

# Automaty Komórkowe

## Wykład 9

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

Witold Bołt, 08.05.2024

# 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 (wielomiany, wyrażenia logiczne), problem klasyfikacji gęstości (DCP).
- **Dwa tygodnie przerwy** 🥲
- **Wykład 5 (zdalny):** Algorytmy ewolucyjne - poszukiwanie automatów komórkowych o określonych własnościach
- **Wykład 6:** Stochastyczne automaty komórkowe - SCAs, pLUT,  $\alpha$ -ACAs, Diploid CAs, stochastic mixture, dekompozycja pLUT
- **Wykład 7:** Afiniczne Ciągłe Automaty Komórkowe - wielomiany, cLUT, relaxed DCP + bonus - praca w IT w Trójmieście (i nie tylko)
- **Wykład 8:** Identyfikacja Deterministycznych Automatów Komórkowych

# Co będzie dalej\*

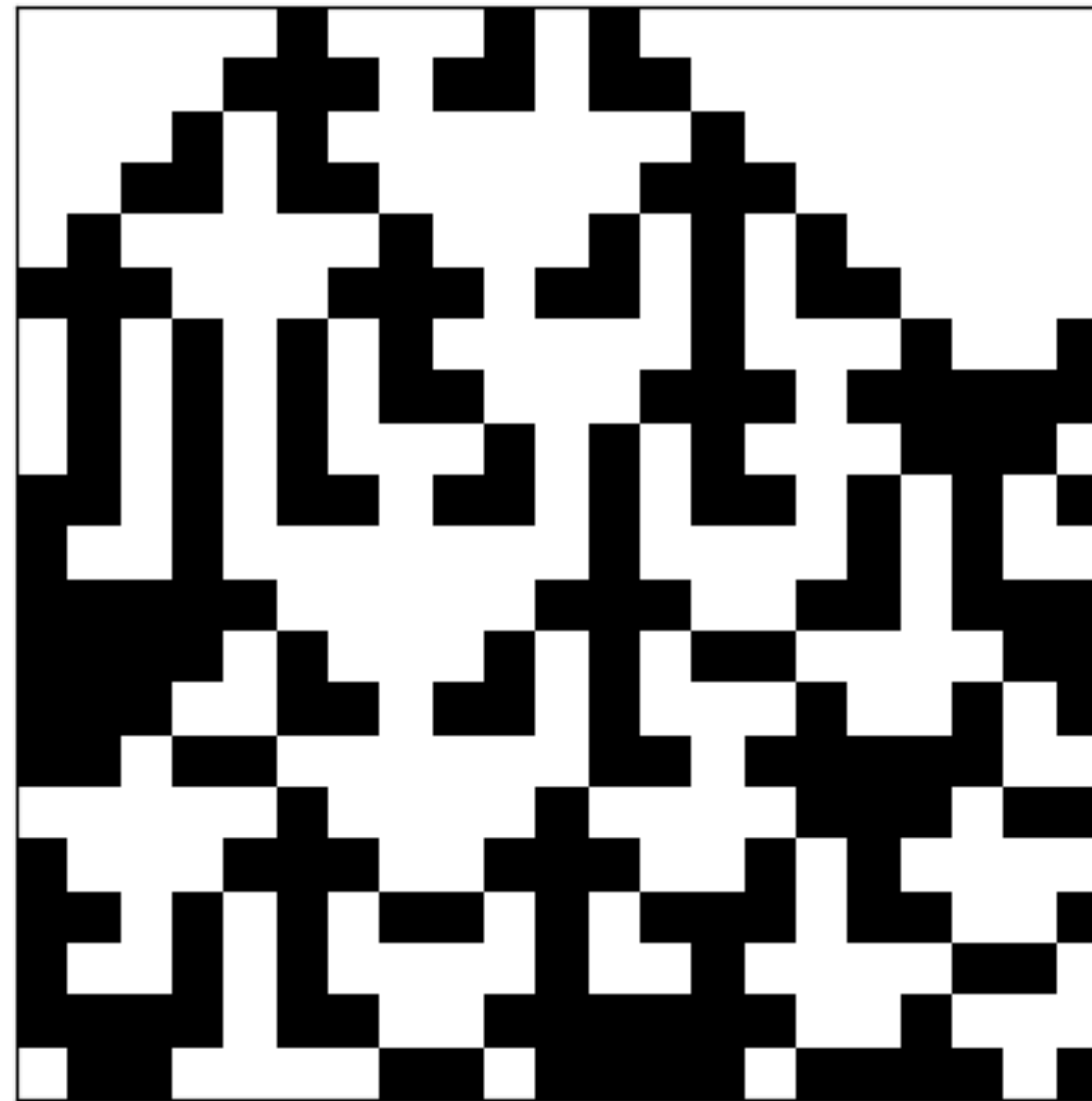
- **Wykład 9:** Identyfikacja Stochastycznych Automatów Komórkowych
- **Wykład 10:** Dwu-wymiarowe Automaty Komórkowe / Reguła Life i Life-like / Totalistyczne i Zewnętrzne-Totalistyczne Automaty Komórkowe (totalistic & outer-totalistic CAs)
- **Wykład 11:** Automaty Komórkowe zachowujące gęstość
- **Wykład 12:** Nie-jednorodne (non-uniform) Automaty Komórkowe; Zastosowania w modelowaniu ruchu ulicznego
- **Wykład 13:** Modele pożaru lasu, rozprzestrzeniania się epidemii, Greenberg–Hastings i podobne modele
- **Wykład 14:** Modele c.d. - w tym: lattice-gas, diffusion
- **Wykład 15:** Neural CAs (Neuronowe Automaty Komórkowe)

Za tydzień 15.05 zajęcia zdalne

# Identyfikacja Stochastycznych Automatów Komórkowych

# Przypominajka: przypadek deterministyczny

- Dana jest obserwacja space-time diagramu, czyli albo cały space-time diagram albo jakiś jego “fragment” (pod-diagram?).
- W najprostszym przypadku mamy cały diagram - na przykład taki:



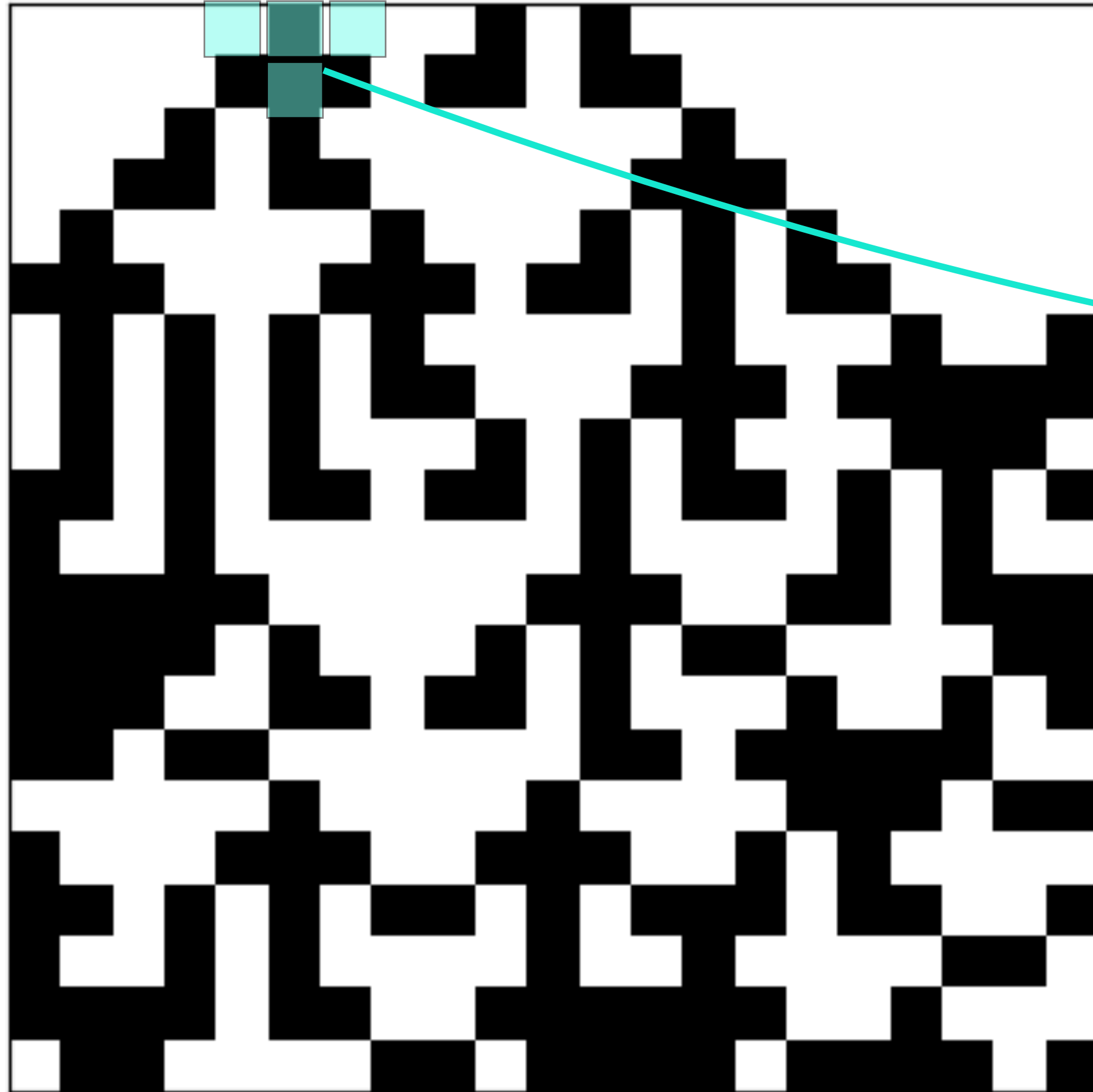
- **Zadanie:** rozpoznać / odczytać, który CA wygenerował ten diagram.

The grid world environment is a 10x10 grid. The start state is located at (1, 1) and the goal state is at (2, 9). A red line indicates a path from the start to the goal, passing through (2, 10), (3, 10), (4, 10), (5, 10), (6, 10), (7, 10), (8, 10), (9, 10), and (10, 10). The path is blocked by obstacles at (3, 9), (4, 9), (5, 9), (6, 9), (7, 9), (8, 9), (9, 9), and (10, 9).

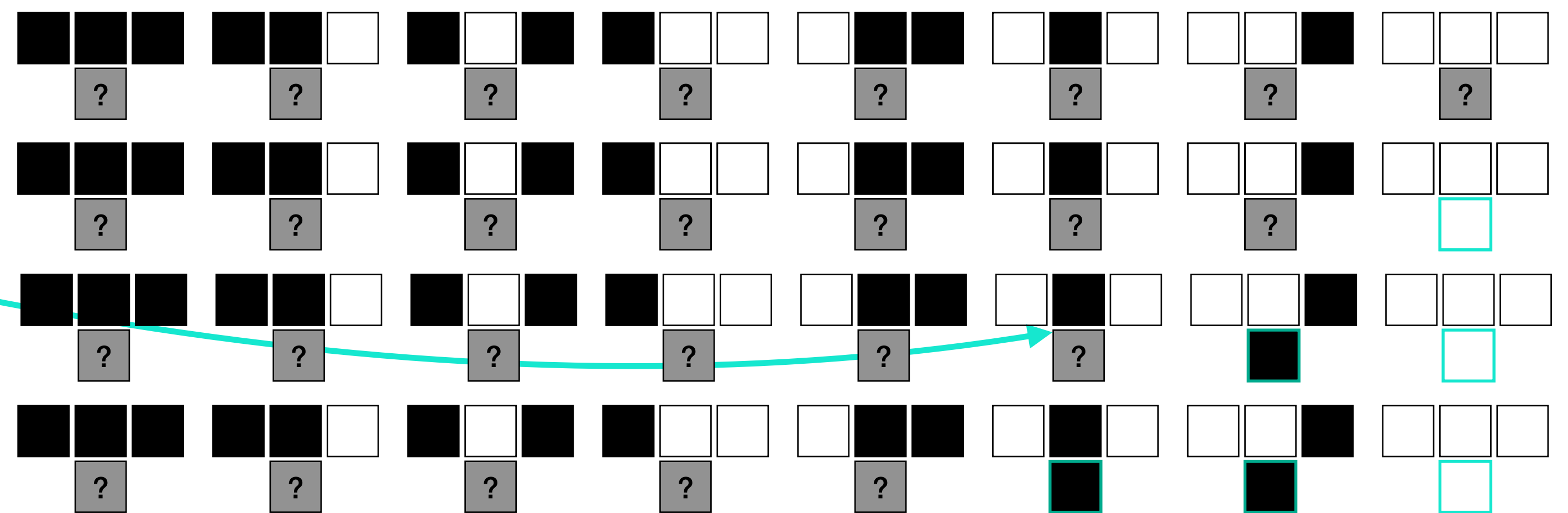




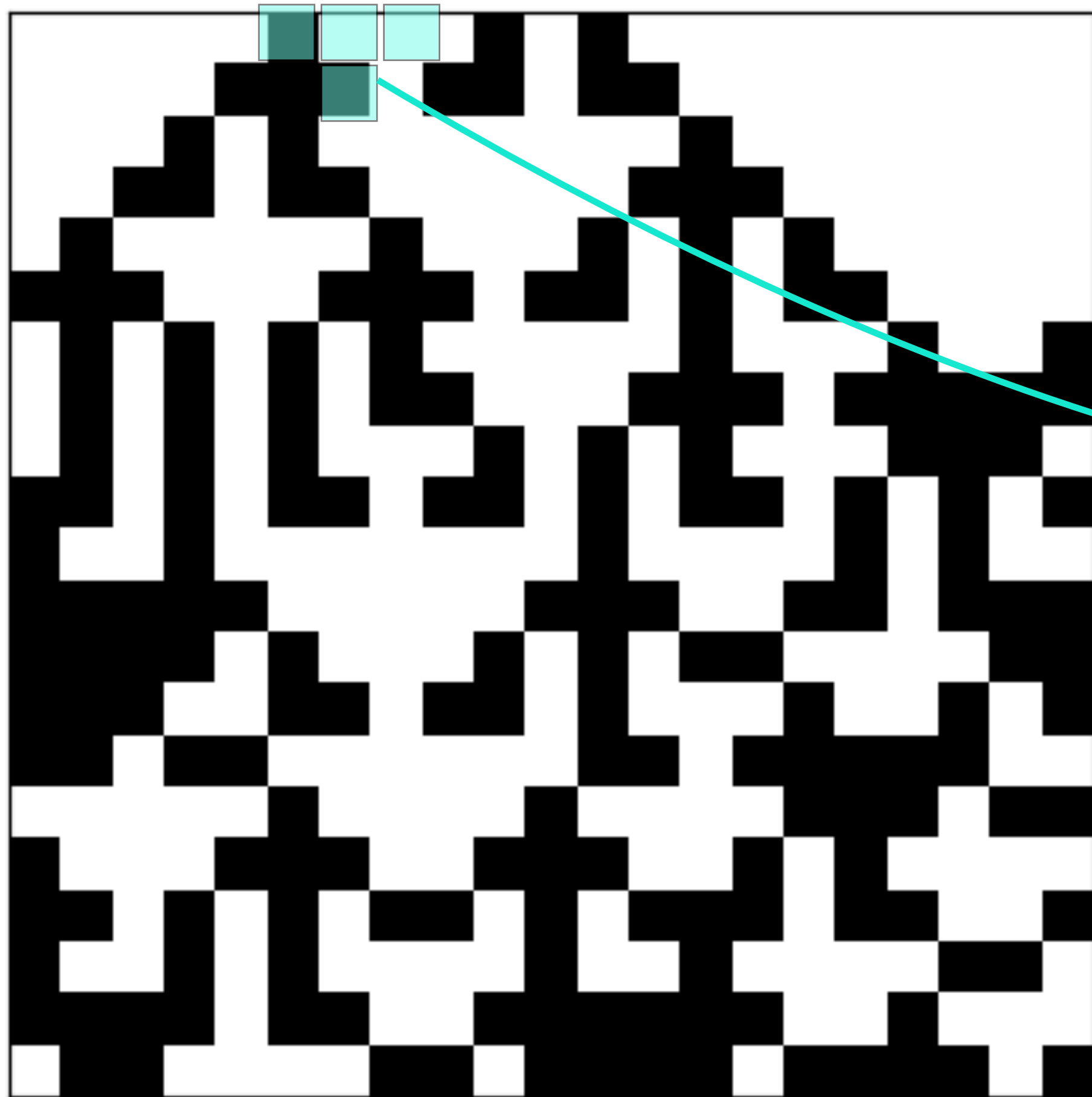
space-time diagram



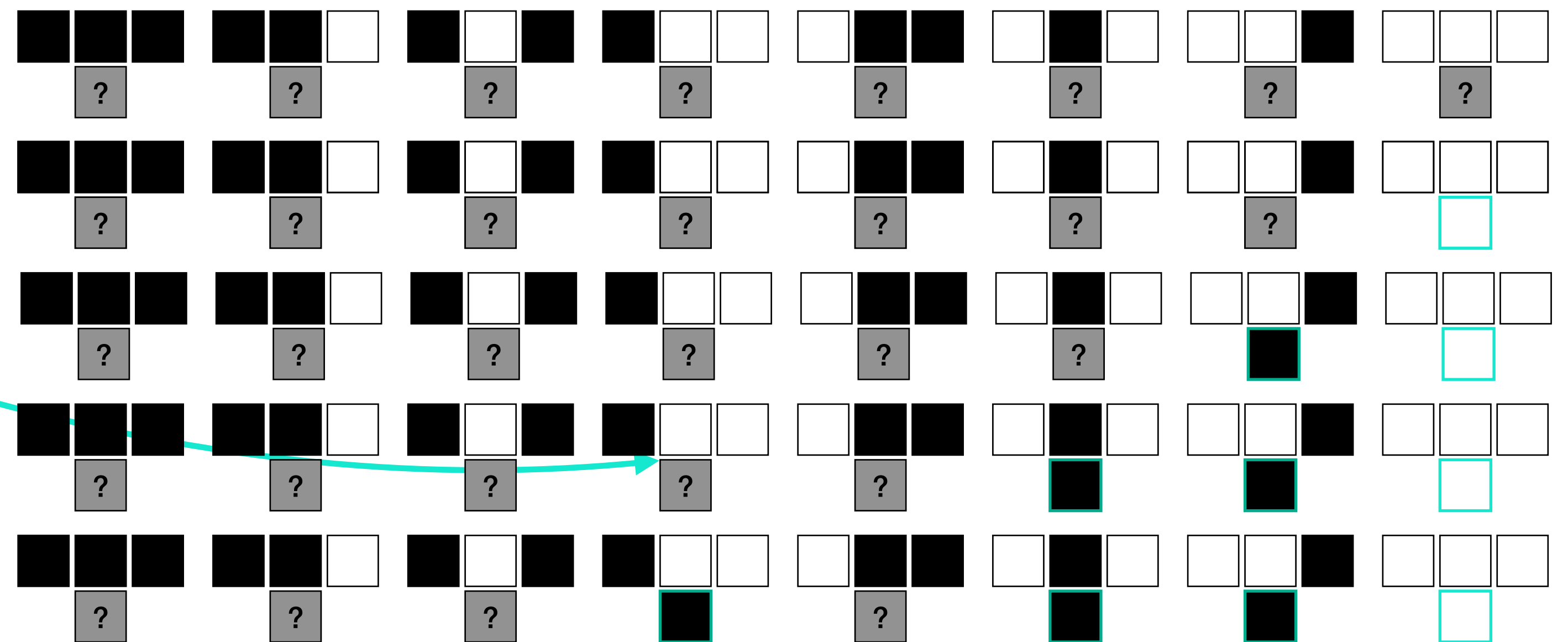
LUT



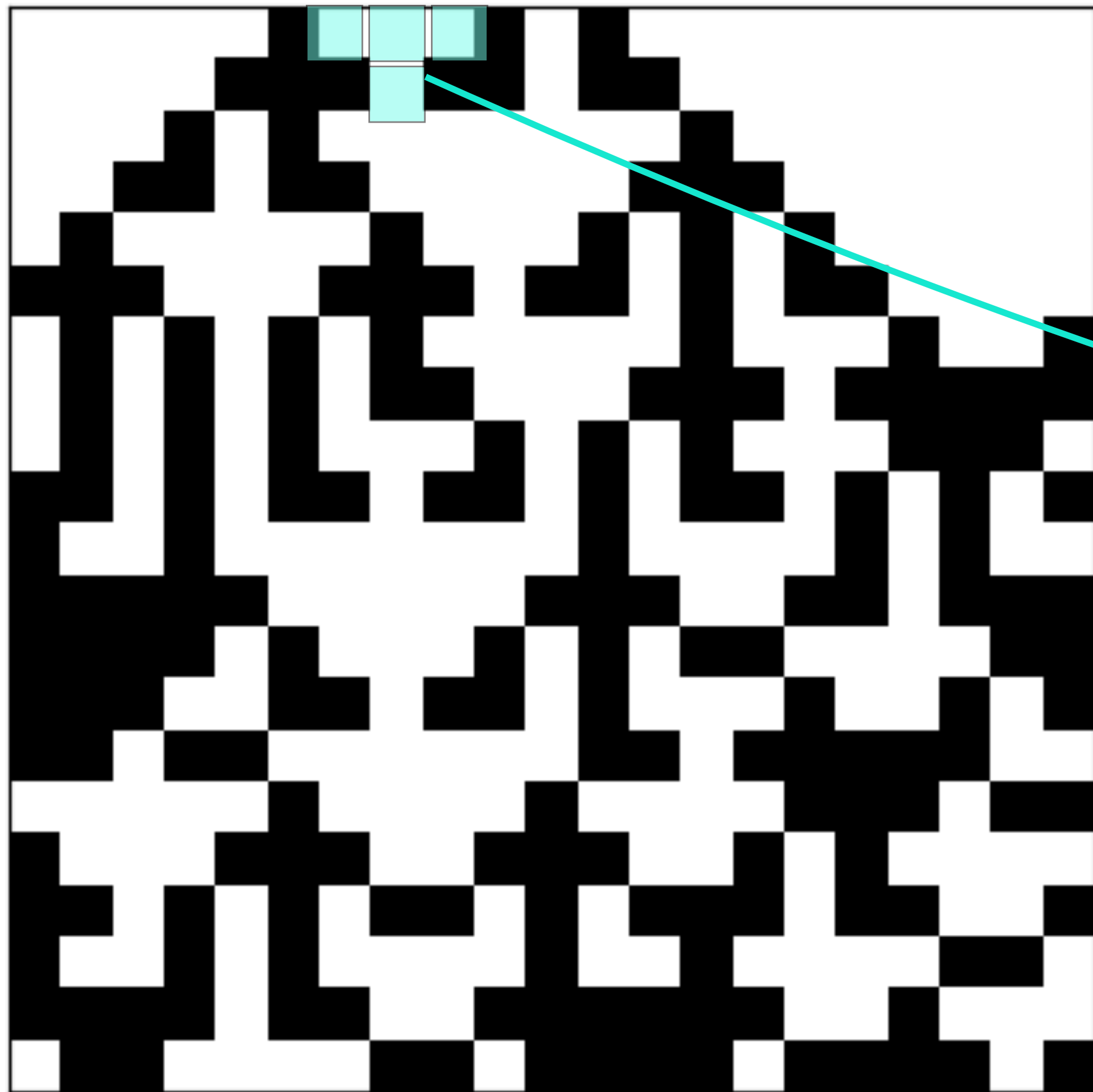
space-time diagram



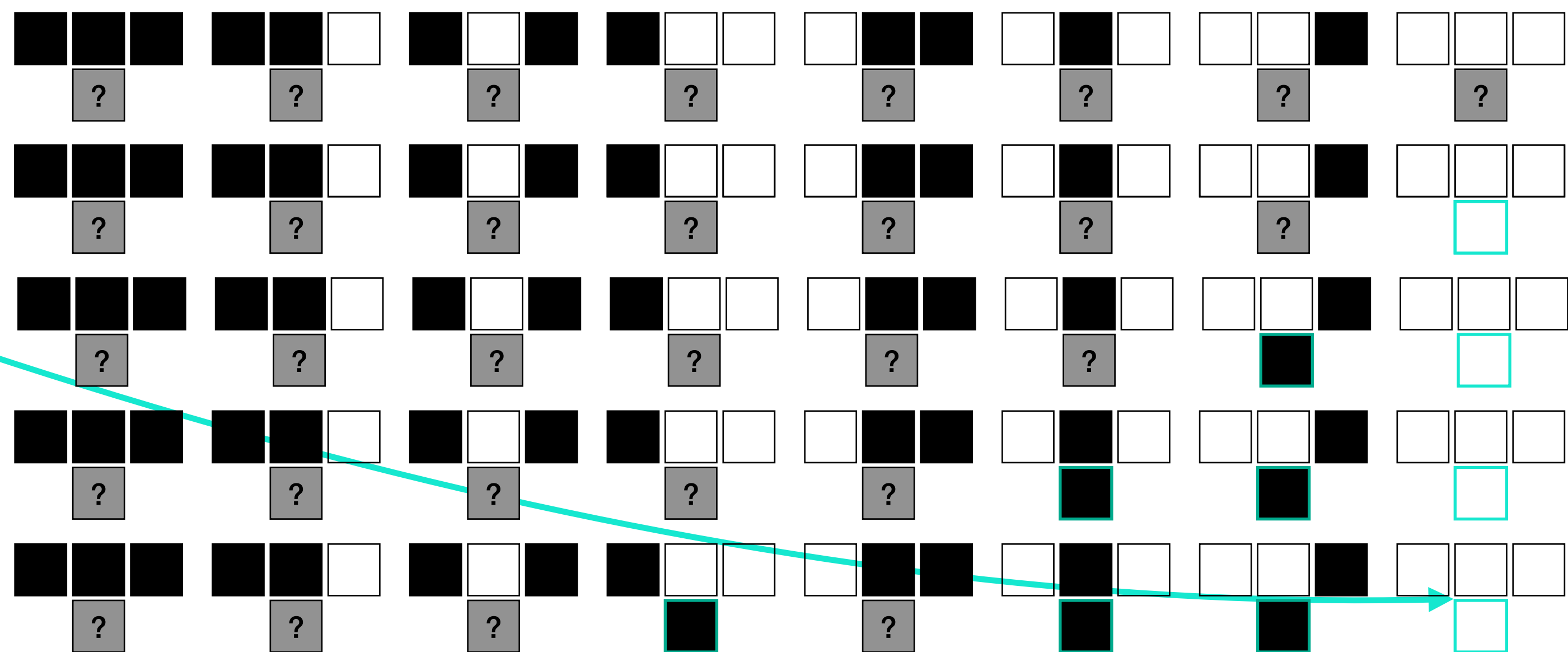
LUT



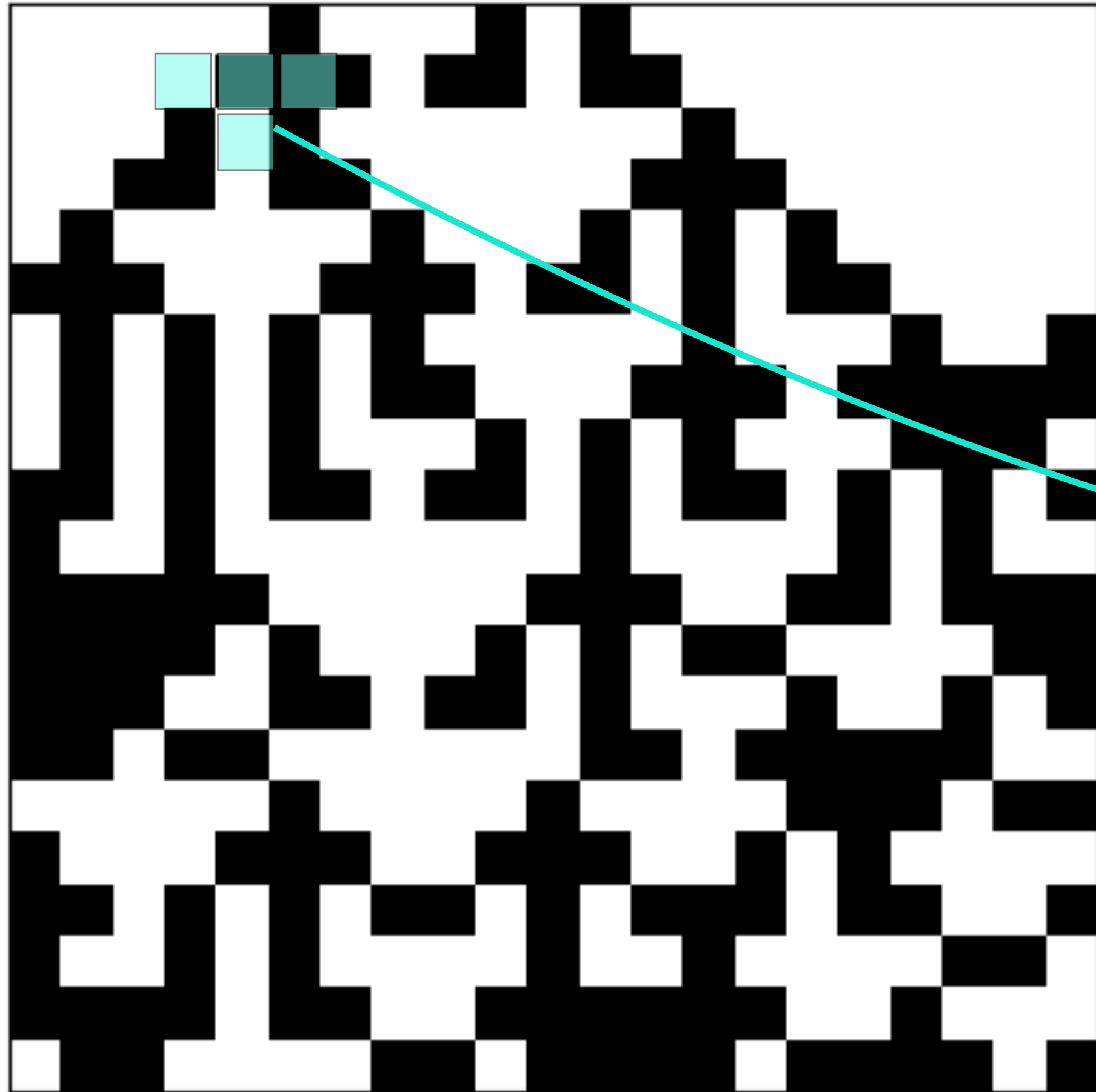
space-time diagram



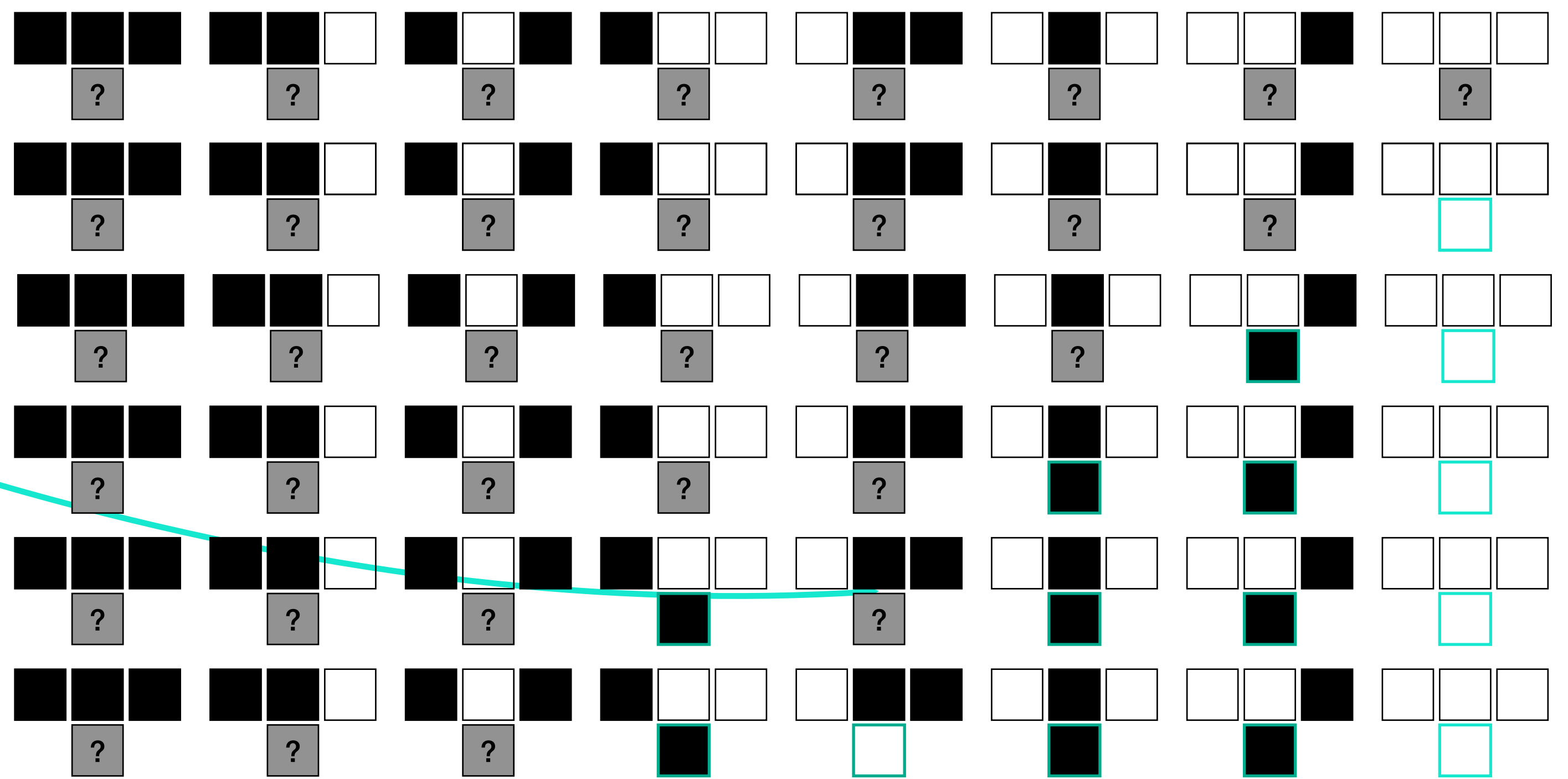
LUT



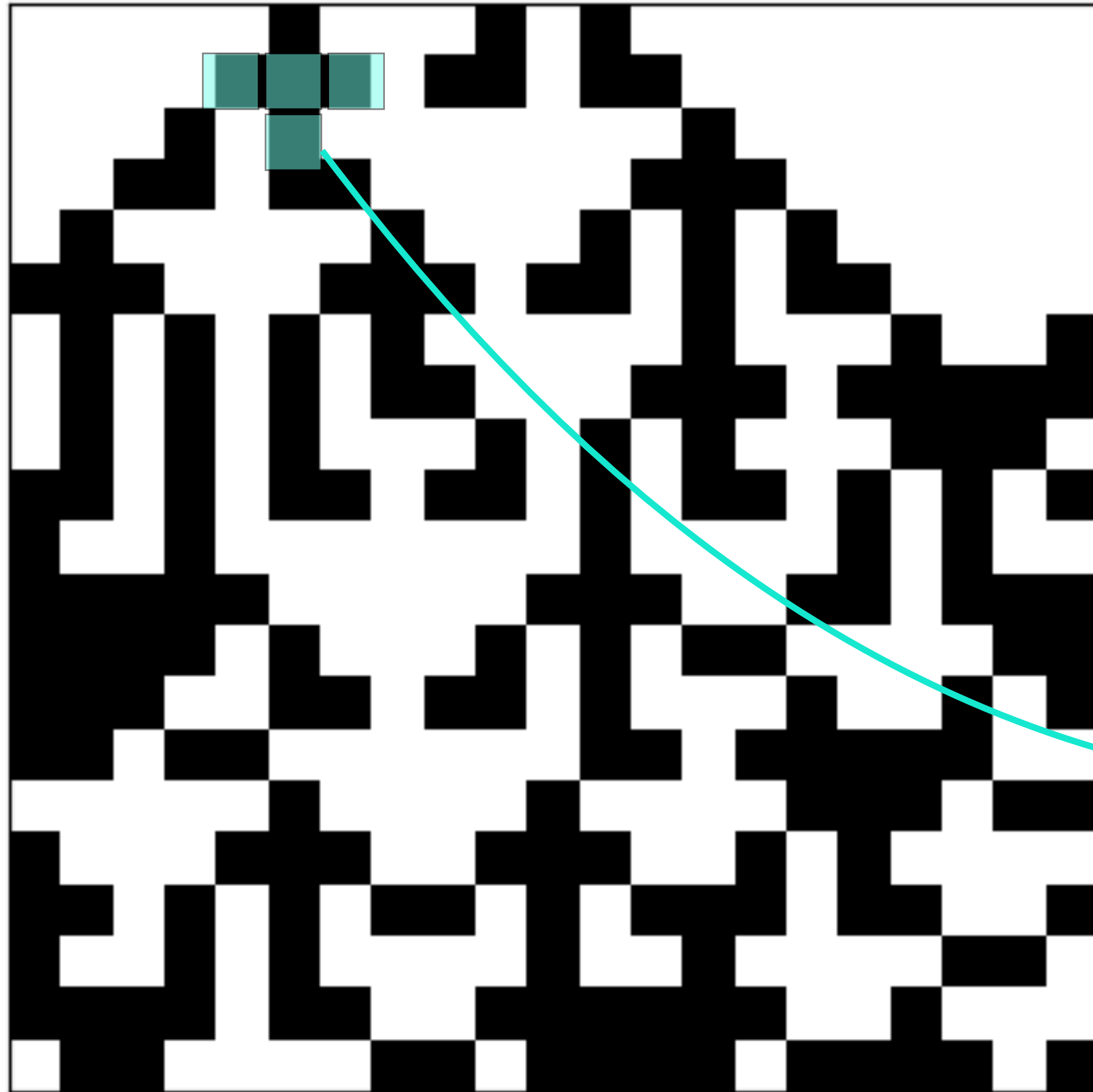
space-time diagram



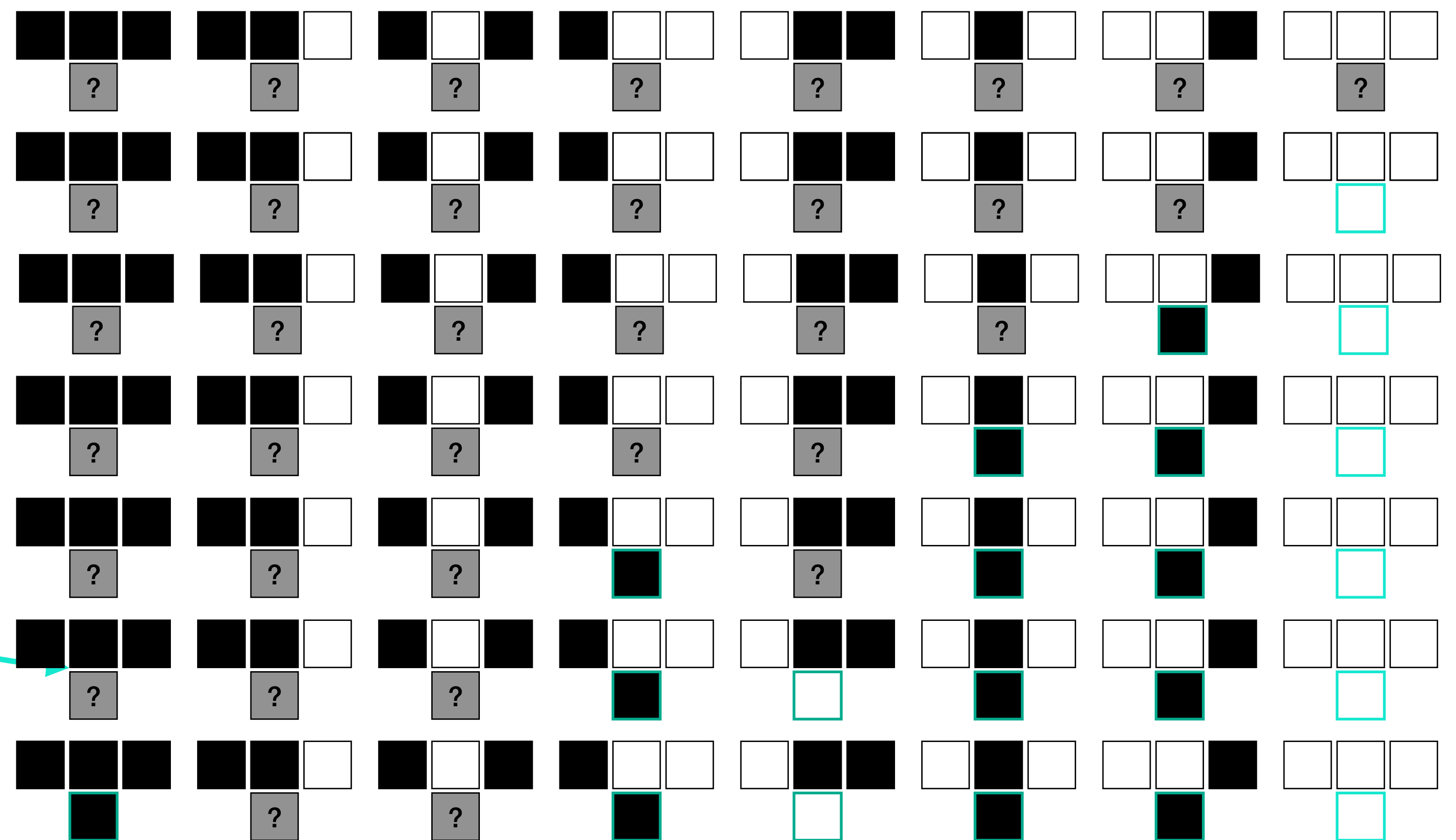
LUT



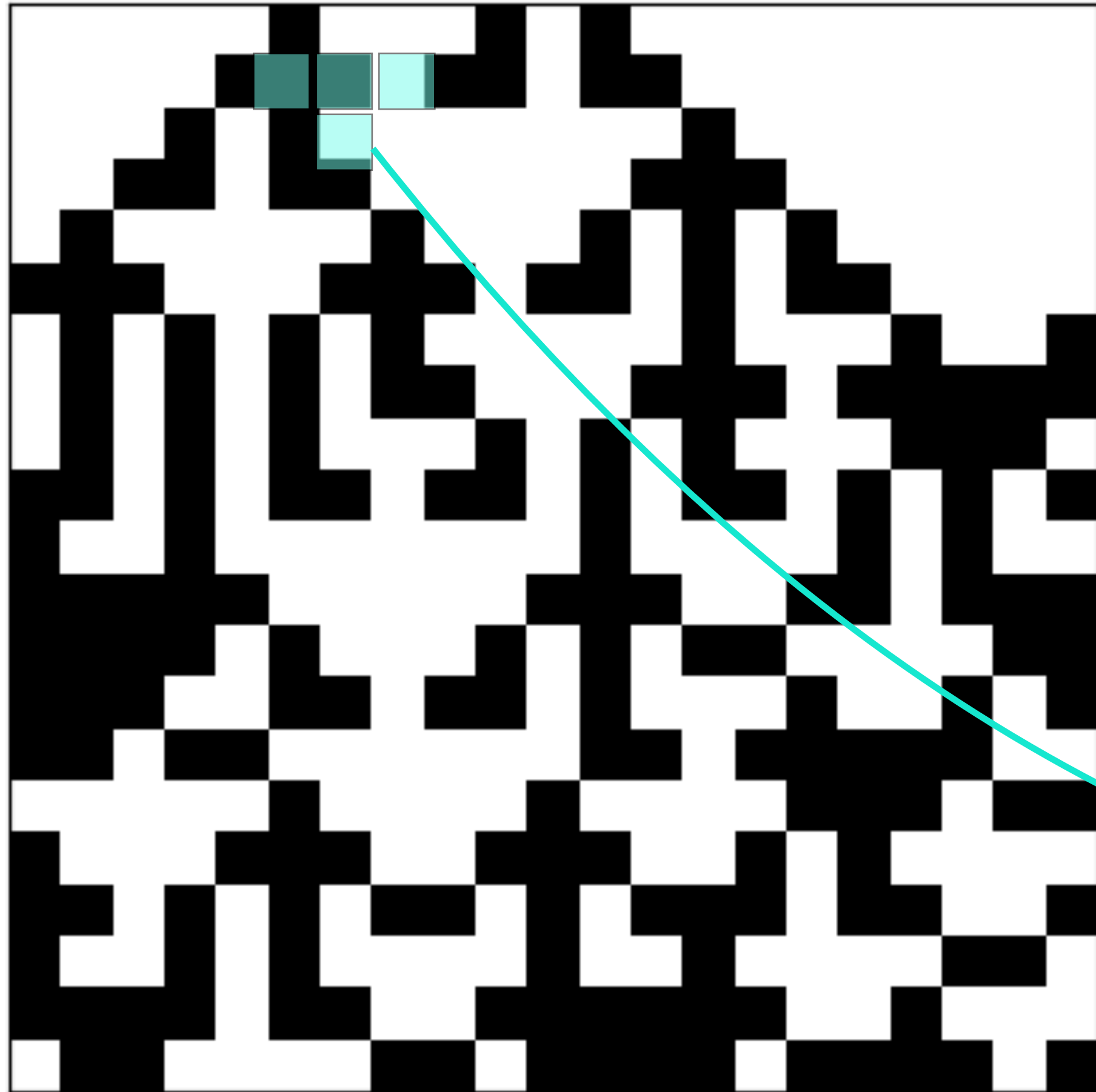
space-time diagram



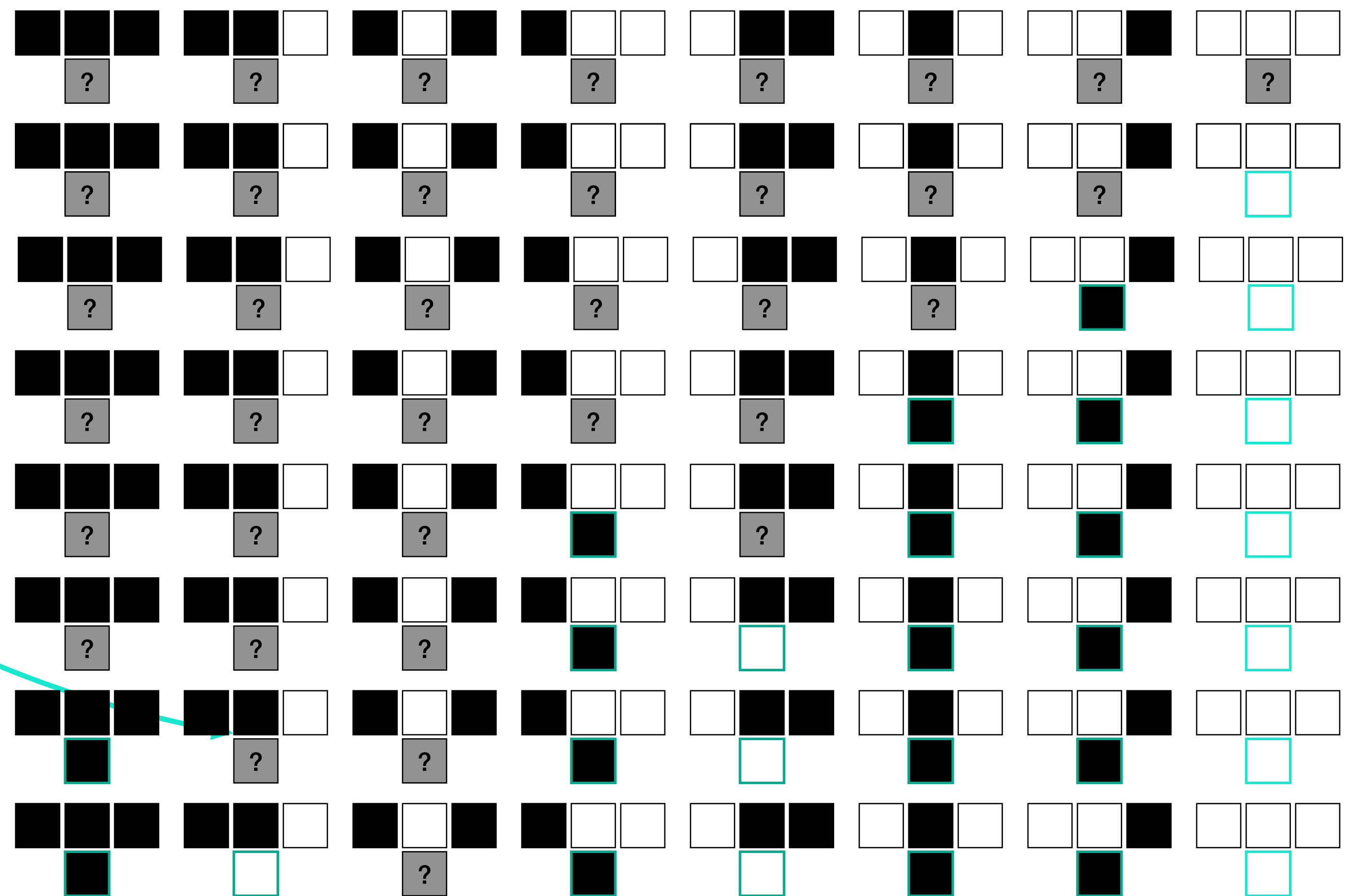
LUT



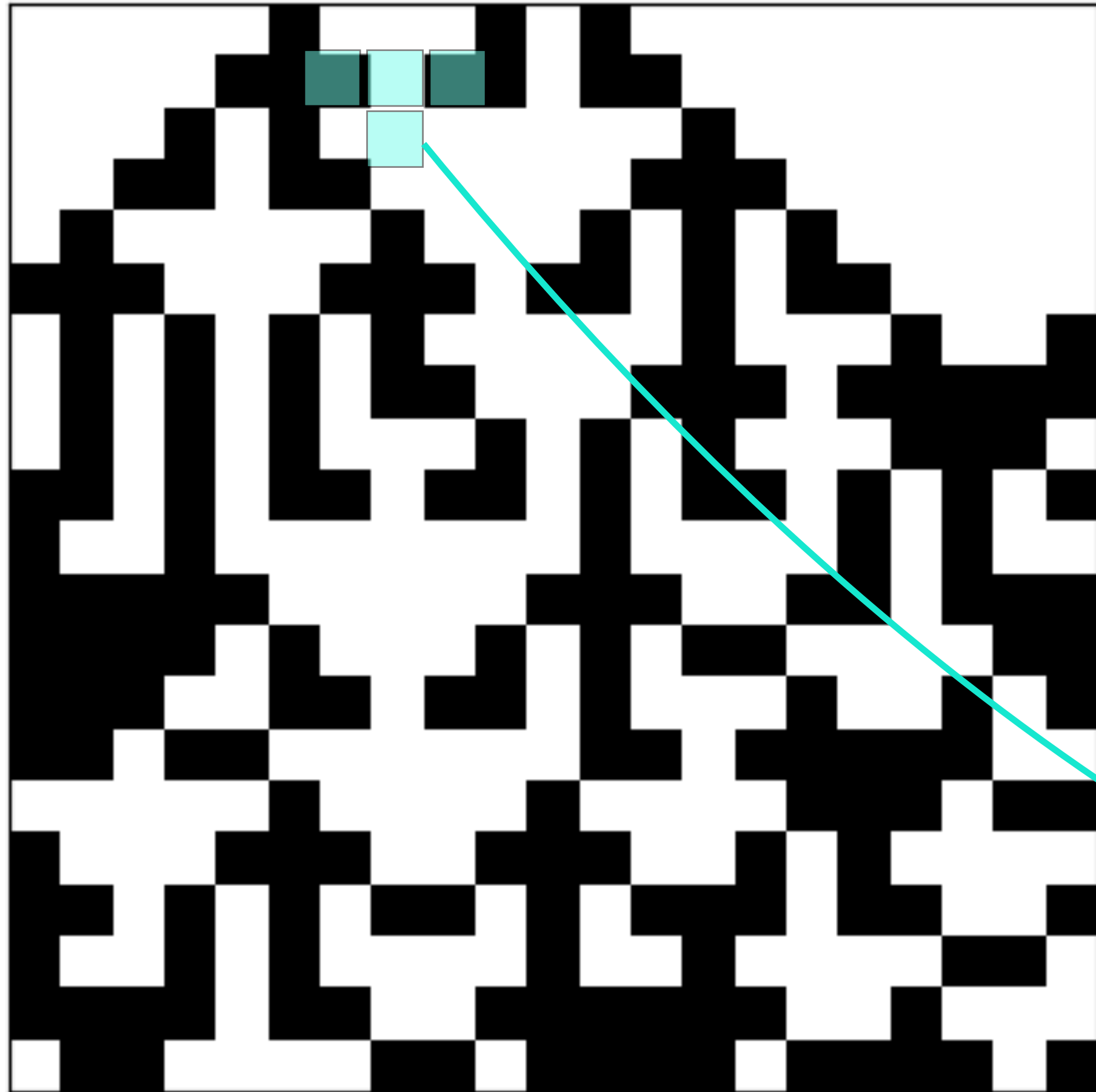
space-time diagram



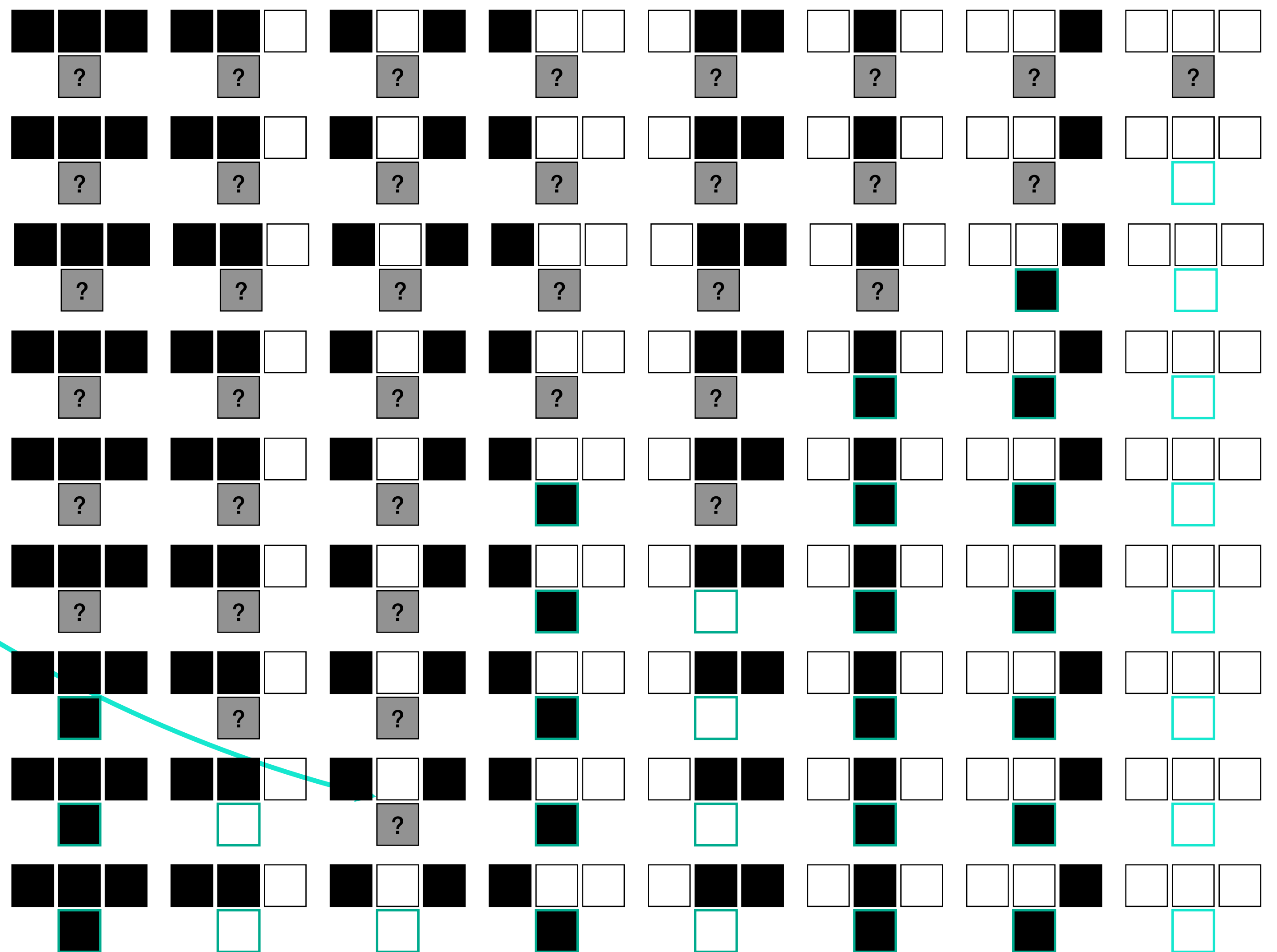
LUT



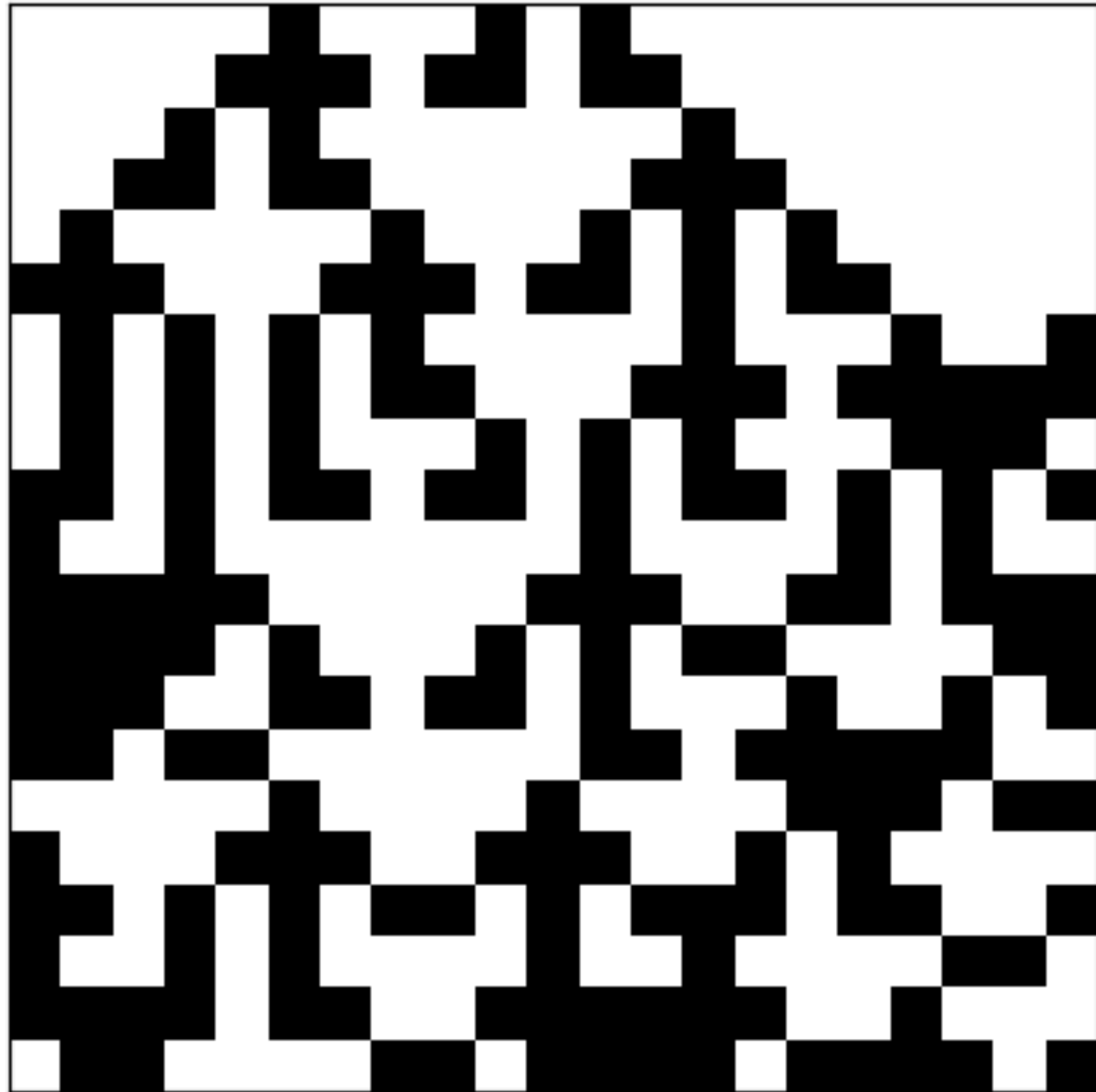
space-time diagram



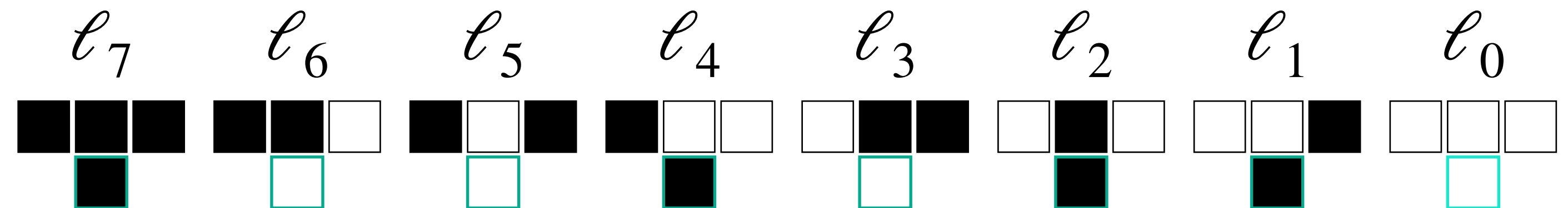
LUT



space-time diagram



LUT



$$\ell = (1, 0, 0, 1, 0, 1, 1, 0)$$

ECA 150 🍑



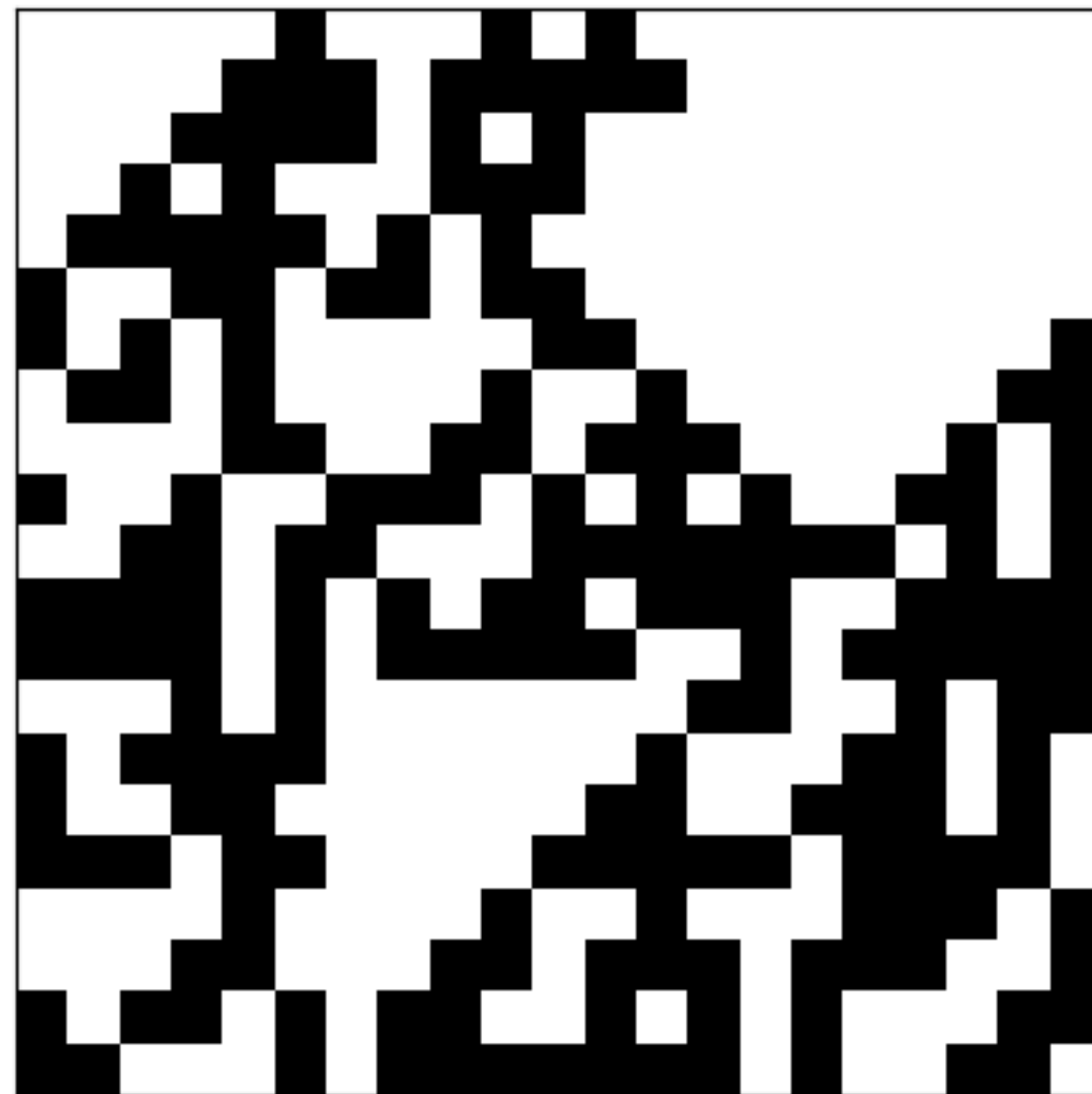
# Co mogło się nam nie udać?

- Obserwacja nie miała “dziur” - wszystkie stany dało się odczytać wprost z obrazka.
- Wszystkie konfiguracje sąsiedztw były dostępne, więc nie było wieloznaczności w odpowiedzi.
- Nie było brakujących kroków czasowych (*time-gaps*) - dlatego nie musieliśmy korzystać z algorytmu ewolucyjnego.
- **Założyliśmy**, że promień sąsiedztwa wynosił 1, czyli szukaliśmy ECA i okazało się, że **mieliśmy rację**, bo diagram został faktycznie wygenerowany przez pewien ECA.
- Oczekiwaliśmy deterministycznego CA, więc **nie musieliśmy** odczytywać całego diagramu naszym “skanerem”.

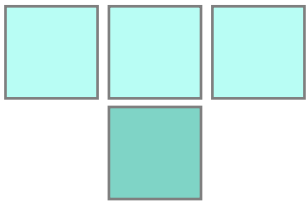
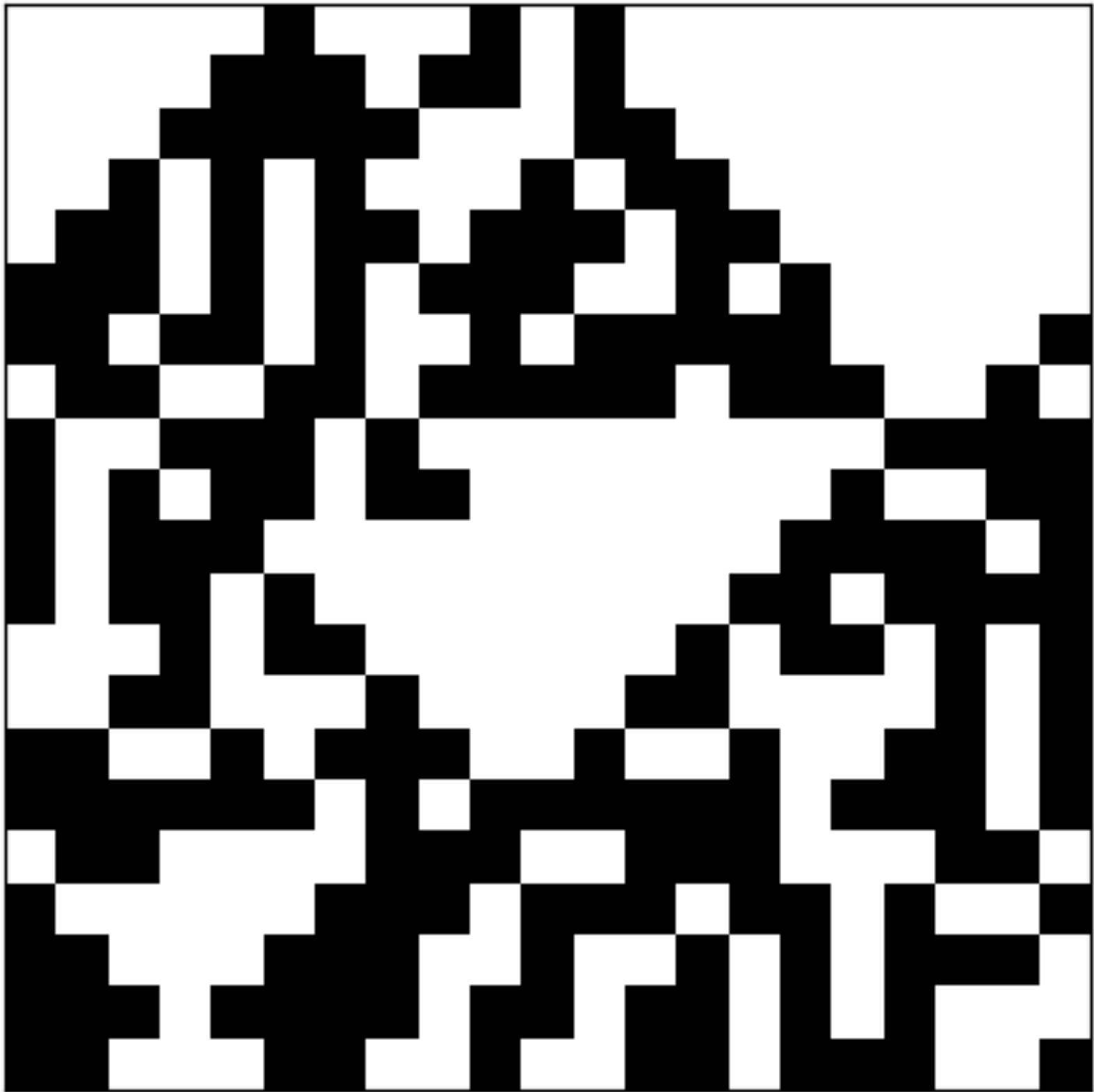


# Identyfikacja SCAs

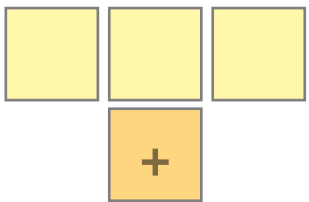
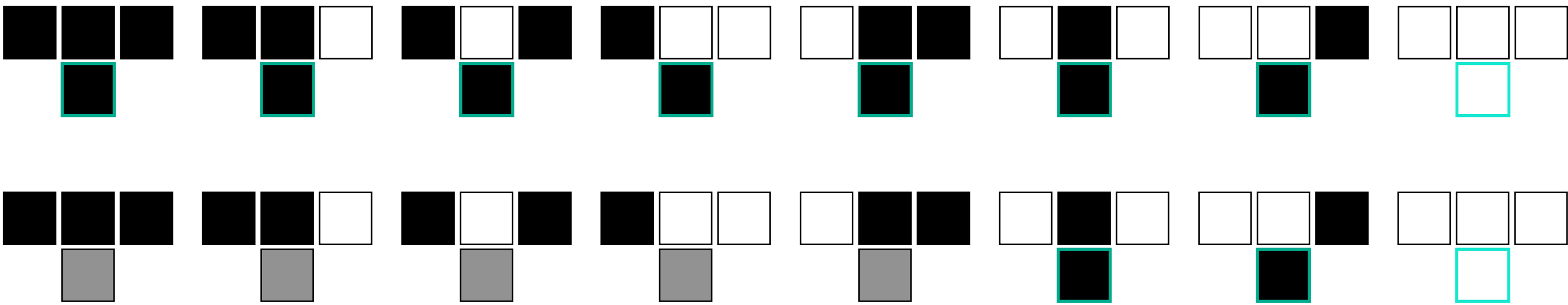
- Dana jest obserwacja space-time diagramu pewnego SCA.
- W najprostszym przypadku mamy cały diagram - na przykład taki:



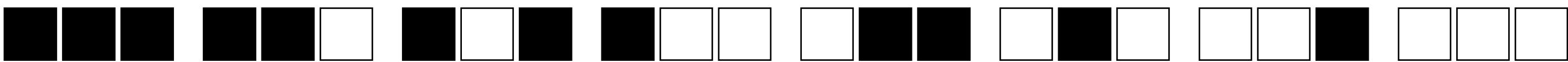
- **Zadanie:** oszacować prawdopodobieństwa w pLUT tego SCA!



**skaner** - odczytuje pojedyncze wartości

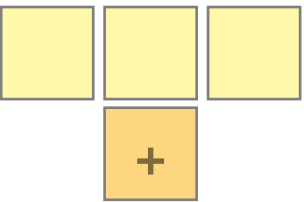
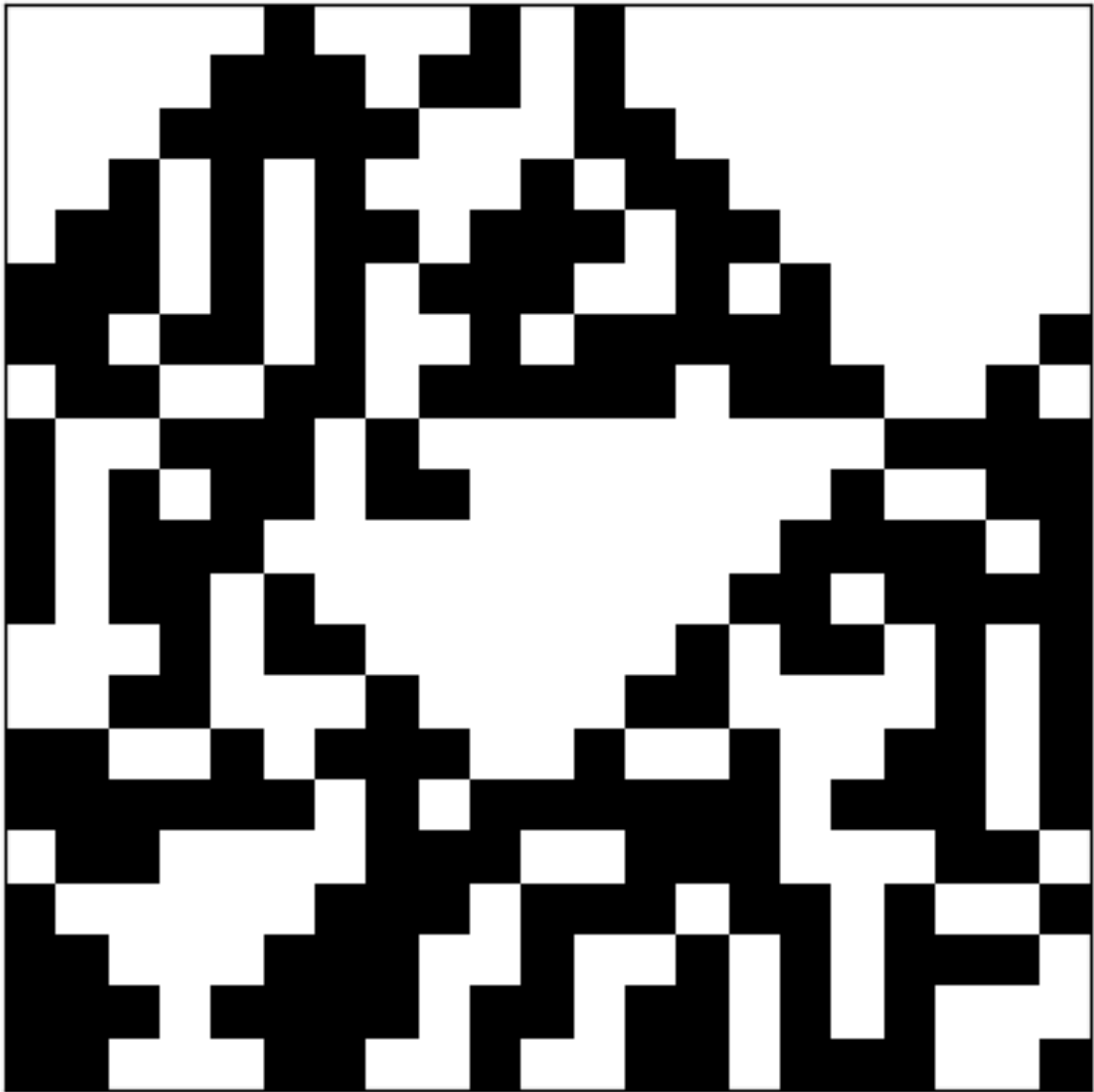


**licznik** - zlicza **wystąpienia** wartości



	19	31	33	12	29	0	0	96
□	19	31	33	12	29	0	0	96
■	28	23	17	29	25	37	41	0

Oczywiście  
"licznik" musi  
przejeść cały  
diagram!



licznik - zlicza wystąpienia wartości

	19	31	33	12	29	0	0	96		
	28	23	17	29	25	37	41	0		

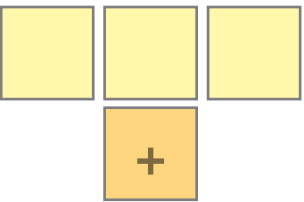
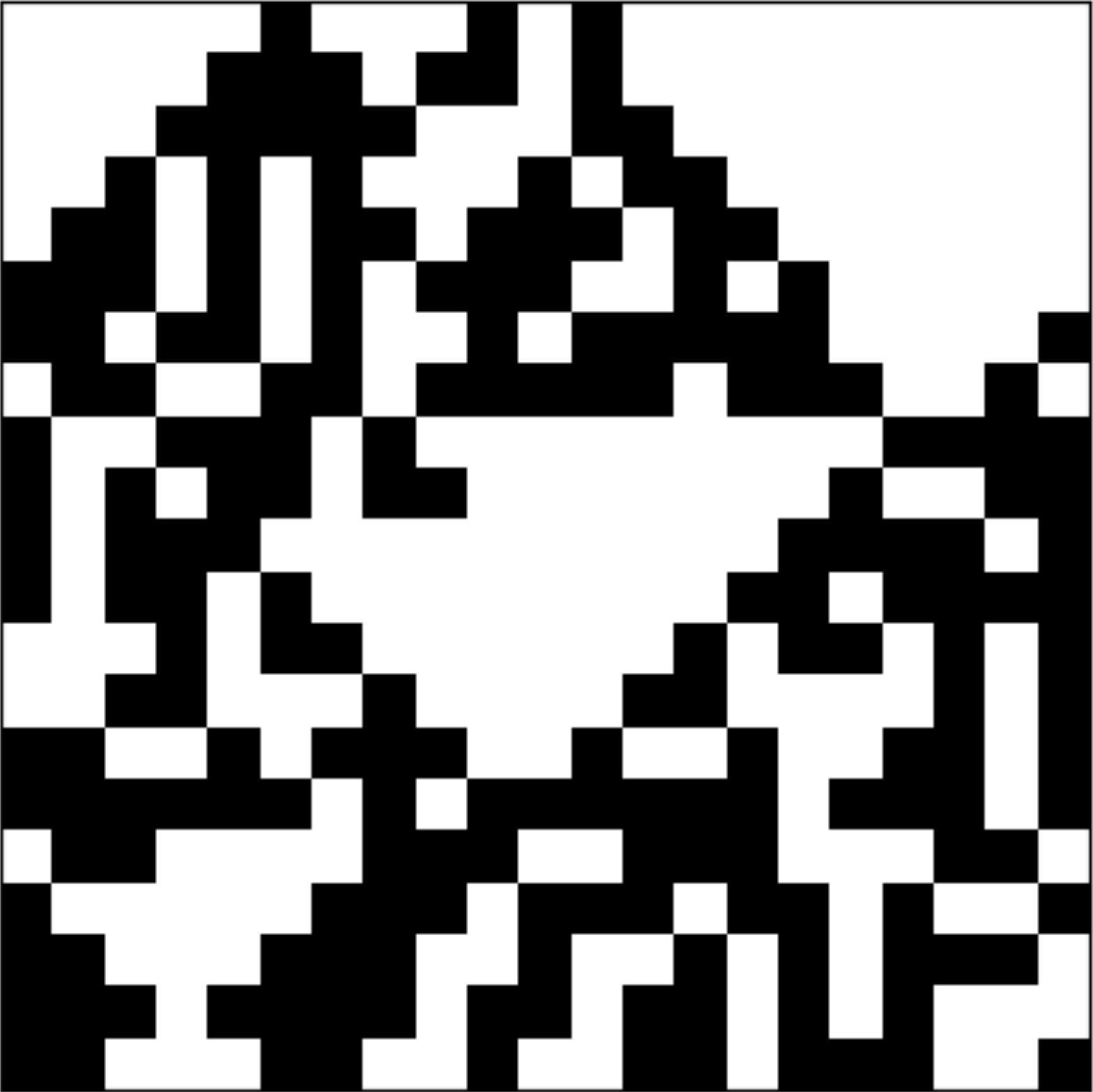
Zauważmy, że wystąpiły **trzy** przypadki:

1. Wynik **zawsze** był 0 (sąsiedztwo 0,0,0).
2. Wynik **zawsze** był 1 (sąsiedztwa 0,1,0 i 0,0,1).
3. Wynik był **albo** 0 **albo** 1 - pozostałe sąsiedztwa.

Szczęśliwie **nie** wystąpił (choć mógł) czwarty przypadek, to znaczy, że danej konfiguracji sąsiedztwa w ogóle nie zaobserwowano.

# Co wynika z naszych przypadków?

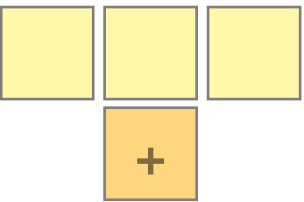
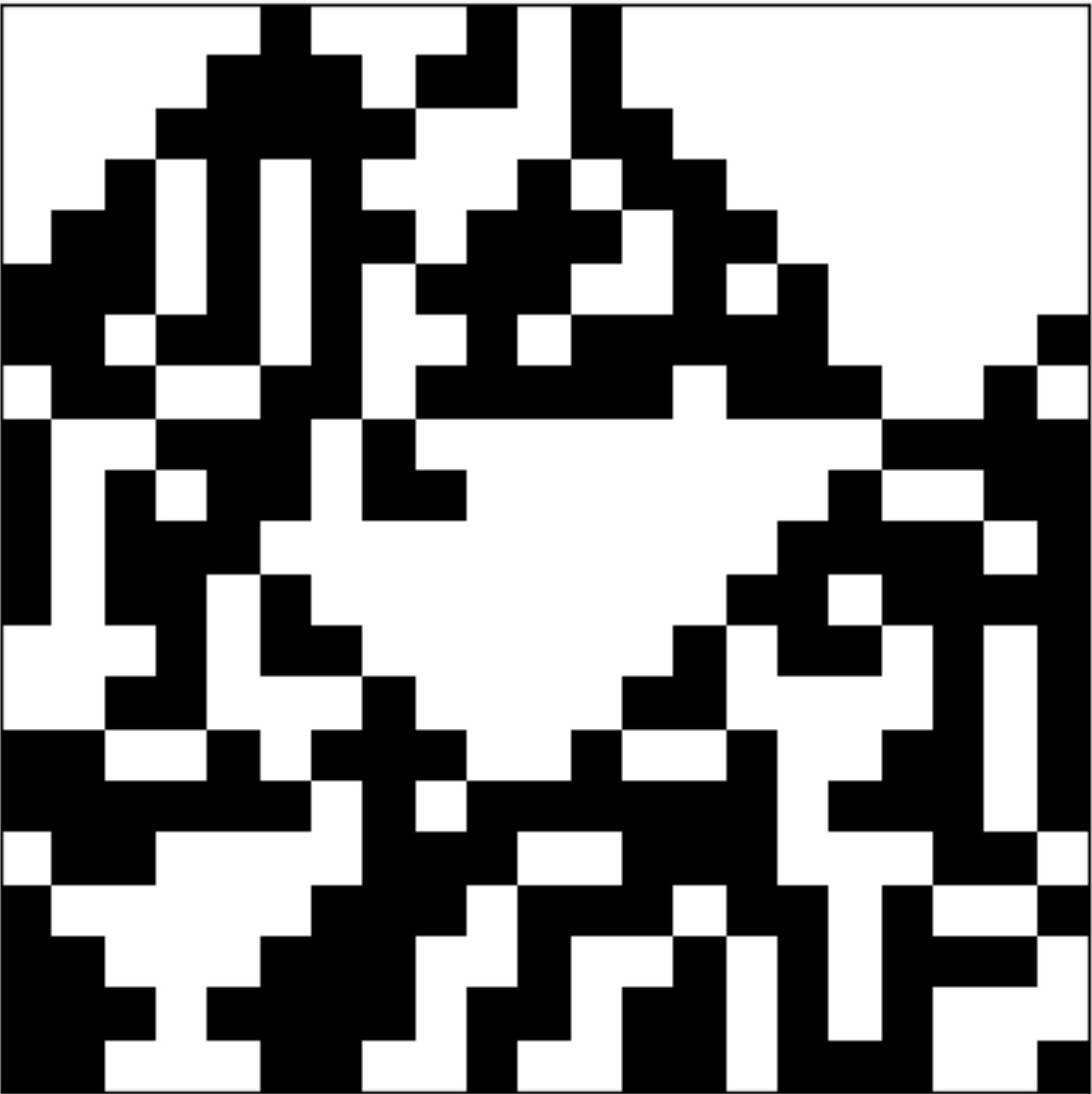
- W przypadku gdy dla danej konfiguracji sąsiedztwa (załóżmy, że jest to sąsiedztwo  $i$ ) zaobserwowaliśmy zarówno przejście na wartość 0 jak i 1 to **jesteśmy pewni**, że reguła na tym sąsiedztwie zachowuje się niedeterministycznie. Innymi słowy wiemy, że  $p_i \in (0,1)$ .
- Jeśli dla sąsiedztwa  $i$  zaobserwowano jedynie przejścia na wartość 0, to **albo**  $p_i = 0$  **albo**  $p_i > 0$  i w dodatku to  $p_i$  jest **raczej** bardzo małe.
- Jeśli dla sąsiedztwa  $i$  zaobserwowano jedynie przejścia na wartość 1, to **albo**  $p_i = 1$  **albo**  $p_i < 1$  i w dodatku to  $p_i$  jest **raczej** bardzo duże.
  - Słowo “raczej” użyto tu aby wskazać na subiektywność i niepewność zarazem. Jeśli ogólna liczba wystąpień danego sąsiedztwa była “mała” to tak na prawdę nie wiele możemy powiedzieć o  $p_i$  w tych przypadkach.



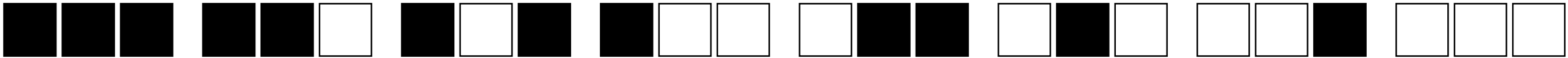
licznik - zlicza wystąpienia wartości

	19	31	33	12	29	0	0	96									
	28	23	17	29	25	37	41	0									
$\hat{p}_i$	0.596	0.426	0.34	0.707	0.463	1	1	0									

$$\hat{p}_i = \frac{\text{black square}}{\text{white square} + \text{black square}}$$



licznik - zlicza wystąpienia wartości



	19	31	33	12	29	0	0	96
	28	23	17	29	25	37	41	0
$\hat{p}_i$	0.596	0.426	0.34	0.707	0.463	1	1	0
$p_i$	0.6	0.4	0.4	0.6	0.4	1	1	0

Czy da się lepiej? 🤔

$$\frac{1}{8} \sum_i |p_i - \hat{p}_i| \approx 0.0326 \quad 😊$$

$$\max_i |p_i - \hat{p}_i| \approx 0.1073 \quad 😓 \quad \max_i \frac{|p_i - \hat{p}_i|}{p_i} \approx 0.1789$$

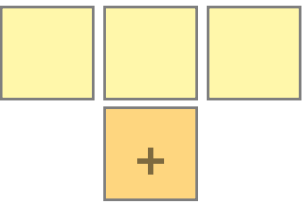
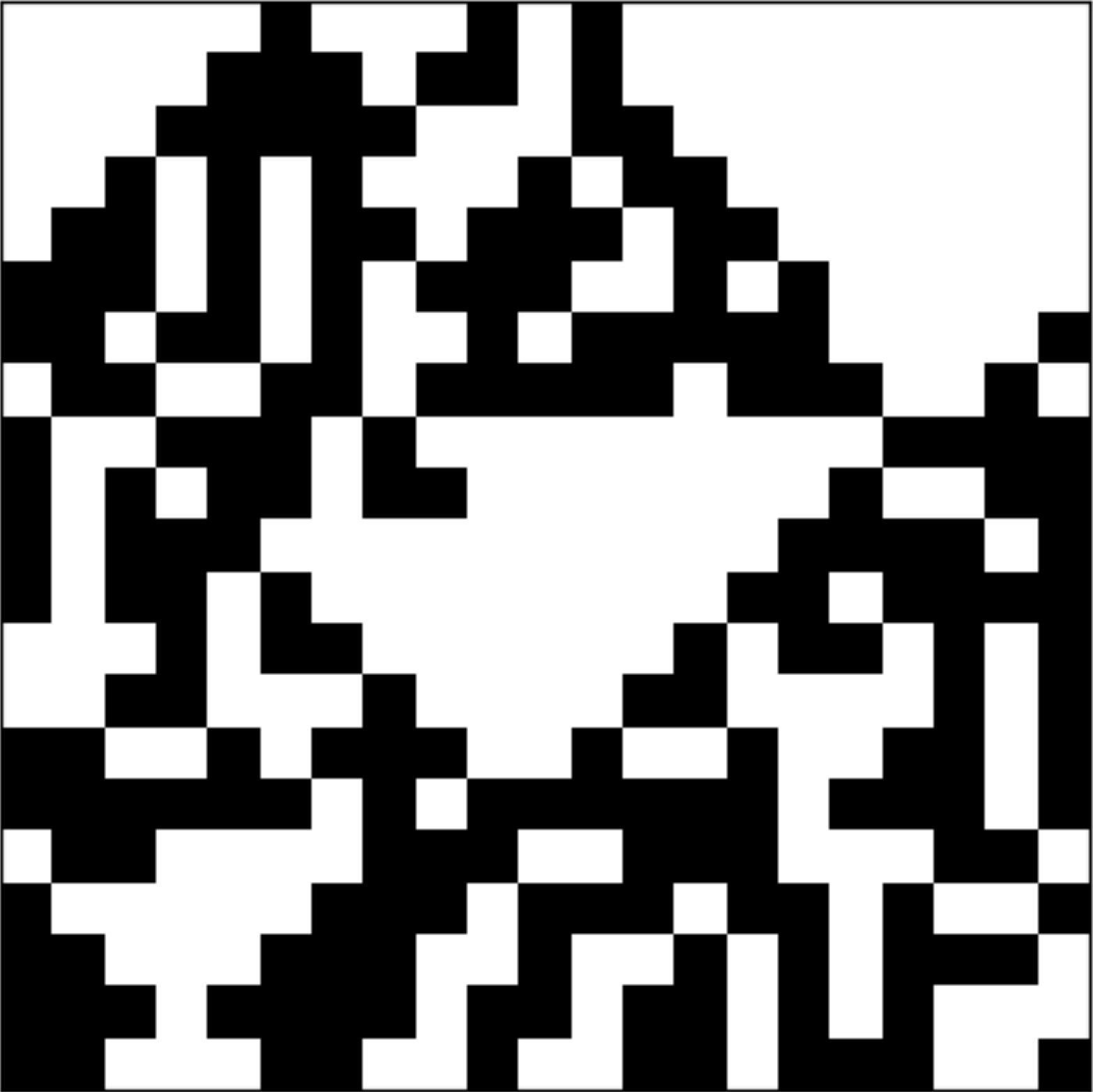


Diploid to SCA, który można zapisać  
jako stochastyczna mikstura dokładnie dwóch,  
dowolnych CAs.

A co by było gdybyśmy założyli,  
że to jest diploid? 🧛

# Identyfikacja diploidów?

- Diploid to mikstura dwóch CA. Założymy dla uproszczenia, że mamy miksturę dwóch ECA zdefiniowanych przez reguły lokalne  $f_1$  oraz  $f_2$ .
- Innymi słowy, w każdej chwili czasu, dla każdej komórki niezależnie, z prawdopodobieństwem  $\alpha \in (0,1)$ ,  $\alpha \neq 0.5$ , stosowana jest reguła  $f_1$  a z prawdopodobieństwem  $1 - \alpha$  reguła  $f_2$ .
- **Fakt.** W pLUT diploidu występują **jedynie** wartości ze zbioru:  $\{0,1,\alpha,1 - \alpha\}$ .
  - **Fakt.** Wartości 0 oraz 1 występują dla tych sąsiedztw dla których obie reguły są “zgodne”. Wartość  $\alpha$  występuje dla sąsiedztw dla których reguła  $f_1$  przyjmuje wartość 1, a reguła  $f_2$  wartość 0. Wartość  $1 - \alpha$  dla sąsiedztw dla których  $f_1$  przyjmuje wartość 0, a reguła  $f_2$  wartość 1.



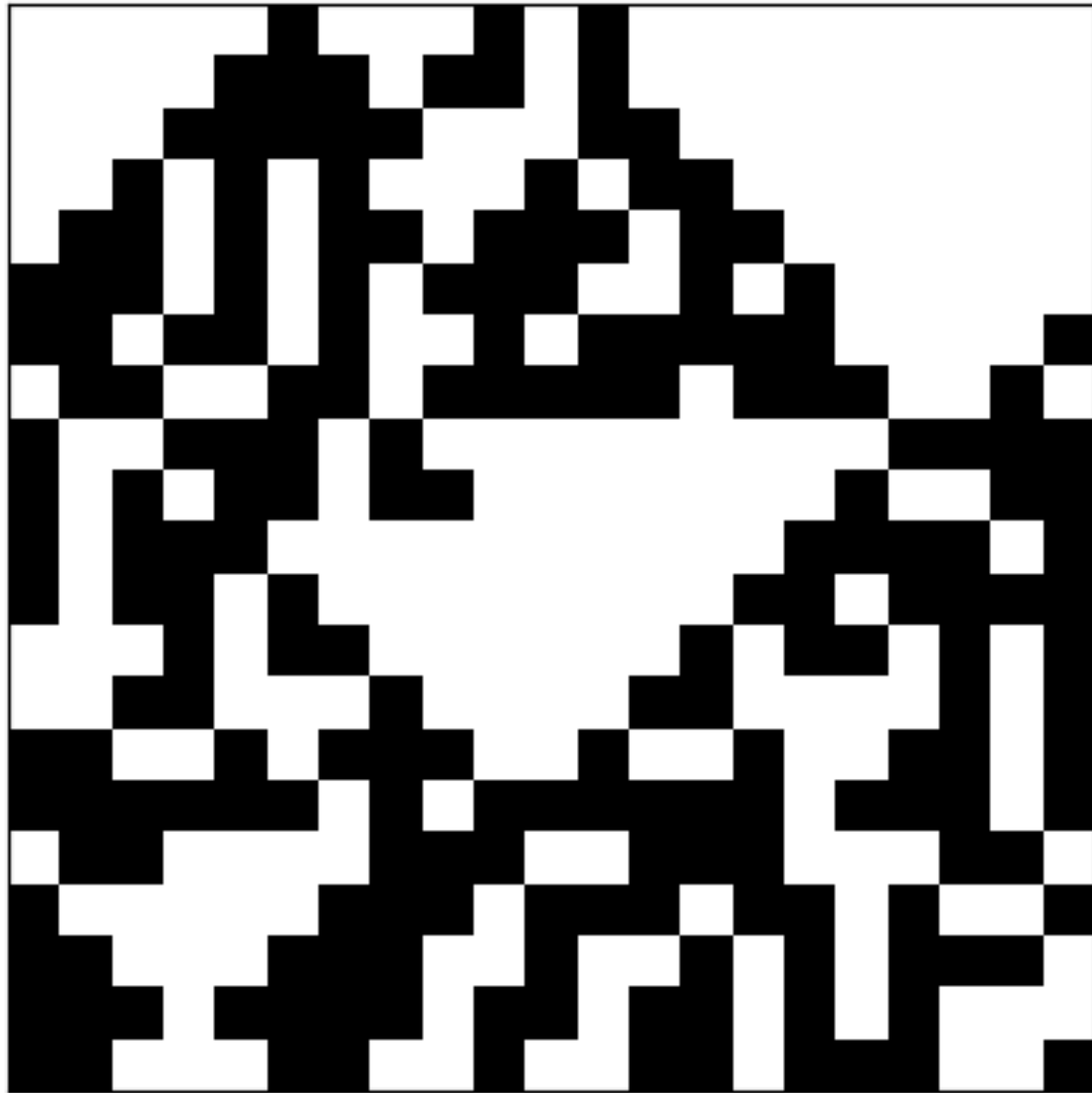
licznik - zlicza wystąpienia wartości

	19	31	33	12	29	0	0	96										
	28	23	17	29	25	37	41	0										
$\hat{p}_i$	0.596	0.426	0.34	0.707	0.463	1	1	0										
$p_i$	0.6	0.4	0.4	0.6	0.4	1	1	0										

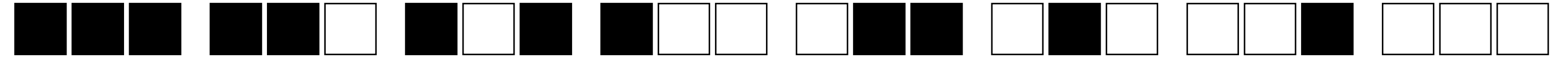
Jak widać nasz tajemniczy SCA to jest diploid, bo w pLUT występują tylko 4 różne wartości!



Oczywiście algorytm identyfikacji nie “widzi” prawdziwych wartości  $p_i$ , a jedynie  $\hat{p}_i$ .

Zakładamy, że szukany SCA to diploid i musimy “wyciągnąć coś” z tego założenia.



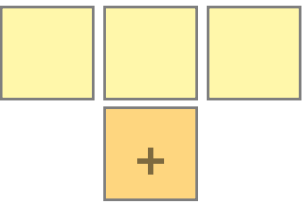
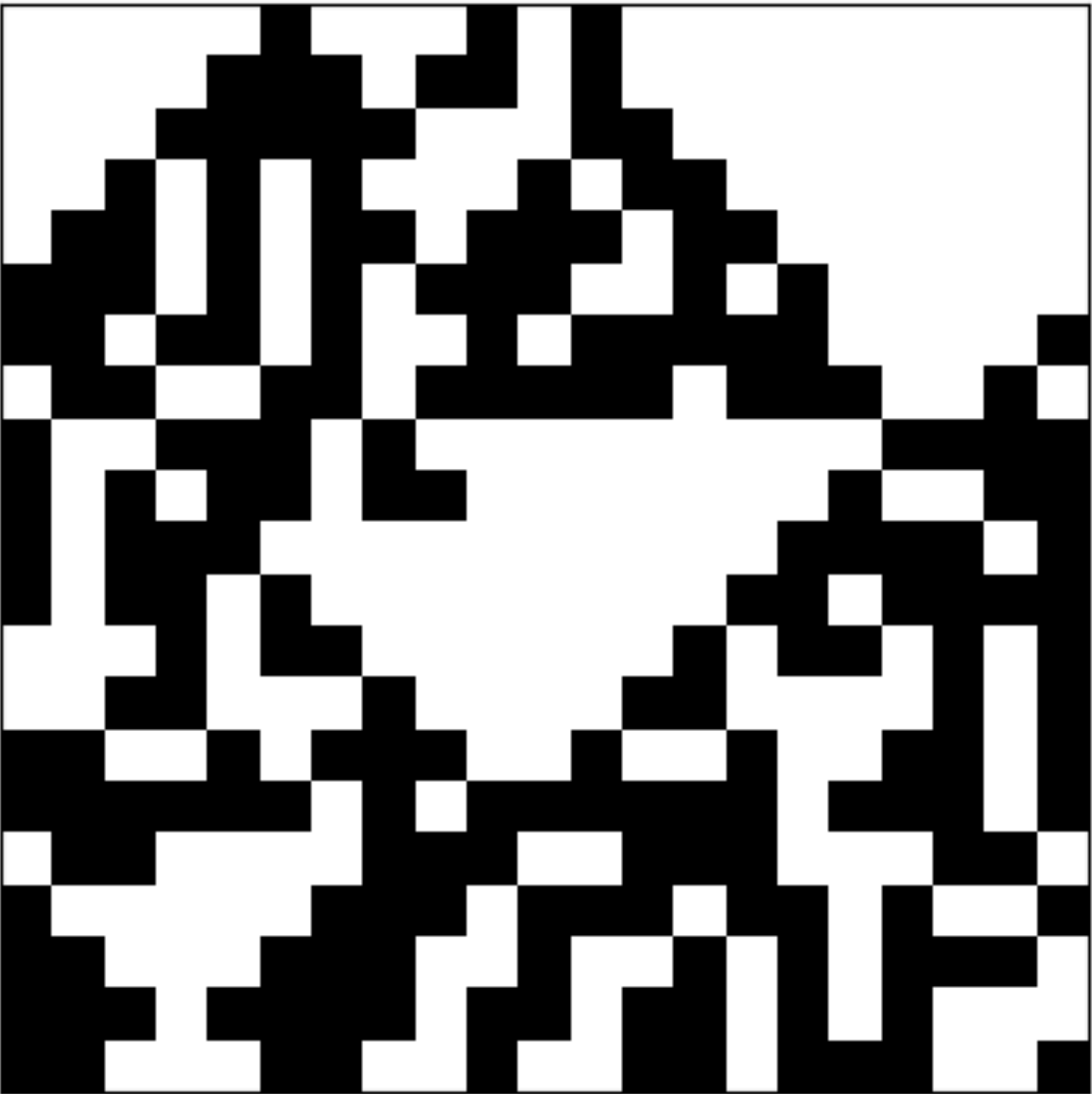
## licznik - zlicza wystąpienia wartości



	19	31	33	12	29	0	0	96
	28	23	17	29	25	37	41	0
$\hat{p}_i$	0.596	0.426	0.34	0.707	0.463	1	1	0
$\hat{f}_1$	?	?	?	?	?	1	1	0
$\hat{f}_2$	?	?	?	?	?	1	1	0

**Tu obie reguły - według obserwacji - zgadzają się ze sobą.**



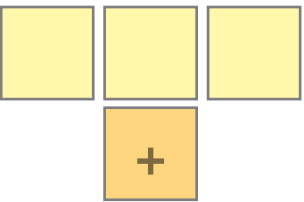
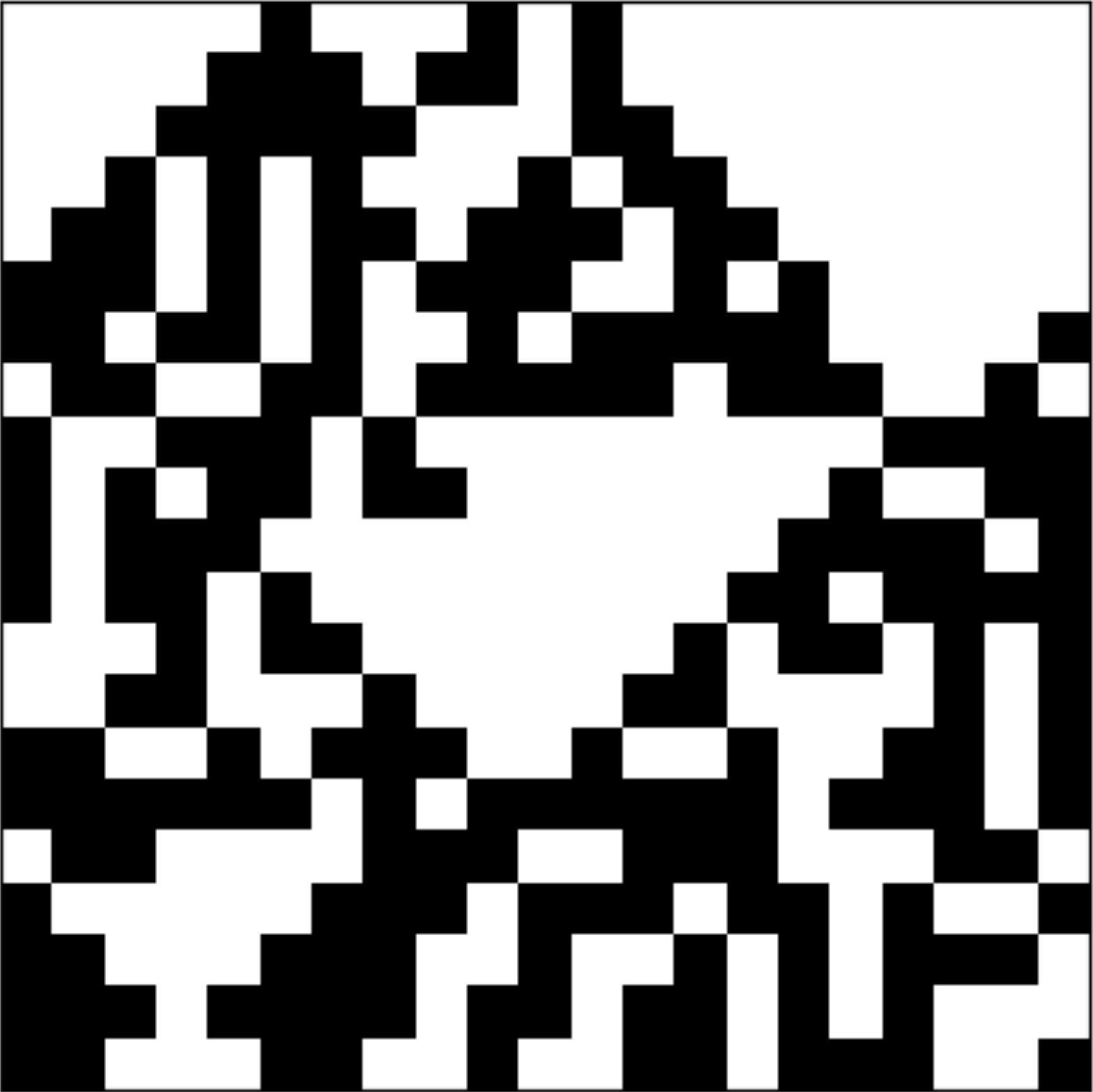


licznik - zlicza wystąpienia wartości

	<div><div></div><div></div><div></div></div>	<div><div></div><div></div><div></div></div>	<div><div></div><div></div><div></div></div>	<div><div></div><div></div><div></div></div>	<div><div></div><div></div><div></div></div>	<div><div></div><div></div><div></div></div>	<div><div></div><div></div><div></div></div>	<div><div></div><div></div><div></div></div>	<div><div></div><div></div><div></div></div>
<div><div></div></div>	19	31	33	12	29	0	0	96	
<div><div></div></div>	28	23	17	29	25	37	41	0	
$\hat{p}_i$	0.596	0.426	0.34	0.707	0.463	1	1	0	
$\hat{f}_1$	1	?	?	?	?	1	1	0	
$\hat{f}_2$	0	?	?	?	?	1	1	0	
Tu natomiast musimy policzyć :)									

Jeśli dla danego sąsiedztwa  $\hat{p}_i > 0.5$  to przyjmujemy, że

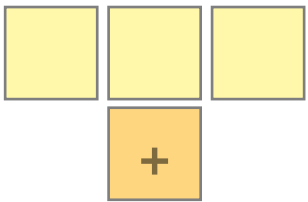
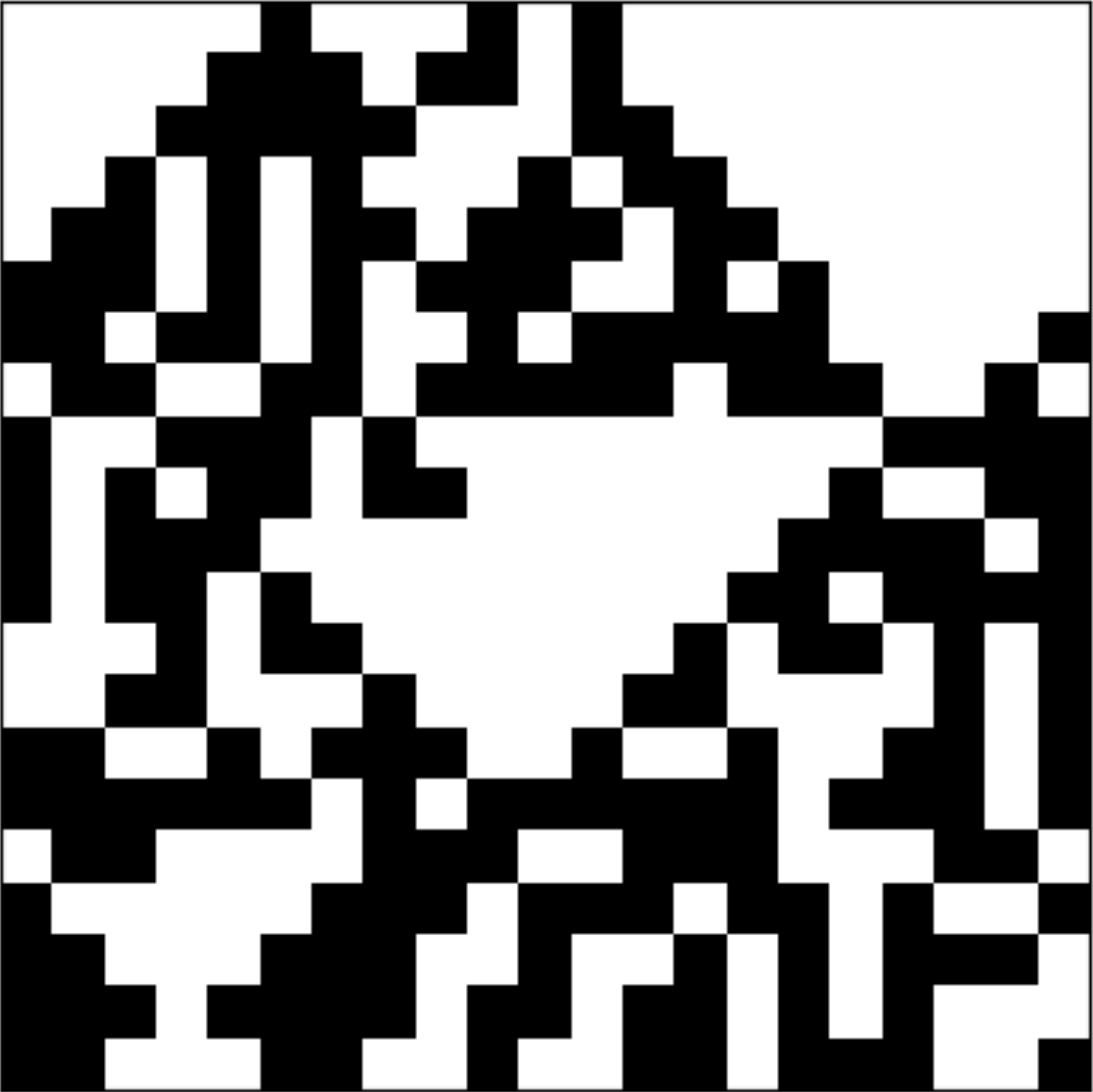
na tym sąsiedztwie:  $f_1 = 1, f_2 = 0$ .



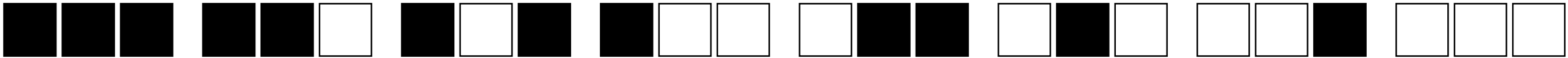
licznik - zlicza wystąpienia wartości

	<div><div></div><div></div><div></div></div>	<div><div></div><div></div><div></div></div>	<div><div></div><div></div><div></div></div>	<div><div></div><div></div><div></div></div>	<div><div></div><div></div><div></div></div>	<div><div></div><div></div><div></div></div>	<div><div></div><div></div><div></div></div>	<div><div></div><div></div><div></div></div>
<div><div></div></div>	19	31	33	12	29	0	0	96
<div><div></div></div>	28	23	17	29	25	37	41	0
$\hat{p}_i$	0.596	0.426	0.34	0.707	0.463	1	1	0
$\hat{f}_1$	1	0	?	?	?	1	1	0
$\hat{f}_2$	0	1	?	?	?	1	1	0
Tu natomiast musimy policzyć :)								

Jeśli dla danego sąsiedztwa  $\hat{p}_i < 0.5$  to przyjmujemy, że  
na tym sąsiedztwie:  $f_1 = 0, f_2 = 1$ .



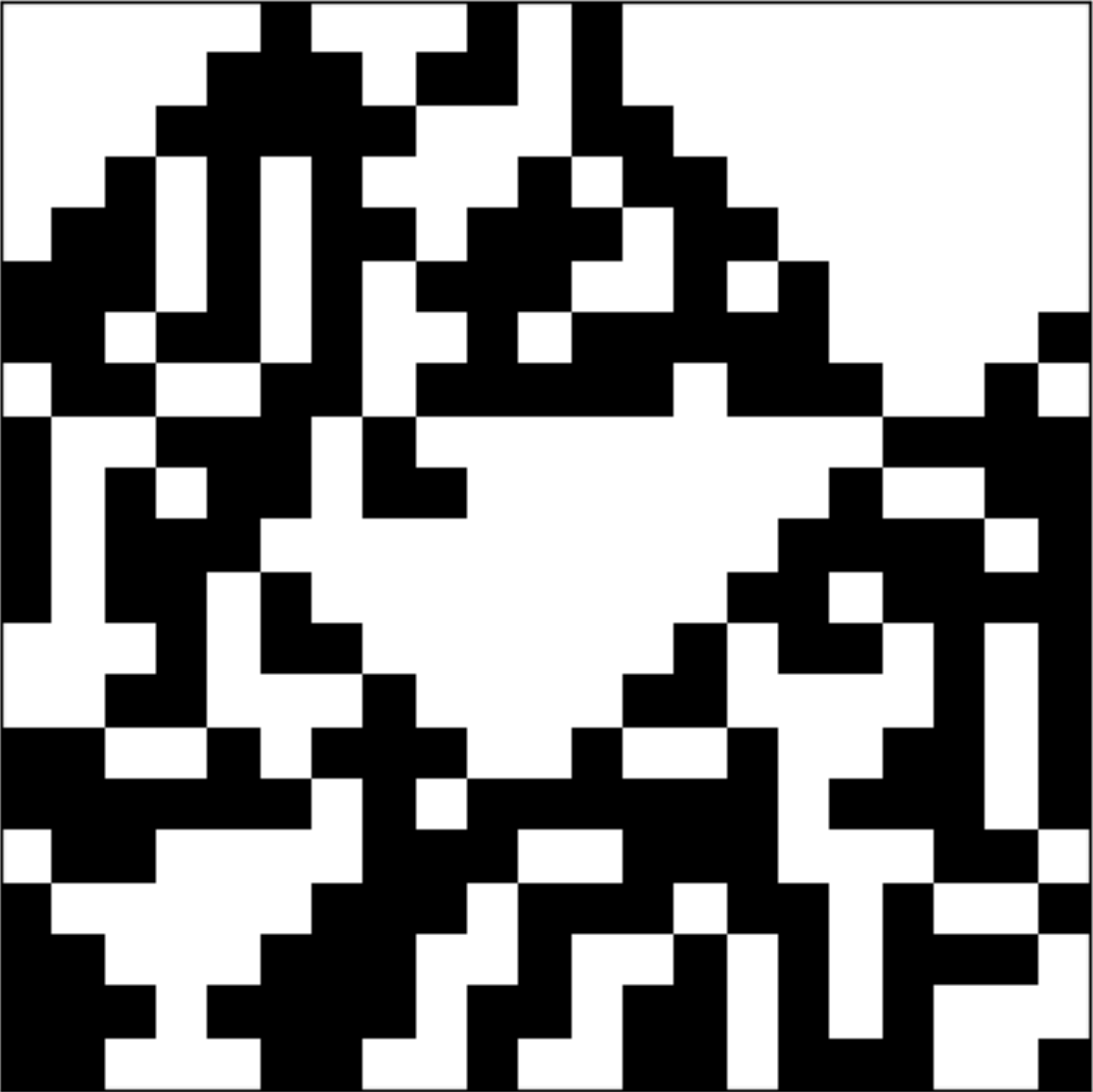
licznik - zlicza wystąpienia wartości



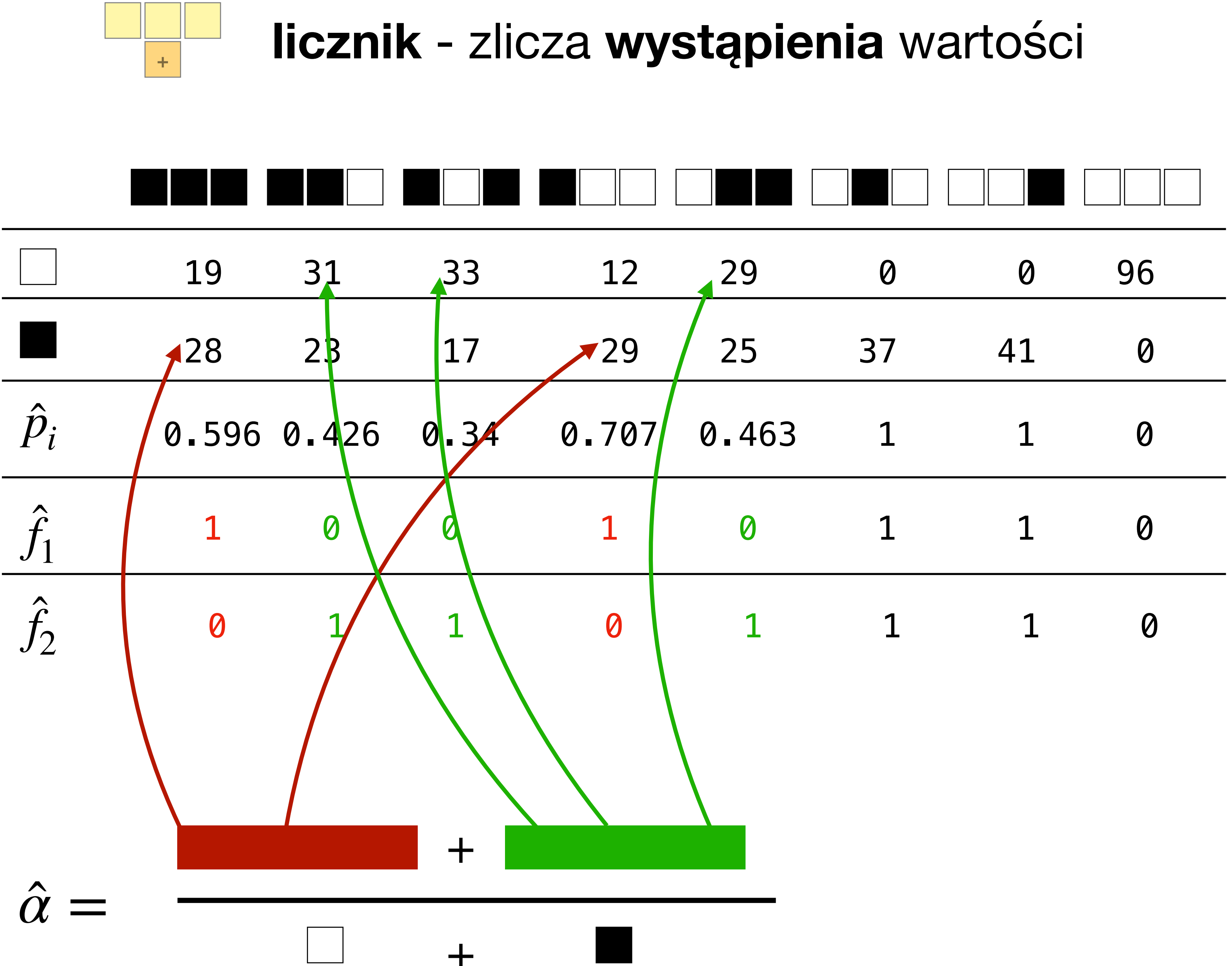
	19	31	33	12	29	0	0	96
	28	23	17	29	25	37	41	0
$\hat{p}_i$	0.596	0.426	0.34	0.707	0.463	1	1	0
$\hat{f}_1$	1	0	0	1	0	1	1	0
$\hat{f}_2$	0	1	1	0	1	1	1	0
No i mamy to!								

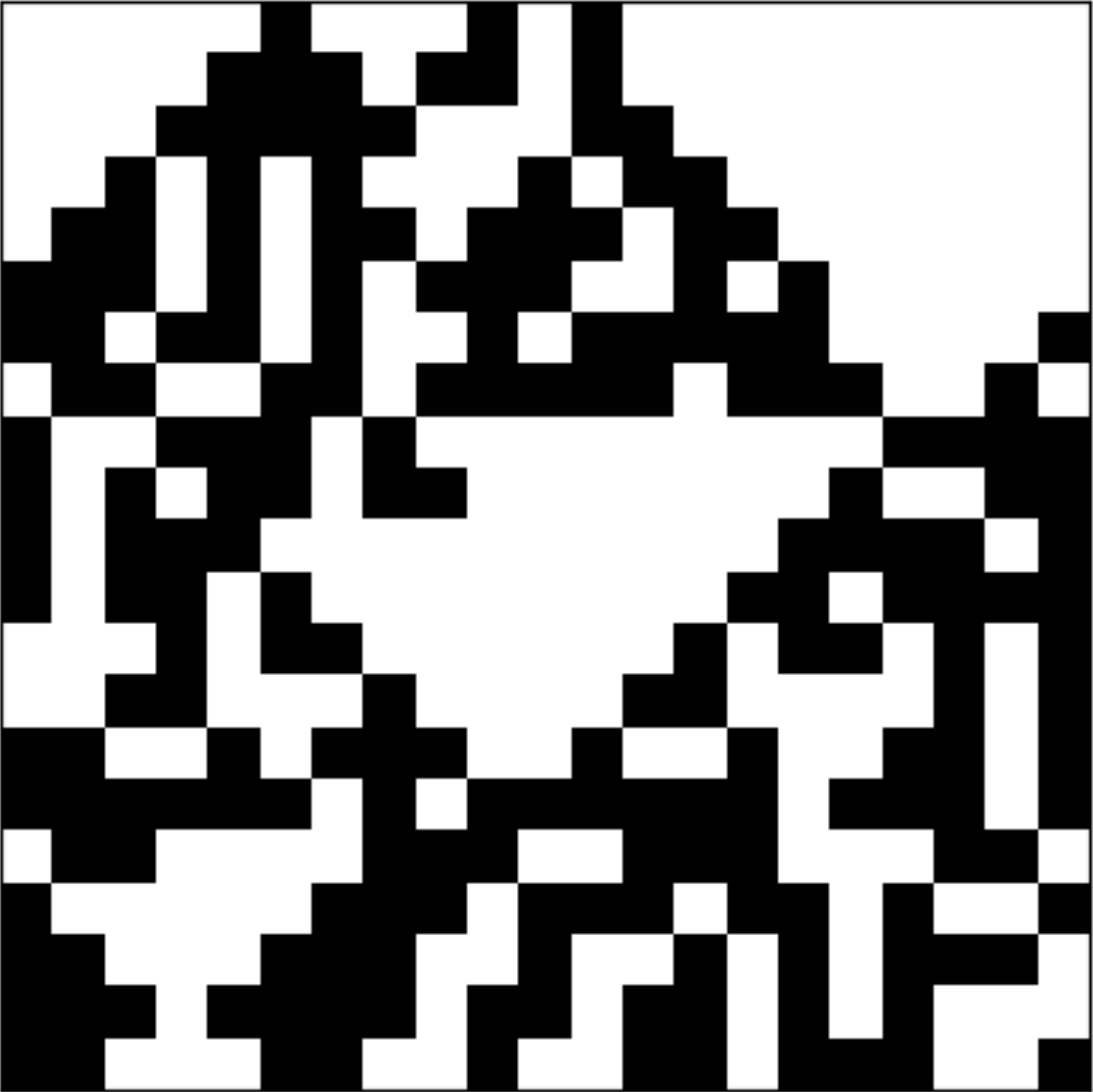
Pytanie ile wynosi  $\hat{\alpha}$  ?



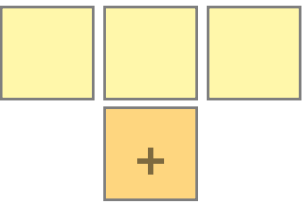


Pytanie ile wynosi  $\hat{\alpha}$  ?





Pytanie ile wynosi  $\hat{\alpha}$  ?



licznik - zlicza wystąpienia wartości

	<div><div>■ ■ ■</div><div>■ ■ □</div><div>■ □ ■</div><div>■ □ □</div><div>□ ■ ■</div><div>□ ■ □</div><div>□ □ ■</div><div>□ □ □</div></div>							
□	19	31	33	12	29	0	0	96
■	28	23	17	29	25	37	41	0
$\hat{p}_i$	0.596	0.426	0.34	0.707	0.463	1	1	0
$\hat{f}_1$	1	0	0	1	0	1	1	0
$\hat{f}_2$	0	1	1	0	1	1	1	0

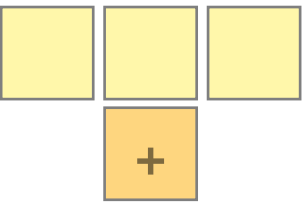
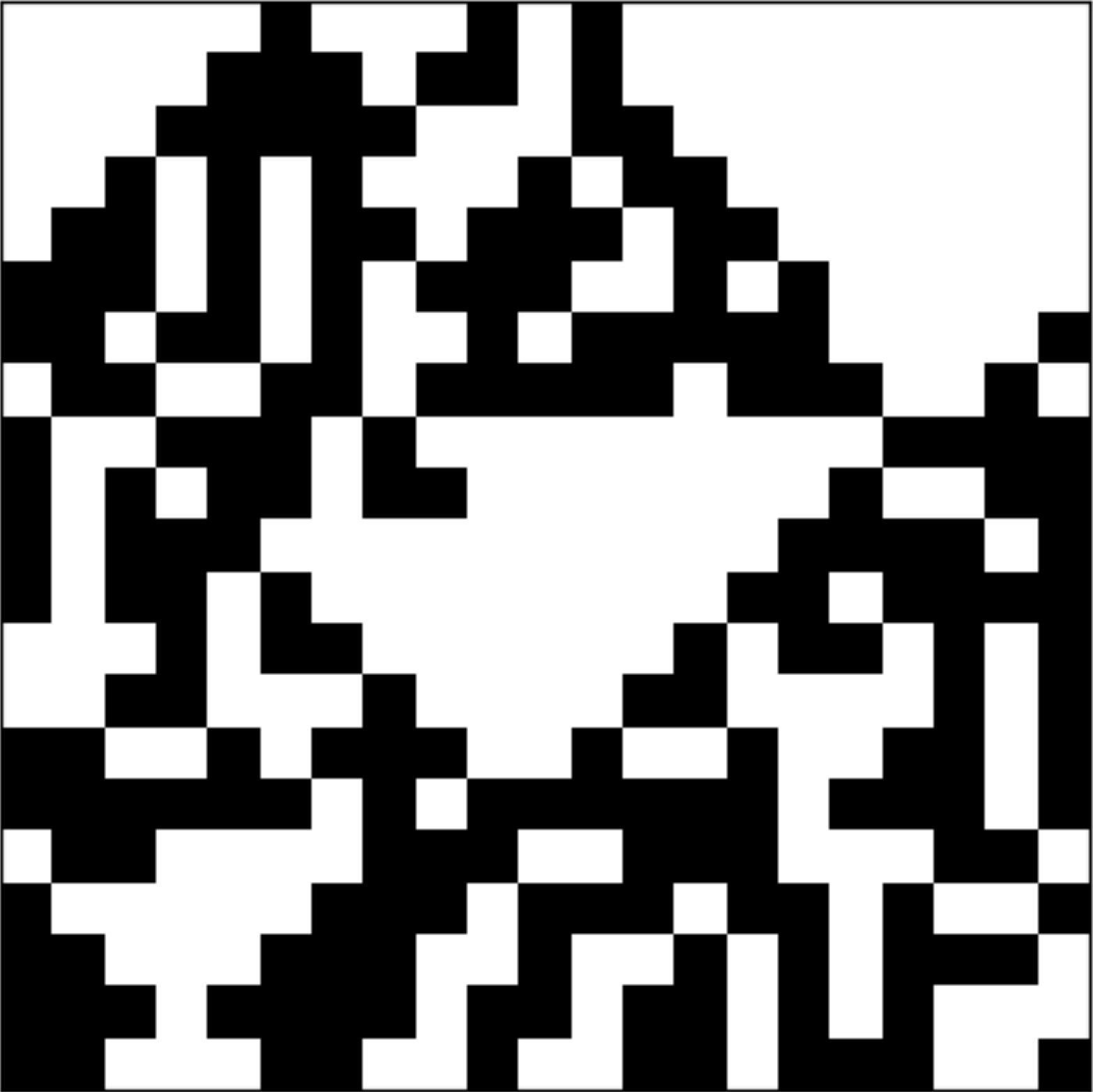
+

□

+

■

≈ 0.6098



licznik - zlicza wystąpienia wartości

	<div><div></div><div></div><div></div></div>	<div><div></div><div></div><div></div></div>	<div><div></div><div></div><div></div></div>	<div><div></div><div></div><div></div></div>	<div><div></div><div></div><div></div></div>	<div><div></div><div></div><div></div></div>	<div><div></div><div></div><div></div></div>	<div><div></div><div></div><div></div></div>
<div><div></div></div>	19	31	33	12	29	0	0	96
<div><div></div></div>	28	23	17	29	25	37	41	0
$\hat{p}_i$	0.596	0.426	0.34	0.707	0.463	1	1	0
$\hat{f}_1$	1	0	0	1	0	1	1	0
$\hat{f}_2$	0	1	1	0	1	1	1	0

$\hat{\alpha} \approx 0.6098$

$1 - \hat{\alpha} \approx 0.3902$

$\hat{p}_i$

0.6098 0.3902 0.3902 0.6098 0.3902 1 1 0

$\frac{1}{8} \sum_i |p_i - \hat{p}_i| \approx 0.0061$

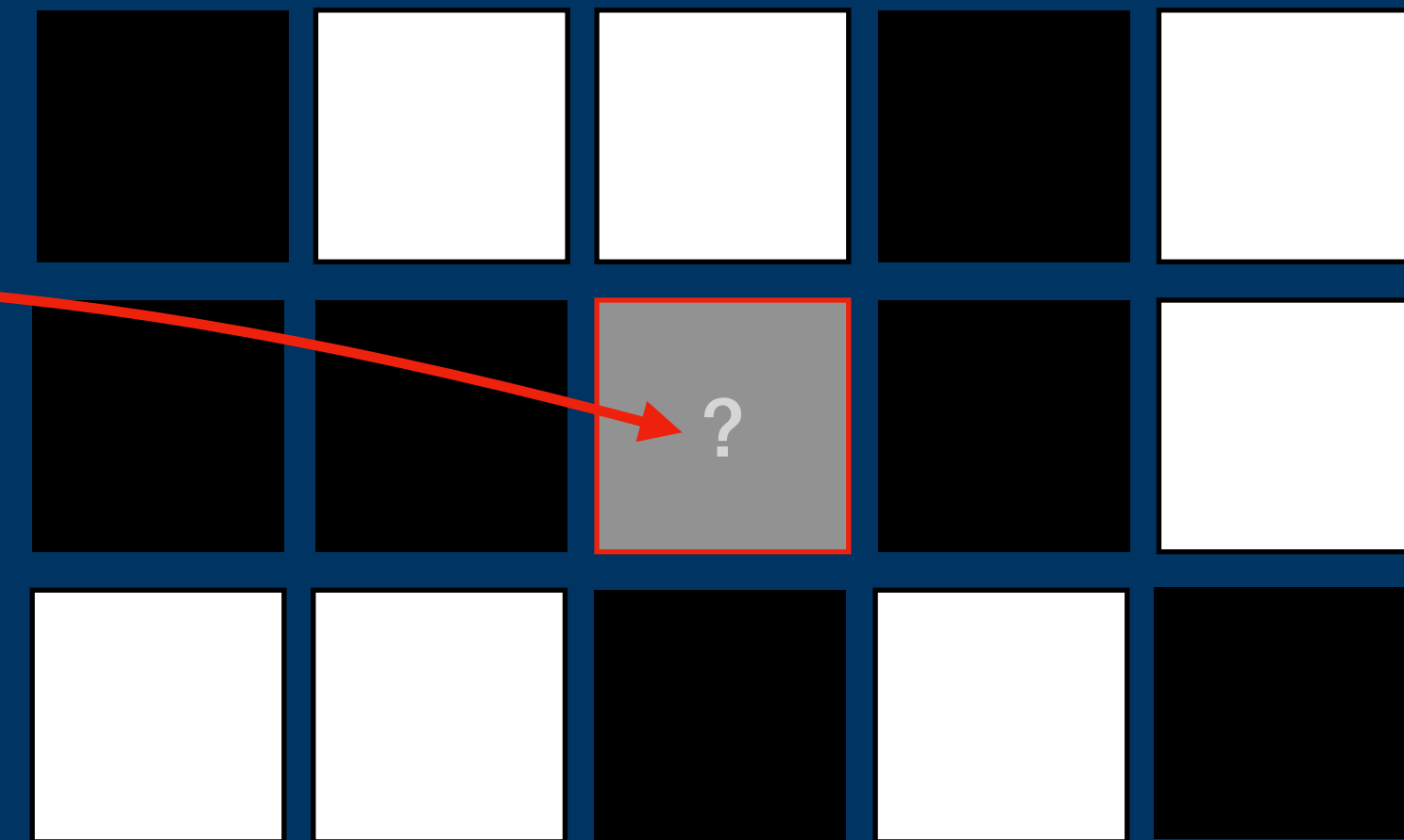
$\frac{1}{8} \sum_i |p_i - \hat{p}_i| \approx 0.0326$

$\max_i |p_i - \hat{p}_i| \approx 0.0097$

$\max_i \frac{|p_i - \hat{p}_i|}{p_i} \approx 0.1789$


$\max_i \frac{|p_i - \hat{p}_i|}{p_i} \approx 0.0244$

$\max_i |p_i - \hat{p}_i| \approx 0.1073$

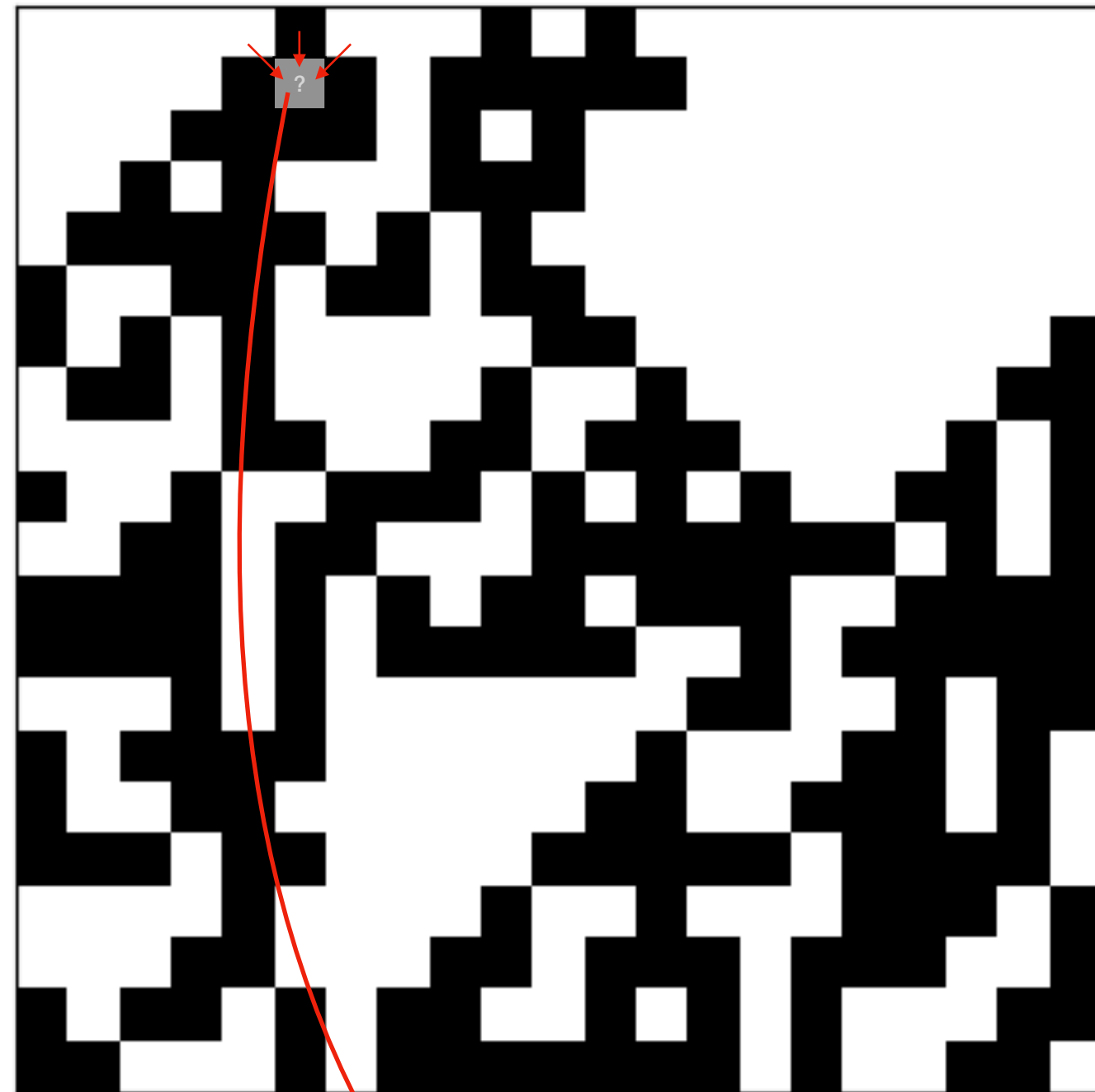


Niekompletne obserwacje

# Niekompletne obserwacje

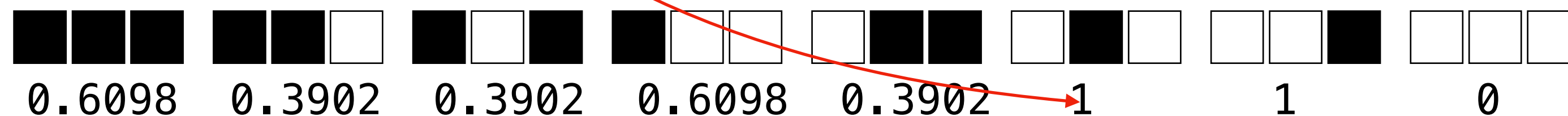
- W przypadku deterministycznym rozważaliśmy dwa typy niekompletnych / częściowych obserwacji: (a) **brakujące komórki** (oznaczone symbolem ‘?’) oraz (b) **brakujące kroki czasowe** - wiersze w space-time diagram (nieoznaczone niczym, po prostu ich nie było).
- Przypadek (a) łatwo rozwiązać. Nasz “skaner” omija komórki oznaczone ‘?’, odczytuje regułę a następnie zgodnie z regułą wypełnia ‘?’. W przypadku (b) omówiliśmy dość zawiły algorytm ewolucyjny, który umie znaleźć rozwiązanie.
- Dla SCAs potrafimy radzić sobie jedynie z przypadkiem (a). Nie wiem, co zrobić z przypadkiem (b) - być może *nic się nie da*.
- Przypadek (a) dla SCAs rozwiązujemy podobnie jak w CAs... ale “podobnie” oznacza tu jednak nieco więcej kłopotów, bo... 

# Niekompletne obserwacje



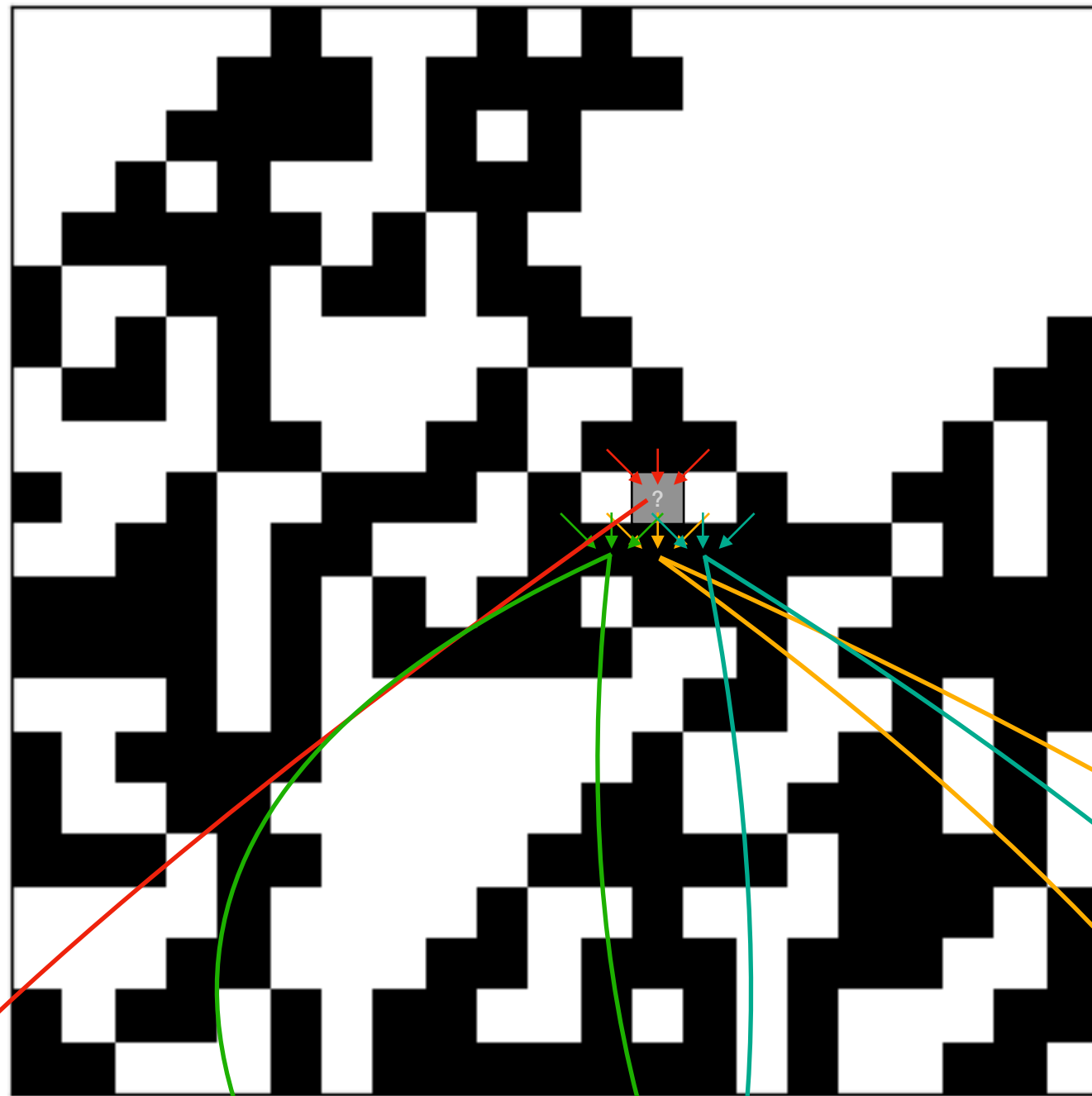
- Dziura może wystąpić w takim miejscu, dla którego poprzedzająca konfiguracja sąsiedztwa skutkuje przejściem deterministycznym.
- Wtedy, podobnie jak w przypadku CAs możemy odczytać wartość z pLUT i nie martwić się zbytnio...

$\hat{p}_i$

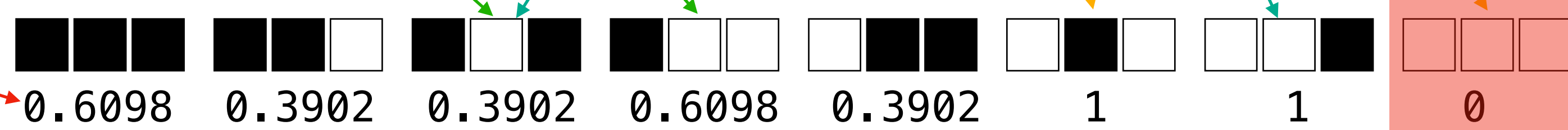


# Niekompletne obserwacje

- Dziura może wystąpić w takim miejscu, dla którego poprzedzająca konfiguracja sąsiedztwa skutkuje przejściem **niedeterministycznym**.
- Musimy wtedy rozpatrzyć kilka różnych sytuacji, policzyć różne prawdopodobieństwa 🧛 i wykluczyć przypadki zakazane...

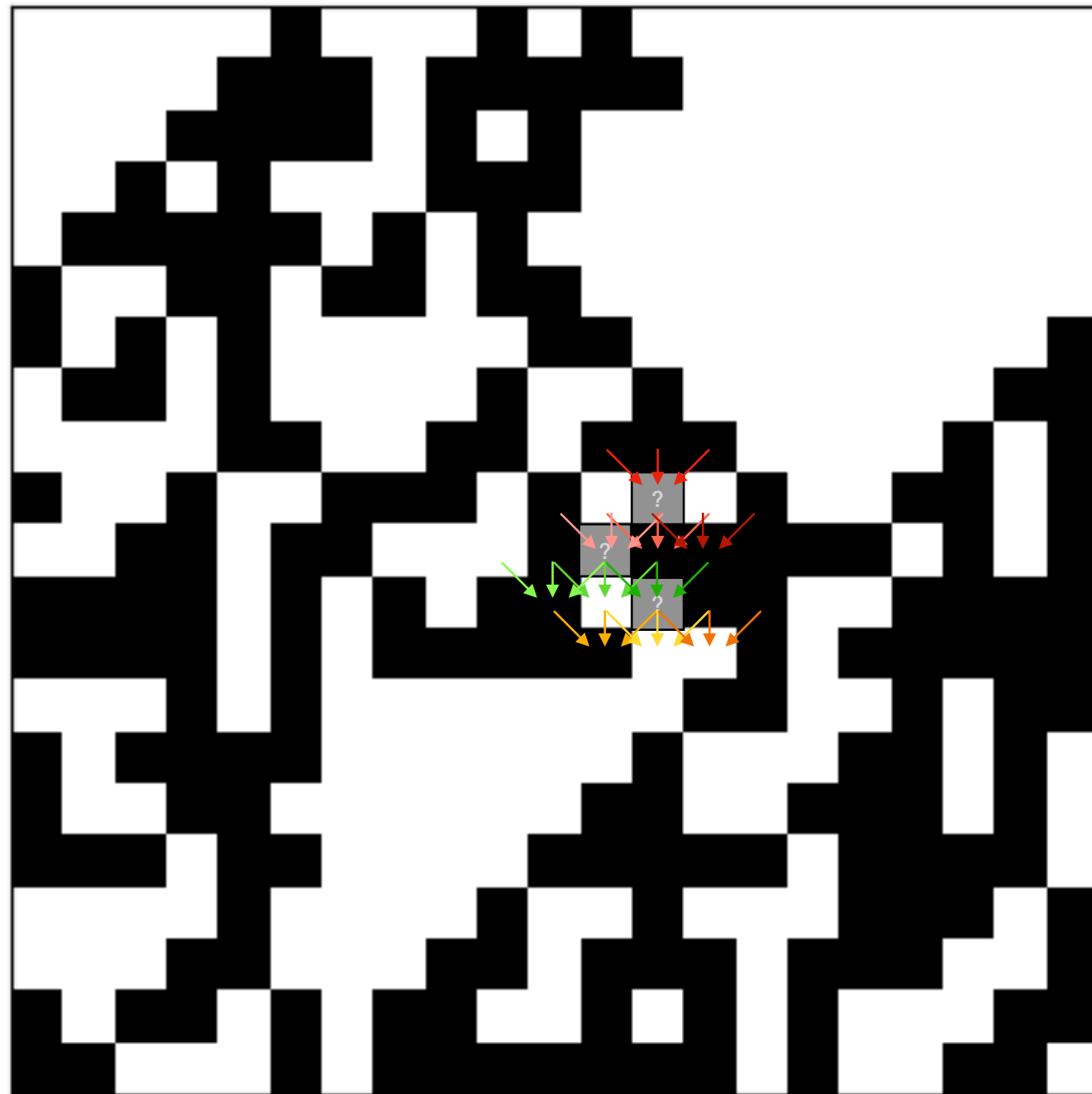


$\hat{p}_i$



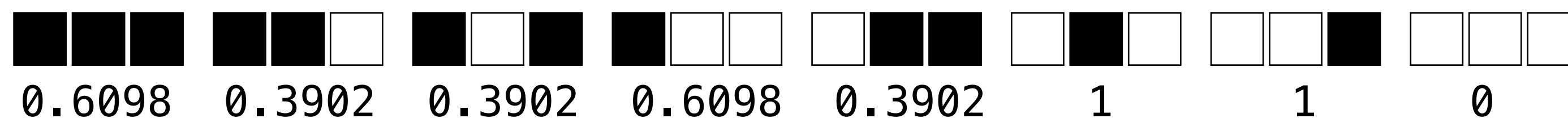


# Niekompletne obserwacje



- Niestety może być jeszcze gorzej :(
- Bo wypełnienie danej dziury wpływa na możliwe wypełnienia dziur w okolicy!
- I w najgorszym wypadku trzeba rozważyć wszystkie możliwości wypełnień, wyeliminować “zakazane” i policzyć prawdopodobieństwa dla pozostałych... 🙈
- Na dzisiaj darujemy sobie to.

$\hat{p}_i$







# Identyfikacja diploidów

- Omówiliśmy podstawy i “esencje” algorytmu. Ale są jeszcze ważne i ciekawe szczegóły... których niestety nie omówimy. Do znalezienia tu:


BioSystems 186 (2019) 103976

Contents lists available at ScienceDirect

 **BioSystems** 

journal homepage: [www.elsevier.com/locate/biosystems](http://www.elsevier.com/locate/biosystems)

---

**A statistical approach to the identification of diploid cellular automata based on incomplete observations** 

Witold Bołt<sup>a,c,\*</sup>, Aleksander Bołt<sup>b</sup>, Barbara Wolnik<sup>b</sup>, Jan M. Baetens<sup>c</sup>, Bernard De Baets<sup>c</sup>

<sup>a</sup> Systems Research Institute, Polish Academy of Sciences, Newelska St. 6, 01-447 Warsaw, Poland  
<sup>b</sup> Institute of Mathematics, Faculty of Mathematics, Physics and Informatics, University of Gdańsk, 80-308 Gdańsk, Poland  
<sup>c</sup> KERMIT, Department of Data Analysis and Mathematical Modelling, Ghent University, B-9000 Ghent, Belgium

---

ARTICLE INFO	ABSTRACT
<b>Keywords:</b> Stochastic cellular automata Diploid cellular automata Parameter estimation System identification	In this paper, the identification problem of diploid cellular automata is considered, in which, based on a series of incomplete observations, the underlying cellular automaton rules and the states of missing cell states are to be uncovered. An algorithm for identifying the rule, based on a statistical parameter estimation method using a normal distribution approximation, is presented. In addition, an algorithm for filling the missing cell states is formulated. The accuracy of these methods is examined in a series of computational experiments.

<https://www.sciencedirect.com/science/article/abs/pii/S030326471830128X>

# Wskazówka do implementacji

- Cały algorytm identyfikacji (zarówno deterministyczny jak i niedeterministyczny) w prostym przypadku kompletnych obserwacji da się zaimplementować w Pythonie w **kilku** liniijkach. (Przypomnę, że kilka z reguły oznacza mniej niż 10.)

```
def ca_identify(observation : np.ndarray):  
    ... neighborhoods = np.roll(observation, -1, axis=1) + observation*2 + 4*np.roll(observation, 1, axis=1)  
    ... # no i tu reszta implementacji ... :)
```

- W powyższym fragmencie kodu zakładamy, że **observation** to tablica 2D przedstawiająca space-time diagram. Warto zrobić sobie eksperyment i dla prawdziwego space-time diagramy wykonać powyższą liniijkę i następnie wyświetlić zawartość **neighborhoods** za pomocą zwykłego **print(...)**



**Dziękuję bardzo**  
**Witold.Bolt@ug.edu.pl**

