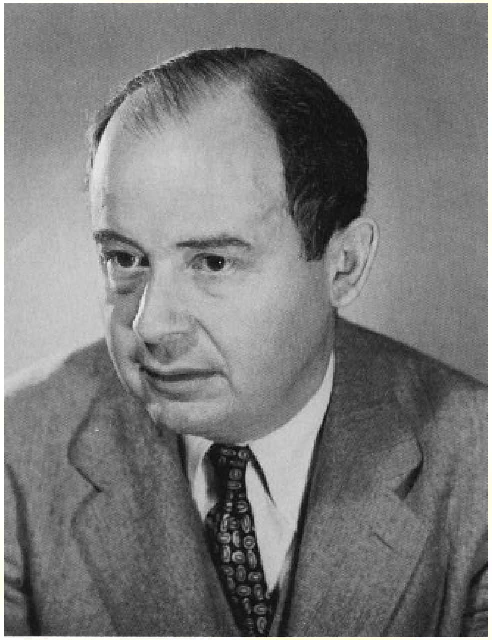
Celulárne automaty

Celulárne automaty

- CA umožňujú simulovať deje, ktoré prebiehajú v v okolitom prostredí
- Namiesto opisu komplexného systému zložitými rovnicami simulujú chovanie tohto systému vzájomným pôsobením buniek správajúcich sa podľa určitých pravidiel
- Komplexné a zdanlivo náhodné procesy sú simulované pomocou malej množiny jednoduchých pravidiel

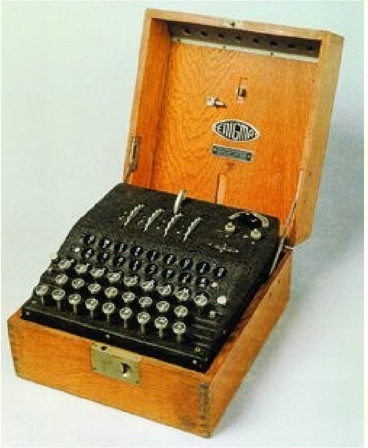
História



- CA boli vynálezom Johna von Neumanna okolo r. 1950 simulovať prácu komponentov počítačom a mechanických zariadení pomocou buniek s pravidlami. Pôvodný návrh nakoniec zjednodušil za pomoci Stana Ulama
- Von Neumanna fascinovala schopnosť živých organizmov sebareprodukcie a snažil sa ju napodobniť v počítačovom svete
- Neskôr sa CA venovali Burke a Holland (genet. algoritmy)
- Vďaka Conwayovi (Game of Life) sa CA stali populárne

Svet pred CA

- Pred von Neumannovými CA existovala myšlienka teoretického počítačového modelu v podobe Turingového stroja
- Turingov stroj bol pomenovaný po Alanovi Turingovi, britskom matematikovi, ktorý pomohol rozlúštiť tajné nemecké kódovanie „Enigma“ v druhej svetovej vojne



Stephen Wolfram

- Jeden z kľúčových výskumníkov správania sa 1D CA
- Prvý článok vydal v 15-tich rokoch a PhD. ukončil v 20-tich, pracoval v Caltechu a Princetone
- Vlastní spoločnosť Mathematica (Wolfram Research)
- Dokázal, že pri použití určitých obmedzujúcich podmienok sa CA chová v súlade s Navierovymi-Stokesovymi rovnicami, t.j. rovnicami opisujúcimi prúdenie tekutiny => CA sa dajú využiť k modelovaniu fyzikálnych procesov

Definícia CA

- Dynamické systémy, ktoré sú diskrétne v priestore a čase, pracujú na pravidelnej *n*-dimenzionalnej mriežke a ich chovanie je určené lokálnymi interakciami.
- Celulárny automat je matematický model určitého systému, ktorého priestor a čas sú diskrétne, fyzikálne veličiny majú diskrétne hodnoty z konečnej množiny hodnôt.
- Každá bunka má určitý diskrétny stav a riadi sa danými pravidlami opisujúcimi stav buniek v ďalšom časovom kroku, v závislosti na stavoch buniek v okolí.
- Stav systému v nasledujúcom časovom kroku závisí na stavu systému v predchádzajúcom kroku a lokálne aplikovaných pravidlách

Vlastnosti CA

- **Paralelizmus** – výpočty vo všetkých bunkách prebiehajú súčasne, nie postupne
- **Lokalita** – nový stav závisí na aktuálnom stave bunky a jej susedov
- **Homogenita** – všetky bunky používajú rovnakú lokálnu prechodovú funkciu => rovnaké pravidlá pre všetky bunky

Prvky CA

- Doména (*D*) = mriežkový priestor (aj nekonečný) buniek
  - 1D: riadok buniek, obruč
  - 2D: obdĺžnik, torus, …
  - 3D: objem
- Susedia bunky (*N*) – Neumanovské vs. Moorovské okolie
- Stav bunky (*S*) je určený hodnotou
- Pravidlá pre bunky (*f*)
- Počiatočný stav (*I*)

$$CA = (D; N; S; f; I)$$

Pravidlá chovania CA

- Každá bunka reaguje iba na stav svojho okolia
- Pravidlá sú vyjadrené logickými, numerickými, … operáciami
- Sú často odvodené empiricky
- Kategórie pravidiel podľa správania CA:
  - Stabilné vzory v celom systéme
  - Stabilné po častiach (s periodickými štruktúrami)
  - Chaotický náhodný vzor
  - Samoregulácia (sebapropagácia a sebareplikácia)

Využitie CA

- Spracovanie obrazu - klasifikácia obrazu (filtre, SIFT, ...)
- Simulácie:
  - prúdenia kvapalín
  - vývoja populácie
  - ekologických modelov (znečistenie, invázia druhov, ...)
  - dopravy
  - požiarov
  - rastu vegetácie (rozšírenie druhov rastlín)
  - vývoja miest (urbanizmus)
  - šírenia olejových škvŕn

Ďalšie, teoretické oblasti využitia CA

- Multiagentné systémy
- Teória chaosu
- Umelá inteligencia (neuronové siete)
- Fraktálna geometria
- Expertné systémy (rozhodovanie, predpovede)

Porovnanie CA a Turing. stroja

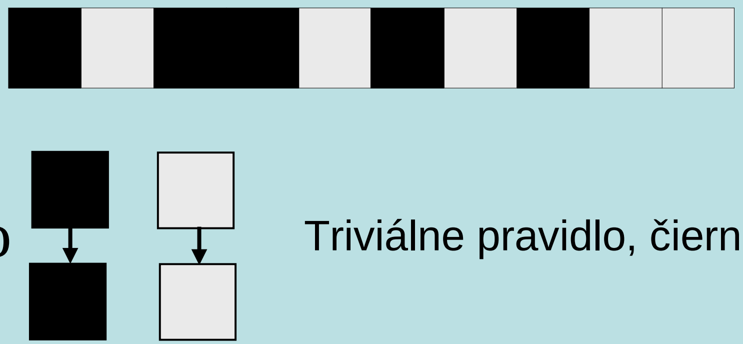
- TS nemôže meniť sám seba (nie je dostatočne univerzálny)
- TS nemôže vytvoriť svoje kópie, ktoré budú simulovať TS
- Neodzrkadľujú dobre chovanie komplexných systémov, pretože nemajú spätnú väzbu, vďaka ktorej by menili svoje chovanie
- CA majú možnosť meniť sami seba
- TS striktne oddeľujú program a dáta

Najjednoduchší CA

- 1D, riadok binárnych buniek
- Susedia bunky = bunka naľavo a napravo

Riadok

Pravidlo



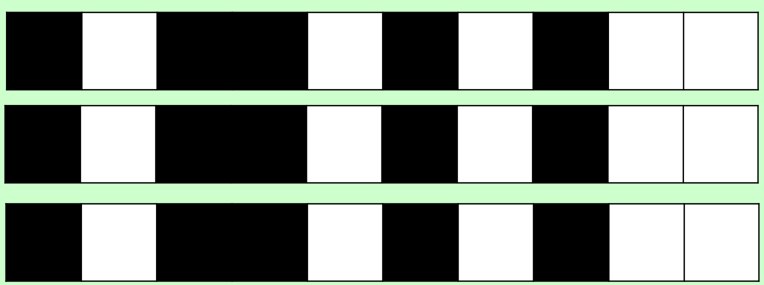
Daný je počiatočný stav

Triviálne pravidlo, čierna ostáva čiernou a biela bielou

Čas 0

Čas 1

Čas 2



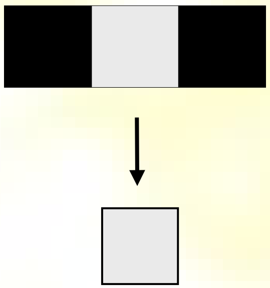
Vývoj systému CA v čase 0, 1, 2

1D Binárne celulárne automaty

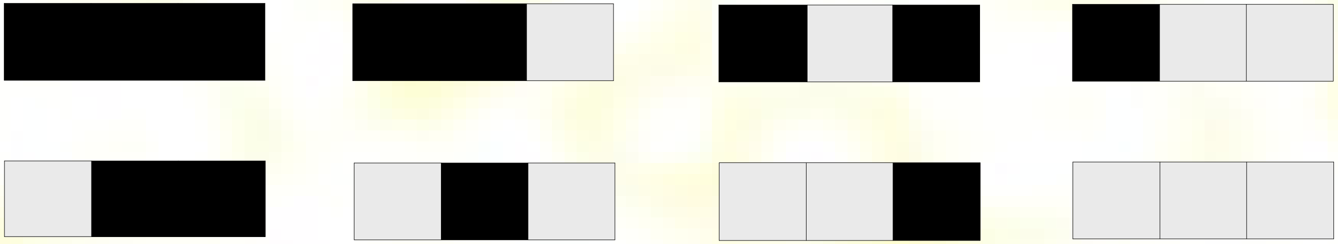
- Majú iba 2 stavy (true = 1, false = 0)
- Pre jednobunkových susedov máme
  - 2^{2^3} možných programov/pravidiel (bunka + susedia)
  - slabá priestorová závislosť a málo možných pravidiel
- Dvojbunkový susedia (z každej strany dvaja)
  - 2^{2^5} možných pravidiel
  - základná priestorová závislosť
- Trojbunkový susedia
  - dostatočná priestorová závislosť
  - najčastejšie používaný pre tvorbu vzorov

Wolframov 1D binárny CA

- Príklad pravidla



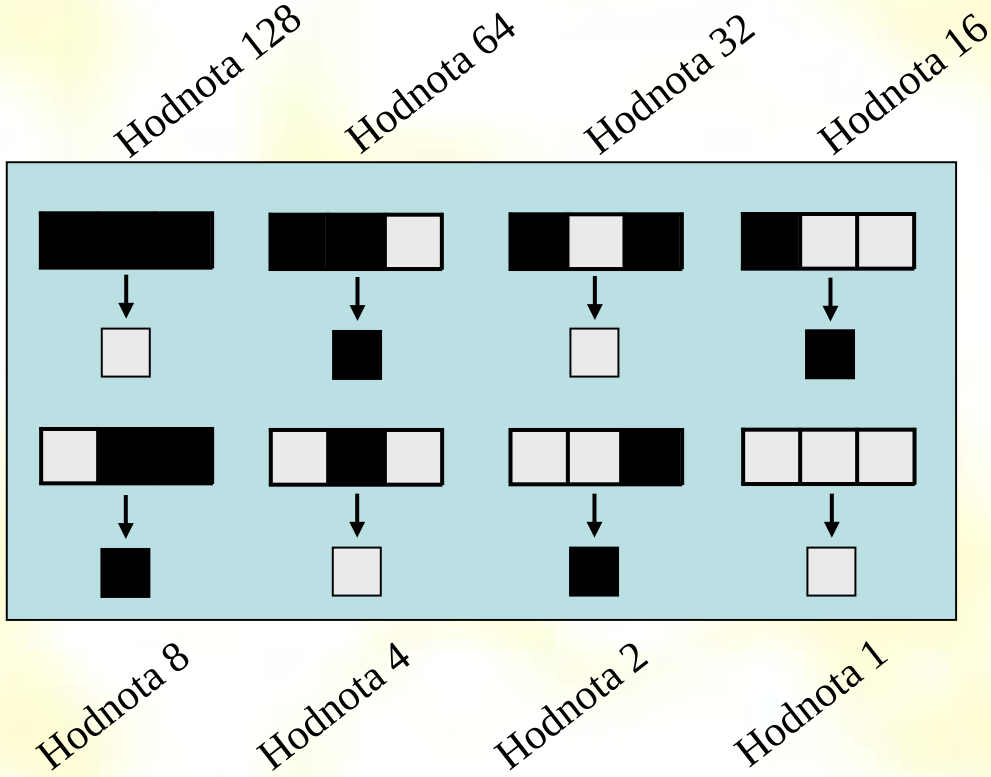
- Možné kombinácie



- 2^8 možných pravidiel (programov)

Wolframovo číslovanie

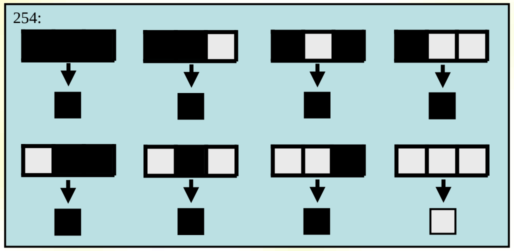
- Pravidlo = celé číslo od 0 do 255
- Príklad: pravidlo 90 (01011010_2)



$= 2+8+16+64 = 90$

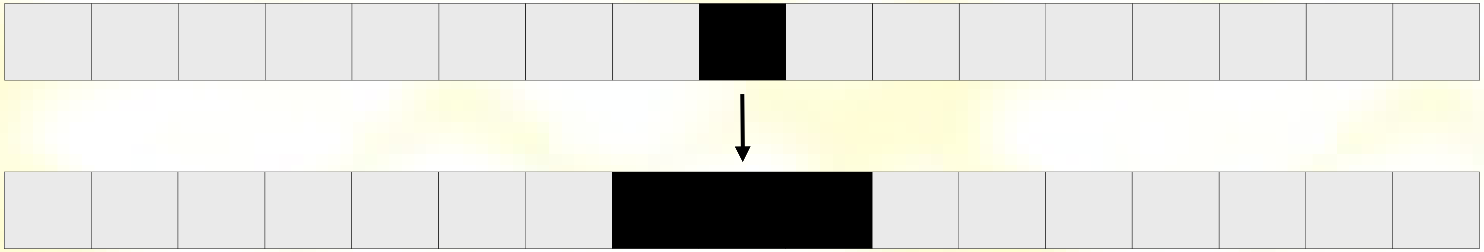
Pravidlo 254

- Pravidlo 254 (11111110_2):

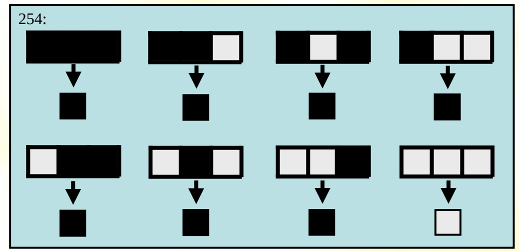


- Počiatočný stav:

- Aplikovanie pravidla 254:



Pravidlo 254 vývoj

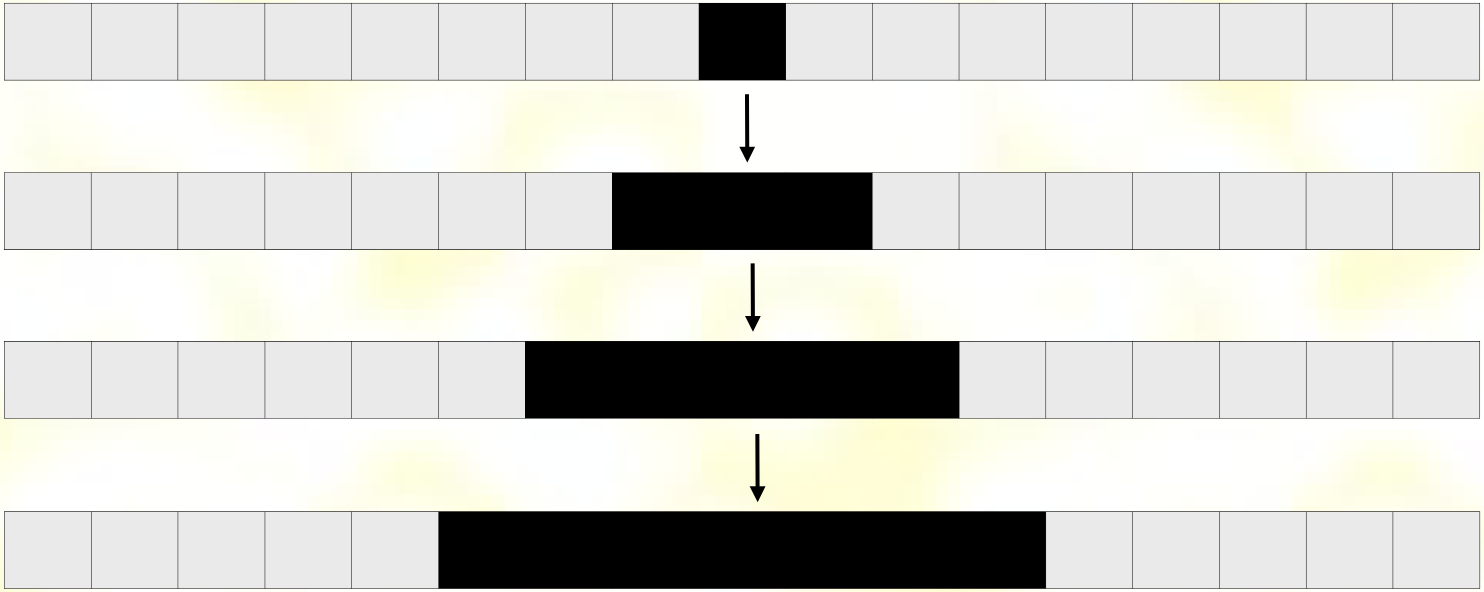


Čas 0

Čas 1

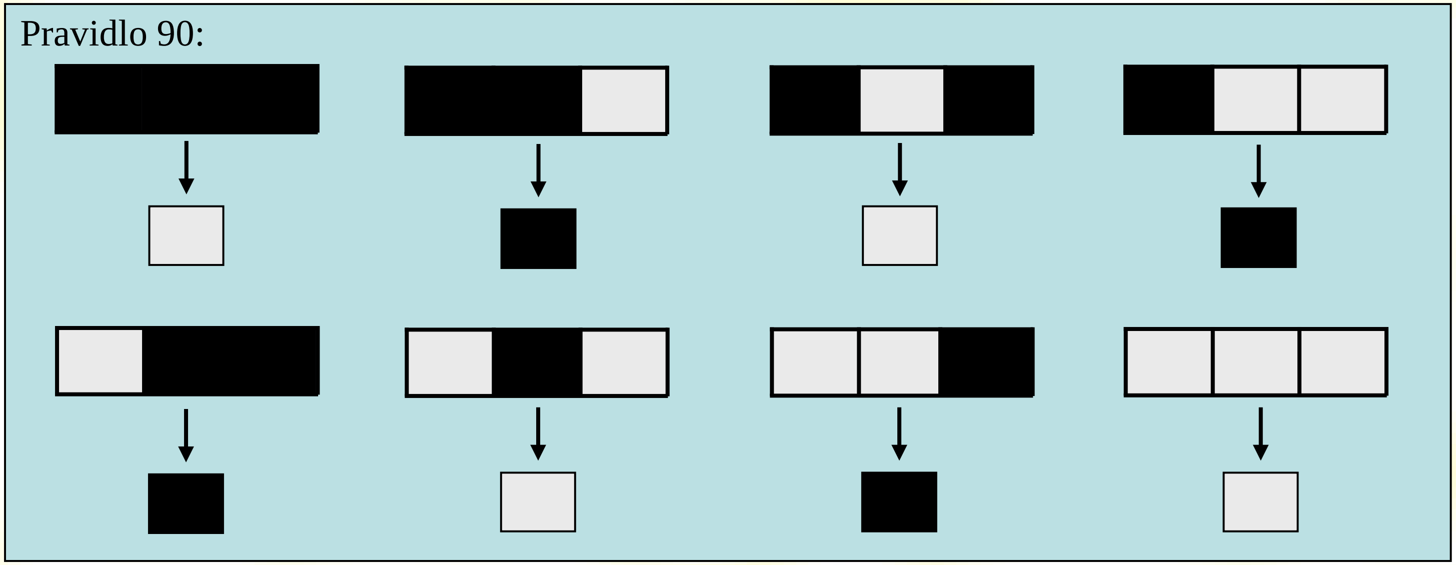
Čas 2

Čas 3

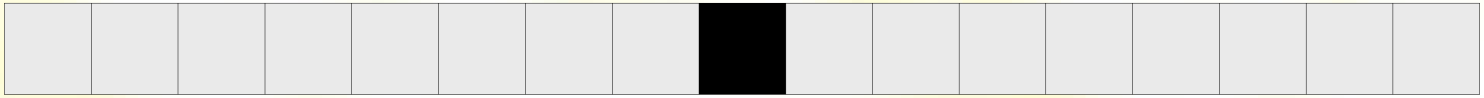


Pravidlo 90

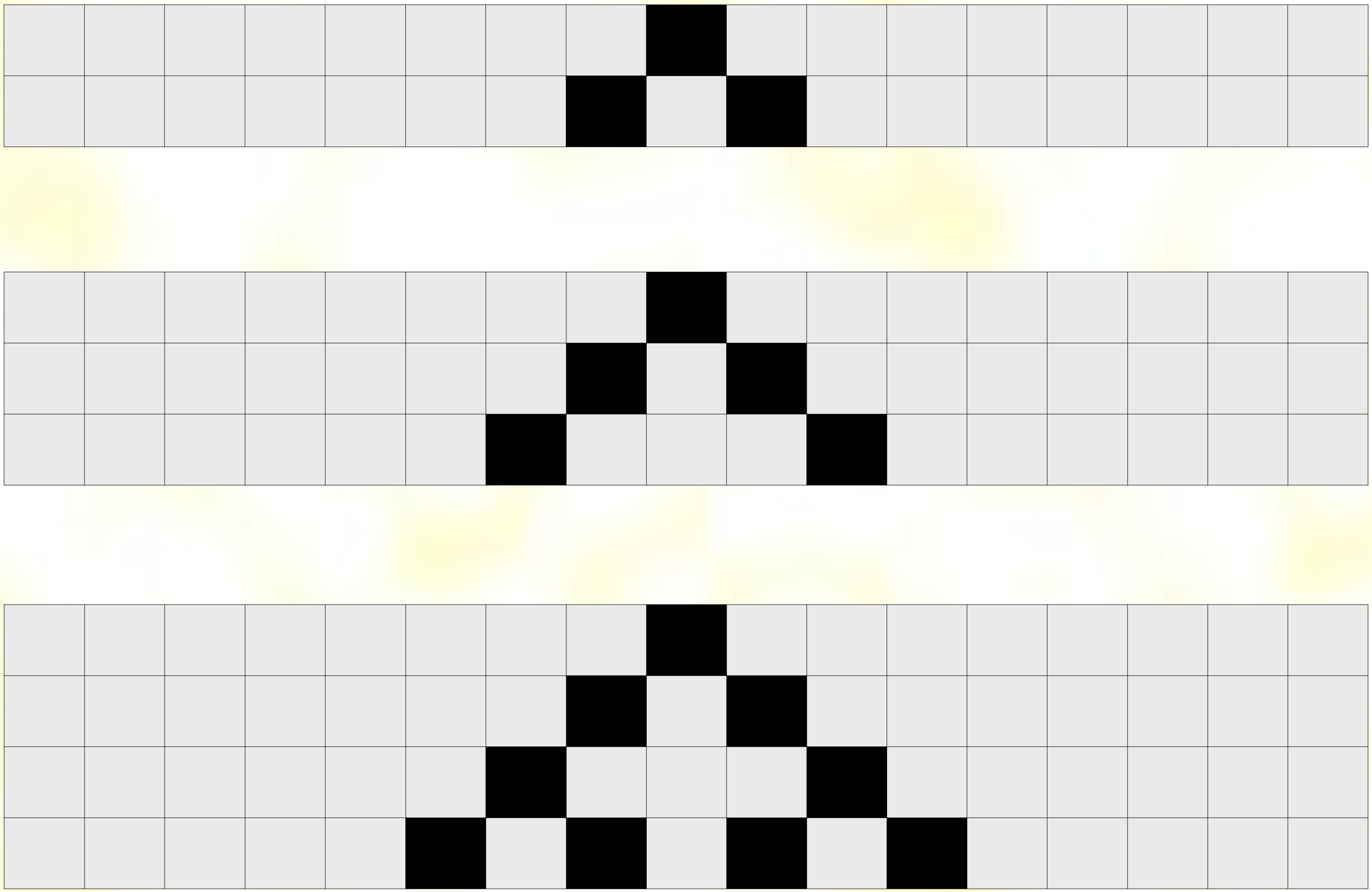
- $2+8+16+64 = 90 \ (01011010_2)$



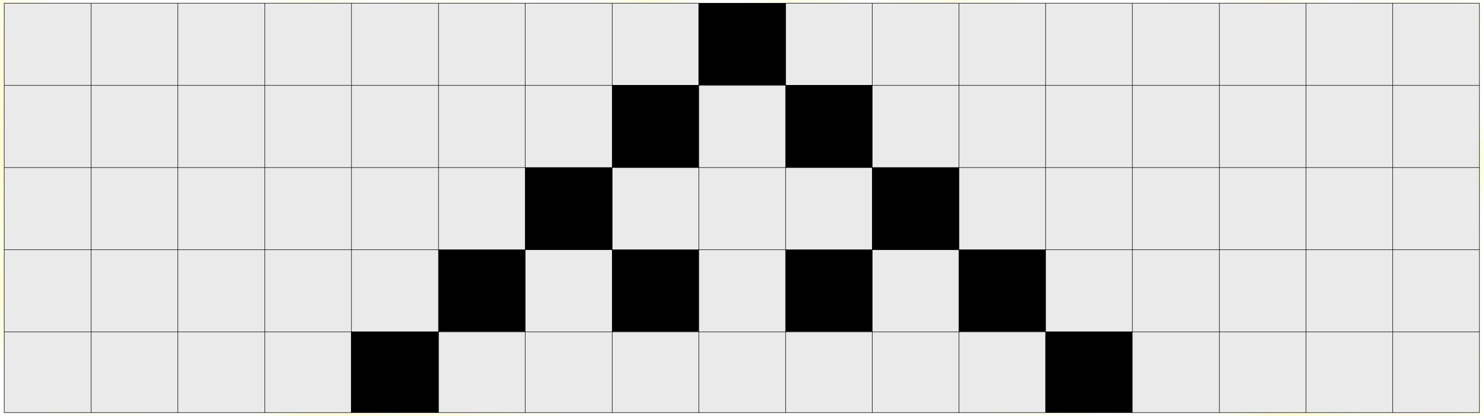
- Počiatočný stav

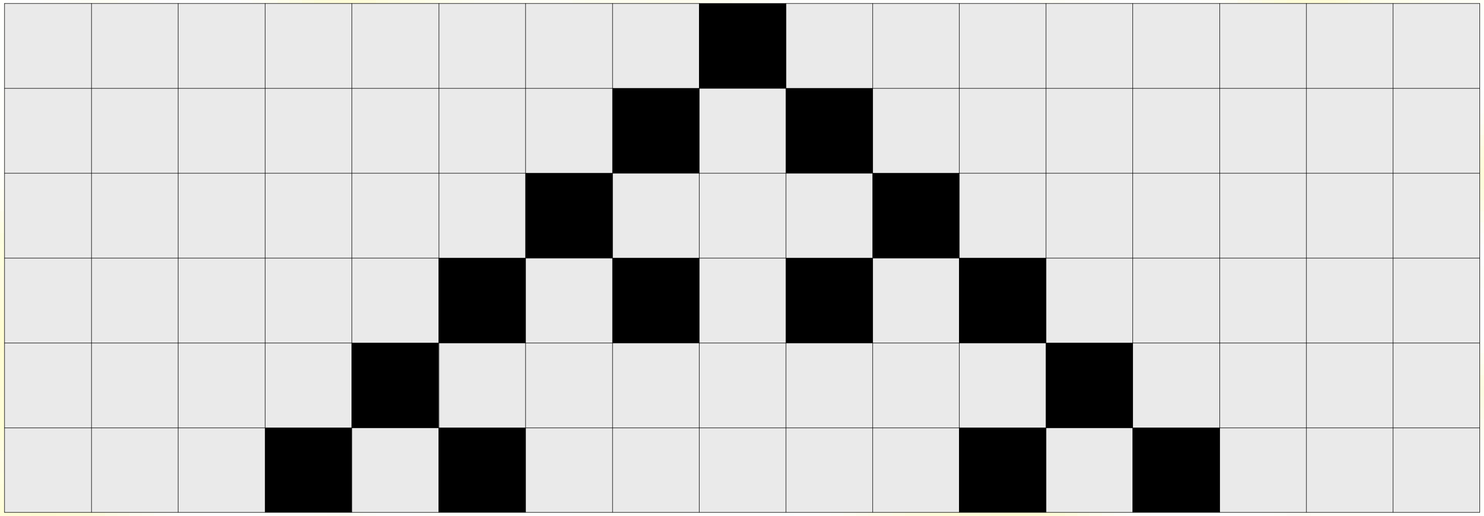


Už tušíte vzor?

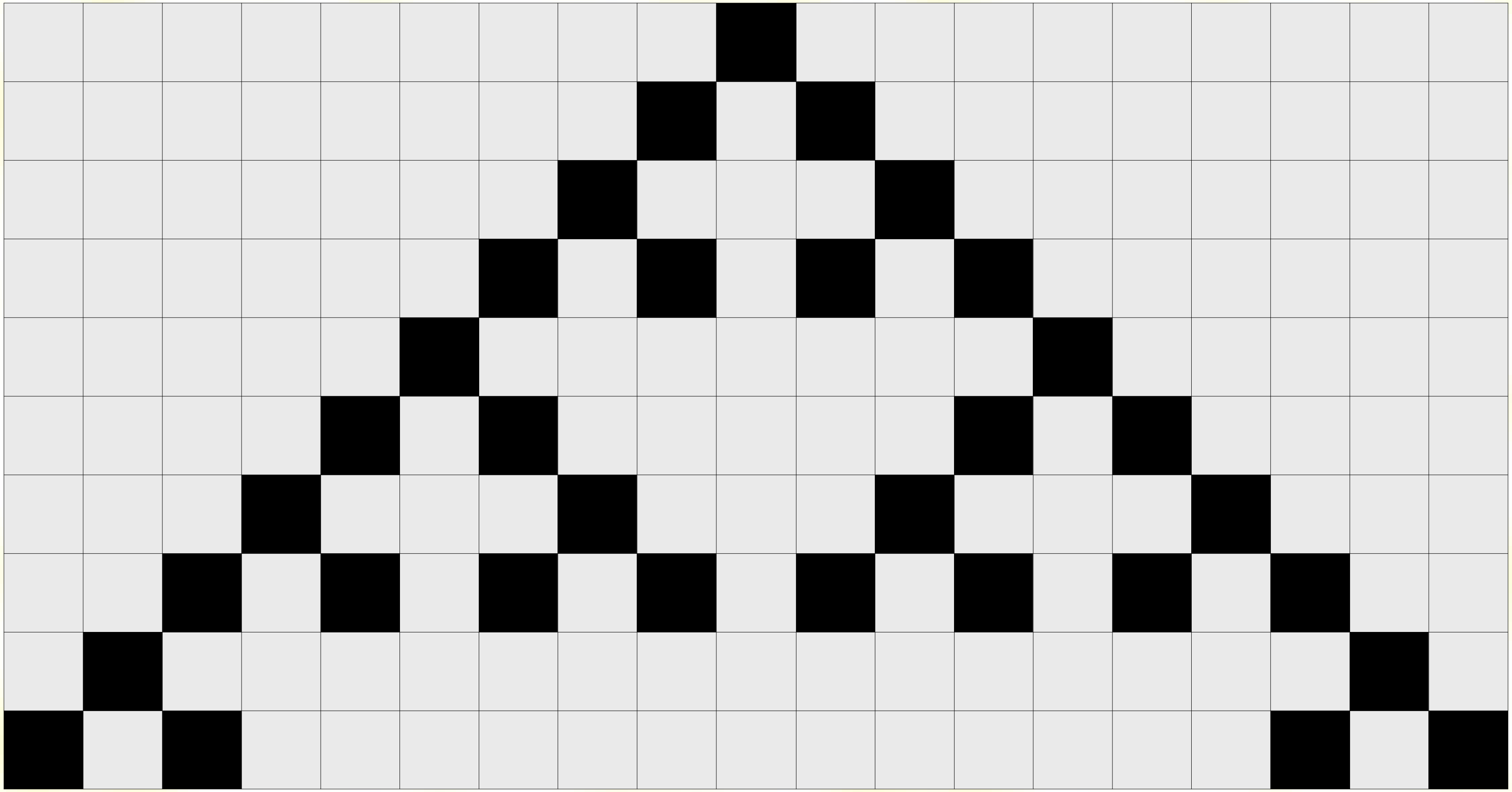


Ešte nie?



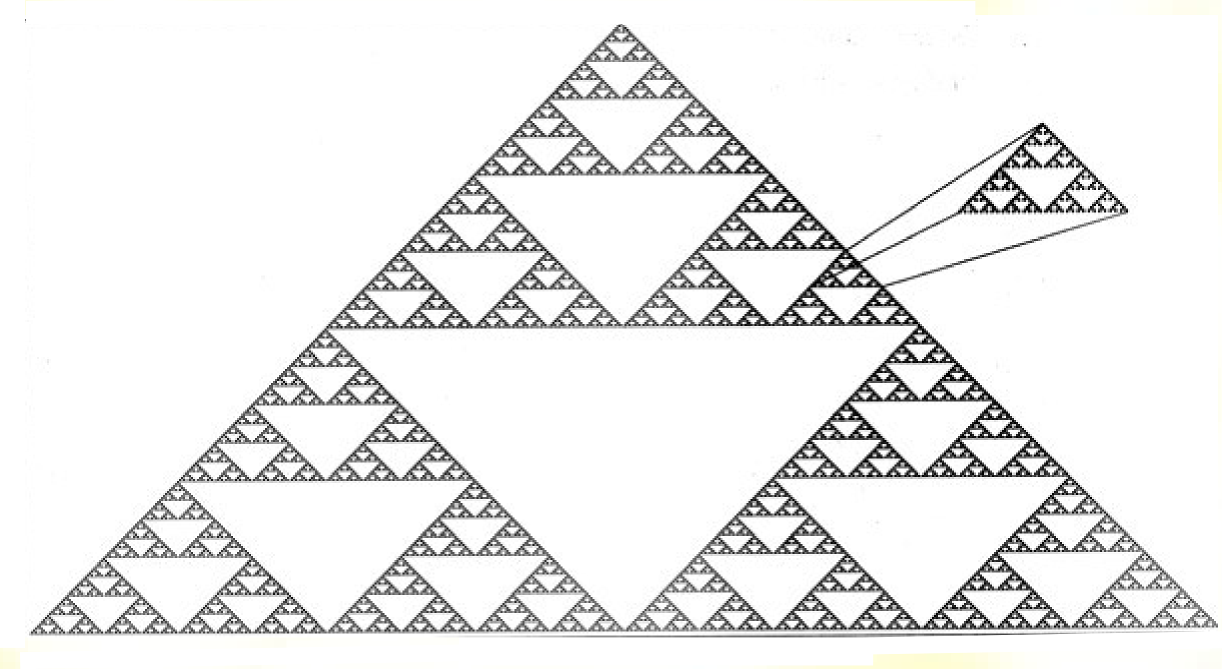


Po 9-tich krokoch...



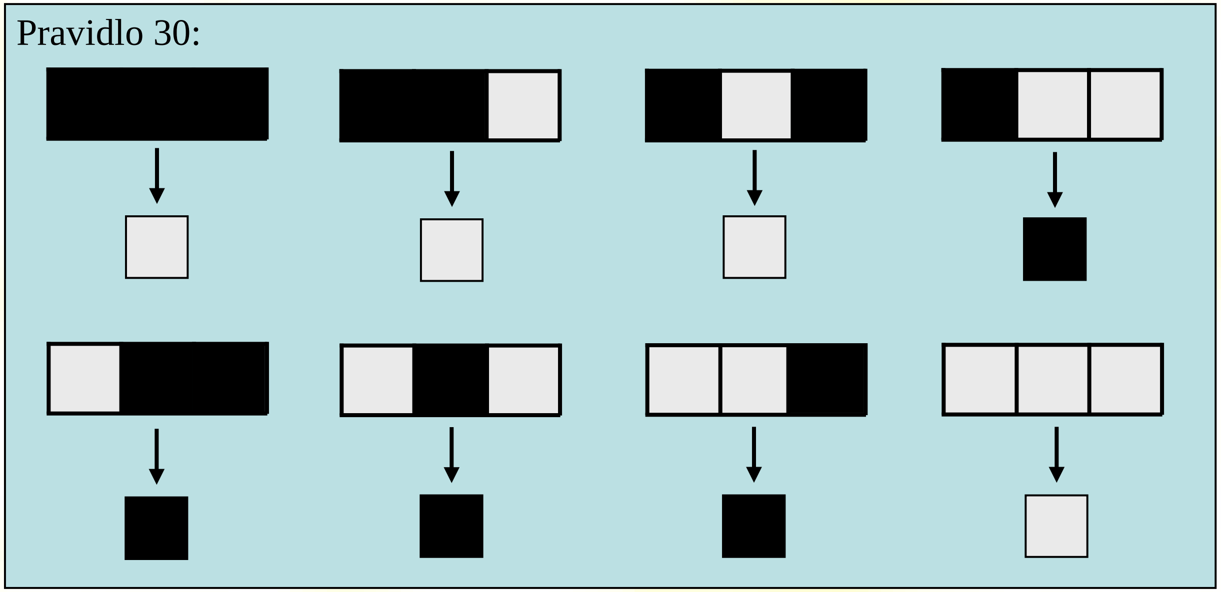
Pravidlo 90 v „nekonečnom čase“

- Sierpinskeho koberec

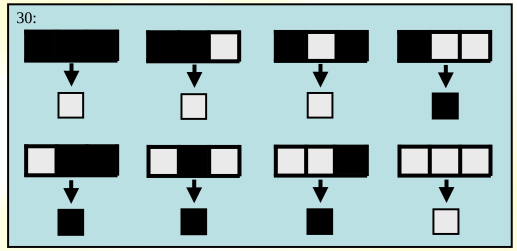


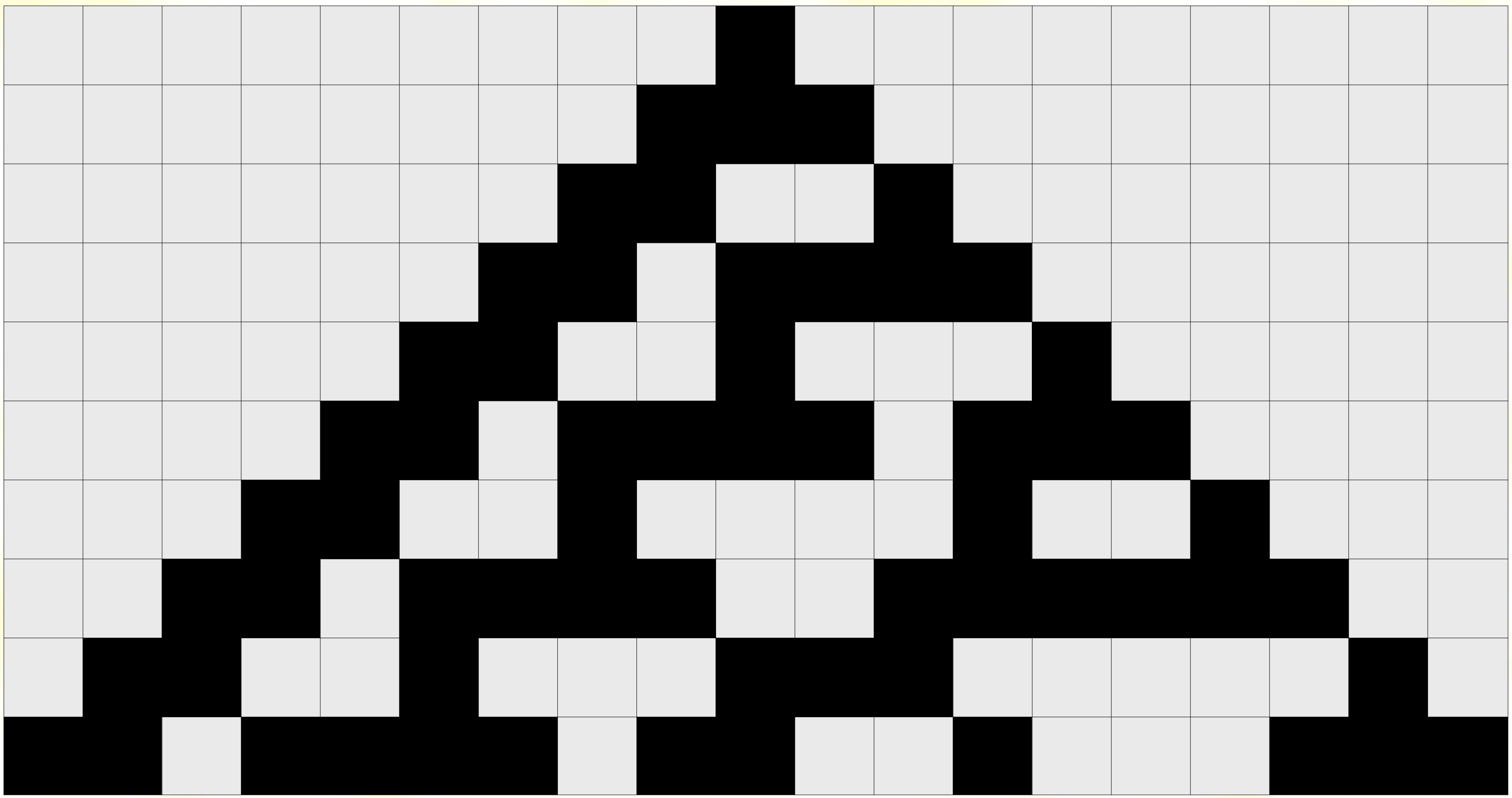
Idú vytvoriť aj nepravidelné?

- Pravidlo 30 (00011110_2)

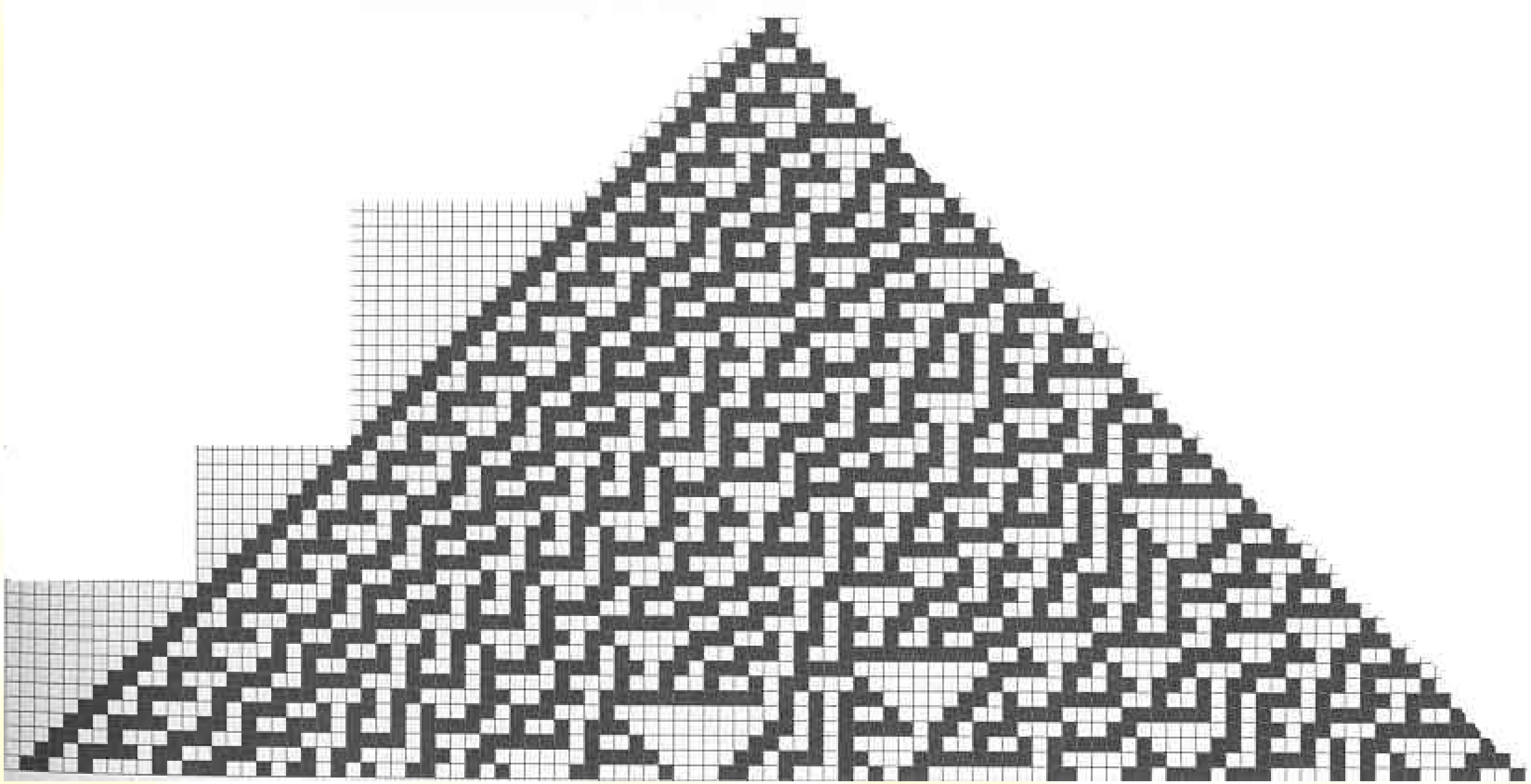


Aplikovanie pravidla 30

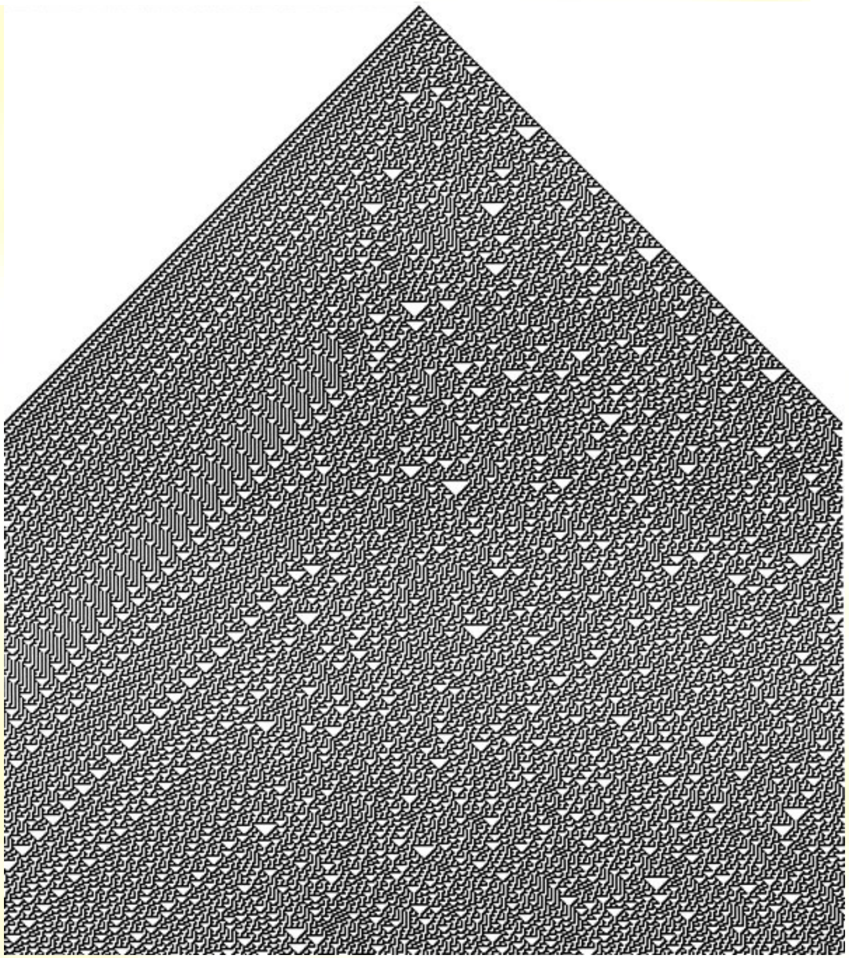




Vývoj pravidla 30

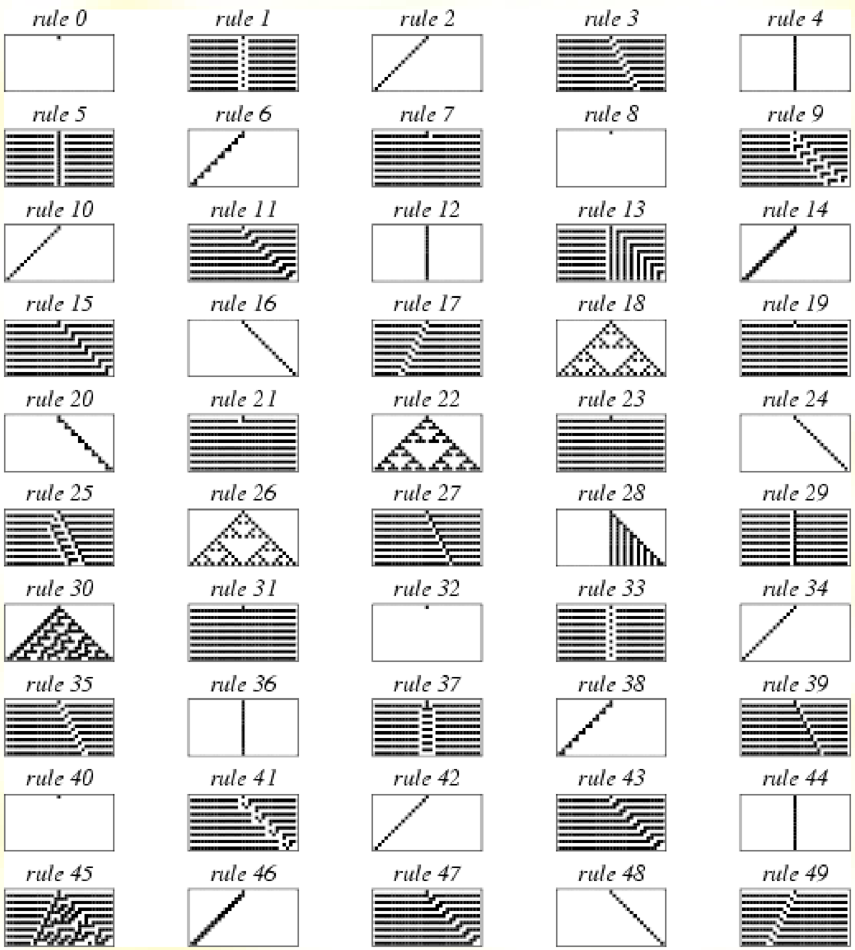


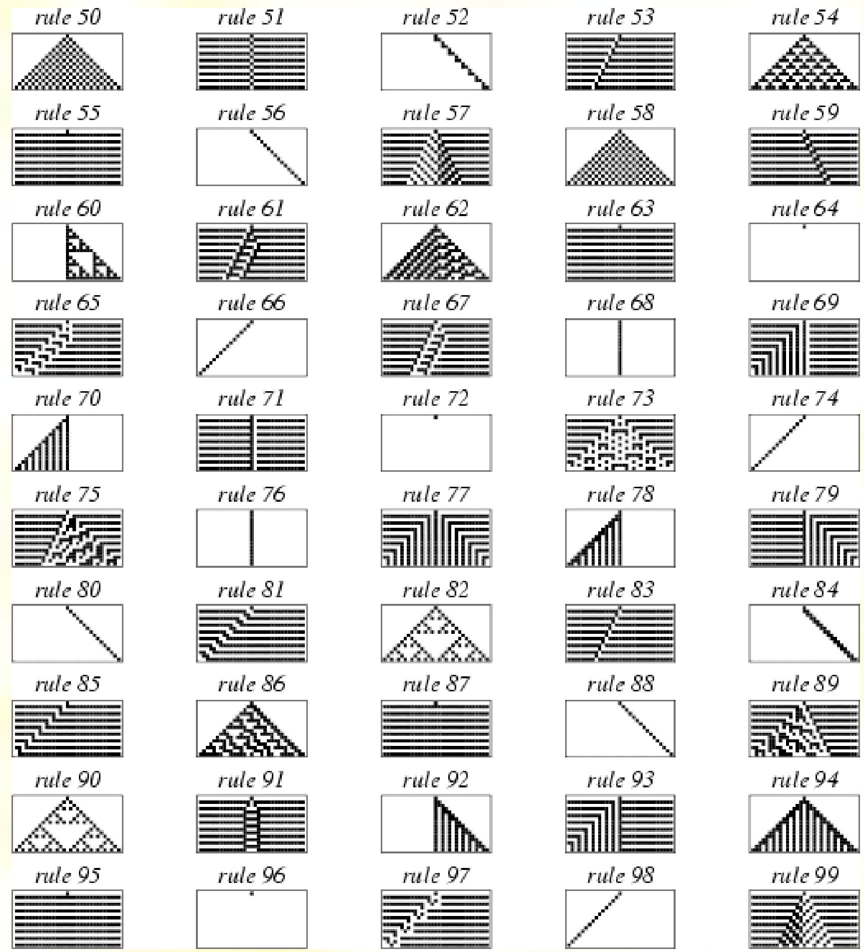
Pravidlo 30 v nekonečnom čase



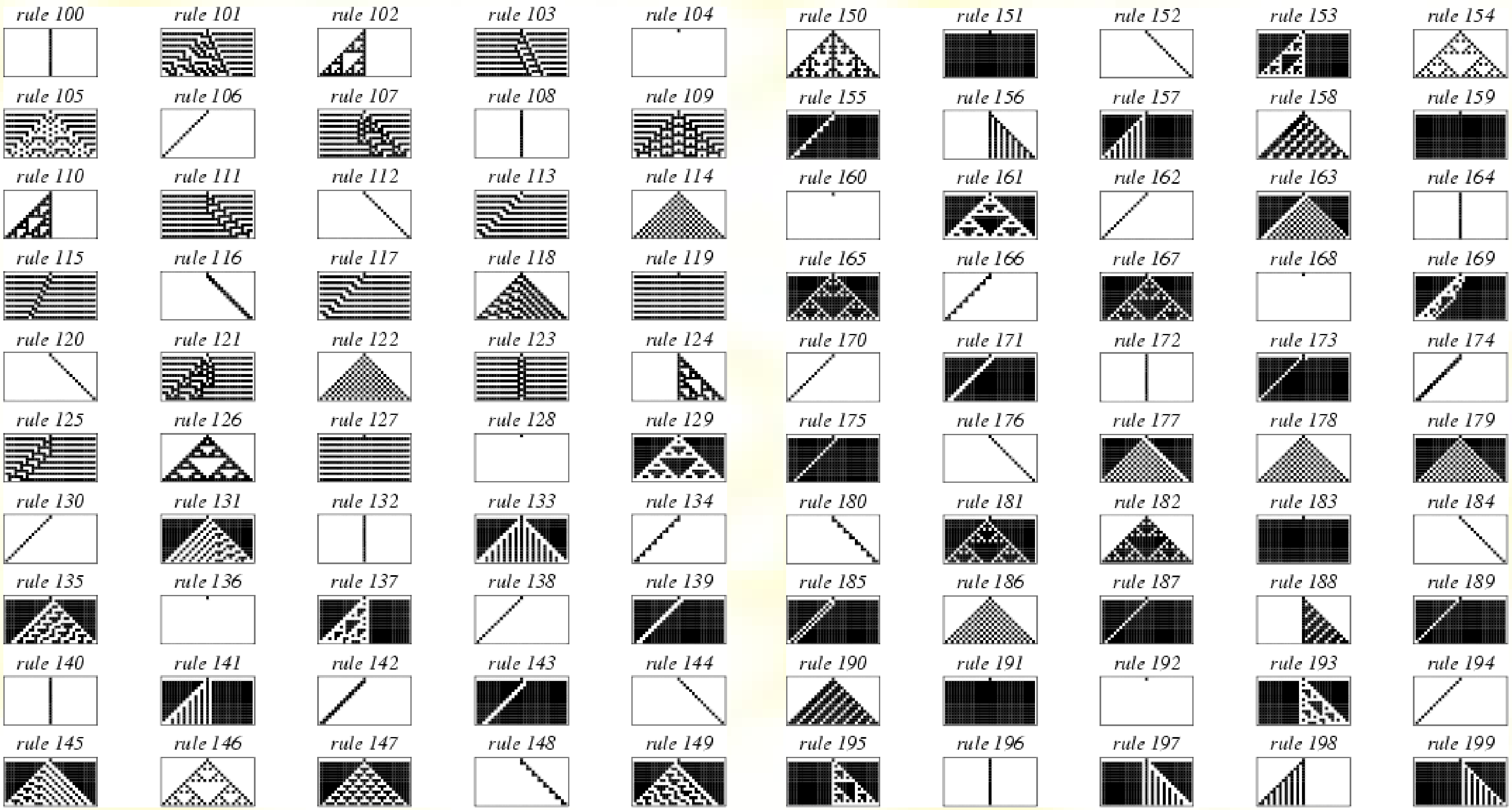
Na jednej strane sa vytvára opakovaný vzor na druhej strane je náhoda.

Atlas pravidiel (1)





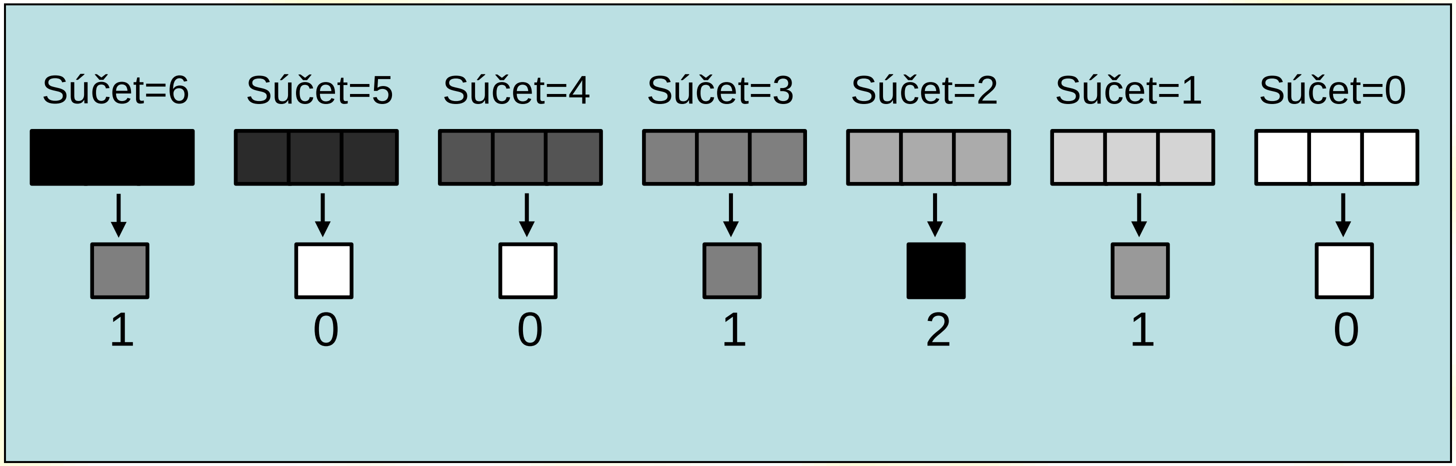
Atlas pravidiel (2)



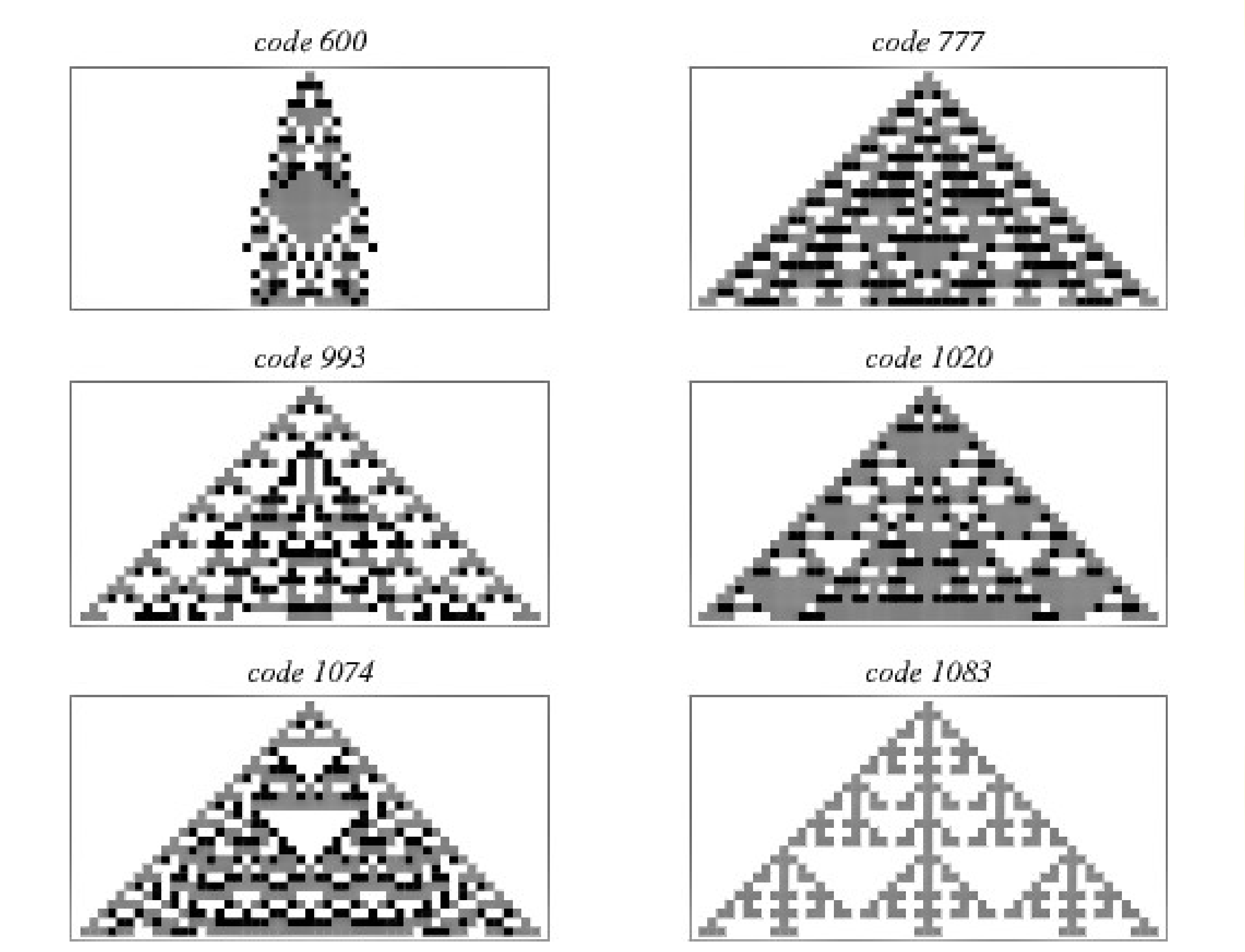
1D *k*-stavový – totalistic CA

- Majú *k*-stavov (farieb), 1 sused vľavo a 1 vpravo
- Hodnoty trojice sa spriemerujú => 3(*k*-1)+1 možností pretože súčet je min. = 0 a max. = 3(*k*-1)
  - k^{3k-2} možných programov/pravidiel

Príklad pre *k* = 3, pravidlo 777 = 1001210_3

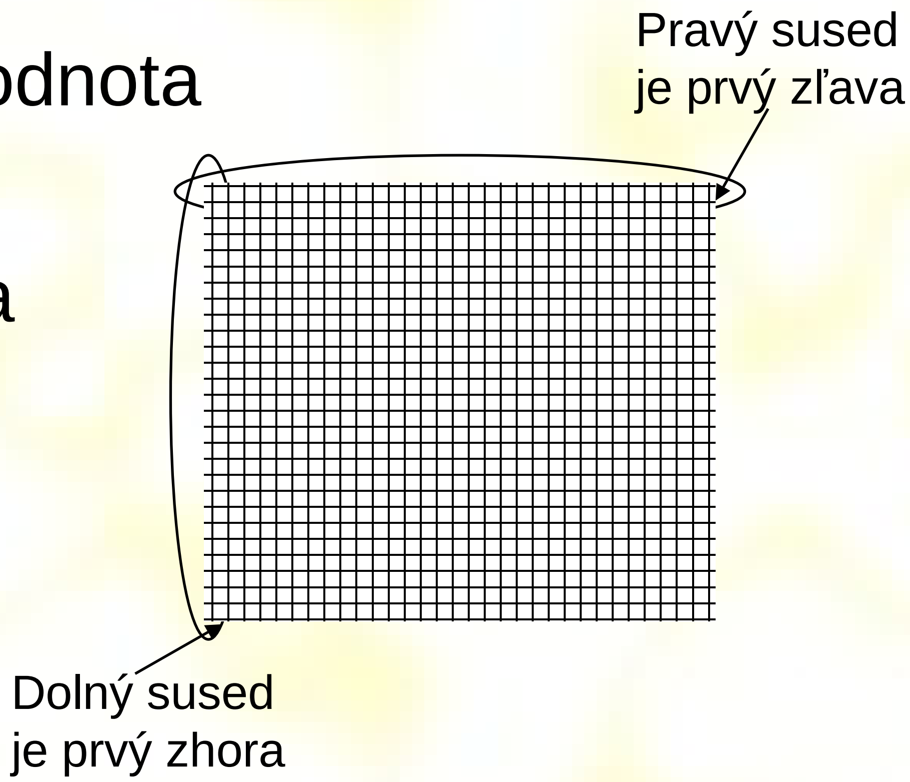


1D *k*-stavový – totalistic CA



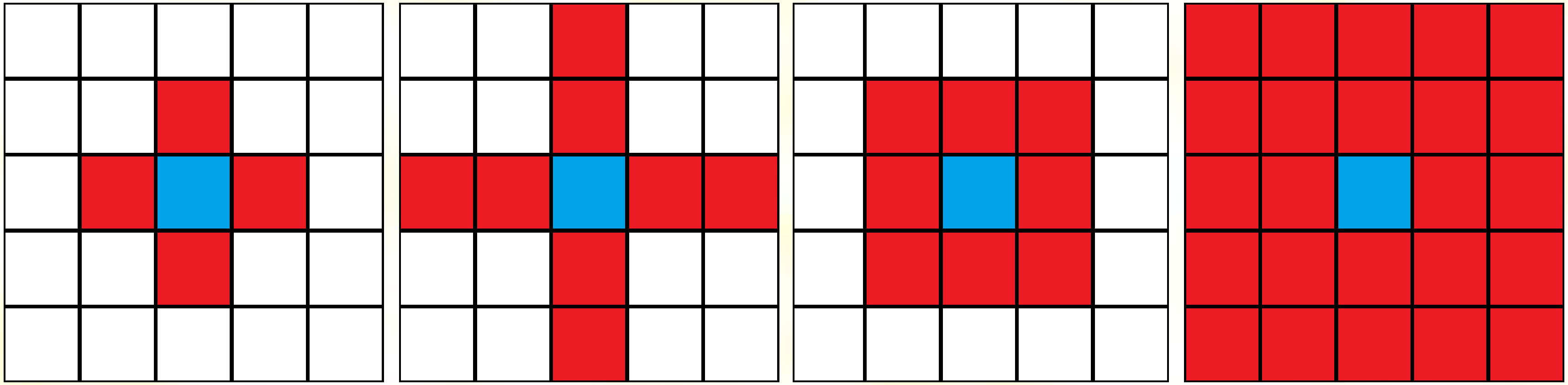
Okrajové podmienky CA

- Pre bunky na okraji neexistujú niektorý susedia, preto sa zvyčajne použije nejaká „náhradná hodnota“:
  - – pevne zvolená hodnota
  - – posledná krajná hodnota
  - – zrkadlová hodnota
  - – periodická hodnota (v 2D je to torus)



2D Binárne CA

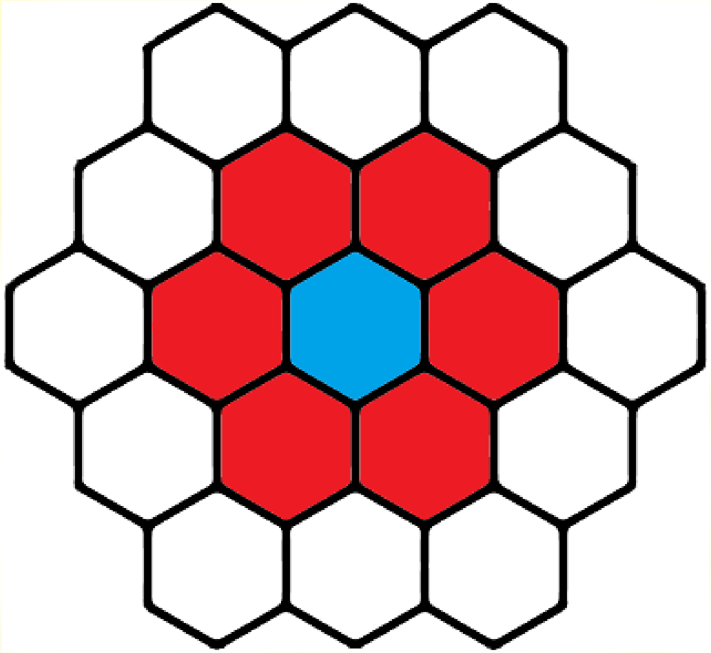
- Doména = bitmapa $m \times n$
- Susedia
  - bunka a 4 (Von Neumann) alebo 8 susedov (Moore)
  - rozšírené okolie
- $2^{2\text{^}5}$ možných pravidiel pre 4-susednosť



Neumannovske okolie | Rozšírené Neumannovske okolie | Moorove okolie | Rozšírené Moorove okolie

2D Binárne CA

- Doména = šesťuholníková mriežka
- Susedia
  - bunka a 6 susedov
  - rozšírené okolie
- 2^{2^7} možných pravidiel
- Uplatnenie je napr. v FHP modeli Lattice Gas CA pre simulovanie prúdenia kvapalín



Najznámejším zástupcom CA je Conwayova „Game of Life“

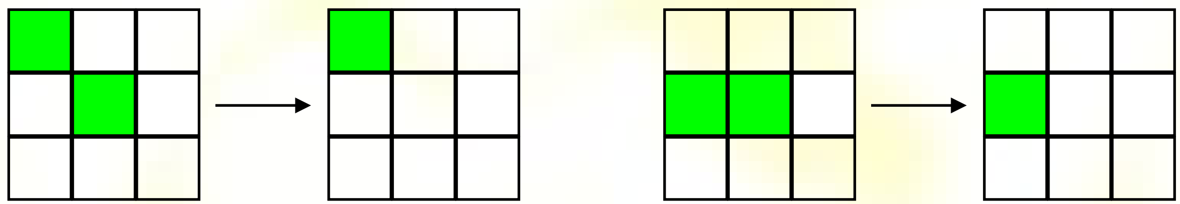
John H. Conway

Pravidlá Game of Life

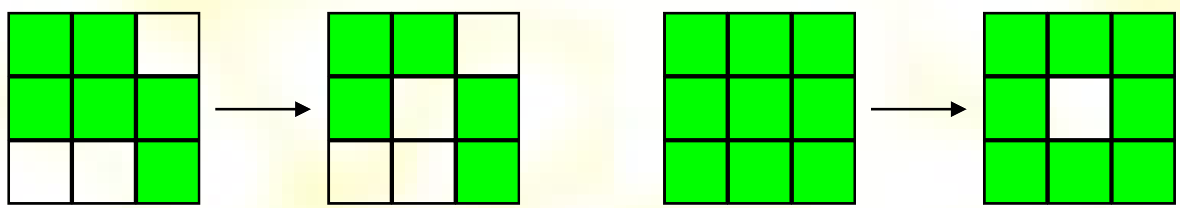
- V každom kroku bude bunka živá alebo mŕtva
- Živá bunka = **1**, mŕtva bunka = **0**
- Pár jednoduchých pravidiel
  - zomrie ak #živých susedov < 2 (osamotená)
  - zomrie ak #živých susedov > 3 (preľudnená)
  - prežije ak #živých susedov = 2,3 (zachovávajúca)
  - narodí ak #živých susedov = 3 (plodiaca)

Príklady pravidiel

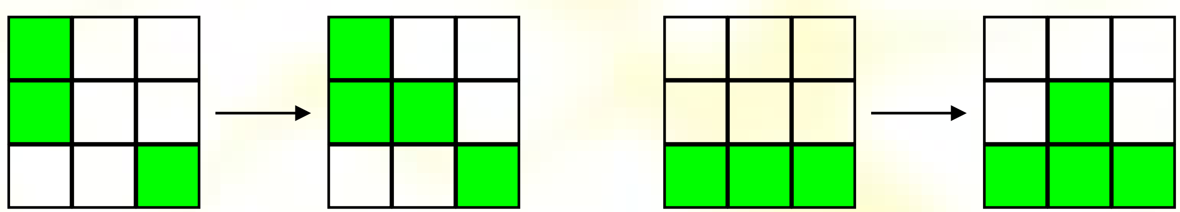
- **osamotená** (zomrie ak #živých < 2)



- **preľudnená** (zomrie ak #živých > 3)

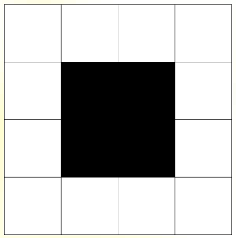


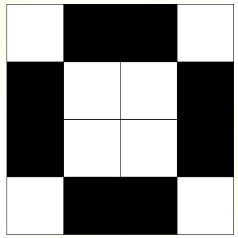
- **plodiaca** (narodí ak #živých = 3)

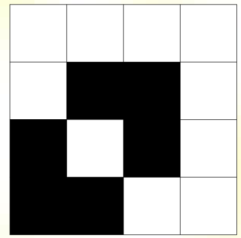


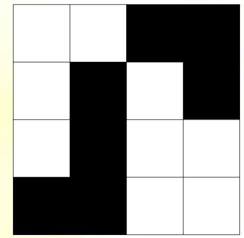
Príklady tvarov

Stabilný

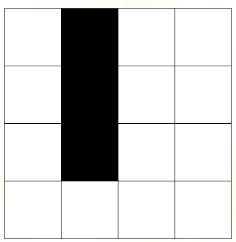

block

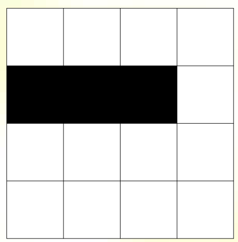

pond


ship

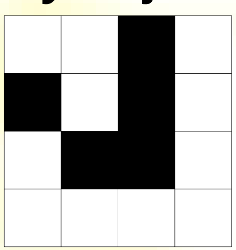

eater

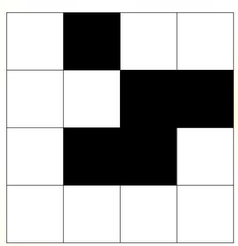
Periodický

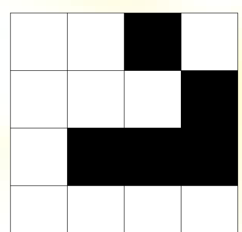

time=0

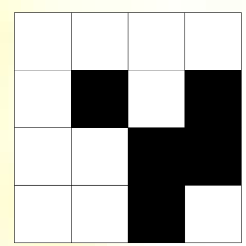

time=1

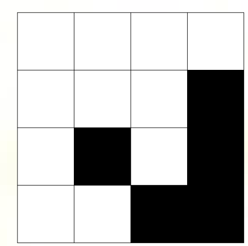
Pohybujúci


time=0


time=1

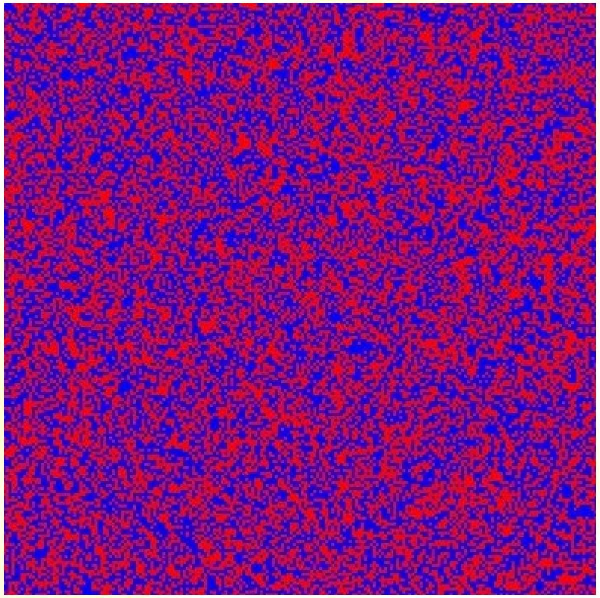

time=2


time=3

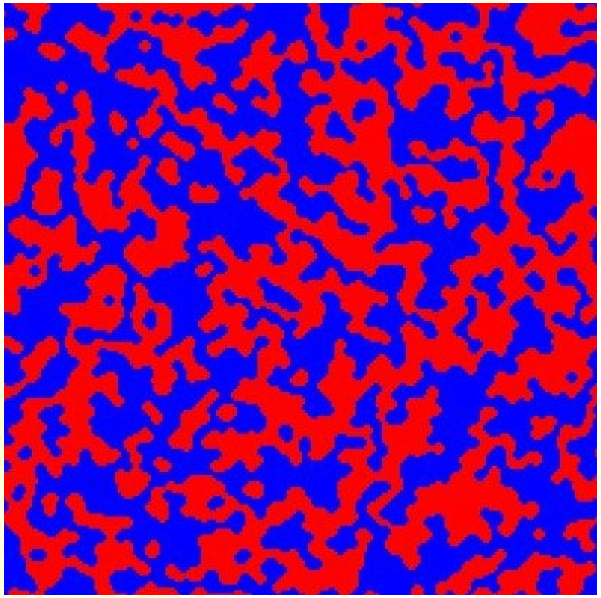

time=4

Pravidlo väčšiny

- **1** ak 5 alebo viac Mooreovych susedov sú 1,
- **0** ak menej ako 5 Mooreovych susedov sú 0
- Počiatočný stav: biely šum

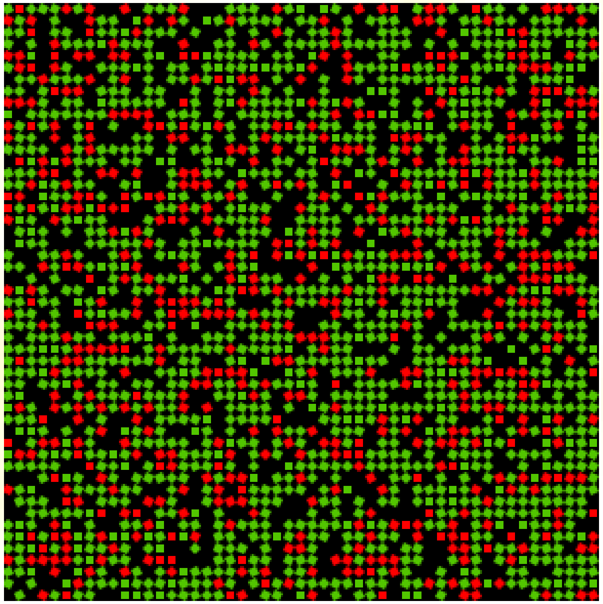


→

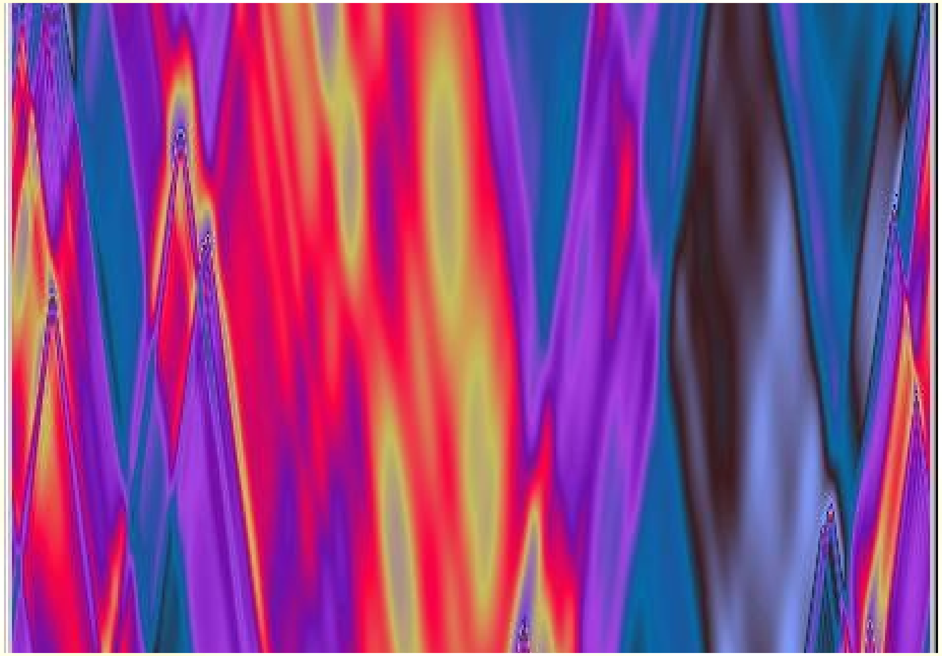


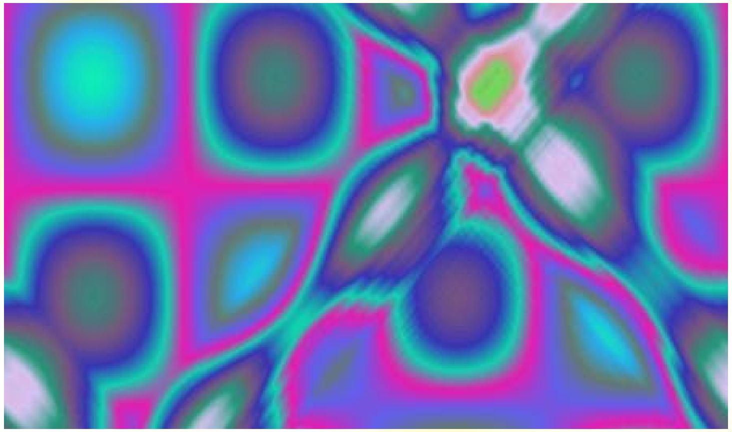
Segregačný model

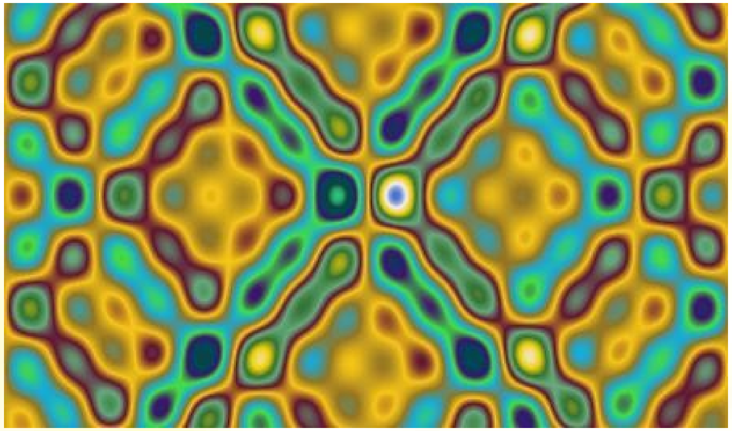
- Veľká sieť 500x500
- 1500 agentov, 1050 zelených, 450 červených, 1000 neobsadených = 3 stavy
- Každý agent ma “*toleranciu*“
- Zelený agent sa “cíti dobre” ak pomer zelených ku červeným Mooreovym susedom je väčší ako zvolená tolerancia, obdobne to platí pre červeného agenta



Viacstavové 2D CA vzory



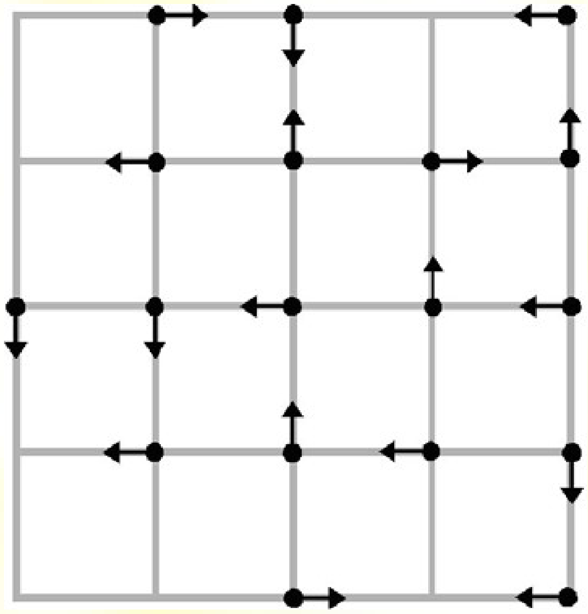




Lattice Gas CA

Príkladom je HPP (Hardy, Pomeau, de Pazzis; 1986)
Model trpí nedôslednosťou na makroskopickej úrovni)

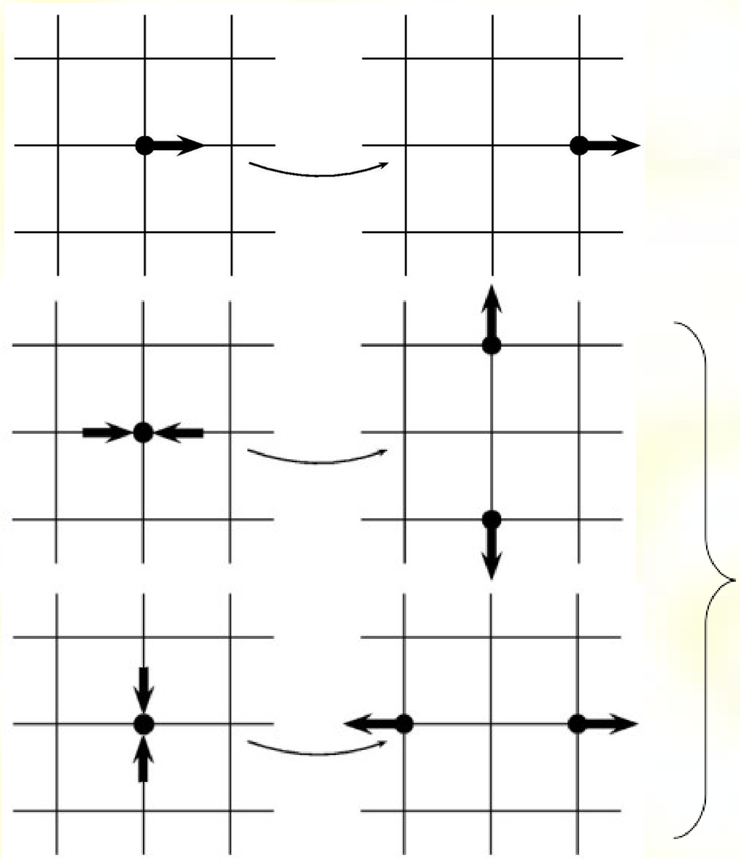
- Dynamika boolovských hodnôt na pravouhlej mriežke
- Proces je synchrónny a je založený na diskrétnych časových krokoch



Nedostatky tohto modelu rieši FHP model na 6-uholníkovej mriežke

Kolízne pravidlá HPP modelu

Čas t Čas $t+\Delta t$

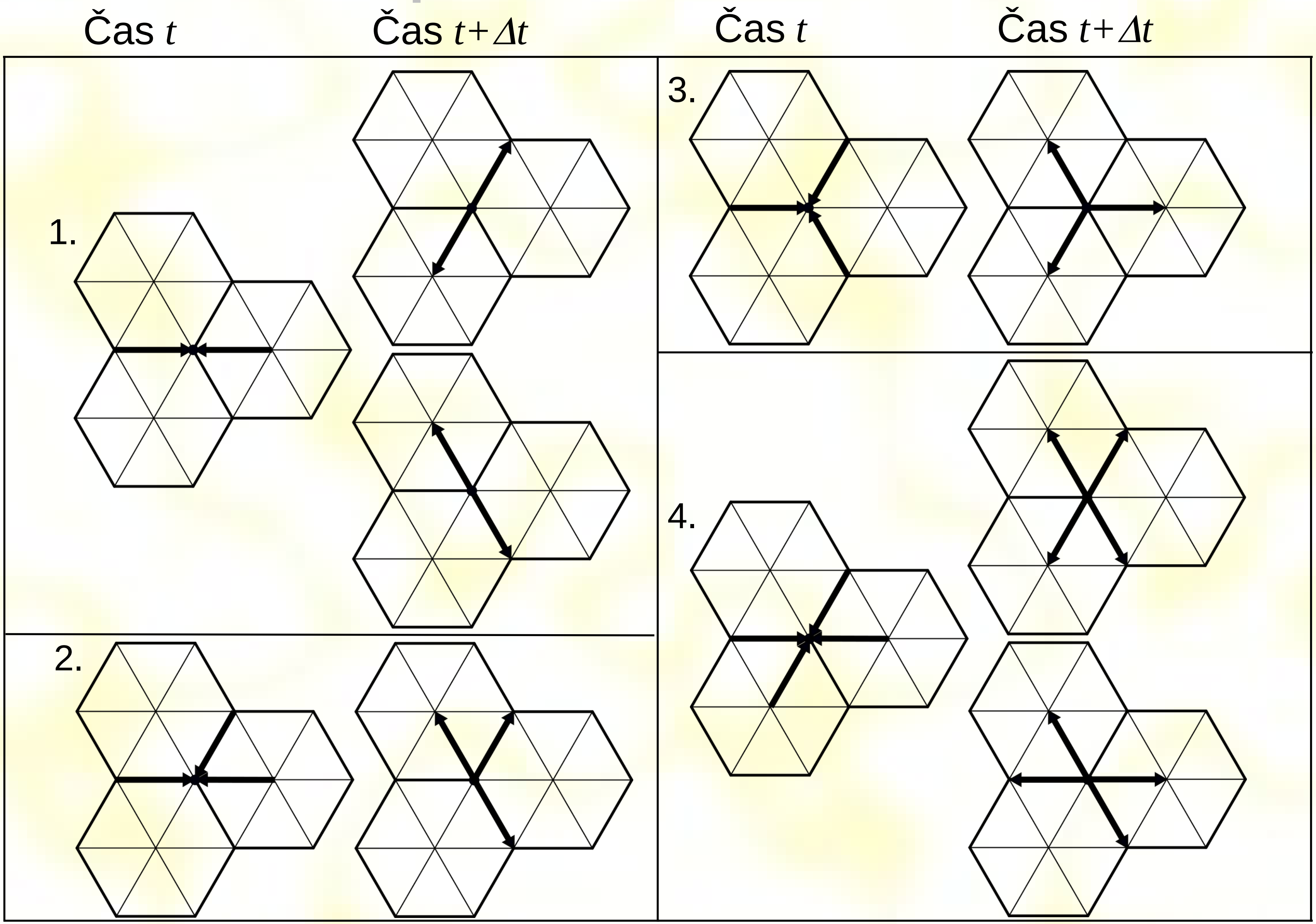


Neobmedzený pohyb bez kolízie

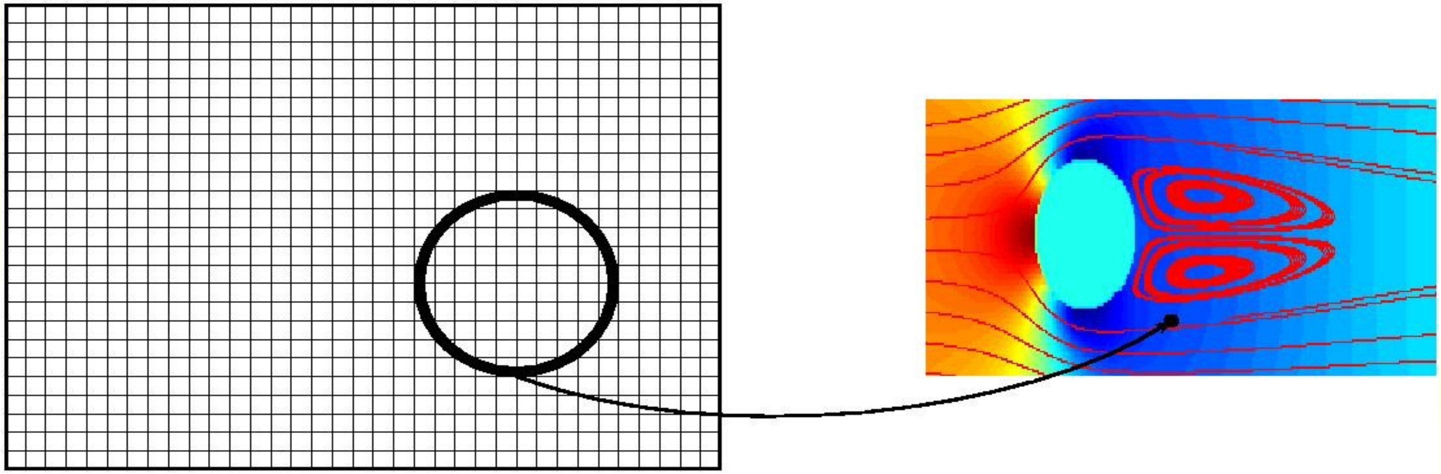
Dve častice sa pri kolízii odrazia kolmo na smer kolízie

Všetky ostatné situácie sú bez kolízií

Kolízne pravidlá FHP modelu



Vyčíslenie LG CA



Diskrétne rozdelenie častíc

Priemerné „makroskopické“ hodnoty

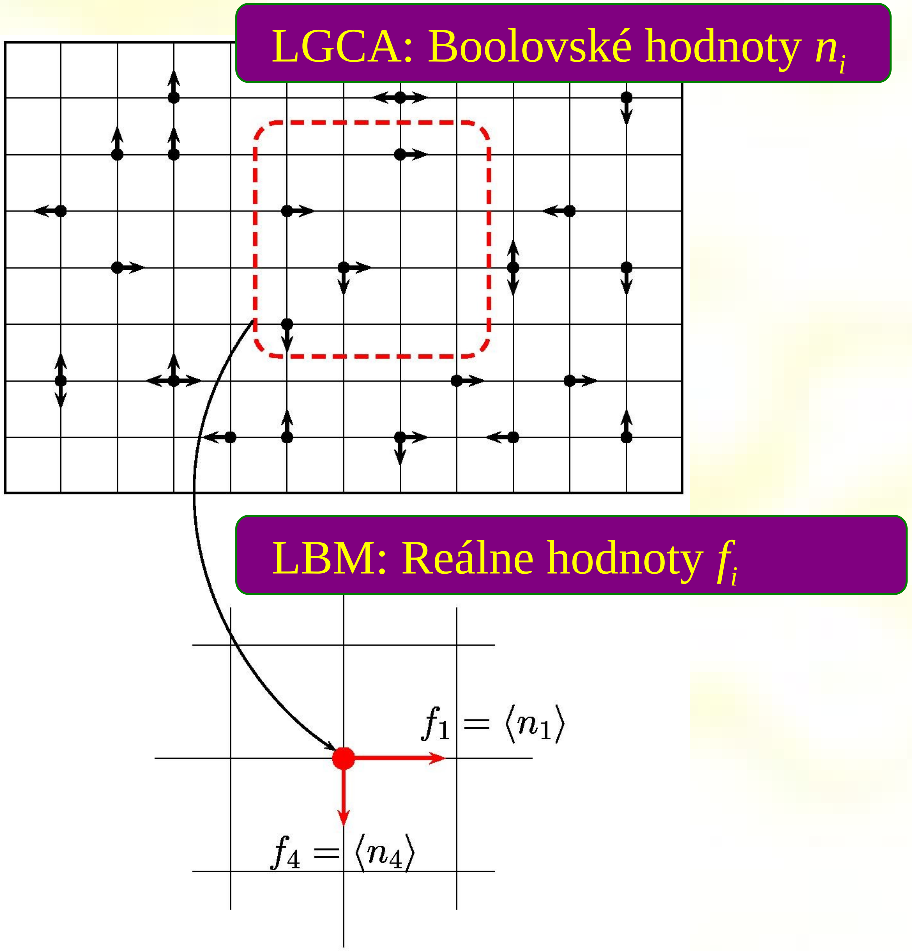
- Realistická makroskopická dynamika tekutiny sa prejaví iba ak je mriežka dostatočne zaplnená
- Problém: Je nutná veľká mriežka a treba počítať priemernú hodnotu

Od LG CA k LBM

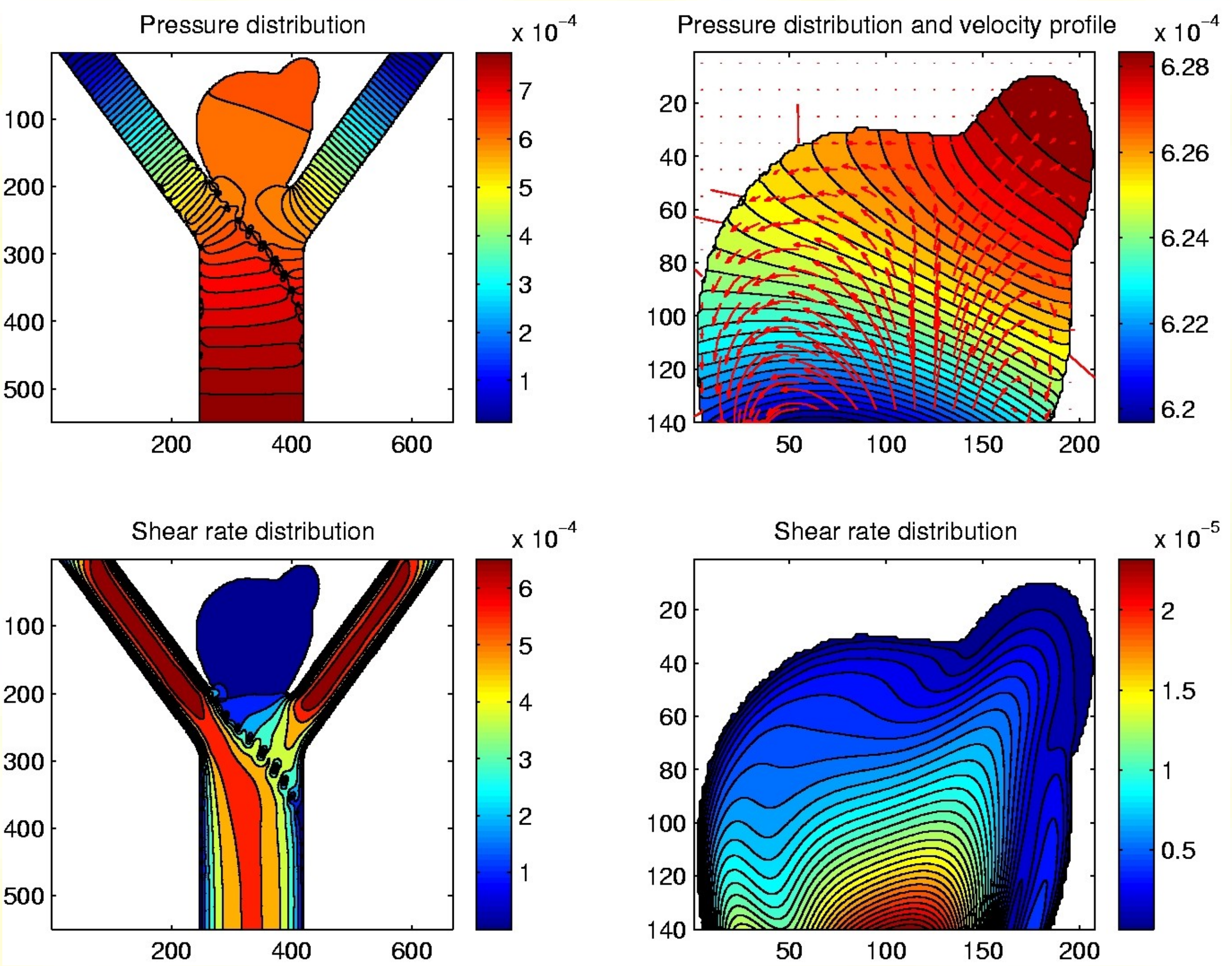
Myšlienka je implementovať dynamiku priamo na priemerné hodnoty

Výhody:
Jeden uzol reprezentuje hustotu častíc a diskrétna dynamika je nahradená plynulým tokom. Keďže nie je treba počítať priemer z viacerých častíc systém sa rýchlejšie vyčísľuje

Nevýhody:
Zanedbávajú sa korelácie vyššieho rádu medzi časticami. Pri výpočte sa môže vyskytnúť numerická nestabilita

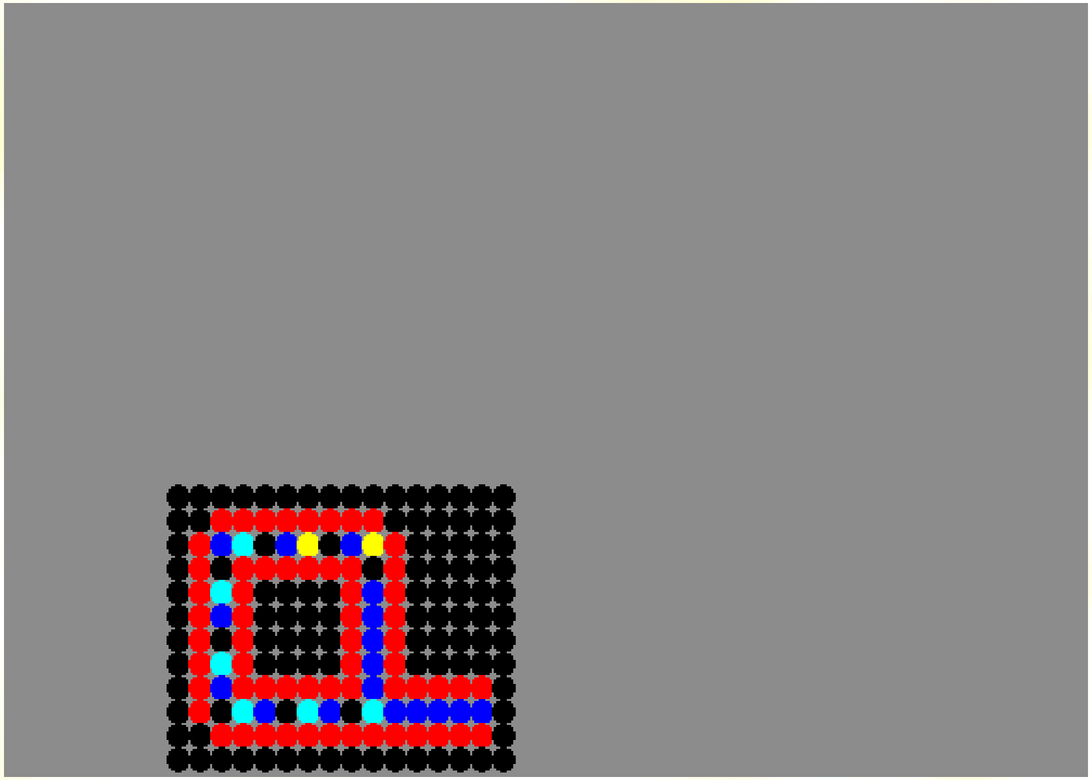


LGCA príklady



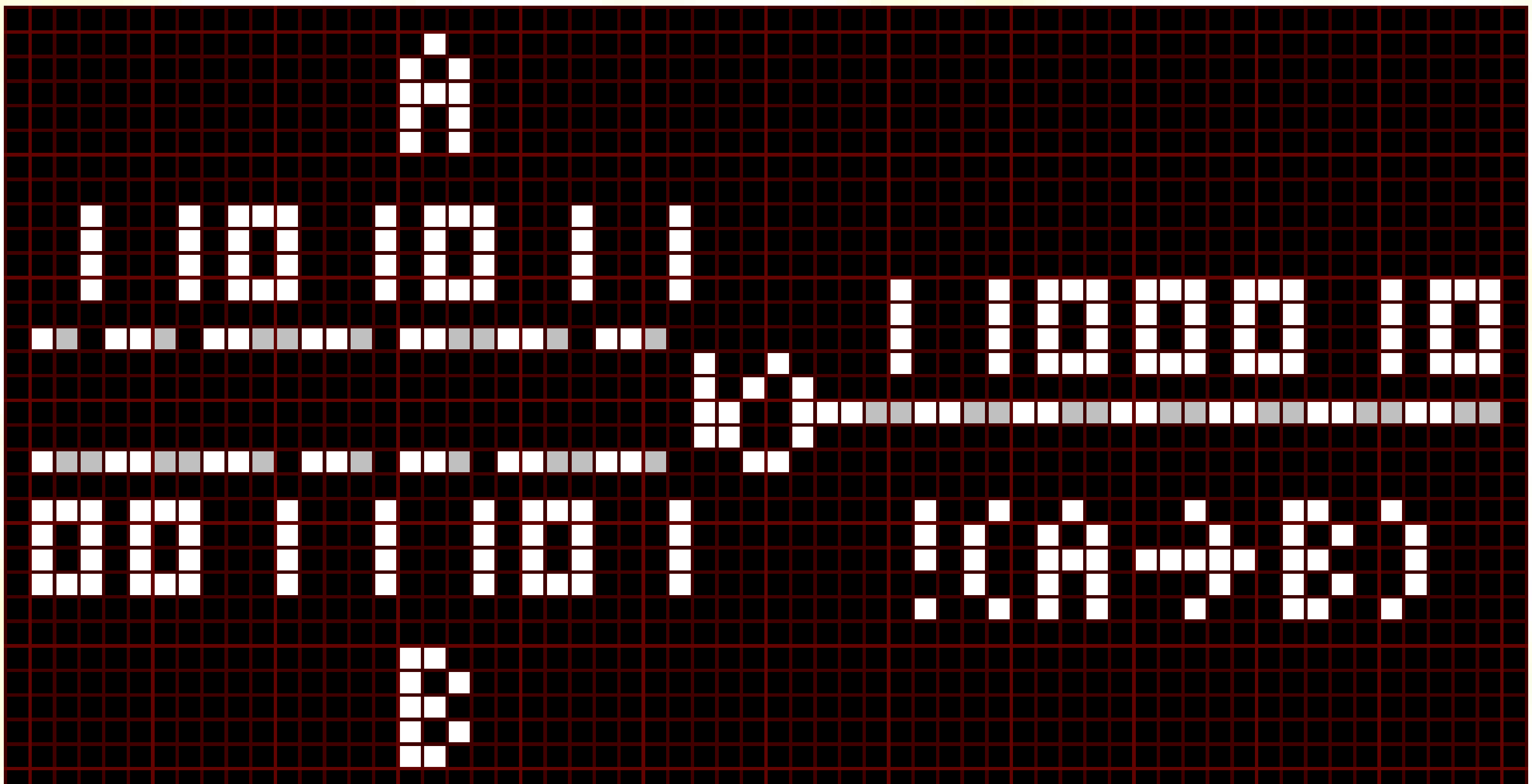
Sebareplikácia

- Langtonova sľučka – má iba 8 stavov
- Vychádza z Coddovho CA
- Obsahuje „obalený vodič“ v ktorom sa šíri „signál“



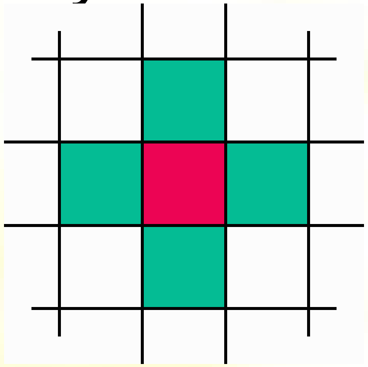
Simulácia logických obvodov

- CA umožňujú simulovať logické operácie ako AND, OR, XOR, NOT a ich kombinácie



CA a (binárne) textúry

- Textúra = vizualizácia stavu CA
- NetLogo, StarLogo – Agent Based Modeling Toolkits IDE vhodné aj pre pre CA
- Zovšeobecnený jazyk pre CA?
- Jedna z možností je využiť myšlienku Wolframovej nomenklatúry pravidiel
- Najjednoduchší príklad
  - torus
  - binárny stav
  - 4-susednosť



Náhodné generovanie programu

- Program = 32-bit číslo
- Náhodné generovanie
- Počiatočný stav:
  - Biely šum
- Vytvorené obrázky:
  - Komplexné vzory
  - ako napr. reakčná difúzia