arXiv:2304.13575v2 [cs.FL] 4 May 2023

七边形网格上具有六个状态的强通用元胞自动机。

Maurice Margenstern

2023年5月5日

摘要

本文证明，七边形网格上存在一种具有六个状态且旋转不变的强通用元胞自动机。这改进了作者此前发表的一篇采用七个状态的论文。本文中的结构略为简单，规则数量也有所减少。

1 引言

在许多论文中，作者研究了在双曲平面的铺砌中构造通用元胞自动机的可能性，其中少数研究涉及双曲 $3D$ 空间。大多数情况下，所构造的元胞自动机是弱通用的。所谓弱通用，是指自动机能够从某个无限初始配置出发，模拟一个通用装置。不过，初始配置不能是任意的。它通常在一个足够大的圆之外呈周期性；事实上，只要被模拟的装置是双寄存器机，它就在两个不同方向上、一个这样的圆之外呈周期性。在几乎所有论文中，所研究的双曲平面铺砌都是五边形网格或七边形网格，即镶嵌 $\{5,4\}$、 $\{7,3\}$ 。这两种镶嵌都只存在于双曲平面中。在五边形网格中，基本瓦片是一个具有直角的正则凸五边形。在七边形网格中，它是一个正则凸七边形，其相邻边之间的角度为 $\frac{2\pi}{3}$ 。下文的图1展示了双曲平面的庞加莱圆盘模型中的七边形网格。

在左侧的图片中，我们可以看到七个瓦片，按逆时针方向从 1 编号到 7。这些瓦片是一个瓦片的邻居，为方便起见，我们将其称为中心瓦片。实际上，七边形网格中并不存在中心瓦片，因为双曲平面中不存在中心点。我们可以把圆盘模型看作双曲平面上的一扇窗，仿佛我们正乘坐一艘抽象的宇宙飞船在该平面上空飞行。圆的中心是我们注意力所聚焦的点，而圆本身就是我们的地平线。因此，中心瓦片就是相对于我们所观察区域而言处于中心的瓦片

区域。相同的图片还向我们展示了七条圆弧。这些圆弧代表双曲平面中的直线。这些直线具有以下性质：它们穿过铺砌中连续瓦片各边的中点。如果直线上的两个连续中点 ℓ 属于同一个瓦片 T，那么第三个中点 ℓ则不属于该瓦片。后一个点属于一个邻接铺砌块 U的 T：当且仅当铺砌中的两个瓦片具有公共边时，我们称它们互为邻居。在这种铺砌中，如果两个瓦片共享一个顶点，它们也会共享一条边。该性质由角度 $\frac{2\pi}{3}$ 所决定。具有这一性质的直线称为中点线。

在同一图片中，我们还可以看到，两条在瓦片 2 的一条边的中点处相交的中点线，会穿过瓦片 1 各边的中点。由此界定的射线形成一个锐角 S；我们将中心位于 S 内的瓦片集合称为以 1 为首的扇区，而瓦片 1 称为该扇区的头部。

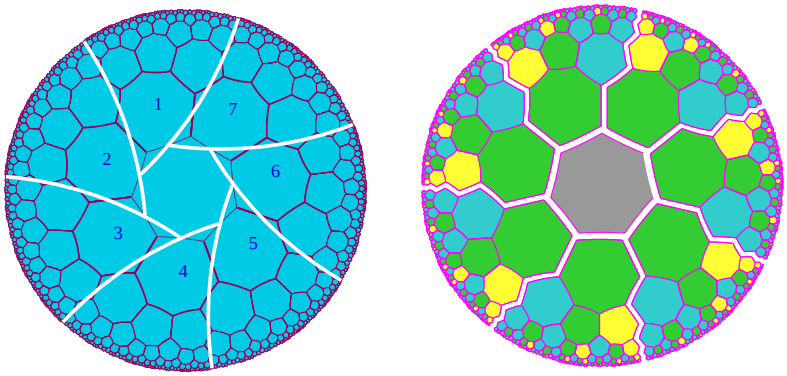


图 1 *To left: the heptagrid; to right: showing the tree structure of the sectors around a central tile indicated by the left-hand side picture.*

在右侧的图片中，我们可以看到中心瓦片以及围绕它的七个扇区：它们由图 1 左侧图片中显示的中点线定义。在每个扇区中，我们有三种瓦片：绿色、黄色和蓝色。若从扇区的顶端向靠近地平线的瓦片方向观察，扇区顶端是绿色的，并且它在扇区中有三个邻居；沿着围绕顶端逆时针转动的方向，它们依次为蓝色、绿色和黄色。蓝色瓦片有两个邻居，依次为蓝色和绿色；绿色和黄色瓦片有三个邻居，依次为蓝色、绿色和黄色。这些说明定义了一些规则，我们可以据此构建一棵覆盖该扇区瓦片的树。关于这一主题以及定位瓦片的方法，详见 [3, 2]。

既已给出整体设定，接下来我们将按如下顺序展开：第2节概述实现的主要思路，并在第2.2小节中对其作精确描述。最后，第3节给出自动机遵循的规则。有关展示规则应用的图，请参阅 [9] 。这些图是根据计算机

程序将自动机的规则应用于第2.2小节所述各配置中的相应窗口后生成的。该计算机程序还检查了规则集是否协调一致，以及各规则在旋转不变性方面是否两两独立。

这使我们得以证明以下性质：

定理1 七边形网格上存在一种强通用元胞自动机，它具有旋转不变性，是真正平面的，并且有六个状态。该自动机使用510条规则。

在论文 [9], 中，该自动机需要1119条规则；而在 [10] 中，即使采用七个状态，也只需要486条规则。

2 计算的主要思路

关于五边形网格（双曲平面的镶嵌）上的通用元胞自动机的第一篇论文是 [1]，其中的镶嵌为 $\{5,4\}$ 。该元胞自动机也具有旋转不变性；在计算的每一步，非静止状态集合都有无限多个环：我们称它为真正平面的元胞自动机。该自动机有22个状态。 [11]中的一种具有9个状态的元胞自动机改进了这一结果。最近，状态数又降至5个，见 [6]。稍后，我证明了在七边形网格（双曲平面的镶嵌）上存在一种具有三个状态的弱通用元胞自动机，它具有旋转不变性且是真正平面的，见 [7]。之后，我将结果进一步改进到两个状态，但其规则不再具有旋转不变性，见 [8]。论文 [4] 构造了三种强通用且具有旋转不变性的元胞自动机：一种位于五边形网格上，一种位于七边形网格上，还有一种位于双曲 $3D$空间的镶嵌 $\{5,3,4\}$ 上。所谓强通用，是指初始配置是有限的，也就是说，它位于一个足够大的圆内。

在本文中，我们借鉴了 [10] 以及前一篇论文[9]中的思想。

对于 [9], ，我们借鉴了这样一种构造思路：实现一个双寄存器结构，使其在计算的每个时刻都有限。它由一段直线构成，并在每次寄存器通过相应指令递增时延长。在这里，每次寄存器通过相应指令递减时，这段直线也会缩短。配置的一部分用于寄存器机的指令，我们将其称为模拟程序。关于程序到寄存器的连接方式以及程序本身，我们借鉴了 [10]中的思想。然而，这种连接并非那么简单；为了将状态数保持在较小范围内而受到的技术限制，促使我们对 [9] 中的思想作出了许多改变，从而大幅减少规则的数量。

该模拟基于 [13] 中提出的铁路模型，并通过作者论文 [10] 中给出的实现进行了重新研究。第2.1小节介绍该模型的主要结构。在第2.2小节中，我们将说明本次模拟所采用的新特性。

2.1 铁路模型

[13] 的铁路模型存在于欧几里得平面中。它由轨道和开关组成，时间 t 时所有开关的配置定义了该时刻的计算配置。如图2所示，开关有三种类型。开关配置的变化由一辆机车执行；该机车沿着由轨道及其连接构成、并由开关组织起来的电路运行。

一个开关在一点汇集三条轨道 a、 b 和 c。在主动交叉中，机车从 a 驶向 b 或 c。在被动交叉中，机车从 b 或 c 驶向 a。

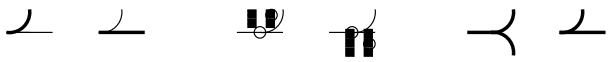


图 **2** *The switches used in the railway circuit of the model. To left、 the fixed switch in the middle 、 、 the flip-flop switch to right the memory switch. In the flip-flop 、 、 switch the bullet indicates which track has to be taken.*

在固定开关中，机车从 a 驶向始终相同的轨道： b 或 c。固定开关可以进行被动交叉。触发器开关始终只能进行主动交叉。如果机车由开关从 a 发送至 b、c ，它将分别被发送至 c、 b 。存储开关既可以进行主动交叉，也可以进行被动交叉。此时，机车在主动通道中驶入的轨道，就是机车在上一次被动交叉中驶入的轨道。

作为示例，我们在此给出用于存储一个单比特信息单元的电路，见图3。机车可以通过门 R 或门 W 进入电路。

如果机车通过安装有存储开关的门 R 进入，它会沿着标记为 1 的轨道或标记为 0 的轨道行驶。当它通过轨道 1 或 0 穿过开关时，分别通过门 E_1、E_0 离开该单元。请注意，在这两种情况下，都有固定开关将机车发送到适当的门 E_i。如果机车通过门 W进入该单元，它会被发送到门 R，从 W沿着轨道 0 或轨道 1 行驶。因此，机车到达 R ，在那里被动地穿过开关，并通过门 E 离开该单元，这是由于一个通向后一个门的固定开关。当机车从 W沿轨道 0 或 1 行驶时，之后的开关分别指示轨道 1 或 0，而机车则通过 R的轨道 1 或 0 到达 R 。轨道编号依据存储在该单元中的值确定。根据定义，当来自 W 和来自 R的两条轨道分别为 0 和 1 时，该单元的值为 0 和 1。因此，正如该研究所示，通过 R进入会读取该单元，而通过 W进入会将该单元从 0 变为 1，或从 1 变为 0：只有在需要改变该单元内容时，才应使用通过 W 进入的方式。该结构的工作方式类似于一种可读取或重写的存储器。这就是为什么我们称其为一位存储器。

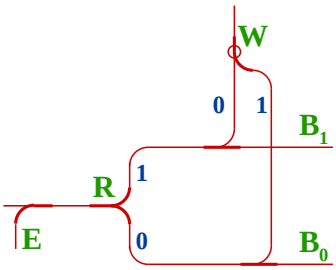


图 **3** *The basic element containing one bit of information.*

在下一小节中，我们将了解如何组合一位存储器，同时会对原始设置进行若干修改，具体原因将在那里说明。

2.2 调整铁路模型

我们首先在第2.2.1小节考察轨道的实现，以及如何定义两条轨道的交叉。在第2.2.3小节中，我们将了解开关的实现。接着，在第2.2.4小节中，我们将了解如何在新的背景下实现一位存储器；然后在第2.2.5小节中，了解如何在不同位置使用它。

2.2.1 轨道

轨道在计算中发挥着关键作用，其重要性不亚于指令和寄存器：事实上，轨道传递信息，而没有这些信息，任何计算都不可能进行。

与 [10]，中一样，轨道是单向的。但与 [1] 中一样，我们将轨道定义为一种特定的颜色。然而，由于机车会从程序驶向寄存器，再从那里返回程序，因此机车应当能够以某种方式朝相反方向行驶。与 [10]，中一样，存储开关被拆分成两种不同的结构：一种用于主动通道，另一种用于被动通道。

考虑四个两两相邻的瓦片，其中第三个瓦片不与第一个瓦片接触。从第一个瓦片到第三个瓦片的运动可表示如下：

```
FWWW    RFWW    WRFW    WWRF    WWWR
```

反向运动如下所示：

```
WWWF    WWFR    WFRW    FRWW    RWWW
```

根据这一观察，我们可以推断，轨道采用统一颜色时可以接受单一方向；不过，只要机车由两个相邻元胞，即前部和后部组成，它也可以接受两个方向。单一

颜色使我们能够相当自由地使用轨道。我们选择让它们沿着线段或圆弧延伸。稍后，我们将展示图来说明这一点。

考虑两个瓦片 U 和 V。从 U 到 V 的路径是瓦片 $\{T_i\}_{i\in[0..n]}$ 组成的有限序列，其中 $T_0 = U$、$T_n = V$，并且对于 i 中的任意一个，$[1..n-1]$、T_i 和 T_{i+1} 共享一条公共边。在这种情况下，$n-1$ 就是该路径的长度。从 U 到 V 的距离，是从 U 到 V 的路径中长度最小者。以 T 为中心、半径为 n 的圆，是与 T 的距离为 n 的瓦片 U 的集合。圆弧是从圆 $\mathcal{C}$ 中的一个瓦片 U 到 $\mathcal{C}$ 中的另一个瓦片 V 的一条路径，且其所有瓦片也都属于 $\mathcal{C}$。轨道的颜色固定为 Y。不过，我们将根据在轨道上运行的机车的颜色来指示轨道。一条 B-轨道或 G-轨道分别由前部为 B-或 G-的机车运行。我们也分别称其为 B-机车和 G型机车。电路某一部分到另一部分的路径由相互连接的轨道片段组成。在电路的大部分区域，机车为 B-。稍后我们将说明，B-和 G-是相反颜色。

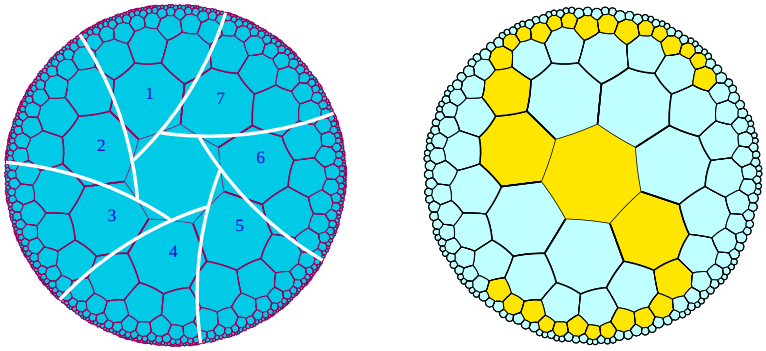


图 **4** *To right in order to locate the tiles of the tracks. the tracks used for testing the implementation. To left, the sectors ,*

图4右侧部分表示一条用于测试轨道实现的轨道。图的左侧复现了图1中所示的扇区，以便定位轨道的瓦片。从上到下，这些瓦片依次为（7,7）至（7,12）、（1,5）至（1,12），然后是（2,2）、（2,1）、（0,0）、（5,1）、（5,3）、（5,7）、（5,6）、（5,5），最后是（4,12）至（4,5）。B型和G型机车都可以沿着这条轨道从（7,7）向下运行至（4,5），也可以反向从（4,5）向上运行至（7,7）：这一点已经通过一个计算机程序进行了测试。

现在转到第2.2.2小节，我们将在其中描述用于实现交叉点的辅助结构和被动固定开关，然后介绍其余的开关，这些开关的实现将在第2.2.3小节中讨论。

2.2.2 辅助结构和交叉点

我们需要的辅助结构包括分叉、转换器和滤波器。转换器和滤波器分别有两个版本：B 型和 G 型。B 转换器和 G转换器分别将 G 型机车和 B-机车转换为 B 型和 G 型机车。B 型滤波器和 G 型滤波器分别让 B-机车和 G-机车继续前进，并分别摧毁 G 型机车和 B-机车。

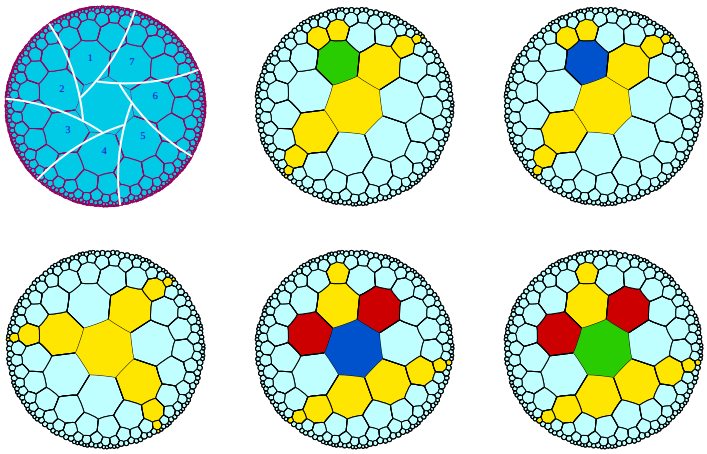


图 **5** *Top from left to right: the sectors the changer from - to - the changer*, B G, G B, , B, G *from - to . Bottom from left to right: the fork the filter for - the filter for -.*

该图展示了这些结构的空闲配置。我们将结构的空闲配置定义为：包含在以中心瓦片为中心、半径为 2 的圆盘内，且该圆盘中没有机车的结构。

在转换器中，轨道通过瓦片 (7,21)、(7,8)、(7,3)、(7,1)、0、(3,1)、(3,3)、(3,8) 和 (3,21) 穿过图形。在 (1,1) 处有一个 B 瓷砖或 G 瓷砖，而在 (1,3) 和 (1,4) 处有一个 Y 瓷砖，用于固定 (1,1) 处的瓦片。在过滤器中，轨道通过瓦片 (6,20)、(6,7)、(6,2)、(5,1)、(4,1)、(4,2)、(4,5) 和 (4,13) 穿过图形。瓦片 0 为 B 瓷砖或 G 瓷砖。在 (2,1) 和 (7,1) 处有 R 瓷砖，在 (1,1) 和 (1,3) 处有两个 Y 瓷砖，其作用将在后文说明。

最后，分叉将三条轨道汇聚在一块中央的 Y瓷砖周围：驶入的轨道经过 (5,21)、(5,8)、(5,3) 和 (5,1)；向左延伸的轨道经过 (2,1)、(2,4)、(2,12) 和 (2,33)；向右延伸的轨道经过 (7,1)、(7,3)、(7,8) 和 (7,21)。

一个计算机程序检查了机车经过这些结构时的运动，并确认这些结构按预期正常工作。

在当前实现中，固定开关仅被动通过，遵循 [10]。

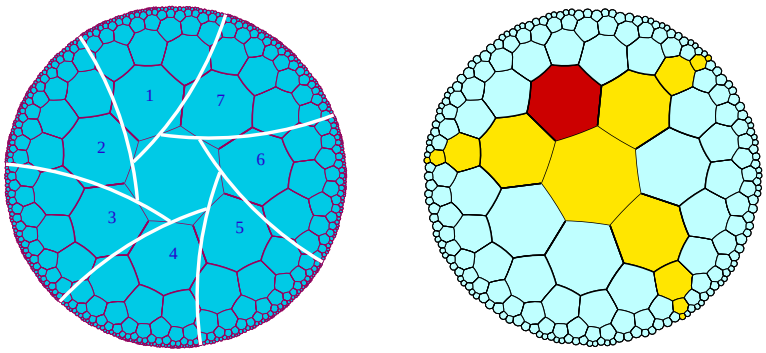


图6

注意 (1,1) 处出现的 R-瓦片。它允许机车沿其一侧通过，同时阻止机车前端进入从该侧延伸出的分支，而开关的另一条分支正是从这一侧接入的。

分叉、转换器、过滤器和固定开关可以组装成一种实现交叉功能的结构。该组织结构如图7所示。

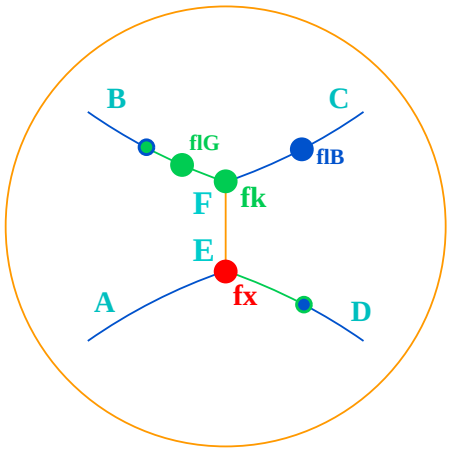


图 **7** *Organisation of a crossing.*

机车到达 C。如果它从 A驶来，就会前往 C；如果从 D 驶来，则会前往 B。如果机车具有指定的颜色，则假定它沿着 C保持颜色；在 E 之后有一个滤波器，只允许这些机车通过。如果机车

从 D驶来，那么在到达 E之前，转换器会将这辆机车变成一辆颜色相反的新机车。假定这辆新机车会前往 D ，那里设有一个合适的滤波器。在 E 处有一个被动固定开关，无论机车是什么颜色，都能让它到达设在 E处的分叉。该分叉会将这辆机车复制一份，颜色保持不变，分别送上通往 B 和 C的每条分支。现在，滤波器只允许颜色与滤波器相同的机车通过。由于从 D驶来的机车在到达 E之前已经改变了颜色，相反变色器会让这辆机车恢复其在 D 轨道上的初始颜色。

2.2.3 开关

本小节讨论触发器开关和存储器开关。

在开关的主动交叉中，若机车经由 a到达，我们称 b, c 为选定轨道，分别对应机车从 a 驶向 b和 c的情形。触发器在经过主动交叉后会改变选择，并且触发器开关从不发生被动交叉。在存储开关中，其被动部分决定其主动部分的选择。因此，我们需要两种结构：一种让机车通过，另一种将其消灭。在触发器中，开关交叉会引起改变；在存储开关中，被动开关的交叉会触发改变。本质上，这是同一项约束。因此，触发器和存储开关中的控制结构都应当是可编程的，这一点我们将在下文进一步解释。

图8展示了触发器的实现（图左侧），以及主动存储开关的实现（图右侧）。图9再次展示主动存储开关，并在右侧展示被动存储开关。

在触发器中，简单机车到达 C。在那里，一个分叉发送出两辆简单机车：一辆驶向 L，另一辆驶向 R。驶向 R 的机车会到达一个滤波器所在的位置。如果滤波器的颜色与机车相同，后者就会继续前进。否则，机车会被消灭。现在，如果位于 R 的滤波器分别为 B-、G-，则位于 L 的滤波器分别为 G-、B-。因此，机车只会沿开关的一侧继续前进，这就定义了选定轨道：它是离开轨道，在该轨道上，滤波器的颜色与到达 C的机车颜色相同。当机车沿选定轨道继续前进时，在距离滤波器一定距离处，它会遇到一个分叉：分叉的一个分支让机车沿选定轨道继续前进，另一个分支将机车发送至 S。在那里设有一个固定开关，任何到达此处的机车都会被动地穿过该开关。从 S出发，机车到达一个分叉，该分叉将它的一个副本发送至 L 处的滤波器，将另一个副本发送至 R处的滤波器。这些副本会将滤波器改为相反颜色，因此开关的选定轨道也随之改变，这符合触发器开关的定义。这种变化将在第3节中解释，该节专门介绍元胞自动机所遵循的规则。

请注意，从 L 到 R 的轨道是一条线段，而从

滤波器后方通往 S 的分叉是一段圆弧。必须在机车通过选定轨道的滤波器后更改过滤器。只需定义支撑该圆弧的圆，其半径等于从 C 到 L 的距离，并使圆弧的角度至少为$\frac{\pi}{2}$：圆的周长是其半径的指数函数。

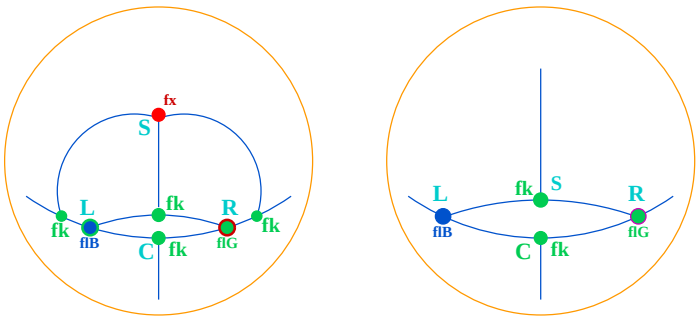


图 **8** *To left: the flip-flop. To right: the active memory switch.*

就运行方式而言，主动存储开关更简单：当机车到达 C 时，只涉及从 C 到 L 以及从 C 到 R 的弧。我们可以假设，当机车未参与主动开关时，就会出现被动交叉。

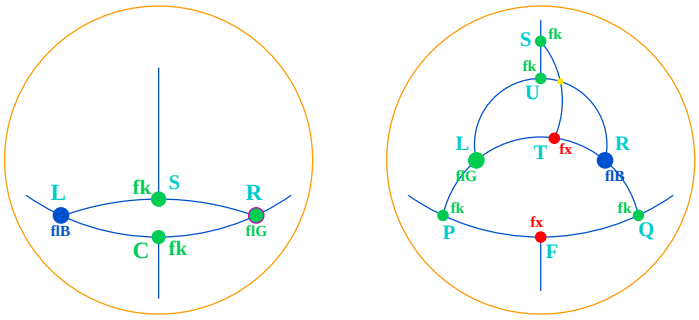


图 **9** *Memory switch: to left, the active switch; to right the passive one.*

被动开关的结构截然不同。简单机车从 P 或从 Q到达。假设它从 P到达。从 Q 到达的情况可通过对称方式处理。 P 处的分叉将一辆机车发送至 F，即一个使机车离开开关的固定开关。另一辆机车则被发送

至 L。在那里，如果主动部分中的滤波器允许机车通过， L处的滤波器就会拦截机车，因此存储开关不会改变选择。如果主动部分中的滤波器拦截机车，被动开关中的滤波器就会允许机车通过。机车经由 T继续前进至 S 。在那里，在 S处，一个固定开关将其发送至 U ，分叉在此处将机车复制一份，发送一份至主动开关的分叉 S ；根据主动开关的方案，该操作将交换其滤波器的作用。 S 处的分叉发送出的第二辆机车前往 U ，在那里，另一个分叉将该机车复制成两份，一份发送至 L ，另一份发送至 R，两份机车分别交换相应滤波器的作用。因此，存储开关的两个部分都能按所需方式运行。在这里，我们同样必须确保从 T 发送的机车在 L 和 R处的到达时间晚于从 P 或从 Q到达许可控制器的机车。图9向我们保证，使用半径足够大的圆弧即可轻松满足这一条件。

请注意，存储开关的两个部分彼此分离，因此可以用一个被动部分来改变多个主动部分；我们很快就会遇到这种情况。

2.2.4 一位存储器

现在，我们已准备好在我们的设定中实现一位存储器。图10展示了这一实现，其左侧部分让我们想起欧几里得空间中的实现。为了便于读者比较，我们采用了与欧几里得示意图相同的记号。

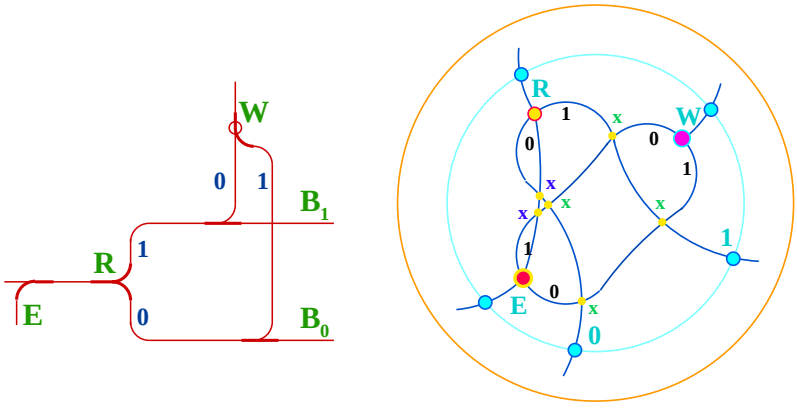


图 **10** *To left, the scheme of Figure* 3*, to right, its implementation in our setting. In the hyperbolic picture, the colours at W, R and E, namely yellow, light green and light purple respectively represent a flip-flop switch an active memory switch and a* , x *passive memory switch. The yellow points with an letter indicate crossings.*

这里同样涉及一辆简单机车。如果它到达标有 R 的蓝色圆盘，它会前往标有相同字母的红色圆盘，然后

根据主动存储开关所选择的轨道，它会通过门 $B0$ 或门 $B1$ 驶出。

如果机车到达蓝色 W，它会被送往红色 W，在那里，翻转开关会根据红色 R处开关所选择的状态，通过轨道0或轨道1将机车送往红色 E。如果轨道为 i，那么机车会通过其轨道 $1-i$到达红色 E。现在，根据我们在图9中看到的情况，红色 E 会向红色 R 发送一辆机车，使开关选择与红色 E处状态兼容的值。但从红色 E处，另一辆简单机车会经由蓝色门 E从开关中驶出。请注意，左图中的存储开关在右图中被分成了两部分。其中，主动部分位于 R，而被动部分位于 E，这是一条连接 E 和 R的路径。

由于我们稍后会多次使用一位存储器，因此我们用一个带有五个门的蓝色圆盘来表示它，这些门分别标记为R、E、W、0和1。

2.2.5 一个寄存器及其判别结构

图 11 展示了在我们的设置中寄存器的实现。如图所示，我们有一段由环绕其周围的路径围住的线段。

寄存器的值是从图 11 左侧的瓦片 0 到右侧部分的瓦片 0 之间 W 型瓷砖的数量，包括这两个瓦片。空寄存器的值按定义为 0，其特征是起始处的瓦片 0 为 M-。

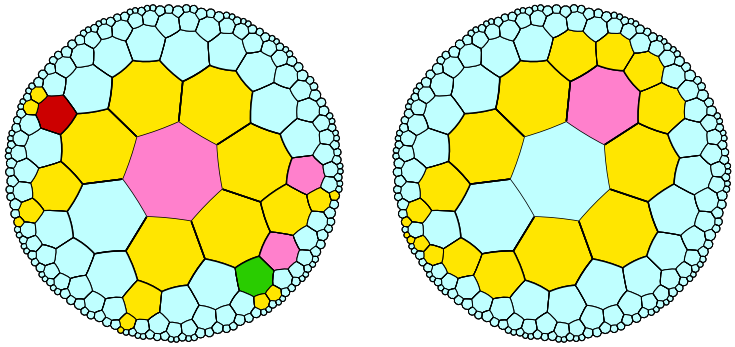


图 **11** *To left the beginning of a register when $n = 0$; to right its end when , , $n > 0$. Both ends are marked as far as they play an important role.*

寄存器是一个契机，让我们谈谈两种机车这一事实。事实上，B-机车用于实现寄存器，而 G 型机车用于递减寄存器。不过，机车的颜色由一个适当的结构在进入寄存器之前确定。在到达该位置之前，无论机车承担什么角色——无论是传送指令，还是作为信号来改变滤波器的工作方式，以便在某些时刻有三辆机车在电路中运行——它都是 B 型。此外，寄存器外还需要三个新的结构：一种结构用于记住需要递

增寄存器的指令，即 $\mathbb{DD}_{\mathbb{I}}$-结构，见 2.2.5.a；一种结构用于记住需要递减寄存器的指令，即 $\mathbb{DD}_{\mathbb{D}}$-结构，见 2.2.5.b；一种结构用于记住指令的类型，即 $\mathbb{DD}_{\mathbb{T}}$-结构，见 2.2.5.c，也就是递增或递减寄存器。这些结构是每个寄存器都需要的。所有这些结构都使用一位存储器。

2.2.5.a 记忆递增指令

如图 11 所示，当机车从寄存器返回时，由于我们只有两种机车，机车并不会记得它是从程序中的哪个位置发出的。为了让它返回该位置以确定下一条指令，必须在寄存器附近以某种方式标记该指令。 $\mathbb{DD}_{\mathbb{I}}$-结构就是用来实现这一功能的。每个寄存器 $\mathcal{R}$都有一个 $\mathbb{DD}_{\mathbb{I}}$-结构。该结构包含的单元数量，等于程序中专门用于递减 $\mathcal{R}$的指令数量。对于每条递增 $\mathcal{R}$指令 I，都有一条路径从程序中的 I 位置延伸到与该路径关联的 $\mathcal{R}$ $\mathbb{DD}_{\mathbb{I}}$ 的单元。该单元本身包含一个一位存储器，见图 12。

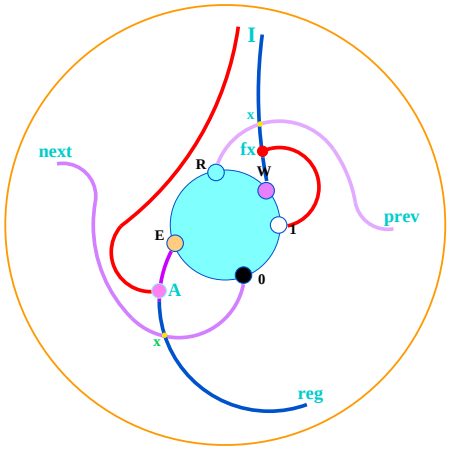


图 **12** *The unit of* $\mathbb{DD}_{\mathbb{I}}$ *devoted to an incrementing instruction.*

在初始配置中，每个 $\mathbb{DD}_{\mathbb{I}}$结构的所有单元里，一位存储器都包含 0。当机车开始行进以执行针对寄存器 $\mathcal{R}$的递增指令 I 时，机车所沿的轨道抵达与 I 相关联、连接至 $\mathcal{R}$ 的 $\mathbb{DD}_{\mathbb{I}}$ 的单元的 W门。当机车抵达 W门时，它会将一位存储器中的 0 位重写为 1。随后，机车通过其 E门离开存储器。它在 A 处遇到一个触发器，该触发器从初始配置开始便将机车发送向 $\mathcal{R}$的 $\mathbb{DD}_{\mathbb{T}}$结构。通过该触发器开关后，选定轨道现在变成一条返回程序的轨道，通向在 I 完成其对 $\mathcal{R}$的操作后必须执行的指令。图 12中以橙色标出了这条轨道。此时，该 $\mathbb{DD}_{\mathbb{I}}$ 中只有一个单元，其一位

存储器包含 1。

当机车从 $\mathcal{R}$返回时，它会经过 $\mathbb{DD}_{\mathbb{T}}$结构。该结构记住它执行过一条递增指令，因此会将机车发送到 $\mathcal{R}$的 $\mathbb{DD}_{\mathbb{I}}$ 。

机车依次经过 $\mathbb{DD}_{\mathbb{I}}$结构的每个单元，直到遇到唯一一个其一位存储器包含1的单元。为此，机车通过该单元的R门进入存储器。如果读到0，就会被发送到下一个单元。当读到1时，它便知道已经到达正确的单元。机车通过1门离开一位存储器后，被发送到W门，以便将一位存储器中的内容从1改写为0。通过E门离开存储器后，机车再次遇到 A ，触发器的选定轨道会将它发送到通往程序正确位置的轨道。当机车穿过 A 处的触发器后，选定轨道再次变为通往 $\mathcal{R}$ 的轨道。因此，当机车离开 $\mathbb{DD}_{\mathbb{I}}$结构时，该结构已经恢复了初始配置。

2.2.5.b 记住递减指令

如果机车必须递减寄存器 $\mathcal{R}$，它会访问一个与 $\mathbb{DD}_{\mathbb{I}}$ 类似的结构，即附加于 $\mathcal{R}$ 的 $\mathbb{DD}_{\mathbb{D}}$-结构。该结构的原理与 $\mathbb{DD}_{\mathbb{I}}$ 的原理类似。不过， $\mathbb{DD}_{\mathbb{D}}$ 的一个单元比 $\mathbb{DD}_{\mathbb{I}}$ 的一个单元更复杂。其原因在于递减指令与递增指令之间的差异。递增寄存器总是可行的。由于寄存器机只处理自然数，因此无法递减空寄存器。当递减机车抵达空寄存器 $\mathcal{R}$ 的起始处时，它会通过另一条轨道离开寄存器，而不是像成功递减寄存器时那样离开。

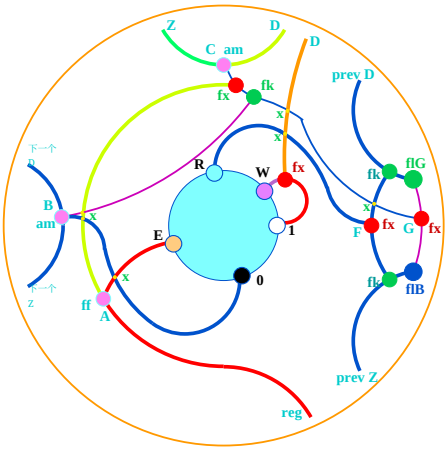


图 **13** *The unit of* $\mathbb{DD}_{\mathbb{D}}$ *devoted to a decrementing instruction.*

因此，如果对于从程序到达的机车而言， $\mathbb{DD}_{\mathbb{D}}$ 的工作方式与 $\mathbb{DD}_{\mathbb{I}}$ 相似，那么对于从

寄存器返回的机车而言，情况并非如此。我们将从程序到达的机车这一情况留给读者。

考虑机车从 $\mathcal{R}$ 返回的情况。一次成功操作后，它从 D轨道 返回；或者由于无法对空的 $\mathcal{R}$ 执行递减，它经由 Z轨道 返回。这两条轨道的作用相同：都通向 R门。如果机车读取到 0，它就会通过与进入当前单元时所使用轨道类型相同的轨道，前往下一个单元。可以看出，当机车从寄存器而来并经由 R门进入单比特时，它有两条可能的出口。但在这两种情况下，出口都会再次分成两种可能，具体取决于机车沿哪条轨道进入该单元。如果出口通往下一个单元，那么当进入轨道分别为 D轨道或 Z轨道时，出口轨道就是 D轨道或 Z轨道。如果出口通往程序，那么当进入轨道分别为 D轨道或 Z轨道时，出口轨道会通往下一条指令，或跳转条件所指定的另一条指令。这一区别在单元入口处即可确定，见图13中的 G 。这就是在 G 处设置被动存储开关的原因。由于在 B 和 C处都必须了解这一差异，因此在这两个位置各有一个主动存储开关，并通过 G与该被动存储开关相连。与这些主动存储开关的连接由一条从 G 出发的轨道建立；该轨道在到达 C 之前经过一个分叉，将一个副本发送至 C ，另一个副本发送至 B。因此，整个结构在所有情况下都实现了预期的功能。

2.2.5.c 记住指令的类型

该结构的工作方式让我们想起了 $\mathbb{DD}_{\mathbb{D}}$。在这里，有两条可能到达该结构的轨道：要么来自 $\mathbb{DD}_{\mathbb{I}}$-结构，要么来自 $\mathbb{DD}_{\mathbb{D}}$-结构。

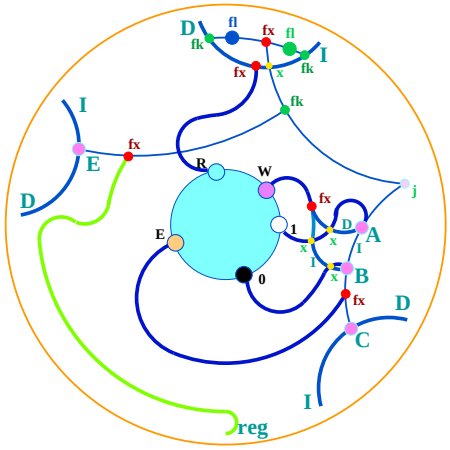


图 **14** *The unit of* $\mathbb{DD}_{\mathbb{T}}$ *devoted to recover the type of the instruction.*

还有两个可能的出口：一个返回 $\mathbb{DD}_{\mathbb{I}}$-或 $\mathbb{DD}_{\mathbb{D}}$-结构，另一个通往寄存器 $\mathcal{R}$。不过，在后一种情况下，根据指令不同，会有两条不同的轨道：指令是要递增 $\mathcal{R}$ 还是要递减

它。通过结构到达点上的一个被动存储开关来确定采用哪条轨道：图14上部对此作了示意。这条信息会传递到 $\mathbb{DD}_{\mathbb{T}}$的出口部分：通往寄存器以及 $\mathbb{DD}_{\mathbb{I}}$或 $\mathbb{DD}_{\mathbb{D}}$。在图14中，我们可以看到一条轨道从到达点出发，经由 j 通向 C，再通向 E；这是借助一个分叉实现的，机车在到达 j之前会经过该分叉。在 C 和 E 处各有一个主动存储开关，与信号相连；该信号可能由靠近通往 $\mathbb{DD}_{\mathbb{T}}$ 的到达轨道的某个滤波器发出。有关指令类型的信息也会分发给位于 A和 B处的其他主动存储开关。之所以需要这些额外的存储开关，是因为读取到 0 或 1 的含义会因所涉及的指令不同而不同。我们规定，1 和 0 分别对应于需要递增和递减寄存器的指令。

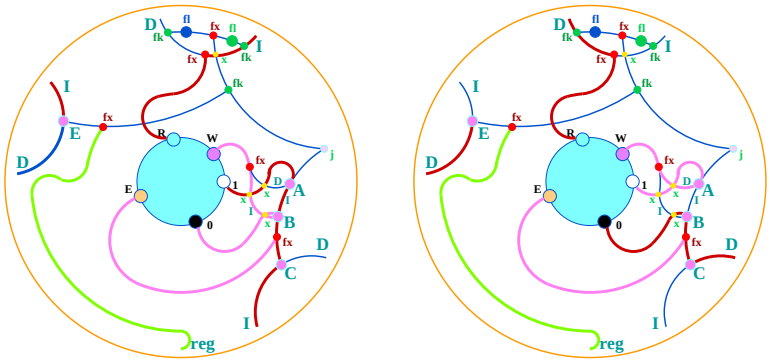


图 **15** *Inside* $\mathbb{DD}_{\mathbb{T}}$ *to left to right the tracks followed by an incrementing decre-* „ , , *menting respectively locomotive.*

图 15 展示了机车在需要递增或递减寄存器时所经过的轨道。在两幅图中，当存储器中的位与指令类型所对应的位相同时，机车会沿红色轨道行进；否则，轨道显示为浅紫红色。在这两种情况下，紫红色轨道都会通向 W门，使位 b 变为 $1-b$。因此，在这种情况下，机车会经由其 E门离开存储器，经过一个被动固定开关前往 C，再从 C 前往寄存器以执行相应操作。

2.2.5.d 递减和递增寄存器

图16展示了机车在寄存器内部执行的操作。该图说明了内容取非平凡值时的标准情况：M瓦片与结构起点之间相距一定距离，具体取决于寄存器的初始内容，而寄存器的初始内容可能为空。在图中最左侧的画面里，我们可以看到围绕 0 处 M-元胞的 Y单元 1(6)、1(7) 和 1(1)，我们将其称为寄存器帽。在图中最右侧的画面里，我们还可以看到围绕 1(3) 处 M-元胞的帽，它由 Y单元 1(4)、0 和 1(2) 组成。在展示递增操作的第三个画面中，帽由围绕 1(7) 处 M-元胞的四个 Y单元组成：2(7)、3(7)、4(7) 和

2(1)。

图16展示了该操作的执行方式。图分为两部分。左侧是操作前的寄存器，右侧是操作执行后的寄存器。在右侧部分中有两种可能：左边是递增，右边是前面提到的递减。

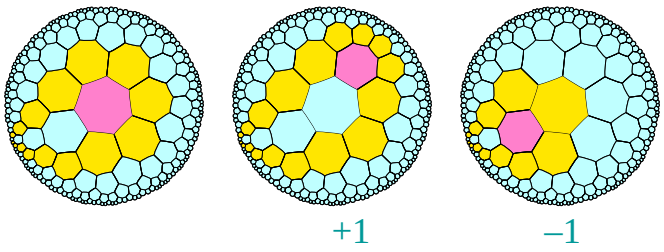


图 **16** *To left the register before the operation. To right, the configuration after the operation.*

图16所示的操作已通过计算机程序进行测试。

3 条规则

规则的定义将使我们有机会更明确地研究自动机的行为。

本节首先介绍一种用于编写规则方案的形式体系，以便减少规则的数量。随后，在之后的每个小节中，我们将依次考察第2.2小节各子子小节中研究的情况。

3.1 统一记号

由于我们假设规则具有旋转不变性，因此，对于旋转意义下具有相同邻域的规则，我们按照对状态设定的字典序 W、Y、B、G、R、M，取其中最小者加以汇总。从现在起，当我们谈到元胞自动机时，将用“元胞”代替“瓦片”。例如，窗口的中心瓦片将称为窗口的中心元胞。原因在于，元胞由一个瓦片支撑，但它还包含有限自动机；该有限自动机对所有元胞都相同，并在离散时钟的每个时刻控制元胞的变化。所有这些条件都只是元胞自动机定义的应用。第一个字母表示元胞的当前状态，最后一个字母表示根据元胞的当前状态及其邻居的当前状态得到的新元胞：如果我们按照从左到右的顺序，将两个逗号之间的单词中的邻居从 1 排列到 7，则 i^{th} 字母给出元胞对应 i^{th} 邻居的状态。在这种情况下，第一个邻居取决于邻域。现在，根据旋转不变性，我们考虑

由邻居状态得到的最小单词；因此，如果至少有两个状态不同，就可以通过这种方式唯一确定第一个邻居。

以蓝色机车在线段上的运动规则为例，即 `Y-WWWBWWY:B`. 这是表10中规则150的最简形式。在表中，规则150显示为 `Y-BWWYWWW:B`. 该规则替代了以下规则：

a `Y-WWWYWWY:Y`	e `Y-BWWYWWW:B`
b `Y-WWWBWWY:B`	f `Y-WWYWWWB:B`
c `Y-WWBWWYW:B`	g `Y-WYWWWBW:B`
d `Y-WBWWYWW:B`	h `Y-YWWWBWW:B`

当且仅当新状态与当前状态相同，并且应用规则时邻居的状态也未发生变化时，称该规则为守恒的。如果仅元胞的状态未发生变化，则称其为见证规则。例如，规则7，即 `W-WWWWWBR:W` 就当前元胞的邻域中的 B-元胞和 R-元胞属于蓝色机车而言，是一个见证规则。

3.2 轨道规则

适用于轨道元胞的机车运动规则见表1。

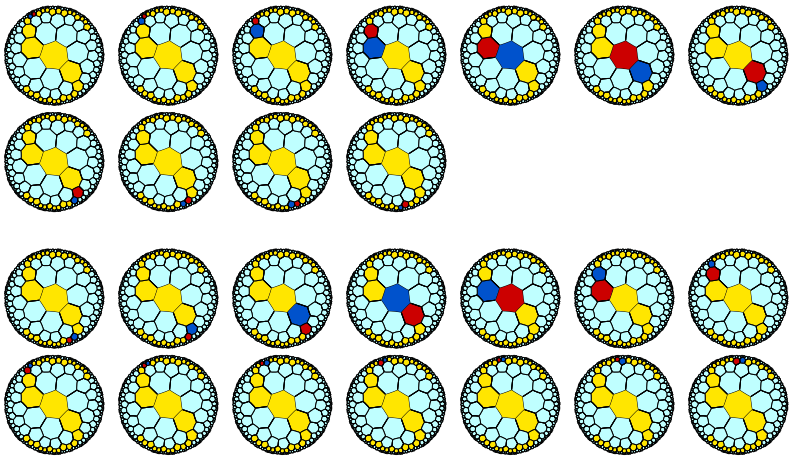
表 **1** *Motion rules of a locomotive on the tracks.*

`Y-WWWBWWY:B` 152	`Y-WWWGWWY:G` 161	`Y-WWWYWWB:B` 64	`Y-WWWYWWG:G` 94
`B-WWWRWWY:R` 17	`G-WWWRWWY:R` 24	`B-WWWYWWR:R` 53	`G-WWWYWWR:R` 88
`R-WWWYWWB:Y` 18	`R-WWWYWWG:Y` 25	`R-WWWBWWY:Y` 60	`R-WWWGWWY:Y` 91
`Y-WWWYWWR:Y` 21	`Y-WWWYWWR:Y` 21	`Y-WWWRWWY:Y` 62	`Y-WWWRWWY:Y` 62

在上述规则中，左侧两列适用于在线段或圆弧上逆时针运动的机车，右侧两列适用于在圆弧上顺时针运动的机车。请注意，规则 21 和 62 各出现两次，因为它们涉及轨道上见证离开机车后部的元胞，因此此类规则适用于任何机车，不论其颜色如何。此外，规则 152 和 161 的邻域相同，但当前状态不同；规则 18 和 64 、规则 161 和 91，以及规则 25 和 94 也有类似情况。G- 机车的规则可由 B-单位的规则得到：将 B-机车规则中的状态 B- 替换为 G- 即可。

沿圆弧铺设的轨道需要额外的规则。这些规则的形式为 `Y-WWWWFWY:F` ，其中 F 表示 B 或 G，即机车的前部。我们有规则 `Y-WWWWFWY:F`, 规则 `F-WWWWRWY:R`, 规则 `R-WWWWYWF:Y` 以及规则`Y-WWWWYWR:Y`. 。对于 B-机车，它们分别是规则 36、38、54 和 42；对于 G-机车，它们分别是规则 75、77、79 和 42。所有这些规则都用于顺时针运动。逆时针运动的规则可通过交换第五个和第七个邻接铺砌块之间的状态，由顺时针规则得到。例如，根据规则 `Y-WWWWFWY:F`, 对于 B-机车和 G-机车，我们分别得到规则 `Y-WWWWYWF:F` ，即规则 54 和 89。

图4展示了轨道元胞的空闲配置，在这些轨道上，顺时针运动会反转为逆时针运动，反之亦然。图17展示了 B-机车沿图4所示轨道向两个方向运动的情况。读者可以在图17中核对表1中的规则；对于 G-机车同样适用，只需将 B瓷砖替换为 G 型机车。



图

17 *Top rows: motion of a blue locomotive from top to bottom. Bottom rows: motion of a blue locomotive from bottom to top. Note that on the arcs of a circle we have both a clockwise and a counter-clockwise motion.*

3.3 固定开关和分叉

我们从固定开关开始，仅考虑其被动版本。

我们分为四种情况：蓝色机车从左侧或右侧分支驶来，以及相应的两种绿色机车情况。读者可参照图6和图5，了解表2所列规则的应用；其中的记号沿用第3.1小节中引入的记号。我们还使用相同的元符号F替代B和G。图18展示了机车被动驶向开关的运动过程。

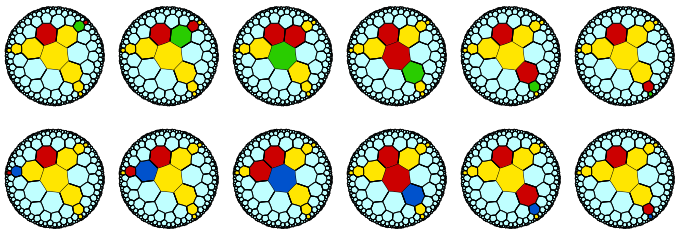


图 **18** *The motion of a locomotive passively crossing a fixed switch. Top, a* G *-locomotive arriving from the right. Bottom a* B *-locomotive arriving from the left. The opposite motions for a* B *- or a* G *-locomotive are easily derived from the pictures.*

表 **2** *Rules for the fixed switch.*

位于0、然后1(2)、再然后1(7)的固定开关的守恒规则

Y-WWYWYRY:Y 119Y-WWYWWYR:Y 135Y-WWYWWRY:Y 122固定开关的运动规则1^{st}、2^{nd} 编号: F= B、 G 分别元胞0左侧的元胞0右侧Y-WWYWYRY:Y 119 Y-WWYWYRY:Y 117

Y-WWYWYRF:F 123、137 Y-WWYWFRY:F 146、155

F-WWYWYRR:R 126、140 F-WWYWRRY:R 149、158

R-WWFWYRY:Y 131、144 R-WWFWYRY:Y 129、144

Y-WWRWYRY:Y 134 Y-WWRWYRY:Y 134元胞1(2) 元胞1(7)

Y-WWYWWYR:Y 135 Y-WWYWWRY:Y 122

Y-WWFWWYR:F 121、136 Y-WWFWWRY:F 145、154

F-WWRWWYR:R 125、139 F-WWRWWRY:R 148、157

R-WWYWWFR:Y 128、142 R-WWYWWRF:Y 153、162

Y-WWYWWRR:Y 133 Y-WWYWWRR:Y 133

表2给出了用于元胞0、1(1)和1(7)的规则。元胞0的规则是见证规则，但规则119除外，因为它是守恒的。对于元胞1(2)和1(7)，规则135和122是守恒的，其余规则是运动规则。注意，当元胞见证机车后部离开时，两个分支使用相同的规则133。还要注意，当机车前端位于1(5)时，1(2)和1(7)所使用的规则不同：左侧分支为128、142，右侧分支为153、162。在前述规则的数字对中，第一个和第二个数字分别处理B型机车和G型机车。这些规则的应用可以在图18中得到验证。

表 **3** *Rules for the fork.*

元胞0处分叉及随后元胞1(1)处分叉的守恒的规则

Y-WWYWYWY:Y 98W-WWWWYYY:W 56元胞0处分叉及元胞1(1)处分叉的运动规则和见证规则

Y-WWYWYWY:Y 98 W-WWWWYYY:W 56

Y-WWFWYWY:F 99, 111 W-WWWWYFY:W 102, 114F-WWRWYWY:R 101, 113 W-WWWWFRF:W 105, 117R-WWYWFWF:Y 104,

116 W-WWWWRYR:W 108Y-WWYWRWR:Y 107

请注意规则104、116和107，以及规则105、117，在这些规则中，我们可以看到复制的机车的两个前部。还请注意，元胞见证机车离开时所适用的规则，对于两种类型的机车都是相同的。表中的规则可通过图19进行核对，图19展示了机车通过分叉的运动过程。

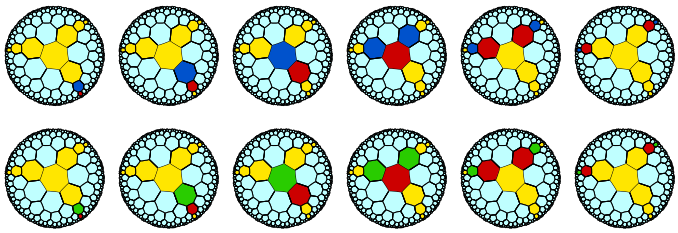


图 **19** *Top: duplication of a* B*-locomotive. Bottom: duplication of a*G*-one.*

3.4 转换器和过滤器

下面两张表给出了管理转换器和过滤器的规则；对于后者这一结构，我们还列出了改变过滤器颜色的规则。表4给出了有关转换器的规则，表5给出了规定过滤器行为的规则。

在下表中，F的出现始终表示B，或始终表示G；相应地，H始终表示G或B。对于规则中的数字对，同样适用表2和表3中的约定。

表 **4** *Rules for the changers. First the blue one and then the green one. In both , , of them cell 0 and the cells 1(1) and 1(7) are considered.*

蓝色转换器

单元格0	元胞1(1)	元胞1(7)
Y-WWWYFWY:Y 180,163	F-WWYYWYY:F 181,164	Y-WWYWWFY:Y 196,179
Y-WWWFFWY:F 185,168	F-WWYFWYY:F 186,169	Y-WWBYWWG:B 184
F-WWWRFWY:R 189,172	F-WWFRWYY:F 190,173	Y-WWBWWGY:G 167
R-WWWYFWF:Y 192,175	F-WWRYWYY:F 193,176	F-WWFYWWR:R 188,171
Y-WWWYFWR:Y 194,177		R-WWYWWFF:Y 191,174
		Y-WWYWWFR:Y 151,160

请注意，规则167是通过交换B-和G-状态从规则184得到的，但这一变化会改变相应最小规则的字母序。可以在图20中检查这些规则。

表 **5** *Rules for the filters. Here too, we use the formalism allowing us to present both filters with the same meta-rules:* F *and* H *stand both for* B- *or* G- *with the convention that and are always different when they occur in the same rule and in the same* F H *group of rules. The same convention as in Tables 2 and 3 holds for the couples of numbers for the rules.*

过滤器的工作原理

单元格0

```
F-WYYWRYR:F 197,220
F-WYFWRYR:F 204,227
F-WFRWRYR:F 207,230
F-WRYWRYR:F 211,234
```

单元格1(5)

```
Y-WWWYWFY:Y 215,238
Y-WWWFWFY:F 201,224
Y-WWWHWFY:Y 216,237
F-WWWRWFY:R 205,228
R-WWWYWFF:Y 209,230
Y-WWWYWFR:Y 213,234
```

更改滤波器

单元格1(1)

```
Y-WWRFRWY:Y 195,218
Y-WWWFWFY:F 199,222
Y-WWWHWFY:Y 214,239
F-WWWRWFY:R 203,228
R-WWWYWFF:Y 207,232
Y-WWWYWFR:Y 211,236
```

单元格0

```
F-WYYWRYR:F 197,220
F-WYYWRHR:H 245,251
H-WYYWRYR:H 220,197
```

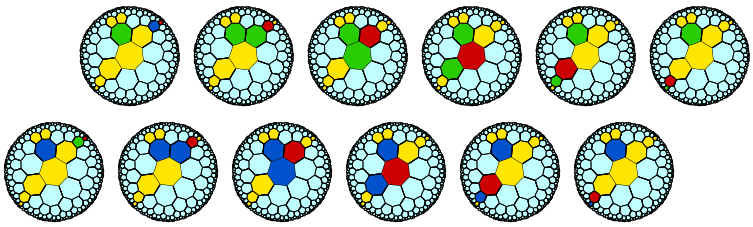


图 **20** *Top: changing a -locomotive to a -one. Bottom: changing a* G B *-locomotive to a -one.*

图20展示了转换器对相反颜色的机车所起的作用：后者会变成转换器的颜色。

图21同样展示了滤波器的作用。我们可以看到，滤波器会让颜色相同的机车通过，同时摧毁颜色相反的机车。

鉴于我们在开关的实现中经常使用过滤器，我们知道，需要通过赋予过滤器改变自身颜色的能力来对其进行编程。表5中的相应规则允许合适的机车完成这一操作。图22可供读者核对这些规则。后者还解释了为什么图5中有两个Y单元紧邻编号为0的元胞，其颜色

决定了过滤器的颜色；它们位于机车所经过、且需要加以控制的轨道上的Y单元的对面。这些Y单元是通向过滤器的一条轨道的末端，机车可沿此轨道改变过滤器的颜色，具体遵循规则245和251，即分别为 `B-WYYWRGR:G` 和 `G-WYYWRBR:B` 。

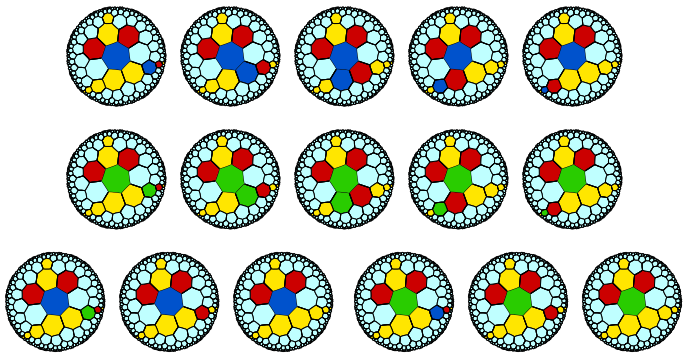


图 **21** *Top: when the filter let the locomotive of its colour go. Bottom, the filter stops the locomotive of the opposite colour.*

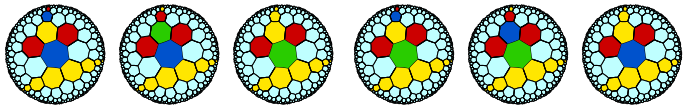


图 **22** *Changing the colour of the filter.*

3.5 寄存器

令 $\mathcal{R}$ 表示一个寄存器。属于 $\mathcal{R}$ 的元胞记作 $\mathcal{R}$(n)，其中$n \in \mathbb{N}$。当 $n = 0$时，寄存器为空。如果 n 是寄存器内容的值，则元胞 $\mathcal{R}$(n) 为 M。因此，当寄存器为空时， $\mathcal{R}$(0) 为 M。如图 11 所示，从 $\mathcal{R}$(n) 到 $\mathcal{R}$(n+1) 或$\mathcal{R}$(n-1)的变换需要进行大量工作。对于这种变换，我们区分 $n = 0$ 的情形与 $n > 0$ 的情形；在后一种情形中，对于递减指令，我们再区分 $n = 1$ 和 $n > 1$两种情形。

当 $n = 0$ 需要为 $\mathcal{R}$(0)采用特殊配置时，不应执行递减操作。相反，G-机车应沿着第2.2.5小节中提到的特定轨道行驶。现在，当机车前往 $\mathcal{R}$(n,)并且 $n > 1$ 以执行某项操作时，它必须穿过 $\mathcal{R}$(0.)因此，离开 $\mathcal{R}$(0,)的机车有两条可能的轨道，具体取决于它是否无法执行该操作，或者是否能够执行该操作；后一种情况下，它必须递增 $\mathcal{R}$ ，或者必须对 $\mathcal{R}$(n)执行操作，以便处理 $n > 0$。图23展示了 $\mathcal{R}$(0.)周围的情况。离开 $\mathcal{R}$(0:)邻域的机车有两条轨道：一条依次经过元胞 1(4)、1(5)、

1(6)、2(7)、7(7)、20(7)，然后继续前往 $\mathcal{R}$(n;)；另一条轨道依次经过元胞1(4)、1(5)、2(6)、7(6)、20(6)，然后继续前往连接到 $\mathcal{R}$的 $\mathbb{DD}_{\mathbb{T}}$ 结构。

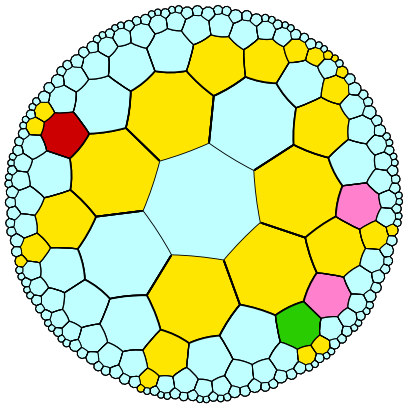


图 **23** *The idle configuration of* $\mathcal{R}$*(0.)Note that cell 0 is* M*. Note the other* M*-cells.*

分岔发生在1(5)：它涉及另外两个元胞2(6)和1(6)。这正是出现两个M-元胞的原因：一个位于4(5)，从1(5)和2(6)都可以看到；另一个位于3(6)，从1(6)和2(6)都可以看到。对于必须递增寄存器的B-机车而言，控制很简单。对于G-机车，情况则不那么简单：如果递增失败，机车必须前往2(6)，否则必须前往1(6)。在元胞0处也可以看出这一差别：当且仅当寄存器的内容为0时，它才是M。图24和图27展示了这种差异。相关规则见表6；表6只处理机车穿过 $\mathcal{R}$(0)的情况，并假定$n \geq 1$。 $n = 0$ 的情况另行处理。图27对此进行了说明，表8给出了一些规则。 $n = 1$ 的情况见图28，规则见表9。

表 **6** *The rules managing the crossing of $\mathcal{R}$(0,)general case.*

递增， $n > 0$

单元格0	单元格1(5)
W-WYYWYYY:W 274	Y-WYWGMYY:Y 357
W-WYYWBYY:W 372	Y-WBWGMYY:B 255
W-WYYWRBY:W 371	B-WGMYYWR:R 256
W-WYYWYRB:W 373	R-WYWGMYB:Y 262
W-WYYWYYR:W 273	Y-WYWGMYR:Y 379

递减， $n > 1$

单元格0	单元格1(5)
W-WYYWYYY:W 274	Y-WYWGMYY:Y 357
W-WYYWGYY:W 503	Y-WGWGMYY:R 455
W-WYYWRRY:W 375	R-WRWGMYY:R 453
W-WYYWYRG:W 504	R-WYWGMYG:Y 465
W-WYYWYYR:W 273	Y-WYWGMYR:Y 379

表6可以在图23中得到验证。

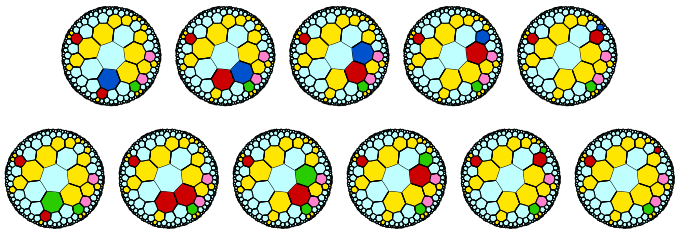


图 **24** *Top a -locomotive going to increment $\mathcal{R}$ in the case when $n > 0$. Bot-*，B，G *tom a -locomotive going to decrement $\mathcal{R}$ in the case when $n > 1$.*

当 $n > 0$ 由表7显示时。规则先处理递增操作，然后处理递减操作，后者用于$n > 1$。

由于无法仅通过机车前端的简单动作来执行寄存器的递增或递减，因此我们可以避免引入两个新状态。寄存器末端的特殊标记M以及机车的两个前部已足以产生不同的工作方式，从而实现图16所示的变换。当存在G前端时，状态M与递减操作关联；当存在B前端时，状态M则与递增操作关联。

当M和B彼此靠近时，M-元胞变为B-，从而触发帽和相邻单元变为B-或M。W单元邻域中出现BB和BM模式后，该单元可以变为G-，随后变为Y-，以构建新的帽，见图25以及规则292和304，还有规则301和308。

当 M 和 G 彼此接近时，还可以采用另一种

方式：M-元胞变成 G，从而可以摧毁帽，同时保留获得新帽所需的元胞。

无论是递增还是递减，都必须以适当颜色的返回机车沿着寄存器的返回轨道返回的方式执行该操作。

在表7、表8和表9中，如果 ξ_u 是规则的带索引编号，则索引 u表示该规则连续迭代的次数。如果没有索引，则该规则仅应用一次。

表 **7** *The rules for the situation when the -cell is $\mathcal{R}(n)$ with $n > 0$ for an* R*, incrementing instruction and with $n > 1$ for a decrementing one.*

递增寄存器

单元格0	元胞1(6)
M-WYYYYYY:M 269	Y-WWWWYMY:Y 270_2
M-WBYYYYY:B 277	Y-WWWWYBM:M 288
B-WRMYYYY:B 281	M-WWWWBBM:Y 298
B-WYMMBBY:W 290	Y-WWWGMWY:Y 306
W-WYYYMRB:W 302	Y-WWWYMWY:Y 315_2
W-WYYYMYR:W 311	
W-WYYYMYY:W 317	

递减寄存器

单元格0	单元格1(4)
M-WYYYYYY:M 269	Y-WWWYMWY:Y 315
M-WGYYYYY:G 387	Y-WWWYMWG:G 386
G-WYWYGGY:Y 394	G-WWWYMWR:R 389
Y-WWWWGMG:Y 401	R-WWWWGWY:Y 87
Y-WWWWRMR:Y 406	Y-WWWWGWY:G 75
Y-WWWWYMY:Y 270	G-WWWWYMY:R 402
	R-WWWWYMY:Y 411
	Y-WWWWYMY:Y 270

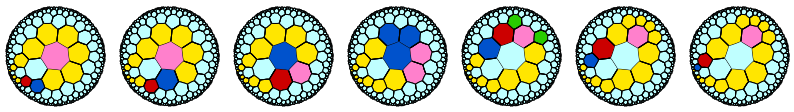


图 **25** *Execution of an incrementing instruction when $n > 0$.*

图25展示了在 $\mathcal{R}$(n) 上执行递增指令的过程，其中 $n > 0$，因此 R-元胞被传送到 $\mathcal{R}$(n+1)。图26展示了执行递减指令的情况，此时同样满足 $n > 0$。在表7中请注意，在递减操作期间，应用了两条规则，即规则87和规则75；这些规则最初是为 G-机车在轨道上的运动而设计的。观察图26和图17可知，图26中局限于单元格1(4)的配置具有相似性。这一观察提醒我们，在对寄存器执行操作期间，表中的所有规则都可用，尤其是那些为轨道和为模拟交叉点和开关而设计的辅助机制所定义的规则，

为轨道以及用于模拟交叉点和开关而设计的辅助机制所定义。

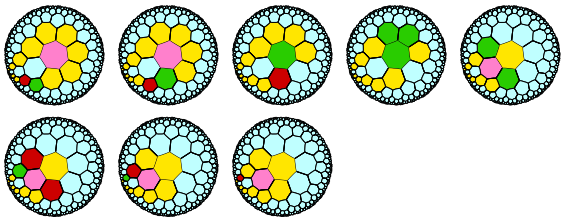


图 **26** *Execution of a decrementing instruction when* $n > 0$.

表8处理寄存器为空时的递增操作和递减操作情况。

表9处理在$n = 1$情况下的递减操作。

我们可以在表7中看到，操作的起始点相同，因此应用相同的规则。

表 **8** *When the register is empty, so that* $\mathcal{R}(0)$ *is*M。

寄存器递增

单元格0	单元格1(4)	元胞1(7)
M-WYYYYYY:M 269	Y-WWYMWWB:B 325	Y-WWWWYMY:Y 270_2
M-WBYYYYY:B 277	B-WWYMWWR:R 335	Y-WWWWYBY:B 282
B-WRMYYYY:B 281	R-WWYWWMB:Y 339	B-WWWWBBM:M 300
B-WYMMBBY:W 290	Y-WWWYWWY:Y 11_3	M-WWGRWYG:M 307
W-WYYYMRB:W 302		M-WYYYYYY:M 269
W-WYYYMYR:W 311		
W-WYYYMYY:W 317		

寄存器递减

单元格0	单元格1(5)	元胞2(6)
M-WYYYYYY:M 269	Y-WGMYYMY:Y 326	Y-WWYMYYM:Y 330_2
M-WGYYYYY:G 387	Y-WGMYYMG:G 420	Y-WWYMYGM:G 431
G-WRGYYYY:M 422	G-WGMYYGR:R 426	G-WWYMYRM:R 440
M-WYRYGGY:M 432	R-WGMGYMY:Y 436	R-WGMYYMY:Y 447
M-WYYYYYY:M 269	Y-WGMRYMY:Y 444	Y-WWRMYYM:Y 449
	Y-WGMYYMY:Y 326	Y-WWYMYYM:Y 330

正如关于图16已经注意到的，对寄存器执行操作会破坏现有的帽，并在M-元胞的新位置周围恢复帽。假设 M-元胞初始位于0，则递增后位于1(7)，递减后位于1(3)。该操作分别由 M-元胞与 B-元胞或 G-元胞组合驱动，形成递增操作或递减操作。请注意，递减后，原来的 M-元胞会成为新 M-元胞的帽中的一个 Y-单元。

观察所考虑的表可以发现，有几条规则出现在两个表中：规则269在这些表中作为形成 M-元胞帽的空闲邻域出现，用于标记寄存器中包含的值。当 M-元胞为单元格0时会出现该规则；此外，当 M-元胞为元胞1(7)时，它还分别出现在递增指令的表8和递减指令的表9中。图27展示了对空寄存器执行这两种操作的情况。

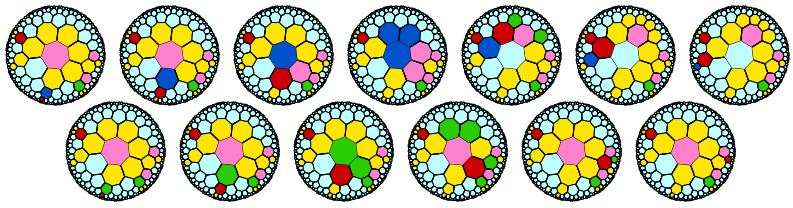


图**27** *Empty register. Top, two rows: incrementing; bottom one row: decrementing.*

我们还可以注意到，邻域中状态的顺序发生了变化，因为新状态可能会改变字母序。例如，规则443的最简形式，即 `Y-WWYWWYM:Y` ，应当接在 `Y-WWYWWYW:Y` 之后，因为状态M变为W。现在，后一个规则的最简形式是`Y-WWWYWWY:Y` ，即规则11，这是一条用于轨道的规则。后一个规则还让我们注意到表的一个性质：这些规则是递增构建的：表10中针对给定配置显示的规则，并不排除之前针对其他配置所确立的规则；后者同样可以在该上下文中使用。

表 **9** *Decrementing the register when its content is* 1, *so that* $\mathcal{R}(2)$ *is* M。

单元格0	元胞1(7)	元胞2(6)
W-WYYYMYY:W 317	M-WYYYYYY:M 269_3	Y-WWYMYYM:Y 330_2
W-WGYYMYY:W 453	M-WGYYYYY:G 387	Y-WWYMYRM:Y 446
W-WRRYMYY:W 456	G-WYWYYYY:Y 473	Y-WWYMGRM:Y 468
W-WYRGMYY:W 388	Y-WYGGYMY:M 476	Y-WWYMYYM:Y 330_7
W-WYYYGYY:M 398	M-WWGMYWY:M 481	
M-WYYYYYY:M 269	M-WWWYGMY:Y 487	
M-WYYYMGY:M 478_2	Y-WWWWMMY:Y 493	
M-WYYYYRG:M 494	Y-WWWWRMY:Y 412	
M-WYYYYYR:M 413	Y-WWWWYMY:Y 270_2	
W-WYYYMYY:W 317		

图28展示了表9中规则的应用。请注意，元胞2(6)的行为与图27中有所不同。当G-机车即将执行递减指令时，只要元胞1(5)从Y变为R，元胞2(6)就保持为Y，如规则446所示。当无法执行递减指令时，元胞1(5)从Y变为G，因此在规则330之后，规则431生效，允许G-机车的前部进入

元胞2(6)。规则431还表明，当元胞2(6)变为G时，元胞1(6)仍保持为Y。相比之下，在 $n > 0$的情况下，当G-机车被派去递减寄存器时，规则446和468展示了机车通过元胞1(6)的过程。

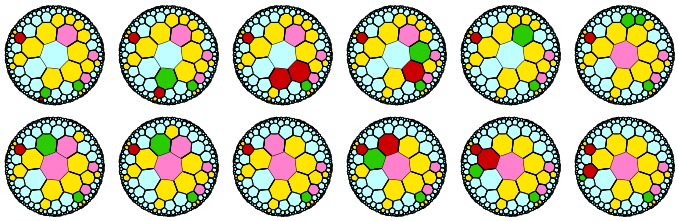


图 **28** *Decrementing a register whose content is* 1.

图28说明了将寄存器变为空寄存器的变换 $n \to n-1$ ：M-元胞从 $\mathcal{R}(1)$ 移动到 $\mathcal{R}(0)$。

表10中有510条规则，这些规则由一个计算机程序进行了测试。该程序根据这些规则生成图，使我能够检查其正确性。由于停止指令只是一个通向G-滤子的轨道，而机车随后是一个B-单位，因此无需为此目的添加其他规则。

表10展示了自动机所使用的完整规则集。我们采用与表1至表9相同的约定：每条规则显示为 `c-ABCDEFG:n`，其中 c 是当前状态， n 是执行该规则后的新状态。元胞邻居所处的状态按照 `ABCDEFG`中的字母序显示，这一顺序不一定是从邻接铺砌块1到邻接铺砌块7的顺序。

表 **10** *Table of the rules used by the cellular automaton.*

初步规则

1	W	WWWWWWW	W
2	W	WWWWWWY	W
3	Y	WWWWWWW	Y
4	W	WWWWWWB	W
5	W	WWWWWWG	W
6	W	WWWWWWR	W
7	W	WWWWWBR	W
8	W	WWWWWGR	W
9	W	WWWWWRB	W
10	W	WWWWWRG	W
11	Y	WWWYWWY	Y
12	W	WWWWWYY	W
13	W	WWWWWYB	W
14	W	WWWWWRY	W
15	Y	WWWWWWY	Y

轨道

B-locomotive			
16	W	WWWWWBY	W
17	B	WWWRWWY	R
18	R	WWWYWWB	Y
19	W	WWWWWYR	W
20	Y	WWWWWWR	Y
21	Y	WWWYWWR	Y
22	Y	WWWWWWB	Y
G-locomotive			
23	W	WWWWWGY	W
24	G	WWWRWWY	R
25	R	WWWYWWG	Y
26	W	WWWWWYG	W
27	Y	WWWWWWG	Y

弧上的轨道

28	W	WWYYYYY	W
B-locomotive			
29	W	WWBYYYY	W
30	W	WWRBYYY	W
31	W	WWYRBYY	W
32	W	WWYYRBY	W
33	W	WWYYYRB	W
34	W	WWYYYYR	W
35	W	WWYYYYB	W
36	Y	WWWWBWY	B
37	W	WWYYYBR	W
38	B	WWWWRWY	R
39	W	WWYYBRY	W
40	R	WWWWYWB	Y
41	W	WWYBRYY	W
42	Y	WWWWYWR	Y

弧上的轨道，续

B-locomotive, cont.

43	W	WWWYYYB	W	
44	W	WWBRYYY	W	
45	W	WWRYYYY	W	
46	W	WWWYBRY	W	
47	W	WWWBRYY	W	
48	W	WWWRYYY	W	
49	W	WWWWYRB	W	
50	W	WWWBYYY	W	
51	W	WWWYYYY	W	
52	R	WWWWBWY	Y	
53	B	WWWYWWR	R	
54	Y	WWWWYWB	B	
55	Y	WWWWYWY	Y	
56	W	WWWWYYY	W	
57	W	WWWWYYR	W	
58	W	WWWRBYY	W	
59	Y	WWWWRWY	Y	
60	R	WWWBWWY	Y	
61	W	WWWYRBY	W	
62	Y	WWWRWWY	Y	
63	B	WWWWYWR	R	
64	Y	WWWYWWB	B	
65	W	WWWYYRB	W	
66	W	WWWYYYR	W	
67	W	WWWWBYY	W	
68	W	WWWWRBY	W	

G-locomotive

69	W	WWGYYYY	W	
70	W	WWRGYYY	W	
71	W	WWYRGYY	W	
72	W	WWYYRGY	W	
73	W	WWYYYRG	W	
74	W	WWYYYYG	W	
75	Y	WWWWGWY	G	
76	W	WWYYYGR	W	
77	G	WWWWRWY	R	
78	W	WWYYGRY	W	
79	R	WWWWYWG	Y	
80	W	WWYGRYY	W	
81	W	WWWGRYY	W	
82	W	WWWYYYG	W	
83	W	WWGRYYY	W	
84	W	WWWYGRY	W	
85	W	WWWWYRG	W	
86	W	WWWGYYY	W	
87	R	WWWWGWY	Y	
88	G	WWWYWWR	R	
89	Y	WWWWYWG	G	
90	W	WWWRGYY	W	
91	R	WWWGWWY	Y	
92	W	WWWYRGY	W	
93	G	WWWWYWR	R	
94	Y	WWWYWWG	G	
95	W	WWWYYRG	W	
96	W	WWWWGYY	W	
97	W	WWWWRGY	W	

fork

98	Y	WWYWYWY	Y	

B-locomotive

99	Y	WWBWYWY	B	
100	W	WWWWYYB	W	
101	B	WWRWYWY	R	
102	W	WWWWYBY	W	
103	W	WWWWYBR	W	
104	R	WWYWBWB	Y	
105	W	WWWWBRB	W	
106	W	WWWWBRY	W	
107	Y	WWYWRWR	Y	
108	W	WWWWRYR	W	
109	W	WWWWRYY	W	
110	R	WWWYWWY	Y	

G-locomotive

111	Y	WWGWYWY	G	
112	W	WWWWYYG	W	
113	G	WWRWYWY	R	
114	W	WWWWYGY	W	
115	W	WWWWYGR	W	
116	R	WWYWGWG	Y	
117	W	WWWWGRG	W	
118	W	WWWWGRY	W	

固定开关

119	Y	WWYWYRY	Y	
120	R	WWWWYYY	R	

from left

B-locomotive

121	Y	WWYRWWB	B	
122	Y	WWYWWRY	Y	
123	Y	WWYWYRB	B	
124	R	WWWWBYY	R	
125	B	WWYRWWR	R	
126	B	WWYWYRR	R	
127	R	WWWWRBY	R	
128	R	WWYWWBR	Y	
129	W	WWWWWRR	W	
130	Y	WWYWWRB	Y	
131	R	WWBWYRY	Y	
132	R	WWWWYRY	R	
133	Y	WWYWWRR	Y	
134	Y	WWRWYRY	Y	
135	Y	WWYWWYR	Y	

G-locomotive

136	Y	WWYRWWG	G	
137	Y	WWYWYRG	G	
138	R	WWWWGYY	R	
139	G	WWYRWWR	R	
140	G	WWYWYRR	R	
141	R	WWWWRGY	R	
142	R	WWYWWGR	Y	
143	Y	WWYWWRG	Y	
144	R	WWGWYRY	Y	

from right

B-locomotive

145	Y	WWBWWRY	B	
146	Y	WWYWBRY	B	
147	R	WWWWYYB	R	
148	B	WWRWWRY	R	
149	B	WWYWRRY	R	
150	R	WWWWYBR	R	
151	Y	WWYWWBR	Y	
152	Y	WWWBWWY	B	
153	R	WWYWWRB	Y	

G-locomotive

154	Y	WWGWWRY	G	
155	Y	WWYWGRY	G	
156	R	WWWWYYG	R	
157	G	WWRWWRY	R	
158	G	WWYWRRY	R	
159	R	WWWWYGR	R	
160	Y	WWYWWGR	Y	
161	Y	WWWGWWY	G	
162	R	WWYWWRG	Y	

转换器

from B-locom. to G-one

163	Y	WWWYGWY	Y	
164	G	WWYYWYY	G	
165	Y	WWWWWYG	Y	

166 Y WWWWWGY Y
167 Y WWBWWGY G
168 Y WWWGGWY G
169 G WWYGWYY G
170 W WWWWYGG W
171 G WWGYWWR R
172 G WWWRGWY R
173 G WWGRWYY G
174 R WWYWWGG Y
175 R WWWYGWG Y
176 G WWRYWYY G
177 Y WWWYGWR Y
178 W WWWWRYG W
179 Y WWYWWGY Y

from G-locom. to B-one

180 Y WWWYBWY Y
181 B WWYYWYY B
182 Y WWWWWYB Y
183 Y WWWWWBY Y
184 Y WWBYWWG B
185 Y WWWBBWY B
186 B WWYBWYY B
187 W WWWWYBB W
188 B WWBYWWR R
189 B WWWRBWY R
190 B WWBRWYY B
191 R WWYWWBB Y
192 R WWWYBWB Y
193 B WWRYWYY B
194 Y WWWYBWR Y
195 W WWWWRYB W
196 Y WWYWWBY Y

B-滤子

B-locomotive, OK

197 B WYYWRYR B
198 Y WWRBRWY Y
199 R WWWWWBY R
200 W WWWYYBR W
201 Y WWWBWBY B
202 W WWWRBYB W
203 R WWWWWYB R
204 B WYBWRYR B
205 B WWWRWBY R
206 W WWWRBBR W
207 B WBRWRYR B
208 W WWWYBBR W
209 R WWWYWBB Y
210 W WWWRBRY W
211 B WRYWRYR B
212 W WWWBRBR W
213 Y WWWYWBR Y
214 W WWWRYBR W
215 Y WWWYWBY Y

G-locomotive, no

216 Y WWWGWBY Y
217 W WWWRBYG W
218 Y WWWRWBY Y
219 W WWWRBYR W

G-滤子

G-locomotive, OK

220 G WYYWRYR G
221 Y WWRGRWY Y
222 R WWWWWGY R
223 W WWWYYGR W
224 Y WWWGWGY G
225 W WWWRGYG W
226 R WWWWWYG R
227 G WYGWRYR G
228 G WWWRWGY R
229 W WWWRGGR W
230 G WGRWRYR G
231 W WWWYGGR W
232 R WWWYWGG Y
233 W WWWRGRY W
234 G WRYWRYR G
235 W WWWGRGR W
236 Y WWWYWGR Y
237 W WWWRYGR W
238 Y WWWYWGY Y

B-locomotive, no

239 Y WWWBWGY Y
240 W WWWRGYB W
241 Y WWWRWGY Y
242 W WWWRGYR W

变化滤子

from B- to G-

243 Y WWRBRWB G
244 W WWWWBYR W
245 B WYYWRGR G
246 G WWRBRWR Y
247 W WWWWRGR W
248 R WWWWWBG R
249 R WWWWWGB R

from G- to B-

250 Y WWRGRWB B
251 G WYYWRBR B
252 B WWRGRWR Y
253 W WWWWRBR W

寄存器

incrementing

across $\mathcal{R}(0)$ to $\mathcal{R}(n)$

254 W WWWBWYY W
255 Y WBWGMYY B
256 B WGMYYWR R
257 W WWWWGBR W
258 G WWYYWMB G
259 M WWWWYBG M
260 Y WWBYMWY B
261 Y WWYMYBM Y
262 R WYWGMYB Y
263 B WWRYMWY R
264 Y WWYMBRM Y
265 M WWWWBYY M
266 W WWWWYBM W
267 R WWYYMWB Y
268 W WWWWBRM W

$n \to n$+1,
$n > 0$

269 M WYYYYYY M
270 Y WWWWYMY Y
271 Y WWWYWMY Y
272 W WRBYMYY W
273 W WYYWYYR W
274 W WYYWYYY W
275 W WWWWYWY W
276 Y WWWYMWB B
277 M WBYYYYY B
278 W WYRBMYY W
279 B WWWYMWR R

280	Y	WWWWYMB	M	325	Y	WWYMWWB	B			back to program				
281	B	WRMYYYY	B	326	Y	WGMYYMY	Y	370	Y	WWYYMWY	Y			
282	Y	WWWWYBY	B	327	G	WWYYWMY	G	371	W	WYYWRBY	W			
283	W	WYYRBYY	W	328	M	WWWWYYG	M	372	W	WYYWBYY	W			
284	R	WWWMBWY	Y	329	W	WWWWMGY	W	373	W	WYYWYRB	W			
285	M	WWWWYBR	M	330	Y	WWYMYYM	Y	374	Y	WWBWRWY	B			
286	W	WWWWWMR	W	331	Y	WWYWYYM	Y			decrementing				
287	W	WWWWWWM	W	332	M	WWWWYYY	M	across $\mathcal{R}(0)$ to $\mathcal{R}(n)$						
288	Y	WWWWYBM	M	333	W	WWWWMYY	W	375	W	WYYWRRY	W			
289	W	WWWWWYM	W	334	W	WWWBMYY	W	376	Y	WWRYMWY	G			
290	B	WYMMBBY	W	335	B	WWYMWWR	R	377	W	WYYWYRY	W			
291	B	WWWWYBB	R	336	Y	WGMYYMB	M	378	G	WWRYMWY	R			
292	W	WWWWWBB	G	337	W	WWWWGYB	W	379	Y	WYWGMYR	Y			
293	Y	WWWYWBB	B	338	Y	WYWBYWR	Y	380	R	WWYYMWG	Y			
294	W	WYYYBYY	W	339	R	WWYWWMB	Y	381	Y	WWYMRYM	Y			
295	Y	WWWMBWY	Y	340	M	WGMYYBR	M	382	M	WWWWRYY	M			
296	M	WWWWMBY	Y	341	W	WWWWGMR	W	383	W	WWWWGRM	W			
297	W	WWWWWMY	W	342	G	WWYYWMM	G	384	Y	WWYYMWR	Y			
298	M	WWWWBBM	Y	343	M	WWWWYMG	M	$n \to n-1$,						
299	W	WWWWWMM	W	344	Y	WWYBMYM	M	$n > 1$						
300	B	WWWWBBM	M	345	Y	WWYMYMM	Y	385	W	WRGYMYY	W			
301	W	WWWWWBM	G	346	Y	WYWBBWR	B	386	Y	WWWYMWG	G			
302	W	WYYYMRB	W	347	W	WWWYBYY	W	387	M	WGYYYYY	G			
303	R	WWWBWMG	Y	348	Y	WWYWWMB	Y	388	W	WYRGMYY	W			
304	G	WWWWWRM	Y	349	M	WGMYMBY	Y	389	G	WWWYMWR	R			
305	W	WYYYWBY	W	350	W	WWWWGMY	W	390	Y	WWWWYMG	W			
306	Y	WWWGMWY	Y	351	M	WWBBMYM	Y	391	G	WYYYYWR	G			
307	M	WWGRWYG	M	352	Y	WWYMMMM	Y	392	Y	WWWWYGY	G			
308	G	WWWWWMY	Y	353	M	WWWWMYY	M	393	W	WYYRGYY	W			
309	W	WWWWWMG	Y	354	B	WWRWRWY	R	394	G	WYWYGGY	Y			
310	W	WWWWWGM	Y	355	R	WWYYWWB	R	395	G	WWWWYGG	W			
311	W	WYYYMYR	W	356	W	WWWYWBY	W	396	W	WWWWWGG	W			
312	Y	WWWRWMY	Y	357	Y	WYWGMYY	Y	397	Y	WWWYWGG	G			
313	Y	WWWWWYM	Y	358	Y	WYYMWGM	Y	398	W	WYYYGYY	M			
314	W	WYYYWRB	W	359	W	WWWWGYM	W	399	Y	WWWWWGG	W			
315	Y	WWWYMWY	Y	360	R	WWYWRWB	Y	400	G	WWWWGGY	W			
316	M	WWYYWYY	M	361	W	WWWWRRY	W	401	Y	WWWWGMG	Y			
317	W	WYYYMYY	W	362	R	WWYYWWR	R	402	G	WWWWYMY	R			
318	W	WYYYWYR	W	363	W	WWWWBRR	W	403	M	WYYGYGY	M			
319	W	WYYYWRY	W	364	W	WWWYWRB	W	404	Y	WWWYWMG	G			
$0 \to 1$				365	Y	WYYMWYM	Y	405	Y	WWWWGMY	Y			
320	Y	WYWMYWR	Y	366	W	WWWWYYM	W	406	Y	WWWWRMR	Y			
321	R	WWYWWYY	R	367	W	WWWYWYR	W	407	R	WWWWGMY	Y			
322	Y	WWWWWYR	Y	368	Y	WWYWRWY	Y	408	M	WYYRYRG	M			
323	Y	WWWWWRY	Y	369	W	WWWYWYY	W	409	G	WWWYWMR	R			
324	W	WWWYMYY	W					410	W	WYYYMGY	W			

411 R WWWWYMY Y

412 Y WWWWRMY Y

413 M WYYYYYR M

414 R WWWGWMY Y

415 W WYYYMRG W

416 W WYYYWGY W

decrementing

$n = 0$, failure

417 Y WWYMWWG G

418 W WWWGMYY W

419 G WWYMWWR R

420 Y WGMYYMG G

421 W WWWWGYG W

422 G WRGYYYY M

423 Y WYWGYWR Y

424 R WWYWYGG Y

425 W WWWWYRY W

426 G WGMYYGR R

427 W WWWWGGR W

428 G WWYYWMG G

429 M WWWWYGG M

430 Y WWYGGYM Y

431 Y WWYMYGM G

432 M WYRYGGY M

433 G WWWWYMG Y

434 Y WYWMGWR Y

435 Y WWYWWRM Y

436 R WGMGYMY Y

437 G WWYYWMR G

438 M WWWWGRG M

439 Y WWGMRGM Y

440 G WWYMYRM R

441 M WWWWYGY M

442 G WWWWGMY Y

443 Y WWYWWYM Y

445 M WWWYRYG M446 Y

449 Y WWRMYYM Y450 M

$1 \to 0$

453 W WGYYMYY W

454 W WWWGWYY W

455 Y WGWGMYY R

456 W WRRYMYY W

457 W WWWRWYY W

458 R WWWYWWR Y

459 R WGMYYWR R

460 M WWWWYRG M

461 W WWWWGRR W

462 Y WYMWRYM G

463 Y WWYWYRM Y

464 W WYYGMYY W

465 R WYWGMYG Y

466 G WYMWRYM Y

467 Y WWYWGRM Y

468 Y WWYMGRM Y

469 M WWWWGYY M

470 W WWWWYGM W

471 Y WYWGWYR Y

472 Y WWGWYYM Y

473 G WYWYYYY Y

474 Y WWWYMYG G

475 G WWWWYYG W

476 Y WYGGYMY M

477 G WWWWGYY W

478 M WYYYMGY M

479 G WWWWYMM G

480 Y WWMMYYM Y

481 M WWGMYWY M

482 W WWWWYMY W

483 Y WWWWWWM W

484 G WWWYMMY M

485 Y WWWWWGM W

486 Y WWYMMYY Y444 Y WGMRYMY Y

WWYMYRM Y447 R WGMYYMY Y448 Y WWWWWRM W

WWWWYYR M451 R WWWYWMY Y452 W WWWWMRY W

487 M WWWYGMY Y

488 Y WWWWWMY W

489 M WYYYYMY M

490 M WWWWYMY R

491 Y WYWMMWR G

492 W WWWWRYM W

493 Y WWWWMMY Y

494 M WYYYYRG M

495 G WYWMRWR R

496 R WWYYWWG R

497 W WWWYMGY W

498 R WGWMYWR Y

499 W WWWYMRG W

500 Y WRWRWMY Y

501 W WWWYMYR W

back to program

502 W WYYWRGY W

503 W WYYWGYY W

504 W WYYWYRG W

505 Y WWGWRWY G

506 W WYYYWRG W

507 G WWRWRWY R

508 W WWWYWGY W

509 R WWYWRWG Y

510 W WWWYWRG W

4 结论

我们可以看到，表中共有510条规则，包括守恒规则、见证规则和运动规则。请注意，仅寄存器就需要257条规则，其中135条用于寄存器内容为0或1时的特殊情况。在表10中，在前面各表规则的基础上增加了许多规则；前面各表只涉及少数元胞，而表10的附加规则涉及电路中的所有元胞。

我们仍面临这样一个开放性问题：是否有可能减少状态的数量。如果放宽旋转不变性这一条件，这很可能是可以实现的。在这种情况下，就没有必要区分运动方向。

参考文献

[1] F. Herrmann，M. Margenstern，《双曲平面中的通用元胞自动机》，理论计算机科学，296，(2003)，327-364。[2] M. Margenstern，《双曲平面元胞自动机的新工具》，《通用计算机科学杂志》，6(12)，(2000)，1226-1252。[3] M. Margenstern，《双曲空间中的元胞自动机》，第1卷，《理论》，丛书：《非常规计算与元胞自动机进展》，主编：Andrew Adamatzky，Old City Publishing，费城，(2007)，422页。[4] M. Margenstern，《五边形网格上强通用双曲元胞自动机的状态数上界》，JAC2010，图尔库，芬兰，(2010)，会议论文集，图尔库计算机科学中心，168-179。[5] M. Margenstern，《双曲空间中的小型通用元胞自动机：一套珍宝》，施普林格出版社，(2013)，331页。[6] M. Margenstern，《五边形网格上具有五个状态的弱通用元胞自动机》，《计算机科学讲义》，C.S. Calude 等（编）：Gruska纪念文集，8808，(2014)，99-113。[7] M. Margenstern，《七边形网格上具有三个状态的弱通用元胞自动机》，arXiv:1410.1864v1，(2014)，27页。[8] M. Margenstern，《双曲平面七边形网格上的弱通用元胞自动机》，《复杂系统》，27(4)，(2018)，315-354。[9] M. Margenstern，《七边形网格上具有七个状态的强通用元胞自动机》，arXiv:2102.04292v1，(2021)，43页。[10] M. Margenstern，《七边形网格上具有6个状态的弱通用加权元胞自动机》，arXiv:2301.10691v2，(2023)，37页。

[11] M. Margenstern，Y. Song，《五边形网格上的一种新型通用元胞自动机》，Parallel Processing Letters，19(2)，(2009)，227-246。[12] M.L. Minsky，《计算：有限与无限机器》，Prentice-Hall，Englewood Cliffs，新泽西州，1967。[13] I. Stewart，《一条名为图灵的地铁》，《Scientific American》中的数学趣题，(1994)，90-92。

