Cellular Automata Classics

Jarkko Kari

Department of Mathematics, University of Turku, Finland

TUCS (Turku Centre for Computer Science)

Cellular Automata (CA): Introduction

Cellular automata are among the oldest models of natural computing. They are versatile objects of study, investigated

- **in physics** as discrete models of physical systems,
- **in computer science** as models of massively parallel computation under the realistic constraints of locality and uniformity,
- **in mathematics** as endomorphisms of the full shift in the context of symbolic dynamics.

Cellular automata possess several fundamental properties of the physical world: they are

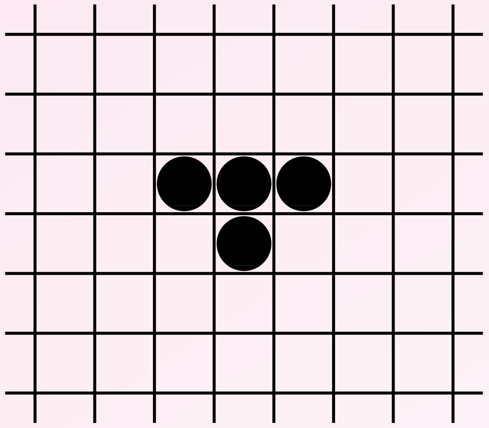
- massively parallel,
- homogeneous in time and space,
- all interactions are local,
- time reversibility and conservation laws can be obtained by choosing the local update rule properly.

Example: the **Game-of-Life** by John Conway.

- Infinite checker-board whose squares (=cells) are colored black (=**alive**) or white (=**dead**).
- At each discrete time step each cell counts the number of living cells surrounding it, and based on this number determines its new state.
- All cells change their state simultaneously.

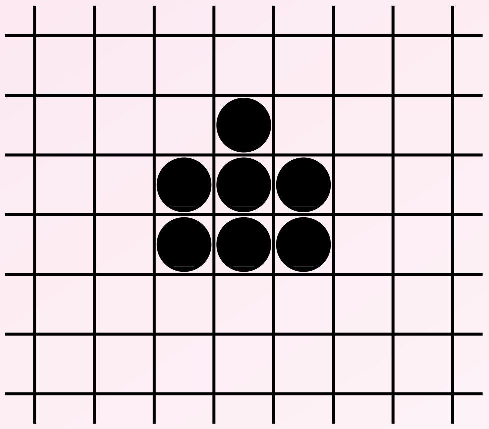
The local update rule asks each cell to check the present states of the eight surrounding cells.

- If the cell is **alive** then it stays alive (survives) iff it has two or three live neighbors. Otherwise it dies of loneliness or overcrowding.
- If the cell is **dead** then it becomes alive iff it has exactly three living neighbors.



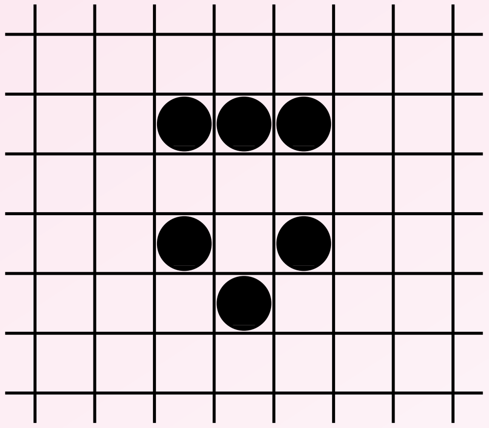
The local update rule asks each cell to check the present states of the eight surrounding cells.

- If the cell is **alive** then it stays alive (survives) iff it has two or three live neighbors. Otherwise it dies of loneliness or overcrowding.
- If the cell is **dead** then it becomes alive iff it has exactly three living neighbors.



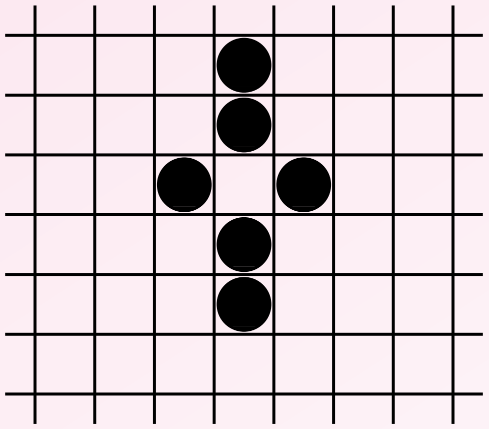
The local update rule asks each cell to check the present states of the eight surrounding cells.

- If the cell is **alive** then it stays alive (survives) iff it has two or three live neighbors. Otherwise it dies of loneliness or overcrowding.
- If the cell is **dead** then it becomes alive iff it has exactly three living neighbors.

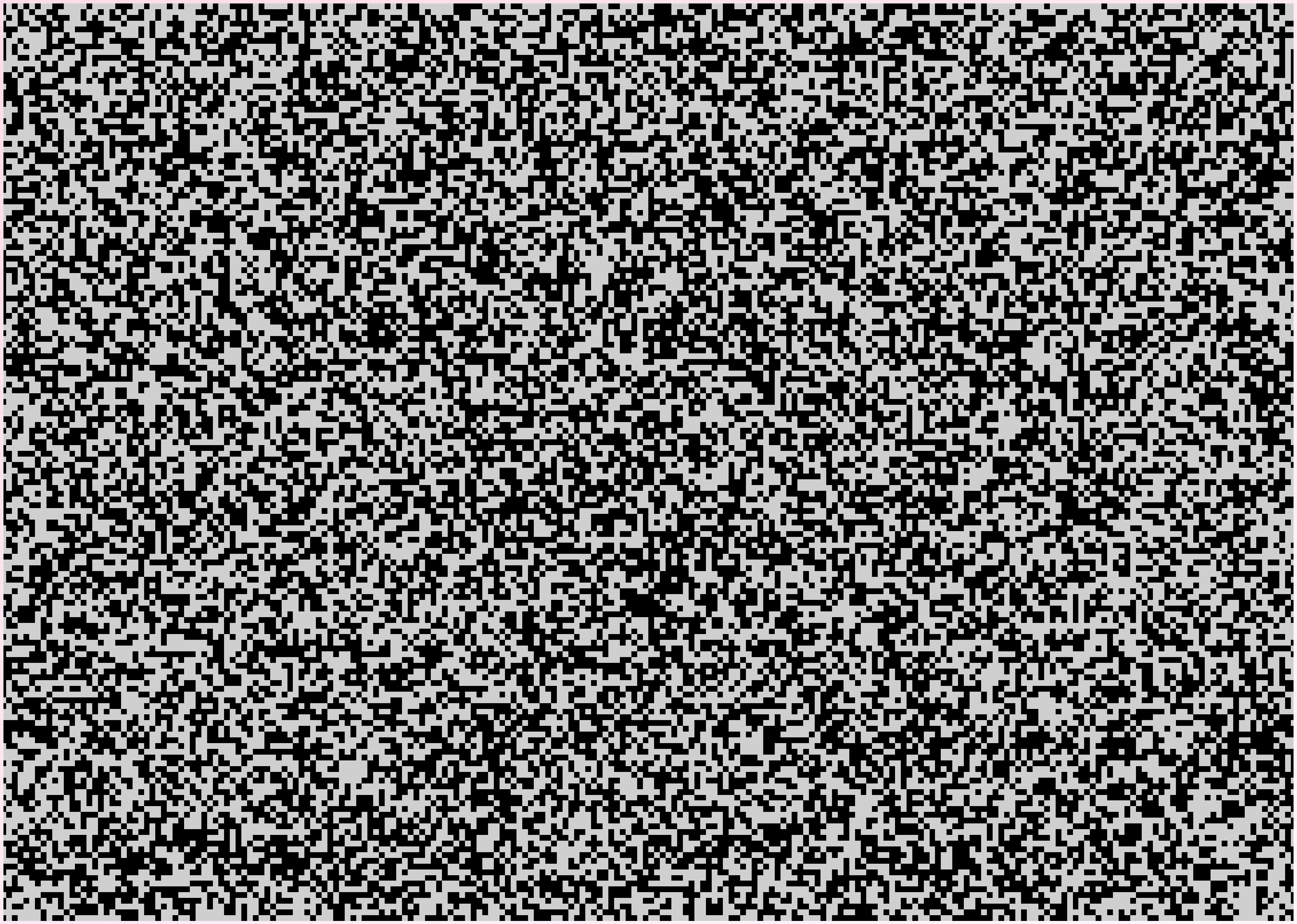


The local update rule asks each cell to check the present states of the eight surrounding cells.

- If the cell is **alive** then it stays alive (survives) iff it has two or three live neighbors. Otherwise it dies of loneliness or overcrowding.
- If the cell is **dead** then it becomes alive iff it has exactly three living neighbors.

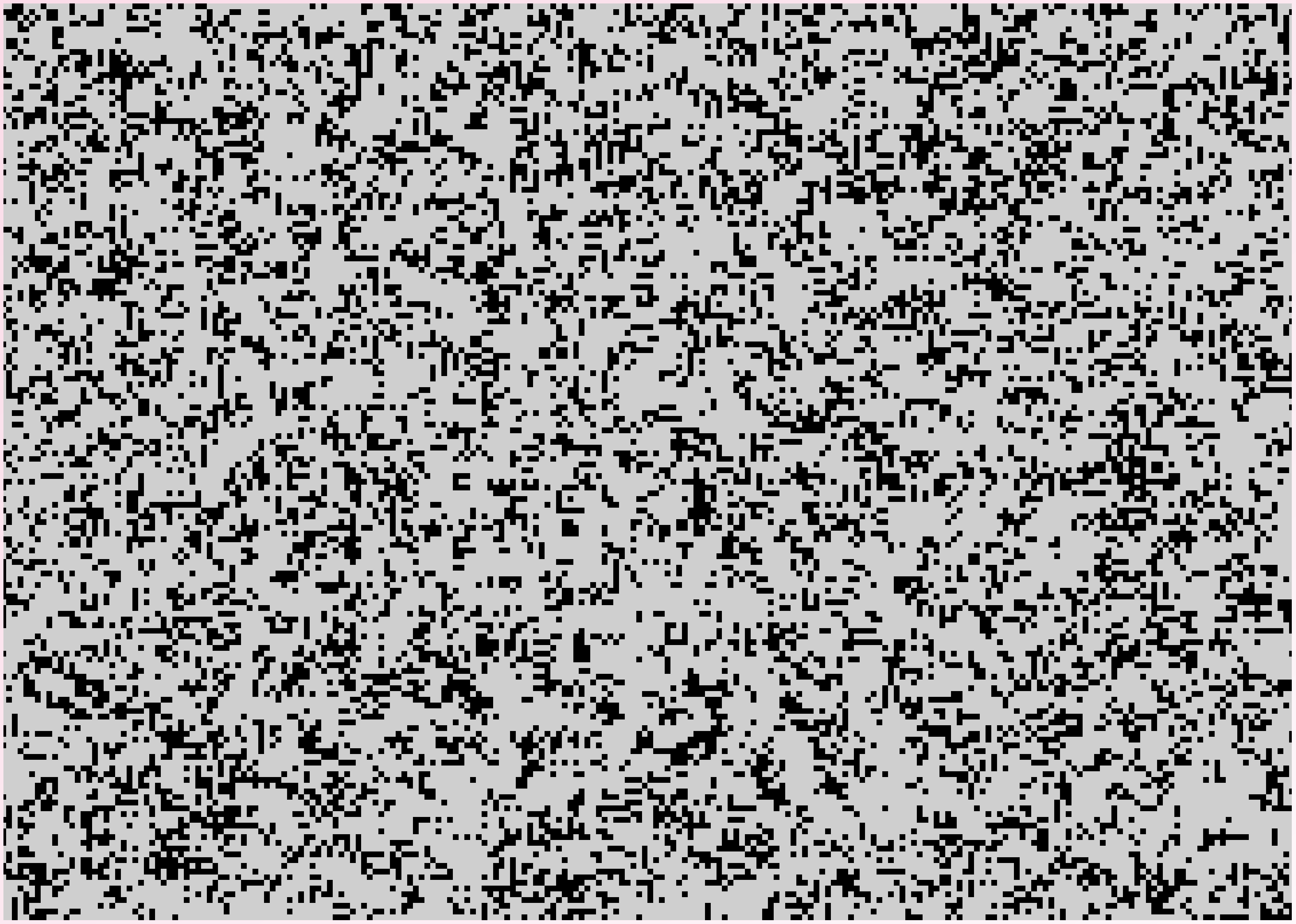


A typical snapshot of a time evolution in Game-of-Life:



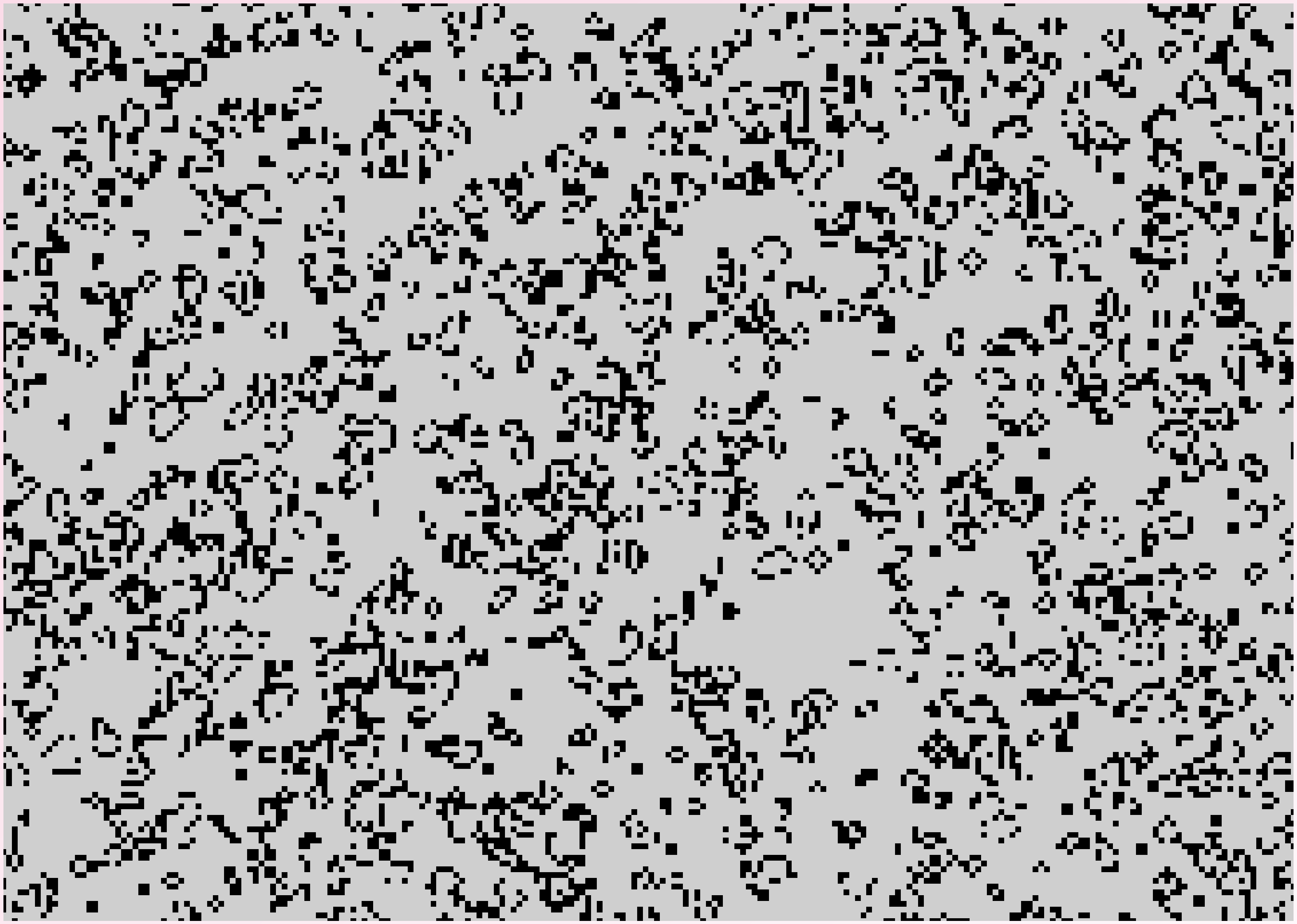
Initial uniformly random configuration.

A typical snapshot of a time evolution in Game-of-Life:



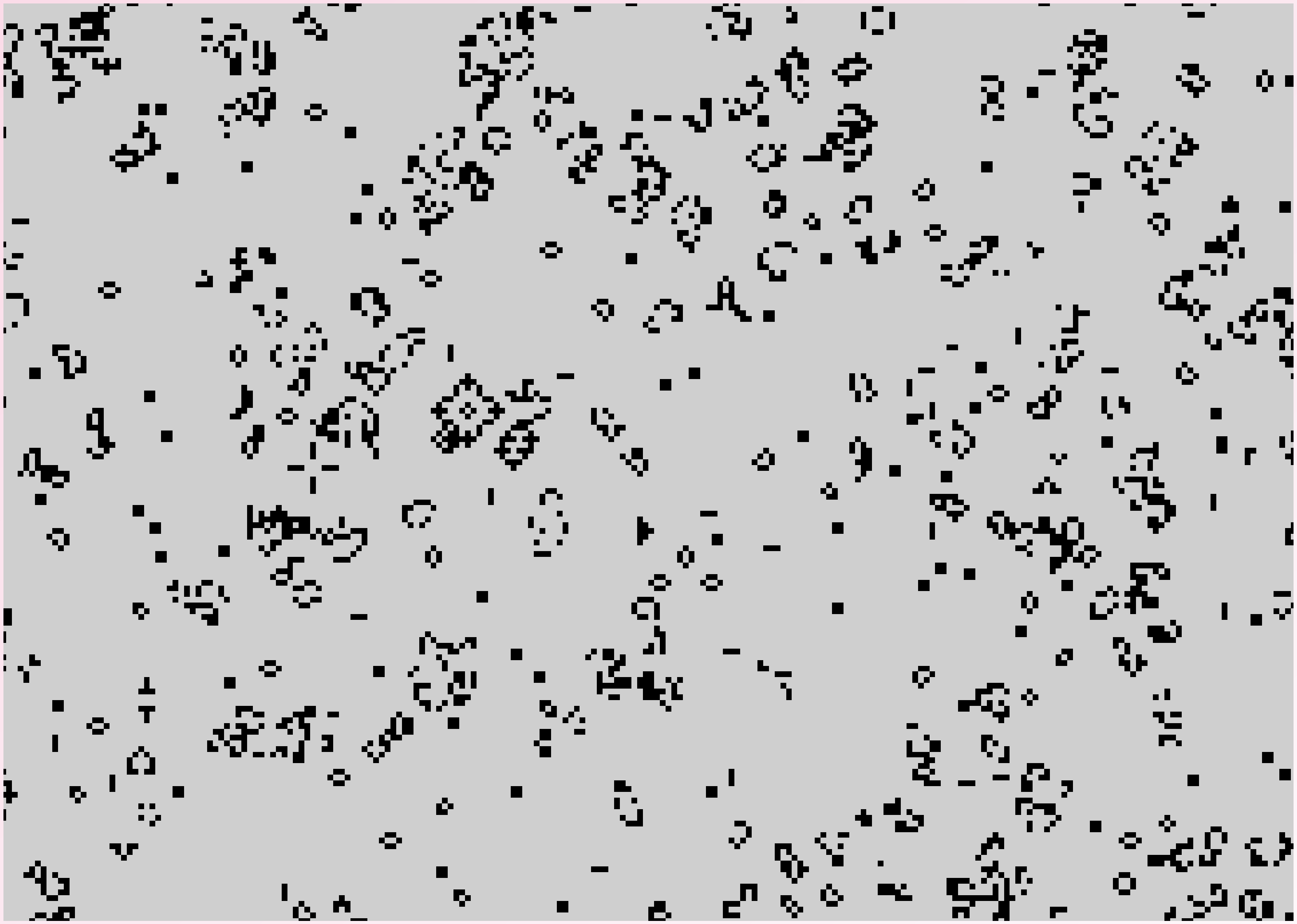
The next generation after all cells applied the update rule.

A typical snapshot of a time evolution in Game-of-Life:



Generation 10

A typical snapshot of a time evolution in Game-of-Life:



Generation 100

GOL is a **computationally universal** two-dimensional CA: using **gliders** as information, one can implement logical gates **AND**, **OR** and **NOT**.

It is then possible to simulate a computationally universal counter machine.

Another famous universal CA: **rule 110** by S.Wolfram.

A one-dimensional CA with binary state set $\{0,1\}$, i.e. a two-way infinite sequence of 0's and 1's.

Each cell is updated based on its old state and the states of its left and right neighbors as follows:

$$\begin{array}{ccc}
111 & \longrightarrow & 0 \\
110 & \longrightarrow & 1 \\
101 & \longrightarrow & 1 \\
100 & \longrightarrow & 0 \\
011 & \longrightarrow & 1 \\
010 & \longrightarrow & 1 \\
001 & \longrightarrow & 1 \\
000 & \longrightarrow & 0
\end{array}$$

Another famous universal CA: **rule 110** by S.Wolfram.

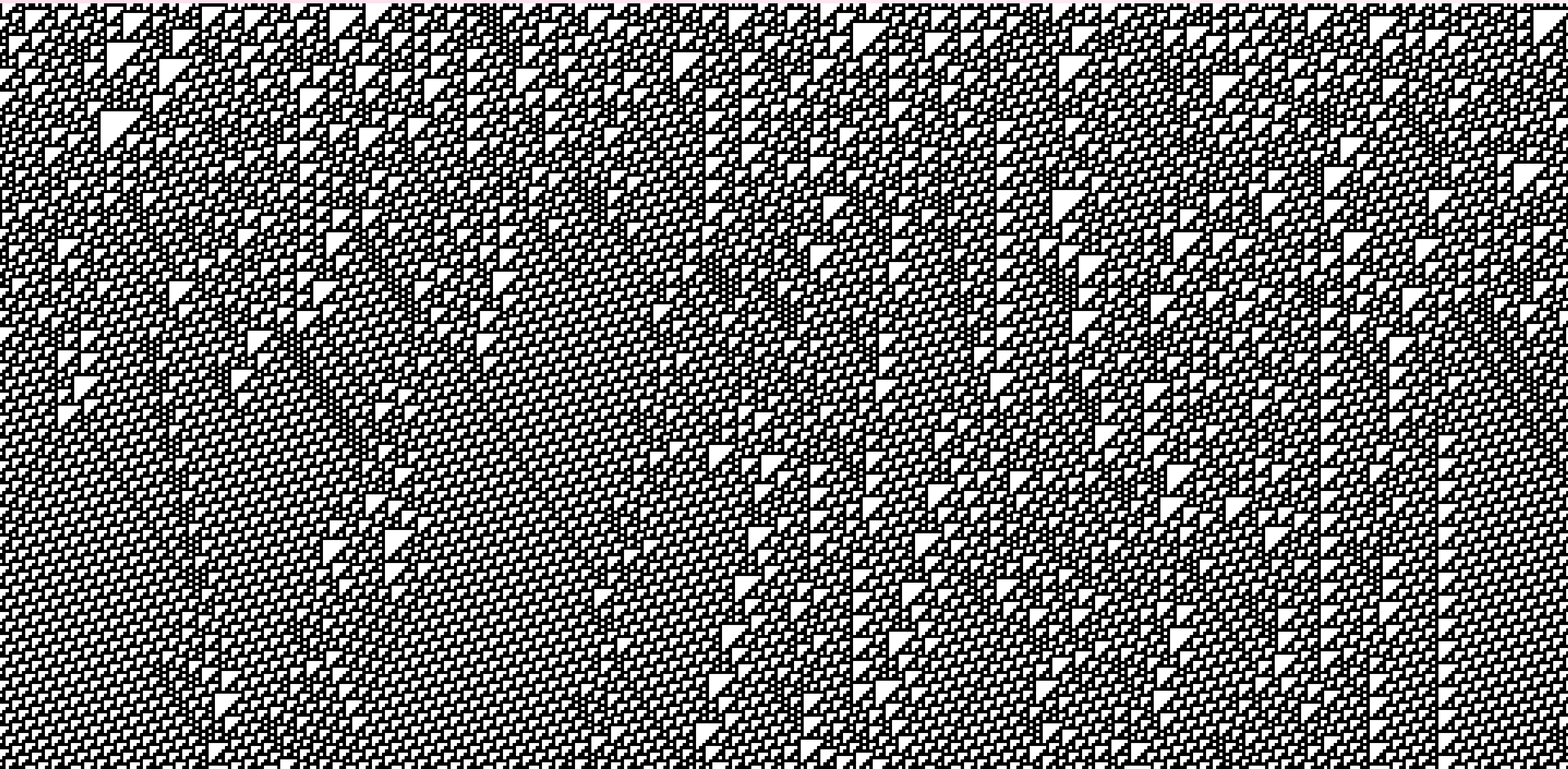
A one-dimensional CA with binary state set $\{0, 1\}$, i.e. a two-way infinite sequence of 0's and 1's.

Each cell is updated based on its old state and the states of its left and right neighbors as follows:

$$\begin{array}{ccc}
111 & \longrightarrow & 0 \\
110 & \longrightarrow & 1 \\
101 & \longrightarrow & 1 \\
100 & \longrightarrow & 0 \\
011 & \longrightarrow & 1 \\
010 & \longrightarrow & 1 \\
001 & \longrightarrow & 1 \\
000 & \longrightarrow & 0
\end{array}$$

110 is the **Wolfram number** of this CA rule.

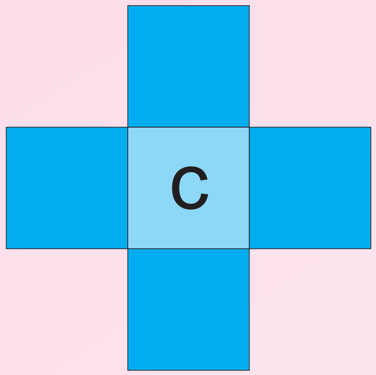
Space-time diagram is a pictorial representation of a time evolution in one-dimensional CA, where space and time are represented by the horizontal and vertical direction:

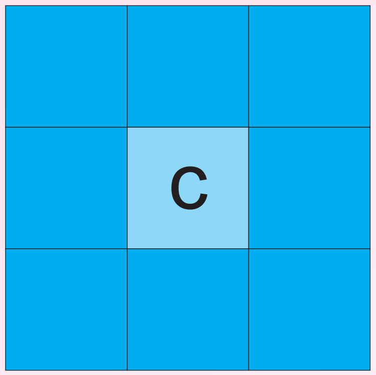


General definition of d-dimensional CA

- Finite **state set** S.
- **Cells** are indexed by integer coordinates $\mathbb{Z}^d$.
- **Configurations** $\mathbb{Z}^d \longrightarrow S$ assign states to all cells. The set of all configurations is $S^{\mathbb{Z}^d}$.
- A **neighborhood** is a finite set $N \subseteq \mathbb{Z}^d$ that provides the relative offsets to neighbors.
- The **neighbors** of a cell at location $\vec{x} \in \mathbb{Z}^d$ are the cells at locations
$$\vec{x} + \vec{n}, \text{ for } \vec{n} \in N.$$

Typical two-dimensional neighborhoods:





Von Neumann

neighborhood

$\{(0,0),(\pm 1,0),(0,\pm 1)\}$

Moore

neighborhood

$\{-1,0,1\}\times\{-1,0,1\}$

The **local rule** is a function

$$f : S^n \longrightarrow S$$

where n is the size of the neighborhood.

State $f(a_1, a_2, \ldots, a_n)$ is the new state of a cell whose n neighbors were at states $a_1, a_2, \ldots, a_n$ one time step before.

Global dynamics of the CA: Configuration c becomes in one time step the configuration e obtained by applying the local rule at every cell.

The transformation

$$G : S^{\mathbb{Z}^d} \longrightarrow S^{\mathbb{Z}^d}$$

that maps $c \mapsto e$ is the **global transition function** of the CA.

Function G is our main object of study and we simply call it a **CA function.** In algorithmic questions we use its finite presentation (the local rule).

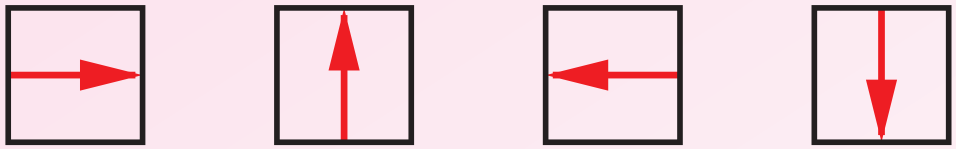
Lattice gas example

CA have traditionally been used in simulations of physical systems. Good examples are **lattice gases**.

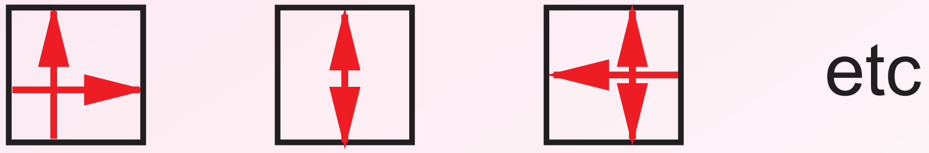
These are CA simulations of fluid or gas dynamics based on storing individual molecules in the cells and implementing particle interactions by the CA local rule.

Simplest lattice gas model: **HPP** (due to Hardy, Pomeau and de Pazzis).

A two-dimensional CA where each cell can store up to four moving particles. Each particle has a direction of movement which can be up, down, left or right:

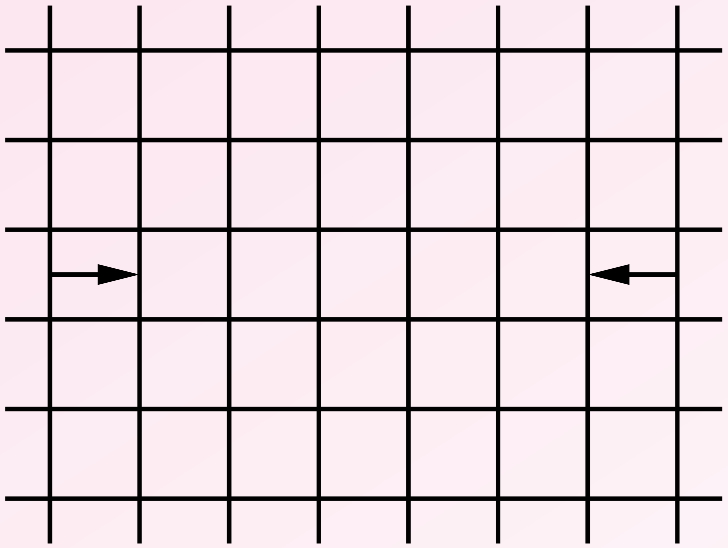


There can be at most one particle of each direction in any individual cell. So there are $2^4 = 16$ possible states:



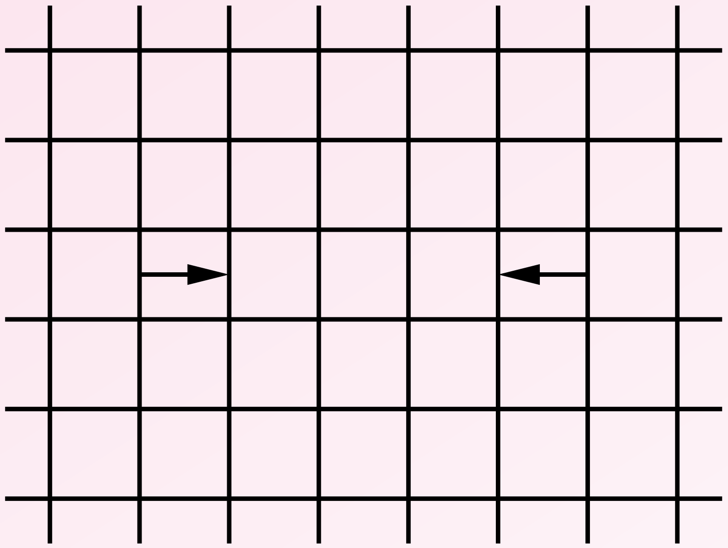
At each time step

- each particle moves to the neighboring cell as indicated by the direction of the particle
- If a cell receives exactly two particles moving in opposite directions then the particles turn 90°.



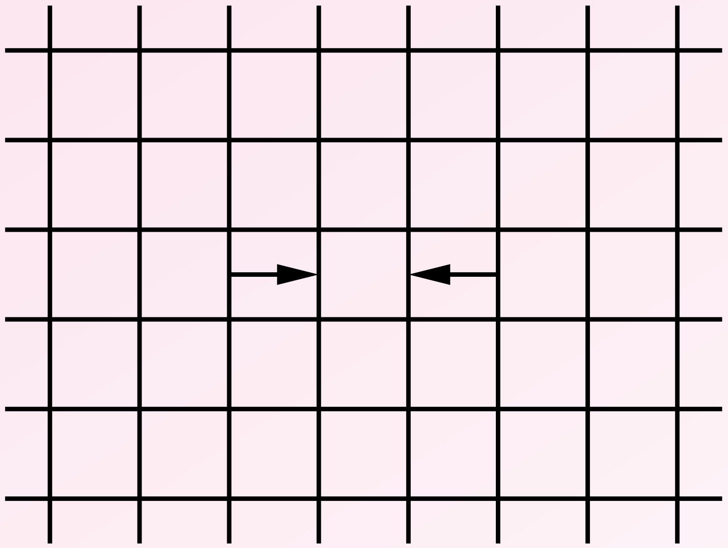
At each time step

- each particle moves to the neighboring cell as indicated by the direction of the particle
- If a cell receives exactly two particles moving in opposite directions then the particles turn 90°.



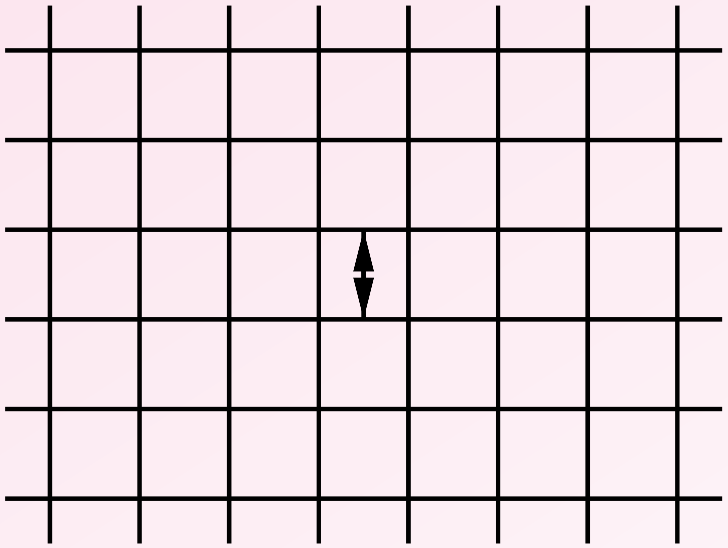
At each time step

- each particle moves to the neighboring cell as indicated by the direction of the particle
- If a cell receives exactly two particles moving in opposite directions then the particles turn 90°.



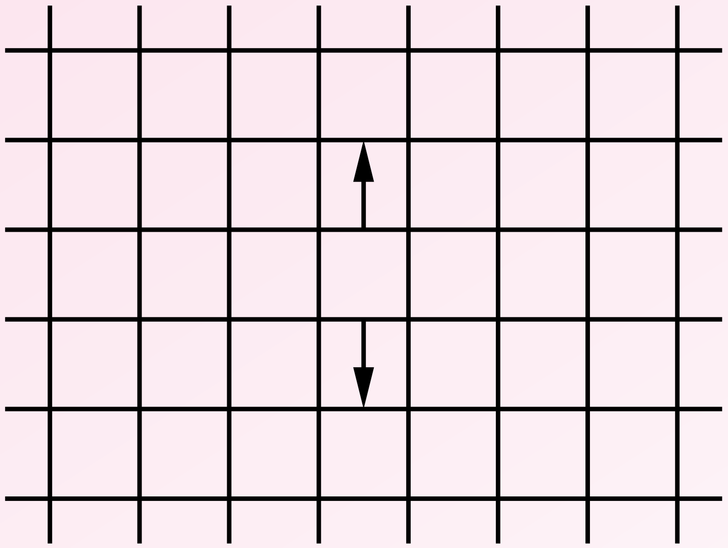
At each time step

- each particle moves to the neighboring cell as indicated by the direction of the particle
- If a cell receives exactly two particles moving in opposite directions then the particles turn 90°.



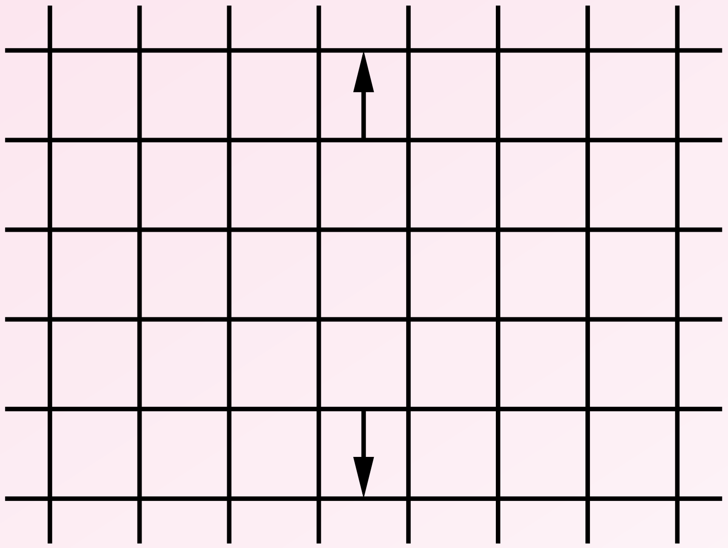
At each time step

- each particle moves to the neighboring cell as indicated by the direction of the particle
- If a cell receives exactly two particles moving in opposite directions then the particles turn 90°.



At each time step

- each particle moves to the neighboring cell as indicated by the direction of the particle
- If a cell receives exactly two particles moving in opposite directions then the particles turn 90°.



The local rule of HPP preserves the total number of particles and their total momentum. These are **conservation laws** that hold in the HPP automaton.

(Remark: HPP is not a realistic lattice gas model as it has several incorrect conservation laws.)

Even more interestingly, the HPP local rule fully preserves information. There is another CA that traces back the configurations in the reverse direction. This **inverse CA** simply moves the particles to the opposite direction, and applies the same collision rule as HPP.

Reversible CA

A CA is called

- **injective** if G is one-to-one,
- **surjective** if G is onto,
- **bijective** if G is both one-to-one and onto.

Reversible CA

A CA is called

- **injective** if G is one-to-one,
- **surjective** if G is onto,
- **bijective** if G is both one-to-one and onto.

A CA G is a **reversible** (RCA) if there is another CA function F that is its inverse, i.e.

$$G \circ F = F \circ G = \text{identity function.}$$

RCA G and F are called the **inverse automata** of each other.

Game-of-Life and **Rule 110** are not reversible:
Configurations may have several pre-images.

HPP lattice gas is reversible.

Theorem (Hedlund et.al 1969) Every bijective cellular automaton is reversible.

Theorem (Hedlund et.al 1969) Every bijective cellular automaton is reversible.

The point of the Theorem is that if G is bijective then each cell can determine its previous state by looking at the current states in some **bounded neighborhood** around it.

Theorem (Hedlund et.al 1969) Every bijective cellular automaton is reversible.

The point of the Theorem is that if G is bijective then each cell can determine its previous state by looking at the current states in some **bounded neighborhood** around it.

Theorem (Kari 1989) It is undecidable if a given two-dimensional cellular automaton is reversible.

Theorem (Hedlund et.al 1969) Every bijective cellular automaton is reversible.

The point of the Theorem is that if G is bijective then each cell can determine its previous state by looking at the current states in some **bounded neighborhood** around it.

Theorem (Kari 1989) It is undecidable if a given two-dimensional cellular automaton is reversible.

This implies that there is **no computable upper bound** on the extend of the neighborhood needed in the inverse time.

Theorem (Toffoli 1977) Any d-dimensional cellular automaton can be simulated by a $(d+1)$-dimensional reversible cellular automaton.

Corollary: Computationally universal two-dimensional reversible cellular automata exist.

Theorem (Toffoli 1977) Any d-dimensional cellular automaton can be simulated by a $(d+1)$-dimensional reversible cellular automaton.

Corollary: Computationally universal two-dimensional reversible cellular automata exist.

Even better:

Theorem (Harao,Morita 1989) Computationally universal reversible **one-dimensional** cellular automata exist.

Garden-Of-Eden and orphans

Configurations that do not have a pre-image are called **Garden-Of-Eden** -configurations. Only non-surjective CA have GOE configurations.

A finite pattern consists of a finite domain $D \subseteq \mathbb{Z}^d$ and an assignment

$$p : D \longrightarrow S$$

of states.

Finite pattern is called an **orphan** for CA G if the pattern does not have a pre-image.

Every configuration containing an orphan is a GOE.

Every configuration containing an orphan is a GOE.

An easy compactness reasoning gives the converse:

Proposition: Every GOE configuration contains an orphan pattern.

Non-surjectivity is hence equivalent to the existence of orphans.

Balance in surjective CA

All surjective CA have **balanced** local rules: for every $a \in S$

$$\left|f^{-1}(a)\right| = |S|^{n-1}.$$

Balance in surjective CA

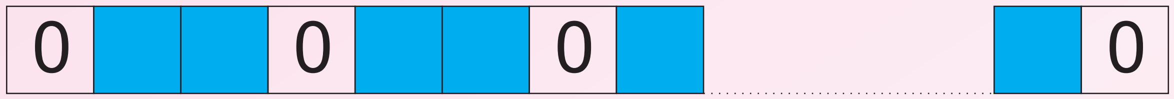
All surjective CA have **balanced** local rules: for every $a \in S$

$$\left|f^{-1}(a)\right| = |S|^{n-1}.$$

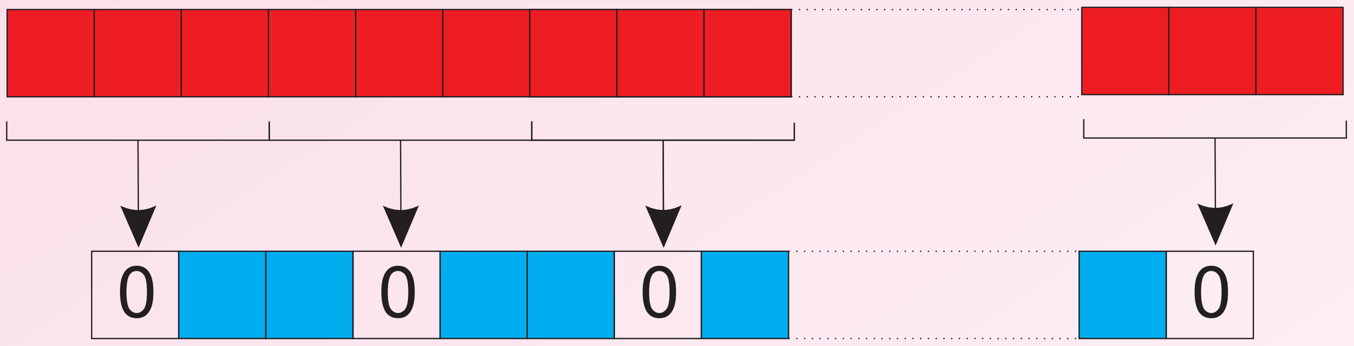
Indeed, consider a non-balanced local rule such as rule 110 where five contexts give new state 1 while only three contexts give state 0:

$$\begin{array}{ccc}
111 & \longrightarrow & 0 \\
110 & \longrightarrow & 1 \\
101 & \longrightarrow & 1 \\
100 & \longrightarrow & 0 \\
011 & \longrightarrow & 1 \\
010 & \longrightarrow & 1 \\
001 & \longrightarrow & 1 \\
000 & \longrightarrow & 0
\end{array}$$

Consider finite patterns where state 0 appears in every third position. There are $2^{2(k-1)} = 4^{k-1}$ such patterns where k is the number of 0's.



Consider finite patterns where state 0 appears in every third position. There are $2^{2(k-1)} = 4^{k-1}$ such patterns where k is the number of 0's.



A pre-image of such a pattern must consist of k segments of length three, each of which is mapped to 0 by the local rule. There are 3^k choices.

As for large values of k we have $3^k < 4^{k-1}$, there are fewer choices for the red cells than for the blue ones. Hence some pattern has no pre-image and it must be an orphan. $\square$

One can also verify directly that pattern

$$01010$$

is an orphan of rule 110. It is the shortest orphan.

Balance of the local rule is not sufficient for surjectivity. For example, the **majority** CA (Wolfram number 232) is a counter example. The local rule

$$f(a, b, c) = 1 \text{ if and only if } a + b + c \geq 2$$

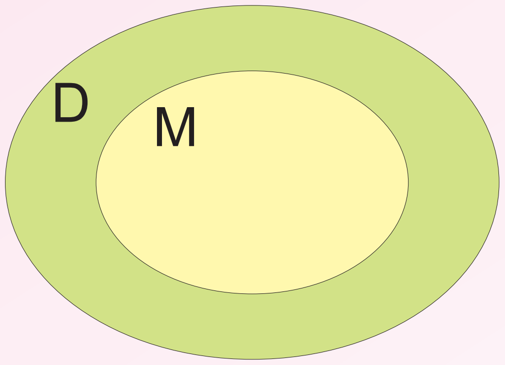
is clearly balanced, but 01001 is an orphan.

The balance property of surjective CA generalizes to finite patterns of arbitrary shape:

Theorem: Let G be surjective. Let $M, D \subseteq \mathbb{Z}^d$ be finite domains such that D contains the neighborhood of M. Then every finite pattern with domain M has the same number

$$n^{|D|-|M|}$$

of pre-images in domain D, where n is the number of states. $\square$



The balance property means that the uniform probability measure is **invariant** for surjective CA. (Uniform randomness is preserved by surjective CA.)

Garden-Of-Eden -theorem

Let us call configurations c_1 and c_2 **asymptotic** if the set

$$\mathit{diff}\,(c_1, c_2) = \{\vec{n} \in \mathbb{Z}^d \mid c_1(\vec{n}) \neq c_2(\vec{n})\ \}$$

of positions where c_1 and c_2 differ is finite.

A CA is called **pre-injective** if any asymptotic $c_1 \neq c_2$ satisfy $G(c_1) \neq G(c_2)$.

The **Garden-Of-Eden -theorem** by Moore (1962) and Myhill (1963) connects surjectivity with pre-injectivity.

Theorem: CA G is surjective if and only if it is pre-injective.

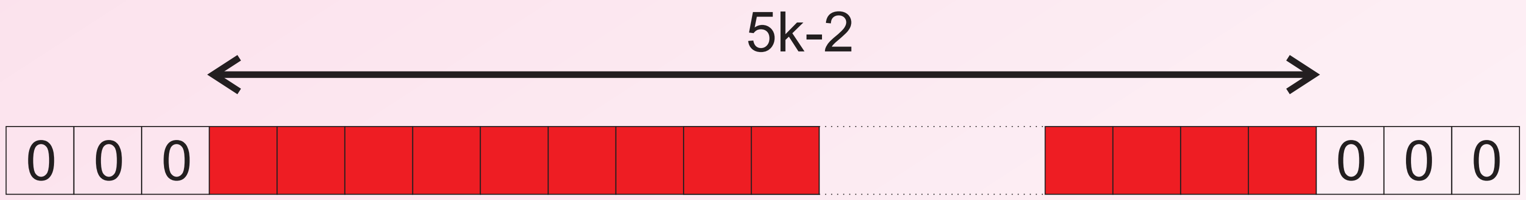
The **Garden-Of-Eden -theorem** by Moore (1962) and Myhill (1963) connects surjectivity with pre-injectivity.

Theorem: CA G is surjective if and only if it is pre-injective.

The proof idea can be easily explained using rule 110 as a running example.

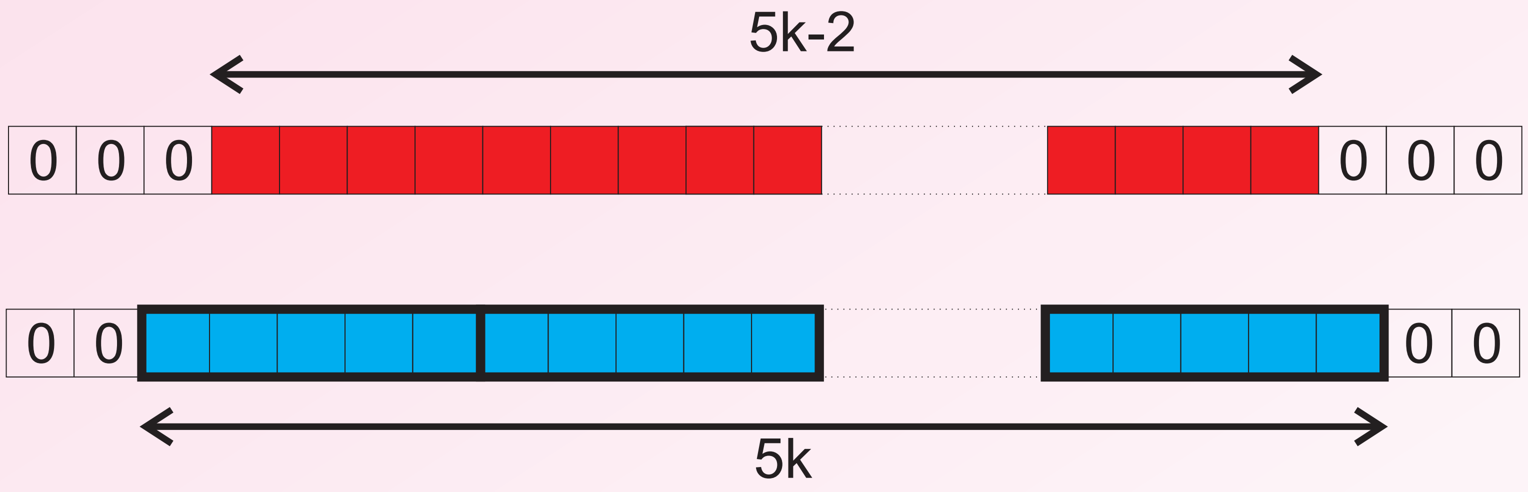
1) G not surjective $\Longrightarrow$ G not pre-injective:

Since rule 110 is not surjective it has an orphan 01010 of length five. Consider a segment of length $5k-2$, for some k, and configurations c that are in state 0 outside this segment. There are $2^{5k-2} = 32^k/4$ such configurations.



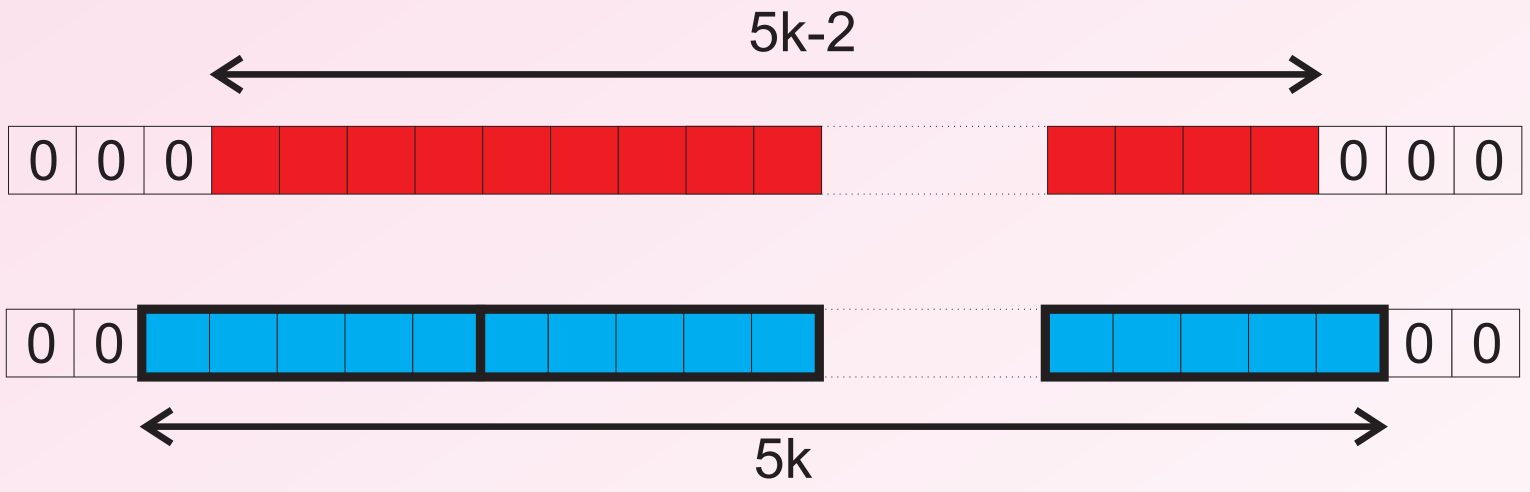
1) G not surjective $\Longrightarrow$ G not pre-injective:

The non-0 part of $G(c)$ is within a segment of length $5k$. Partition this segment into k parts of length 5. Pattern 01010 cannot appear in any part, so only $2^5 - 1 = 31$ different patterns show up in the subsegments. There are at most 31^k possible configurations $G(c)$.



1) G not surjective $\Longrightarrow$ G not pre-injective:

The non-0 part of $G(c)$ is within a segment of length $5k$. Partition this segment into k parts of length 5. Pattern 01010 cannot appear in any part, so only $2^5 - 1 = 31$ different patterns show up in the subsegments. There are at most 31^k possible configurations $G(c)$.



As $32^k/4 > 31^k$ for large k, there are more choices for red than blue segments. So there must exist two different red configurations with the same image. $\square$

2) G not pre-injective $\Longrightarrow$ G not surjective: Can be explained as easily!

Garden-Of-Eden -theorem: CA G is surjective if and only if it is pre-injective.

Garden-Of-Eden -theorem: CA G is surjective if and only if it is pre-injective.

Corollary: Every injective CA is also surjective. Injectivity, bijectivity and reversibility are equivalent concepts.

Proof: If G is injective then it is pre-injective. The claim follows from the Garden-Of-Eden -theorem. $\square$

Examples:

The majority rule is not surjective: finite configurations

$$\ldots 0000000 \ldots \qquad \text{and} \qquad \ldots 0001000 \ldots$$

have the same image, so G is not pre-injective. Pattern

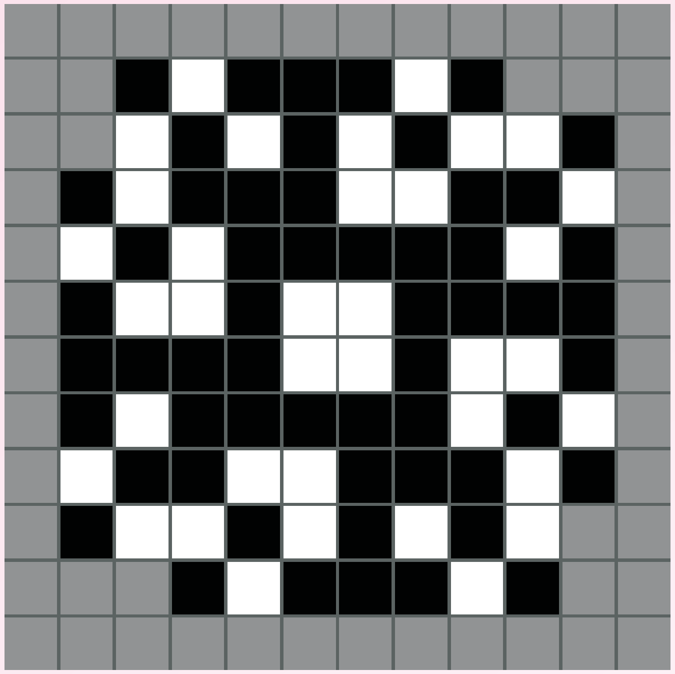
$$01001$$

is an orphan.

Examples:

In Game-Of-Life a lonely living cell dies immediately, so G is not pre-injective. GOL is hence not surjective.

Interestingly, no small orphans are known for Game-Of-Life. Currently, the smallest known orphan consists of 92 cells (56 life, 36 dead):



M. Heule, C. Hartman, K. Kwekkeboom, A. Noels (2011)

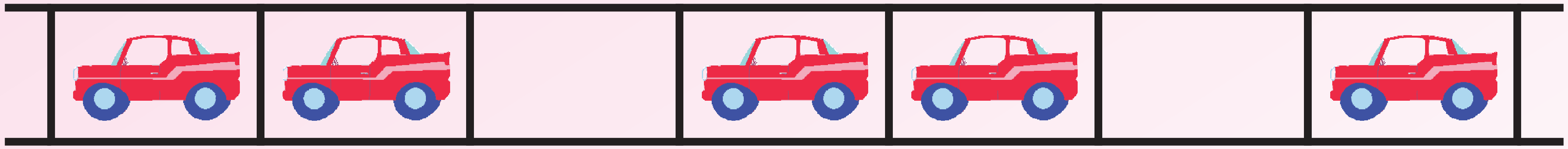
Examples:

The **Traffic CA** is the elementary CA number 226.

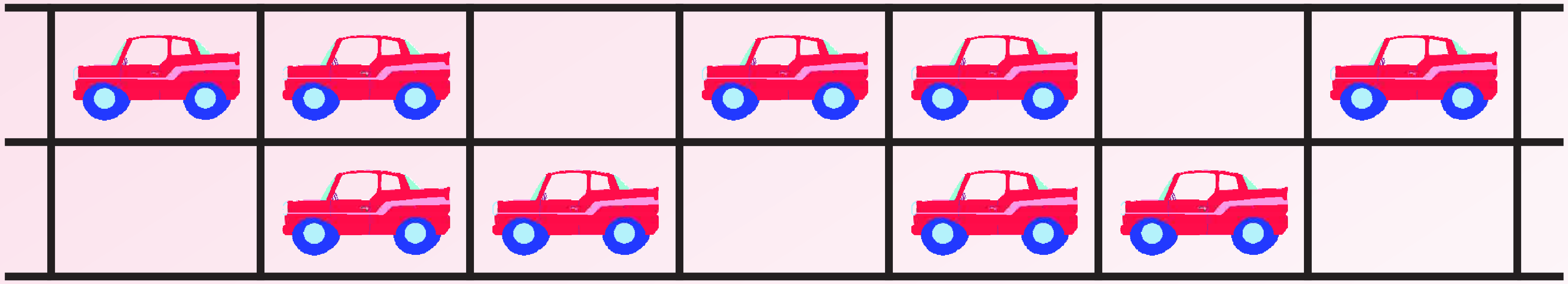
$$\begin{array}{ccc}
111 & \longrightarrow & 1 \\
110 & \longrightarrow & 1 \\
101 & \longrightarrow & 1 \\
100 & \longrightarrow & 0 \\
011 & \longrightarrow & 0 \\
010 & \longrightarrow & 0 \\
001 & \longrightarrow & 1 \\
000 & \longrightarrow & 0
\end{array}$$

The local rule replaces pattern 01 by pattern 10.

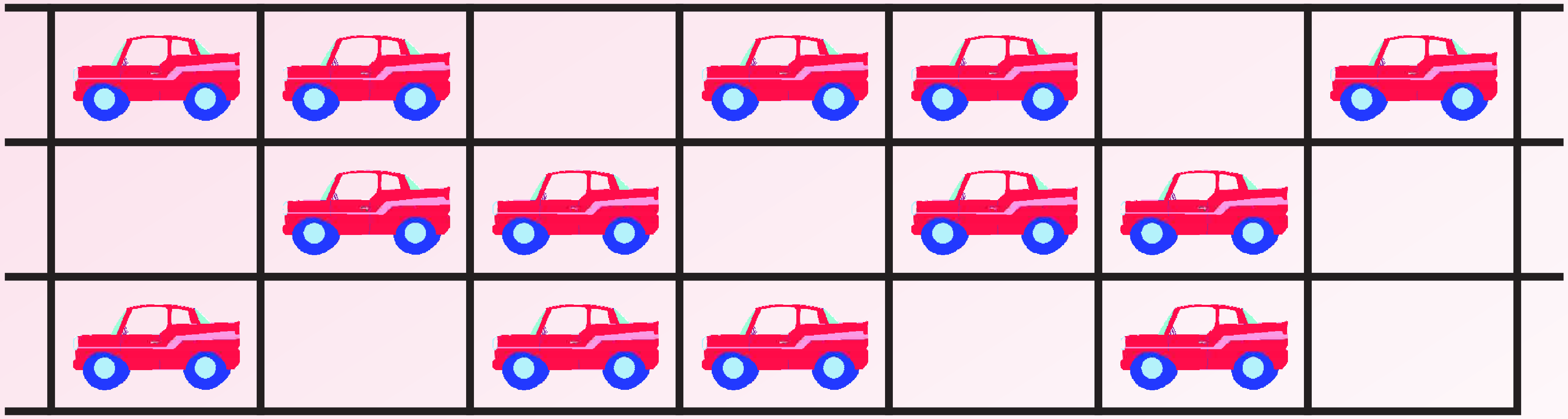
111	$\longrightarrow$	1
110	$\longrightarrow$	1
101	$\longrightarrow$	1
100	$\longrightarrow$	0
011	$\longrightarrow$	0
010	$\longrightarrow$	0
001	$\longrightarrow$	1
000	$\longrightarrow$	0



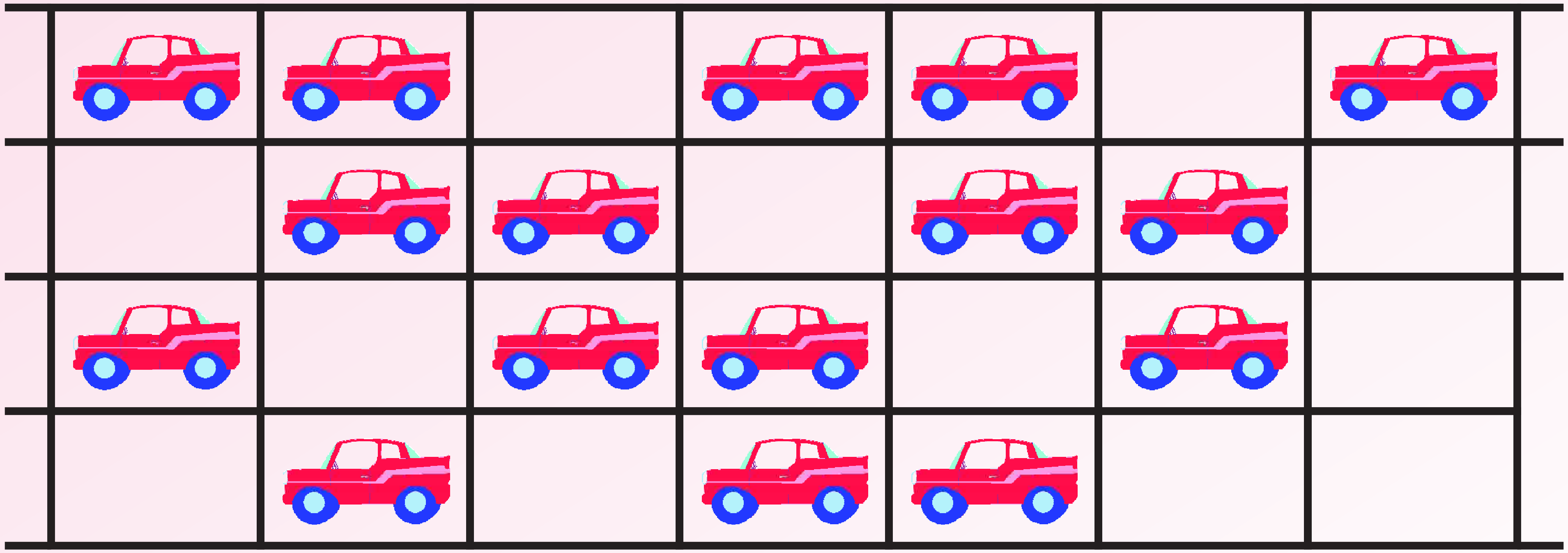
111	$\longrightarrow$	1
110	$\longrightarrow$	1
101	$\longrightarrow$	1
100	$\longrightarrow$	0
011	$\longrightarrow$	0
010	$\longrightarrow$	0
001	$\longrightarrow$	1
000	$\longrightarrow$	0



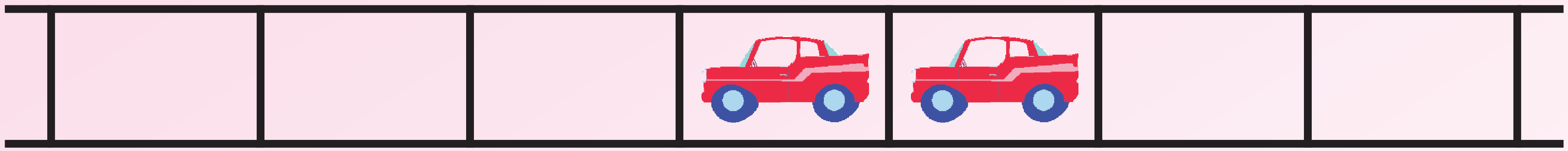
111	$\longrightarrow$	1
110	$\longrightarrow$	1
101	$\longrightarrow$	1
100	$\longrightarrow$	0
011	$\longrightarrow$	0
010	$\longrightarrow$	0
001	$\longrightarrow$	1
000	$\longrightarrow$	0

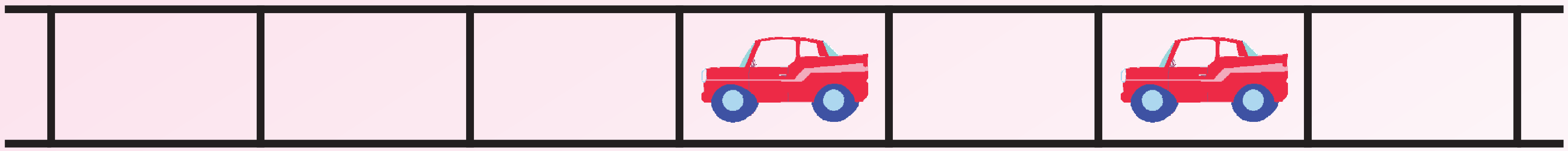


$$
\begin{array}{ccc}
111 & \longrightarrow & 1 \\
110 & \longrightarrow & 1 \\
101 & \longrightarrow & 1 \\
100 & \longrightarrow & 0 \\
011 & \longrightarrow & 0 \\
010 & \longrightarrow & 0 \\
001 & \longrightarrow & 1 \\
000 & \longrightarrow & 0
\end{array}
$$



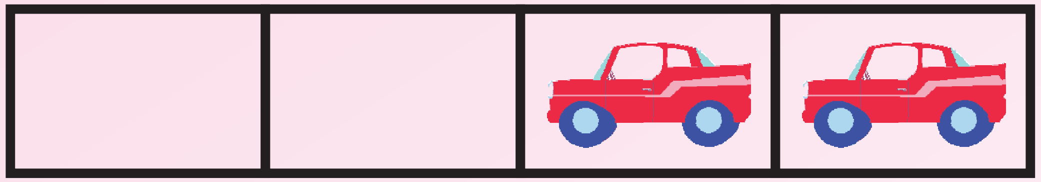
The local rule is balanced. However, there are two finite configurations with the same successor:



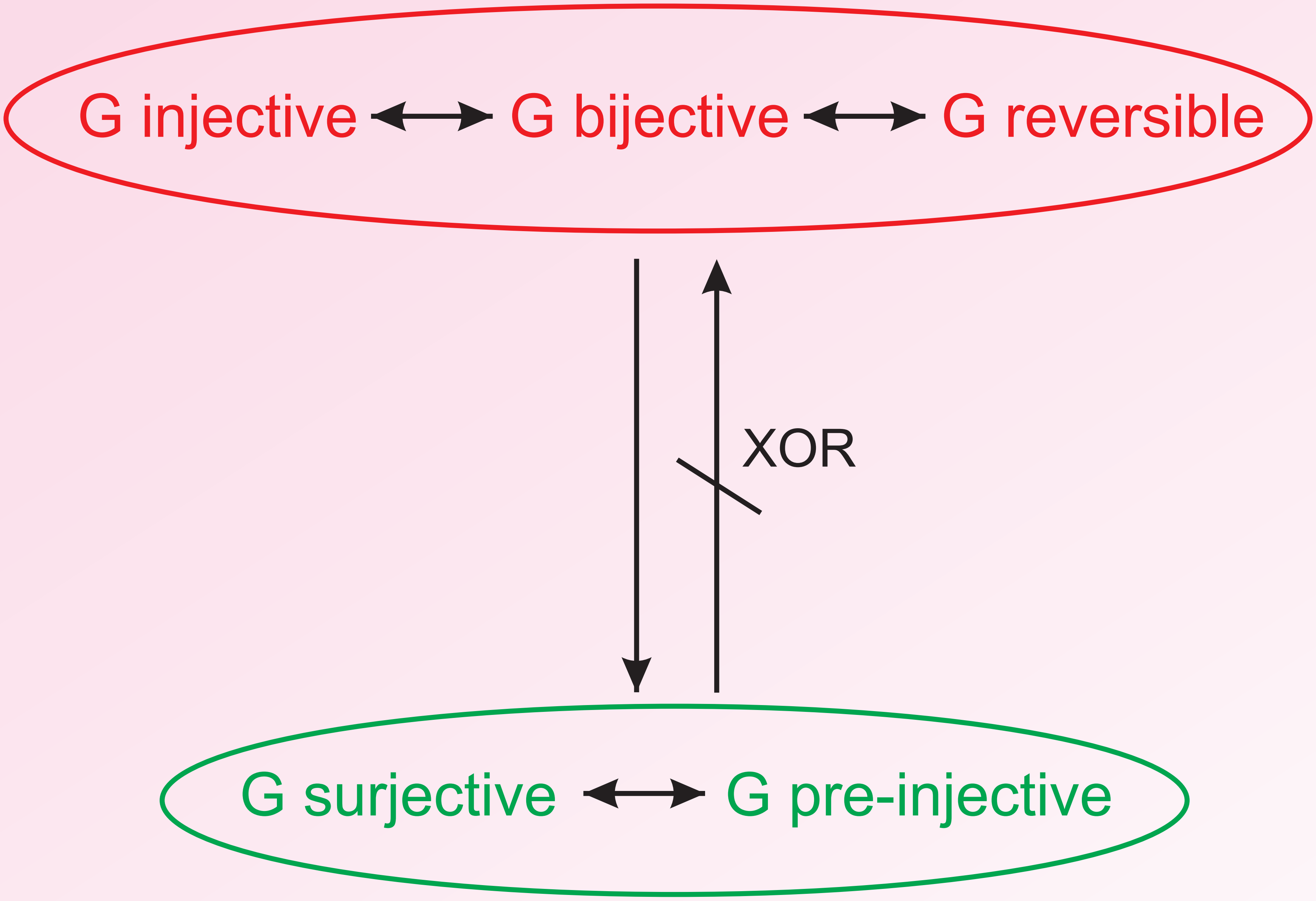


and hence traffic CA is not surjective.

There is an orphan of size four:



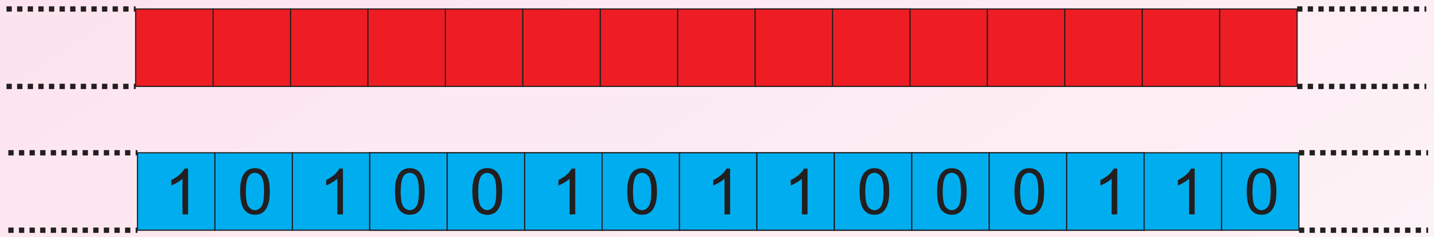
$$\begin{tikzcd}
\text{G injective} \arrow[r, leftrightarrow] & \text{G bijective} \arrow[r, leftrightarrow] \arrow[d] & \text{G reversible} \\
 & \text{G surjective} \arrow[r, leftrightarrow] & \text{G pre-injective}
\end{tikzcd}$$


G injective
G bijective
G reversible
XOR
G surjective
G pre-injective

The xor-CA is the binary state CA with neighborhood $(0, 1)$ and local rule

$$f(a, b) = a + b \pmod 2.$$

In the xor-CA every configuration has exactly two pre-images, so G is surjective but not injective:

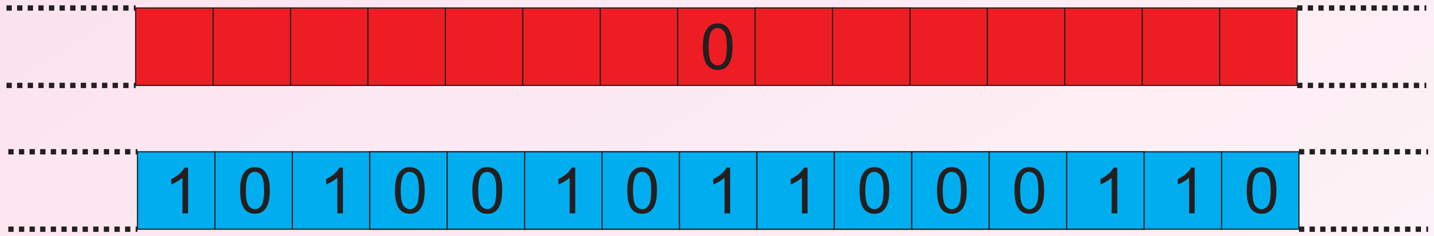


One can freely choose one value in the pre-image, after which all remaining states are uniquely determined by the **left-permutativity** and the **right-permutativity** of xor.

The xor-CA is the binary state CA with neighborhood $(0, 1)$ and local rule

$$f(a, b) = a + b \pmod 2.$$

In the xor-CA every configuration has exactly two pre-images, so G is surjective but not injective:

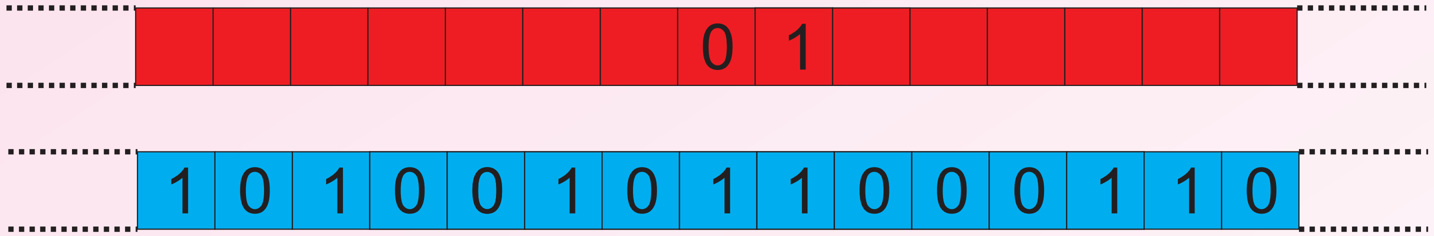


One can freely choose one value in the pre-image, after which all remaining states are uniquely determined by the **left-permutativity** and the **right-permutativity** of xor.

The xor-CA is the binary state CA with neighborhood $(0, 1)$ and local rule

$$f(a, b) = a + b \pmod 2.$$

In the xor-CA every configuration has exactly two pre-images, so G is surjective but not injective:

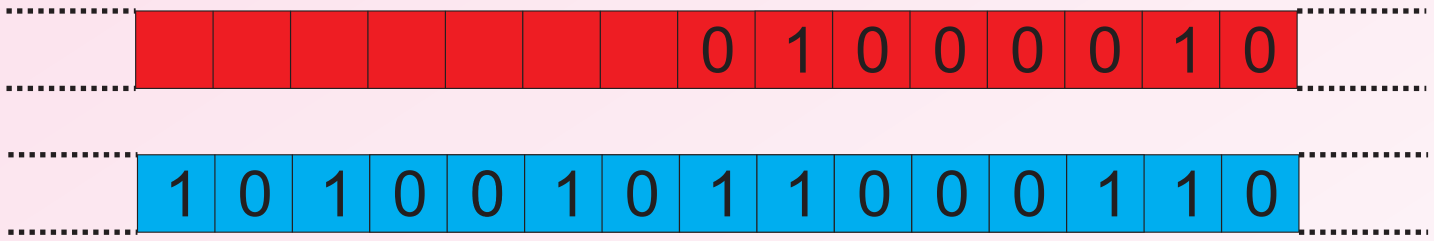


One can freely choose one value in the pre-image, after which all remaining states are uniquely determined by the **left-permutativity** and the **right-permutativity** of xor.

The xor-CA is the binary state CA with neighborhood $(0, 1)$ and local rule

$$f(a, b) = a + b \pmod 2.$$

In the xor-CA every configuration has exactly two pre-images, so G is surjective but not injective:

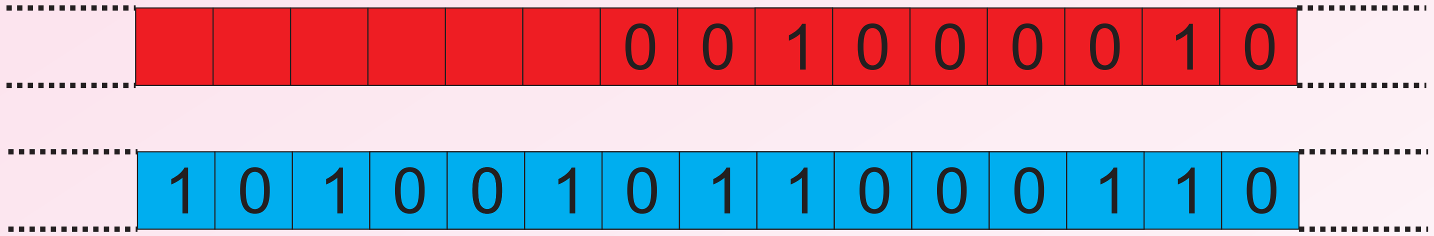


One can freely choose one value in the pre-image, after which all remaining states are uniquely determined by the **left-permutativity** and the **right-permutativity** of xor.

The xor-CA is the binary state CA with neighborhood $(0, 1)$ and local rule

$$f(a, b) = a + b \pmod 2.$$

In the xor-CA every configuration has exactly two pre-images, so G is surjective but not injective:

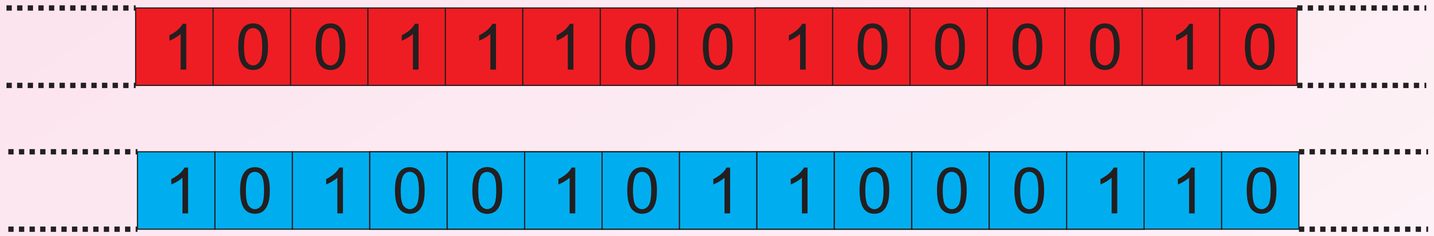


One can freely choose one value in the pre-image, after which all remaining states are uniquely determined by the **left-permutativity** and the **right-permutativity** of xor.

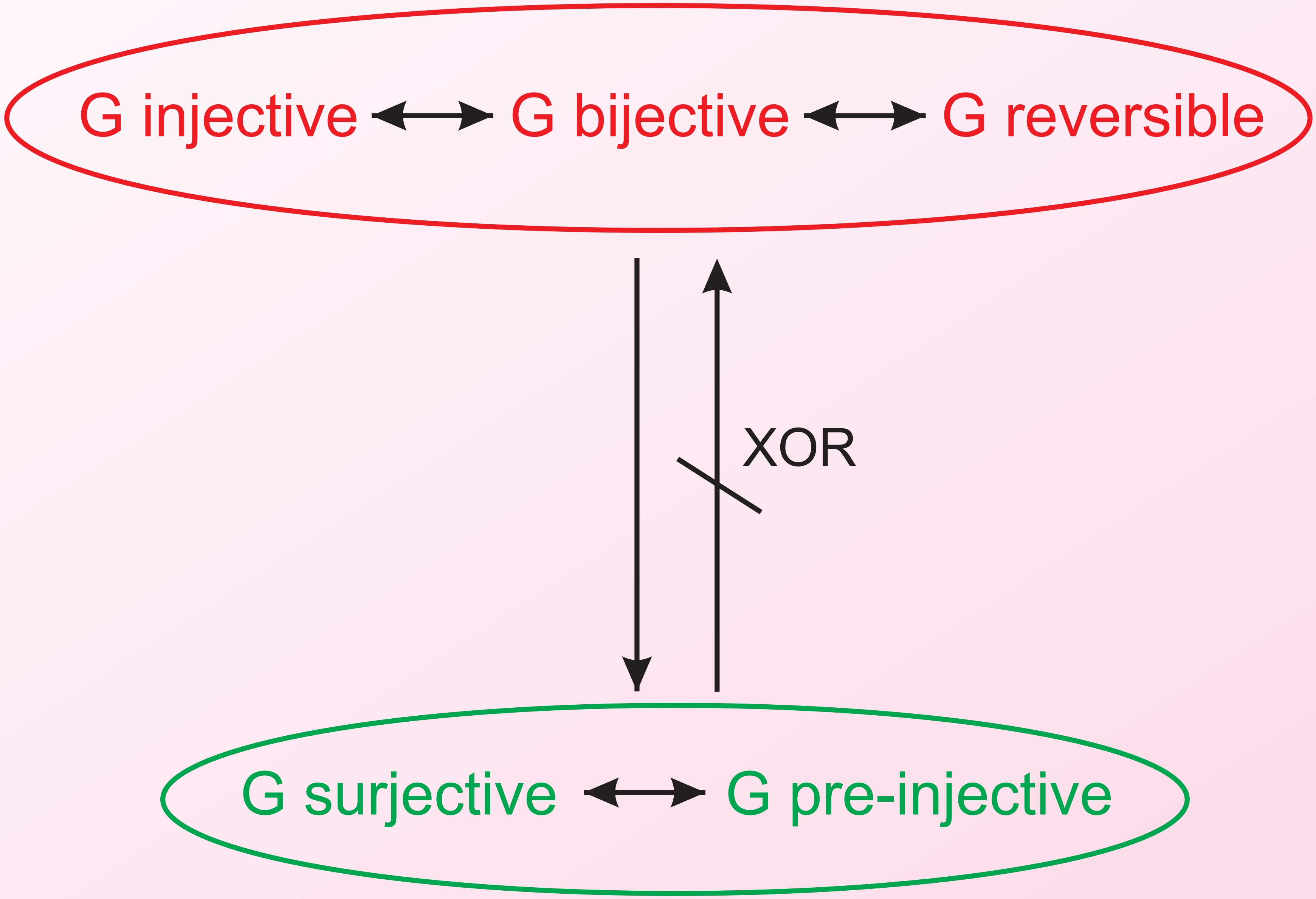
The xor-CA is the binary state CA with neighborhood $(0, 1)$ and local rule

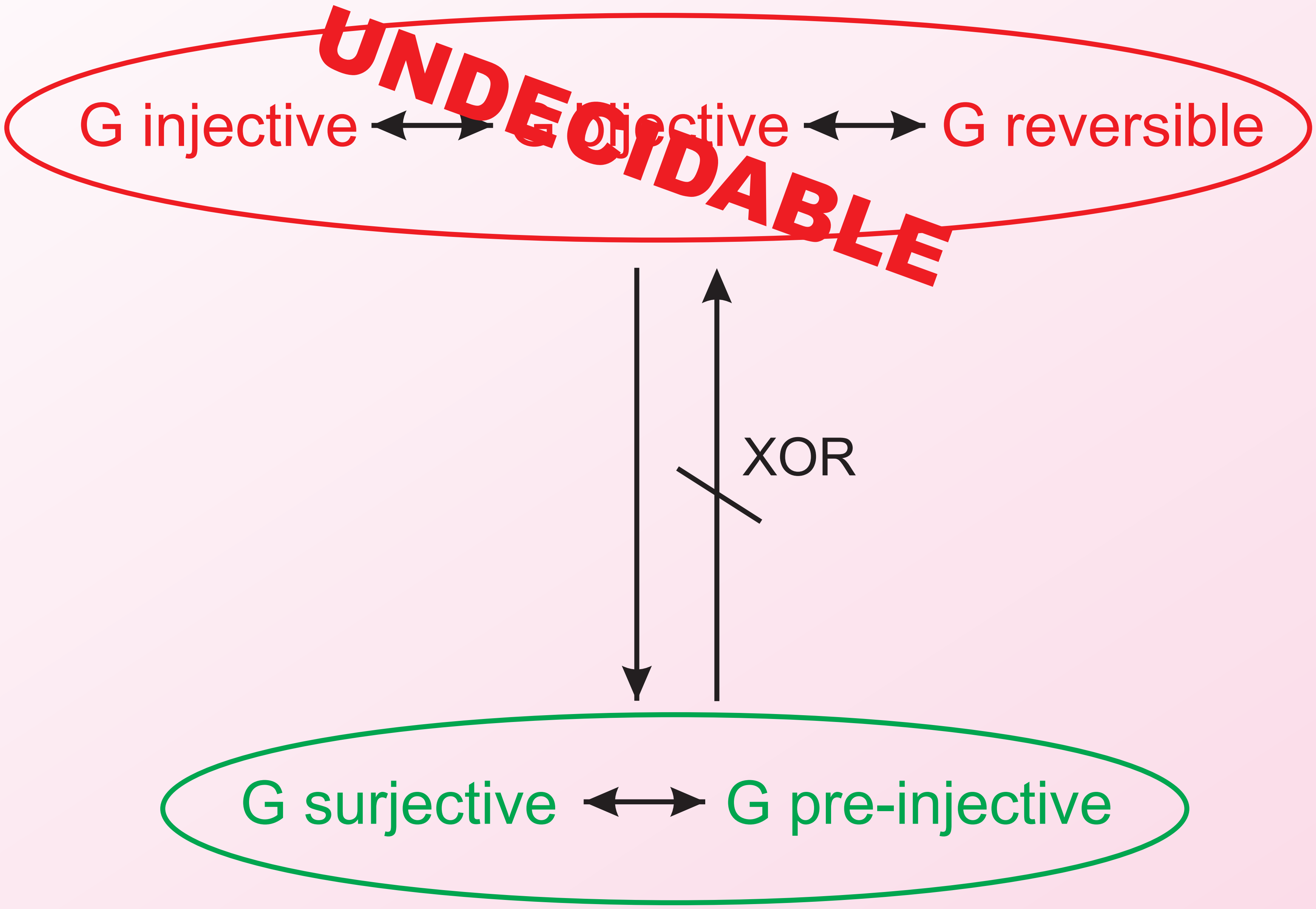
$$f(a, b) = a + b \pmod 2.$$

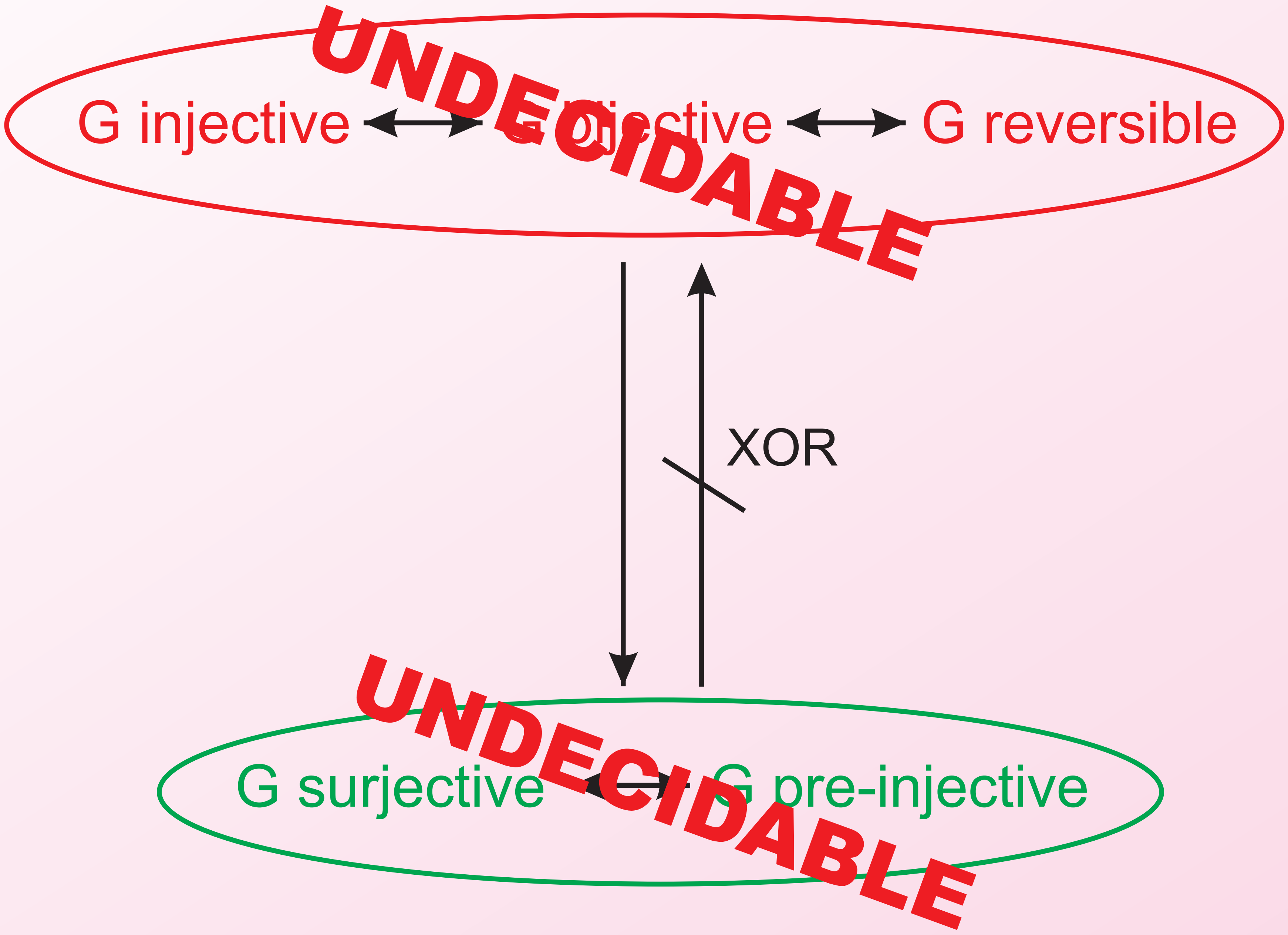
In the xor-CA every configuration has exactly two pre-images, so G is surjective but not injective:



One can freely choose one value in the pre-image, after which all remaining states are uniquely determined by the **left-permutativity** and the **right-permutativity** of xor.


G injective
G bijective
G reversible
XOR
G surjective
G pre-injective


UNDECIDABLE
G injective
G bijective
G reversible
XOR
G surjective
G pre-injective


G injective
G bijective
G reversible
XOR
G surjective
G pre-injective
UNDECIDABLE
UNDECIDABLE

Theorem (Kari 1989) It is undecidable if a given two-dimensional CA is surjective.

Corollary: There is no computable upper bound on the extend of the smallest orphan of non-surjective CA.

(Both reversibility and surjectivity can be tested in polynomial time in dimension 1.)

Conclusion

Cellular Automata have a rich theory developed over half a century. Yet many questions remain open. For example:

- Does there exist a one-dimensional CA with states $\{0, 1\}$ such that every periodic configuration eventually leads to the blinking orbit $\ldots 00000 \ldots \longleftrightarrow \ldots 11111 \ldots$?

Conclusion

Cellular Automata have a rich theory developed over half a century. Yet many questions remain open. For example:

- Does there exist a one-dimensional CA with states $\{0, 1\}$ such that every periodic configuration eventually leads to the blinking orbit $\dots 00000 \dots \longleftrightarrow \dots 11111 \dots$?

Let G be a two-dimensional surjective cellular automaton.

- Does every periodic configuration have a periodic pre-image?
- Does every finite pattern occur in a configuration that is temporally repeating (that is $G^n(c) = c$ for some n) ?

Conclusion

Cellular Automata have a rich theory developed over half a century. Yet many questions remain open. For example:

- Does there exist a one-dimensional CA with states $\{0, 1\}$ such that every periodic configuration eventually leads to the blinking orbit $\ldots 00000 \ldots \longleftrightarrow \ldots 11111 \ldots$?

Let G be a two-dimensional surjective cellular automaton.

- Does every periodic configuration have a periodic pre-image?
- Does every finite pattern occur in a configuration that is temporally repeating (that is $G^n(c) = c$ for some n) ?

Thank You