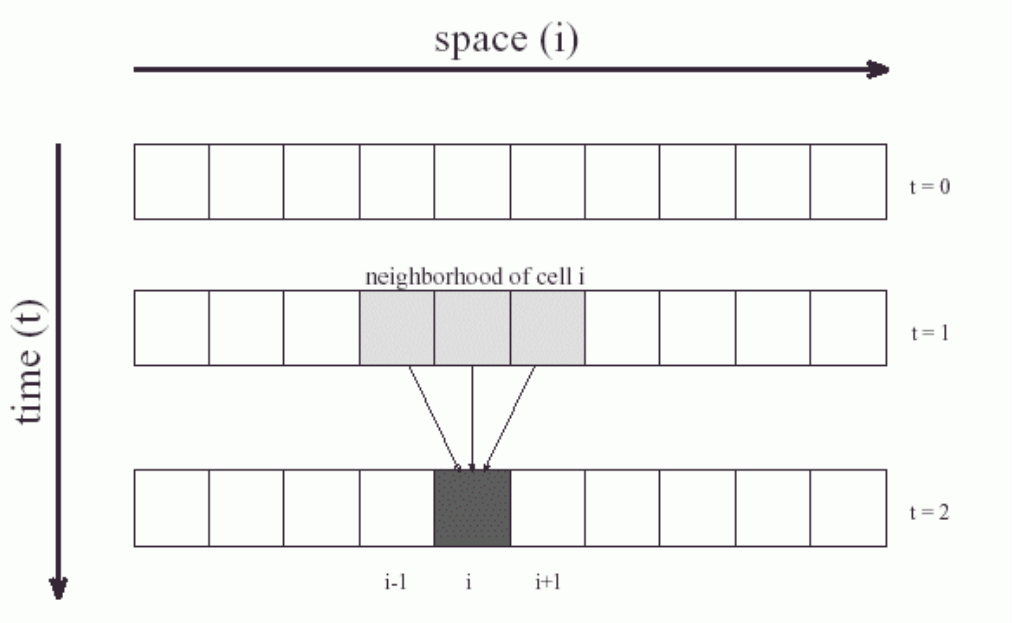
Zelluläre Automaten

(auch: Zellularautomaten, Polyautomaten,
cellular automata, CA)

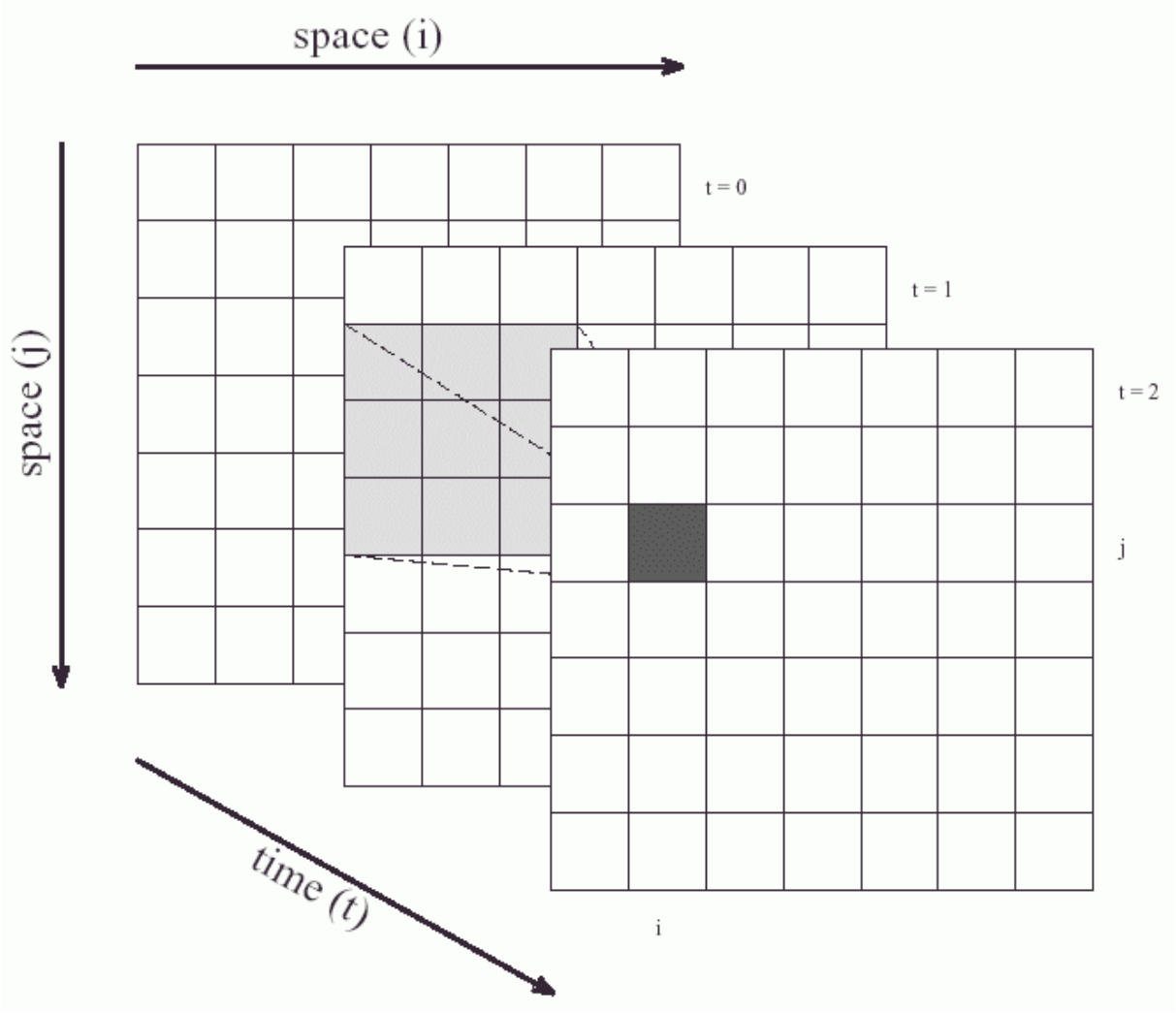
- mathematische Modelle mit diskretem Raum und diskreter Zeit
- die Zeit verläuft in Schritten
- Raum wird als *Gitter* oder *Array von Zellen* repräsentiert
- in den klassischen CA-Modellen kann jede Zelle nur endlich viele Zustände annehmen
- für jede Zelle gilt eine Menge lokaler Regeln, die festlegen, wie sich der neue Zustand dieser Zelle aus ihrem Zustand und dem der Nachbarzellen (im vorherigen Zeitschritt) ergibt.

- Ein zellulärer Automat heißt *homogen*, wenn alle Zellen dieselbe Regelmenge haben. Wir betrachten im Folgenden nur homogene CA.



Je nach Dimensionszahl des zugrundeliegenden Gitters unterscheidet man 1-, 2- und höherdimensionale CA.

Raum und Zeit in einem 2-dimensionalen CA:



Formale Definition von CA:

4-Tupel (L,S,N,f) mit

L regelmäßiges Gitter (Elemente von L: „Zellen“)

S endliche Menge von Zuständen

N endliche Menge von Nachbarschaftsindizes mit

$\forall c \in N, r \in L: r+c \in L$

$f: S^n \rightarrow S$ Übergangsfunktion

Konfiguration $C_t: L \rightarrow S$ weißt jeder Zelle einen Zustand zu,

f ändert C_t zu C_{t+1}: $C_{t+1}(r) = f(\{C_t(i): r \text{ in } N(r)\})$ mit N(r) Nachbarschaft von r.

- Wichtige Eigenschaften:
  - Homogenität
    Jede Zelle hat die selbe Zustandsmenge und Übergangsfunktion.
  - Lokalität
    Zustandsübergänge sind in Zeit und Raum lokal.

Wegen ihrer diskreten Struktur ist die Realisierung von CA auf Rechnern besonders einfach.

Simulation von Zellulären Automaten

- Für jede Generation
  - F = Feld der aktuellen Generation
  - Für jedes Gitterelement F(i,j)
    Wende alle Regeln auf F(i,j) an und speichere Ergebnis in G(i,j)
  - Kopiere G nach F

- Verbesserungen:
  - Statt Kopierschritt abwechselnd aktueller ZA in einem der beiden Felder
  - Kompilieren der Regeln in eine „look-up-table“

- CA können regelbasiert in der Sprache XL implementiert werden

- mit Parallelrechnern hochgradige Parallelisierung der Berechnung möglich.

Das Problem der Gitter-Ränder:

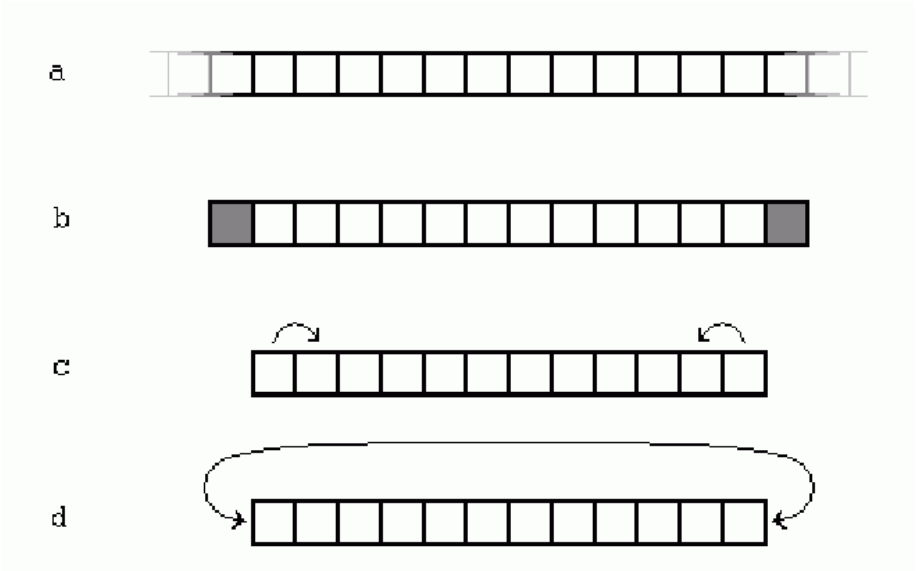
- Simulation eines wirklich unendlichen Gitters unmöglich (simulationstechnisch und möglicherweise durch reales System bedingt - natürliche Grenzen)
- Periodische Grenzen

Bei zwei-dimensionalen Automaten - „Torus“

- Reflektive Grenzen

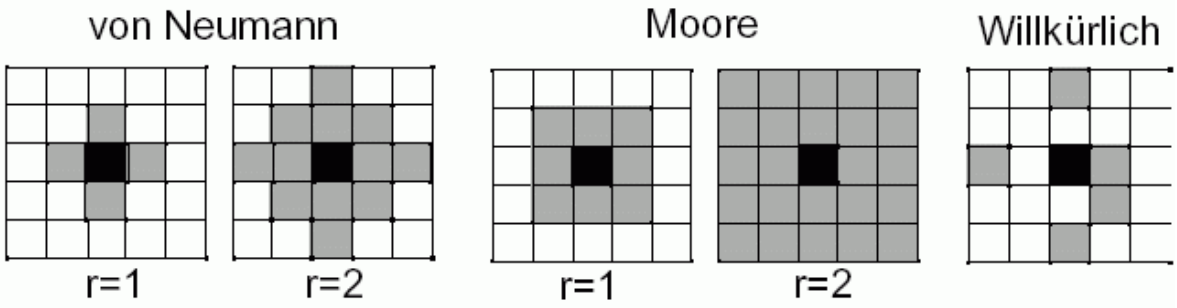
Wenn das reale System Grenzen hat, aber der Wert an der Grenze nicht vorgegeben ist.

- Kombinationen möglich, z.B. um einen langen Tunnel zu simulieren sind periodische Grenzen in der Horizontalen, reflektive Grenzen in der Vertikalen sinnvoll.
- Fixe Grenzen:
  Zellen an der Grenze besitzen fixen Wert.



a: unendliches Gitter, b: fixe Grenzen, c: reflektive Grenzen,
d: periodische Grenzen

Nachbarschaften



Gegeben: Zelle i,j

- Von Neumann-Nachbarschaft
  $N_{i,j} = \{(k,l) \in L : |k-i| + |l-j| \leq r\}$
- Moore-Nachbarschaft
  $N_{i,j} = \{(k,l) \in L : |k-i| \leq r \wedge |l-j| \leq r\}$
- Grundsätzlich kann jede Art von Nachbarschaftsmenge gebildet werden, solange sie für alle Zellen gleich und endlich ist.
- Große Nachbarschaften ➔ bessere Isotropie, beim Simulieren ineffizient.

Zustände

- jede Zelle in G befindet sich in einem definierten Zustand, der Element der endlichen Menge von Elementarzuständen ist, z.B. E={ein,aus}

Lokale Funktion / Regeln

- definiert für jede Zelle und alle möglichen Umgebungen den nächsten Zustand, z.B. als Regel
- Regeln heißen **totalistisch**, falls sie nur summarisch von den Nachbarzuständen abhängen
- Regeln sind i.Allg. nicht umkehrbar - eine Konfiguration kann verschiedene Vorgänger haben
- **probabilistische** Regeln erlauben, im Gegensatz zu **deterministischen** Regeln eine stochastische Auswahl mehrerer Folgezustände für die gleiche Umgebung

Regelanwendung

- **synchrone** Automaten wenden f zu jedem Zeitschritt auf alle Zellen gleichzeitig an
- **asynchrone** Automaten wenden f sequentiell auf alle Zellen an:
  - in fester, deterministischer Folge
  - in stochastischer Folge
  - in dynamisch gesteuerter, deterministischer Folge
  - ...

im Folgenden werden nur synchrone CA betrachtet.

Beispiel eines zweidimensionalen zellulären Automaten:

John H. Conway (späte 60er Jahre): "*Game of Life*"

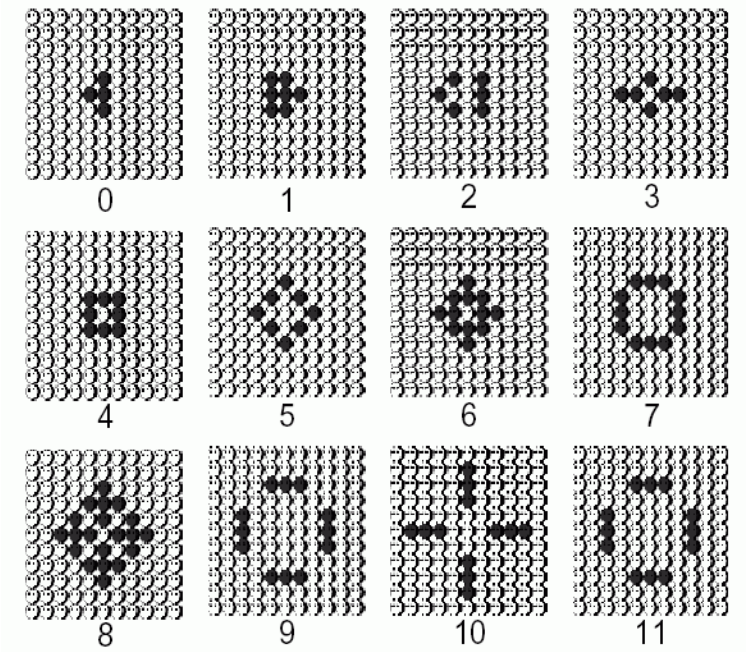
1970 von Martin Gardner in seiner mathematischen Kolumne in "Scientific American" vorgestellt, löste Welle weltweiter Forschung an diesem CA aus.

- regelmäßiges Quadratgitter
- nur 2 Zustände ("tot" und "lebendig", bzw. "0" und "1")
- Moore-Nachbarschaft mit Radius 1, d.h. es werden 8 Nachbarzellen betrachtet
- Regeln:

Life-Regeln

- Eine Zelle ist im nächsten Takt „lebendig“, wenn genau drei ihrer Nachbarn lebendig sind
- Eine lebendige Zelle ist im nächsten Takt „tot“, wenn weniger als 2 oder mehr als 4 ihrer Nachbarn lebendig sind.

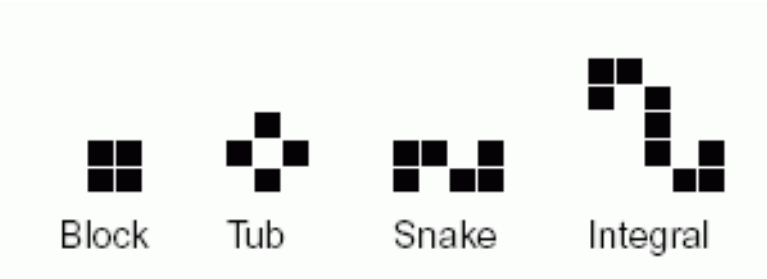
Beispiel-Entwicklung:



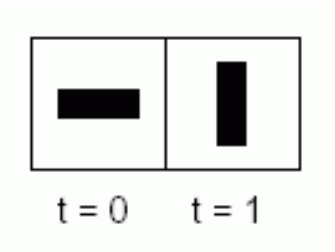
Muster lassen sich durch ihr Verhalten (bei der weiteren Entwicklung) klassifizieren.

Beispiele:

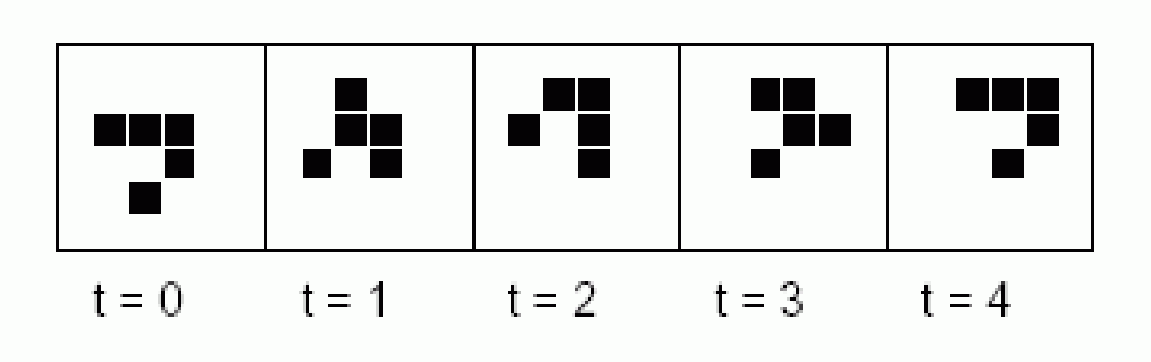
Typ I: "Stillleben", Muster, die sich nicht ändern (Fixpunkte der Übergangsfunktion):



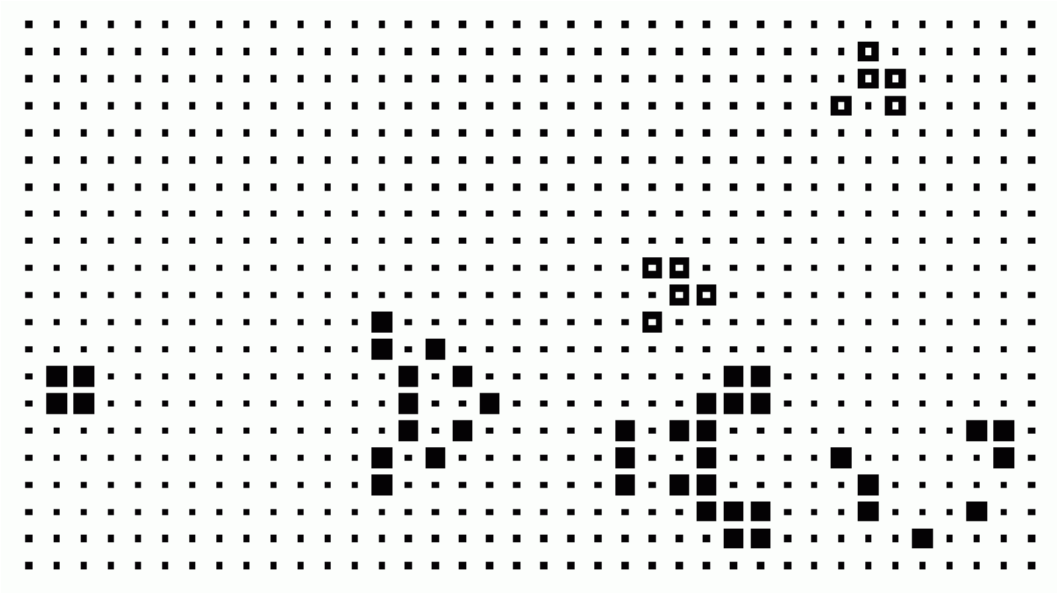
Typ II: Oszillatoren (periodische Muster): z.B. der "Blinker"



Typ III: "Raumschiffe" = Muster, die nach einer endlichen Zahl von Schritten wiederkehren, aber räumlich verschoben. Muster, die sich mit konstanter Geschwindigkeit über das Gitter bewegen. Beispiel: der "Gleiter"



Typ IVa: "Kanonen" = Oszillatoren, die in jedem Zyklus Raumschiffe emittieren.
Gleiterkanone:



Typ IVb: "Dampfer" = Raumschiffe, die Stillleben, Oszillatoren und / oder Raumschiffe hinter sich lassen.

Typ IVc: "Brüter" = Muster, die ihre Populationsgröße quadratisch (oder noch schneller) vergrößern (z.B. ein Raumschiff, das Gleiterkanonen produziert).

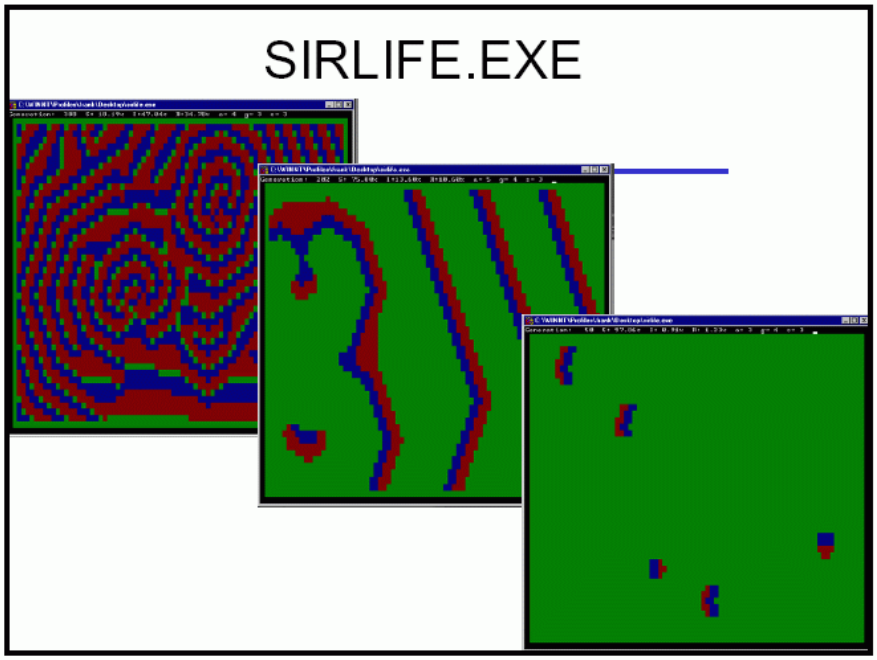
Typ V: unstabile Muster. Muster, die sich durch eine Folge von Zuständen weiterentwickeln und nie zum Ausgangszustand zurückkehren. Kleine Muster mit "langer" Entwicklungsphase vor der Stabilisierung heißen "Methusalems".

The Game of Life
Beobachtungen

- für eine gegebene Anfangskonfiguration kann i.Allg. das zukünftige Verhalten nicht ohne Simulation vorhergesagt werden.
- einige Startzustände können nicht im Laufe der Simulation entstehen. Diese Zustände heißen *Garten-Eden* Konfigurationen.
- das Spiel ist (berechnungs-)universell. Der Beweis wird über die Konstruktion und Interaktion logischer Gatter geführt.

Praktischer Einsatz zweidimensionaler CA:
in der Simulation natürlicher Phänomene, z.B. der Ausbreitung von Epidemien

Beispiel:



(Universität Leipzig)