中国地质大学

2019年大学生数学建模暑期备赛活动

专题：元胞自动机与格子理论

数理学院　黄昌盛

2019-08-28

元胞自动机的起源与发展

1950提出

1970生命游戏

1983初等元胞自动机

1986至今

元胞自动机是在40年代由Ulam首先提出,随后计算机之父冯.诺伊曼提出构造一个不确定的生命模型系统的设想，这个系统可以智能的自我进化。后来，冯.诺伊曼参照生物现象的自繁殖原理，将这个模型发展为一个网格状的自动机网络，每个网格为一个单元自动机，单元状态有生和死，相当于人体组织的存活和消亡。

元胞自动机的起源与发展

1950提出

1970生命游戏

1983初等元胞自动机

1986至今

20世纪70年代，微型计算机的发展为元胞自动机的模拟提供了基本条件。英国数学家John Conway试图简化Von Neumann的通用计算模型，编制了一个名为“生命游戏”的计算机程序，并由Martin Gardner刊登在《Scientific American》杂志的“数学游戏”专栏，引起了物理、数学、生物、计算机、地理等领域专家的兴趣，“生命游戏”被认为是元胞自动机研究的真正开始。

元胞自动机的起源与发展

1950提出

1970生命游戏

1983初等元胞自动机

1986至今

20世纪80年代初，美国物理学家Stephen Wolfram对一维元胞自动机，尤其是初等元胞自动机的深入分析，使元胞自动机理论和应用研究有了突破性的进展。他借用统计力学的概念和方法，并在大量计算机实验的基础上，将所有元胞自动机的动力学行为归纳为四类，对于元胞自动机的研究具有很大的指导意义。它反映出这种分类方法可能具有某种普适性，很可能有许多物理系统或生命系统可以按这样的分类方法来研究。

元胞自动机的起源与发展

1950提出

1970生命游戏

1983初等元胞自动机

1986至今

20 世纪 90 年代元胞自动机在各个领域得到了广泛的应用，包括社会、经济、军事和科学研究的各个领域。应用领域涉及社会学、生物学、生态学、信息科学、计算机科学、数学、物理学、化学、地理、歹境、军事学等。

在社会学中，元胞自动机用于研究个人行为的社会性问题的流行现象，例如人口的迁移，公共场所内人员的疏散，流行病的传播。这类社会学问题也是潜在的数学建模竞赛问题。

元胞自动机的起源与发展

1950提出

1970生命游戏

1983初等元胞自动机

1986至今

在生物学中，元胞自动机的设计思想本身就来源于生物学自繁殖的思想，因而它在生物学上的应用更为自然而广泛。例如元胞自动机朋于肿瘤细胞的增长机理和过程模拟、人类大脑的机理探索、艾滋病病毒HIV的感染过程、自组织、自繁殖等生命现象的研究。

元胞自动机的起源与发展

1950提出

1970生命游戏

1983初等元胞自动机

1986至今

生态学领域：元胞自动机被用于兔子-草，鲨鱼-小鱼等生态系统动态变化过程的模拟，展示出令人满意的动态效果;元胞自动机还成功地应用于蚂蚁的行走路径，大雁、鱼类洄游等动物的群体行为的模拟；另外，基于元胞自动机模型的生物群落的扩散模拟也是当前的一个应用热点。

元胞自动机的起源与发展

1950提出

1970生命游戏

1983初等元胞自动机

1986至今

交通科学领域：1986年，M. Cremer和J. Ludwig初次将元胞自动机运用到车辆交通的研究中。随后，元胞自动机在车辆交通中的应用主要沿着两条主线展开：对高速公路交通流的研究，以Nagel-Schreckenberg模型为代表;对城市交通网络的研究，以BML模型为代表。另外，80年代以来，计算机水平日新月异的发展为元胞自动机的 应用提供了强有力的支持。因此，在进入上个世纪90年代后，元胞自动机在交通流理论研究领域中得到了广泛的应用。

元胞自动机的起源与发展

1950提出

1970生命游戏

1983初等元胞自动机

1986至今

计算机科学与信息学领域：元胞自动机的逻辑思维方法为并行机的发展提供了另一个理论框架。20世纪80年代，T. Toffoli和N.H. Margolus 制造出第一台通用元胞自动机计算机CAM6，其性能可与当时的巨型计算机相比拟，并且其图形显示功能明显优于其他类型的计算机。元胞自动机还被用来研究信息的保存、传递、扩散的过程。除此之外，元胞自动机在图像处理和模式识别中也体现出了其独到的优势 。

元胞自动机的起源与发展

1950提出

1970生命游戏

1983初等元胞自动机

1986至今

物理学领域：在元胞自动机基础之上发展出来的格子气自动机(LGA)和格子-波尔兹曼方法(LBM)在计算流体领域获得了巨大的成功。不仅能够解决传统流体力学计算方法所能解决的绝大多数问题，并且在多孔介质、多相流、微小尺度方面具有其独特的优越性。格子-波尔兹曼方法还被成功地应用于磁场、电场、热扩散和热传导的模拟。另外，元胞自动机还被用来模拟雪花等枝晶的形成、液态金属材料的凝固结晶过程以及颗粒材料的垮塌现象等。

元胞自动机在数学建模竞赛中的应用

- 美赛中用到元胞自动机的特等奖论文不完全统计*

年份题号	题目	特等奖论文个数
2001MCM-B	逃避飓风	1
2003MCM-B	Gamma 刀治疗方案	1
2005MCM-B	收费亭的最优数量	3
2007MCM-B	飞机座位方案	4
2009MCM-A	交通环岛的设计	2
2012MCM-B	沿着“大长河”露营	1
2014MCM-A	交通右行规则	6

*数据来源于《大学生数学建模竞赛指南》，肖华勇主编，电子工业出版社，2015

元胞自动机在数学建模竞赛中的应用

- 国赛适宜应用元胞自动机的题目：

  2013年A题：车道被占用对城市道路通行能力的影响

  2016年B题：小区开放对道路通行的影响

引例：生命游戏

生命游戏其实是一个零玩家游戏，它包括一个二维矩形世界，这个世界中的每个方格居住着一个活着的或死了的细胞。一个细胞在下一个时刻生死取决于相邻八个方格中活着的或死了的细胞的数量。如果相邻方格活着的细胞数量过多，这个细胞会因为资源匮乏而在下一个时刻死去；相反，如果周围活细胞过少，这个细胞会因太孤单而死去。

生命游戏

每个格子的生死遵循下面的原则：

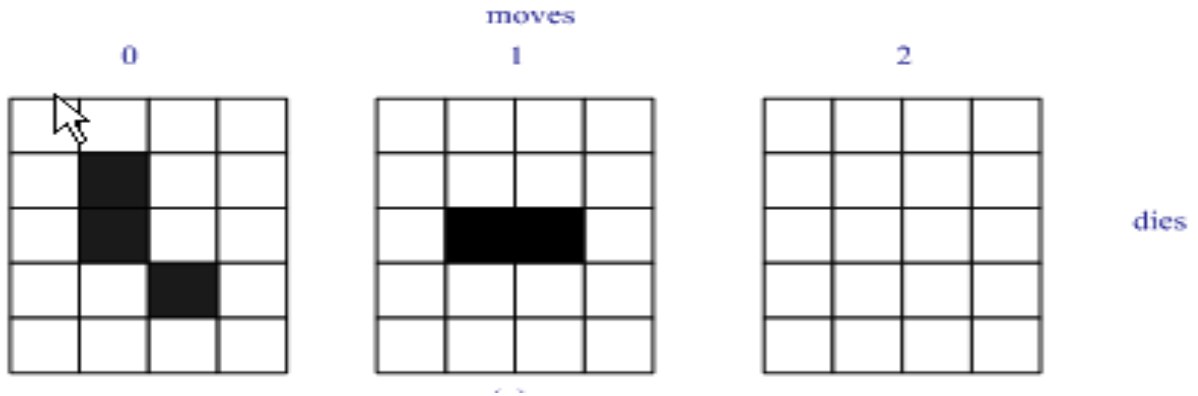
1． 如果一个细胞周围有3个细胞为生（一个细胞周围共有8个细胞），则该细胞为生（即该细胞若原先为死，则转为生，若原先为生，则保持不变） 。

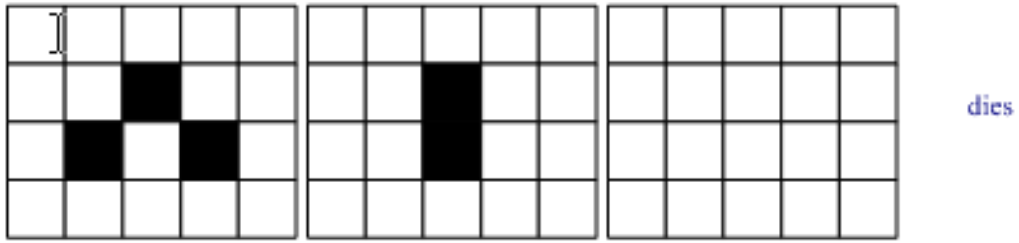
2． 如果一个细胞周围有2个细胞为生，则该细胞的生死状态保持不变；

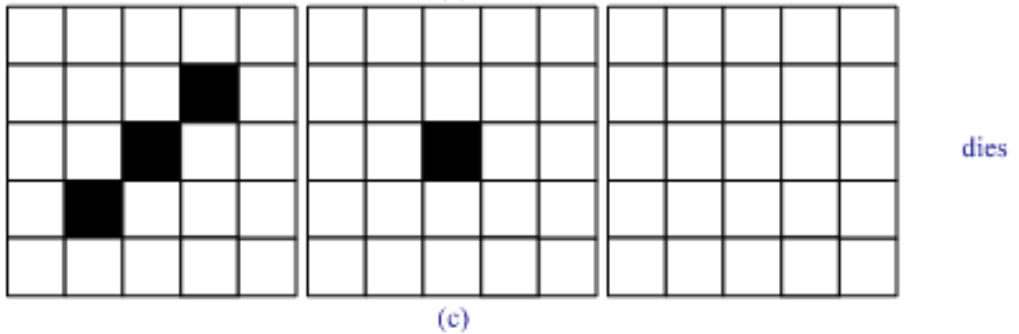
3． 在其它情况下，该细胞为死。

设定图像中每个像素的初始状态后依据上述的游戏规则演绎生命的变化，由于初始状态和迭代次数不同，将会得到令人叹服的优美图案）。

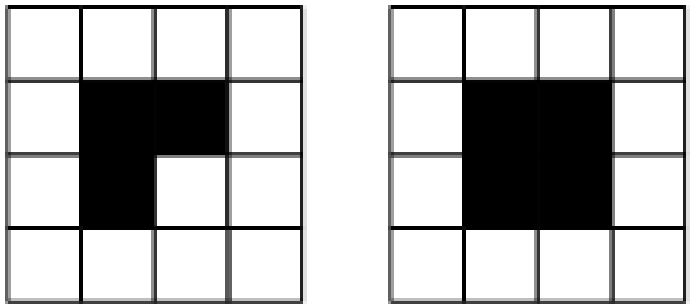
生命游戏





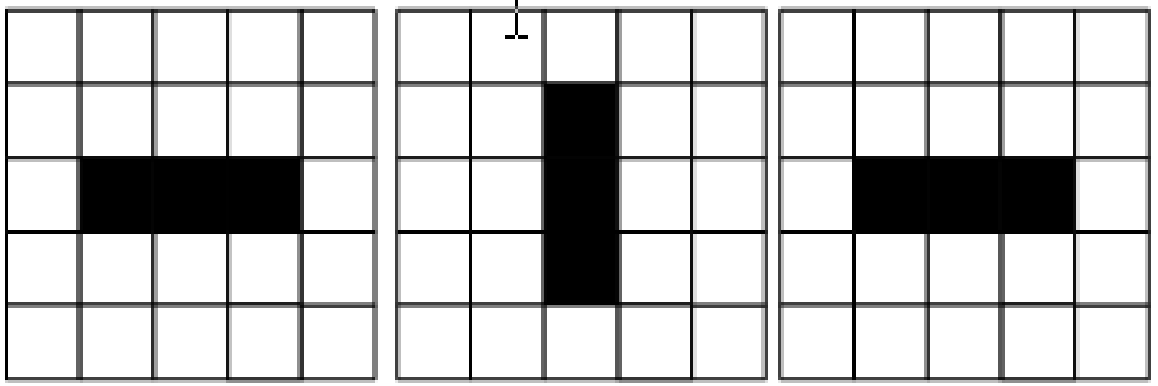


生命游戏



Block(stable)

(d)



Blinker (period 2)

(c)

生命游戏

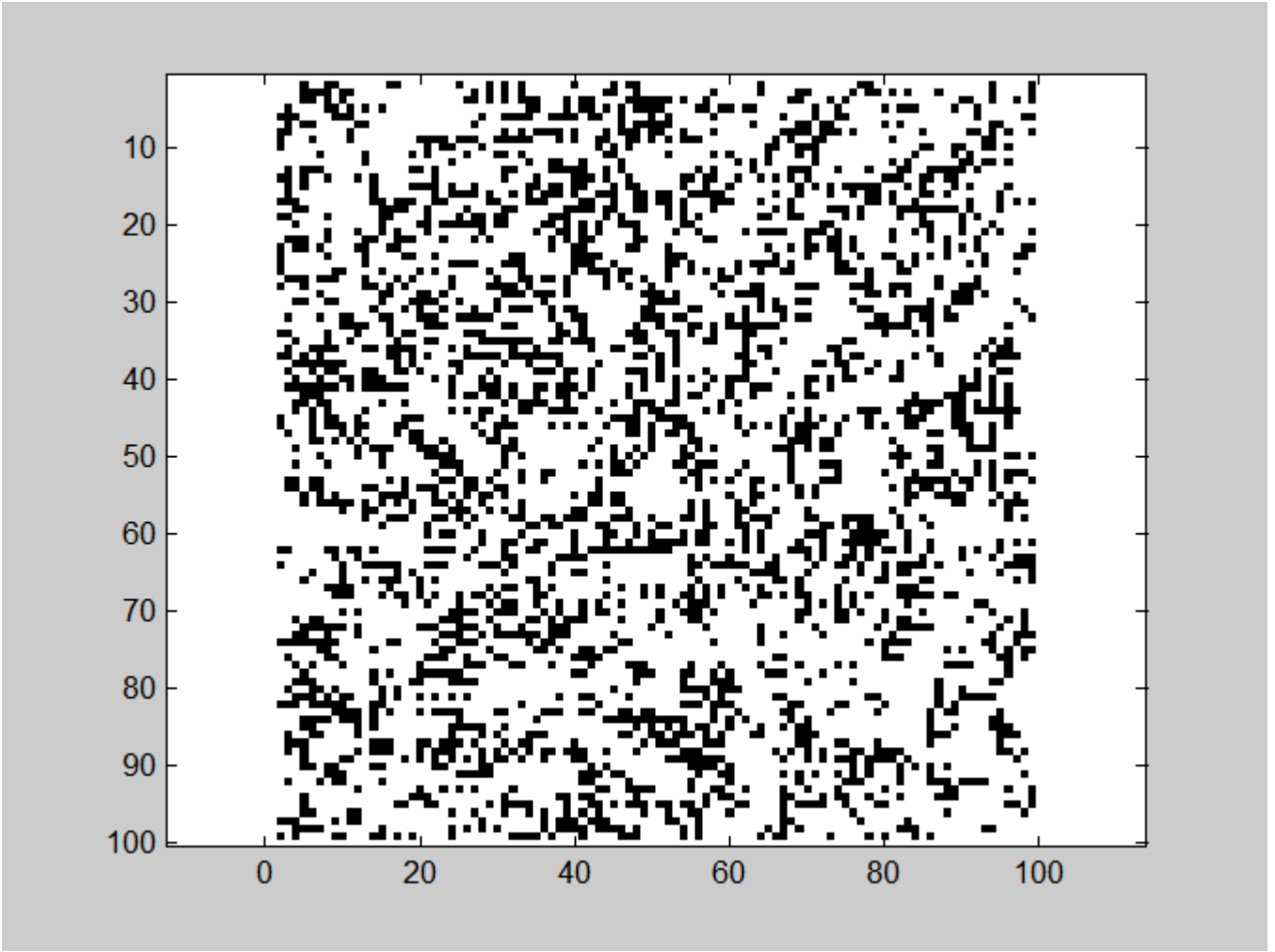
核心代码：

```
x=2:n-1;
y=2:n-1;


sum(x,y)=cells(x,y-1) + cells(x,y+1) +...
         cells(x-1,y) + cells(x+1,y) +...
         cells(x-1,y-1) + cells(x-1,y+1) +...
         cells(x+1,y-1) + cells(x+1,y+1);

cells = (sum==3) | (sum==2 & cells);
```

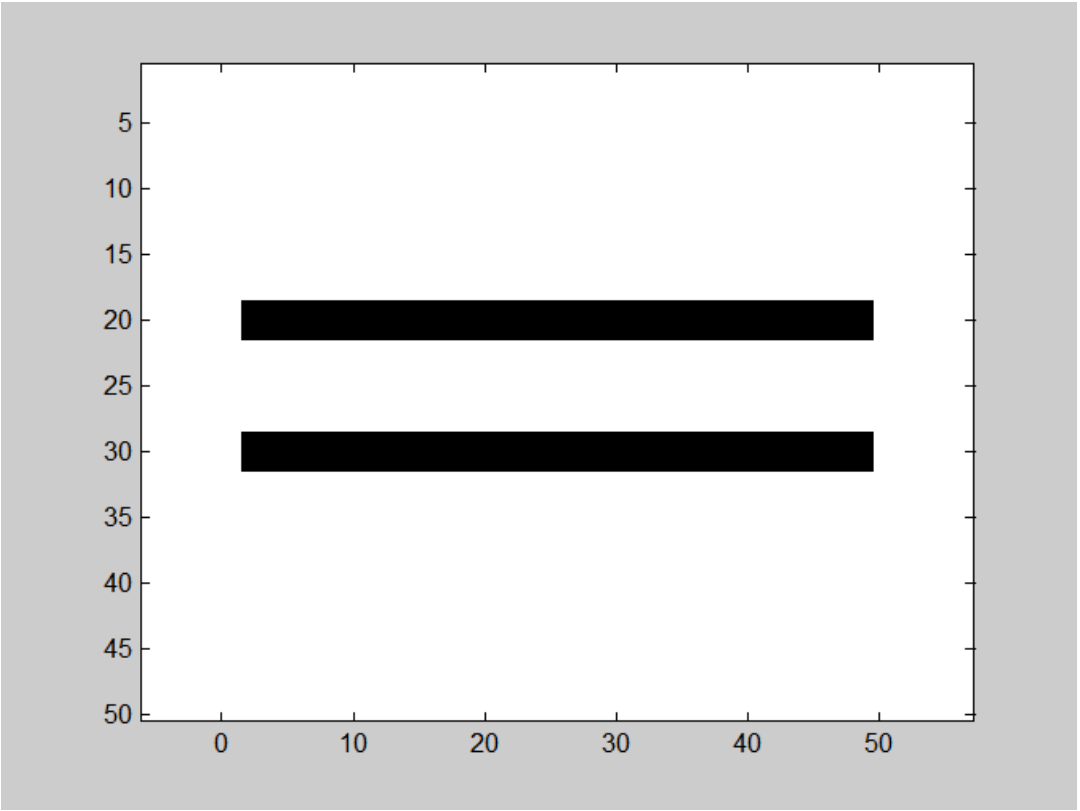
生命游戏

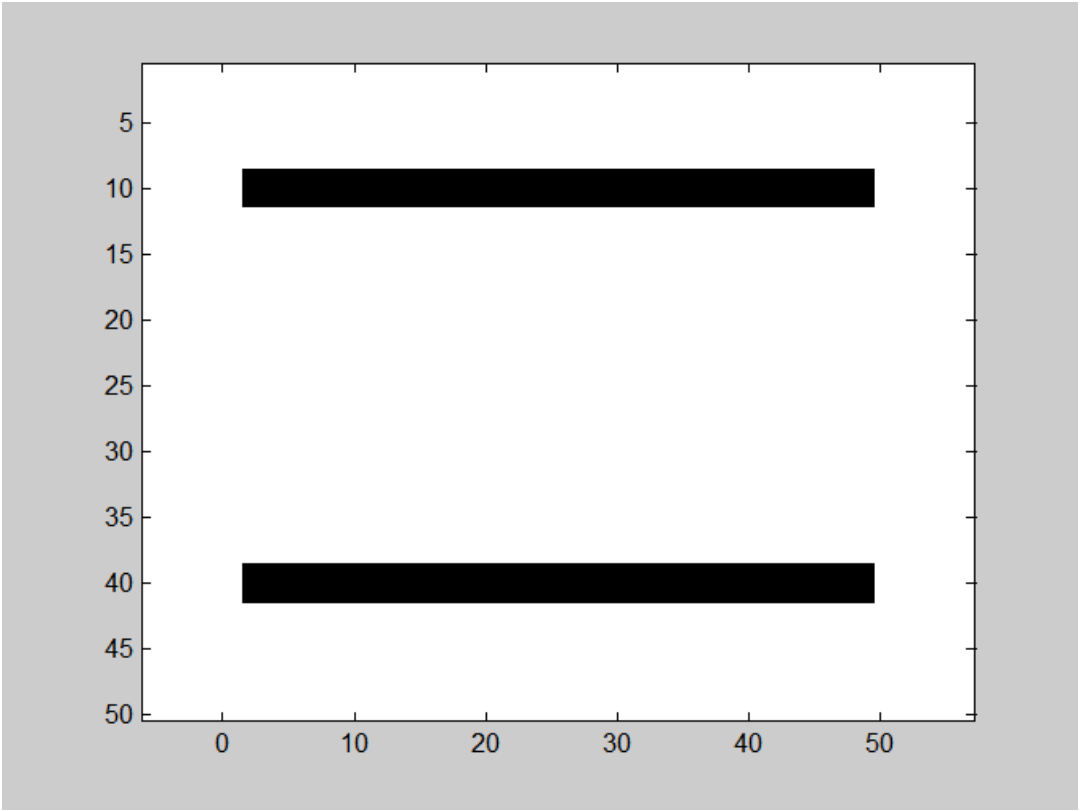
```
n=100;
cells = rand(n,n)>0.5;
sum = zeros(n,n);
x=2:n-1;
y=2:n-1;
imh=image(cat(3,cells==0,cells==0,cells==0));
for i=1:1000
    sum(x,y)=cells(x,y-1) + cells(x,y+1) +...
        cells(x-1,y) + cells(x+1,y) +...
        cells(x-1,y-1) + cells(x-1,y+1) +...
        cells(x+1,y-1) + cells(x+1,y+1);

    cells = (sum==3) | (sum==2 & cells);

    set(imh, 'cdata',
cat(3,cells==0,cells==0,cells==0))
    drawnow
end
```

生命游戏

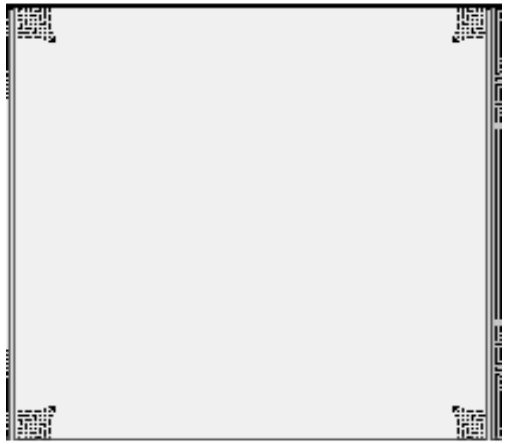


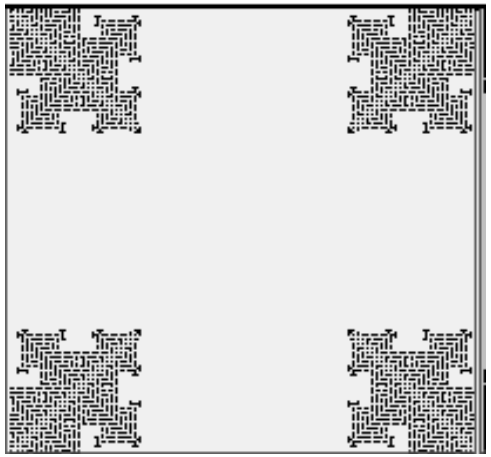
生命游戏

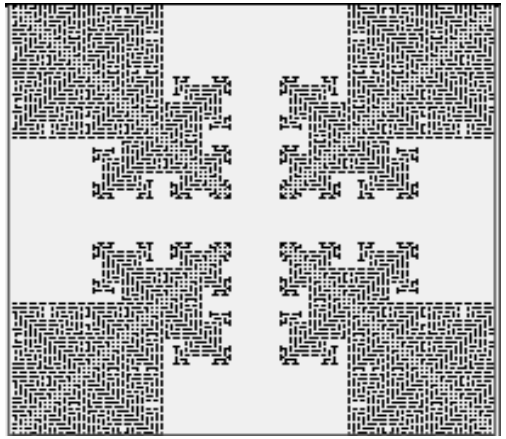

5
10
15
20
25
30
35
40
45
50
0
10
20
30
40
50

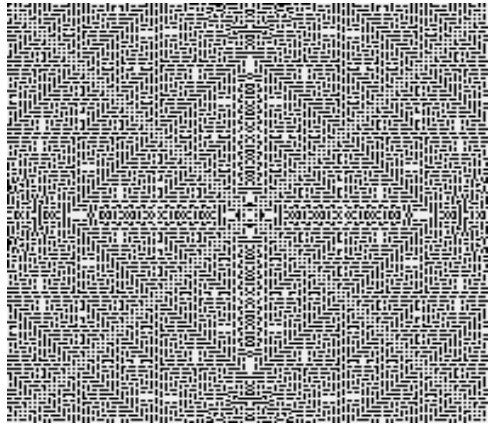

5
10
15
20
25
30
35
40
45
50
0
10
20
30
40
50

生命游戏







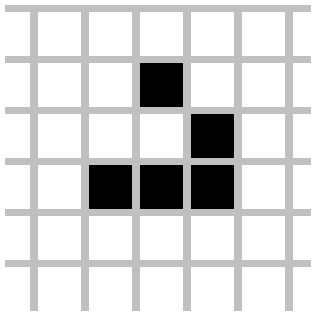


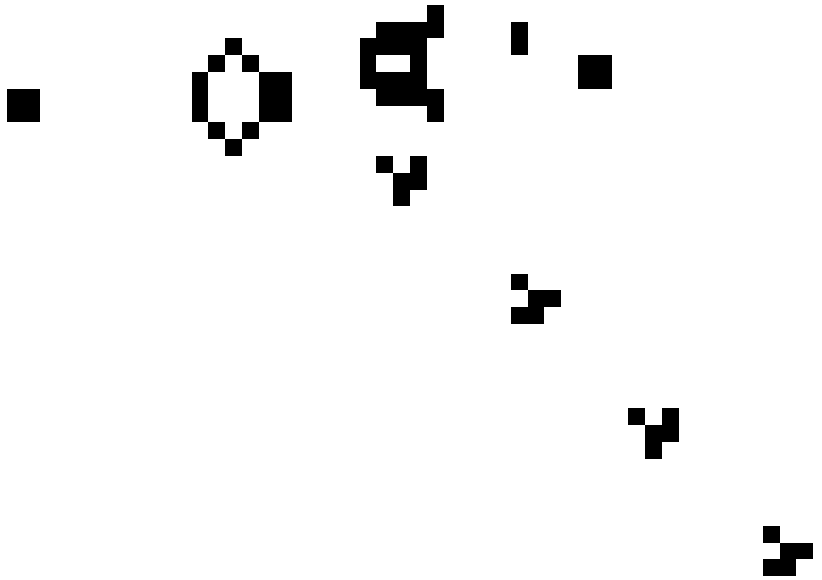
生命游戏

生命游戏是具有产生动态图案和动态结构能力的元胞自动机模型，它能产生丰富的、有趣的图案。生命游戏的优化与初始元胞状态值的分布有关，给定任意的初始状态分布。经过若干步的运算，有的图案会很快消失；而有的图案则固定不动，有的周而复始重复两个或几个图案，有的婉蜒而行；有的则保持图案定向移动，形似阅兵阵……。

生命游戏

最为著名的是滑翔机枪(Glider Gun)的图案，它可以周期性生产滑翔机发射器，每个发射器还能再发射滑翔机。





元胞自动机的定义与构成

概念

元胞

状态

元胞空间

邻居

规则

边界条件

元胞自动机（CA）是时间、空间、状态都离散，空间的相互作用及时间上因果关系皆局部的网格动力学模型。元胞自动机模型不同于一般的动力学模型，没有明确的方程形式，而是包含了一系列模型构造的规则，凡是满足这些规则的模型都可以算作是元胞自动机模型。因此，确切地说，元胞自动机是一类模型的总体、或者说是一个方法框架。

元胞自动机的定义与构成

概念

元胞

状态

元胞空间

邻居

规则

边界条件

元胞自动机实质上是定义在一个由具有离散、有限状态的元胞组成的元胞空间上，并按照一定的局部规则，在离散的时间维度上演化的动力学系统。

元胞自动机的定义与构成

概念

元胞

状态

元胞空间

邻居

规则

边界条件

元胞:元胞又可称为单元、细胞或基元，是元胞自动机的最基本的组成部分。元胞分布在离散的一维、二维或多维欧几里德空间的晶格点上。 具有以下特点：

1. 元胞自动机最基本的单元.

2. 元胞有记忆贮存状态的功能.

3. 所有元胞状态都安照元胞规则不断更新

元胞自动机的定义与构成

- 元胞的状态可以是二进制形式，如：（0，1），（生，死），（黑、白）等；也可以在一个有限整数集内S内取值：如交通领域的CA模型中，有时元胞状态可在[-（Vmax+1)～Vmax+1)]之间取值。

- 状态参量：严格意义上的CA只能有一个状态参量；但是，在实际应用中，可以具有多个状态参量。

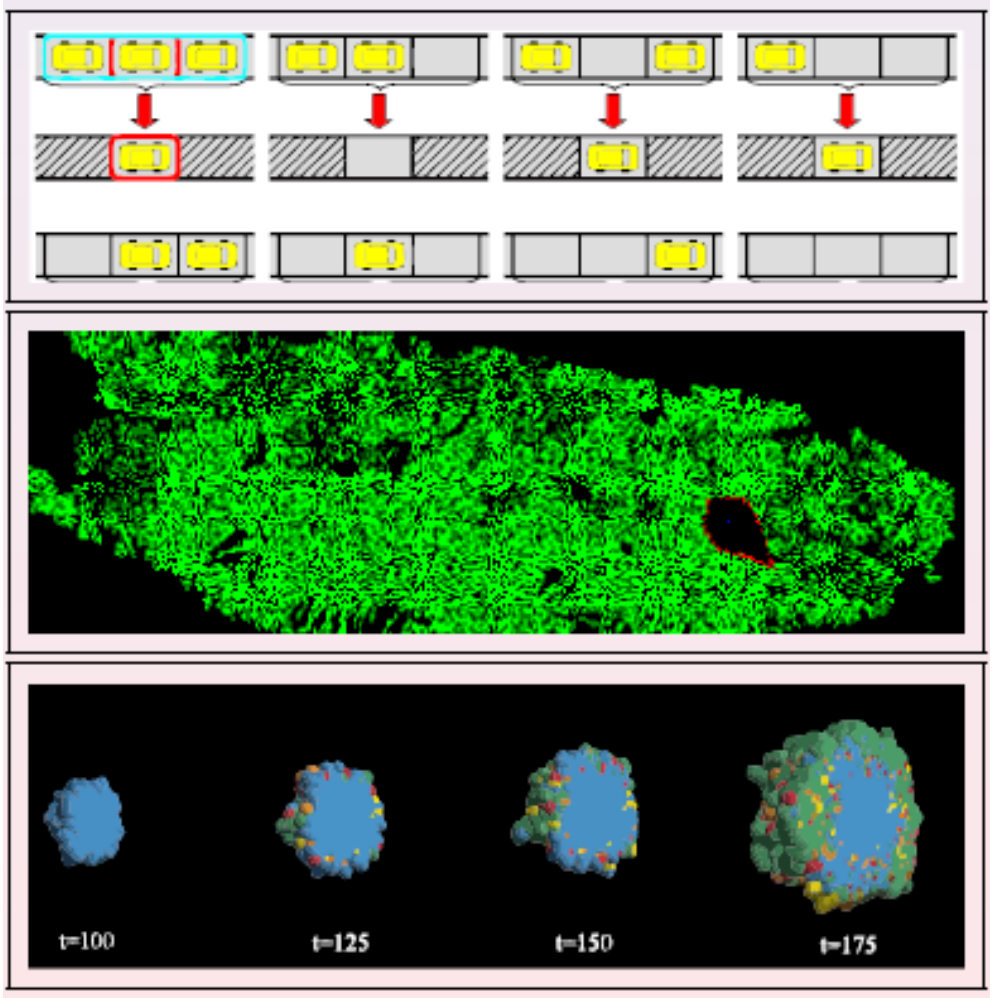
元胞自动机的定义与构成

元胞空间

元胞所分布在的空间网点集合就是这里的元胞空间。

理论上，它可以是任意维数的欧几里德空间规则划分。目前研究多集中在一维和二维元胞自动机上。对于一维元胞自动机。元胞空间的划分只有一种。而高维的元胞自动机，元胞空间的划分则可能有多种形式。对于最为常见的二维元胞自动机。二维元胞空间通常可按三角、四方或六边形三种网格排列。

元胞自动机的定义与构成

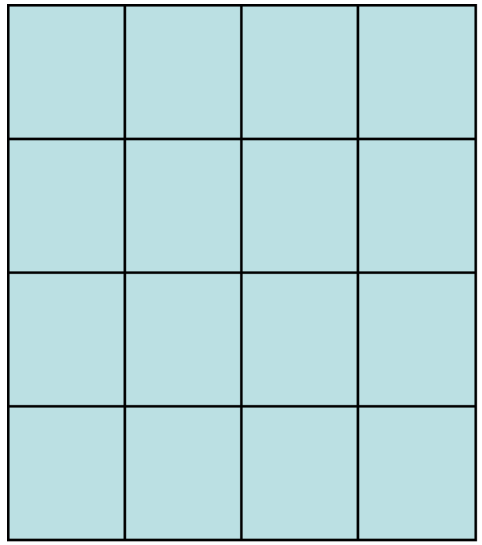


所有的元胞被置放在一个网格中：

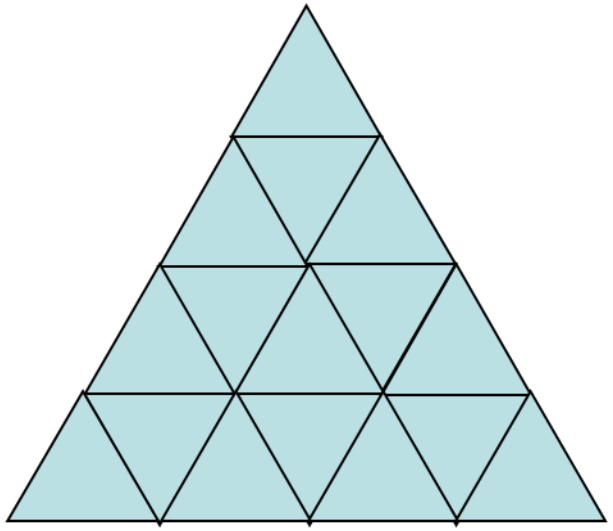
- 最简单的是一维网格，所有元胞排成一条线．模拟单车道的交通流时就是一维情况.
- 最常用的是二维网格，所有元胞排列在一个面内．如在模拟森林火灾时就是二维情况.
- 较复杂的可以是三维网格或者更高维．模拟人体内的三维肿瘤生长时就是三维情况.

元胞自动机的定义与构成

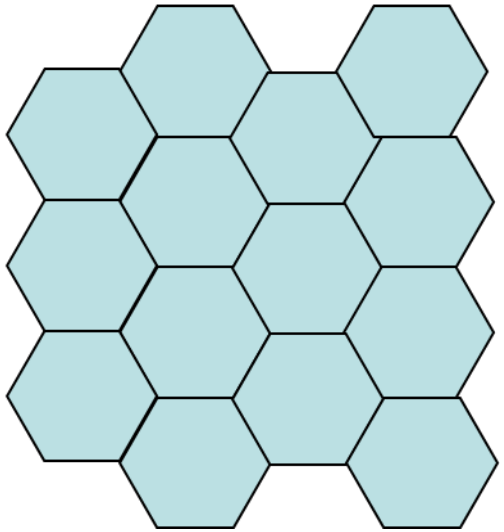
- 常用的二维元胞网格



正方形网格



三角形网格



六边形网格

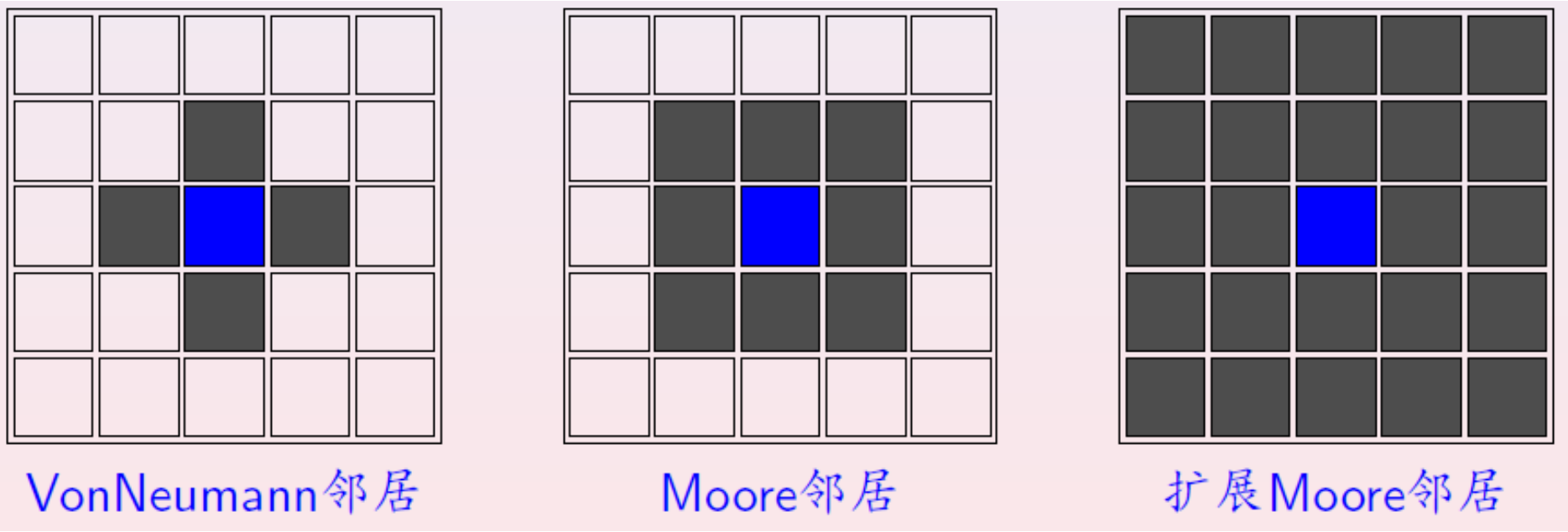
元胞自动机的定义与构成

网格类型	优点	缺点
三角形	拥有相对较少的邻居数目，易于处理复杂边界	在计算机的表达与显示不方便，需要转换为四方网格。
正方形	直观而简单，而且特别适合于在现有计算机环境下进行表达显示	不能较好地模拟各向同性的现象
正六边形	能较好地模拟各向同性的现象，因此，模型能更加自然而真实	在表达显示上较为困难、复杂

元胞自动机的定义与构成

对于一个元胞，在空间位置上与它相邻的元胞称为它的**邻元**（有时也称作邻居）。

所有由邻元组成的区域称为它的**邻域**。



元胞自动机的定义与构成

概念

元胞

状态

元胞空间

邻居

规则

边界条件

规则：元胞自动机关于元胞的局部演化规则(即元胞状态转换规则)有一个通用的描述:中心元胞的下一个状态由中心元胞的当前状态和其邻居的当前状态按照一定的规则确定。可见,元胞状态转换规则是一个动力学函数,其实质是一个状态转移函数,这个函数构造了一种简单的、离散的空间和时间范围的局部物理成分

元胞自动机的定义与构成

边界条件

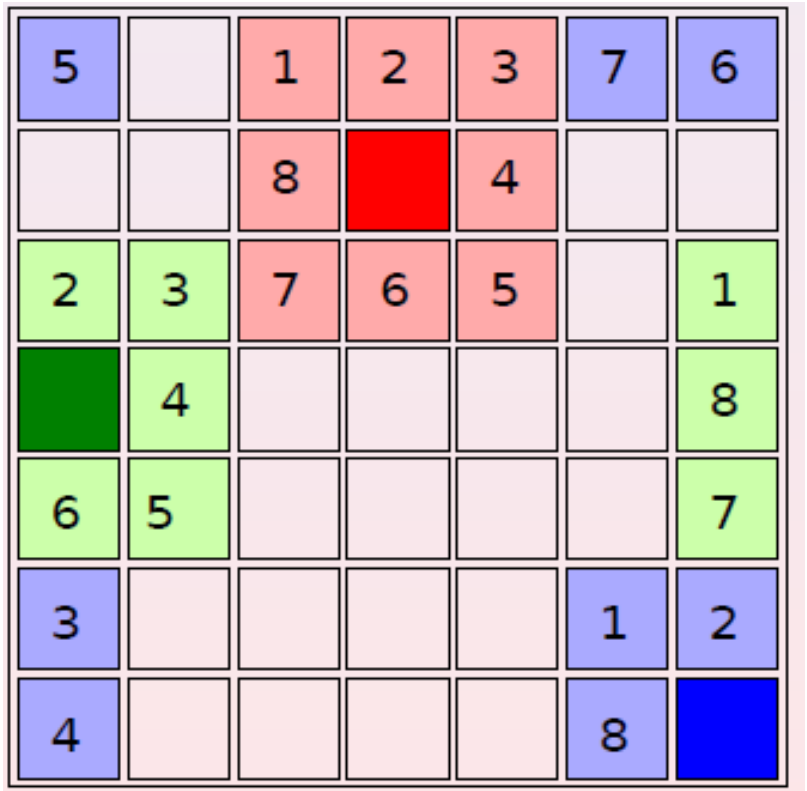
在理论上，元胞空间通常是在各维向上是无限延展的，这有利于在理论上的推理和研究。但是在实际应用过程中，我们无法在计算机上实现这一理想条件，因此，需要定义不同的边界条件。

三种类型:周期型、反射型和定值型。

周期型：是指相对边界连接起来的元胞空间。对于一维空间，元胞空间表现为一个首尾相接的“圈”。对于二维空间，上下相接，左右相接。而形成一个拓扑圆环面 ，形似车胎或甜点圈。周期型空间与无限空间最为接近，在理论探讨时，常以此类空间型作为试验。

元胞自动机的定义与构成

- 概念
- 元胞
- 状态
- 元胞空间
- 邻居
- 规则
- 边界条件



二维方网格周期边界条件

元胞自动机的定义与构成

概念

元胞

状态

元胞空间

邻居

规则

边界条件

反射型：指在边界外邻居的元胞状态是以边界为轴的镜面反射。

定值型：指所有边界外元胞均取某一固定常量，如0，1等。

在实际应用中，尤其是二维或更高维数的构模时，**可以相互结合**。

如在二维空间中，上下边界采用反射型，左右边界可采用周期型

元胞自动机的定义与构成

1) 同质性、齐性:同质性，每个元胞的变化服从相同的规律；齐性，元胞的分布方式相同，大小形状相同，空间分布规则整齐

2) 时间离散:元胞按一定规律分布在离散的元胞空间上

3) 空间离散:演化按等间隔时间分步进行,时间只取等步长的时刻点

4) 状态离散有限：元胞的状态只能取有限个离散值

5) 同步计算:元胞自动机的处理同步进行，适合并行计算

6) 时空局域性:元胞在t+1时刻的状态，取决于其周围半径r的邻域中的元胞在t时刻的状态，及所谓的时间、空间局限性

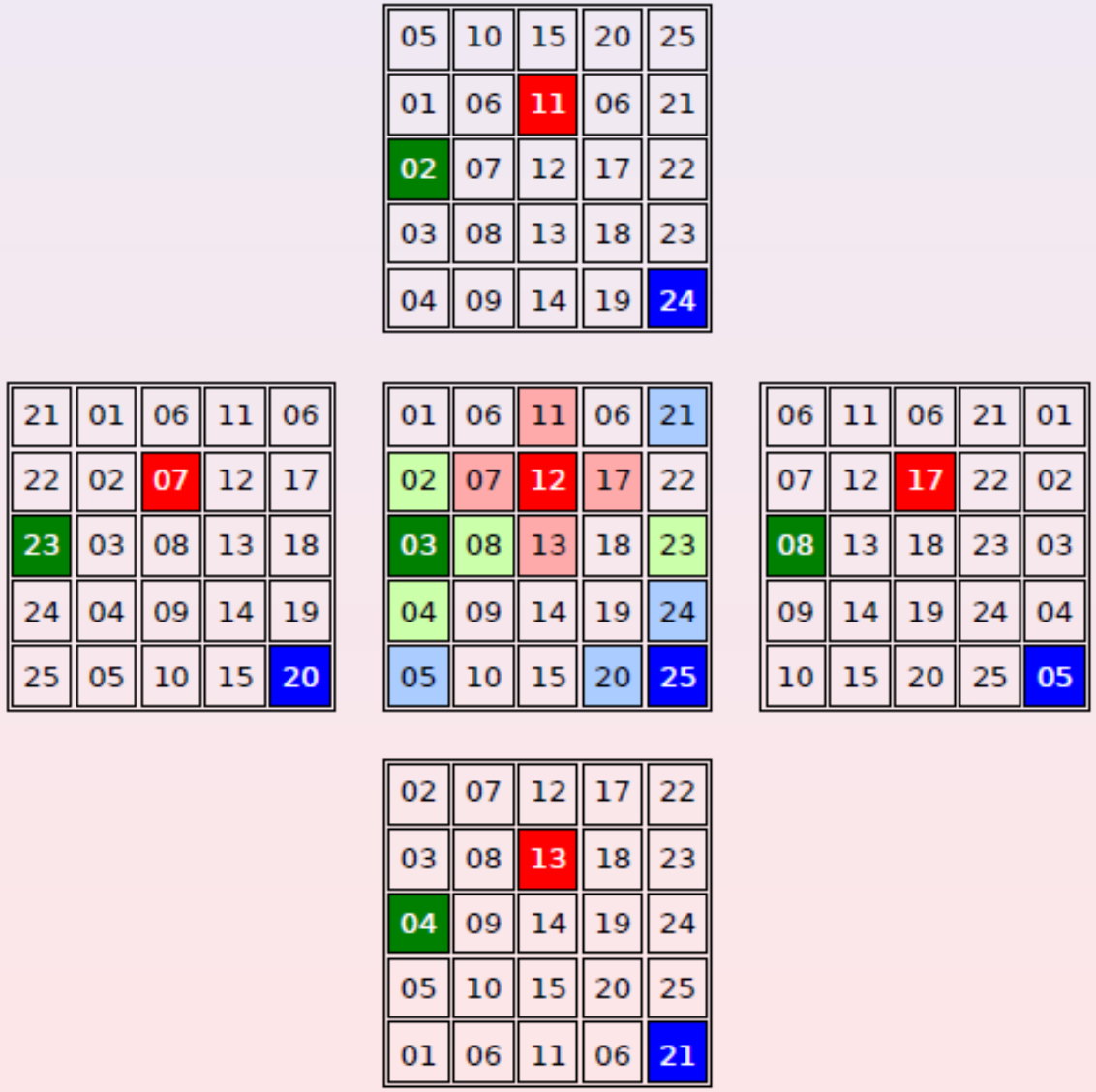
7) 维数高:在动力系统中一般讲变量的个数称为维数。由于任何完备元胞自动机的元胞空间是定义在一维、二维或多维空间上的无限集，每个元胞的状态便是这个动力学系统的变量。因此，元胞自动机是一类无穷维动力系统。

森林火空

森林火灾的元胞自动机模型有三种状态：空位，燃烧着的树木及树木. 则某元胞下时刻状态由该时刻本身的状态和周围四个邻居(VonNeumann邻居)的状态决定状态由构成及规则：

- 若某树木元胞的4个邻居中有燃烧着的，那么该元胞下时刻的状态是燃烧.
- 一个燃烧着的元胞在下一时刻变成空位的.
- 所有树木元胞以一个低概率开始烧(以模拟闪电引起的火灾).
- 所有空元胞以一个低概率变为树木(以模拟新的树木生长).

森林火灾



若A矩阵有n行n列，则矩阵A的四个邻居分别为

- **上**: A([n, 1:n-1], :)
- **左**: A(:, [n, 1:n-1])
- **右**: A(:, [2:n, 1])
- **下**: A([2:n, 1], :)

思考:对于周围八个邻居(Moore型邻居)的情况，矩阵A的各个邻居的表达式为什么?

森林火灾

```
01  n = 300;                                  定义表示森林的矩阵大小
02  Plight  = .000005; Pgrowth = .01;         定义闪电和生长的概率
03  UL = [n 1:n-1];  DR = [2:n 1];            定义上左，下右邻居
04  veg=zeros(n,n);                           初始化表示森林的矩阵
05  imh = image(cat(3,veg,veg,veg));          可视化表示森林的矩阵
06  % veg = empty=0 burning=1 green=2
07  for i=1:3000                              主循环开始
08      sum =          (veg(UL,:)==1) + ...
09           (veg(:,UL)==1)    +    (veg(:,DR)==1) + ...
10                     (veg(DR,:)==1);        求上下左右四个邻居和
11                                            根据规则更新森林矩阵：树 = 树 - 着火的树 + 新生的树
12      veg = 2*(veg==2) - ...
13            ( (veg==2) & (sum>0|(rand(n,n)<Plight)) ) + ...
14            2*((veg==0) & rand(n,n)<Pgrowth) ;
15      set(imh, 'cdata', cat(3,(veg==1),(veg==2),zeros(n)) )
16      drawnow                               可视化表示森林的矩阵
17  end                                       主循环结束
```

森林火灾

```
01  n = 300;
02  Plight  = .000005; Pgrowth = .01;
03  UL = [n 1:n-1];  DR = [2:n 1];
04  veg=zeros(n,n);
05  imh = image(cat(3,veg,veg,veg));
06  % veg = empty=0 burning=1 green=2
07  for i=1:3000
08      sum =          (veg(UL,:)==1) + ...
09           (veg(:,UL)==1)    +    (veg(:,DR)==1) + ...
10                     (veg(DR,:)==1);
11
12      veg = 2*(veg==2) - ...
13            ( (veg==2) & (sum>0|(rand(n,n)<Plight)) ) + ...
14            2*((veg==0) & rand(n,n)<Pgrowth) ;
15      set(imh, 'cdata', cat(3,(veg==1),(veg==2),zeros(n)) )
16      drawnow
17  end
```

森林火灾

```
01  n = 300;
02  Plight  = .000005; Pgrowth = .01;
03  UL = [n 1:n-1];  DR = [2:n 1];
04  veg=zeros(n,n);
05  imh = image(cat(3,veg,veg,veg));
06  % veg = empty=0 burning=1 green=2
07  for i=1:3000
08      sum =          (veg(UL,:)==1) + ...
09           (veg(:,UL)==1)    +    (veg(:,DR)==1) + ...
10                     (veg(DR,:)==1);
11
12      veg = 2*(veg==2) - ...
13            ( (veg==2) & (sum>0|(rand(n,n)<Plight)) ) + ...
14            2*((veg==0) & rand(n,n)<Pgrowth) ;
15      set(imh, 'cdata', cat(3,(veg==1),(veg==2),zeros(n)) )
16      drawnow
17  end
```

森林火灾

```
01  n = 300;
02  Plight  = .000005; Pgrowth = .01;
03  UL = [n 1:n-1];  DR = [2:n 1];
04  veg=zeros(n,n);
05  imh = image(cat(3,veg,veg,veg));
06  % veg = empty=0 burning=1 green=2
07  for i=1:3000
08      sum =          (veg(UL,:)==1) + ...
09           (veg(:,UL)==1)   +   (veg(:,DR)==1) + ...
10                     (veg(DR,:)==1);
11
12      veg = 2*(veg==2) - ...
13            ( (veg==2) & (sum>0|(rand(n,n)<Plight)) ) + ...
14            2*((veg==0) & rand(n,n)<Pgrowth) ;
15      set(imh, 'cdata', cat(3,(veg==1),(veg==2),zeros(n)) )
16      drawnow
17  end
```

森林火灾

```
01  n = 300;
02  Plight  = .000005; Pgrowth = .01;
03  UL = [n 1:n-1];  DR = [2:n 1];
04  veg=zeros(n,n);
05  imh = image(cat(3,veg,veg,veg));
06  % veg = empty=0 burning=1 green=2
07  for i=1:3000
08      sum =          (veg(UL,:)==1) + ...
09           (veg(:,UL)==1)   +   (veg(:,DR)==1) + ...
10                     (veg(DR,:)==1);
11
12      veg = 2*(veg==2) - ...
13            ( (veg==2) & (sum>0|(rand(n,n)<Plight)) ) + ...
14            2*((veg==0) & rand(n,n)<Pgrowth) ;
15      set(imh, 'cdata', cat(3,(veg==1),(veg==2),zeros(n)) )
16      drawnow
17  end
```

森林火灾

```
01  n = 300;
02  Plight  = .000005; Pgrowth = .01;
03  UL = [n 1:n-1];  DR = [2:n 1];
04  veg=zeros(n,n);
05  imh = image(cat(3,veg,veg,veg));
06  % veg = empty=0 burning=1 green=2
07  for i=1:3000                                主循环开始
08      sum =          (veg(UL,:)==1) + ...
09           (veg(:,UL)==1)    +    (veg(:,DR)==1) + ...
10                     (veg(DR,:)==1);          求上下左右四个邻居和
11      根据规则更新森林矩阵: 树 = 树 - 着火的树 + 新生的树
12      veg = 2*(veg==2) - ...
13            ( (veg==2) & (sum>0|(rand(n,n)<Plight)) ) + ...
14            2*((veg==0) & rand(n,n)<Pgrowth) ;
15      set(imh, 'cdata', cat(3,(veg==1),(veg==2),zeros(n)) )
16      drawnow
17  end                                         主循环结束
```

森林火灾

```
01  n = 300;
02  Plight  = .000005; Pgrowth = .01;
03  UL = [n 1:n-1];  DR = [2:n 1];
04  veg=zeros(n,n);
05  imh = image(cat(3,veg,veg,veg));
06  % veg = empty=0 burning=1 green=2
07  for i=1:3000
08      sum =          (veg(UL,:)==1) + ...
09           (veg(:,UL)==1)   +   (veg(:,DR)==1) + ...
10                     (veg(DR,:)==1);
11      根据规则更新森林矩阵: 树 = 树 - 着火的树 + 新生的树
12      veg = 2*(veg==2) - ...
13            ( (veg==2) & (sum>0|(rand(n,n)<Plight)) ) + ...
14            2*((veg==0) & rand(n,n)<Pgrowth) ;
15      set(imh, 'cdata', cat(3,(veg==1),(veg==2),zeros(n)) )
16      drawnow
17  end
```

定义表示森林的矩阵大小
定义闪电和生长的概率
定义上左、下右邻居
初始化表示森林的矩阵
可示化表示森林的矩阵
主循环开始
求上下左右四个邻居和
可示化表示森林的矩阵
主循环结束

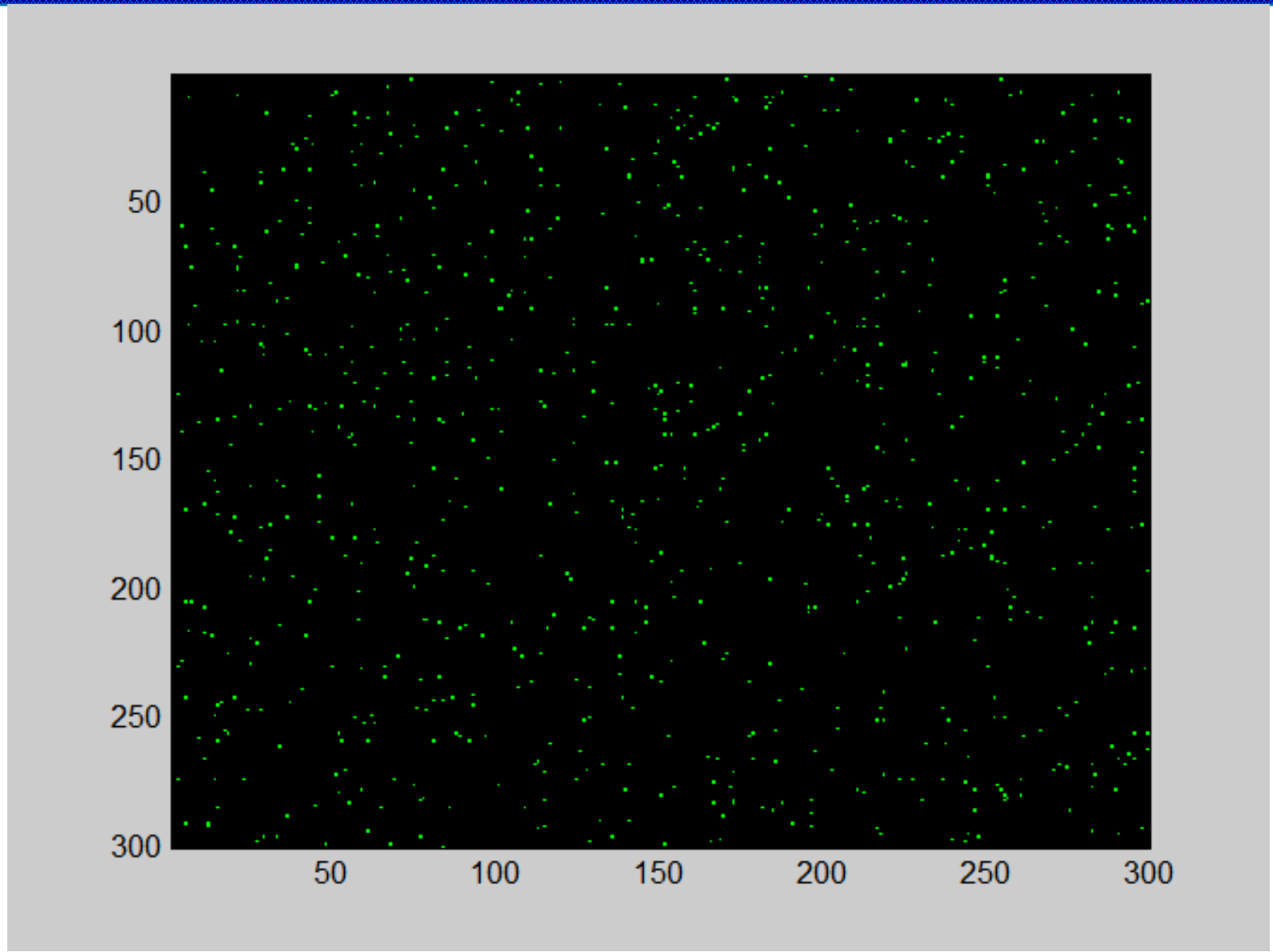
森林火灾

```
01  n = 300;
02  Plight  = .000005; Pgrowth = .01;
03  UL = [n 1:n-1];  DR = [2:n 1];
04  veg=zeros(n,n);
05  imh = image(cat(3,veg,veg,veg));
06  % veg = empty=0 burning=1 green=2
07  for i=1:3000
08      sum =          (veg(UL,:)==1) + ...
09           (veg(:,UL)==1)   +   (veg(:,DR)==1) + ...
10                     (veg(DR,:)==1);
11      根据规则更新森林矩阵: 树 = 树 - 着火的树 + 新生的树
12      veg = 2*(veg==2) - ...
13            ( (veg==2) & (sum>0|(rand(n,n)<Plight)) ) + ...
14            2*((veg==0) & rand(n,n)<Pgrowth) ;
15      set(imh, 'cdata', cat(3,(veg==1),(veg==2),zeros(n)) )
16      drawnow
17  end
```

森林火灾

```
01  n = 300;
02  Plight  = .000005; Pgrowth = .01;
03  UL = [n 1:n-1];  DR = [2:n 1];
04  veg=zeros(n,n);
05  imh = image(cat(3,veg,veg,veg));
06  % veg = empty=0 burning=1 green=2
07  for i=1:3000
08      sum =          (veg(UL,:)==1) + ...
09           (veg(:,UL)==1)   +   (veg(:,DR)==1) + ...
10                     (veg(DR,:)==1);
11      根据规则更新森林矩阵: 树 = 树 - 着火的树 + 新生的树
12      veg = 2*(veg==2) - ...
13            ( (veg==2) & (sum>0|(rand(n,n)<Plight)) ) + ...
14            2*((veg==0) & rand(n,n)<Pgrowth) ;
15      set(imh, 'cdata', cat(3,(veg==1),(veg==2),zeros(n)) )
16      drawnow                          可示化表示森林的矩阵
17  end
```

森林火灾



交通流中的应用

- 1986年，Cremer和Ludwig初次将元胞自动机运用到车辆交通的研究中。
  - 交通流的元胞自动机模型大致可分为两大类：
    - 研究高速公路交通的模型（以NS模型为代表）；
    - 研究城市网络交通的模型（以BML模型为代表）
  - 这两类模型是以Wolfram命名的184号模型为基础发展而来的。

初等元胞自动机

- 最简单的元胞自动机模型
- 一维元胞自动机
- 状态集S只有两个元素，可用0或1表示
- 邻居半径为1

初等元胞自动机

- 对于每个元胞，考虑到前后两个邻居的状态，一共有2×2×2＝8种情况
- 对于每一种情况，该元胞下一个时刻可以取2种不同的状态，共有2^8=256种对应规则，因此共有256种不同的元胞自动机。

t	111	110	101	100	011	010	001	000
t+1	0	1	1	0	1	1	1	0
	$\alpha_7=0$	$\alpha_6=1$	$\alpha_5=1$	$\alpha_4=0$	$\alpha_3=1$	$\alpha_2=1$	$\alpha_1=1$	$\alpha_0=0$

命名规则：二进制数 $\alpha_7\ \alpha_6\ \alpha_5\ \alpha_4\ \alpha_3\ \alpha_2\ \alpha_1\ \alpha_0$ 转化为十进制 $R=\sum_{i=0}^{7}\left(\alpha_i\times 2^i\right)$

$$R=\sum_{i=0}^{7}\left(\alpha_i\times 2^i\right)=0\times 2^7+1\times 2^6+1\times 2^5+0\times 2^4+1\times 2^3+1\times 2^2+1\times 2^1+0\times 2^0=110$$

初等元胞自动机

256种初等元胞自动机规则

t	111	110	101	100	011	010	001	000	
	0	0	0	0	0	0	0	0	rule 0
	0	0	0	0	0	0	0	1	rule 1
	0	0	0	0	0	0	1	0	rule 2
	0	0	0	0	0	0	1	1	rule 3
t+1	…	…	…	…	…	…	…	…	…
	1	0	1	1	1	0	0	0	rule 184
	…	…	…	…	…	…	…	…	…
	1	1	1	1	1	1	1	0	rule 254
	1	1	1	1	1	1	1	1	rule 255

初等元胞自动机

Stephen Wolfram 对初等元胞自动机的分为四类：

平稳型:自任何初始状态开始,经过一定时间运行后，元胞空间趋于一个空间平稳的构形，这里空间平稳即指每一个元胞处于固定状态。不随时间变化而变化。

周期型:经过一定时间运行后，元胞空间趋于一系列简单的固定结构(Stable Paterns)或周期结构(Periodical Patterns)。

混沌型:自任何初始状态开始，经过一定时间运行后，元胞自动机表现出混沌的非周期行为，所生成的结构的统计特征不再变止，通常表现为分形分维特征。

复杂型:出现复杂的局部结构，或者说是局部的混沌，其中有些会不断地传播。

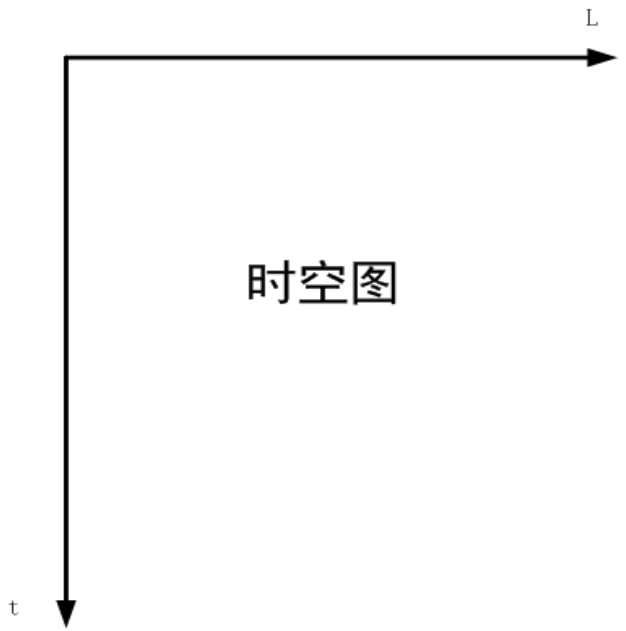
初等元胞自动机

初等元胞自动机的时空图

0——白色　1——黑色　元胞数L=200

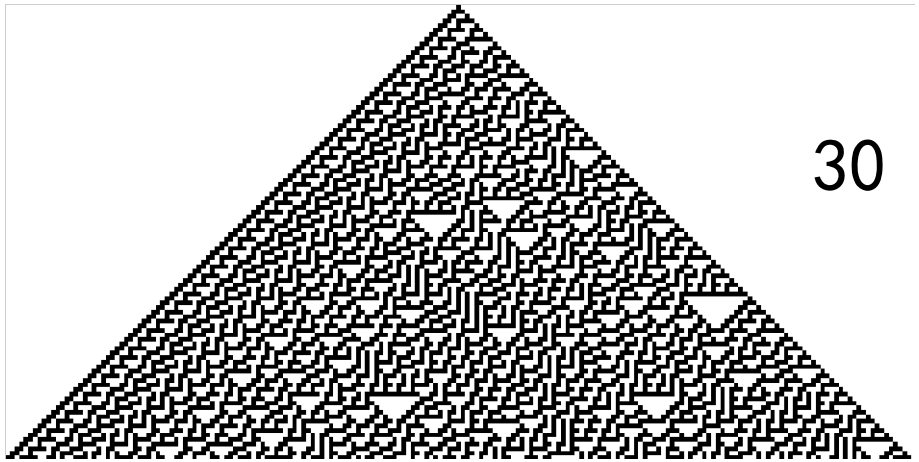
采用周期边界，初值取第50个格子为1，对每个规则演化100步。

如下结构时空图

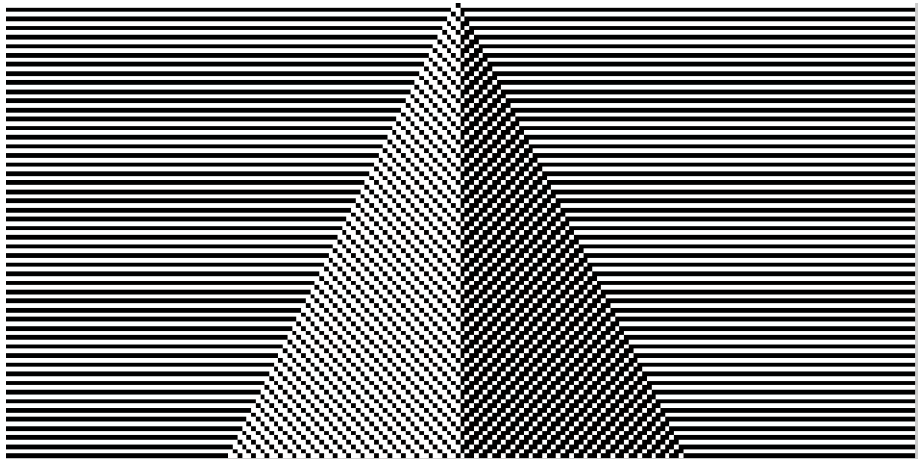


初等元胞自动机

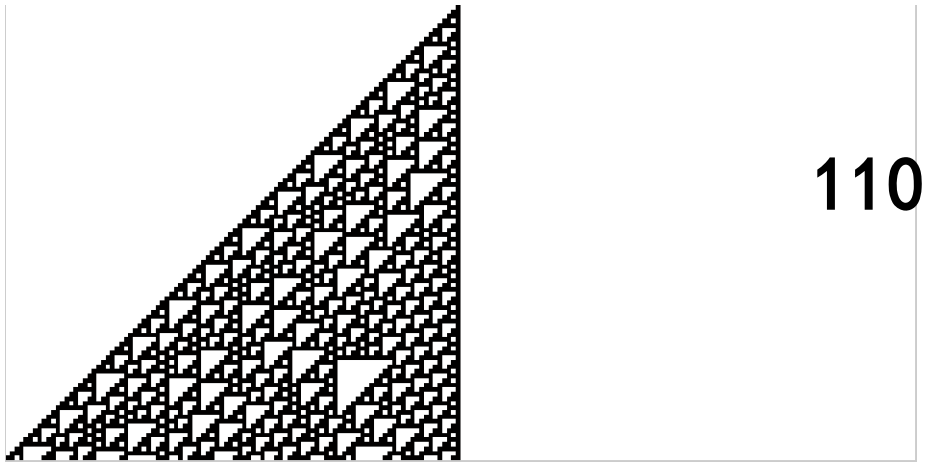
几种典型的元胞自动机



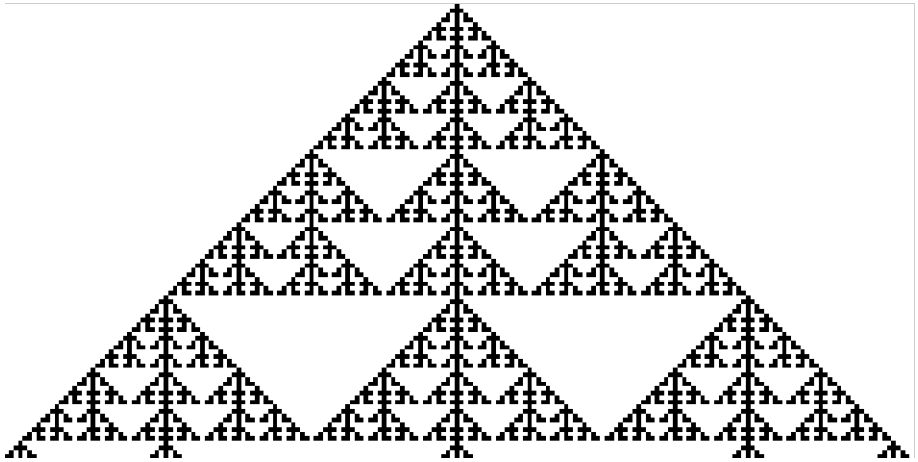
30



57



110



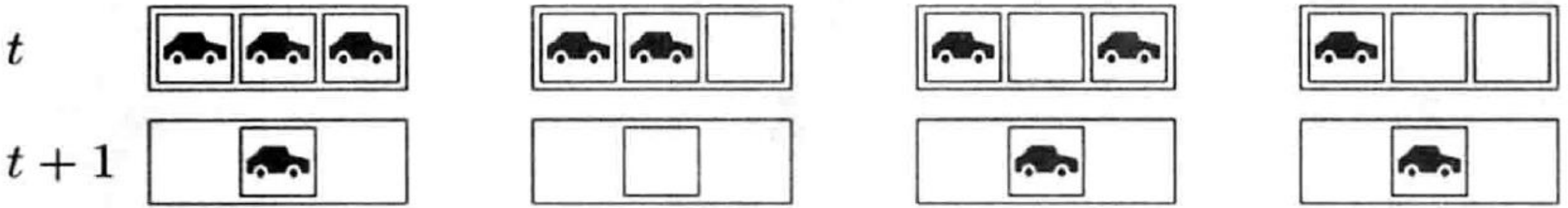
150

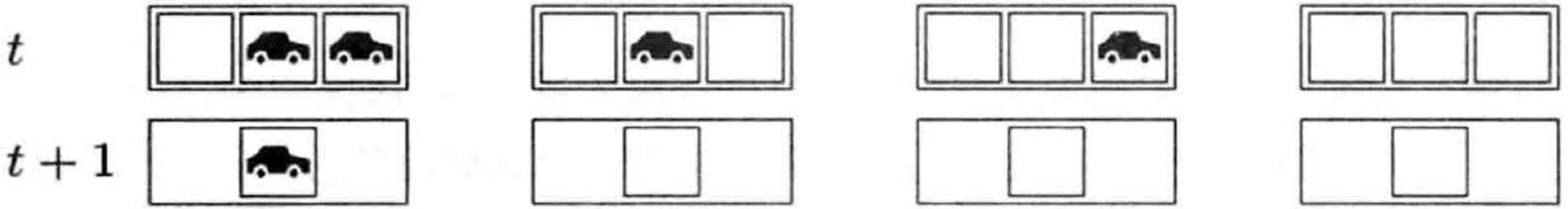
初等元胞自动机

```
function ca(R,w,n)
%R为规则编号，w为元胞个数，n为演化时间步数
a=zeros(n,w);
a(1,w/2)=1;
rule = bitget(R,8:-1:1);
rule = fliplr(rule);

l=[w 1:w-1];
r=[2:w 1];
index = zeros(1,w);

for i=2:n
    index = a(i-1, l).*4 + a(i-1,:).*2 + a(i-1, r)+1;
    a(i,:)=rule(index);
end
a = 1-a;
imshow(a)
```

184号初等元胞自动机　10111000 ⟶ 184

t	111	110	101	100	011	010	001	000
t+1	1	0	1	1	1	0	0	0

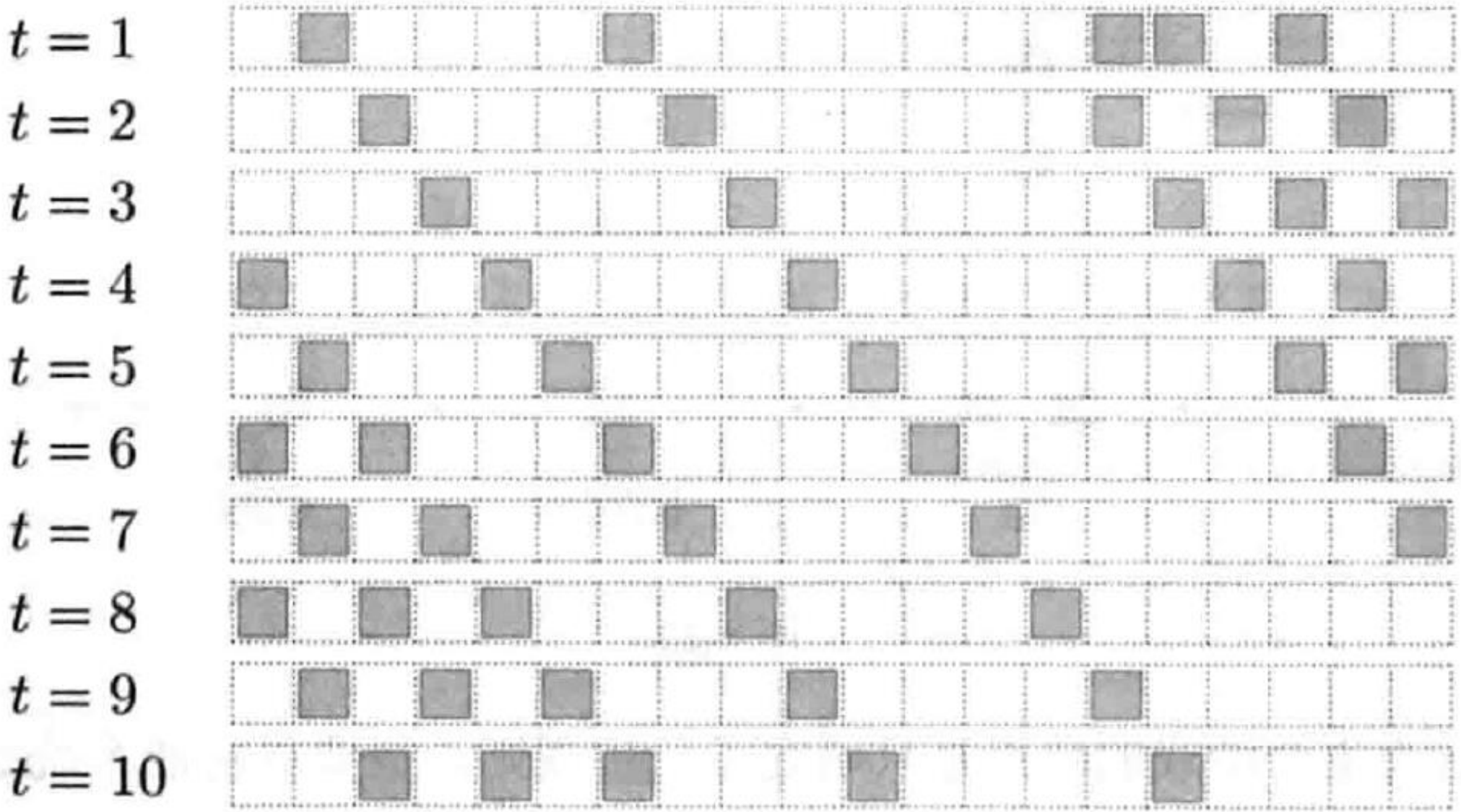




184号模型

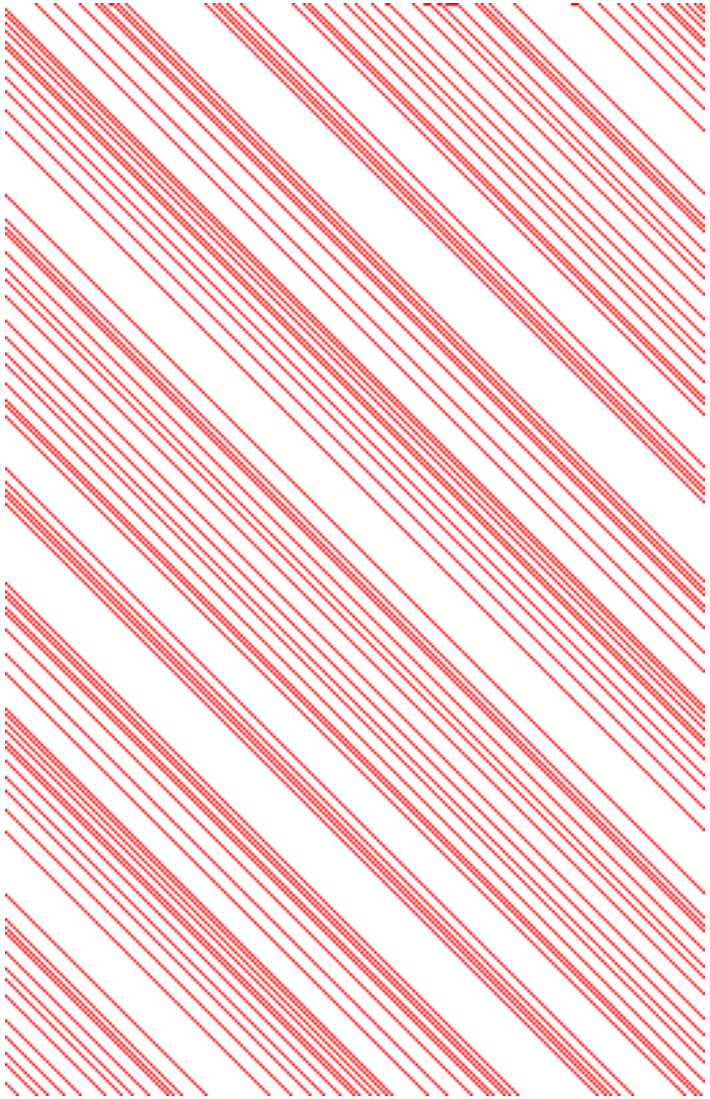
- ✓ 道路被划分为等距格子，每个格点表示一个元胞；
- ✓ 某个时刻，元胞或者是空的，或者被一辆车占据；
- ✓ 所有车辆的行进方向都是一致的（如向右）；
- ✓ 在每一个时间步内：若第n辆车的前方元胞是空的，则该车可以向前行驶一步；
- ✓ 若前面的元胞被另一辆车n+1所占据，即使第n+1辆车在本时间步内离开此元胞，第n辆车也停在原地不动；
- ✓ 整个系统采用周期性边界条件以确保车辆数守恒。

184号模型



184号模型

时空分布图



横轴：空间

纵轴：时间

NS模型

- 作为对184号规则的推广，Nagel和Schreckberg在1992年提出了一个模拟车辆交通的元胞自动机模型，即NS模型（也有人称它为NaSch模型）。
  - 时间、空间和车辆速度都被整数离散化
  - 道路被划分为等距离的离散的格子，即元胞每个元胞或者是空的，或者被一辆车所占据
  - 车辆的速度可以在（0～Vmax）之间取值

NS模型

- 在时刻t到时刻t+1的过程中按照下面的规则进行更新：

  - a) 加速，$v_n \to \min(v_n + 1, v_{max})$；
    对应于现实中司机期望以最大速度行驶的特性。

  - b) 减速，$v_n \to \min(v_n, d_n)$；
    驾驶员为了避免和前车发生碰撞而采取减速的措施。

NS模型

- 在时刻t到时刻t+1的过程中按照下面的规则进行更新：

  - c) 随机慢化，以概率 p ， $v_n \to \max(v_n - 1, 0)$ ；
    由各种不确定因素 (如路面状况不好、驾驶员的不同心态等等) 造成的车辆减速。

  - d) 运动， $x_n \to x_n + v_n$ 。
    车辆按照调整后的速度向前行驶。

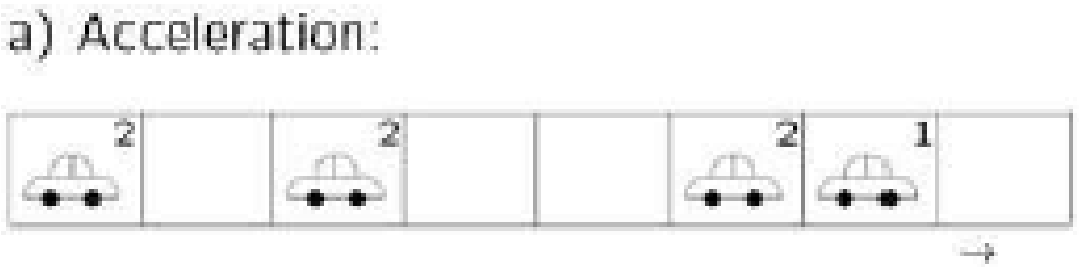
    这里， x_n ， v_n 分别表示 n 车的位置和速度，

    $d_n = x_{n+1} - x_n - 1$ 表示 n 车和前车 $n+1$ 的间距

NS模型

更新过程图示：

$v_{\max} = 2$

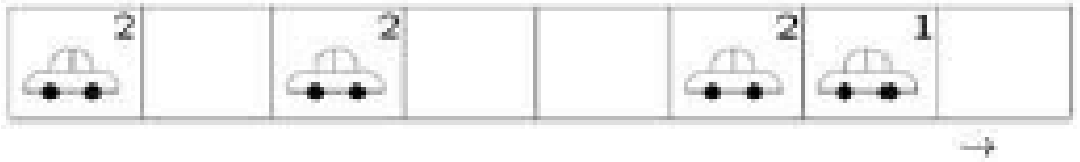


NS模型

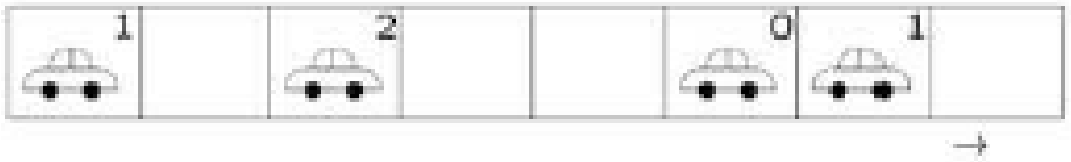
更新过程图示：

$v_{\max} = 2$

a) Acceleration:



b) Braking:

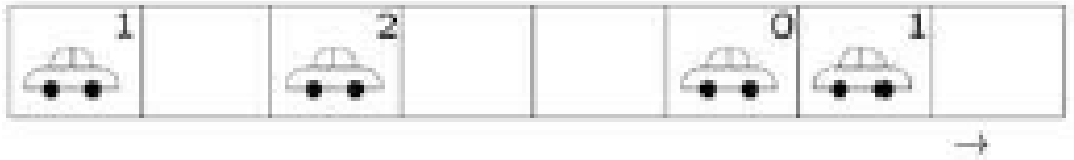


NS模型

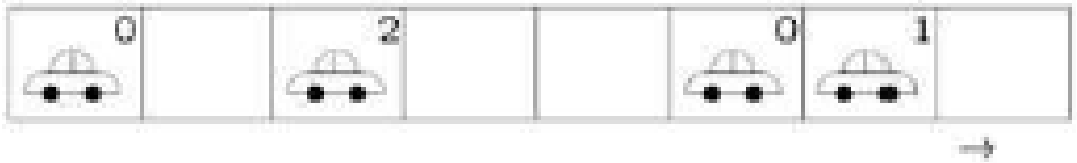
更新过程图示：

$v_{\max} = 2$

b) Braking:



c) Randomization ($p = 1/4$):

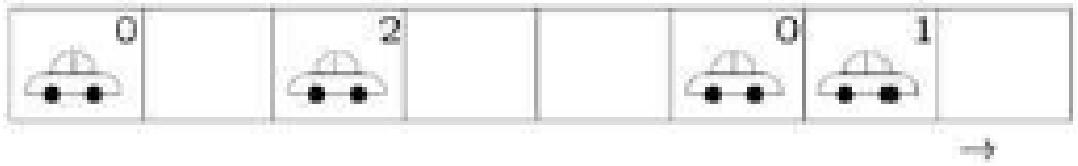


NS模型

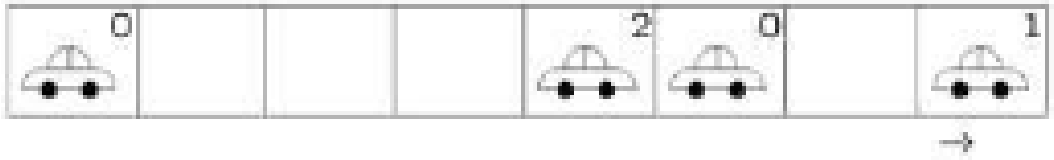
更新过程图示：

$v_{\max} = 2$

c) Randomization ($p = 1/4$):



d) Driving (= configuration at time $t + 1$):



NS模型

- 周期性边界条件
  - 在每次更新结束后，我们要监测道路上头车的位置X_lead，如果X_lead>L_road，那么这两车将从道路的另一端进入系统，变为道路上的尾车，并且X_lead = X_lead-L_road, V_last= V_lead。
- 开口边界条件
  - 假设道路最左边的元胞对应于X=1，并且道路的入口端包含V_max个元胞，也就是说，车辆可以从元胞 (1, 2,…, V_max)进入到道路中。在t → t+1时刻，当道路上的车辆更新完成后，监测道路上的头车和尾车的位置X_lead和X_last。如果X_last>V_max，则一辆速度为V_max的车将以概率α进入元胞min[X_last-V_max, V_max]。在道路的出口处，如果X_lead>L_road，那么道路上的头车以概率β驶出路段，而紧跟其后的第二辆车成为新的头车。

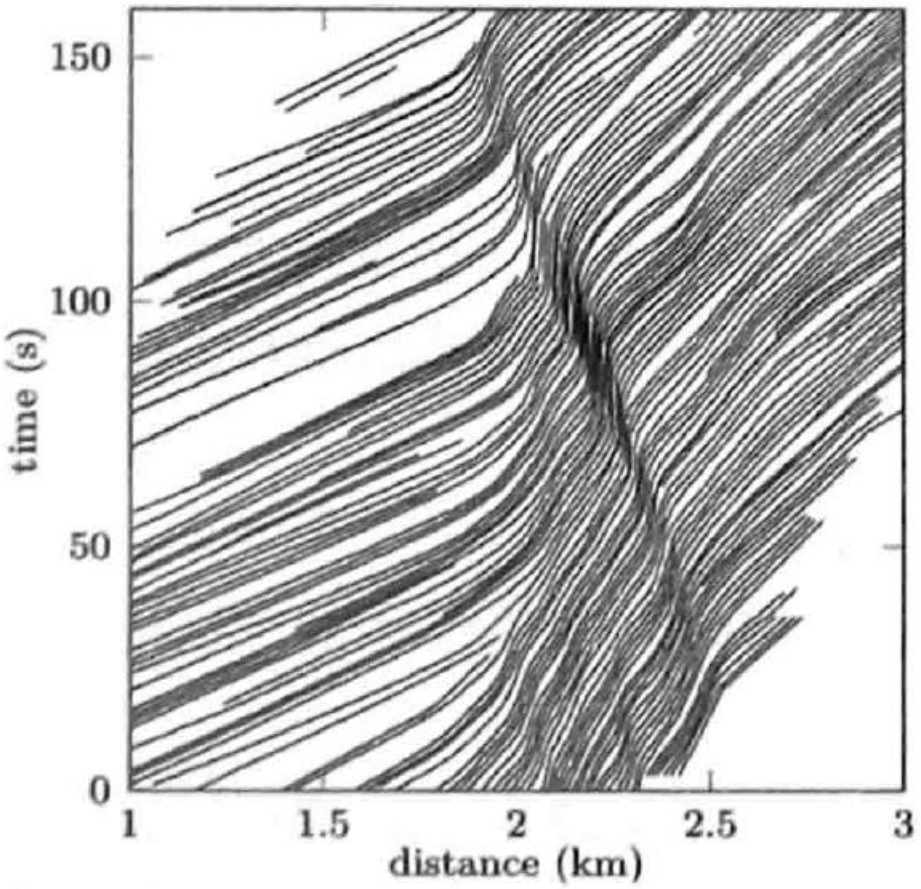
NS模型

- ✓ 与184号模型相比，NS模型的主要改进之处是引入了慢化概率和最大车速不再是1。

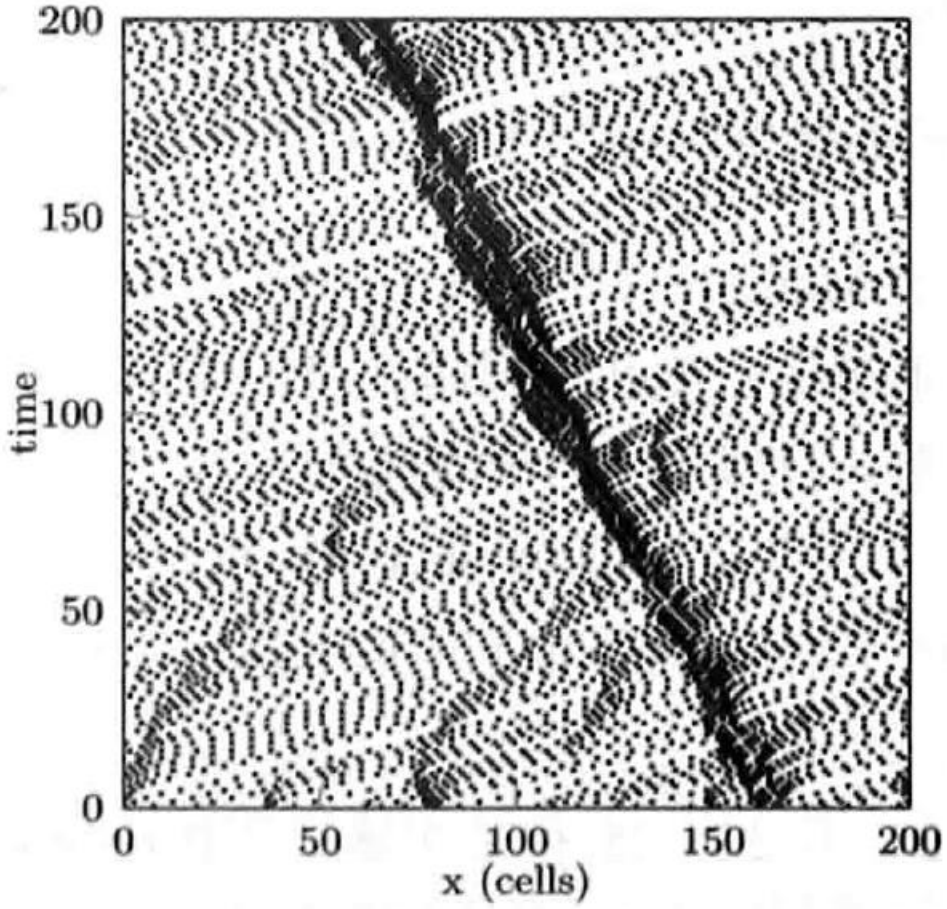
- ✓ NS模型虽然具有十分简单的形式，但却可以描述一些实际交通现象。比如NS模型可以模拟出自发产生的堵塞现象以及拥挤交通情况下的时走时停波等。

NS模型

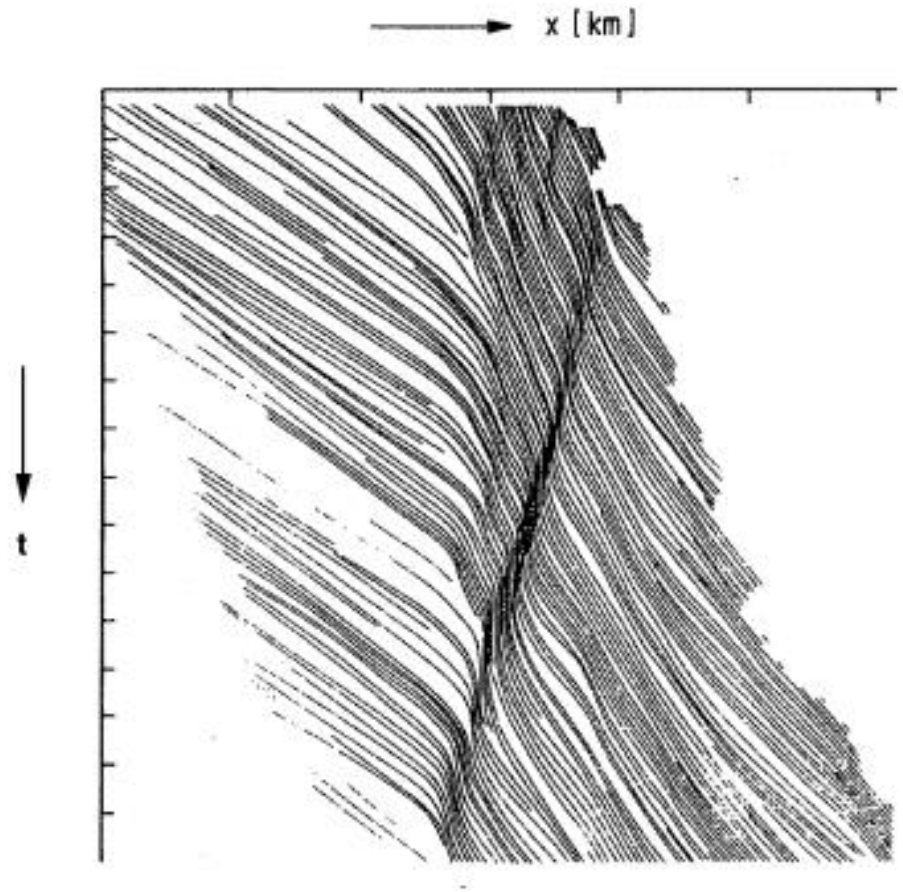
航拍数据 (Treiterer&Myers, 1967)

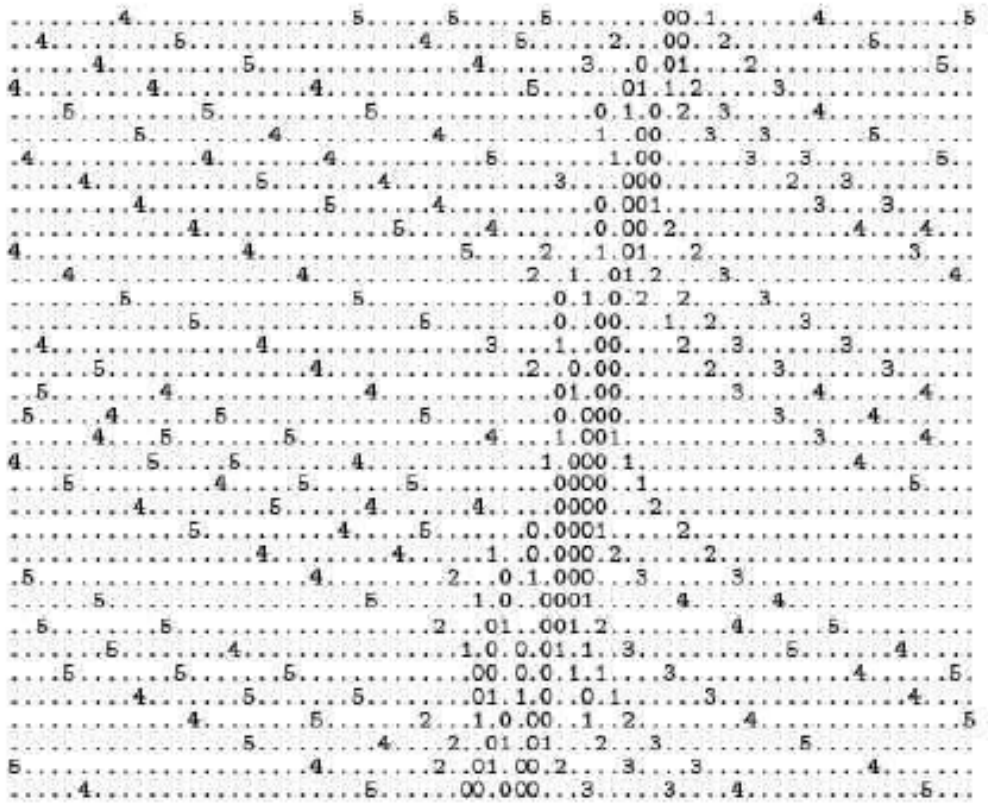


CA 计算结果 ($p_{\text{brake}} = 0.15$)



NS模型





堵塞的形成

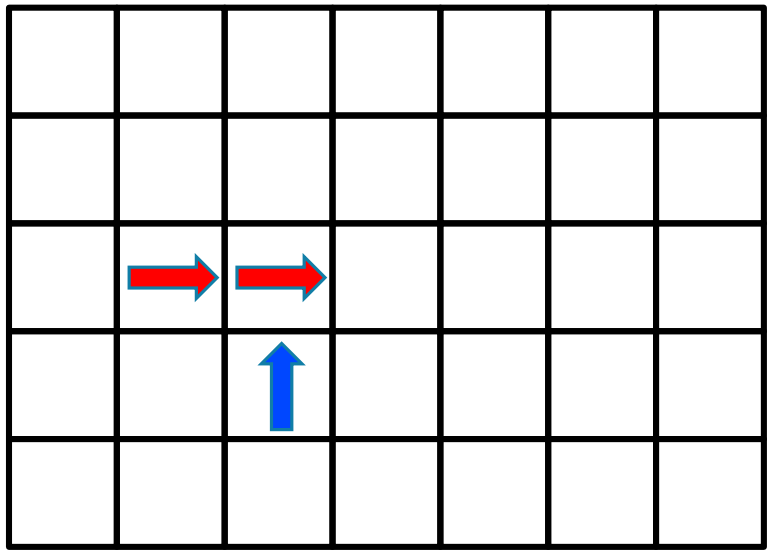
BML模型

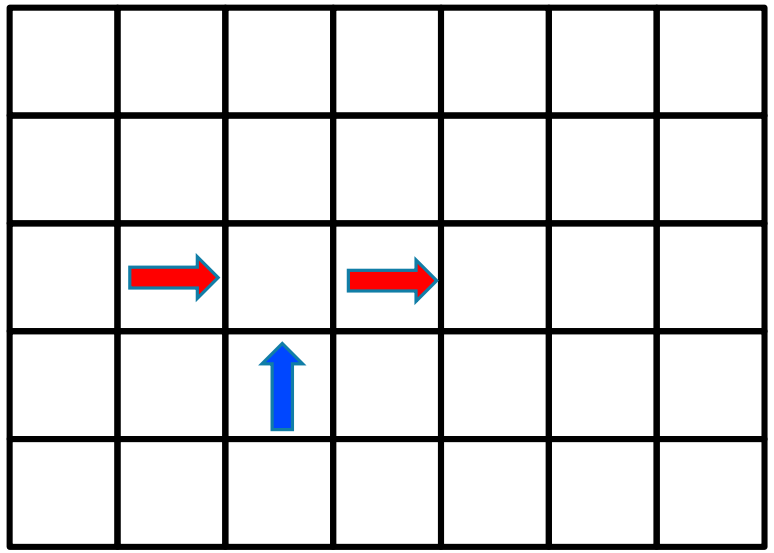
1992年，Biham, Middleton, Levine提出了第一个二维交通流元胞自动机模型。

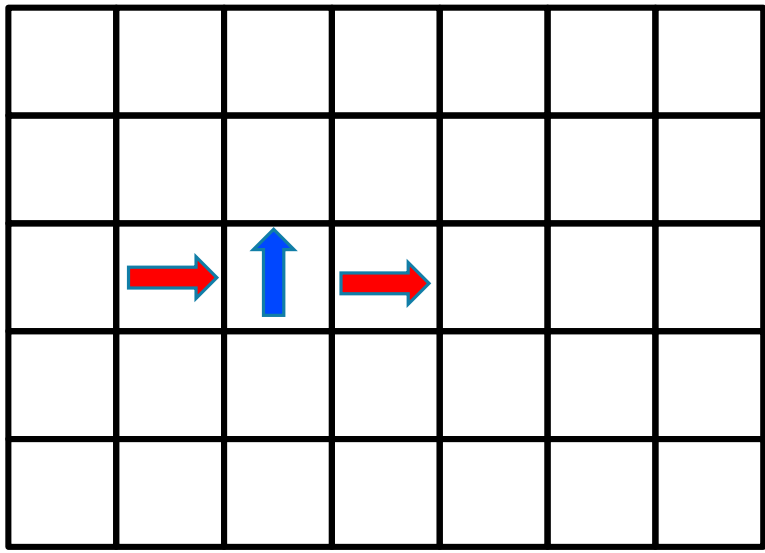
- ✓ 模型定义于一个N×N的方形格点的网络上；
- ✓ 每一个格点具有三种状态：

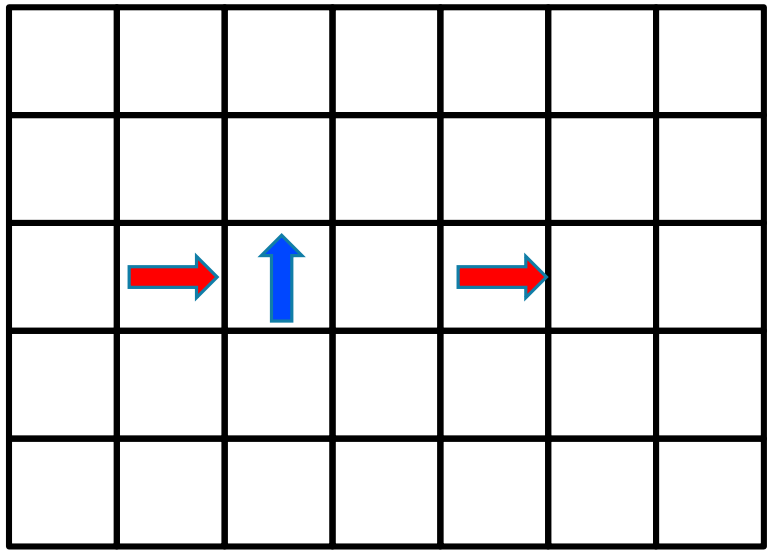
  1）没有车辆；2）被一辆向北行驶的车辆占据(↑)；3）被一辆向东行驶的车辆占据（→）；
- ✓ 在奇数时间步，按184号规则并行更新东向行驶的车的速度和位置；
- ✓ 在偶数时间步时，按184号规则并行更新北向行驶的车的速度和位置；

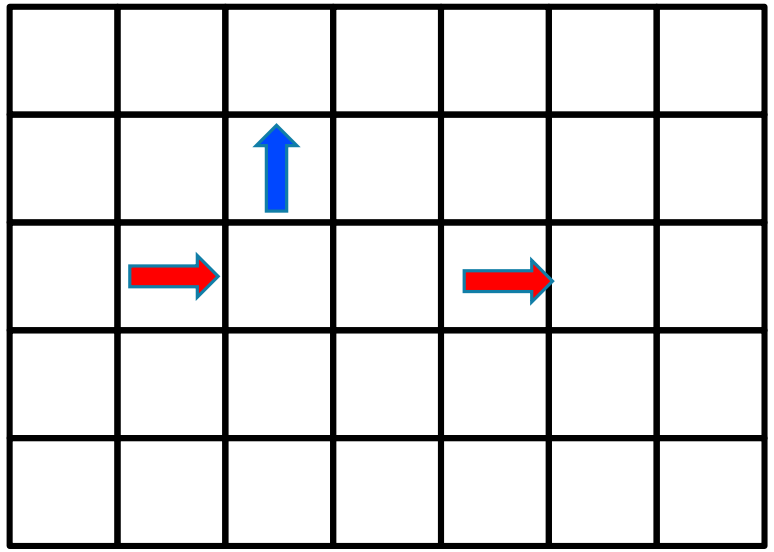
BML模型

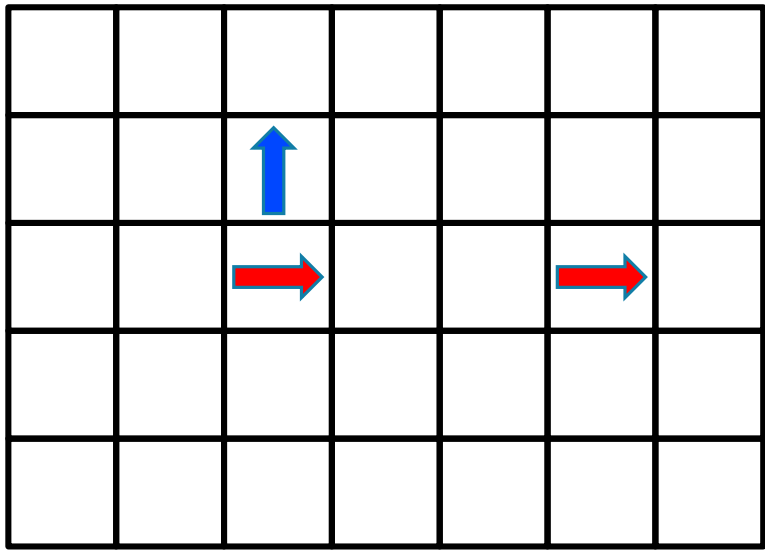




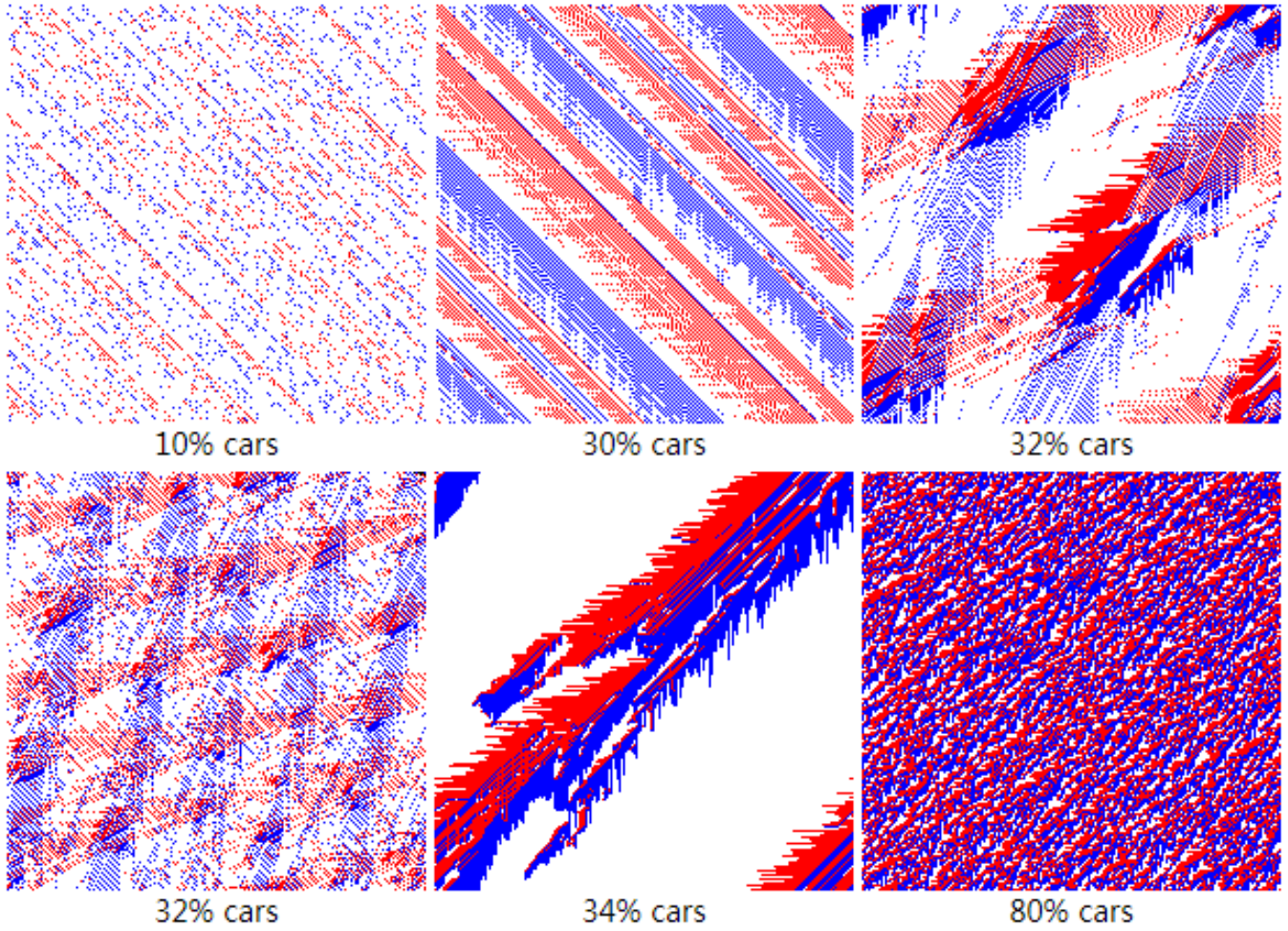








BML模型



- 200×200的网格
- 周期边界
- 演化20000步结果

http://aeholroyd.org/bml/

Thank you for your attention !