Information, communication and complexity of sofic tilings

Antonin Callard (LIP, ENS de Lyon)

Joint work with Léo Paviet Salomon and Pascal Vanier

Journées postdocs du LIP, ENS de Lyon

24 March 2026

Shift spaces

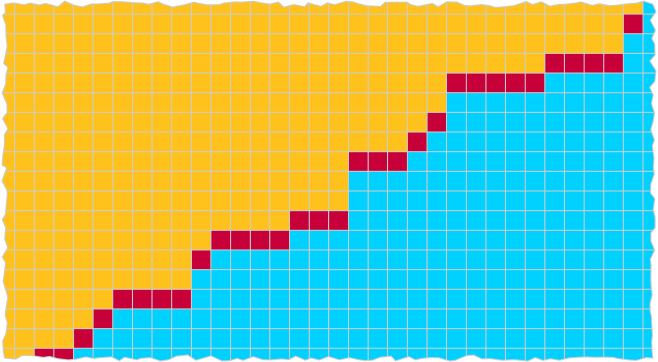
Definition

A *shift space* on $\mathbb{Z}^d$ is a set of colorings $\mathbb{Z}^d \to \mathcal{A}$ ("*tilings*") defined by forbidden patterns $\mathcal{F}$:

$$X_{\mathcal{F}} = \left\{ x \in \mathcal{A}^{\mathbb{Z}^d} : \forall w \in \mathcal{F},\, w \text{ does not appear in } x \right\}.$$

Example:

$\mathcal{F} = \{$  $\}$



Shift spaces

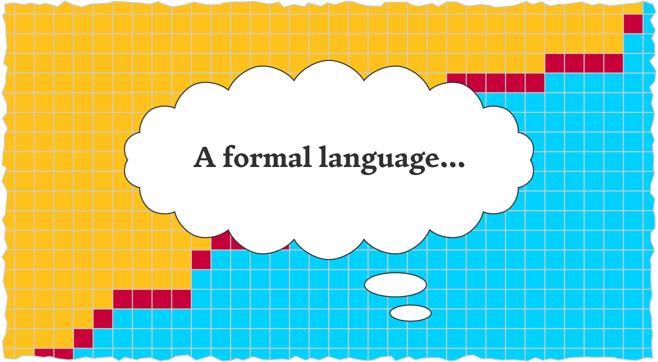
Definition

A *shift space* on $\mathbb{Z}^d$ is a set of colorings $\mathbb{Z}^d \to \mathcal{A}$ ("*tilings*") defined by forbidden patterns $\mathcal{F}$:

$$X_{\mathcal{F}} = \left\{ x \in \mathcal{A}^{\mathbb{Z}^d} : \forall w \in \mathcal{F},\ w \text{ does not appear in } x \right\}.$$

Example:

$\mathcal{F} = \{$  $\}$



Classifying shift spaces by computational expressive power

Definition

A *shift space* on $\mathbb{Z}^d$ is a set of colorings $\mathbb{Z}^d \to \mathcal{A}$ (“*tilings*”) defined by forbidden patterns $\mathcal{F}$:

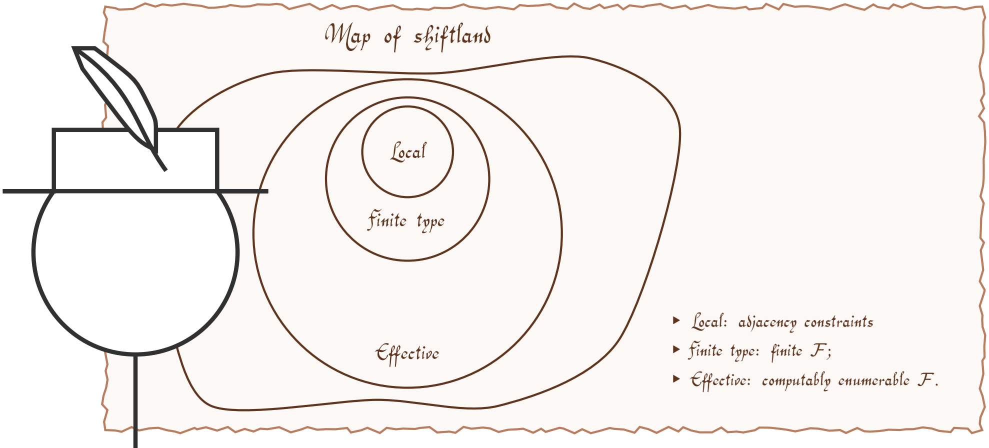
$$X_{\mathcal{F}} = \left\{x \in \mathcal{A}^{\mathbb{Z}^d} : \forall w \in \mathcal{F},\ w \text{ does not appear in } x\right\}.$$

Classifying shift spaces by computational expressive power

Definition

A *shift space* on $\mathbb{Z}^d$ is a set of colorings $\mathbb{Z}^d \to \mathcal{A}$ ("*tilings*") defined by forbidden patterns $\mathcal{F}$:

$$X_{\mathcal{F}} = \left\{ x \in \mathcal{A}^{\mathbb{Z}^d} : \forall w \in \mathcal{F},\ w \text{ does not appear in } x \right\}.$$

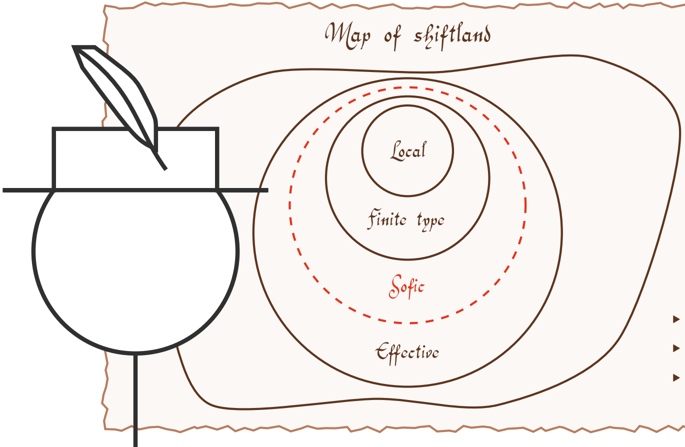


Classifying shift spaces by computational expressive power

Definition

A *shift space* on $\mathbb{Z}^d$ is a set of colorings $\mathbb{Z}^d \to \mathcal{A}$ ("*tilings*") defined by forbidden patterns $\mathcal{F}$:

$$X_{\mathcal{F}} = \left\{ x \in \mathcal{A}^{\mathbb{Z}^d} : \forall w \in \mathcal{F},\ w \text{ does not appear in } x \right\}.$$



- Local: adjacency constraints
- Finite type: finite $\mathcal{F}$;
- Effective: computably enumerable $\mathcal{F}$.

Classifying shift spaces by computational expressive power

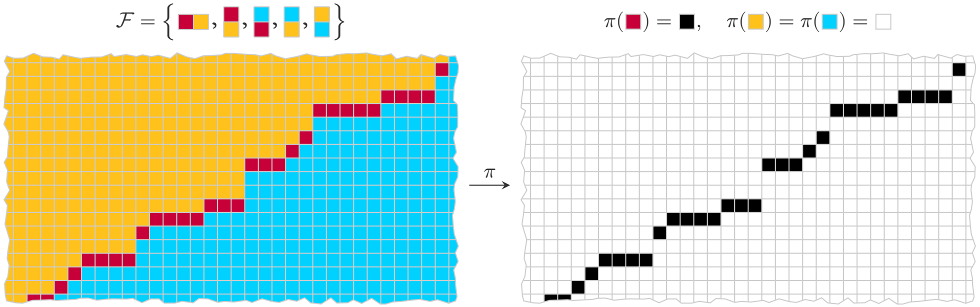
Definition

A *shift space* on $\mathbb{Z}^d$ is a set of colorings $\mathbb{Z}^d \to \mathcal{A}$ (“*tilings*”) defined by forbidden patterns $\mathcal{F}$:

$$X_{\mathcal{F}} = \left\{ x \in \mathcal{A}^{\mathbb{Z}^d} : \forall w \in \mathcal{F},\ w \text{ does not appear in } x \right\}.$$

Definition (Sofic space, ≃ 1973)

A shift space is *sofic* if it can be defined as the cell-by-cell projection of a local space.

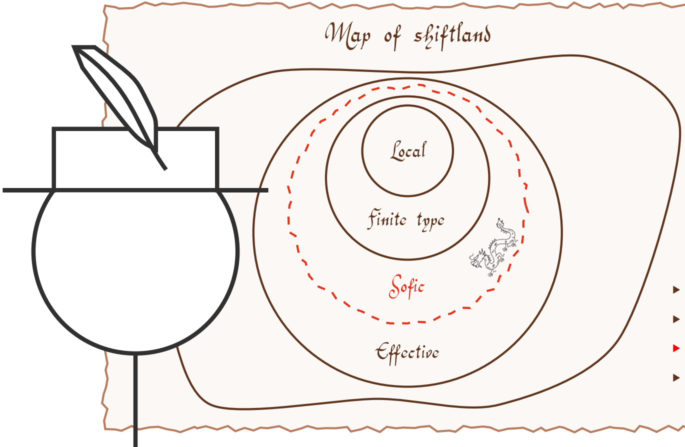


Classifying shift spaces by computational expressive power

Definition

A *shift space* on $\mathbb{Z}^d$ is a set of colorings $\mathbb{Z}^d \to \mathcal{A}$ ("*tilings*") defined by forbidden patterns $\mathcal{F}$:

$$X_{\mathcal{F}} = \left\{x \in \mathcal{A}^{\mathbb{Z}^d} : \forall w \in \mathcal{F},\ w \text{ does not appear in } x\right\}.$$



- Local: adjacency constraints
- Finite type: finite $\mathcal{F}$;
- Sofic: projection of local;
- Effective: computably enumerable $\mathcal{F}$.

Shift spaces on $\mathbb{Z}$

Shift space $X \subseteq \mathcal{A}^{\mathbb{Z}}$	**Language** $L \subseteq \mathcal{A}^*$
Local/finite type	Local
Sofic	Rational/regular
Effective	Comp. co-enumerable

Example

The sunny-side-up $X_{\blacksquare} \subseteq \{\square, \blacksquare\}^{\mathbb{Z}}$ is sofic.

Shift spaces on $\mathbb{Z}$

Shift space $X \subseteq \mathcal{A}^{\mathbb{Z}}$	**Language** $L \subseteq \mathcal{A}^*$
Local/finite type	Local
Sofic	Rational/regular
Effective	Comp. co-enumerable

Example

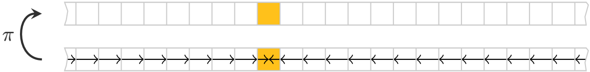
The sunny-side-up $X_{■} \subseteq \{□, ■\}^{\mathbb{Z}}$ is sofic.

Shift spaces on $\mathbb{Z}$

Shift space $X \subseteq \mathcal{A}^{\mathbb{Z}}$	**Language** $L \subseteq \mathcal{A}^*$
Local/finite type	Local
Sofic	Rational/regular
Effective	Comp. co-enumerable

Example

The sunny-side-up $X_{■} \subseteq \{□, ■\}^{\mathbb{Z}}$ is sofic.

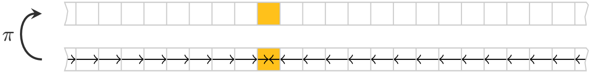


Shift spaces on $\mathbb{Z}$

Shift space $X \subseteq \mathcal{A}^{\mathbb{Z}}$	**Language** $L \subseteq \mathcal{A}^*$
Local/finite type	Local
Sofic	Rational/regular
Effective	Comp. co-enumerable

Example

The sunny-side-up $X_{\blacksquare} \subseteq \{\square, \blacksquare\}^{\mathbb{Z}}$ is sofic.



Example

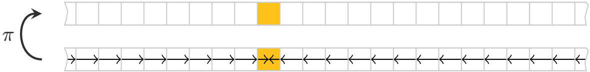
The space of all periods $X_p \subseteq \{\square, \blacksquare\}^{\mathbb{Z}}$ is not sofic.

Shift spaces on $\mathbb{Z}$

Shift space $X \subseteq \mathcal{A}^{\mathbb{Z}}$	**Language** $L \subseteq \mathcal{A}^*$
Local/finite type	Local
Sofic	Rational/regular
Effective	Comp. co-enumerable

Example

The sunny-side-up $X_{■} \subseteq \{□, ■\}^{\mathbb{Z}}$ is sofic.



Example

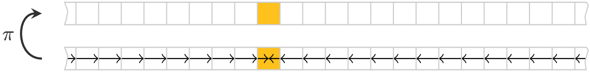
The space of all periods $X_p \subseteq \{□, ■\}^{\mathbb{Z}}$ is not sofic.

Shift spaces on $\mathbb{Z}$

Shift space $X \subseteq \mathcal{A}^{\mathbb{Z}}$	**Language** $L \subseteq \mathcal{A}^*$
Local/finite type	Local
Sofic	Rational/regular
Effective	Comp. co-enumerable

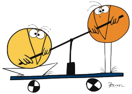
Example

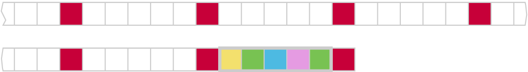
The sunny-side-up $X_{■} \subseteq \{□, ■\}^{\mathbb{Z}}$ is sofic.



Example

The space of all periods $X_p \subseteq \{□, ■\}^{\mathbb{Z}}$ is not sofic.



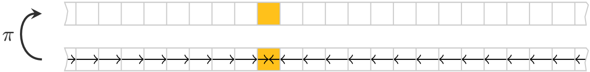


Shift spaces on $\mathbb{Z}$

Shift space $X \subseteq \mathcal{A}^{\mathbb{Z}}$	**Language** $L \subseteq \mathcal{A}^*$
Local/finite type	Local
Sofic	Rational/regular
Effective	Comp. co-enumerable

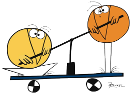
Example

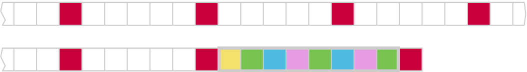
The sunny-side-up $X_{■} \subseteq \{□, ■\}^{\mathbb{Z}}$ is sofic.



Example

The space of all periods $X_p \subseteq \{□, ■\}^{\mathbb{Z}}$ is not sofic.



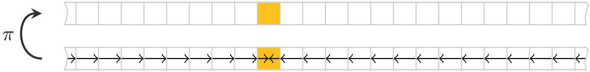


Shift spaces on $\mathbb{Z}$

Shift space $X \subseteq \mathcal{A}^{\mathbb{Z}}$	**Language** $L \subseteq \mathcal{A}^*$
Local/finite type	Local
Sofic	Rational/regular
Effective	Comp. co-enumerable

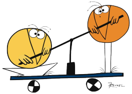
Example

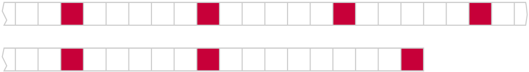
The sunny-side-up $X_{■} \subseteq \{□, ■\}^{\mathbb{Z}}$ is sofic.



Example

The space of all periods $X_p \subseteq \{□, ■\}^{\mathbb{Z}}$ is not sofic.



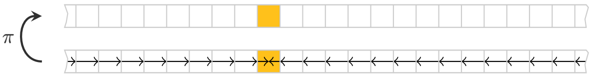


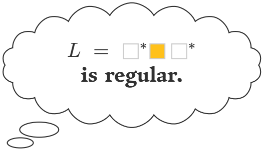
Shift spaces on $\mathbb{Z}$

Shift space $X \subseteq \mathcal{A}^{\mathbb{Z}}$	**Language** $L \subseteq \mathcal{A}^*$
Local/finite type	Local
Sofic	Rational/regular
Effective	Comp. co-enumerable

Example

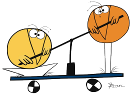
The sunny-side-up $X_{\blacksquare} \subseteq \{\square, \blacksquare\}^{\mathbb{Z}}$ is sofic.

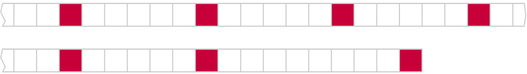


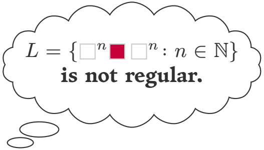


Example

The space of all periods $X_p \subseteq \{\square, \blacksquare\}^{\mathbb{Z}}$ is not sofic.





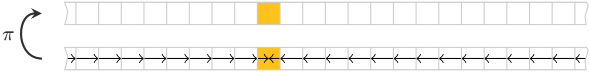


Shift spaces on $\mathbb{Z}$

Shift space $X \subseteq \mathcal{A}^{\mathbb{Z}}$	**Language** $L \subseteq \mathcal{A}^*$
Local/finite type	Local
Sofic	Rational/regular
Effective	Comp. co-enumerable

Example

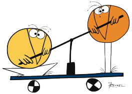
The sunny-side-up $X_{\blacksquare} \subseteq \{\square, \blacksquare\}^{\mathbb{Z}}$ is sofic.

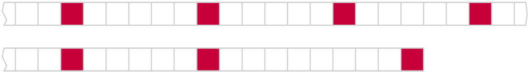


$L = \square^* \blacksquare \square^*$
is regular.

Example

The space of all periods $X_p \subseteq \{\square, \blacksquare\}^{\mathbb{Z}}$ is not sofic.





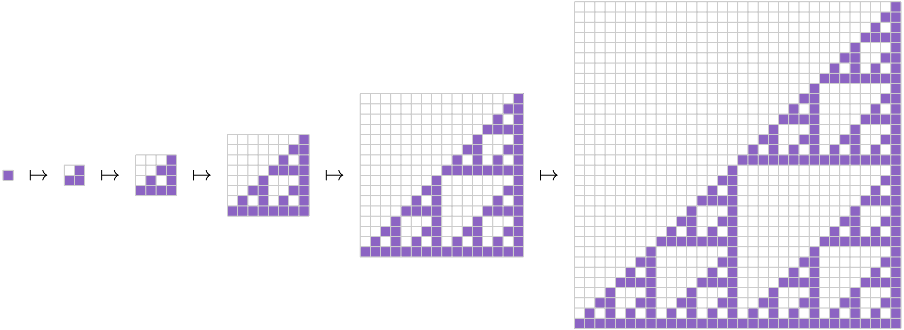
$L = \{\square^n \blacksquare \square^n : n \in \mathbb{N}\}$
is not regular.

Lemma (Myhill-Nerode)

A shift space $X \subseteq \mathcal{A}^{\mathbb{Z}}$ is sofic if and only if it defines finitely many *extender sets* (*i.e.* "contexts").

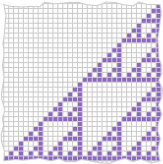
Soficity of shift spaces on $\mathbb{Z}^d$: examples

- Substitutive shifts;
  [Mozes, 1989]

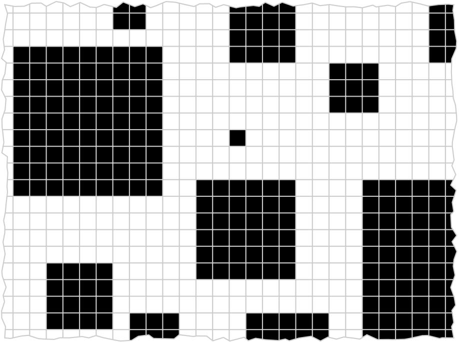


Soficity of shift spaces on $\mathbb{Z}^d$: examples

- Substitutive shifts; [Mozes, 1989]

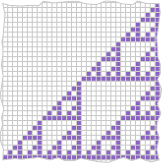


- Seas of squares;

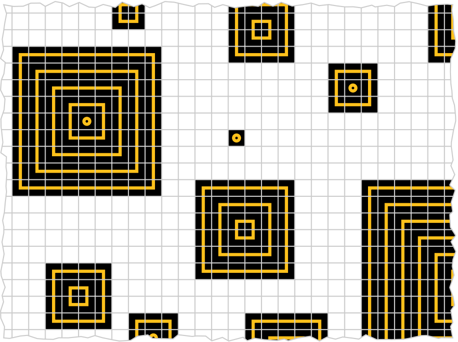


Soficity of shift spaces on $\mathbb{Z}^d$: examples

- Substitutive shifts;
  [Mozes, 1989]

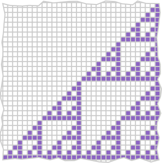


- Seas of squares;

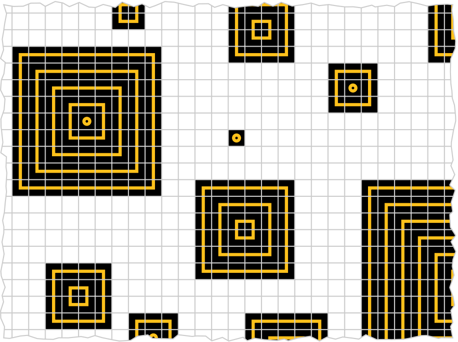


Soficity of shift spaces on $\mathbb{Z}^d$: examples

- Substitutive shifts;
  [Mozes, 1989]

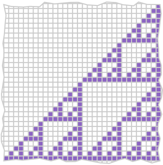


- Seas of squares (of Π_1^0 sizes);
  [Westrick, 2017]

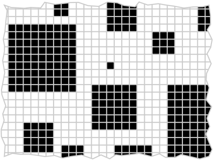


Soficity of shift spaces on $\mathbb{Z}^d$: examples

- Substitutive shifts;
  [Mozes, 1989]



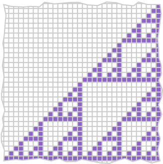
- Seas of squares (of Π^0_1 sizes);
  [Westrick, 2017]



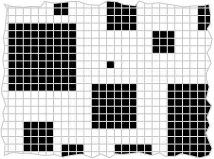
- Extensions of effective $\mathbb{Z}$ shifts;
  [Hochman, … 2009+]

Soficity of shift spaces on $\mathbb{Z}^d$: examples

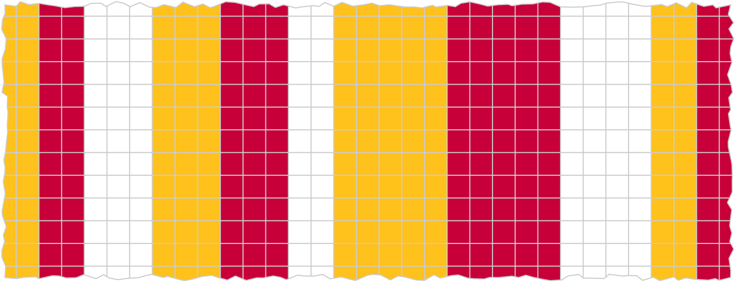
- Substitutive shifts;
  [Mozes, 1989]



- Seas of squares (of Π_1^0 sizes);
  [Westrick, 2017]



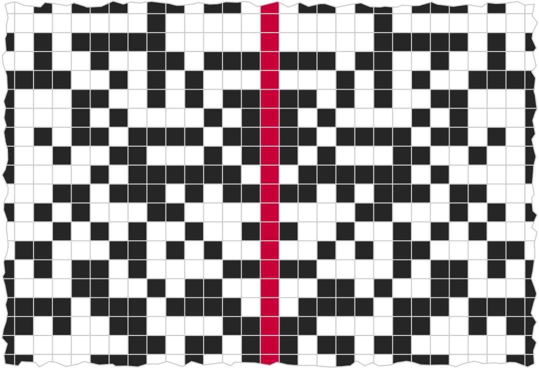
- Extensions of effective $\mathbb{Z}$ shifts;
  [Hochman, ... 2009+]



Soficity of shift spaces on $\mathbb{Z}^d$: expressiveness limits

Example

The mirror shift space is **not** sofic on $\mathbb{Z}^d$ for any $d \in \mathbb{N}$:



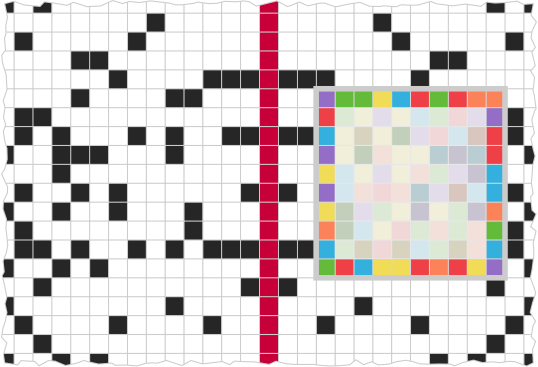
(folklore)

Soficity of shift spaces on $\mathbb{Z}^d$: expressiveness limits

Example

The mirror shift space is **not** sofic on $\mathbb{Z}^d$ for any $d \in \mathbb{N}$:

Pattern $O(n^d)$;
Border $O(n^{d-1})$;



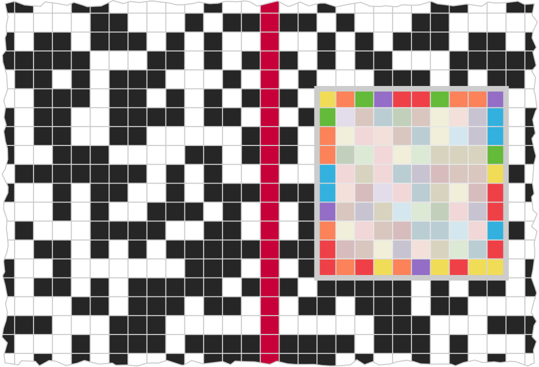
(folklore)

Soficity of shift spaces on $\mathbb{Z}^d$: expressiveness limits

Example

The mirror shift space is **not** sofic on $\mathbb{Z}^d$ for any $d \in \mathbb{N}$:

Pattern $O(n^d)$;
Border $O(n^{d-1})$;



(folklore)

Soficity of shift spaces on $\mathbb{Z}^d$: expressiveness limits

Example

The mirror shift space is **not** sofic on $\mathbb{Z}^d$ for any $d \in \mathbb{N}$:

Pattern $O(n^d)$;
Border $O(n^{d-1})$;

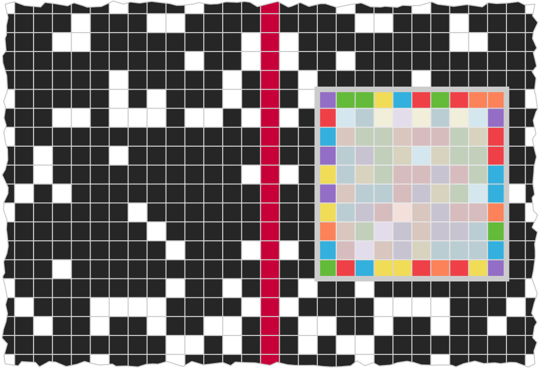
(folklore)

Soficity of shift spaces on $\mathbb{Z}^d$: expressiveness limits

Example

The mirror shift space is **not** sofic on $\mathbb{Z}^d$ for any $d \in \mathbb{N}$:

Pattern $O(n^d)$;
Border $O(n^{d-1})$;



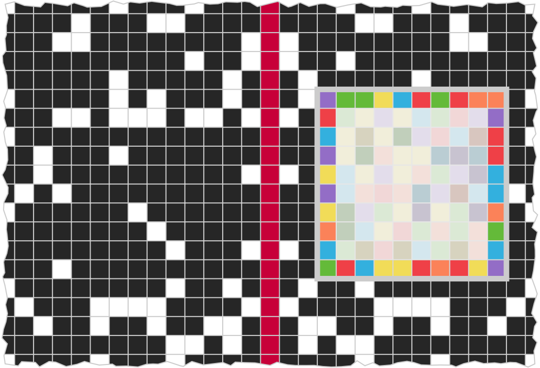
(folklore)

Soficity of shift spaces on $\mathbb{Z}^d$: expressiveness limits

Example

The mirror shift space is **not** sofic on $\mathbb{Z}^d$ for any $d \in \mathbb{N}$:

Pattern $O(n^d)$;
Border $O(n^{d-1})$;



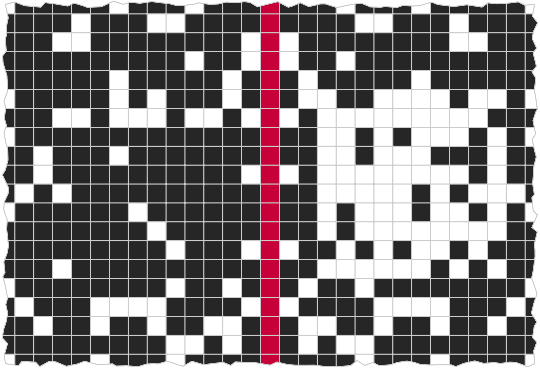
(folklore)

Soficity of shift spaces on $\mathbb{Z}^d$: expressiveness limits

Example

The mirror shift space is **not** sofic on $\mathbb{Z}^d$ for any $d \in \mathbb{N}$:

Pattern $O(n^d)$;
Border $O(n^{d-1})$;



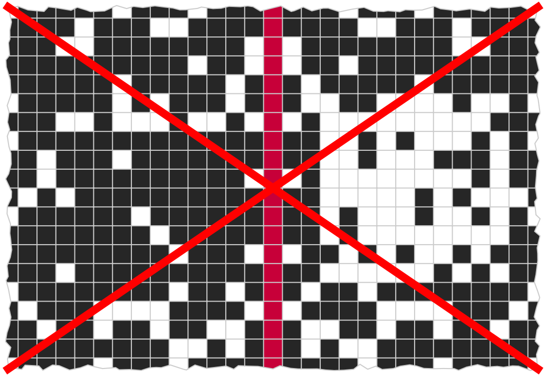
(folklore)

Soficity of shift spaces on $\mathbb{Z}^d$: expressiveness limits

Example

The mirror shift space is **not** sofic on $\mathbb{Z}^d$ for any $d \in \mathbb{N}$:

Pattern $O(n^d)$;
Border $O(n^{d-1})$;



(folklore)

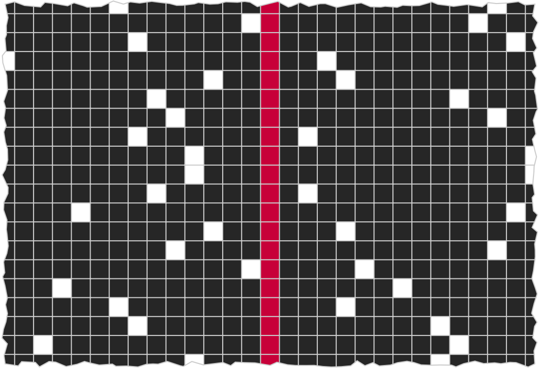
Intuition

Patterns of domain $[\![n]\!]^d$ in a sofic shift can only exchange $O(n^{d-1})$ bits with their exterior.

And most of the time...

Example

Is the shift X_{NEQ} sofic on $\mathbb{Z}^2$?

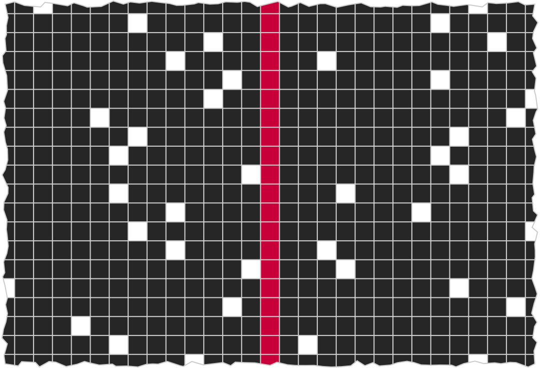


We don't know!

And most of the time...

Example

Is the shift X_{NEQ} sofic on $\mathbb{Z}^2$?



We don't know!

Communication complexity

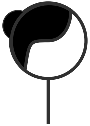
Communication complexity

Definition

Communication complexity quantifies the communication required to solve a distributed problem.

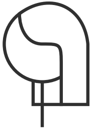
In a **communication problem**, we compute a problem $P\colon \{0,1\}^* \times \{0,1\}^* \to \{\top, \bot\}$.

Alice



$w_A \in \{0,1\}^*$

Bob



$w_B \in \{0,1\}^*$

Communication complexity

Definition

Communication complexity quantifies the communication required to solve a distributed problem.

In a **communication problem**, we compute a problem $P\colon \{0,1\}^* \times \{0,1\}^* \to \{\top, \bot\}$.

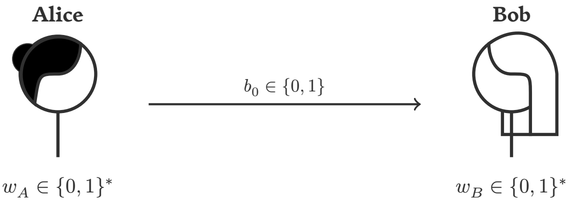
$w_A \in \{0,1\}^*$ $w_B \in \{0,1\}^*$

Communication complexity

Definition

Communication complexity quantifies the communication required to solve a distributed problem.

In a **communication problem**, we compute a problem $P\colon \{0,1\}^* \times \{0,1\}^* \to \{\top, \bot\}$.

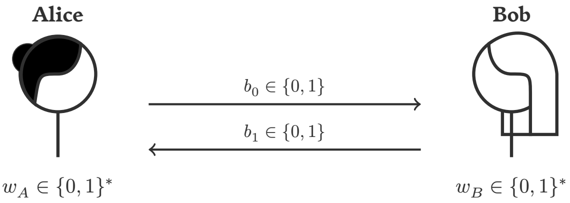


Communication complexity

Definition

Communication complexity quantifies the communication required to solve a distributed problem.

In a **communication problem**, we compute a problem $P\colon \{0,1\}^* \times \{0,1\}^* \to \{\top, \bot\}$.

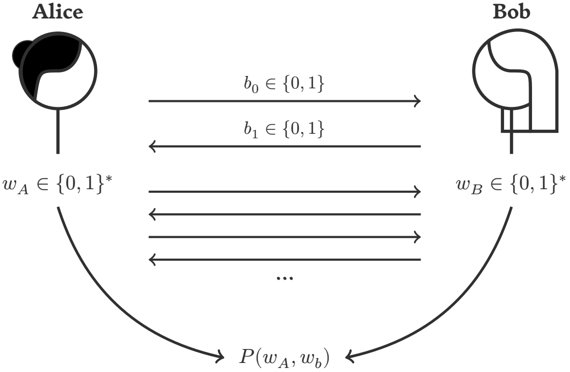


Communication complexity

Definition

Communication complexity quantifies the communication required to solve a distributed problem.

In a **communication problem**, we compute a problem P: $\{0,1\}^* \times \{0,1\}^* \to \{\top, \bot\}$.



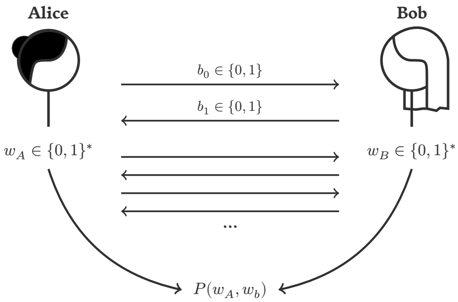
Communication complexity

Definition

For a problem P: $\{0,1\}^* \times \{0,1\}^* \to \{\top, \bot\}$, denote:

- $\mathcal{C}(P)$ the size[1] of a minimal protocol solving P;
- $\mathcal{NC}(P)$ the size of a minimal *non-deterministic*[2] protocol solving P.

[1] Where the *size* of a protocol is the length of its maximal transcript.
[2] If $P(w_A, w_B) = \top$, there must *exist* an accepting communication. Otherwise, all communications must reject.



Example

The problem EQ(n): for two words $w_A, w_B \in \{0,1\}^n$, do we have $w_A = w_B$?

$$\mathcal{C}(\mathrm{EQ}(n)) = \Theta(n);$$
$$\mathcal{NC}(\mathrm{EQ}(n)) = \Theta(n).$$

Example

The problem NEQ(n): for two words $w_A, w_B \in \{0,1\}^n$, do we have $w_A \neq w_B$?

$$\mathcal{C}(\mathrm{NEQ}(n)) = \Theta(n)(= \mathcal{C}(\mathrm{EQ}(n)));$$
$$\mathcal{NC}(\mathrm{NEQ}(n)) = \Theta(\log n).$$

Soficity and communication complexity

Lemma

In sofic spaces, pattern gluing of size n has *non-deterministic* communication complexity $O(n^{d-1})$.

Soficity and communication complexity

Lemma

In sofic spaces, pattern gluing of size n has *non-deterministic* communication complexity $O(n^{d-1})$.

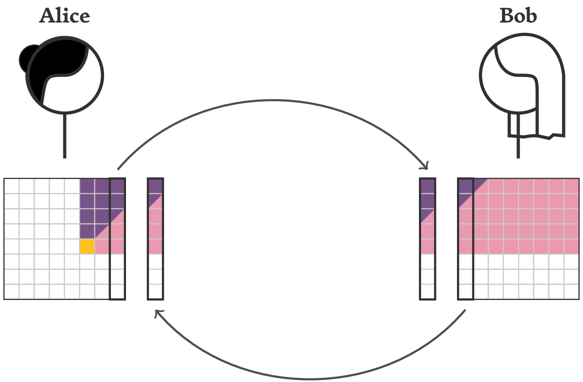
Protocol of complexity $O(n^{d-1})$:

1. Guess a pattern cover;
2. Communicate the border;
3. Check adjacency.

Soficity and communication complexity

Lemma

In sofic spaces, pattern gluing of size n has *non-deterministic* communication complexity $O(n^{d-1})$.



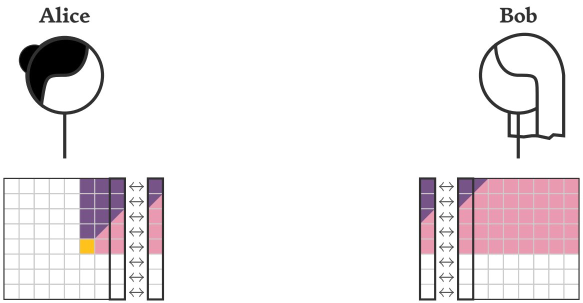
Protocol of complexity $O(n^{d-1})$:

1. Guess a pattern cover;
2. Communicate the border;
3. Check adjacency.

Soficity and communication complexity

Lemma

In sofic spaces, pattern gluing of size n has *non-deterministic* communication complexity $O(n^{d-1})$.



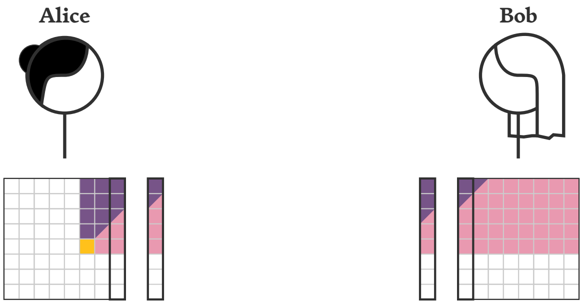
Protocol of complexity $O(n^{d-1})$:

1. Guess a pattern cover;
2. Communicate the border;
3. Check adjacency.

Soficity and communication complexity

Lemma

In sofic spaces, pattern gluing of size n has *non-deterministic* communication complexity $O(n^{d-1})$.



Protocol of complexity $O(n^{d-1})$:

1. Guess a pattern cover;
2. Communicate the border;
3. Check adjacency.

On $\mathbb{Z}$, this is an equivalence!

Proposition ( )

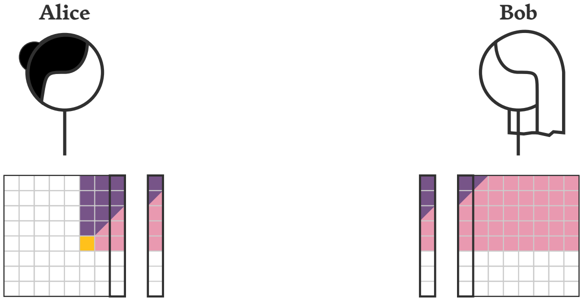
A language $L \subseteq \mathcal{A}^*$ is iff $\mathcal{NC}(L) = O(1)$.[1]

[1] The problem L is the function $w_A, w_B \in \{0,1\}^* \times \{0,1\}^* \mapsto w_A \cdot w_B \overset{?}{\in} L$

Soficity and communication complexity

Lemma

In sofic spaces, pattern gluing of size n has *non-deterministic* communication complexity $O(n^{d-1})$.



Protocol of complexity $O(n^{d-1})$:

1. Guess a pattern cover;
2. Communicate the border;
3. Check adjacency.

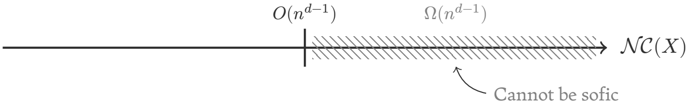
On $\mathbb{Z}$, this is an equivalence!

Proposition ("Myhill-Nerode" revisited)

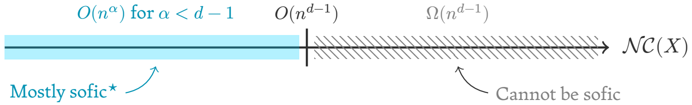
A language $L \subseteq \mathcal{A}^*$ is regular iff $\mathcal{NC}(L) = O(1)$.[1]

[1] The problem L is the function $w_A, w_B \in \{0,1\}^* \times \{0,1\}^* \mapsto w_A \cdot w_B \stackrel{?}{\in} L$

Soficity and communication complexity



Soficity and communication complexity

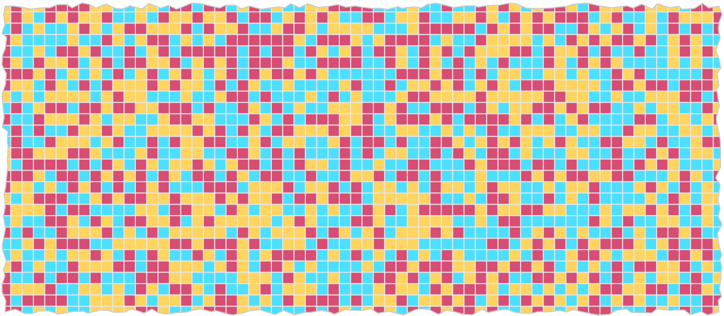


Theorem (C., Paviet Salomon, Vanier; 2026+)

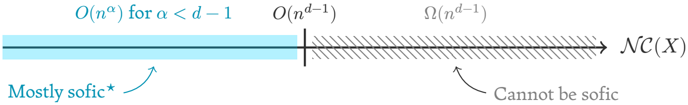
Let $X \subseteq \mathcal{A}^{\mathbb{Z}^d}$ ($d \geq 2$) be an effective shift in which tiling validity can be checked inductively and:

- Communication steps are computable in time $O(n^{\alpha})$ for $\alpha < d-1$;

Then X is a sofic shift space. (Do not quote me on this!)



Soficity and communication complexity

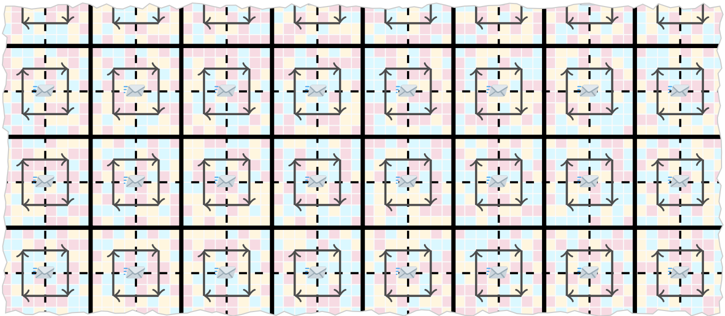


Theorem (C., Paviet Salomon, Vanier; 2026+)

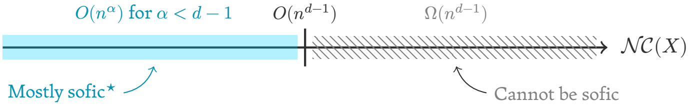
Let $X \subseteq \mathcal{A}^{\mathbb{Z}^d}$ ($d \geq 2$) be an effective shift in which tiling validity can be checked inductively and:

- Communication steps are computable in time $O(n^\alpha)$ for $\alpha < d-1$;

Then X is a sofic shift space. (Do not quote me on this!)



Soficity and communication complexity

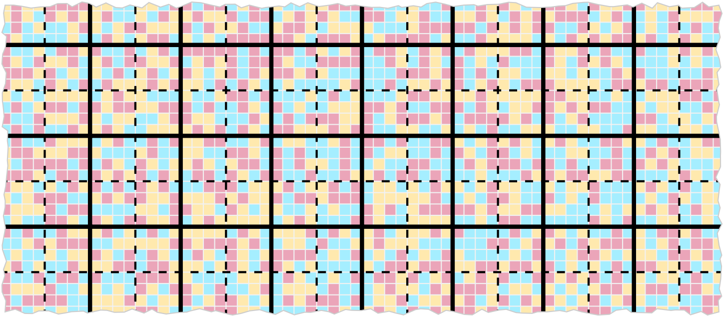


Theorem (C., Paviet Salomon, Vanier; 2026+)

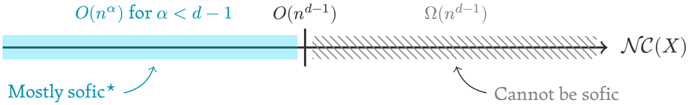
Let $X \subseteq \mathcal{A}^{\mathbb{Z}^d}$ ($d \geq 2$) be an effective shift in which tiling validity can be checked inductively and:

- Communication steps are computable in time $O(n^\alpha)$ for $\alpha < d-1$;

Then X is a sofic shift space. (Do not quote me on this!)



Soficity and communication complexity

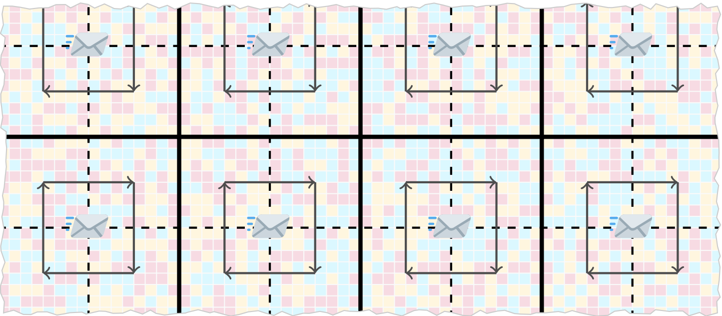


Theorem (C., Paviet Salomon, Vanier; 2026+)

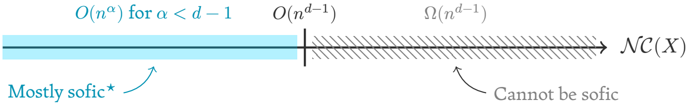
Let $X \subseteq \mathcal{A}^{\mathbb{Z}^d}$ ($d \geq 2$) be an effective shift in which tiling validity can be checked inductively and:

- Communication steps are computable in time $O(n^\alpha)$ for $\alpha < d-1$;

Then X is a sofic shift space. (Do not quote me on this!)



Soficity and communication complexity

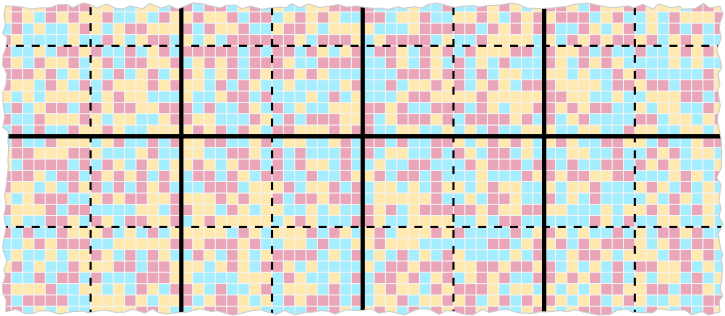


Theorem (C., Paviet Salomon, Vanier; 2026+)

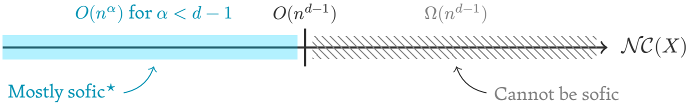
Let $X \subseteq \mathcal{A}^{\mathbb{Z}^d}$ ($d \geq 2$) be an effective shift in which tiling validity can be checked inductively and:

- Communication steps are computable in time $O(n^\alpha)$ for $\alpha < d-1$;

Then X is a sofic shift space. (Do not quote me on this!)



Soficity and communication complexity

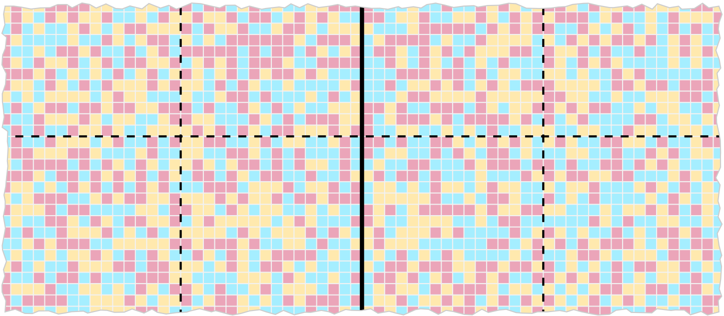


Theorem (C., Paviet Salomon, Vanier; 2026+)

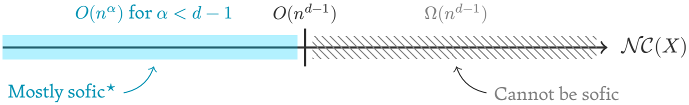
Let $X \subseteq \mathcal{A}^{\mathbb{Z}^d}$ $(d \geq 2)$ be an effective shift in which tiling validity can be checked inductively and:

- Communication steps are computable in time $O(n^\alpha)$ for $\alpha < d-1$;

Then X is a sofic shift space. (Do not quote me on this!)



Soficity and communication complexity

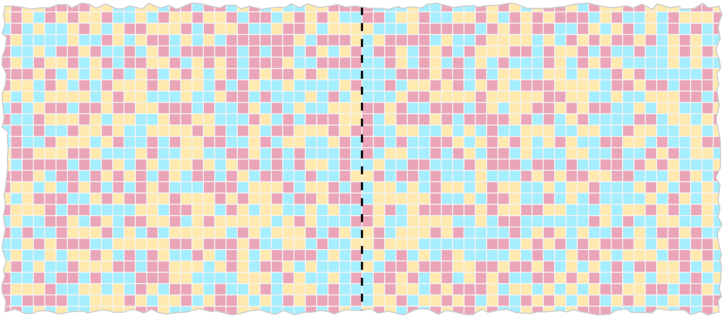


Theorem (C., Paviet Salomon, Vanier; 2026+)

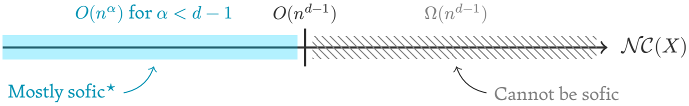
Let $X \subseteq \mathcal{A}^{\mathbb{Z}^d}$ $(d \geq 2)$ be an effective shift in which tiling validity can be checked inductively and:

- Communication steps are computable in time $O(n^\alpha)$ for $\alpha < d-1$;

Then X is a sofic shift space. (Do not quote me on this!)



Soficity and communication complexity

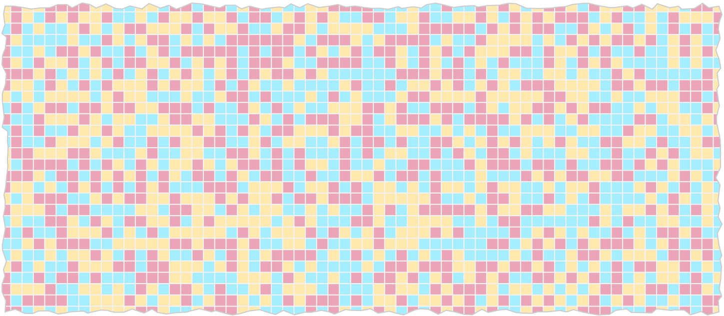


Theorem (C., Paviet Salomon, Vanier; 2026+)

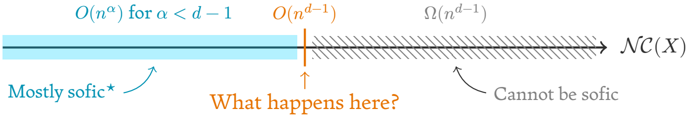
Let $X \subseteq \mathcal{A}^{\mathbb{Z}^d}$ ($d \geq 2$) be an effective shift in which tiling validity can be checked inductively and:

- Communication steps are computable in time $O(n^\alpha)$ for $\alpha < d-1$;

Then X is a sofic shift space. (Do not quote me on this!)



Soficity and communication complexity

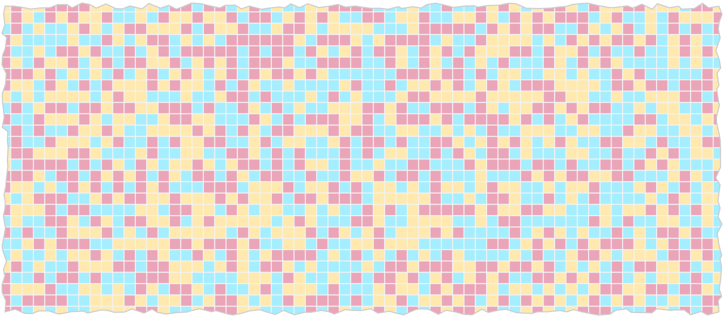


Theorem (C., Paviet Salomon, Vanier; 2026+)

Let $X \subseteq \mathcal{A}^{\mathbb{Z}^d}$ ($d \geq 2$) be an effective shift in which tiling validity can be checked inductively and:

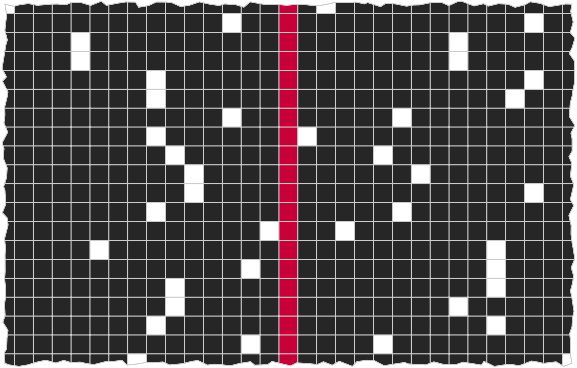
- Communication steps are computable in time $O(n^\alpha)$ for $\alpha < d-1$;

Then X is a sofic shift space. (Do not quote me on this!)



A toy example: the NEQ problem

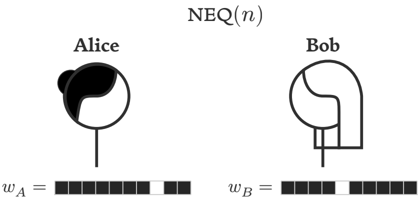
The NEQ problem



The NEQ problem

Definition (NEQ(n))

Define the communication problem NEQ$(n) = \{w_A, w_B \in \{0,1\}^n : |w_A|_1 = |w_B|_1 = 1 \text{ and } w_A \neq \overline{w_B}\}$.



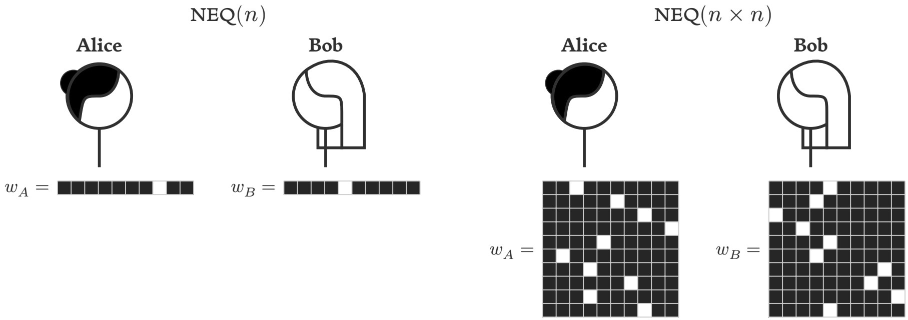
Proposition

$\mathcal{NC}(\text{NEQ}(n)) = O(\log n)$...

The NEQ problem

Definition (NEQ(n))

Define the communication problem NEQ$(n) = \{w_A, w_B \in \{0,1\}^n : |w_A|_1 = |w_B|_1 = 1 \text{ and } w_A \neq \overline{w_B}\}$.



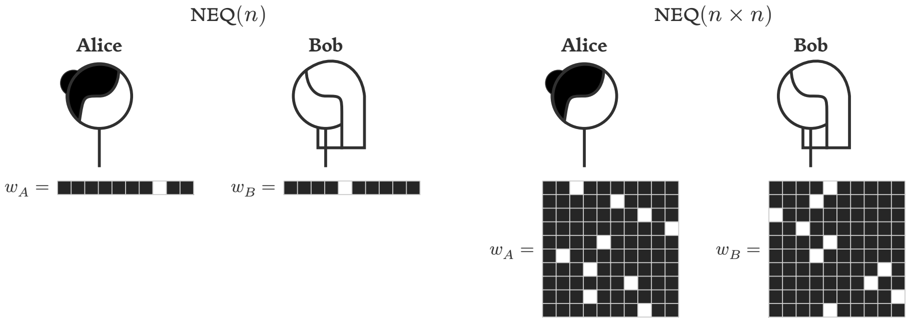
Proposition

$\mathcal{NC}(\text{NEQ}(n)) = O(\log n)$...

The NEQ problem

Definition (NEQ(n))

Define the communication problem NEQ$(n) = \{w_A, w_B \in \{0,1\}^n : |w_A|_1 = |w_B|_1 = 1$ and $w_A \neq \overline{w_B}\}$.



Proposition

$\mathcal{NC}(\text{NEQ}(n)) = O(\log n)$... but $\mathcal{NC}(\text{NEQ}(n \times n)) = O(n)$.

Question: Is the shift X_{NEQ} sofic on $\mathbb{Z}^2$?

A protocol for NEQ$(n \times n)$

Part 1: assuming shared randomness

$r = 0\ 1\ 1\ 0\ 1\ 0\ 1\ 1\ 0\ 1\ 1\ 1\ 0\ \cdots$

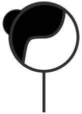
A protocol for NEQ$(n \times n)$

Part 1: assuming shared randomness

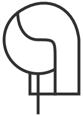
1. Draw a random $n \times n$ matrix M;
2. Alice computes the scalar products $a_i = \langle w_A|_{\{i\}\times\{1,\dots,n\}}, M|_{\{i\}\times\{1,\dots,n\}} \rangle \bmod 2$ (resp. Bob … $b_i = \dots$);
3. Communicate $(a_i)_{1 \le i \le n}$ and $(b_i)_{i \le i \le n}$, and check that all $a_i \ne b_i$.

$r =$ 0 1 1 0 1 0 1 1 0 1 1 1 0 ⋯

Alice



Bob



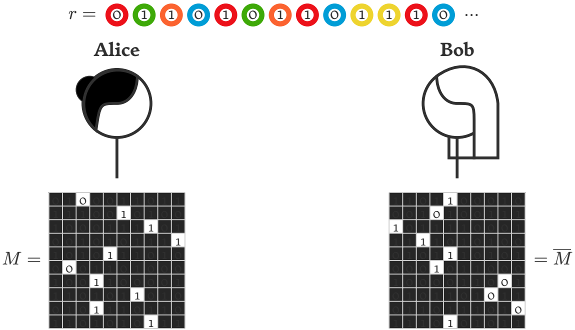
$$M = \begin{array}{|c|c|c|c|c|c|c|c|c|c|} \hline 0&1&0&0&0&1&1&0&1&1 \\ \hline 1&1&1&0&0&1&0&1&0&0 \\ \hline 1&0&0&0&1&1&1&1&0&1 \\ \hline 0&0&1&0&1&0&1&1&1&1 \\ \hline 1&0&1&0&1&1&1&0&1&0 \\ \hline 0&0&0&1&1&0&1&0&0&0 \\ \hline 1&0&0&1&1&0&0&0&0&1 \\ \hline 1&1&0&1&0&1&1&0&1&0 \\ \hline 0&1&0&1&1&1&0&1&1&0 \\ \hline 1&1&0&1&0&1&0&1&1&1 \\ \hline \end{array}$$

$$\begin{array}{|c|c|c|c|c|c|c|c|c|c|} \hline 1&1&0&1&1&0&0&0&1&0 \\ \hline 0&0&1&0&1&0&0&1&1&1 \\ \hline 1&0&1&1&1&1&0&0&0&1 \\ \hline 1&1&1&1&0&1&0&1&0&0 \\ \hline 0&1&0&1&1&1&0&1&0&1 \\ \hline 0&0&0&1&0&1&1&0&0&0 \\ \hline 1&0&0&0&0&1&1&0&0&1 \\ \hline 0&1&0&1&1&0&1&0&1&1 \\ \hline 0&1&1&0&1&1&1&0&1&0 \\ \hline 1&1&1&0&1&0&1&0&1&1 \\ \hline \end{array} = \overline{M}$$

A protocol for NEQ$(n \times n)$

Part 1: assuming shared randomness

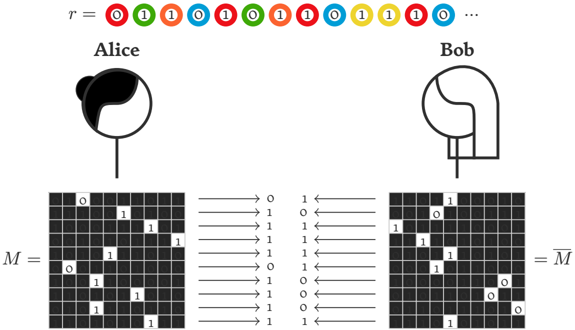
1. Draw a random $n \times n$ matrix M;
2. Alice computes the scalar products $a_i = \langle w_A|_{\{i\}\times\{1,\dots,n\}}, M|_{\{i\}\times\{1,\dots,n\}}\rangle \bmod 2$ (resp. Bob … $b_i = \dots$);
3. Communicate $(a_i)_{1\leq i\leq n}$ and $(b_i)_{i\leq i\leq n}$, and check that all $a_i \neq b_i$.



A protocol for NEQ$(n \times n)$

Part 1: assuming shared randomness

1. Draw a random $n \times n$ matrix M;
2. Alice computes the scalar products $a_i = \langle w_A|_{\{i\}\times\{1,\ldots,n\}}, M|_{\{i\}\times\{1,\ldots,n\}}\rangle \bmod 2$ (resp. Bob … $b_i = \ldots$);
3. Communicate $(a_i)_{1\leq i\leq n}$ and $(b_i)_{i\leq i\leq n}$, and check that all $a_i \neq b_i$.



A protocol for NEQ$(n \times n)$

Part 1: assuming shared randomness

1. Draw a random $n \times n$ matrix M;
2. Alice computes the scalar products $a_i = \langle w_A|_{\{i\}\times\{1,\ldots,n\}}, M|_{\{i\}\times\{1,\ldots,n\}}\rangle \bmod 2$ (resp. Bob ... $b_i = \ldots$);
3. Communicate $(a_i)_{1\leq i\leq n}$ and $(b_i)_{i\leq i\leq n}$, and check that all $a_i \neq b_i$.

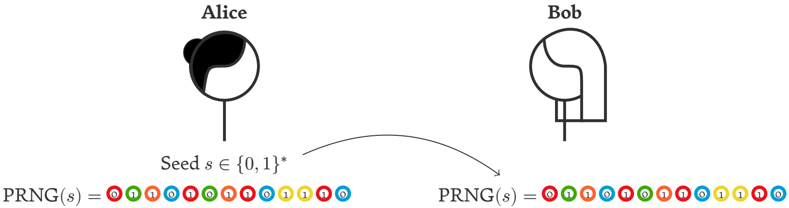
Part 2: removing shared randomness

1. Communicate a non-deterministic seed $s \in \{0,1\}^*$;
2. GOTO [Part 1] using a pseudo-random generator.

A protocol for NEQ$(n \times n)$

Part 1: assuming shared randomness

1. Draw a random $n \times n$ matrix M;
2. Alice computes the scalar products $a_i = \langle w_A|_{\{i\}\times\{1,\ldots,n\}}, M|_{\{i\}\times\{1,\ldots,n\}}\rangle \bmod 2$ (resp. Bob … $b_i = \ldots$);
3. Communicate $(a_i)_{1\leq i\leq n}$ and $(b_i)_{i\leq i\leq n}$, and check that all $a_i \neq b_i$.



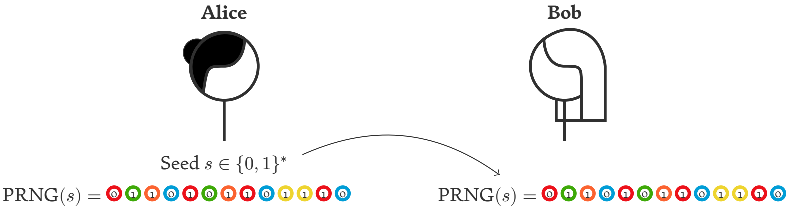
Part 2: removing shared randomness

1. Communicate a non-deterministic seed $s \in \{0,1\}^*$;
2. `GOTO` [Part 1] using a pseudo-random generator.

A protocol for NEQ$(n \times n)$

Part 1: assuming shared randomness

1. Draw a random $n \times n$ matrix M;
2. Alice computes the scalar products $a_i = \langle w_A|_{\{i\}\times\{1,\ldots,n\}}, M|_{\{i\}\times\{1,\ldots,n\}}\rangle \bmod 2$ (resp. Bob … $b_i = \ldots$);
3. Communicate $(a_i)_{1\leq i\leq n}$ and $(b_i)_{i\leq i\leq n}$, and check that all $a_i \neq b_i$.



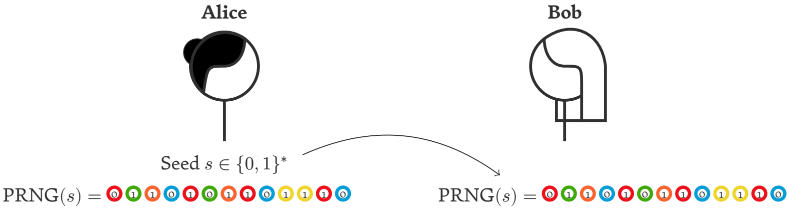
Part 2: removing shared randomness

1. Communicate a non-deterministic seed $s \in \{0,1\}^*$;
2. `GOTO` [Part 1] using a pseudo-random generator.

A protocol for NEQ$(n \times n)$

Part 1: assuming shared randomness

1. Draw a random $n \times n$ matrix M;
2. Alice computes the scalar products $a_i = \langle w_A|_{\{i\}\times\{1,\ldots,n\}}, M|_{\{i\}\times\{1,\ldots,n\}}\rangle \bmod 2$ (resp. Bob … $b_i = \ldots$);
3. Communicate $(a_i)_{1\le i\le n}$ and $(b_i)_{i\le i\le n}$, and check that all $a_i \neq b_i$. Communication $O(n)$ bits



Part 2: removing shared randomness

1. Communicate a non-deterministic seed $s \in \{0,1\}^*$; Communication $O(\log n)$ bits
2. `GOTO` [Part 1] using a pseudo-random generator.

Linear Feedback Shift Registers

Definition (LFSR)

Two words $(f, s) \in \{0,1\}^m \times \{0,1\}^m$ generate a *shift register sequence* $(r_k)_{k\in\mathbb{N}}$ defined by:

$$r_k = \begin{cases} s_k & \text{if k < m} \\ \langle f, r|_{k-m,\ldots,k-1} \rangle & \text{otherwise.} \end{cases}$$

Example

For $f = 0\,0\,0\,0\,1\,1\,0\,1\,1\,1$ and $s = 1\,1\,0\,1\,0\,0\,1\,1\,0\,0$:

$$r = 1\ 1\ 0\ 1\ 0\ 0\ 1\ 1\ 0\ 0$$

Linear Feedback Shift Registers

Definition (LFSR)

Two words $(f, s) \in \{0,1\}^m \times \{0,1\}^m$ generate a *shift register sequence* $(r_k)_{k\in\mathbb{N}}$ defined by:

$$r_k = \begin{cases} s_k & \text{if k < m} \\ \langle f, r|_{k-m,\dots,k-1} \rangle & \text{otherwise.} \end{cases}$$

Example

For $f = 0000110111$ and $s = 1101001100$:

$$r = 1\ 1\ 0\ 1\ 0\ 0\ 1\ 1\ 0\ 0\ 1$$

$$0\ 0\ 0\ 0\ 1\ 1\ 0\ 1\ 1\ 1 \nearrow$$

Linear Feedback Shift Registers

Definition (LFSR)

Two words $(f, s) \in \{0,1\}^m \times \{0,1\}^m$ generate a *shift register sequence* $(r_k)_{k\in\mathbb{N}}$ defined by:

$$r_k = \begin{cases} s_k & \text{if k < m} \\ \langle f, r|_{k-m,\dots,k-1}\rangle & \text{otherwise.} \end{cases}$$

Example

For $f = 0000110111$ and $s = 1101001100$:

$$r = 1\ 1\ 0\ 1\ 0\ 0\ 1\ 1\ 0\ 0\ 1\ 0$$

$$0\ 0\ 0\ 0\ 1\ 1\ 0\ 1\ 1\ 1 \nearrow$$

Linear Feedback Shift Registers

Definition (LFSR)

Two words $(f, s) \in \{0,1\}^m \times \{0,1\}^m$ generate a *shift register sequence* $(r_k)_{k\in\mathbb{N}}$ defined by:

$$r_k = \begin{cases} s_k & \text{if k < m} \\ \langle f, r|_{k-m,\dots,k-1}\rangle & \text{otherwise.} \end{cases}$$

Example

For $f = 0\,0\,0\,0\,1\,1\,0\,1\,1\,1$ and $s = 1\,1\,0\,1\,0\,0\,1\,1\,0\,0$:

$$r = \;1\;1\;0\;1\;0\;0\;1\;1\;0\;0\;1\;0\;1$$

$$\qquad\qquad 0\;0\;0\;0\;1\;1\;0\;1\;1\;1 \nearrow$$

Linear Feedback Shift Registers

Definition (LFSR)

Two words $(f, s) \in \{0,1\}^m \times \{0,1\}^m$ generate a *shift register sequence* $(r_k)_{k \in \mathbb{N}}$ defined by:

$$r_k = \begin{cases} s_k & \text{if k < m} \\ \langle f, r|_{k-m,\dots,k-1} \rangle & \text{otherwise.} \end{cases}$$

Example

For $f = 0000110111$ and $s = 1101001100$:

$$\begin{array}{l} r = 1\ 1\ 0\ 1\ 0\ 0\ 1\ 1\ 0\ 0\ 1\ 0\ 1\ 1 \\ \qquad\qquad\quad 0\ 0\ 0\ 0\ 1\ 1\ 0\ 1\ 1\ 1\nearrow \end{array}$$

Linear Feedback Shift Registers

Definition (LFSR)

Two words $(f, s) \in \{0,1\}^m \times \{0,1\}^m$ generate a *shift register sequence* $(r_k)_{k\in\mathbb{N}}$ defined by:

$$r_k = \begin{cases} s_k & \text{if k < m} \\ \langle f, r|_{k-m,\dots,k-1}\rangle & \text{otherwise.} \end{cases}$$

Example

For $f = 0\,0\,0\,0\,1\,1\,0\,1\,1\,1$ and $s = 1\,1\,0\,1\,0\,0\,1\,1\,0\,0$:

$$r = 1\ 1\ 0\ 1\ 0\ 0\ 1\ 1\ 0\ 0\ 1\ 0\ 1\ 1\ 0$$

$$0\ 0\ 0\ 0\ 1\ 1\ 0\ 1\ 1\ 1 \nearrow$$

Linear Feedback Shift Registers

Definition (LFSR)

Two words $(f, s) \in \{0,1\}^m \times \{0,1\}^m$ generate a *shift register sequence* $(r_k)_{k\in\mathbb{N}}$ defined by:

$$r_k = \begin{cases} s_k & \text{if k < m} \\ \langle f, r|_{k-m,\dots,k-1}\rangle & \text{otherwise.} \end{cases}$$

Example

For $f = 0000110111$ and $s = 1101001100$:

$$r = 1\ 1\ 0\ 1\ 0\ 0\ 1\ 1\ 0\ 0\ 1\ 0\ 1\ 1\ 0$$

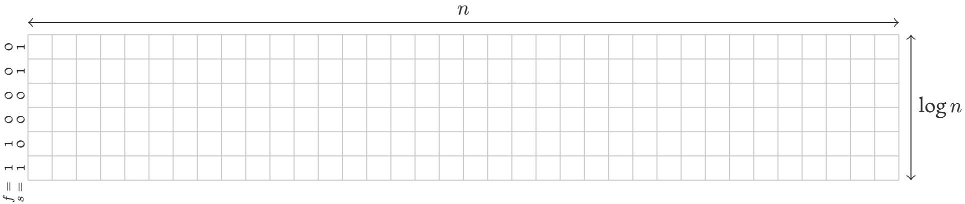
$$0\ 0\ 0\ 0\ 1\ 1\ 0\ 1\ 1\ 1$$

Lemma

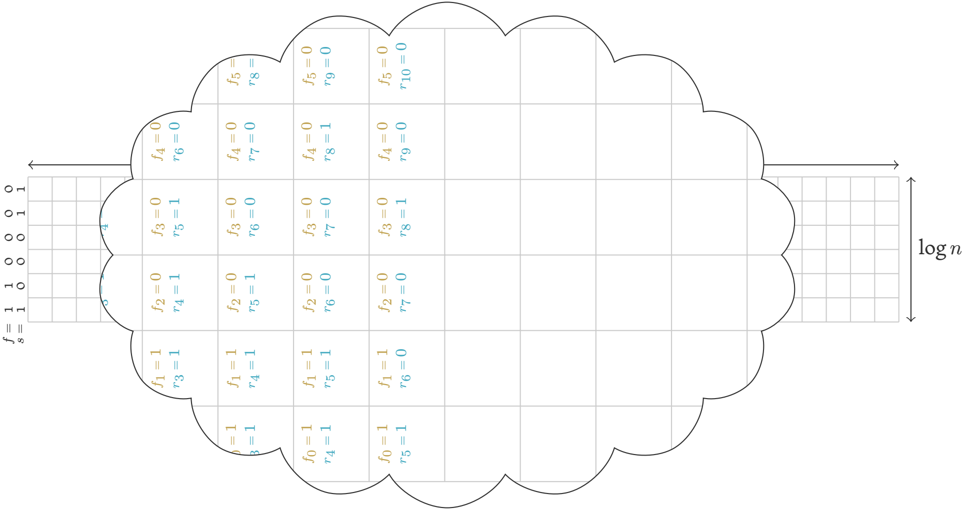
For every $m \in \mathbb{N}$ and indices $0 \le i_1 < \cdots < i_m < 2^m$, the set

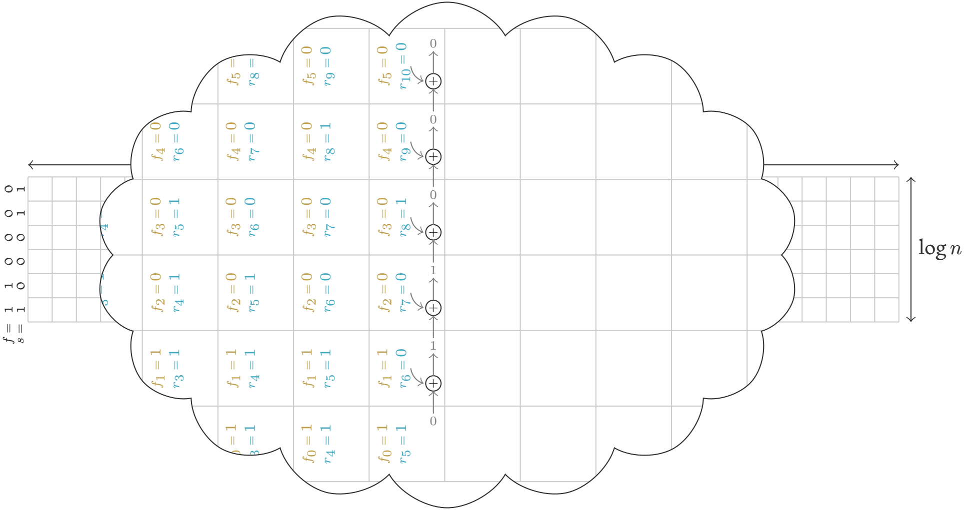
$$\{r|_{i_1,\dots,i_m} \in \{0,1\}^m : r \in \{0,1\}^{\mathbb{N}} \text{ generated by an LFSR of length } O(m)\}$$

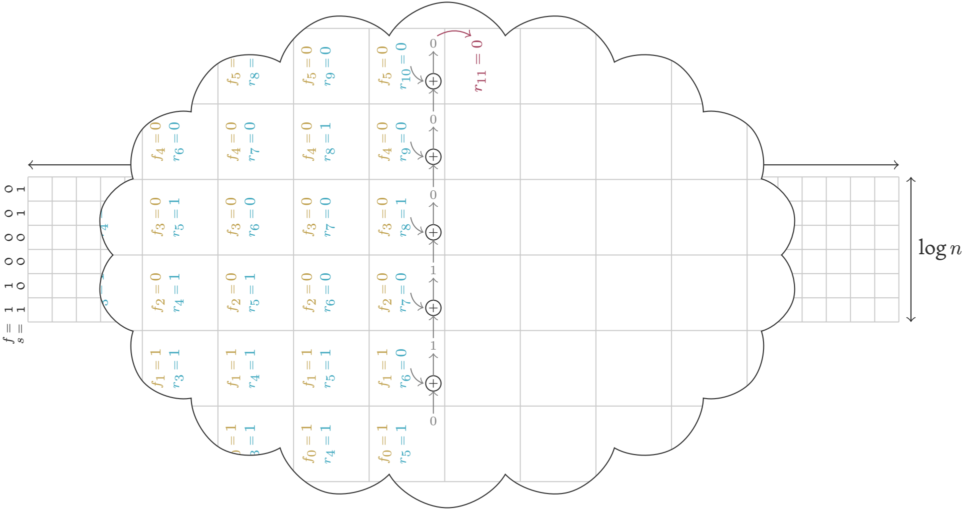
entirely covers $\{0,1\}^m$.

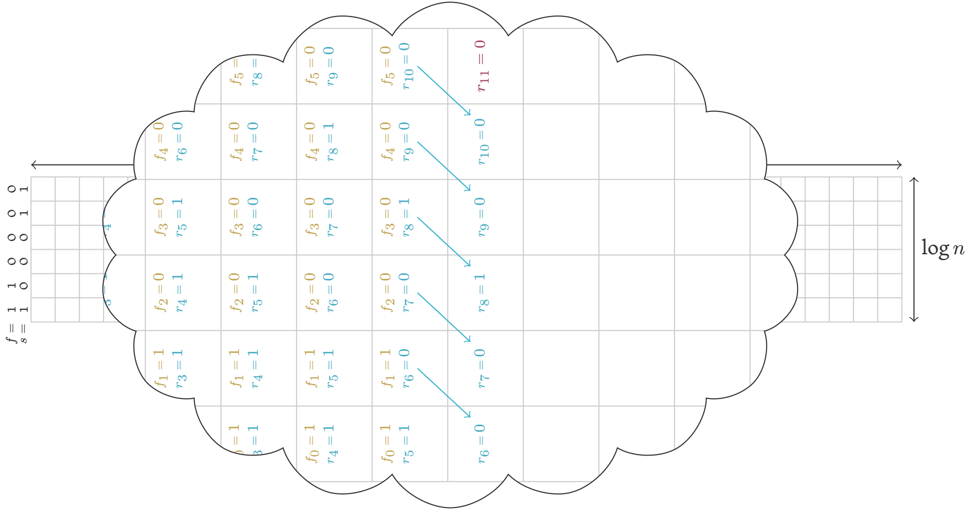


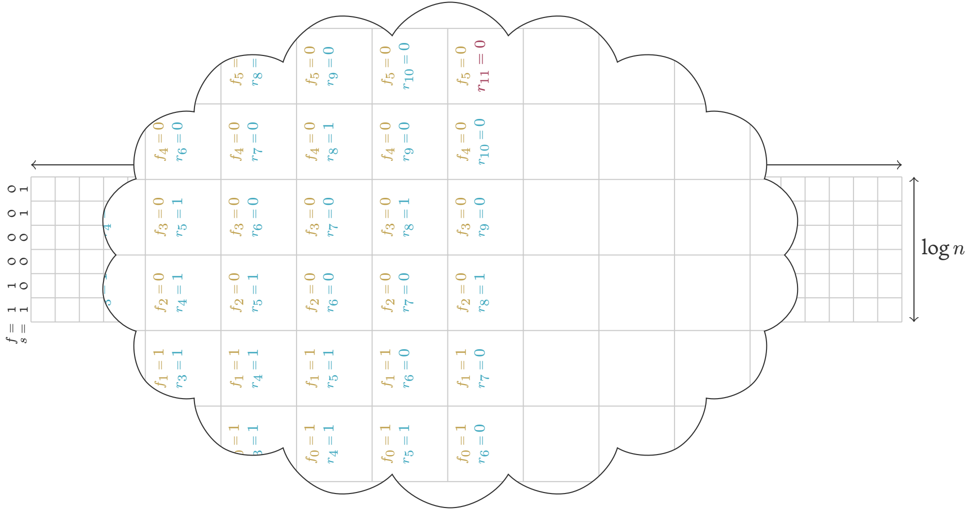
n
$\log n$
$f =$ 1 1 0 0 0 0
$s =$ 1 0 0 0 1 1

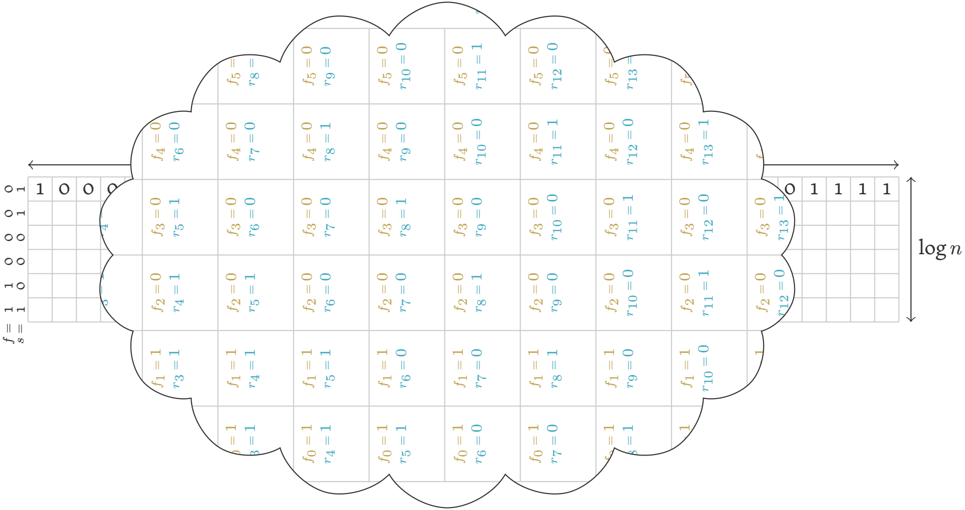

f =
s =
$f_4 = 0$
$r_6 = 0$
$f_3 = 0$
$r_5 = 1$
$f_2 = 0$
$r_4 = 1$
$f_1 = 1$
$r_3 = 1$
$f_4 = 0$
$r_7 = 0$
$f_3 = 0$
$r_6 = 0$
$f_2 = 0$
$r_5 = 1$
$f_1 = 1$
$r_4 = 1$
$f_5 = 0$
$r_9 = 0$
$f_4 = 0$
$r_8 = 1$
$f_3 = 0$
$r_7 = 0$
$f_2 = 0$
$r_6 = 0$
$f_1 = 1$
$r_5 = 1$
$f_0 = 1$
$r_4 = 1$
$f_5 = 0$
$r_{10} = 0$
$f_4 = 0$
$r_9 = 0$
$f_3 = 0$
$r_8 = 1$
$f_2 = 0$
$r_7 = 0$
$f_1 = 1$
$r_6 = 0$
$f_0 = 1$
$r_5 = 1$
$\log n$

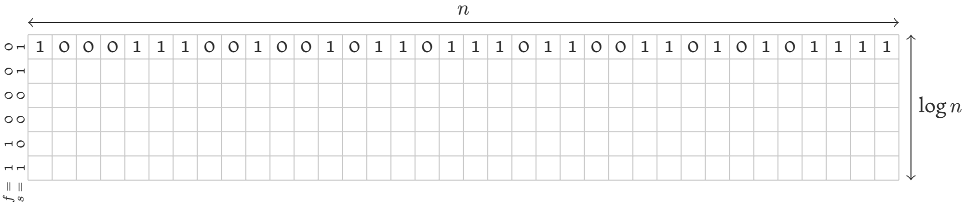

f = 1 1 0 0 0 0
s = 1 0 0 0 1 1
f_4 = 0 r_6 = 0
f_3 = 0 r_5 = 1
f_2 = 0 r_4 = 1
f_1 = 1 r_3 = 1
f_4 = 0 r_7 = 0
f_3 = 0 r_6 = 0
f_2 = 0 r_5 = 1
f_1 = 1 r_4 = 1
f_5 = 0 r_9 = 0
f_4 = 0 r_8 = 1
f_3 = 0 r_7 = 0
f_2 = 0 r_6 = 0
f_1 = 1 r_5 = 1
f_0 = 1 r_4 = 1
f_5 = 0 r_10 = 0
f_4 = 0 r_9 = 0
f_3 = 0 r_8 = 1
f_2 = 0 r_7 = 0
f_1 = 1 r_6 = 0
f_0 = 1 r_5 = 1
0
0
0
0
1
1
0
log n

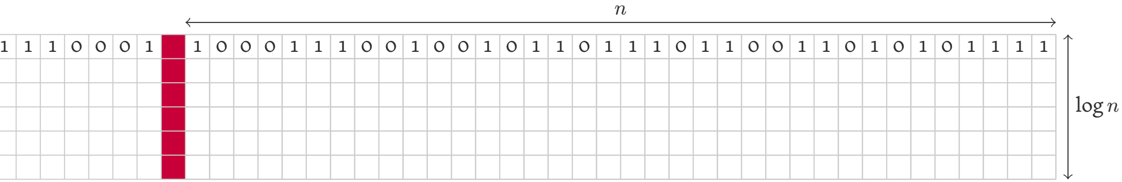

$f =$ 1 1 0 0 0 0
$s =$ 1 0 0 0 1 1
$f_4 = 0$ $r_6 = 0$
$f_3 = 0$ $r_5 = 1$
$f_2 = 0$ $r_4 = 1$
$f_1 = 1$ $r_3 = 1$
$f_4 = 0$ $r_7 = 0$
$f_3 = 0$ $r_6 = 0$
$f_2 = 0$ $r_5 = 1$
$f_1 = 1$ $r_4 = 1$
$f_5 = 0$ $r_9 = 0$
$f_4 = 0$ $r_8 = 1$
$f_3 = 0$ $r_7 = 0$
$f_2 = 0$ $r_6 = 0$
$f_1 = 1$ $r_5 = 1$
$f_0 = 1$ $r_4 = 1$
$f_5 = 0$ $r_{10} = 0$
$f_4 = 0$ $r_9 = 0$
$f_3 = 0$ $r_8 = 1$
$f_2 = 0$ $r_7 = 0$
$f_1 = 1$ $r_6 = 0$
$f_0 = 1$ $r_5 = 1$
$r_{11} = 0$
$\log n$

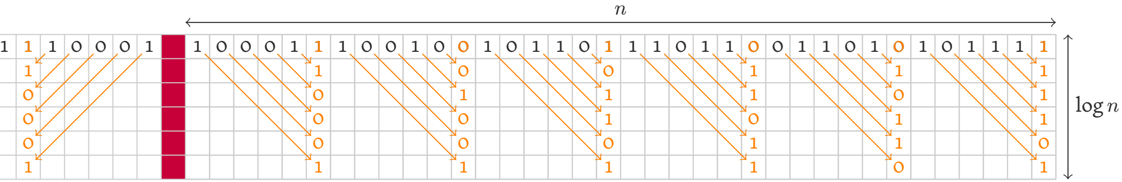

log n


f = 1 1 0 0 0 0
s = 1 0 0 0 1 1
log n

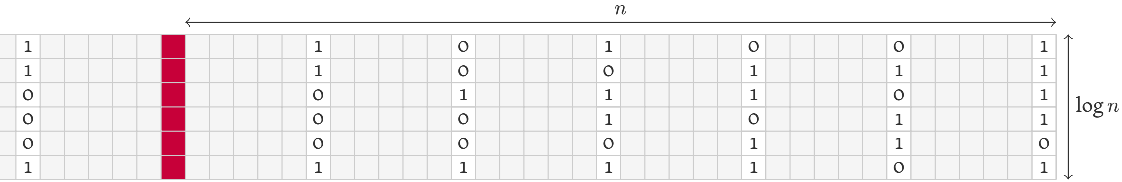

1 0 0 0
0 1 1 1 1
log n
f =
s =

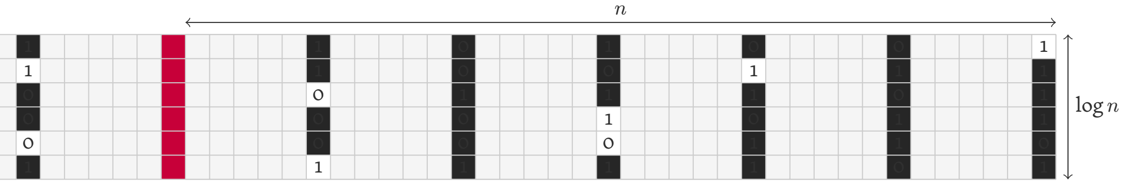

n
1 0 0 0 1 1 1 0 0 1 0 0 1 0 1 1 0 1 1 1 0 1 1 0 0 1 1 0 1 0 1 0 1 1 1 1
log n
f = 1 1 0 0 0 0
s = 1 0 0 0 1 1

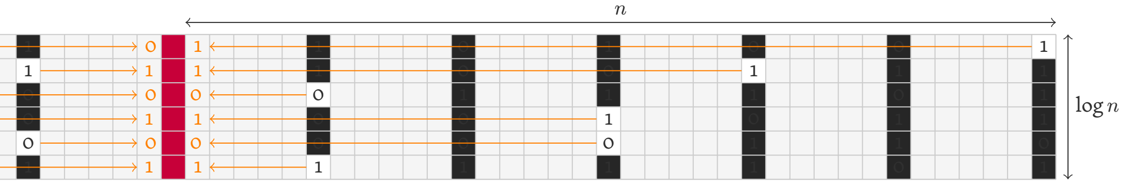

n
1 1 1 0 0 0 1
1 0 0 0 1 1 1 0 0 1 0 0 1 0 1 1 0 1 1 1 0 1 1 0 0 1 1 0 1 0 1 0 1 1 1 1
$\log n$



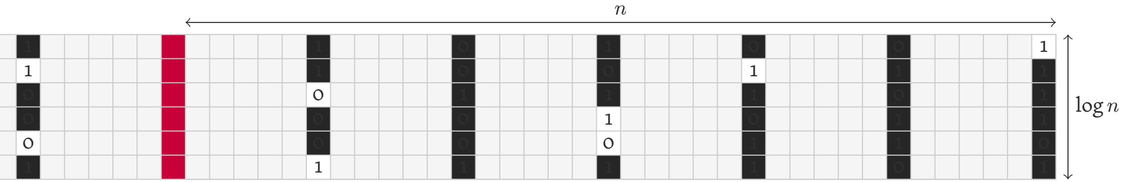
n
$\log n$


n
1 1 0 1 0 0 1
1 1 0 0 1 1 1
0 0 1 1 1 0 1
0 0 0 1 0 1 1
0 0 0 0 1 1 0
1 1 1 1 1 0 1
log n


n
log n


n
log n

Soficity of X_{NEQ}



Conclusion

If we put the columns at distance $\log n$ from each other, the shift X_{NEQ} becomes sofic.

Soficity of $X_{\mathbf{NEQ}}$: some remarks

By stacking blocks on top of each other, we solve $\textsc{NEQ}(n \times n)$ in a sofic manner:

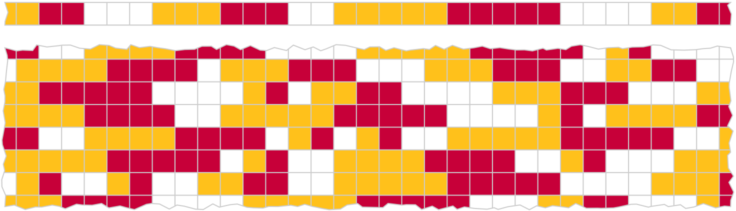
1. *Block encoding*: since $\log n^d = O(\log n)$, we also solve $\textsc{NEQ}(n^d \times n)$;
2. *Iterating the construction*: by piling this construction on top of itself, we can reduce the distance between columns to arbitrary $\log \log \ldots \log n$.
3. *Motivation*: Actually inspired by a question of Guillon and Jeandel:

Conjecture (Guillon & Jeandel, ≃ 2015)

Let $X \subseteq \mathcal{A}^{\mathbb{Z}}$ be a shift space on $\mathbb{Z}$, and assume that $X^{\rightleftharpoons} \subseteq \mathcal{A}^{\mathbb{Z}^2}$ is a $\mathbb{Z}^2$ sofic space

$$X^{\rightleftharpoons} = \{x \in \mathcal{A}^{\mathbb{Z}^2} : \forall n \in \mathbb{Z},\ x|_{\mathbb{Z}\times\{n\}} \in X\}.$$

Is X sofic on $\mathbb{Z}$?



Conclusion

**Sofic tilings generalize regular languages
into infinite and higher-dimensional words.**

Main question

When is a given space $X \subseteq \mathcal{A}^{\mathbb{Z}^d}$ sofic?

We don't know! But: **communication complexity can help!**

- Sofic spaces can synchronize $O(n^{d-1})$ bits across the border of their $[\![1,n]\!]^d$ patterns;
- Inside a domain $[\![1,n]\!]^d$, a limited amount of computations can be performed.
  - About X_{NEQ}: are there "space-efficient" PRNG?
  - About X_{NEQ}: can we avoid moving the columns away from each other? (Maybe with a cellular automaton?)

Future work: open questions about multidimensional soficity.

- Jeandel's conjecture about the complexity of free extensions;
- Weiss' conjecture about entropies of local covers.


That's all Folks!