Natural Computing

LMU Munich
summer term 2025

Thomas Gabor

Definition 3 (Conway's game of life (standard)). Let $G = (V, E)$ be a graph with vertices V and (undirected) edges $E \subseteq V \times V$ with a fixed degree of 8 for all nodes. We define *surroundings* : $V \to \wp(V)$ via

$$surroundings(v) = \{w \mid (v, w) \in E\}$$

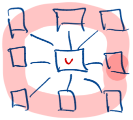
so that $|surroundings(v)| = 8$ and $v \notin surroundings(v)$ for all $v \in V$. A state $x \in \mathcal{X}$ is a mapping of vertices to the labels $\{dead, alive\}$, i.e., the state space $\mathcal{X}$ is given via $\mathcal{X} = (V \to \{dead, alive\})$. Let x_t be a state that exists at time step $t \in \mathbb{N}$. We define

$$|v|_{x_t} = |\{w \mid w \in surroundings(v) \wedge x_t(w) = alive\}|.$$

In the game of life, the evolution of a state x_t to its subsequent state x_{t+1} is given deterministically via

$$x_{t+1}(v) = \begin{cases} dead & \text{if } |v|_{x_t} \leq 1, \\ x_t(v) & \text{if } |v|_{x_t} = 2, \\ alive & \text{if } |v|_{x_t} = 3, \\ dead & \text{if } |v|_{x_t} \geq 4, \end{cases}$$

for all $v \in V$. A tuple (G, x_0) is called an instance of the game of life for initial state $x_0 \in \mathcal{X}$.



$|v|_{x_t}$

Let’s try

conwaylife.com

Still Lifes

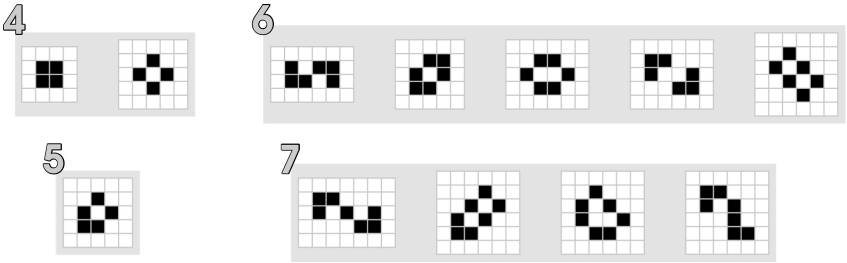


Figure 2.1: All still lifes with 7 or fewer live cells, arranged by their cell count. From left to right: (4 cells) block, tub, (5 cells) boat, (6 cells), snake, ship, beehive, aircraft carrier, barge, (7 cells) long snake, long boat, loaf, and eater 1 (which is sometimes called **fishhook**).

source: conwaylife.com/book/

Oscillators

(a) bipole (p2). **(b) jam** (p3). **(c) mold** (p4). **(d) octagon 2** (p5).

Figure 3.1: Some more small naturally occurring (but rare) oscillators that can be found via computer-assisted soup searches. These oscillators were found by (a) early Lifenthusiasts at M.I.T. in 1970, (b,c) Achim Flammenkamp in 1988,[2] and (d) Sol Goodman and Arthur Taber in 1971.

source: conwaylife.com/book/

Space Ships

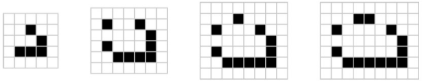
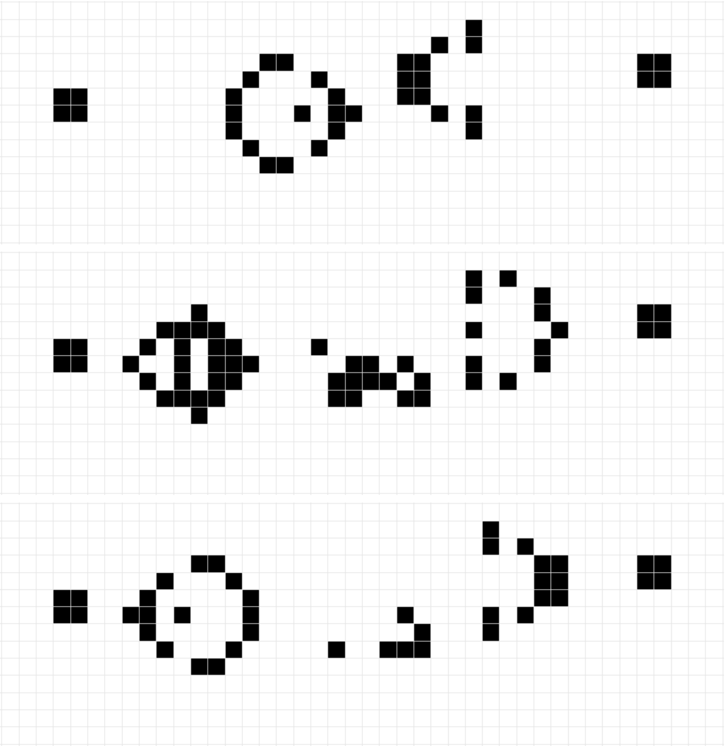


Figure 4.1: The four basic spaceships in Conway's Game of Life. From left to right, these are the glider (which moves diagonally at a speed of $c/4$) and the light/middle/heavyweight spaceships (which each move orthogonally[1] at a speed of $c/2$).

source: conwaylife.com/book/

Guns



t = 1

t = 14

t = 15

[try online](https://conwaylife.com)

source: conwaylife.com

Space Ship Maneuvering

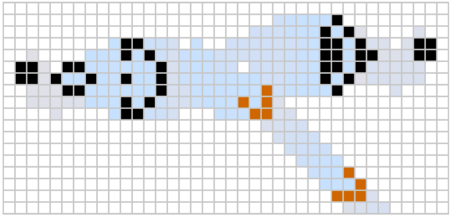


Figure 6.2: A Gosper glider gun producing gliders at a spacing of 30 generations.

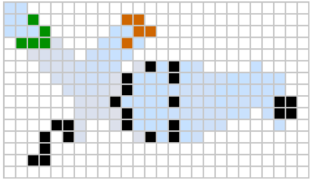


Figure 6.3: A buckaroo can reflect a glider by 90 degrees, from the position marked in green to the one in orange 30 generations later.

source: conwaylife.com/book/

More Space Ship Maneuvering

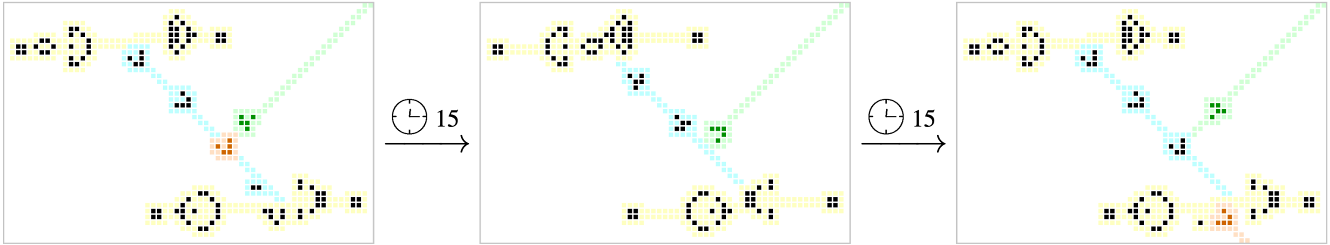


Figure 6.7: Two Gosper glider guns can be placed near each other so as to create a finite stream of gliders (highlighted here in aqua) between them. Here, we bounce a single glider (in green) off of one of those gliders (in orange) so that it is destroyed and thus released by the inline inverter to the southeast.

source: conwaylife.com/book/

conwaylife.com/book/periodic_circuitry

- No initial pattern with simply proven unlimited growth
- Initial patterns with apparently unlimited growth
- Simple initial patterns with behavior for a long period of time

Gardner, 1970.

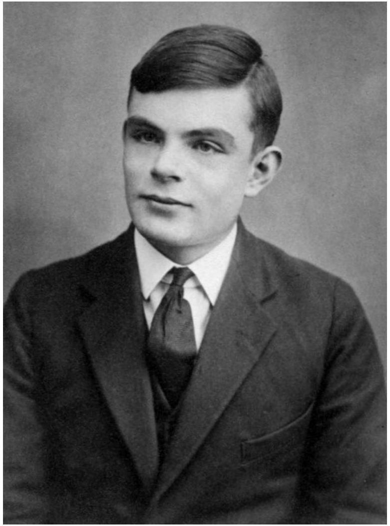
Why is this interesting?

Turing-Completeness

Alan Turing

1912 – 1954

- invented the field of (theoretical) computer science and artificial intelligence
- influential in breaking Enigma and thus shortening WWII



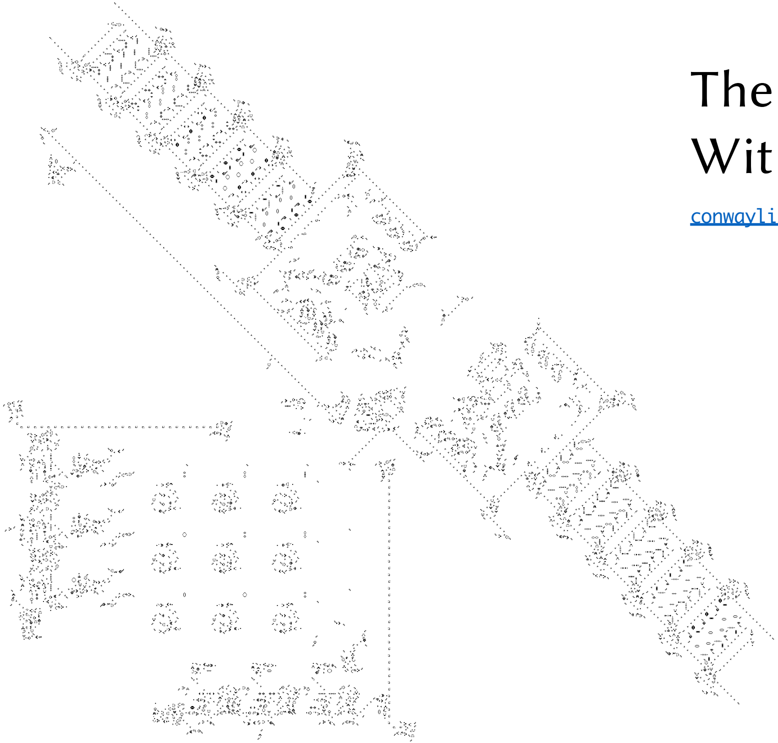
source: Wikipedia

The Turing Machine

The Turing Machine

Definition 4 (Turing machine [1]). A Turing machine is a tuple $(Q, \Sigma, \Gamma, \sqcup, q_S, q_F, \delta)$ so that

- Q is a finite set of states,
- Σ is a finite input/output alphabet (e.g., $\Sigma = \{0, 1\}$),
- Γ is a finite tape alphabet with $\Sigma \subset \Gamma$,
- $\sqcup \in \Gamma$ is a blank symbol with $\sqcup \in \Sigma$,
- $q_S \in Q$ is a start state,
- $q_F \in Q$ is a stop state,
- and $\delta : Q \setminus \{q_F\} \times \Gamma \to Q \times \Gamma \times \{L, R, N\}$ is a partially defined transition function.



The Turing Machine Within the Game of Life

conwaylife.com/wiki/Turing_machine

But… why?

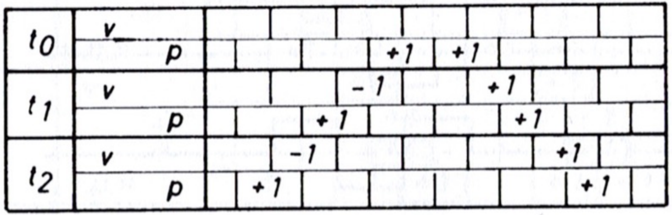
Theorem 1 (Church-Turing thesis)**.** Any computable function can be computed by a Turing machine.

Theorem 1 (Church-Turing thesis). Any computable function can be computed by a Turing machine.

Theorem 2 (extended Church-Turing thesis). Any computable function can be computed by a Turing machine *with at most polynomial overhead*.

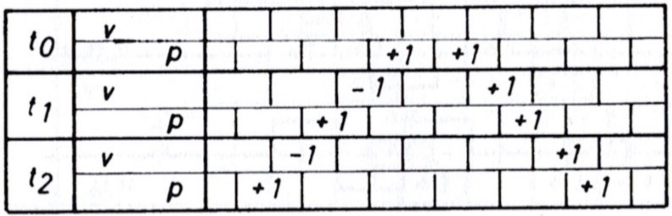
Can cellular automata explain the universe?

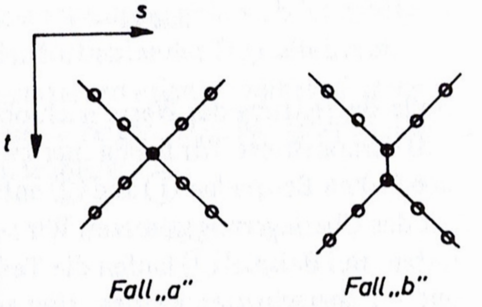
state space $\mathcal{X} = (V \to \mathbb{Z} \times \mathbb{Z})$



Zuse. Rechnender Raum. 1969

state space $\mathcal{X} = (V \to \mathbb{Z} \times \mathbb{Z})$





Zuse. Rechnender Raum. 1969

Konrad Zuse

1910 – 1995

- built first programmable computer (Zuse Z3)
- designed first high-level programming language (Plankalkül)

source: Wikipedia

Example 1 (Cellular Automaton for 1D Particle Collision, inspired by [1]). Consider the following variation of a 1D cellular automaton (cf. Definitions 1 and 2 on the definition sheet): Let the state space $\mathcal{X} = (V \to \mathbb{Z} \times \mathbb{Z})$. Let

$$f\big((v_l, p_l), (v_c, p_c), (v_r, p_r)\big) = (v_l^+ + v_r^- + \lfloor \frac{(p_l - p_c)^+}{2} \rfloor - \lfloor \frac{(p_r - p_c)^+}{2} \rfloor,$$
$$p_c - |v_c| + v_l^+ - v_r^- + \lfloor \frac{(p_l - p_c)^+}{2} \rfloor + \lfloor \frac{(p_r - p_c)^+}{2} \rfloor - \lfloor \frac{(p_c - p_l)^+}{2} \rfloor - \lfloor \frac{(p_c - p_r)^+}{2} \rfloor)$$

where $x^+ = \begin{cases} x & \text{if } x > 0, \\ 0 & \text{otherwise,} \end{cases}$ and $x^- = \begin{cases} x & \text{if } x < 0, \\ 0 & \text{otherwise,} \end{cases}$ for any $x \in \mathbb{Z}$.

Note: $\lfloor x \rfloor$ for $x \in \mathbb{R}$ is x rounded down to an integer as implemented in a typical `floor` function (see Python, e.g.).
From the start configuration of this 1d cellular automaton with 5 cells below, compute the next 3 time steps.

$(0, 0)$	$(+1, +1)$	$(0, 0)$	$(-1, +1)$	$(0, 0)$

```python
from math import floor

def eplus(x):
    return x if x > 0 else 0

def eminus(x):
    return x if x < 0 else 0

def f(left,center,right):
    vl,pl = left
    vc,pc = center
    vr,pr = right
    pressure_from_left = floor(eplus(pl-pc)/2)
    pressure_from_right = floor(eplus(pr-pc)/2)
    pressure_towards_left = floor(eplus(pc-pl)/2)
    pressure_towards_right = floor(eplus(pc-pr)/2)
    vnew = eplus(vl) + eminus(vr) + pressure_from_left - pressure_from_right
    pnew = pc - abs(vc) + eplus(vl) - eminus(vr) + pressure_from_left + pressure_from_right - pressure_towards_left - pressure_towards_right
    return vnew, pnew

def apply(vector, f):
    new_vector = []
    for e,element in enumerate(vector):
        new_element = f(vector[e-1], element, vector[(e+1)%len(vector)])
        new_vector.append(new_element)
    return new_vector

rounds = range(4)
vector = [(0,0), (+1,+1), (0,0), (-1,+1), (0,0)]
for _ in rounds:
    print(vector)
    vector = apply(vector,f)
```

Example 1 (Cellular Automaton for 1D Particle Collision, inspired by [1]). Consider the following variation of a 1D cellular automaton (cf. Definitions 1 and 2 on the definition sheet): Let the state space $\mathcal{X} = (V \to \mathbb{Z} \times \mathbb{Z})$. Let

$$f\big((v_l, p_l), (v_c, p_c), (v_r, p_r)\big) = (v_l^+ + v_r^- + \lfloor \frac{(p_l - p_c)^+}{2} \rfloor - \lfloor \frac{(p_r - p_c)^+}{2} \rfloor,$$
$$p_c - |v_c| + v_l^+ - v_r^- + \lfloor \frac{(p_l - p_c)^+}{2} \rfloor + \lfloor \frac{(p_r - p_c)^+}{2} \rfloor - \lfloor \frac{(p_c - p_l)^+}{2} \rfloor - \lfloor \frac{(p_c - p_r)^+}{2} \rfloor)$$

where $x^+ = \begin{cases} x & \text{if } x > 0, \\ 0 & \text{otherwise,} \end{cases}$ and $x^- = \begin{cases} x & \text{if } x < 0, \\ 0 & \text{otherwise,} \end{cases}$ for any $x \in \mathbb{Z}$.

Note: $\lfloor x \rfloor$ for $x \in \mathbb{R}$ is x rounded down to an integer as implemented in a typical `floor` function (see Python, e.g.).
From the start configuration of this 1d cellular automaton with 5 cells below, compute the next 3 time steps.

$(0,0)$	$(+1,+1)$	$(0,0)$	$(-1,+1)$	$(0,0)$
$(0,0)$	$(0,0)$	$(0,2)$	$(0,0)$	$(0,0)$
$(0,0)$	$(-1,+1)$	$(0,0)$	$(+1,+1)$	$(0,0)$
$(-1,+1)$	$(0,0)$	$(0,0)$	$(0,0)$	$(+1,+1)$

What about
non-local
information?

Quantum Computing

What is quantum computing?

What is quantum computing within natural computing?

two strange
quantum effects