Automaty komórkowe

D. Szeliga A. Cichoń M. Kielian

Kraków, 2016

Plan

1. Trochę historii.
2. Czym są automaty komórkowe?
3. Podział Wolframa.
4. Automaty jednowymiarowe.
5. Automaty dwuwymiarowe.
6. Sąsiedztwo komórek.
7. Warunki brzegowe.
8. Warunki początkowe.
9. Reguły przejść.
10. Ewolucja automatu komórkowego.
11. Przykłady.

Trochę historii

Za twórcę automatów komórkowych uważa się Janosa von Neumanna, Węgra pracującego w Princeton. Wprowadził on koncepcję samoreprodukcji, docelowo chciał stworzyć model maszyny samosterującej, tzn. takiej, iż powielałaby ona swoją budowę i przekazywała swoje cechy. Na przełomie lat czterdziestych i pięćdziesiątych Neumann opracował swoją teorię opierając się na maszynie Turinga. Opracował on pięć modeli samoreplikujących się automatów, których jednak realizacja okazała się zbyt trudna. Zainspirowany pomysłem Stanisława Ulama, lwowskiego matematyka, który przebywał wtedy w Los Alamos, wprowadził do swego modelu dyskretny czas i przestrzeń.

Pracami Neumanna zainteresował się dopiero Edgar Frank Codd, który uczynił automaty możliwymi do wykorzystania. Codd zaprojektował automat komórkowy, który mógł obliczyć wszystkie możliwe funkcje, i który mógł się rozmnażać. Projekt posłużył do skonstruowania powszechnie stosowanej Gry w życie.

Przełomowym wydarzeniem w historii automatów komórkowych było sklasyfikowanie ich – po wcześniejszych czysto teoretycznych projektach w 1983 roku Stephen Wolfram dokonuje klasyfikacji automatów komórkowych.

Czym są automaty komórkowe?

- Automat komórkowy składa się z regularnej sieci, której komórki mogą znajdować się w skończonej liczbie stanów.
- Automaty komórkowe, których struktury opisane są przez siatkę komórek oraz ich stany, przejścia i reguły tych przejść, są modelami matematycznymi. Tworzą one środowisko dla większych dyskretnych klas modeli, ponieważ wszystkie opisujące je struktury przyjmują wartości dyskretne.
- Są one alternatywną matematyką, szczególnie przydatną w obliczeniach równoległych, wolną od błędów zaokrągleń oraz narzędziem do symulacji procesów fizycznych, w których bierze udział wiele układów oddziałujących ze sobą.
- Dla każdej komórki definiuje się sąsiedztwo i w chwili t=0 przypisuje stan początkowy. Nowa generacja jest tworzona na podstawie podanych zasad wiążących nowy stan komórki ze stanem poprzednim jej i jej sąsiadów.

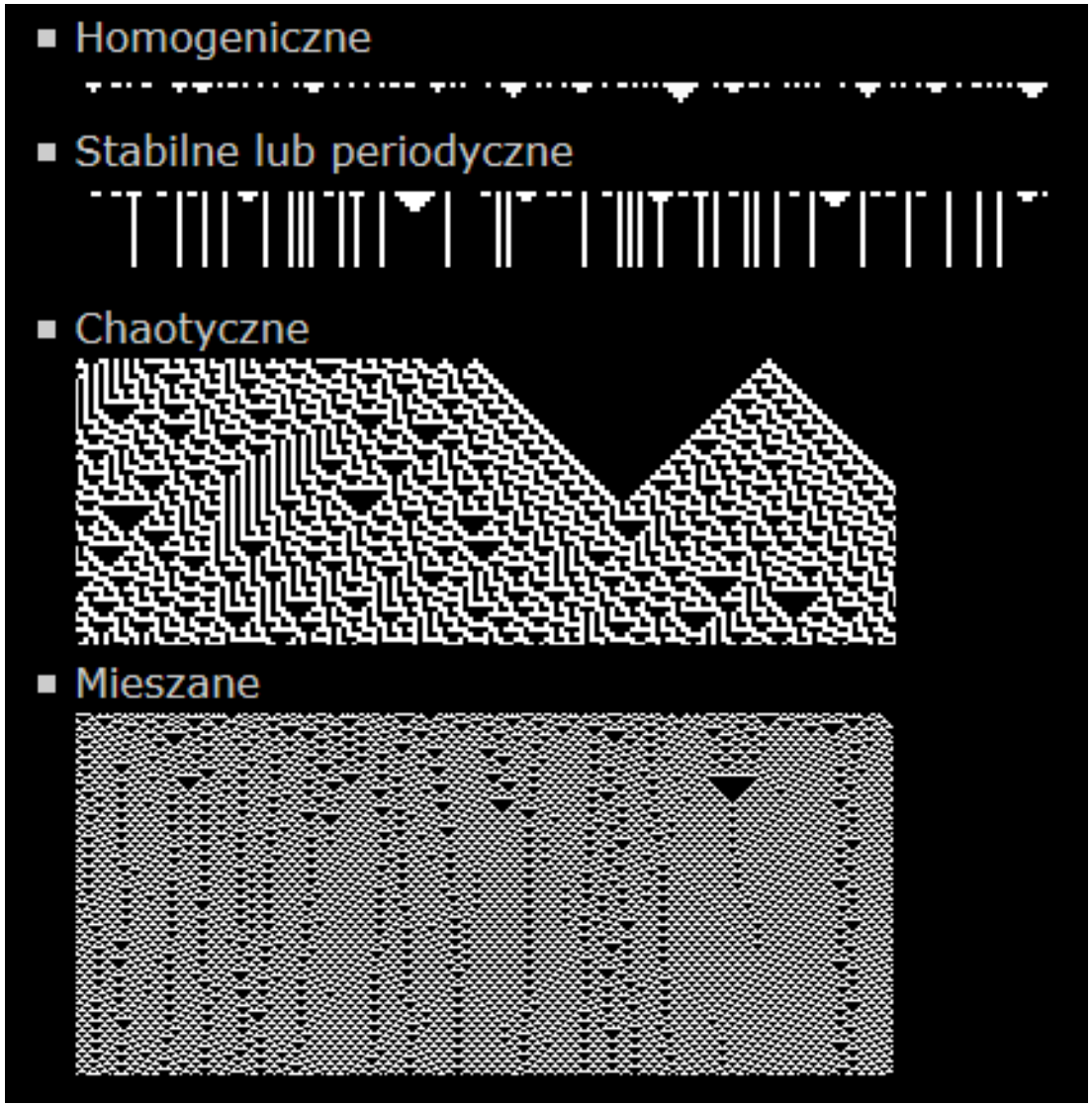
Podział Wolframa

- Klasa I: Automaty niezmienne – ewoluują do czasu, kiedy wszystkie komórki osiągną identyczny stan niezależnie od stanu początkowego (zbieżne).
- Klasa II: Automaty ewoluujące do stanu stabilnego lub okresowych wzorców (okresowe).
- Klasa III: Automaty wykazujące nieporządek zarówno lokalnie jak i globalnie, nie wykazujące żadnego wzorca (chaotyczne).
- Klasa IV:Automaty wykazujące bardziej złożone, długotrwałe zachowanie („żywe").

Automaty jednowymiarowe

Automaty komórkowe mogą tworzyć struktury zadziwiająca przypominające swoją budową twory naturalne. Jednowymiarowy automat o regule 22 tworzy nieregularne, sprawiające wrażenie przypadkowości, struktury o chaotycznych zagęszczeniach. Taka struktura przypomina powierzchnie niektórych rodzai muszli morskich.

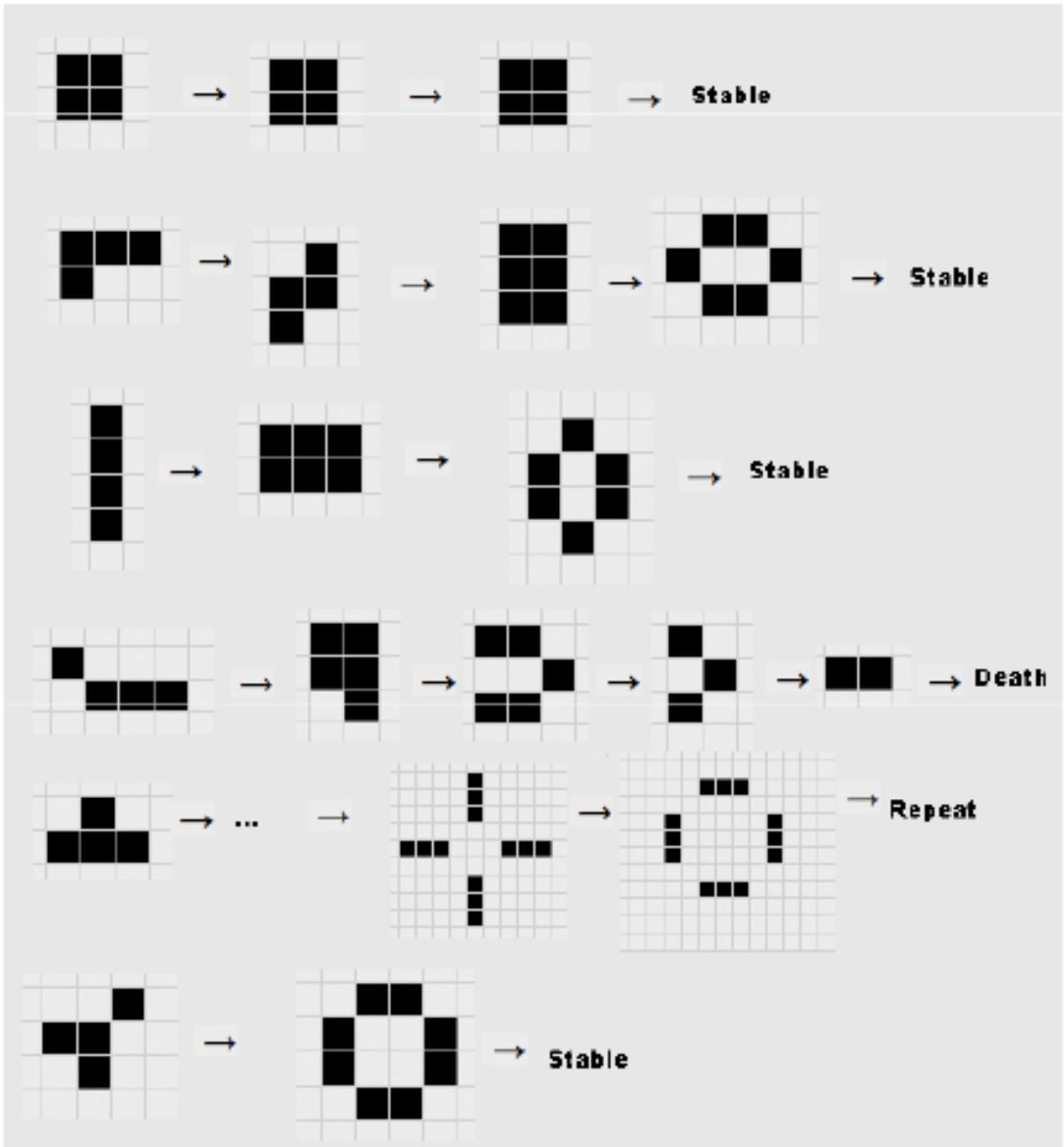
Stephen Wolfram podzielił rozważaną przez siebie grupę automatów na cztery klasy.



Automaty dwuwymiarowe

Dwuwymiarowe automaty komórkowe - klasy (2,1).
Komórki w grze wymyślonej przez Conwaya, zwanej "Gra w życie", mogą być w jednym z dwu stanów, a funkcja decydująca o zmianie stanu jest bardzo prosta. Komórka martwa, mająca dokładnie trzech sąsiadów, staje się żywa. Komórka żywa żyje nadal, jeśli ma dokładnie dwóch lub trzech żywych sąsiadów. Sama powyższa reguła pozwoliła odnaleźć prawdziwe bogactwo form, m.in. stałe układy komórek, oscylatory, obiekty przesuwające się, a nawet takie które wykonują działania arytmetyczne, logiczne czy nawet symulują komputer (w sensie Turinga).

Gra w życie zawiera mnóstwo wzorców, które pozostają stabilne od iteracji do iteracji oraz wzorców, które potrafią się replikować.



Sąsiedztwo komórek

- Sąsiedztwo von Neumanna- Jeśli oznaczymy kierunki na zasadzie róży wiatrów: N, S, E, W oraz kierunki pośrednie NW, NE, SE, SW, to sąsiedztwem von Neumanna będzie zbiór czterech komórek: N, S, E, W.
- Sąsiedztwo Moore'a- Posługując się powyższymi oznaczeniami, sąsiedztwem Moore'a będzie zbiór wszystkich ośmiu komórek dookoła komórki centralnej.
- Sąsiedztwo Margolusa- Stosuje się je w automatach do symulacji spadającego piasku, czy też interakcji cząsteczek gazu. Reguły przejścia opierają się na kwadratowych blokach tworzonych przez cztery sąsiadujące komórki. Stany tych sąsiednich komórek zmieniają się jednocześnie, przy czym komórki przyjmują wartości binarne 1 i 0. Tak dzieje się na całej siatce automatu. W następnym kroku reguły są obliczane podobnie, tylko że zmieniają się grupy komórek. Bloki tworzące owe grupy przesuwają się o jeden w prawo i w dół.

Warunki brzegowe

- Periodyczne- definiują one zamkniętą siatkę w taki sposób, że np. symulując poruszającą się cząstkę po dojściu do krawędzi pojawi się ona z drugiej strony. Komórka znajdująca się na brzegu siatki ma za sąsiada komórkę leżącą po drugiej stronie siatki.
- Zamknięte pochłaniające- siatka jest zdefiniowana w taki sposób, że brzegi siatki wypełnione są z góry ustaloną wartością, która poprzez funkcję przejścia ustala wpływ na zachowanie automatu. W praktyce, symulując np. cząstkę gazu, po przekroczeniu krawędzi siatki przestaje ona istnieć.
- Zamknięte odbijające- warunki brzegowe na krawędzi siatki tworzą barierę, od które symulowane cząstki się odbijają. Stosowane do symulacji zamkniętych przestrzeni doświadczalnych.

Warunki początkowe

Ważnym elementem konstrukcyjnym automatów komórkowych są warunki początkowe, czyli stany poszczególnych komórek w zerowej iteracji czyli na samym początku. To od ustawienia początkowego komórek zależy dalsza ewolucja automatu, jego zachowanie, stan końcowy, tym samym powodzenie całej symulacji. Niektóre automaty komórkowe z założenia muszą mieć w odpowiedni sposób ustalone warunki początkowe.

Reguły przejść

Reguły przejść określają ewolucję automatu komórkowego w dyskretnym czasie. Stany poszczególnych komórek aktualizuje się w każdej dyskretnej chwili t = 1, 2, 3, ... Tym samym każdy automat komórkowy jest obiektem dynamicznym w czasie. Jak już wcześniej wspomniano, stan każdej komórki można określić na podstawie aktualnych stanów komórek sąsiednich. Stan komórki x w chwili t oznaczmy jako x_t, a stan sąsiedztwa jako $u(x_t)$. Dla takich kryteriów stan komórki w kolejnym kroku iteracji można opisać wzorem: $x_{t+1} = f(u(x_t), x_t)$ gdzie f określa funkcję przejścia, która może być opisana różnego rodzaju zależnościami, np. jako tabela przejść, w postaci algorytmicznej lub jako zbiór reguł. Reguły przejść muszą być zdefiniowane obok przestrzeni stanów oraz zdefiniowanego sąsiedztwa, ponieważ w innym przypadku automat nie mógłby ewoluować.

Przepis na nowy stan węzła x opisywany jest przez reguły. Jako przykład weźmy 1-D atomat dwustanowy $N = 2$ i sąsiedztwo $r = 1$.

stan sąsiedztwa dla t	111	110	101	100	011	010	001	000
stan węzła dla t	0	1	1	0	1	1	1	0

Tak zapiisaną regułę można odczytać jako liczbę binarną i przedstawić ją w systemie dziesiętnym jako 110. Z powyższej tabelki widzimy, że tak rozumianych reguł w tym przypadku mamy $2^8 = 256$. Używana terminologia:

- *legal* całe sąsiedztwo 0 maouje się na stan 0, i mamy izotropię
- *totalistic* stan węzła zależy tylko od sumy sąsiedztwa $x(t+1) = f\left(\sum_{\text{sąsiad}} x(t)\right)$
- *perypherial* stan węzła nie wpływa na swój przyszły stan

Ewolucja automatu komórkowego

Cały proces ewolucji automatu można podzielić na kilka części:
1. Stan początkowy – jest to, wspomniane już, ustalenie warunków początkowych. Zwykle są to stany neutralne nie powodujące konfliktów w automacie.
2. Aktualizacja siatki automatu – w każdej iteracji każda z komórek automatu przechodzi przez poniższą sekwencję instrukcji:

- Sprawdzenie reguł przejść.
- Sprawdzanie sąsiedztwa.
- Sprawdzanie warunków brzegowych.
- Sprawdzenie ilości iteracji.
- Zwiększenie licznika iteracji i przejście do kroku 2.1.

Przykłady

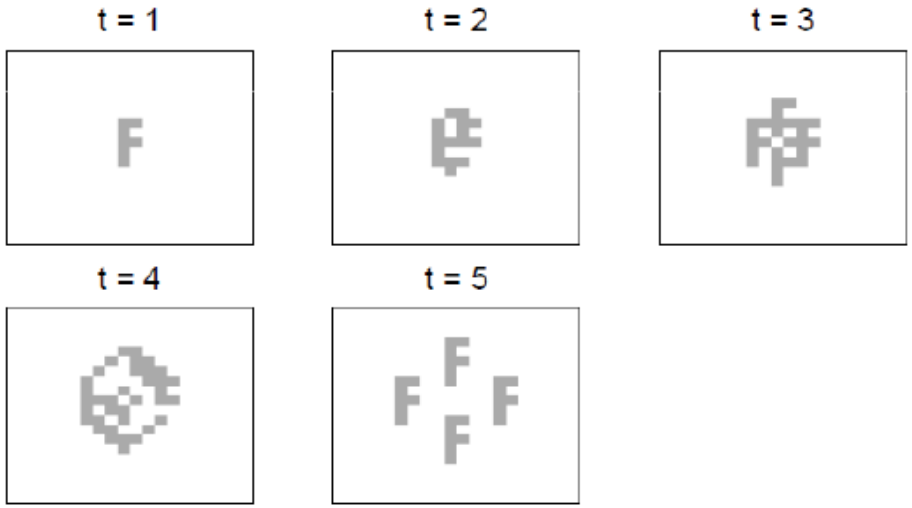
Technika automatów komórkowych jest używana do symulacji komputerowych w wielu problemach nauki i techniki. Automaty te dostarczają również wielu zagadnień teorii dynamiki nieliniowej. Popularne automaty komórkowe oprócz "Gry w życie", to również "Mrówka Langtona", "Gra Fredkina", "Wireworld", "Pętle Langtona".

Mrówka Langtona

- Mrówka Langtona – automat komórkowy stworzony przez Chrisa Langtona w 1986r.
- W każdym kroku wyróżniona jest jedna komórka, która ma określony kierunek, w którym się porusza. Zasady poruszania się:
  - jeśli znajduje się na białym polu to obraca się w lewo, zmienia kolor pola na czarny i przechodzi do następnej komórki,
  - jeśli znajduje się na polu czarnym to obraca się w prawo, zmienia kolor pola na biały i przechodzi na następną komórkę;
- Początkowo wszystkie komórki są białe. Po kilkudziesięciu krokach komórki tworzą formę chaotyczną. Jednak po ok. 10 tysiącach kroków mrówka zaczyna replikować pewien charakterystyczny wzór (tzw. autostradę).
- Mrówka Langtona jest maszyną Turinga.

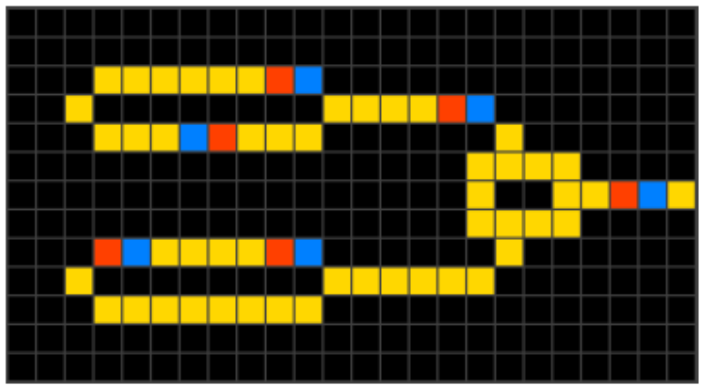
Gra Fredkina

- Gra Fredkina – automat komórkowy pokazujący samoreplikację.
- Każda komórka ma dwa możliwe stany: żywy lub martwy.
- Wszystkie komórki są aktualizowane w jednym kroku algorytmu. Zliczamy ilość żywych i martwych sąsiadów w otoczeniu von Neumanna.
- Każda komórka posiadająca 0, 2 lub 4 żywych sąsiadów będzie martwa w następnym kroku. W innym przypadku komórka ożywa.
- Wiele warunków początkowych prowadzi do czterokrotnej replikacji po 2^n iteracji. 4 repliki będą odległe o 2^n komorek od oryginału.



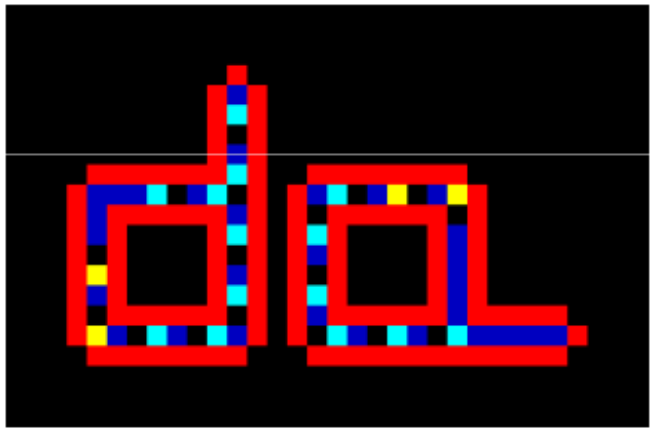
Wireworld

- Wireworld to automat zaproponowany przez Silvermana w 1987.
- Automat symuluje elementy logiczne, takie jak bramki.
- Komórka może być w 4 stanach:
  - pusty (czarny)
  - głowa elektronu (niebieski)
  - ogon elektronu (czerwony)
  - przewodnik (żółty)
- Reguły automatu
  - pusty → pusty
  - głowa elektronu → ogon elektronu
  - ogon elektronu → przewodnik
  - przewodnik → głowa elektronu, jeśli 1 lub 2 sąsiadów są głowami; inaczej pozostaje przewodnikiem.

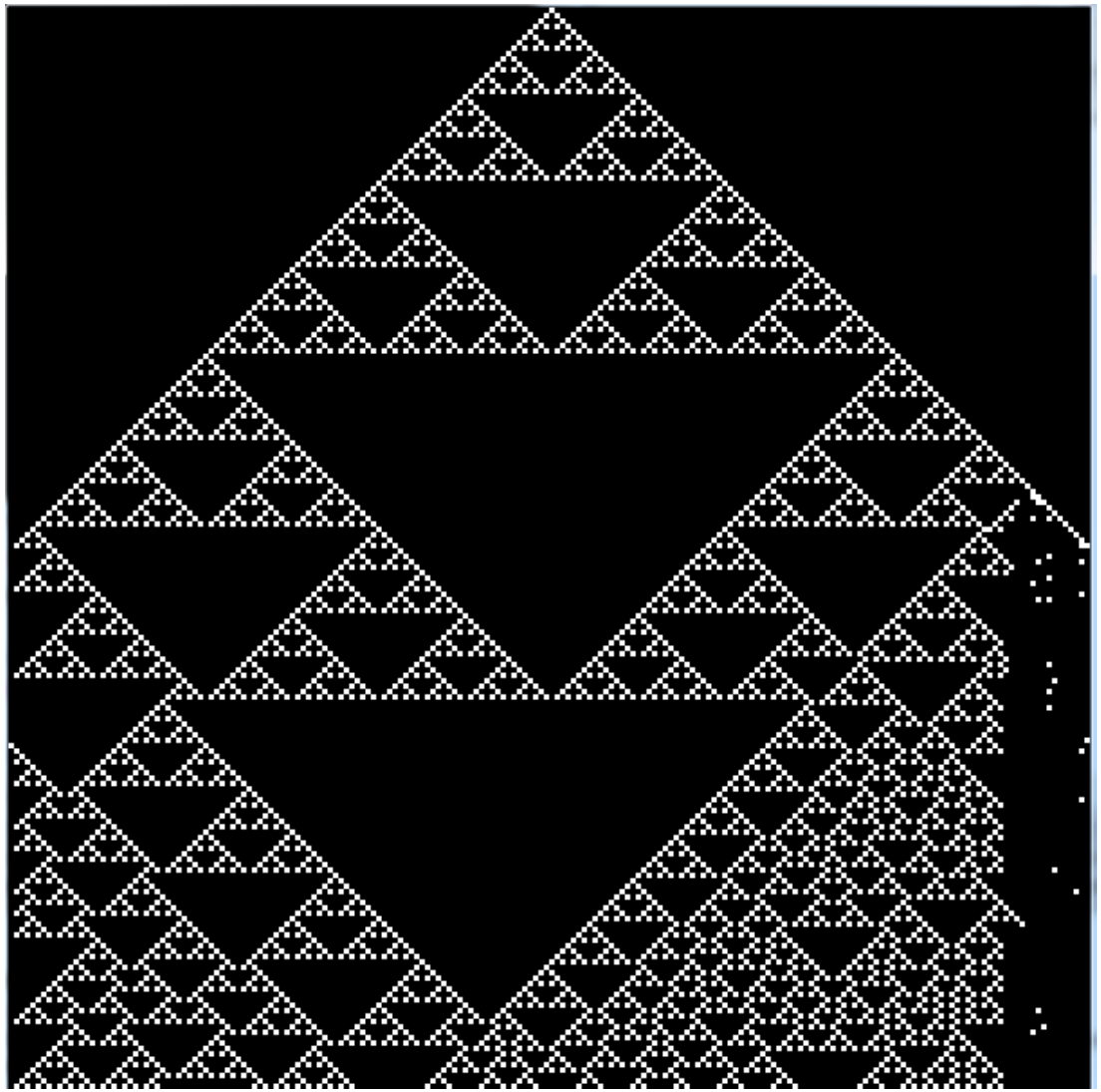


Pętle Langtona

- W 1952 John von Neumann stworzył pierwszy automat komórkowy, który był samoreplikującą się maszyną .
- W 1968 Edgar F. Codd zredukował liczbę stanów komórek z 29 do 8.
- Christopher Langton zbudował jeden z najprostszych automatów tego typu w 1984. Budowane są pętle komórek niosących informację genetyczną, która płynie po pętli tworzy pseudopodia, które odłączają się tworząc kolejne pętle . Geny instruują jak tworzyć pętlę, która się odłącza.



Przykład programu dla automatu 1D



Przykład programu dla automatu 1D

```
1  #include <allegro5/allegro.h>
2  #include <allegro5/allegro_primitives.h>
3  #include <cstdlib>
4  #include <cstdio>
5  #include <cmath>
6  using namespace std;
7
8  int main(int argc, char **argv)
9  {
10  al_init();
11  al_install_keyboard();
12  al_install_mouse();
13  al_init_primitives_addon();
14
15  ALLEGRO_DISPLAY *display=al_create_display(600, 600);
16
17  bool redraw=true;
18  const float FPS=60;
19  ALLEGRO_TIMER *timer=al_create_timer(1.0/FPS);
20  al_start_timer(timer);
```

Przykład programu dla automatu 1D

```
21   ALLEGRO_EVENT_QUEUE *event_queue=al_create_event_queue();
22   al_register_event_source(event_queue,
23  al_get_display_event_source(display));
24   al_register_event_source(event_queue, al_get_timer_event_source(timer));
25
26  const int L=200;
27  int siec[L][L];
28  int regula[8]={0,1,0,1,0,1,1,0};
29
30   while(1)
31   {
32   ALLEGRO_EVENT ev;
33   al_wait_for_event(event_queue,&ev);
34   if(ev.type==ALLEGRO_EVENT_TIMER)
35   {
36   redraw=true;
37   }
38   else if(ev.type==ALLEGRO_EVENT_DISPLAY_CLOSE)
39   {
40   break;
```

Przykład programu dla automatu 1D

```
41      }
42      if(redraw && al_is_event_queue_empty(event_queue))
43      {
44      redraw = false;
45      al_clear_to_color(al_map_rgb(0,0,0));
46
47      for(int j=0;j<L;j++)siec[j][0]=0;
48      siec[100][0]=1;
49
50      for(int j=0;j<L-1;j++)
51      for(int i=0;i<L;i++)
52      {
53          if(siec[i][j]==0&&siec[(i-1+L)%L][j]==0&&siec[(i+1)%L][j]==0)siec[i][j+1]=regula[0];
54          if(siec[i][j]==0&&siec[(i-1+L)%L][j]==0&&siec[(i+1)%L][j]==1)siec[i][j+1]=regula[1];
55          if(siec[i][j]==0&&siec[(i-1+L)%L][j]==1&&siec[(i+1)%L][j]==0)siec[i][j+1]=regula[2];
56          if(siec[i][j]==0&&siec[(i-1+L)%L][j]==1&&siec[(i+1)%L][j]==1)siec[i][j+1]=regula[3];
57          if(siec[i][j]==0&&siec[(i-1+L)%L][j]==0&&siec[(i+1)%L][j]==0)siec[i][j+1]=regula[4];
58          if(siec[i][j]==0&&siec[(i-1+L)%L][j]==0&&siec[(i+1)%L][j]==1)siec[i][j+1]=regula[5];
59          if(siec[i][j]==0&&siec[(i-1+L)%L][j]==1&&siec[(i+1)%L][j]==0)siec[i][j+1]=regula[6];
60          if(siec[i][j]==0&&siec[(i-1+L)%L][j]==1&&siec[(i+1)%L][j]==1)siec[i][j+1]=regula[7];
```

Przykład programu dla automatu 1D

```
61      }
62      for(int j=0;j<L;j++)
63      for(int i=0;i<L;i++)
64      {
65          if(siec[i][j]==1)al_draw_filled_rectangle(3*i,3*j,3*i+3,3*j+3,al_map_rgb(255,255,255));
66      }
67
68      al_flip_display();
69      }
70      }
71      al_destroy_timer(timer);
72      al_destroy_display(display);
73      al_destroy_event_queue(event_queue);
74
75      return 0;
76  }
```