ADS: Algorithmen und Datenstrukturen 2

Teil 11

Dr. Thomas Efer

Institut für Informatik
Abteilung Automatische Sprachverarbeitung
Universität Leipzig

20. Juni 2018

[Letzte Aktualisierung: 21/06/2018, 10:58]

Nature-Inspired Computation

Themenüberblick:

- Einführung
- Zelluläre Automaten und agentenbasierte Simulationen
  - Elementare Zelluläre Automaten
  - Game of Life
  - Wireworld
  - Langton's Ant
- Metaheuristiken und Optimierung
  - Ameisenalgorithmen
  - Particle Swarm Optimization
  - Genetische Algorithmen *(revisited)*
- Künstliche Neuronale Netze
  - Biologische Grundlagen und notwendige Abstraktionen
  - Perzeptron: Strukturen, Aktivierung und Backpropagation
  - Komplexere Netzwerkarchitekturen

Nature-Inspired Computation: Einführung

Idee: Prozesse aus der Natur untersuchen, verstehen, abstrahieren und auf andere Probleme übertragen:

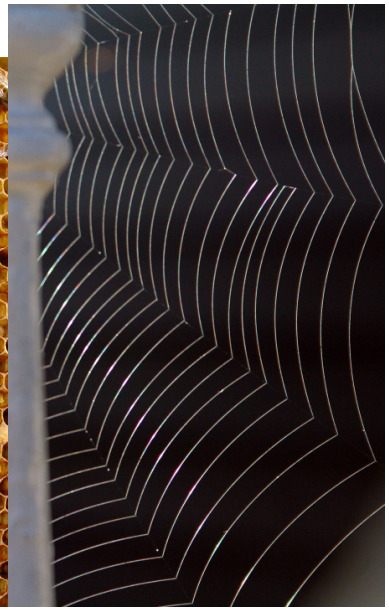
- In der Natur kommen oft hochoptimierte, effiziente Mechanismen zum Einsatz. (Ineffizienz ist schlecht für die Fitness → Evolutionseffekt!)
- Abstraktion und Simplifizierung führen zu formalisierbaren, mathematischen Modellen, die (so die Hoffnung) wünschenswerte Eigenschaften des realen Vorbilds besitzen.
- Diese Eigenschaften der Prozesse lassen sich evtl. für die (annähernde) Lösung schwerer Probleme nutzen. Statt perfekter Lösungen werden in der Natur meist nur „ausreichend gute“ Lösungen gefunden, diese sind jedoch meist recht ressourcenschonend.
- Oft existieren in der Natur mehrere Problemlösungsstrategien parallel. Einige sind zudem sehr flexibel – können sich also wechselnden Erfordernissen anpassen. Auch aus dieser Beobachtung können Erkenntnisse für generelle Strategien gezogen werden.

Nature-Inspired Computation: Einführung

Die Natur als Vorbild – keine exklusive Idee der Informatik:







„Biomimikry“ in Materialkunde, Ingeneurwesen, Architektur, etc.

Nature-Inspired Computation: Einführung

- Die Informatik kann im Bezug auf (lokale) Entscheidungsfindung, Abarbeitungsreihenfolgen großer Probleme, numerische Optimierungsprobleme und kollaborative Problemlösungsansätze viel von vorgängen in der Natur lernen.
- Es werden dazu jedoch (möglichst einfache) formale Beschreibungsmechanismen und theoretische Grundgerüste und implementierbare Modelle für ein Verständnis von (natürlichen und künstlichen) Prozessen benötig. Für einen einfachen Einstieg in die Thematik kohnt sich ein blick auf Zelluläre Automaten.

Zelluläre Automaten

- spezielle Form deterministischer zeitdiskreter dynamischer Systeme
- identische „Zellen“, die sich in bestimmten Zuständen befinden und diese gleichzeitig in Abhängigkeit von den aktuellen Zuständen der Zellen ihrer Nachbarschaft ändern
- Buch „*A new kind of Science*“ (2002)
  `http://www.wolframscience.com/nks/`
  von Stephen Wolfram (Entwickler von Mathematica)
- Möglichkeit zur Untersuchung komplexer „universeller“ Phänomene, die in simplen Systemen mit einfachen Regeln entstehen können

Formale Beschreibung

- Zeit: $t \in \mathbb{N}$
- Positionsmenge: P
- Nachbarschaft: $N : P \rightarrow \langle p : p \in P \rangle$
- Zustandsmenge: S
- Übergangsfunktion: $f_N : \langle s : s \in S \rangle \rightarrow S$
- Zustandszuweisung: $v : P \rightarrow S$
- Startkonfiguration: v_0
- Zeitabhängigkeit: $v_t(pos) = f_N(\langle v_{t-1}(n) : n \in N(pos) \rangle)$

Elementare Zelluläre Automaten

Eindimensionale binäre Zelluläre Automaten mit direkter 1–Nachbarschaft

- Zeit: $t \in \mathbb{N}$
- Positionsmenge: $P = \mathbb{Z}$
- Nachbarschaft: $N : P \rightarrow \langle p : p \in P \rangle = \langle pos - 1, pos, pos + 1 \rangle$
- Zustandsmenge: $S = \{\square, \blacksquare\}$
- Übergangsfunktion: $f_N : \langle s : s \in S \rangle \rightarrow S$ legen wir gleich fest!
- Zustandszuweisung: $v : P \rightarrow S$
- Startkonfiguration: $v_0 = \{\blacksquare$ wenn $pos = 0,$ sonst $\square$
- Zeitabhängigkeit: $v_t(pos) = f_N(\langle v_{t-1}(n) : n \in N(pos) \rangle)$

Elementare Zelluläre Automaten

- Die Übergangsfunktion f_N bildet eine (positionstreue) Liste von Zuständen auf einen neuen Zustand ab.
- Die Zeitabhängigkeit bestimmt, dass die zukünftige Zustandszuweisung für eine Position nur von den aktuell zugewiesenen Zuständen in der Umgebung der Position abhängt.
- Die Umgebung besteht hier aus dem linken Nachbarn, der Position selbst und dem rechten Nachbarn.
- Die möglichen Zustandsfolgen sind:

■■■, ■■□, ■□■, ■□□, □■■, □■□, □□■ *und* □□□

Elementare Zelluläre Automaten

Die Übergangsfunktion (auch Regel, *Rule* genannt) legt nun für jede mögliche Eingabe einen Ausgabewert für die Position fest (in nachfolgender Zeile notiert)

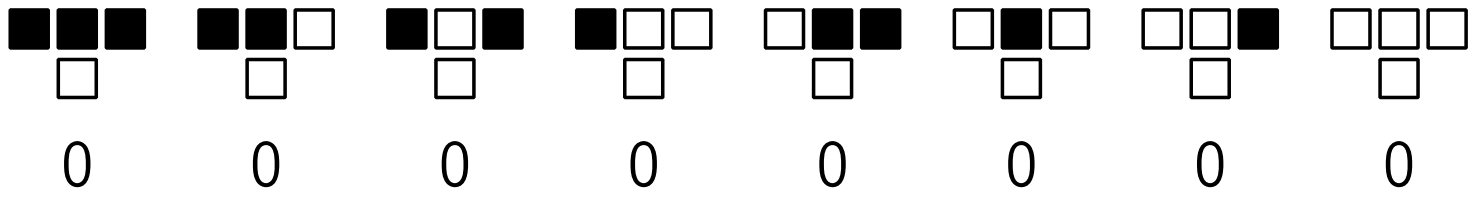
Eingabe: ■■■ ■■□ ■□■ ■□□ □■■ □■□ □□■ □□□
Ausgabe: □ □ □ □ ■ ■ ■ ■

Es gibt hierbei eine endliche Zahl möglicher Regeln.

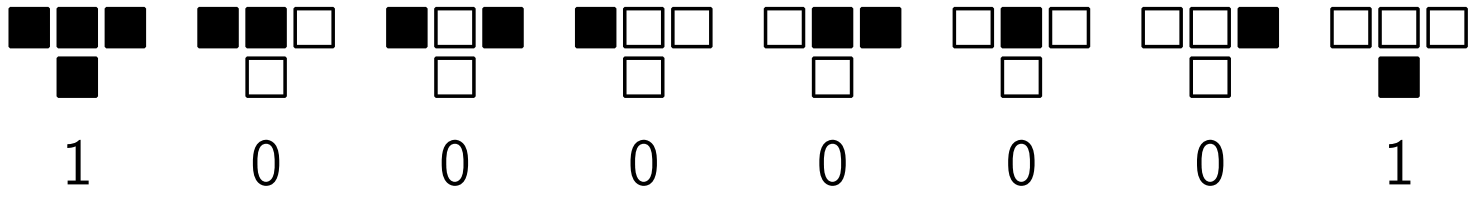
Wie viele?

Elementare Zelluläre Automaten

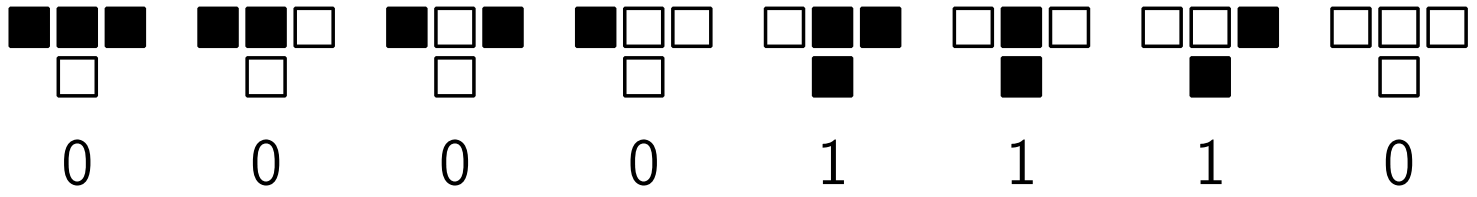
Die Folge der Ausgabezustände kann als Binärzahl aufgefasst werden:



= 0



= 129



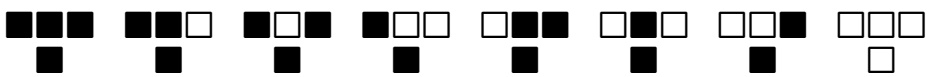
= 14

Elementare Zelluläre Automaten

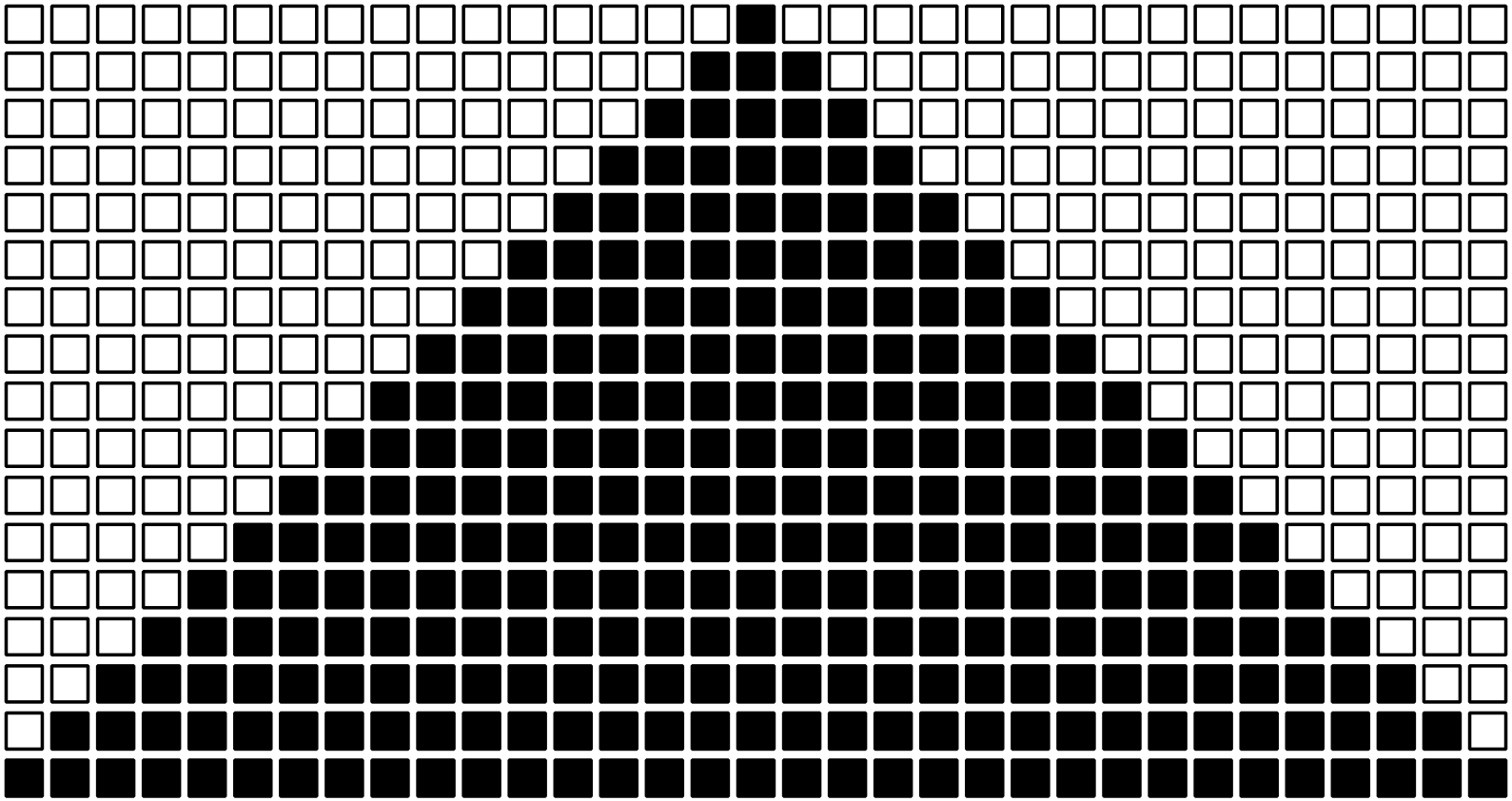
Um den in diskreten Zeitschritten ablaufenden Prozess untersuchen zu können, kann eine Visualisierung gewählt werden, bei der die vertikale Achse genutzt wird, um die Zustandszuweisung im t ten Schritt in der Zeile unter dem $t-1$ten zu notieren.

Dadurch entsteht ein zweidimensionales Bild, das Aufschlüsse über die Dynamik der Zustandszuweisungen gibt. Es muss dabei jedoch bedacht werden, dass sich die Positionen der Zellen nur eindimensional in den Zeilen befinden und dass die lokalen Zustandszuweisungen nur die neuen Zuständen „nach unten“ beeinflussen.

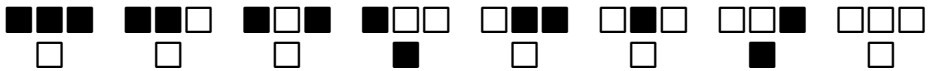
Elementare Zelluläre Automaten



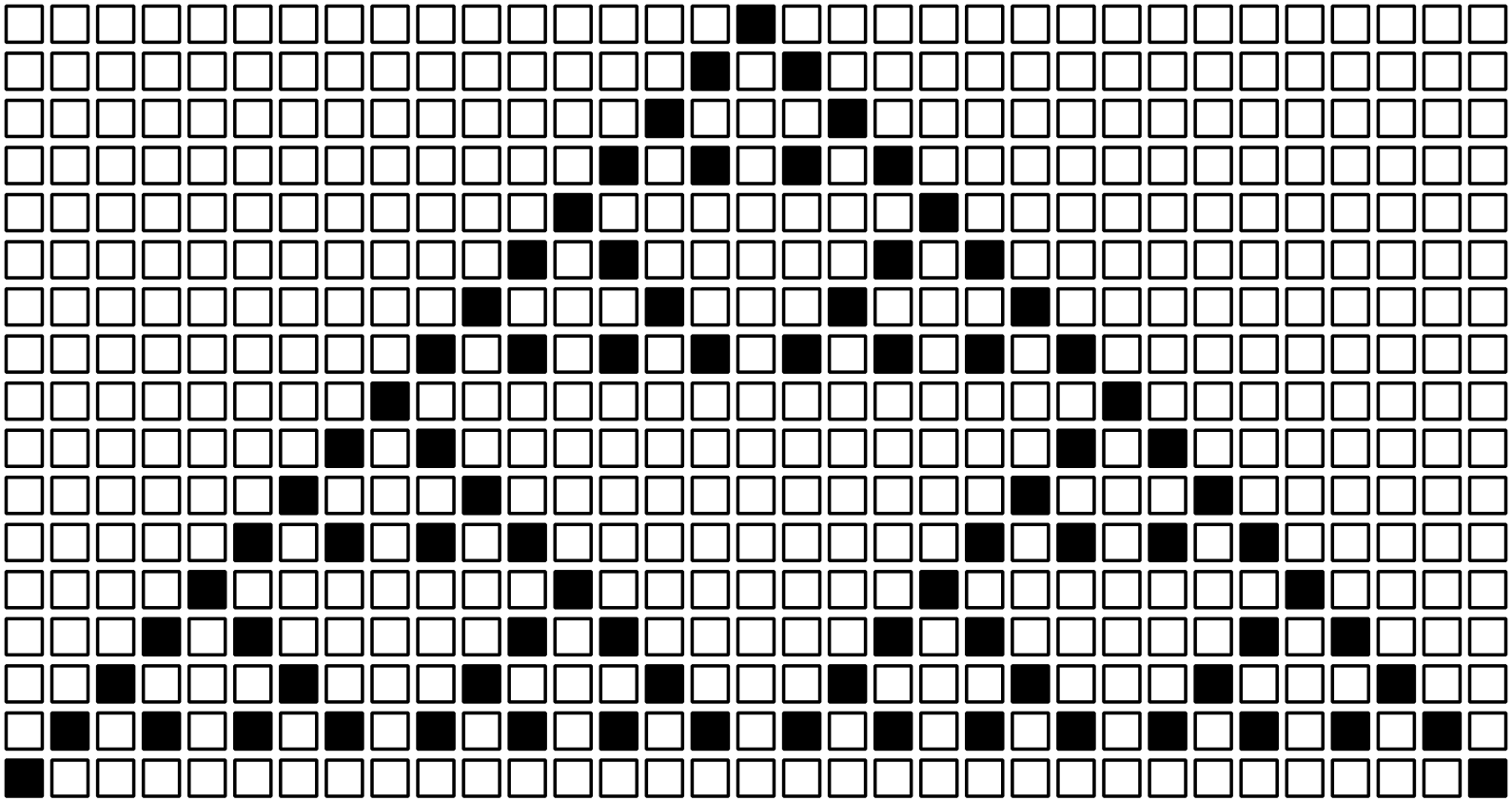
Rule 254



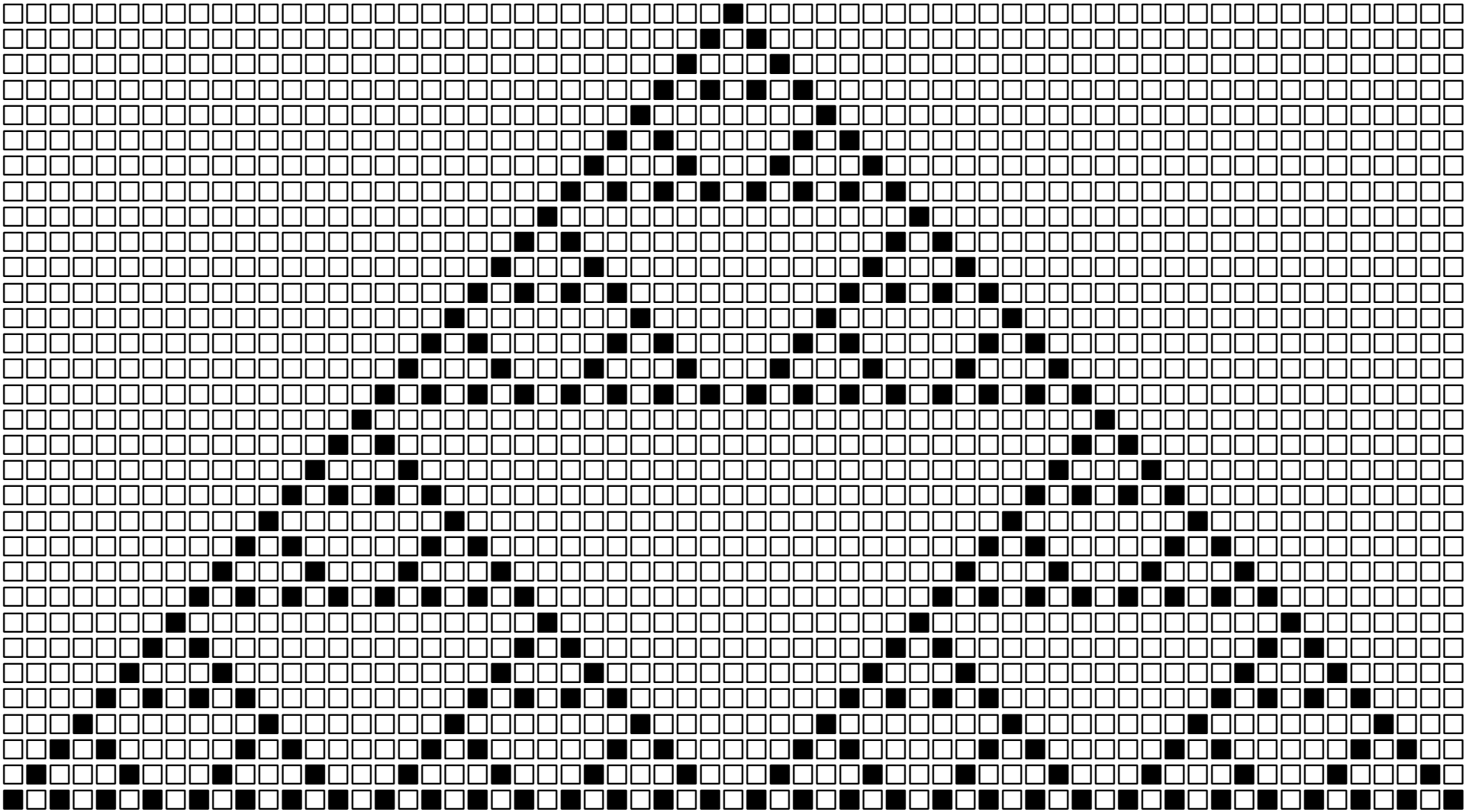
Elementare Zelluläre Automaten



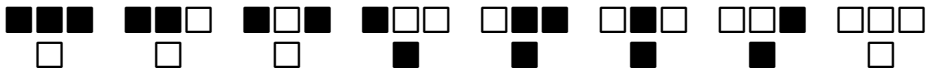
Rule 18



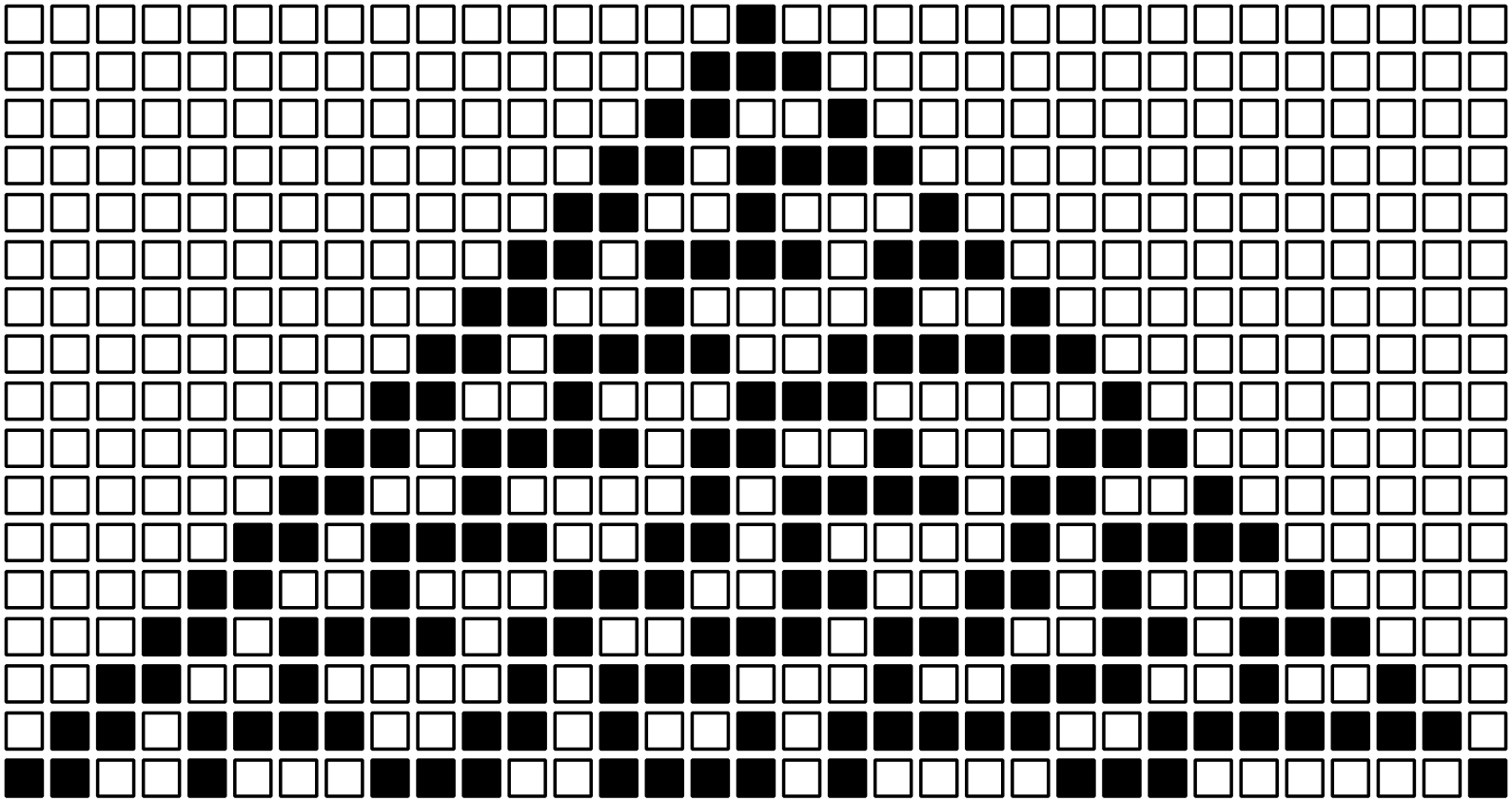
Elementare Zelluläre Automaten - Rule 18



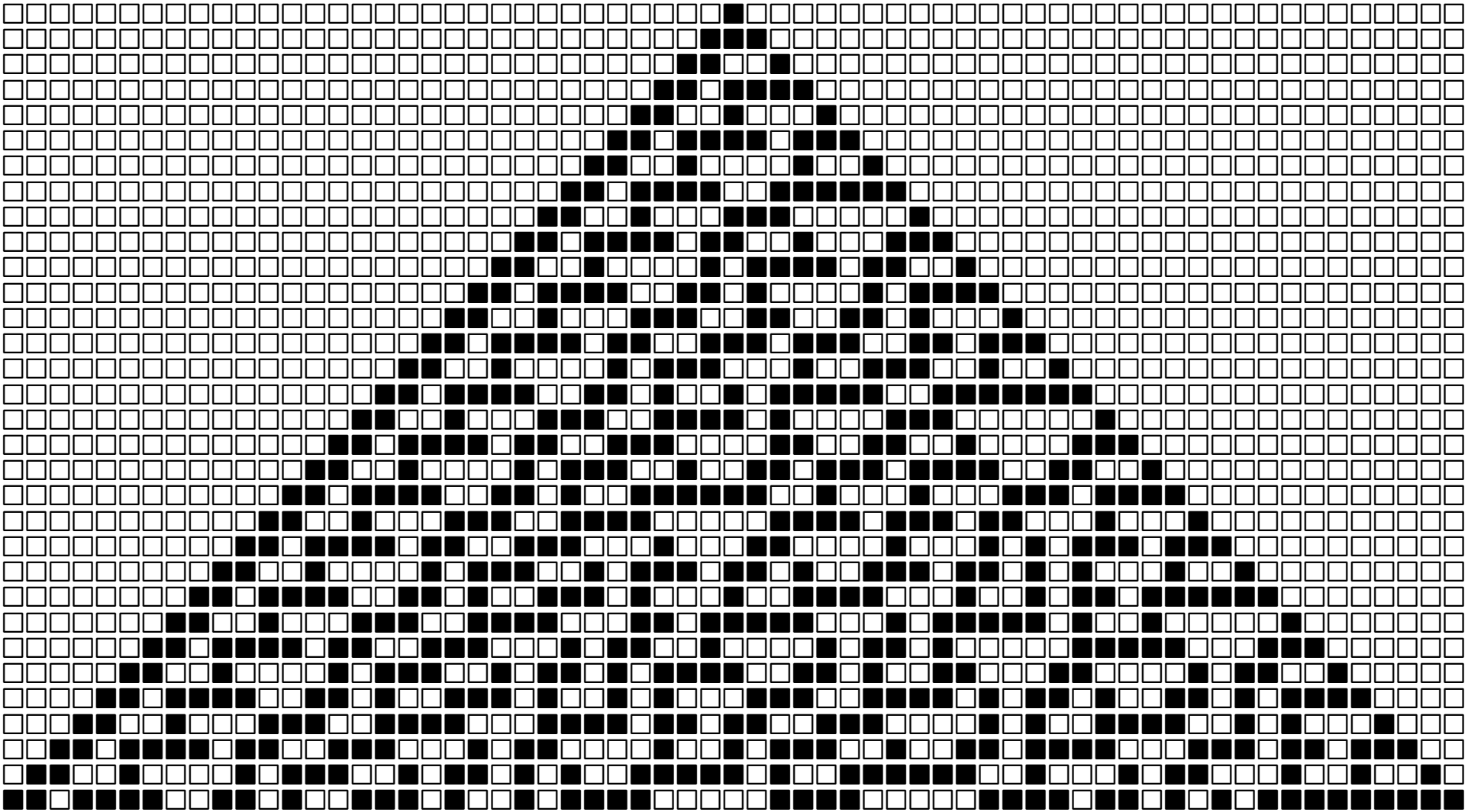
Elementare Zelluläre Automaten



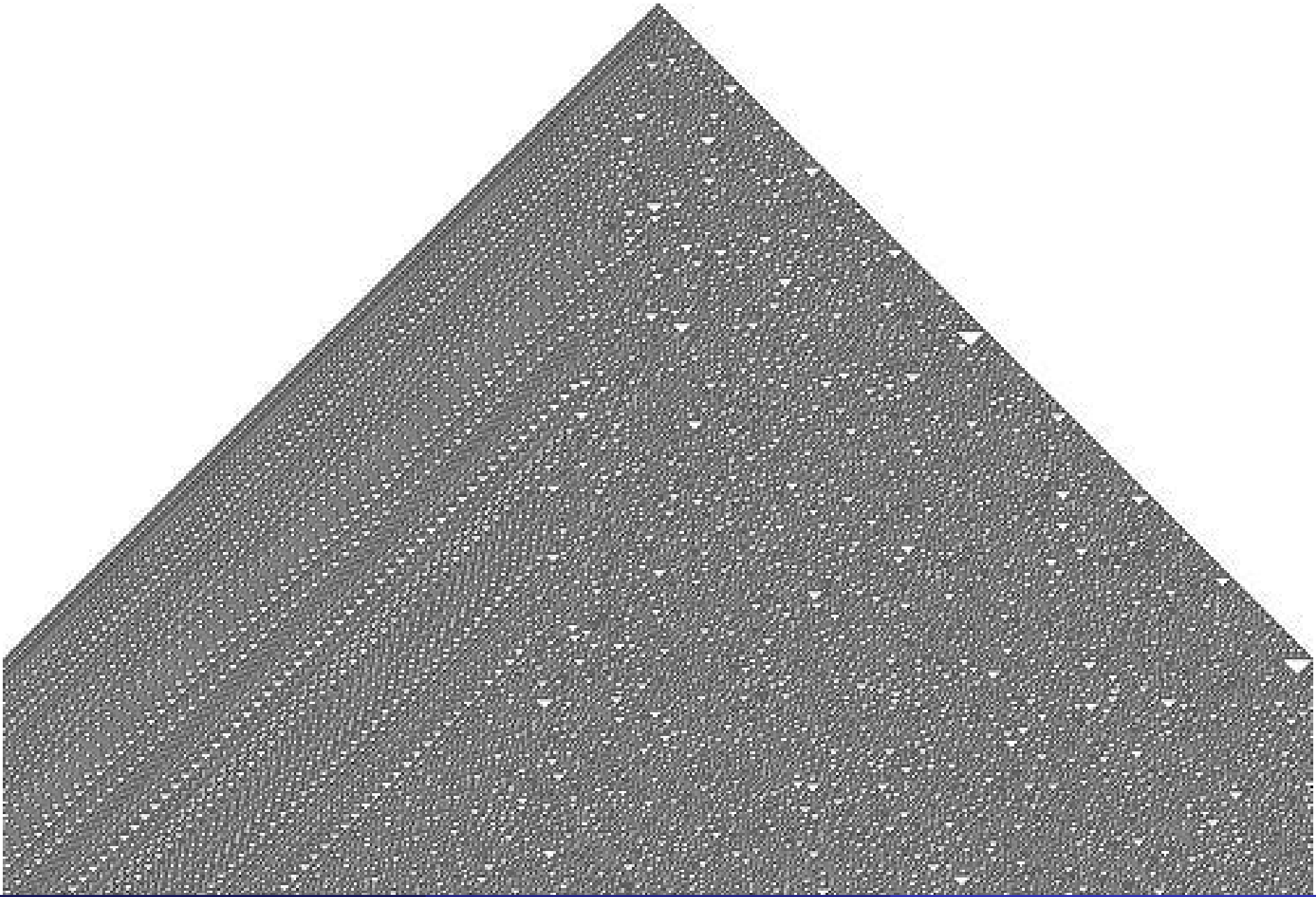
Rule 30



Elementare Zelluläre Automaten - Rule 30



Elementare Zelluläre Automaten - Rule 30

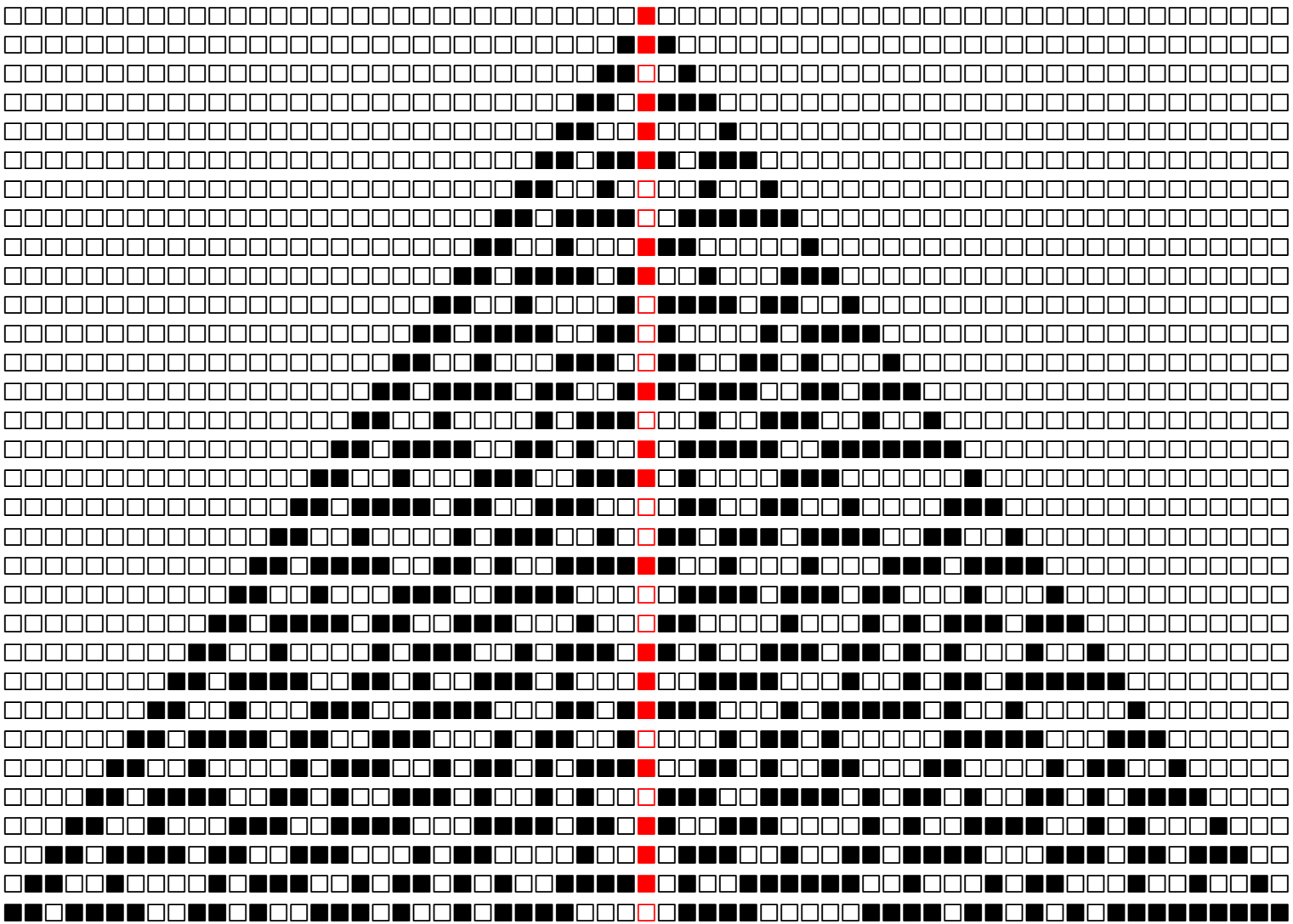


Elementare Zelluläre Automaten - Rule 30

- Aus der Natur: Weberkegelschnecke *(conus textile)*

- ⚠ *für Conotoxine gibt es bislang keine Gegenmittel!*

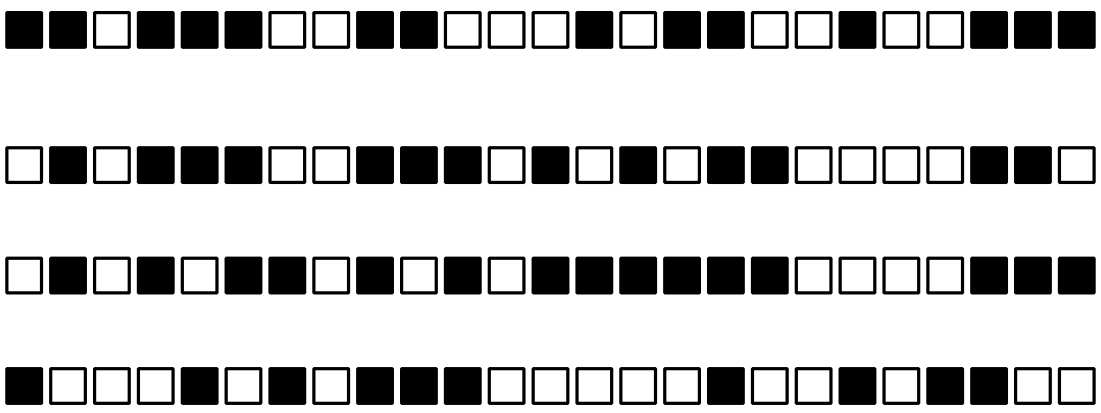
Elementare Zelluläre Automaten - Rule 30



Sequenz: 

Elementare Zelluläre Automaten - Rule 30

Die ersten 100 Zustände:

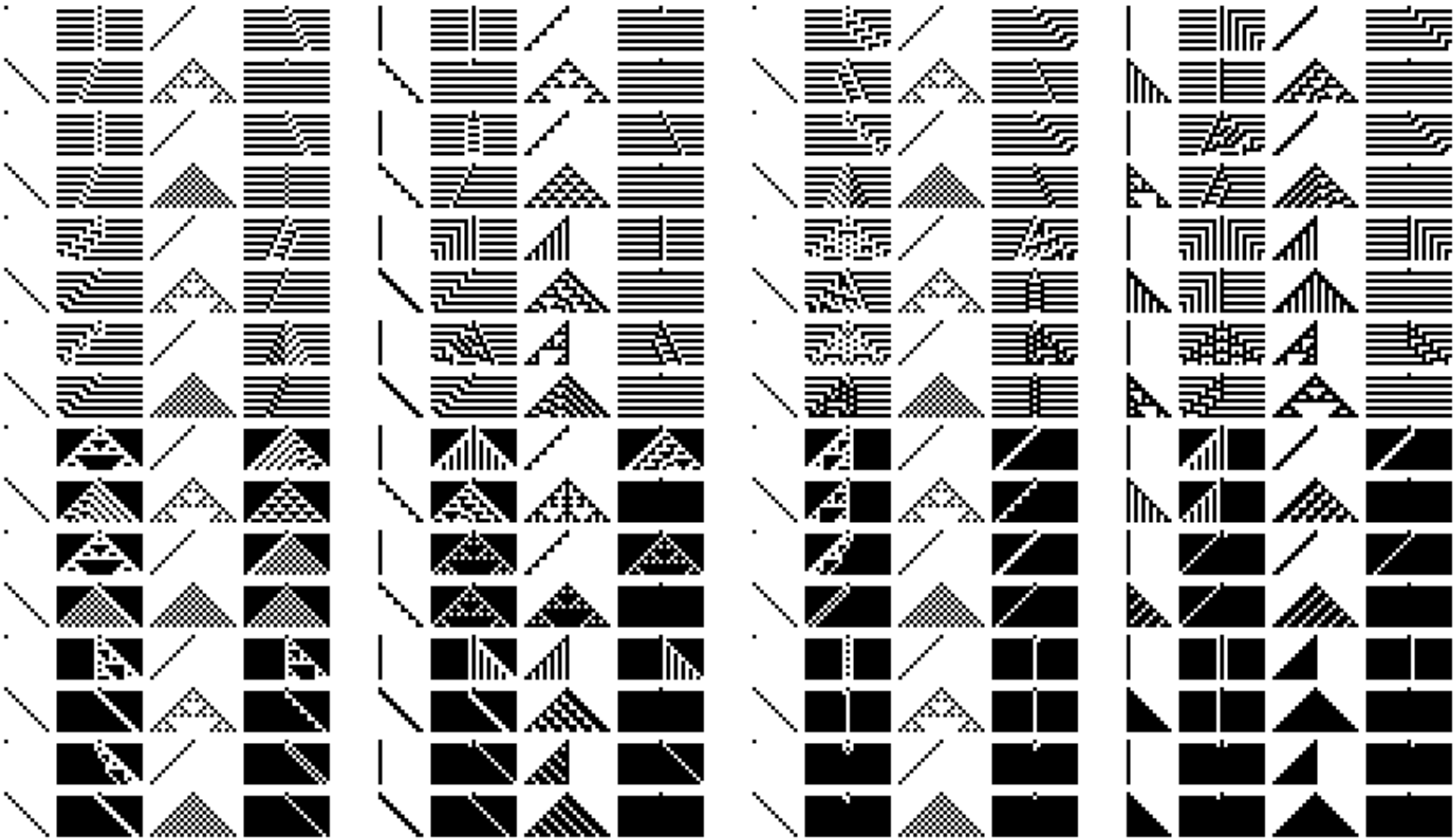


- 52x■, 48x□
- 27x■□, 26x□■, 24x■■, 21x□□
- Runs:
  - 13x■, 7x■■, 5x■■■, 1x■■■■, 1x■■■■■■
  - 15x□, 7x□□, 2x□□□, 2x□□□□, 1x□□□□□

Ist Rule 30 ein guter Zufallszahlengenerator?

Elementare Zelluläre Automaten

Durch auseinander ableitbare Varianten (Spiegelung, Inversion, alternierende Zeilen) bleiben nur 88 „einzigartige“ Regeln übrig.



Elementare Zelluläre Automaten

Klasssifizierung von Regeln nach aufsteigender „Komplexität“ nach dem Verhalten für (fast) jede Anfangskonfiguration nach Wolfram:

- Klasse 1: konvergiert schnell in eine stabile gleichförmige Zustandszuweisung für alle Zellen
- Klasse 2: konvergiert schnell in eine stabile oder oszillierende Zustandszuweisung für alle Zellen, wobei einige lokal begrenzte Stellen noch den Einfluss der Anfangskonfiguration aufweisen
- Klasse 3: entwickelt sich zu einer pseudo-zufälligen oder chaotischen Zustandszuweisung ohne bleibende Strukturen.
- Klasse 4: entwickelt sich zu komplexen Mustern, in denen lokale Strukturen zeitweise oder permanent stabil bleiben und sich neu bilden können. Möglicherweise konvergieren die Muster nach vielen Interationen gegen das Verhalten von Regeln einer niedrigeren Klasse.

Elementare Zelluläre Automaten

Noch mehr Spaß im Selbststudium!

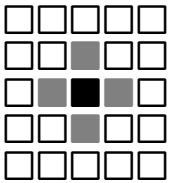
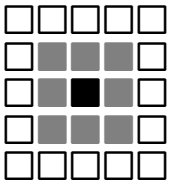

Welche Muster erzeugt Regel 90? Wo kommen diese noch vor?


Regel 110 ist Turing-vollständig. Was bedeutet das?


Kann Regel 184 helfen, Staus vorherzusagen?

Mehrdimensionale Zelluläre Automaten

Die Definition der Nachbarschaft kann auf verschiedene Arten erfolgen:

-  Von-Neumann-Nachbarschaft
-  Moore-Nachbarschaft

Grau: in Nachbarschaft; Schwarz: Ausgangsposition, in Nachbarschaft enthalten

Exkurs: Interessante Erklärung zum programmatischen Umgang mit hexagonalen Gittern auf
`https://www.redblobgames.com/grids/hexagons/`

Conway's Game of Life

Zweidimensionaler binärer Zellulärer Automat mit festen Regeln auf Moore-Nachbarschaft, nach John Horton Conway (1970)

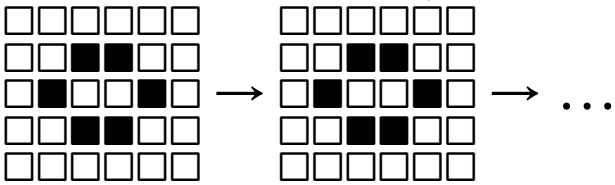
Regeln:

- aus □ mit genau 3 angrenzenden Nachbarn, die ■ sind, wird im nächsten Schritt ■
- aus ■ mit 5 oder 6 angrenzenden Nachbarn, die □ sind, wird im nächsten Schritt □

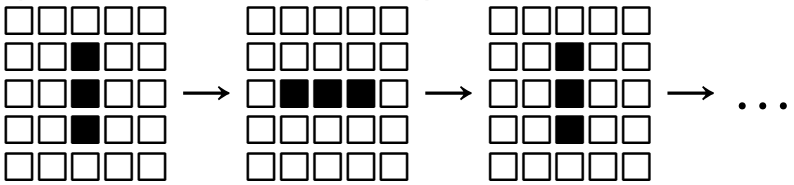
Welche Datenstrukturen würden Sie für eine Implementierung des „Spielfeldes“ wählen?

Conway's Game of Life - Impressionen

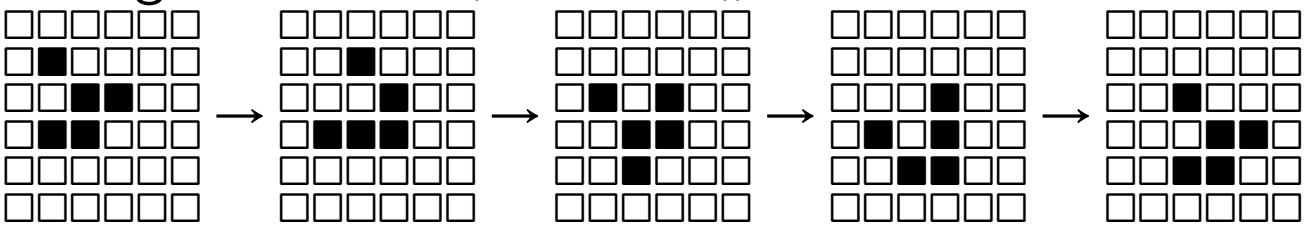
- Statische Muster, z.B. der „Beehive“:



- Oszillierende Muster mit bestimmter Periodenlänge, z.B. der „Blinker“ (Periodenlänge 2):

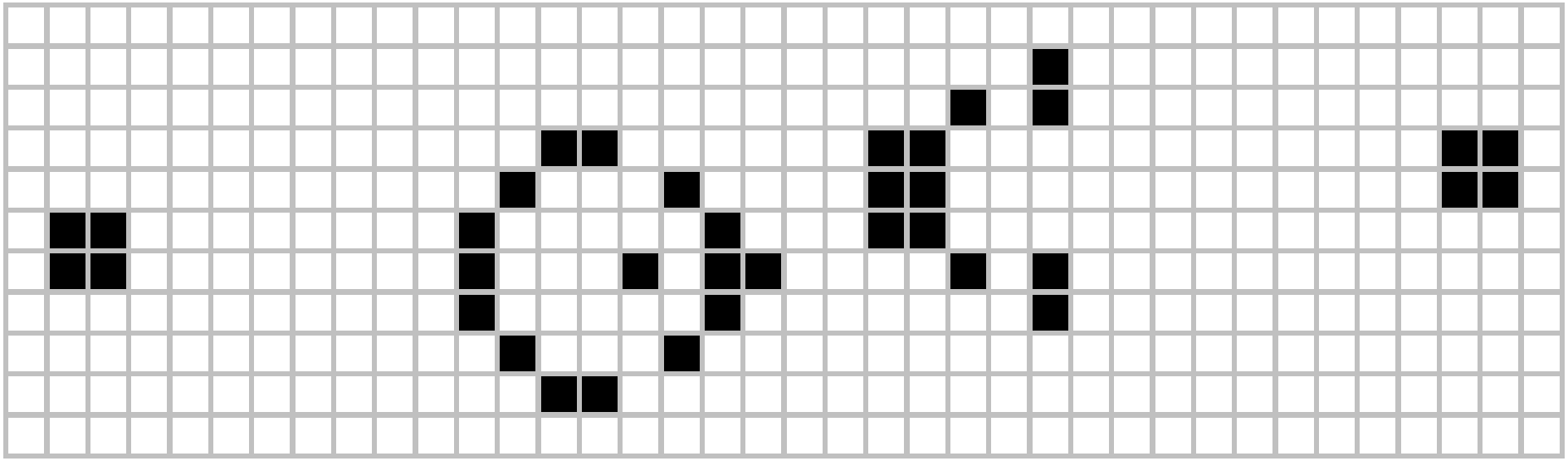


- Bewegliche Muster, z.B. der „Glider“:



Conway's Game of Life - Impressionen

Periodische Muster können bewegliche Muster „generieren“, z.B. die folgende „Glider Gun“:



Ausprobieren! `https://playgameoflife.com/`

Wireworld

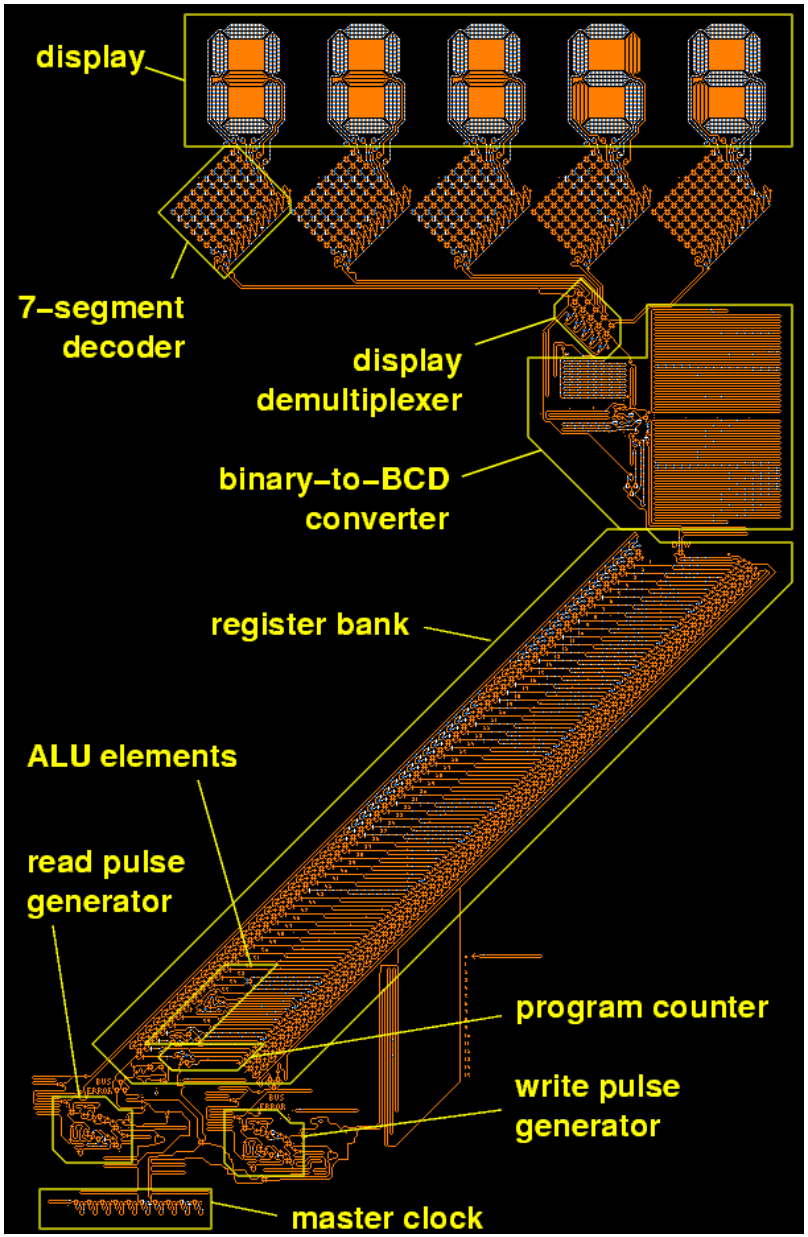
Zweidimensional, Moore-Nachbarschaft, vier Zustände:

- ■ - Nichts
- ■ - elektrischer Leiter
- ■ - „Elektronenkopf“
- ■ - „Elektronenende“

Die Regeln sind so gestaltet, dass sich das „Elektron“ pro Zeitschritt in seine Kopfrichtung bewegt (und da bei ggf. aufgesplittet oder ausgelöscht wird):

- ■ → ■
- ■ → ■
- ■ → ■
- ■ → {■, falls 1 oder 2 mal ■ in Nachbarschaft vorkommt, sonst ■}

Wireworld



Mehr nächste Woche...