

Automaty Komórkowe

Wykład 12

<https://github.com/houp/ca-class>

Witold Bołt, 05.06.2024

Poprzednio omówiliśmy

- **Wykład 1:** Sprawy organizacyjne, motywację do zajmowania się CA, podstawowe pojęcia / definicje / intuicje.
- **Wykład 2:** Definicja (formalna) i podstawowe fakty o ECA. Reprezentacja Wolframa.
- **Wykład 3:** Symetrie w zbiorze ECA, relacje do ogólnej teorii układów dynamicznych, własności CA/ECA.
- **Wykład 4:** Alternatywne reprezentacje reguły lokalnej, problem klasyfikacji gęstości (DCP).
- **Wykład 5 (zdalny):** Algorytmy ewolucyjne - poszukiwanie automatów komórkowych o określonych własnościach
- **Wykład 6:** Stochastyczne automaty komórkowe, pLUT, α -ACAs, Diploid CAs, stochastic mixture, dekompozycja
- **Wykład 7:** Afiniczne Ciągłe Automaty Komórkowe - wielomiany, cLUT, relaxed DCP + bonus - praca w IT
- **Wykład 8:** Identyfikacja Deterministycznych Automatów Komórkowych
- **Wykład 9:** Identyfikacja Stochastycznych Automatów Komórkowych
- **Wykład 10 (zdalny):** Dwu-wymiarowe Automaty Komórkowe / Reguła Life i Life-like / Totalistyczne i Zewnętrzne-Totalistyczne Automaty Komórkowe (totalistic & outer-totalistic CAs)
- **Wykład 11:** Modele pożaru lasu, rozprzestrzeniania się epidemii (SIR), GH i podobne modele

Co będzie dalej*

- (05.06) Wykład 12: Automaty Komórkowe zachowujące gęstość; Zastosowania w modelowaniu ruchu ulicznego
- (12.06) **Wykład 13:** Nie-jednorodne (non-uniform) Automaty Komórkowe; Neural CAs (Neuronowe Automaty Komórkowe);
- (19.06 ?) **Wykład 14:** spotkanie w siedzibie **Jit Team** w Gdyni (ul. **Łużycka 8C**, Budynek **Tensor Y**, **Gdynia** Redłowo) - temat *niespodzianka* ;)
 - Alternatywne terminy: 14.06, 18.06, ...

Intuicja

Gęstość

- Niech $\mathbf{x} \in \{0,1\}^N$. **Gęstością** konfiguracji \mathbf{x} nazywamy liczbę: $\rho(\mathbf{x}) = N^{-1} \sum_{i=0}^{N-1} x_i$.

Zwróć uwagę, że gęstość bezpośrednio odpowiada liczbie jedynek w konfiguracji.

- Analogiczną definicję można podać dla konfiguracji $\mathbf{x} \in \{0,1,\dots,q-1\}^N$ i interpretować gęstość jako “średni” stan komórki.
- Niech $\mathbf{x} \in \{0,1,\dots,q-1\}^N$. **Gęstość stanu** $p \in \{0,1,\dots,q-1\}$ w konfiguracji \mathbf{x} nazywamy liczbę: $\rho_p(\mathbf{x}) = N^{-1} \#_p(\mathbf{x})$, gdzie $\#_p(\mathbf{x})$ to liczba wystąpień stanu p w konfiguracji \mathbf{x} .

Oczywiście zachodzi: $\sum_{p=0}^{q-1} \rho_p(\mathbf{x}) = 1$. Gęstość stanu i gęstość to **inne** (acz powiązane)

konceptcje!

Zachowanie gęstości...

- Ogólna intuicja jest prosta. Chcielibyśmy zbadać CA, które **zachowuje** gęstość, czyli w trakcie ewolucji automatu gęstość się nie zmienia. Musimy oczywiście ustalić, o *którą gęstość nam chodzi* (będziemy rozważać różne).
- Dlaczego to jest **ważne**?
 - Stan 0 - puste miejsce
 - Stany > 0 - reprezentacja “masy” / “energii” / “potencjału” / “obiektu”
 - Zachowanie gęstości - zachowanie wybranej własności fizycznej układu
- Dlaczego to jest **trudne**?

Dear ChatGPT. Can you implement a deterministic 2D density conserving Cellular Automata showing simple car traffic model? Please consider using Python, Numpy and pygame for visualization. Please concentrate on code readability and runtime performance. I'd like to use this code to explain the concept of number conserving / density conserving CAs.

To nie koniecznie jest CA!!!

Dlaczego?

```
def update_grid(grid):
    new_grid = np.zeros_like(grid)
    for i in range(grid.shape[0]):
        for j in range(grid.shape[1]):
            if grid[i, j] == 1:
                # Simple rightward movement with periodic boundary conditions
                new_j = (j + 1) % grid.shape[1]
                if grid[i, new_j] == 0:
                    new_grid[i, new_j] = 1
                else:
                    new_grid[i, j] = 1
            else:
                new_grid[i, j] = 1
    return new_grid
```


Agent Based Modeling

- “An **agent-based model** (ABM) is a computational model for simulating the actions and interactions of autonomous agents (both individual or collective entities such as organizations or groups) in order to understand the behavior of a system and what governs its outcomes.” ([wikipedia](#))
- Zbiór “agentów”, których stan jest aktualizowany w czasie, w oparciu o ustaloną regułę / reguły. Agenci wchodzą w interakcje ze sobą i otoczeniem (przestrzenią). Mogą m.in. poruszać się, komunikować, ulegać modyfikacjom itd.
- Podobnie jak w CA przestrzeń może być podzielona na przeliczalne komórki. Podobnie jak w CA zbiór stanów może być skończony itd.
- W odróżnieniu od CA, w ABM uaktualniamy stan każdego z agentów a nie każdej z komórek. W CA każda komórka niezależnie zmienia swój stan. Stany (np. Lokalizacja) agentów nie zawsze mogą być zmienione niezależnie od siebie - chociażby dlatego, że 2 obiekty nie mogą znajdować się fizycznie w tym samym miejscu.
- Istnieją klasy ABM, które można zrealizować w oparciu o CA i vice-versa, istnieją CA, które można wyrazić w języku ABM, ale ogólnie to **nie** są tożsame pojęcia. Niestety są czasem **mylone**! Również przez naukowców...

**Czy zatem istnieją CA
zachowujące gęstość?**

Density conserving CAs

Recent insights into number-conserving cellular automata

Barbara Wolnik, Witold Bolt, and Bernard De Baets

Abstract This chapter provides a summary of recent results on number-conserving cellular automata, along with a general introduction to the field. We cover results in arbitrary dimensions, with particular attention to the von Neumann neighborhood, while also addressing specific classes of cellular automata, such as Wolfram's well-known elementary cellular automata and non-uniform variants thereof, affine continuous cellular automata and cellular automata on triangular grids.

Key words: Affine continuous cellular automata, cellular automata, non-uniform cellular automata, number conservation, triangular cellular automata

Artykuł został zaakceptowany i zostanie opublikowany w 2024 r.
Wersja pre-print na GitHub.

Density (number) conservation

- Niech $\mathbf{x} \in \{0,1\}^N$ będzie konfiguracją jednowymiarowego CA.
- Gęstością konfiguracji \mathbf{x} nazywamy liczbę:
$$\rho(\mathbf{x}) = N^{-1} \sum_{i=1}^N x_i.$$
- Zauważmy, że $\rho(\mathbf{x}) \in [0,1]$ dla dowolnego $\mathbf{x} \in \{0,1\}^N$.
- Mówmy, że automat komórkowy zdefiniowany przez regułę globalną $F: \{0,1\}^N \rightarrow \{0,1\}^N$ **zachowuje gęstość** wtedy i tylko wtedy, gdy:
$$\rho(\mathbf{x}) = \rho(F(\mathbf{x}))$$
 dla dowolnego $\mathbf{x} \in \{0,1\}^N$.
- Dużo bardziej ciekawi nas przypadek $F: \{0,1\}^* \rightarrow \{0,1\}^*$, czyli zachowanie gęstości dla każdego $N > 0$. I tak będziemy zawsze rozumieli tą definicję.
- Definicje uogólnia się na automaty komórkowe o większej liczbie **wymiarów** oraz większym **zbiorniku stanów**. Jedyne założenie, którego musimy pilnować to aby elementy zbioru stanów dały się dodawać i dzielić przez liczby naturalne - czyli najlepiej żeby po prostu były to liczby.

Dlaczego to jest ważna własność?

- Zasada zachowania masy w fizyce
- Inne zasady fizyczne: zachowanie pędu, równowaga sił, zachowanie energii, zachowanie ładunku i wiele innych. *“W przyrodzie nic nie ginie. Jedynie się zmienia.”* Stała liczba cząstek elementarnych we wszechświecie? **Kwanty** 🧛
- Modelowanie zjawisk biologicznych, chemicznych, przyrodniczych, społecznych...
 - W układzie nie powstają (z niczego) nowe cząstki, a także cząstki nie znikają.
 - Oczywiście ta własność jest istotna w wielu modelach, **ale nie we wszystkich**. Jeśli modelujemy proces rozrostu rośliny, rozwój nowotworu, pożar lasu - to “masa” nie jest zachowana.

Dlaczego ta własność jest ciekawa?

- Zastanówmy się przez chwilę skąd wziąć automat komórkowy, który zachowuje gęstość, skoro definicja tej własności opisana jest na poziomie **reguły globalnej**, a prawie wszystko co dotychczas robiliśmy z automatami komórkowymi działa się na poziomie **reguły lokalnej**.
- Bezpośrednie rozwiązanie układu równań:

$$\begin{cases} F(\mathbf{x}) = (f(x_{-1}, x_0, x_1), \dots, f(x_{N-2}, x_{N-1}, x_0)) \\ \rho(\mathbf{x}) = \rho(F(\mathbf{x})) \end{cases} \quad \text{i poszukiwanie w ten sposób}$$

reguł lokalnych jest conajmniej **trudne**. Co więcej, w ogólności musielibyśmy pomyśleć jak poradzić sobie z różnymi N , a także napisać równania dla większych promieni (niż $r > 1$).

Dlaczego ta własność jest ciekawa?

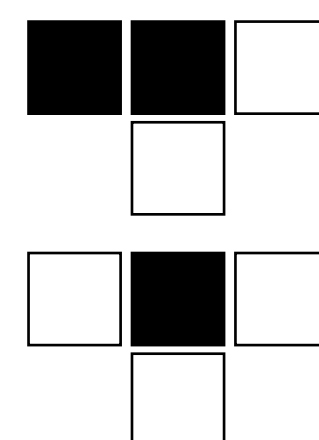
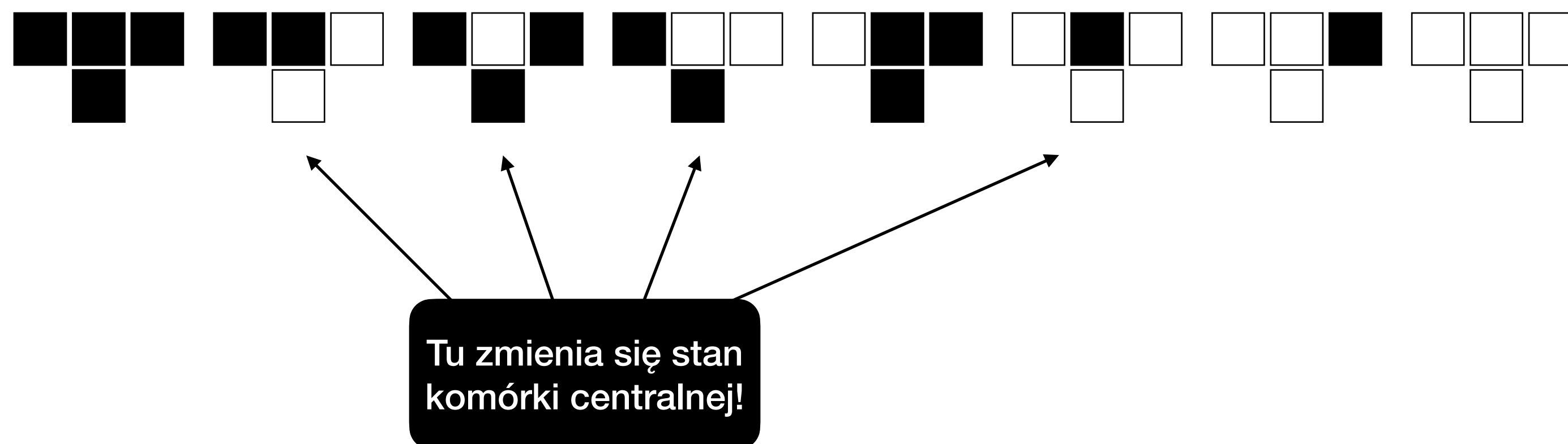
- Skoro tak trudno szukać reguły zachowujące gęstość, to skąd w ogóle wiadomo czy one istnieją?
- Przykłady (trywialne):
 - **Identyczność**, bo $F(\mathbf{x}) = \mathbf{x}$ dla każdego \mathbf{x} , więc ta reguła zachowuje wszystko. Dla przypomnienia reguła lokalna dana jest przez $f(x, y, z) = y$.
 - **Shift-left / Shift-right**, czyli automaty komórkowe zadane przez regułę lokalną $f(x, y, z) = x$ lub $f(x, y, z) = z$.
- Pytanie (nietrywialne): czy istnieją **inne** (nietrywialne) automaty komórkowe zachowujące gęstość?
- Okazuje się, że **TAK**.


Przypadek ECA (powtórka)

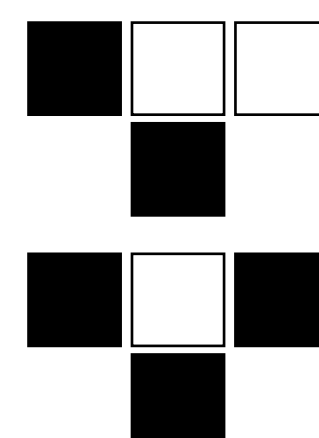
- **Identyczność**, bo $F(\mathbf{x}) = \mathbf{x}$ dla każdego \mathbf{x} , więc ta reguła zachowuje wszystko. Dla przypomnienia reguła lokalna dana jest przez $f(x, y, z) = y$. (ECA **204**)
- **Shift-left / Shift-right**, czyli automaty komórkowe zadane przez regułę lokalną $f(x, y, z) = x$ lub $f(x, y, z) = z$. (ECA **170** i ECA **240**)
- **Traffic-left / Traffic-right**, (ECA **184** i ECA **226**).
- Powyżej wymienione reguły to **kompletna** lista elementarnych automatów komórkowych zachowujących gęstość. To znaczy, że wszystkie inne ECA **nie są** density conserving.



Reguła Traffic ECA 184

- Analizując poszczególne przejścia reguły ECA 184 można zrozumieć, że choć komórki zmieniają swoje stany, to te zmiany są ze sobą “sprzężone”.
- Jeśli jakaś komórka była w stanie 1 i zgodnie z LUT w następnej chwili czasu powinna być w stanie 0, to jej najbliższy sąsiad po prawej stronie jest na pewno w sytuacji, która skutkuje zmianą z 0 na 1.
- Podobnie jeśli komórka była w stanie 0 i zgodnie z LUT zmiana stan na 1, to ma sąsiada po lewej stronie, który jest w odwrotnej sytuacji.
- Innymi słowy te zmiany dzieją się parami i przez to liczba zer i jedynek nie zmienia się!
- Stąd łatwo zauważyć, że ta reguła zmienia **wszystkie** bloki $[0,1]$ w $[1,0]$. Reguła bliźniacza ECA 226 działa analogicznie i zmienia $[1,0]$ w $[0,1]$.



W tym przejściach “znika” jedynka. No ale sąsiad po prawej stronie jest na pewno w jednej z sytuacji:   czyli był w stanie zero, a teraz będzie w stanie jeden.




Analogicznie te dwa przejścia “tworzą” jedynkę. No ale sąsiad po lewej stronie jest na pewno w jednej z sytuacji:   czyli był w stanie jeden, a teraz będzie w stanie zero.

Dlaczego traffic? [przypomnienie]

- Niech stan 0 oznacza **pustą drogę**  a stan 1 oznacza **samochód** . Zatem reguła ECA 184 zmienia   w  .
- Zatem dla konfiguracji:         
- dostaniemy:         
- a następnie:         
- a następnie:         
- a następnie:          i tak dalej i tak dalej
- Czyli reguła modeluje **prosty ruch uliczny** po jednopasmowej drodze, w której wszystkie samochody albo stoją w miejscu ($v=0$) albo jadą przed siebie w tym samym kierunku ze stałą prędkością ($v=1$) tak długo jak mogą. Jeśli dodamy periodyczny warunek brzegowy to jeżdżą po torze, który jest pętlą.

Czy ECA Traffic to ABM?

- Mówiliśmy o tym, że CA to nie ABM i nie można mieszać tych pojęć.
- No ale jednak czasem można :)
- Każda jedynka to agent , który ma dwa możliwe stany: prędkość $v=0$ lub prędkość $v=1$. Każdy agent ma takie samą dążenie - chciałby jechać z maksymalną prędkością $v=1$ **jeśli to tylko możliwe**.
- Jeśli agent ma prędkość $v=0$, to patrzy przed siebie i sprawdza czy może ruszyć. Jeśli może to zmienia prędkość na $v=1$ i wykonuje ruch.
- Jeśli agent ma prędkość $v=1$, to próbuje się dalej poruszać - chyba, że napotyka przeszkodę. Wtedy staje w miejscu, w którym jest i zmienia prędkość na $v=0$ i czeka aż będzie mógł jechać dalej.

Co można zrobić lepiej?

- A co gdyby prędkość z jaką poruszają się samochody była by bardziej zróżnicowana? To znaczy niech każdy samochód może poruszać się z określoną prędkością $v \in \{0, 1, \dots, v_{\max}\}$.
- Na początku każdy samochód na drodze ma $v=0$.
- Każdy samochód próbuje ruszyć przed siebie **jeśli tylko może**. Aby ruszyć trzeba przyspieszyć czyli inkrementować swoją prędkość.
- Samochody, które mają przed sobą wolne pole zmieniają prędkość na $v=1$ i przesuwają się o jedno pole. Czyli pierwszy krok jest taki sam jak w ECA 184, **ale!**
- W następnym kroku ci ci już ruszyli i mają $v=1$ chcą dalej przyspieszać jeśli mogą! Przyspieszają zatem do $v=2$ i poruszają się o **dwa pola**. A co jeśli nie da się poruszyć o dwa pola? Wtedy nie przyspieszamy i poruszamy się o jedno! A jeśli się nie da o jedno? To stajemy.
- Ci co przyspieszyli do $v=2$, w kolejnym kroku próbują dalej przyspieszać do $v=3$ i ruszyć się o trzy pola. I tak dalej i tak dalej.
- Dodatkowo można wprowadzić czynnik losowy - losowe zwalnianie z prawdopodobieństwem p .
- Ewidentnie jest to ABM, ale czy jest to również CA?

A cellular automaton model for freeway traffic

Kai Nagel⁽¹⁾ and Michael Schreckenberg⁽²⁾

⁽¹⁾ Mathematisches Institut, Universität zu Köln, Weyertal 86-90, W-5000 Köln 41, Germany

⁽²⁾ Institut für Theoretische Physik, Universität zu Köln, Zùlpicher Str. 77, W-5000 Köln 41, Germany

(Received 3 September 1992, accepted 10 September 1992)

Abstract. — We introduce a stochastic discrete automaton model to simulate freeway traffic. Monte-Carlo simulations of the model show a transition from laminar traffic flow to start-stop-waves with increasing vehicle density, as is observed in real freeway traffic. For special cases analytical results can be obtained.

<https://hal.science/jpa-00246697/document>

Model NaSch (Nagel - Schreckenberg)

Our computational model is defined on a one-dimensional array of L sites and with open or periodic boundary conditions. Each site may either be occupied by one vehicle, or it may be empty. Each vehicle has an integer velocity with values between zero and v_{\max} . For an arbitrary configuration, one update of the system consists of the following four consecutive steps, which are performed in parallel for all vehicles:

- 1) **Acceleration:** if the velocity v of a vehicle is lower than v_{\max} and if the distance to the next car ahead is larger than $v + 1$, the speed is advanced by one [$v \rightarrow v + 1$].
- 2) **Slowing down (due to other cars):** if a vehicle at site i sees the next vehicle at site $i + j$ (with $j \leq v$), it reduces its speed to $j - 1$ [$v \rightarrow j - 1$].
- 3) **Randomization:** with probability p , the velocity of each vehicle (if greater than zero) is decreased by one [$v \rightarrow v - 1$].
- 4) **Car motion:** each vehicle is advanced v sites.

Through the steps one to four very general properties of single lane traffic are modelled on the basis of integer valued probabilistic cellular automaton rules [9, 10]. Already this simple model shows nontrivial and realistic behavior. Step 3 is essential in simulating realistic traffic

Czy model NaSch to jest CA?

- Aby był to CA, to musimy określić conajmniej:

- Zbiór stanów

- Sąsiedztwo

- Regułę lokalną

Opisana w artykule, tylko trzeba patrzeć na nią “z perspektywy” komórek drogi a nie aut.

Stan 0' - puste miejsce

Stany $0, \dots, v_{\max}$ - samochód z określoną prędkością

Komórka w stanie 0' zostanie zajęta przez samochód jadący z prawej, który jest w odległości równej swojej aktualnej prędkości zatem sąsiedztwo musi widzieć v_{\max} sąsiadów z jednej ze stron. Co więcej musi widzieć v_{\max} komórek po drugiej ze stron aby ustalić nową prędkość.

Komórka w stanie $0, \dots, v_{\max}$ musi widzieć v_{\max} sąsiada po jednej ze stron (w kierunku ruchu) aby wiedzieć czy może się jakkolwiek ruszyć i jaką prędkość może ustalić jeśli jednak się nie ruszy.

Czyli sąsiedztwo ma promień v_{\max} .

Czy model NaSch to CA
zachowujący gęstość?



NIE!

No ale jak to?

- Przecież mówiłeś, że do modelowania ruchu musi być zachowana gęstość, żeby nie znikaly pojazdy. A tu one nie znikają!
- Tak! Pojazdy nie znikają - jest zachowana masa. Ale - zmienia się prędkość, która **nie** jest zachowana.
- Zatem ten CA **zachowuje gęstość**, ale jedynie **jednego** ze stanów: 0' i to nam wystarczy.

Więcej wymiarów... i ciekawe
wyniki

Theorem 2 *Let $d \geq 1$. There are exactly $4d + 1$ d -dimensional **binary** NCCAs with the **von Neumann** neighborhood: the identity rule, the shift rules and the traffic rules in each of the $2d$ directions.*

Czy tych reguł jest mało?

To są ECA!

$k \backslash m$	2	3	4	5	6	7
2	2	5	22	428	133 184	1 571 814 309
3	4	144	5 448 642	?	?	?
4	10	89 585	?	?	?	?
5	30	1 876 088 314	?	?	?	?
6	106	?	?	?	?	?

Table 4 Number of one-dimensional m -input NCCAs with state set $Q = \{0, 1, \dots, k - 1\}$.

Automaty “ m -input” to takie, których reguła lokalna ma m argumentów.

k	2	3	4	5	6	7
	3	3	21	21	471	1 669

Table 5 Number of reversible one-dimensional 3-input NCCAs with state set $Q = \{0, 1, \dots, k - 1\}$.

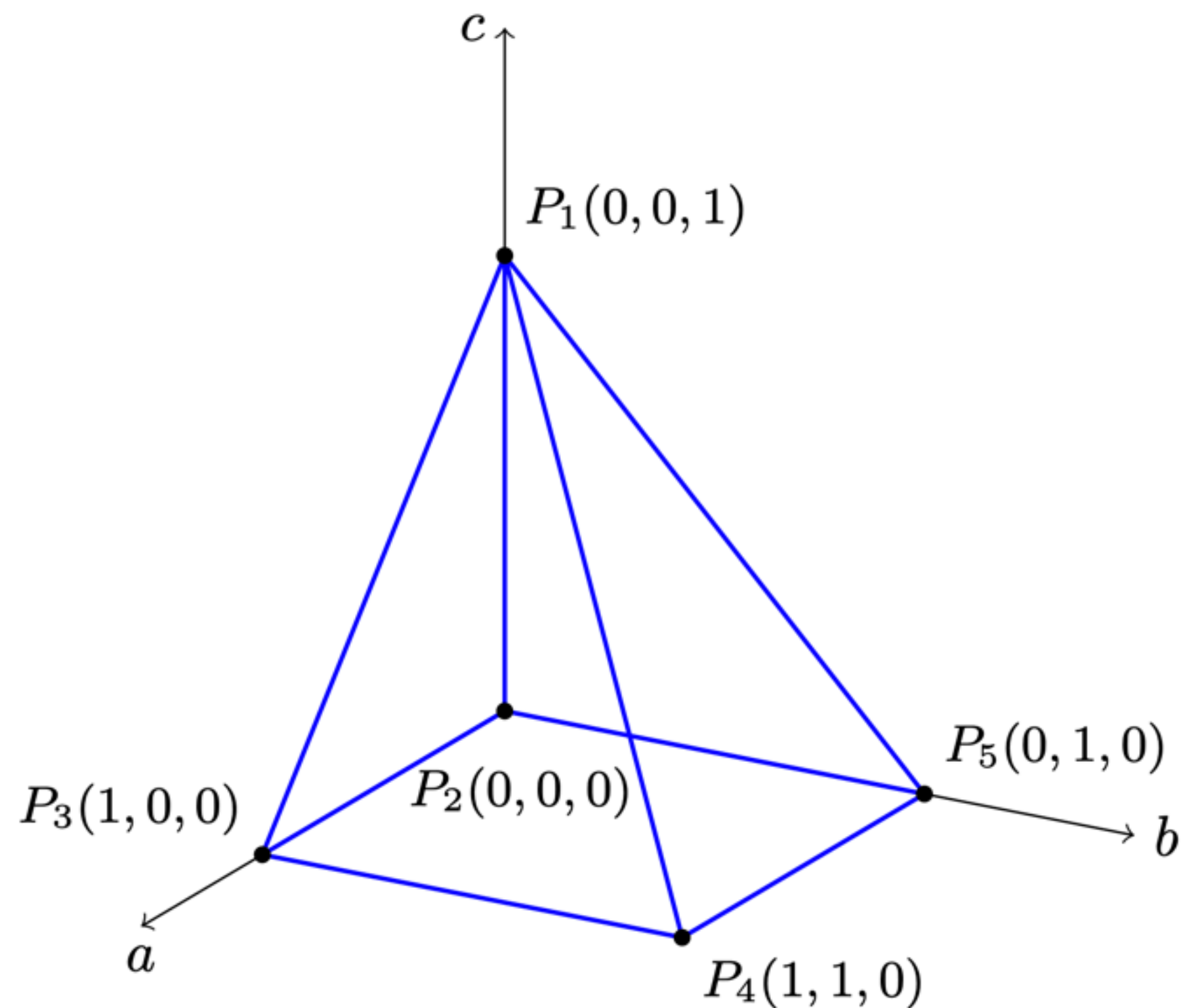
k	2	3	4	5	6	7
	1	1	1	4	116	30 144

Table 6 Number of rotation-symmetric two-dimensional NCCAs with the von Neumann neighborhood and k states (assuming state set $Q = \{0, 1, \dots, k - 1\}$).

Density Conserving ACCAs

- Definicja zachowania gęstości w sposób oczywisty rozszerza się na ACCA, bo mamy tam stany z przedziału $[0,1]$, które oczywiście można dodawać.
- Można pokazać, żeby f jest regułą lokalną ACCA, który jest density conserving wtedy i tylko wtedy gdy:
$$f(x, y, z) = (a + b + c - 1)(x - z)y + (1 - a - c)x + cy + az, \text{ gdzie } a, b, c \in [0,1] \text{ oraz } a, b \leq 1 - c.$$
- Ten wzór pewnie nic ciekawego nam nie mówi :) **Ale!** Dla każdego zestawu parametrów (a, b, c) otrzymujemy pewną regułę i możemy narysować zbiór punktów $(a, b, c) \in \mathbb{R}^3$ i zastanowić się co to jest :)
- Z powyższego wzoru można odczytać postać **cLUT** takiego ACCA!

Okazuje się, że wierzchołki P_i odpowiadające pewnym ACCA, mają swoje konkretne znaczenie:



- P_1 - identyczność (ECA 204)
- P_2 - traffic-right (ECA 184)
- P_3 - shift-left (ECA 170)
- P_4 - traffic-left (ECA 226)
- P_5 - shift-right (ECA 240)

Innymi słowy ACCA zachowujące gęstość to kombinacja (średnia ważona) uogólnionych, ale **znanych nam dobrze** ECA zachowujących gęstość.

Theorem 4 *Let f be the local rule of a density-conserving ACCA parameterized by $(a, b, c) \in P$ and let F be the corresponding global rule. If $0 < c < 1$, then for any $\mathbf{x} \in X_{\mathbf{N}}$, it holds that:*

$$\lim_{t \rightarrow \infty} F^t(\mathbf{x}) = (\rho(\mathbf{x}), \rho(\mathbf{x}), \dots, \rho(\mathbf{x})), \quad (6)$$

where $\rho(\mathbf{x})$ is the density of \mathbf{x} , i.e., $\rho(\mathbf{x}) = \frac{1}{N}(x_0 + x_1 + \dots + x_{N-1})$.

Density Conserving SCAs

- Dla SCAs możemy zdefiniować uogólnienie pojęcia zachowania gęstości w terminach **wartości oczekiwanej**. Czyli oczekiwana gęstość konfiguracji ma być zachowana.
- Okazuje się, że przy tak postawionej definicji można pokazać, żeby SCA zachowywał gęstość to jego pLUT musi być dokładnie taki jak cLUT ACCA zachowującego gęstość. Czyli dokładnie ta sama piramida i te same ECA znów się pojawiają...
- Czyli SCA zachowujące gęstość jest po prostu stochastyczną miksturą reguł deterministycznych zachowujących gęstość.

State-conserving CAs

State-density conservation

- Jak widzieliśmy na przykładzie modelu NaSch czasem jesteśmy zainteresowani prawem zachowania gęstości konkretnego stanu.
- Możemy jednak być zainteresowani również mocniejszą własnością, tzn. $\forall_p \rho_p(\mathbf{x}) = \rho_p(F(\mathbf{x}))$ czyli, że w trakcie ewolucji automatu **nie** zmienia się liczba każdego ze stanów. Innymi słowy odpowiada to sytuacji, że mamy obiekty różnego typu, które przemieszczają się po świecie, ale **w pewnym sensie** nie wchodzą ze sobą w interakcje, które zmieniłyby ich stan.
- W przypadku CAs dwu-stanowych, oczywiście state-density conserving oraz zwykłe density conserving to dokładnie ta sama własność.
- Ale dla większej liczby stanów... tak **nie** jest.

State-density conservation

Corollary 1. *Let $\Upsilon_{3,k}$ denote the number of state-conserving one-dimensional local rules with k states and with radius one. Then for $k \geq 2$, it holds that*

$$\Upsilon_{3,k} = 2 + \sum_{i=0}^k \binom{k}{i} (2^{k-i} - 1)^i.$$

In particular, the first few elements of the sequence $(\Upsilon_{3,k})_{k=2}^{\infty}$ read as follows

5, 15, 89, 843, 11 645, 227 895, 6 285 809, 24 3593 043, 13 262 556 723, 1 014 466 283 293, ...

State-density conservation

Table 2 presents all 15 one-dimensional state-conserving ternary CAs (*i.e.*, $Q = \{0, 1, 2\}$) with radius one, together with the corresponding restrictions $F_{0,1}$, $F_{0,2}$ and $F_{1,2}$.

lookup table	$F_{0,1}$	$F_{0,2}$	$F_{1,2}$
210210210210210210210210	Sh-L	Sh-L	Sh-L
210222222210111000210111000	Id	$\langle 2, 0 \rangle$	$\langle 2, 1 \rangle$
212222000212111000212111000	Id	Id	$\langle 2, 1 \rangle$
212222010212111010212000010	$\langle 0, 1 \rangle$	Id	$\langle 2, 1 \rangle$
220110222220110111220110000	$\langle 1, 0 \rangle$	$\langle 2, 0 \rangle$	Id
220111222220111000220111000	Id	$\langle 2, 0 \rangle$	Id
222110000222110111222110000	$\langle 1, 0 \rangle$	Id	Id
222111000222111000222111000	Id	Id	Id
222111010222111010222000010	$\langle 0, 1 \rangle$	Id	Id
222111200222111200000111200	Id	$\langle 0, 2 \rangle$	Id
222111210222111210000000210	$\langle 0, 1 \rangle$	$\langle 0, 2 \rangle$	Id
222210000111210111222210000	$\langle 1, 0 \rangle$	Id	$\langle 1, 2 \rangle$
222211000111211000222211000	Id	Id	$\langle 1, 2 \rangle$
222211200111211200000211200	Id	$\langle 0, 2 \rangle$	$\langle 1, 2 \rangle$
22222222211111111000000000	Sh-R	Sh-R	Sh-R

Tutaj “restrykcja” $\langle a, b \rangle$ oznacza, że reguła zamienia blok $\langle a, b \rangle$ na $\langle b, a \rangle$ podobnie jak to robił ECA 184 (Traffic).
Sh-L = shift left, Sh-R = shift right, Id = identity.

State-density conservation dla wielu-stanów

- Okazuje się, że również dla większej liczby stanów **wszystkie** state-density conserving CAs można wytłumaczyć (w pełni scharakteryzować) za pomocą zmian bloków typu: $\langle a, b \rangle \rightarrow \langle b, a \rangle$
- Innymi słowy można podać algorytm konstruowania wszystkich takich reguł dla zadanej liczby stanów.

State-density conservation dla 3-stanów

- Okazuje się więc, że wśród automatów 3-stanowych, 1-wymiarowych o promieniu sąsiedztwa równym 1 nie ma jednego uogólnienia reguły traffic.
- Zamiast tego jest tych wariantów aż $6 + 6$ (**6 różnych** + 6 odbić - na takiej samej zasadzie jak wśród ECA są dwie reguły traffic ECA 184 i ECA 226, które działają tak samo tylko jedna w lewo a druga w prawo).
- Do czego możemy to użyć... 🤔
- Do modelowania ruchu ulicznego różnych typów samochodów? Być może, ale nie znam się na tym 😊
- Ale jest inny pomysł... 💡

State-majority classification

Majority classification

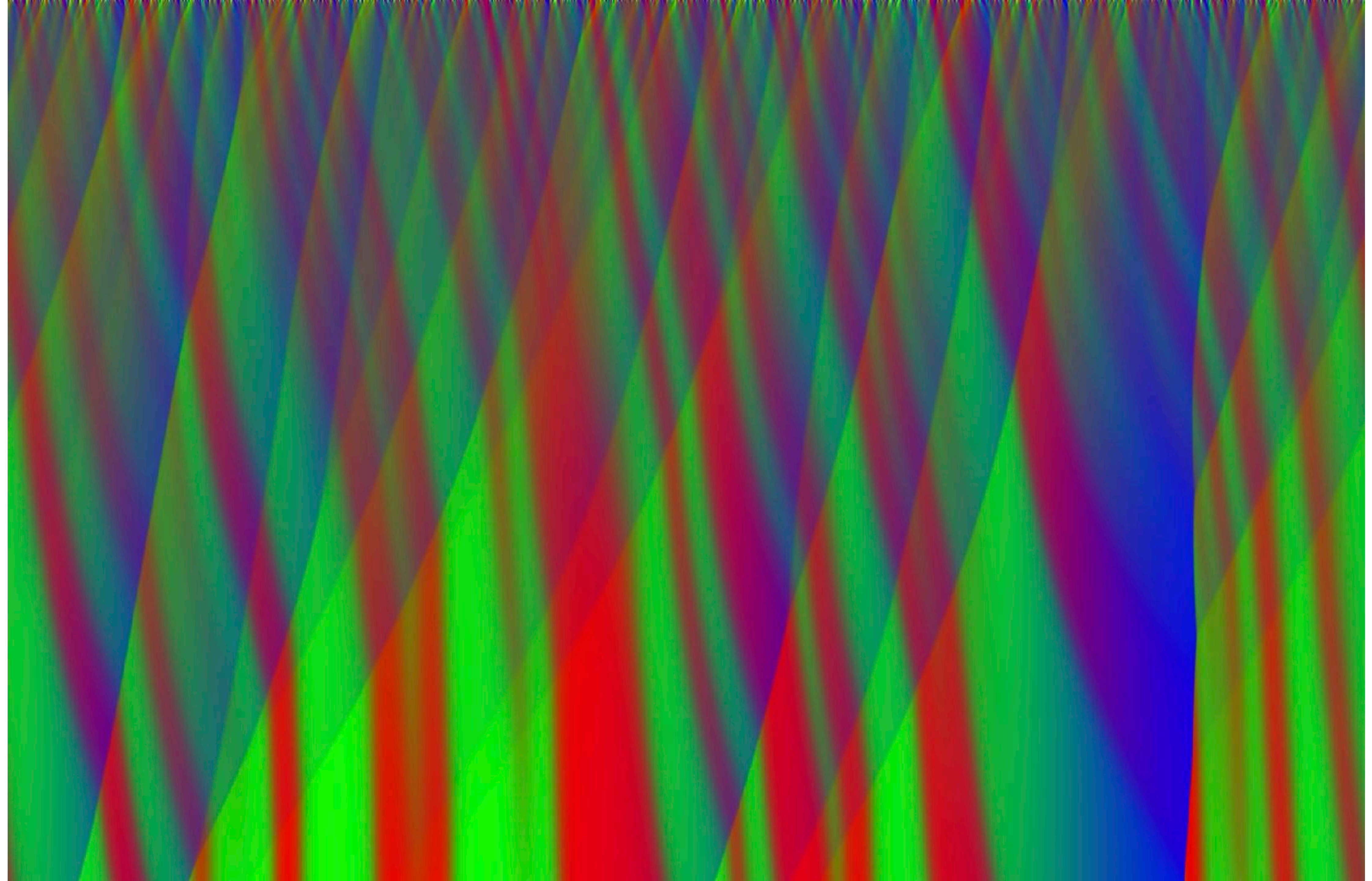
- W przypadku 2 stanów rozważaliśmy problem DCP, w którym oczekiwaliśmy, że jakiś CA odpowie na pytanie czy gęstość początkowa była większa czy mniejsza niż 0.5.
- Innymi słowy DCP odpowiada na pytanie czy było więcej 0 czy 1 w konfiguracji początkowej. Innymi słowy odpowiada na pytanie czy było “**więcej głosów**” na 0 czy na 1. Stąd czasem mówi się o majority voting w kontekście tego problemu.
- A co jeśli zrobić by głosowanie na więcej niż 2 opcje!?
- Wtedy celem takiego majority voting albo majority classification byłoby wskazanie zwycięzcy w głosowaniu pośród wielu opcji o ile jedna z opcji rzeczywiście ma większość (nie musi być to większość absolutna).
- Innymi słowy szukamy CA, który będzie zbiegać do konfiguracji homogenicznej składającej się ze stanu, którego było **najwięcej** spośród stanów w konfiguracji początkowej.

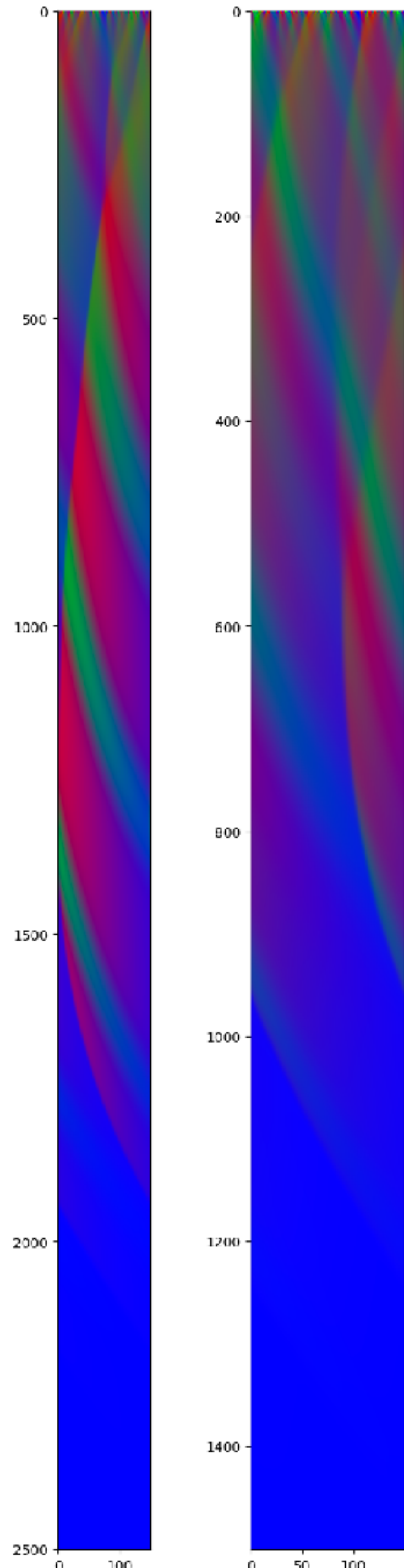
Majority classification

- Zła wiadomość: nie ma rozwiązania idealnego tak jak w DCP 😞
- Ale czy da się uogólnić pomysł prof. Fuksia (najpierw traffic potem majority), prof. Fatesa (stochastic mixture traffic-majority), Marcina (relaxed problem with ACCAs lub exact ACCA for fixed length), GKL?
- Odpowiedź jest taka jak zwykle: i **tak** i **nie**.
- **Źródło problemu.** W przypadku automatów dwustanowych istnieje reguła “majority” która przyjmuje nieparzystą liczbę argumentów i ponieważ są jedynie dwa stany, zawsze jest w stanie wskazać ten stan, który w podanym sąsiedztwie był w większości (bo zawsze jest większość). Chociażby GKL mocno z tego korzysta.
 - W przypadku 3-stanów jest jednak problem. Jaki powinien być wynik $\text{maj}(0,1,2)$? Żaden stan nie jest tu w większości! Co gorsza powiększanie promienia nie pomoże, co zawsze może być jakiś “remis”!

State-majority classification

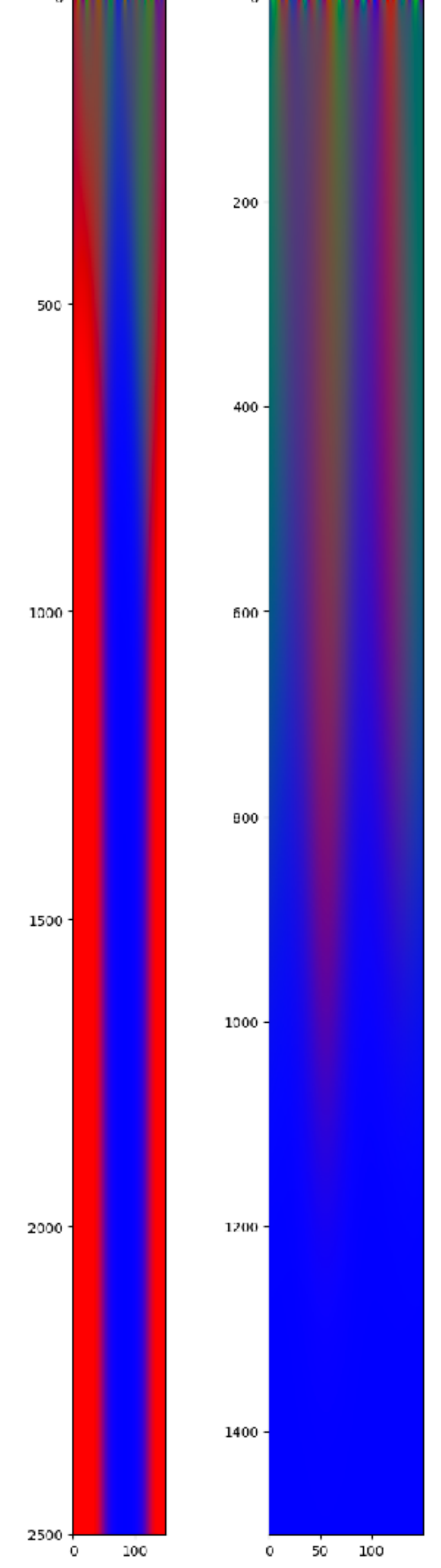
- **GKL**: póki co **nie ma** dobrego uogólnienia przez problemy z majority
- **Fukś**: nie ma jednego traffic tylko 6 - można je stosować na przemienne co pozwala uzyskać podobny efekt jak u Fuksia (równomierne rozproszenie stanów), ale ze względu na problemy z majority - **nie działa**
- **Fates**: stochastic mixture różnych trafficów i majority... **nie działa** z uwagi na problemy z majority.
- **Relaxed-ACCA**: działa! Reprezentacja wektorowa 3-stanowych ACCA. Mieszanki shiftów to średnie ważone. Tak jak w przypadku 2-stanowym, zbiegają do homogenicznego wektora zawierające gęstości poszczególnych stanów.
- **ACCA fixed-length**: wygląda na to, że **działa**! Trzeba mieszać kilka trafficów (np. 3 różne) z majority, ale w wersji ACCA i dostajemy piękne space-time diagramy!





Czasem trzeba poczekać długo na zbieżność...

- Po lewej traffic-majority ACCA
- Po prawej average-majority ACCA
- Jak widać average-majority ACCA nie zawsze dobrze sobie radzi :)
- **Badania trwają...**



In the case of a CCA, the local rule is a vector function $f: \mathcal{S}_c^R \rightarrow \mathcal{S}_c$, given by $f = (f_1, \dots, f_N)$, for which there exists a matrix $P \in \mathcal{S}_c^{N^R}$, referred to as the generalized LUT, such that for $j \in \{1, \dots, N\}$ it holds that $f_j: \mathcal{S}_c^R \rightarrow [0, 1]$ and:

$$f_j(s_1, \dots, s_R) = \sum_{i=1}^{N^R} P_{ij} \left(\prod_{m=1}^R s_{m, \text{ind}(i)[m]} \right) . \quad (18)$$

Wszystkie możliwe iloczyny
poszczególnych współrzędnych
argumentów...

```

def make_index():
    ... result = []
    ... for i in range(0, 27):
    ...     inds = np.base_repr(i, base=3).zfill(3)
    ...     result.append(np.array([int(digit) for digit in inds]))
    ... return np.array(result)

idx = make_index()

def eval_acca(mlut, conf):
    ... left = np.roll(conf, -1, axis=0)
    ... right = np.roll(conf, 1, axis=0)
    ... center = conf
    ...
    ... result = np.zeros_like(conf)
    ...
    ... for i in range(27):
    ...     prod = left[:, idx[i, 0]] * center[:, idx[i, 1]] * right[:, idx[i, 2]]
    ...     Pi = mlut[26 - i]
    ...     result += prod[:, np.newaxis] * Pi[np.newaxis, :]

    ... # this should not be needed.
    ... # but it helps with numerical stability
    ... result[:, 2] = 1 - (result[:, 1] + result[:, 0])

    ... return np.clip(result, 0.0, 1.0)

```

Błędy zaokrągleń 🤯

Dziękuję bardzo
Witold.Bolt@ug.edu.pl

