] ]
end
```

```
ask patch-at dx dy      访问相对坐标dx,dy处的patch
ask patch x y           访问绝对坐标x,y处的patch
neighbors with [pycor=row]   跟我同一行的邻居patchs
```

# 一维元胞自动机的编码

- 对于任意一个1半径，2状态的一维元胞自动机，其输入无非就是8种：111,110,101,100,011,010,001,000
- 因此可以用下表表示
- 一个8位长的二进制串就决定一个元胞自动机

| 111 | 110 | 101 | 100 | 011 | 010 | 001 | 000 |
|---|---|---|---|---|---|---|---|
| 1 | 0 | 0 | 0 | 0 | 1 | 1 | 0 |

规则：当邻域中有奇数个1的时候，就变成1，否则为0

11010001对应的规则是：

| 111 | 110 | 101 | 100 | 011 | 010 | 001 | 000 |
|---|---|---|---|---|---|---|---|
| 1 | 1 | 0 | 1 | 0 | 0 | 0 | 1 |

# 一维元胞自动机编码

- 对于任意一个1半径，2状态的一维元胞自动机，其规则表被一个8位长的二进制串给定，因此一共有$2^8$=256种，对应的元胞自动机也有256种。
- 每一个二进制串都对应一个十进制编码，所以这256个元胞自动机就对应了一个十进制数作为它的编码。
- 反过来，给定一个小于256的十进制数，也就给定了一个元胞自动机

二进制到十进制的转换：

$$n=\sum_{i=1}^{l} c_i 2^{i-1}$$

十进制到二进制的转换：

$$c_i=\mathrm{mod}(\frac{n}{2^i},2)$$

# 编码举例

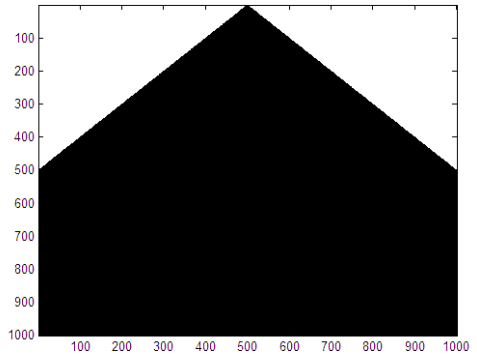
| 111 | 110 | 101 | 100 | 011 | 010 | 001 | 000 |
|---|---|---|---|---|---|---|---|
| 1 | 0 | 0 | 0 | 0 | 1 | 1 | 0 |

编码为：134

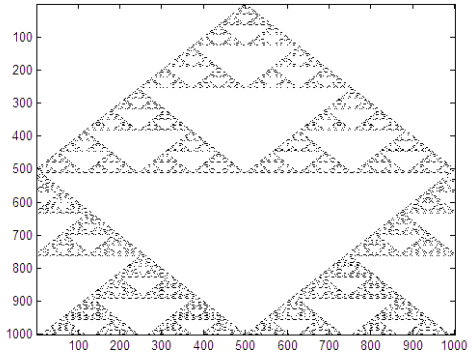
编码为110的元胞自动机规则表为：

| 111 | 110 | 101 | 100 | 011 | 010 | 001 | 000 |
|---|---|---|---|---|---|---|---|
| 0 | 1 | 1 | 0 | 1 | 1 | 1 | 0 |

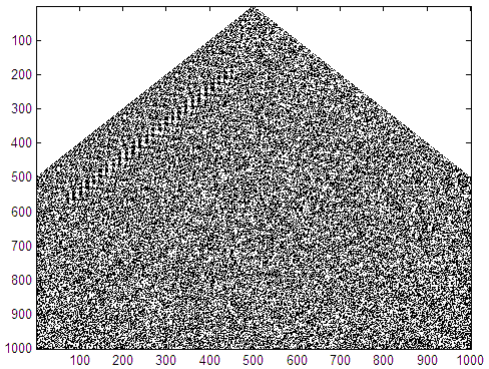
# 一维元胞自动机的运行



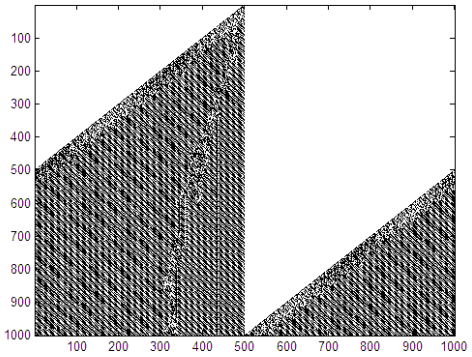

固定值：254




周期结构：90

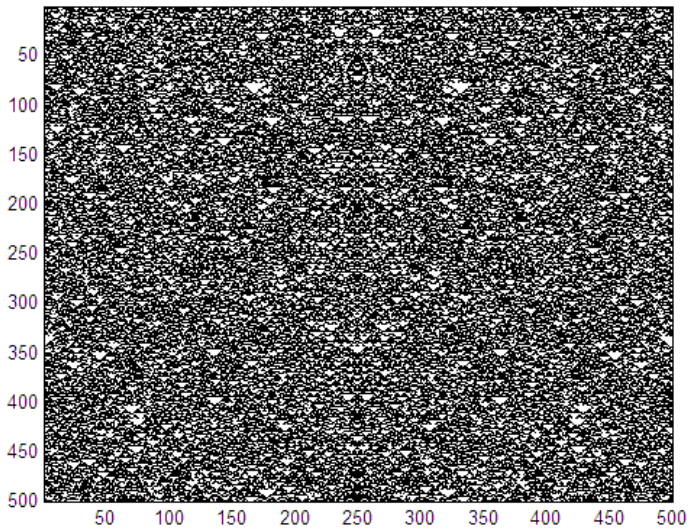


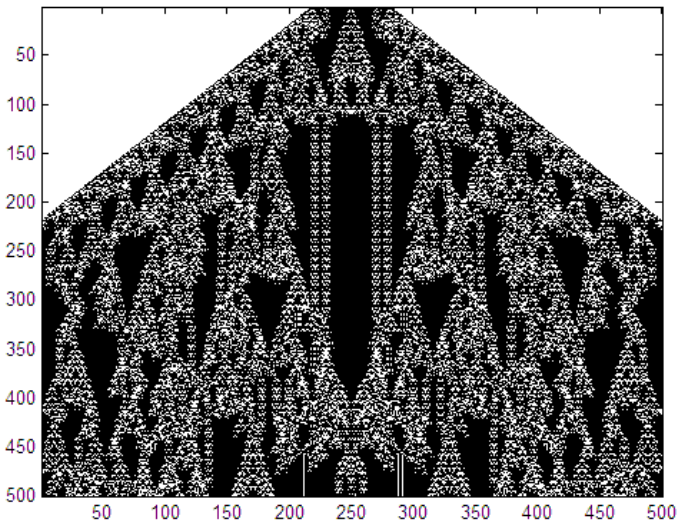

随机：30




复杂：110
（局域的不规则结构）

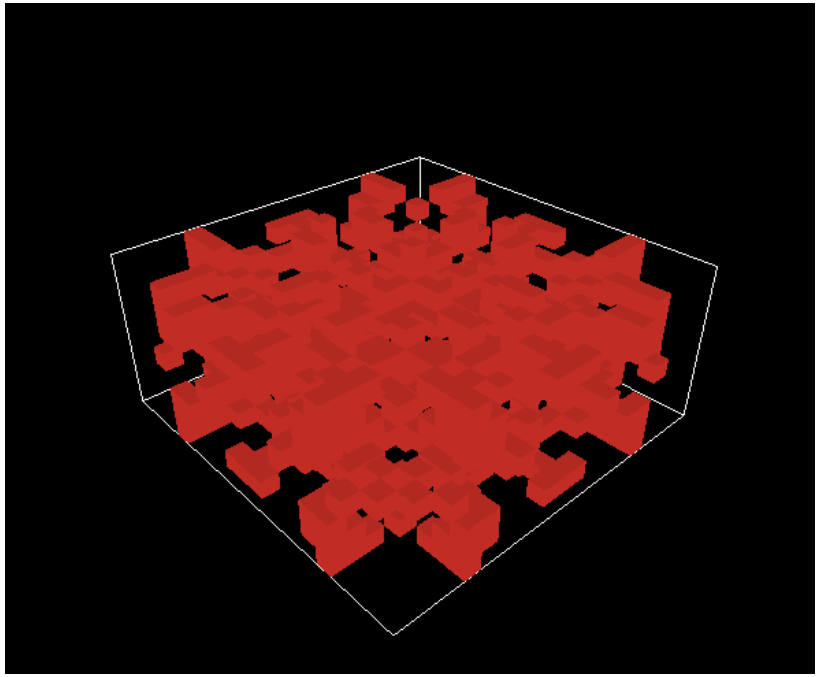
# 一维二邻居3状态元胞自动机的运行



# 三维的元胞自动机

- 三维 的生命游戏
- 规则：
  - 邻居为红色的元胞数在区间[r1,r2]的由红变黑
  - 邻居为红色的元胞数在区间[r3,r4]的由黑变红



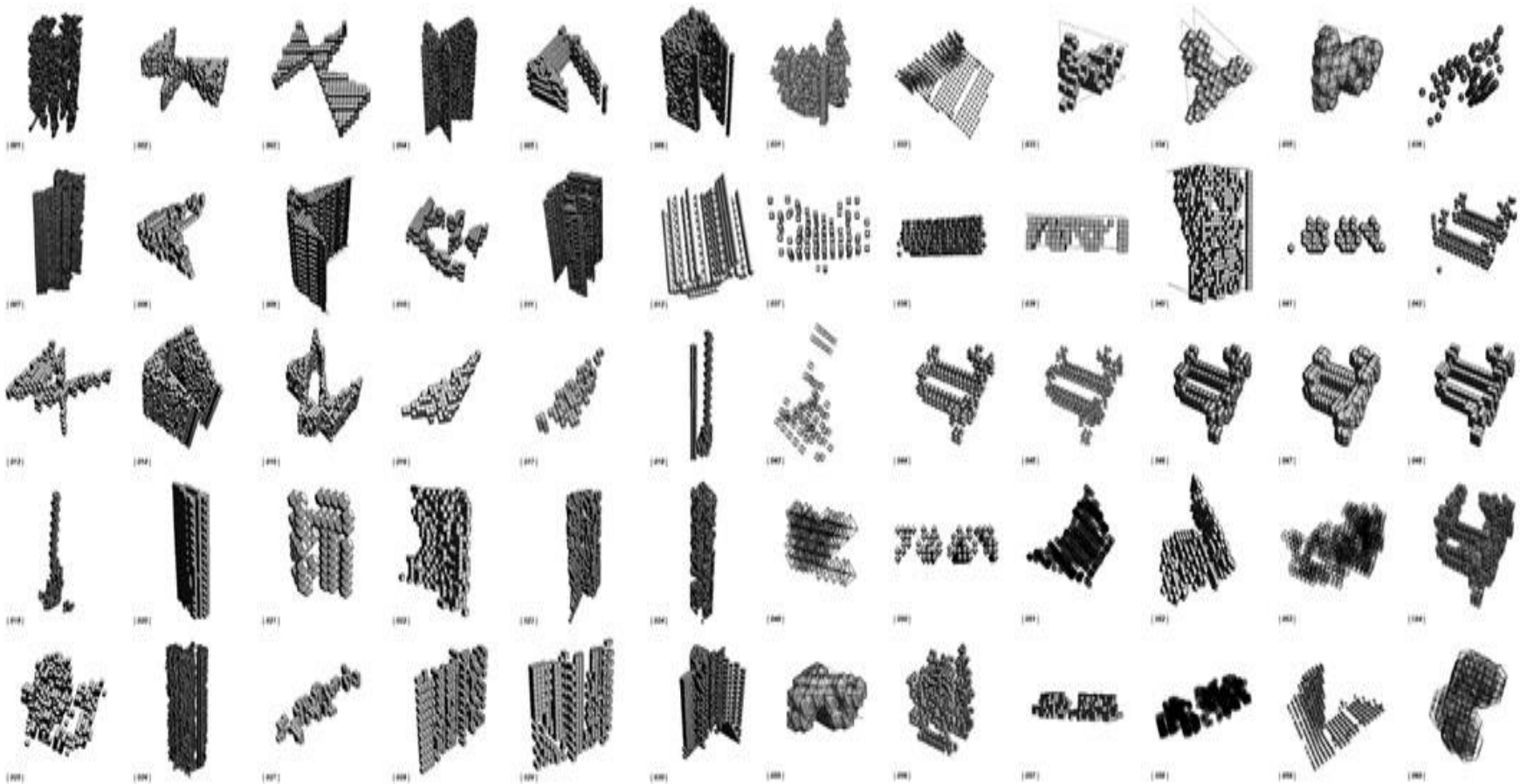
# 作业

如果将二维二状态元胞自动机的黑色方格作为砖，时间轴作为高度，则
自动机的运行在三维立体空间中就会形成一幢建筑。
要求你寻找一种元胞自动机的规则，演化出一幢建筑，使得：

这幢建筑非常美观
该建筑足够复杂：可以用黑色方格数/(size*size*T) 来度量
尽量满足结构稳定性的要求，即该建筑从外观上看不能存在明显的结构不稳定性

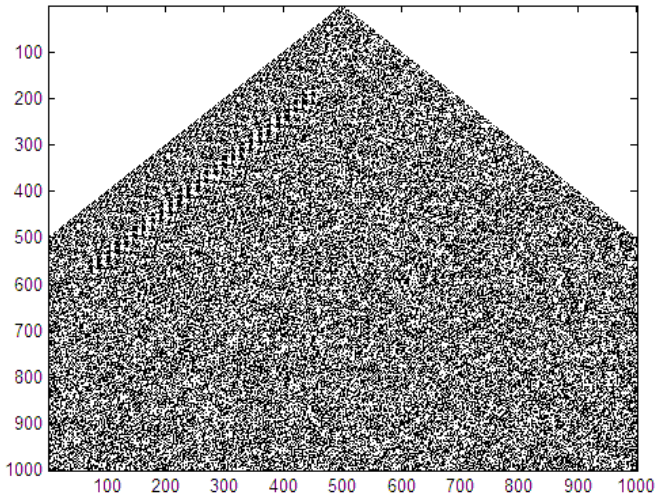
要求：提交源代码及其展示图

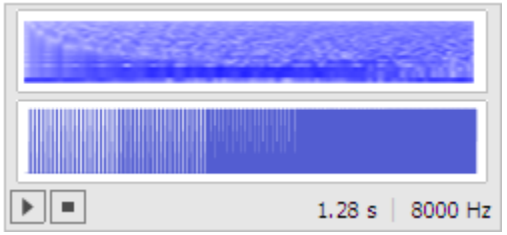
# 元胞自动机建筑示例



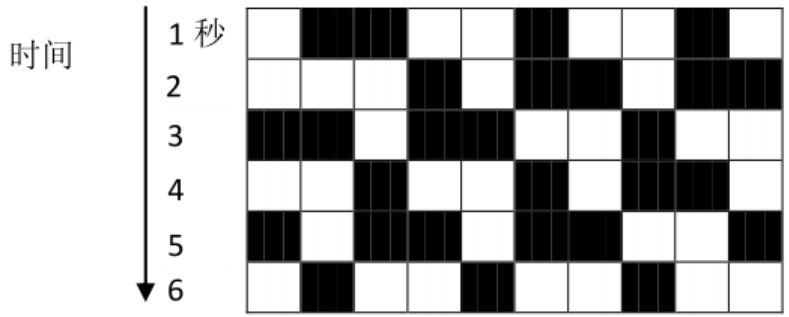
# 用元胞自动机奏乐

- 振幅为音量
- 频率为音高



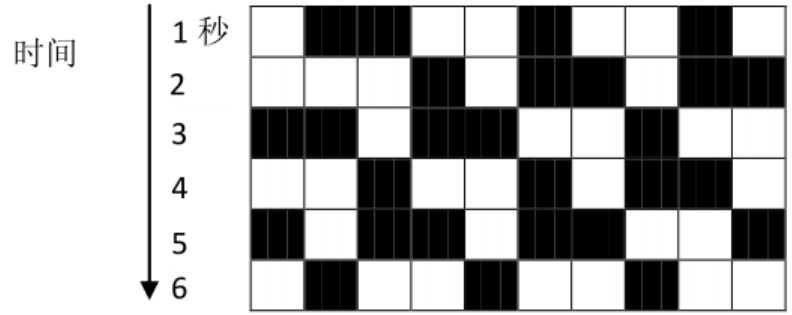
# 将二进制串映射成音符




1
3
4
+1
2

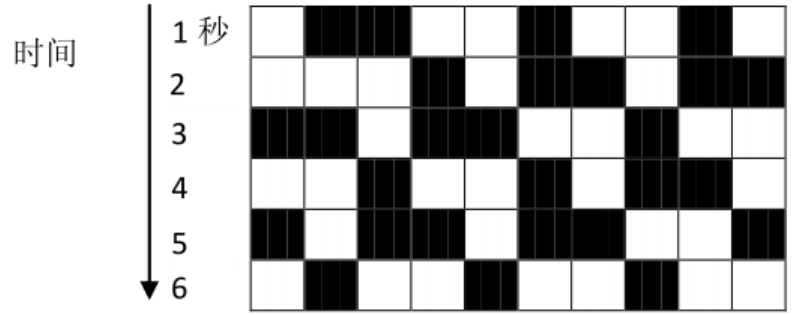
# 将二进制串映射成音符

0000000000000000



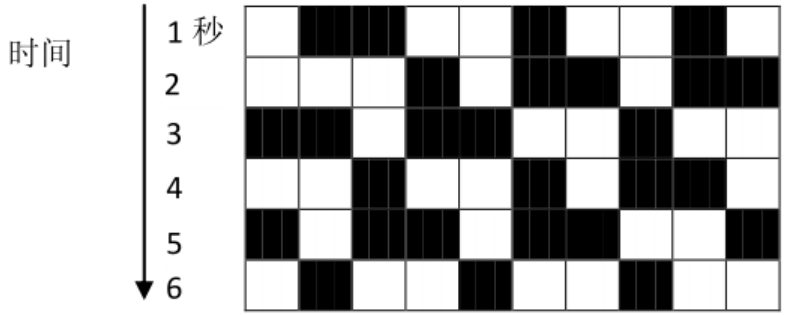

1
3
4
+1
2

0000000000000001




1
3
4
+1
2

0000000000000010




1
3
4
+1
2

# 混沌边缘的复杂

# Langton的λ参数

- 对于任意一个规则表，可以定义一个参数λ，来衡量元胞自动机运行后的混沌状态。
- 设背景状态为0
- 对于任意的规则表

| 222 | 221 | 220 | … | 211 | 220 | 221 | 222 |
|---|---|---|---|---|---|---|---|
| 0 | 2 | 1 | … | 0 | 1 | 2 | 0 |

计算该表中非0元素的比例

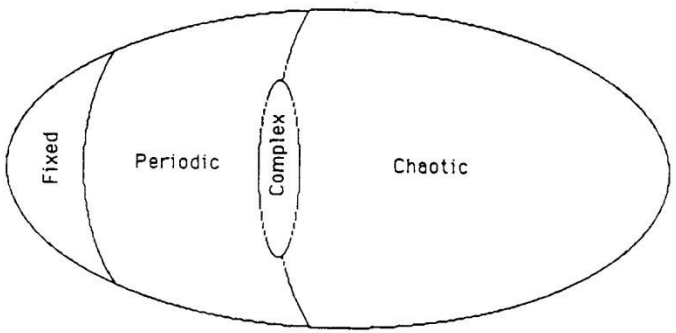
$$\lambda = \frac{s^n - n_q}{s^n}$$

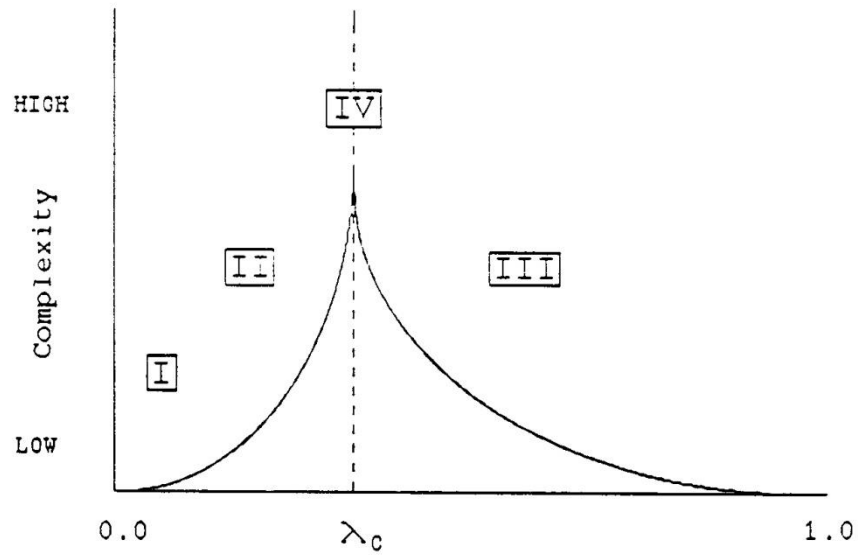
其中，s为每个元胞的状态个数,n邻域集合大小，$n_q$为0元素的个数

# Langton的λ参数

- 当λ=0~0.1，所有的元胞被吸引到一种固定的状态，这相当于第一类元胞自动机；
- λ=0.2附近，系统在一些固定的状态之间周期的循环，这相当于第二类元胞自动机，λ=0.3的元胞自动机比λ=0.2的在开始的时候具有更复杂的结构；
- λ介于大约0.3到0.6之间的时候，会出现相当复杂的结构。这些结构既不属于固定的周期或者固定值，也不属于完全的随机，因此这些元胞自动机属于第四类即“复杂型”。并且，随着λ的增长，复杂结构的维持时间也会变得越来越大；
- λ>=0.6的时候，复杂的结构消失，系统将被吸引于一种完全随机的混沌状态。

# 复杂——混沌的边缘


Fixed
Periodic
Complex
Chaotic


HIGH
Complexity
LOW
I
II
III
IV
0.0
λc
1.0

# 如何得到复杂?

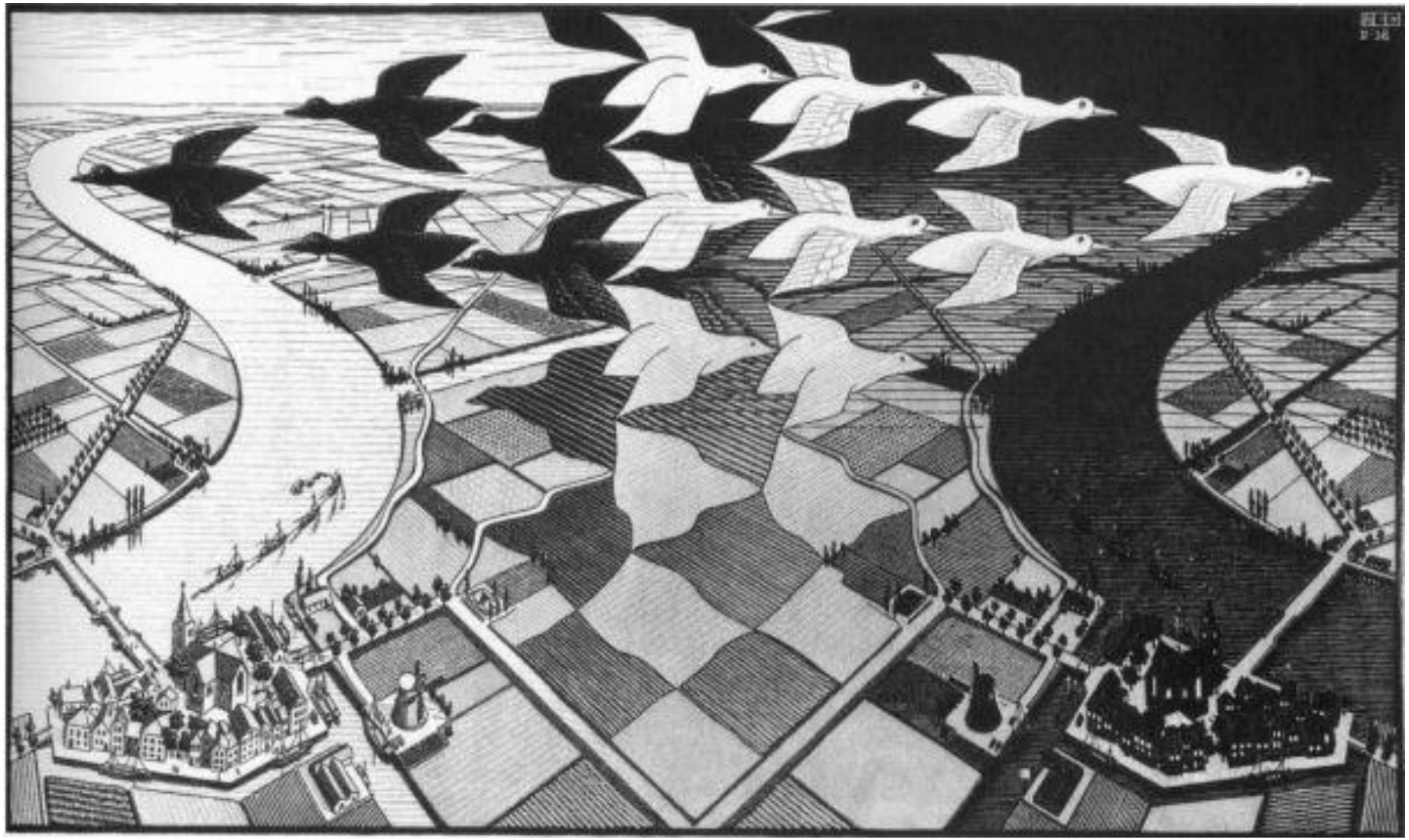
- 让系统运行在混沌的边缘

- 生命游戏：3是关键参数
- 鸟群模型：（对齐、靠近）：秩序、避免碰撞：混沌
- 森林火灾模型
- 自组织临界

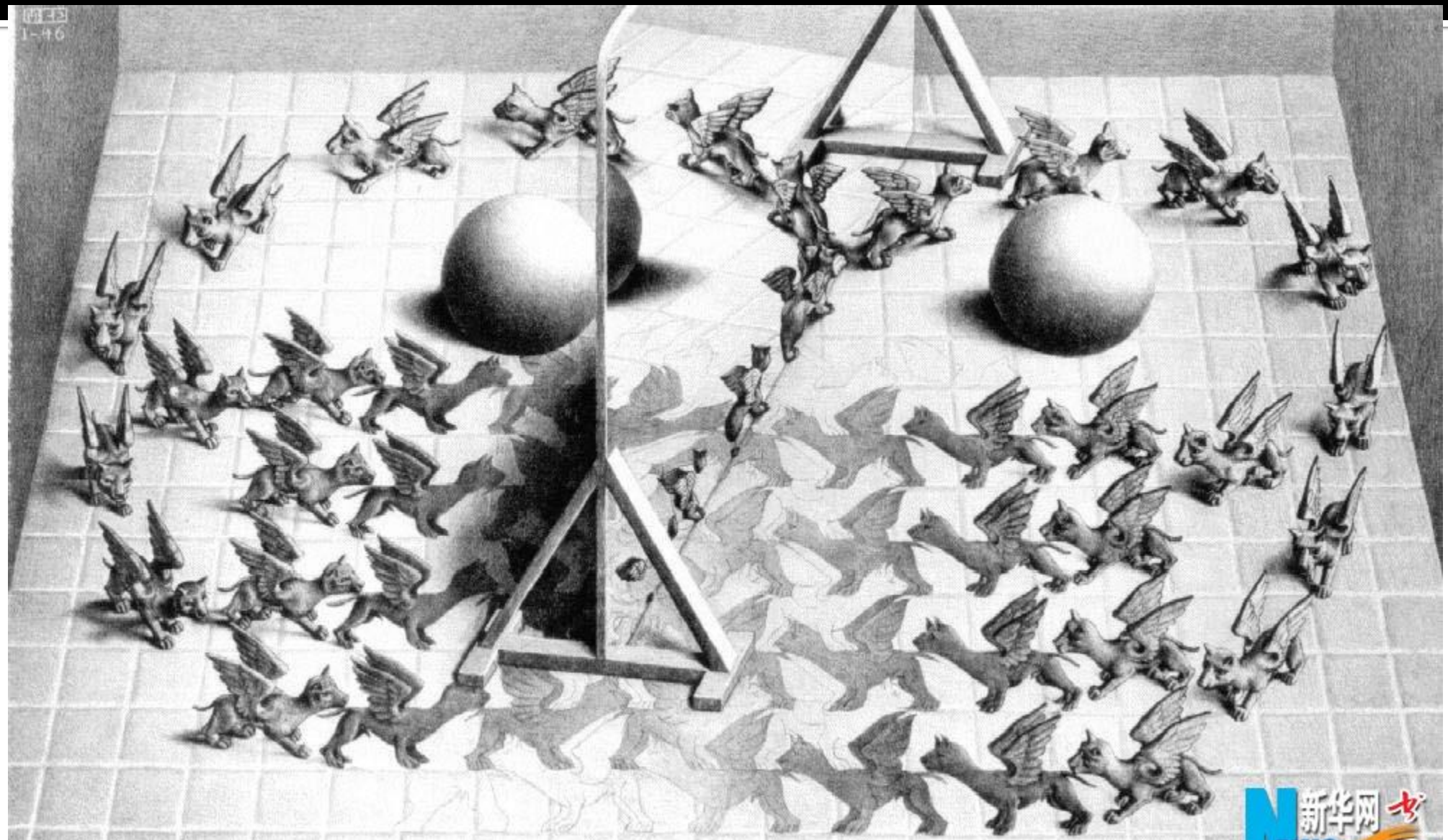
# 混沌边缘的思维

- 中庸之道
- 万事不求绝对
- 物极必反
- 既需要秩序又需要混沌

# 图形和衬底



# 阴阳两界



# 阴阳两界