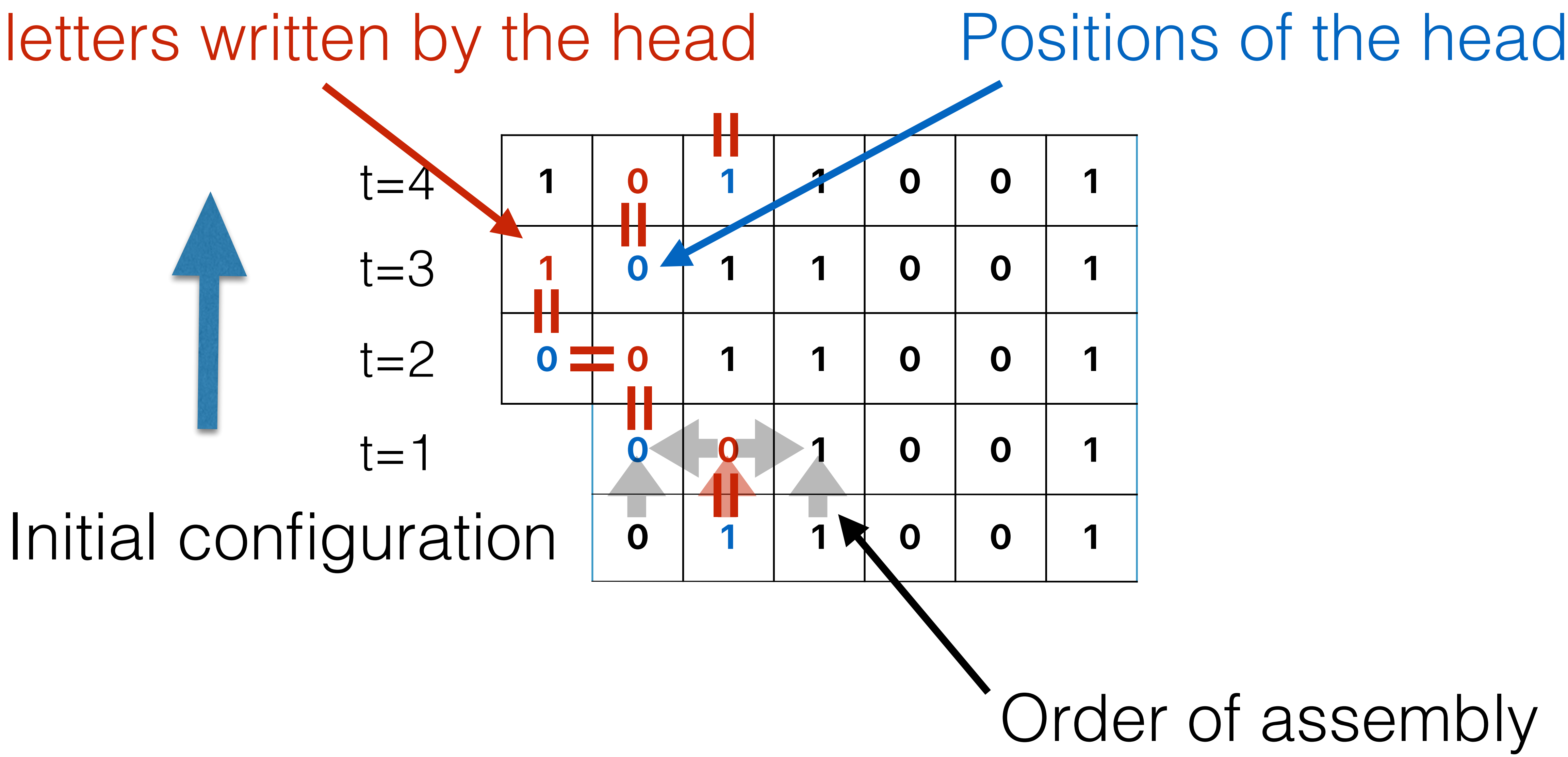
Universality in assembly models

Nicolas Schabanel
CNRS
LIP & IXXI - ÉNS de Lyon

Universality in algorithmic self-assembly

- Can we decide if an assembly terminates?
- Can we design a tile system for a given shape?
- Do I need more than one tileset?
- Do I need more than one tile?
- Do I need more than one molecule?

Algorithmic self-assembly simulates any Turing machine at T°2 in 2D



Algorithmic self-assembly simulates any Turing machine at T°2 in 2D

Position and state of the head

Current tape right end

t=4	0	0	1,Halt	1	★
t=3	0	0	0,q'''	1	★
t=2	0	0	0	0,q''	★
t=1	0	0	1,q'	★	
t=0	0	1,q	1	★	

Transitions:

$$(0,q''') \longmapsto (1,\mathrm{Halt},\bullet)$$

$$(0,q'') \longmapsto (1,q''',\leftarrow)$$

$$(1,q') \longmapsto (0,q'',\rightarrow)$$

$$(1,q) \longmapsto (0,q',\rightarrow)$$

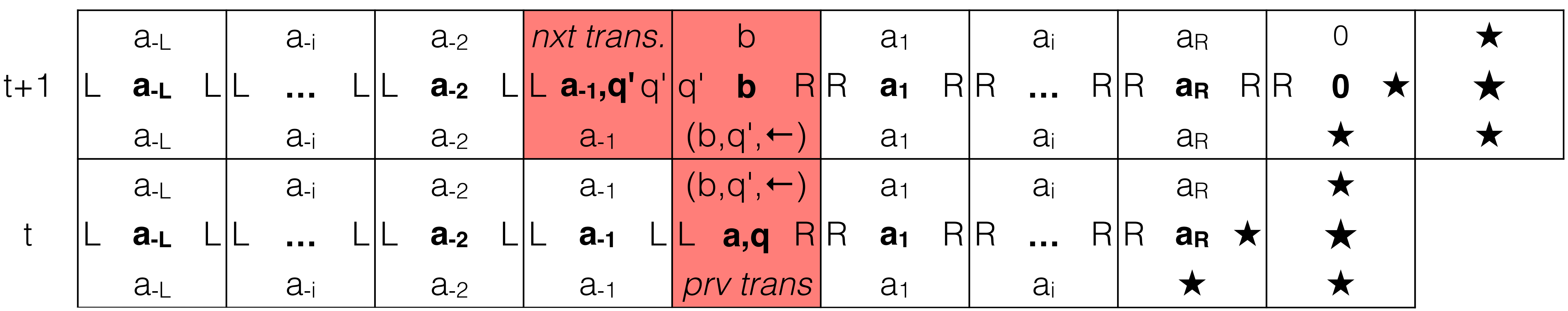
Algorithmic self-assembly simulates any Turing machine at T°2 in 2D

t+1	$\mathbf{a_{-L}}$	**…**	$\mathbf{a_{-2}}$	$\mathbf{a_{-1},q'}$	**b**	$\mathbf{a_1}$	**…**	$\mathbf{a_R}$	**0**	★
t	$\mathbf{a_{-L}}$	**…**	$\mathbf{a_{-2}}$	$\mathbf{a_{-1}}$	**a,q**	$\mathbf{a_1}$	**…**	$\mathbf{a_R}$	★	

Tiles for $(a,q) \longmapsto (b,q',\leftarrow)$

Algorithmic self-assembly simulates any Turing machine at T°2 in 2D

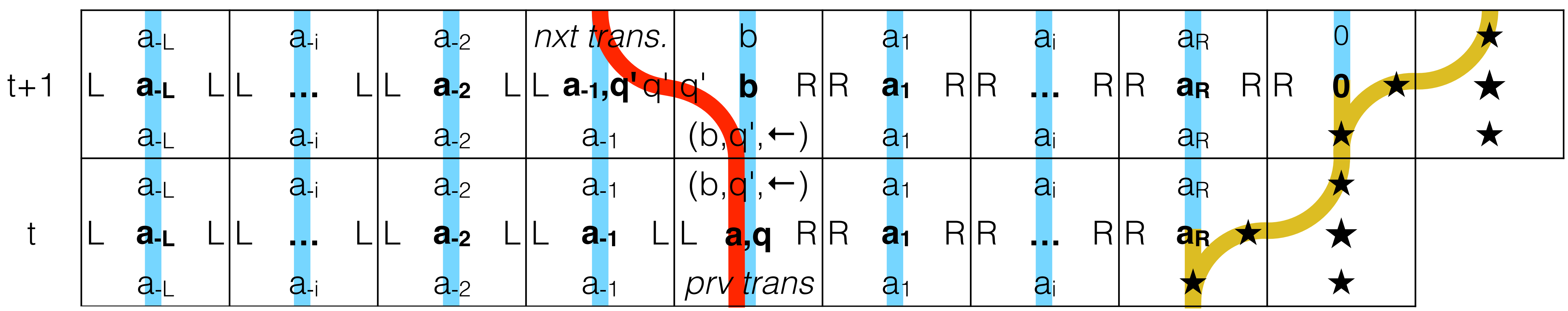
1) Organize the information flow passing through the glues on the sides



Tiles for $(a,q) \longmapsto (b,q',\leftarrow)$

Algorithmic self-assembly simulates any Turing machine at T°2 in 2D

1) Organize the information flow passing through the glues on the sides

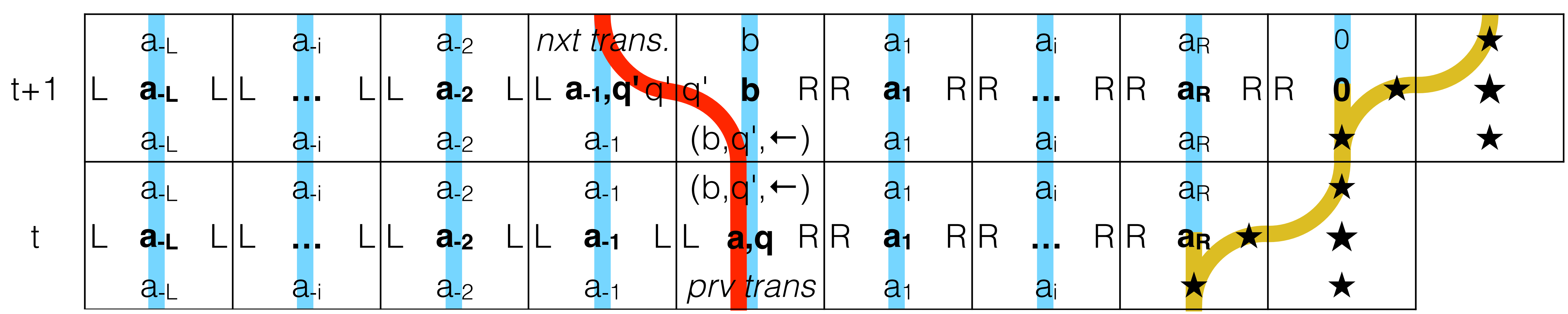


Tiles for $(a,q) \longmapsto (b,q',\leftarrow)$

Algorithmic self-assembly simulates any Turing machine at T°2 in 2D

1) Organize the information flow passing through the glues on the sides

2) Identify each type of tiles



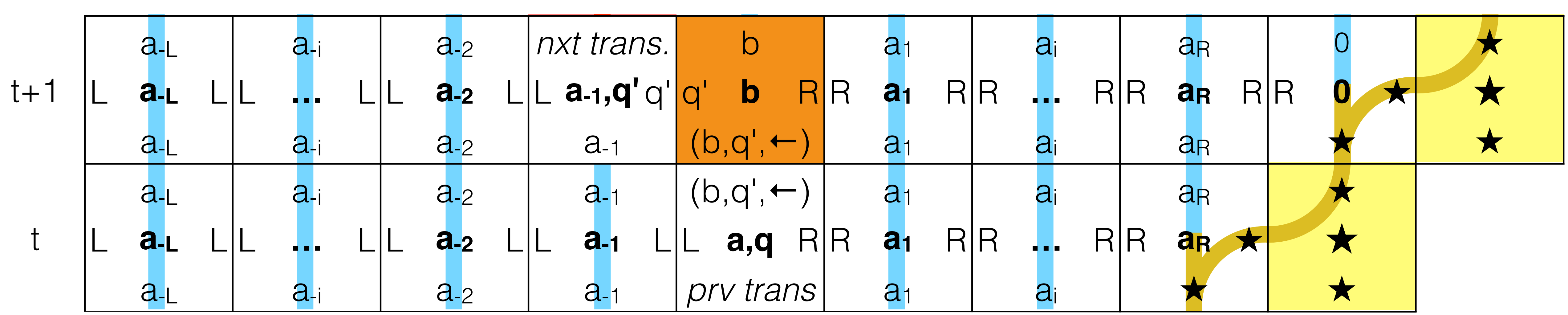
Tiles for (a,q) ⟼ (b,q',←)

Algorithmic self-assembly simulates any Turing machine at T°2 in 2D

1) Organize the information flow passing through the glues on the sides

2) Identify each type of tiles

The skeleton



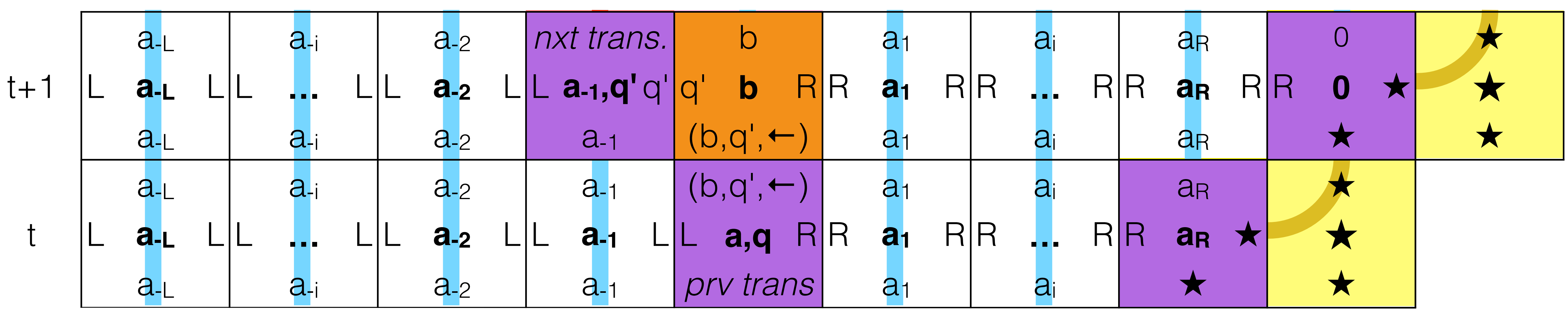
Tiles for (a,q) ⟼ (b,q',←)

Algorithmic self-assembly simulates any Turing machine at T°2 in 2D

1) Organize the information flow passing through the glues on the sides

2) Identify each type of tiles

The carriers



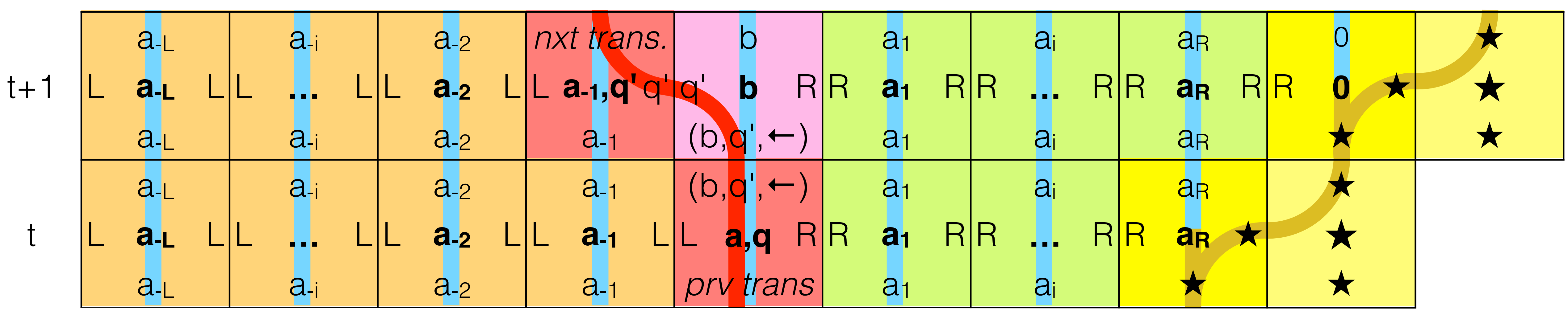
Tiles for $(a,q) \longmapsto (b,q',\leftarrow)$

Algorithmic self-assembly simulates any Turing machine at T°2 in 2D

1) Organize the information flow passing through the glues on the sides

2) Identify each type of tiles

The filling tiles



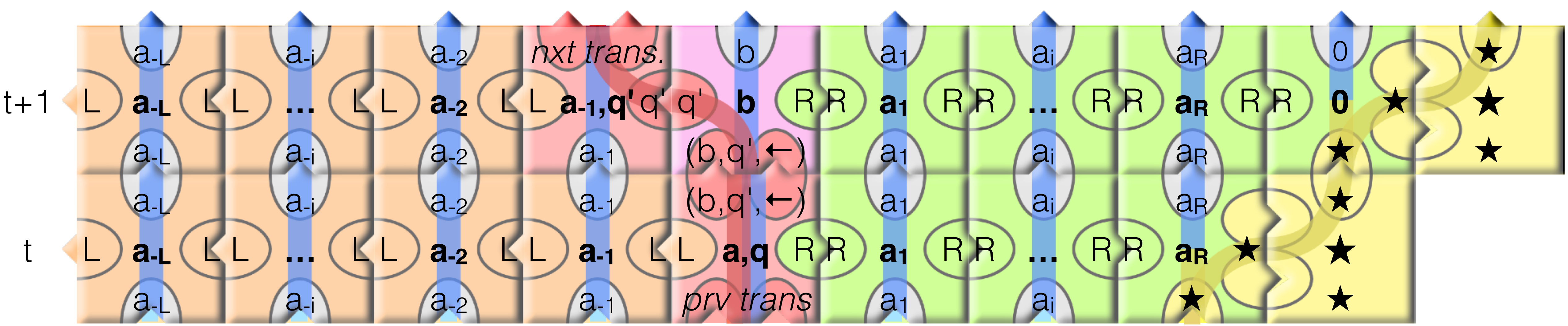
Tiles for (a,q) $\longmapsto$ (b,q',←)

Algorithmic self-assembly simulates any Turing machine at T°2 in 2D

1) Organize the information flow passing through the glues on the sides

2) Identify each type of tiles

3) Set the order, glue strength & colors to match the flow

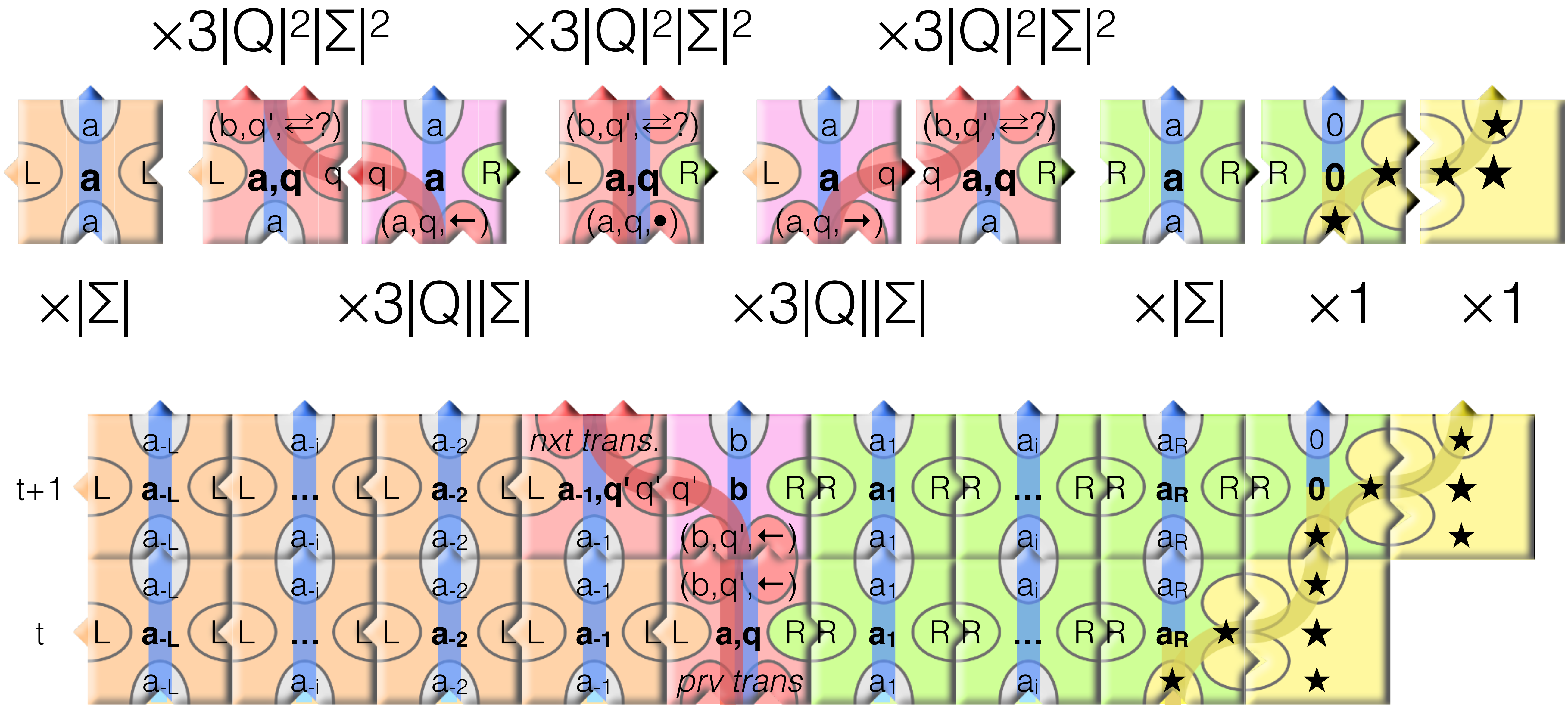


Tiles for (a,q) ⟼ (b,q',←)

The tileset

with size $O(|Q|^2|\Sigma|^2)$

The tiles for each transition $(a,q) \longmapsto (b,q',\rightleftarrows?)$



Encoding the input as a seed

- **Solution 1: hardcoding**
  Uses **n** tiles for n bits

A **0** B	B **0** C	C **1** C	C **1** D	D **0** E	E **1** F	F **1** G	G **1** H	H **0** I	I **0** J	J **1** K	K **1** ★

- Each tile requires **4 log n bits** to encode its glues, thus this encoding uses **4n log n bits** in total
- Can we do better, with less tiles ?

Kolmogorov complexity

- Given a universal Turing machine U:

$$K_U(x) = \min \{ |p| : U(p,\varepsilon) = x \}$$

  is the size of the smallest program in U that outputs x

- **Fact.** $\forall U, \exists A$ s.t. $\forall x$, $K_U(x) \leq |x| + A$

  Proof. A is the size of the program "print" in U.

- **Theorem.** $K_U(x)$ **is independent of U**, indeed:

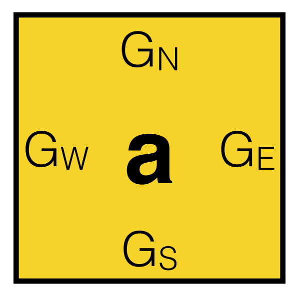
$$\forall U,U' \exists A,B \text{ s.t. } \forall x, K_{U'}(x) - A \leq K_U(x) \leq K_{U'}(x) + B$$

  Proof. A and B are the sizes of the programs that execute U and U' respectively in U' and U.

Lower bounding the required number of tiles to encode the seed

- The Kolmogorov complexity **K(x)** is the size of the smallest program that outputs **x**

- Bit size of the encoding with **T** tiles is $\mathbf{\leq 4\,T \log_2 T}$

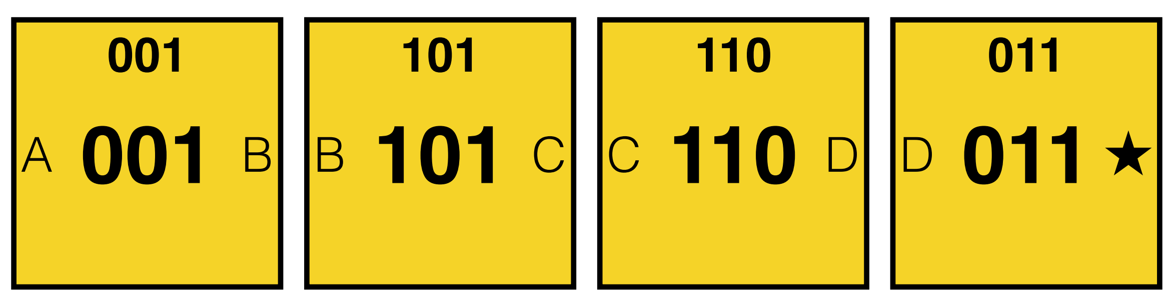


- A tileset that self-assembles x is a program that outputs x,

  Thus: $\mathbf{4T \log T \geq K(x)}$, I.e. $\mathbf{T \geq K(x) / 4 \log K(x)}$

Unpacking
a binary string

- Cut string **x** in **n / b** chunks of **b** bits and uncompress it
- Example: **x = 001 101 110 011** and **b = 3**

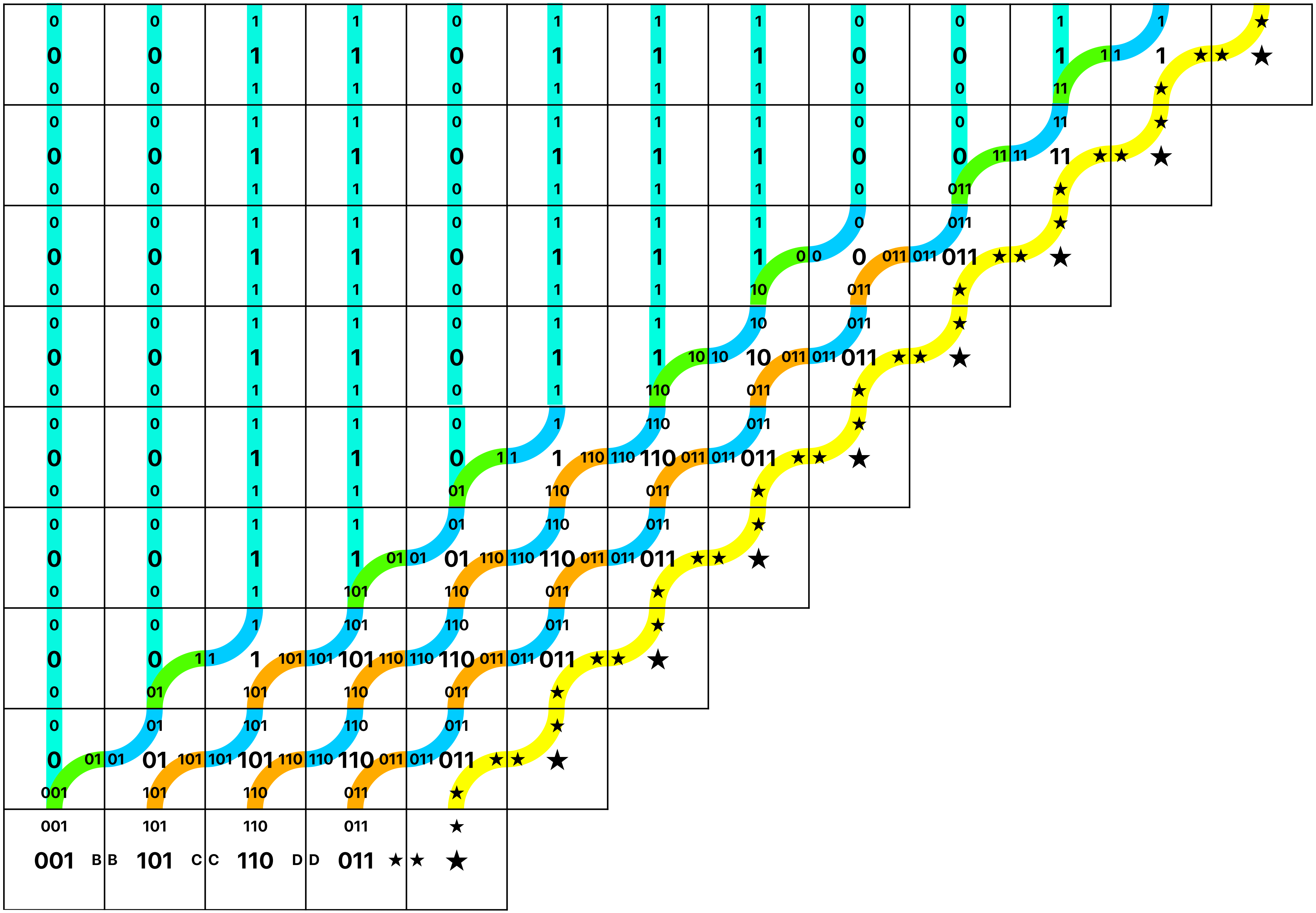


n / b tiles

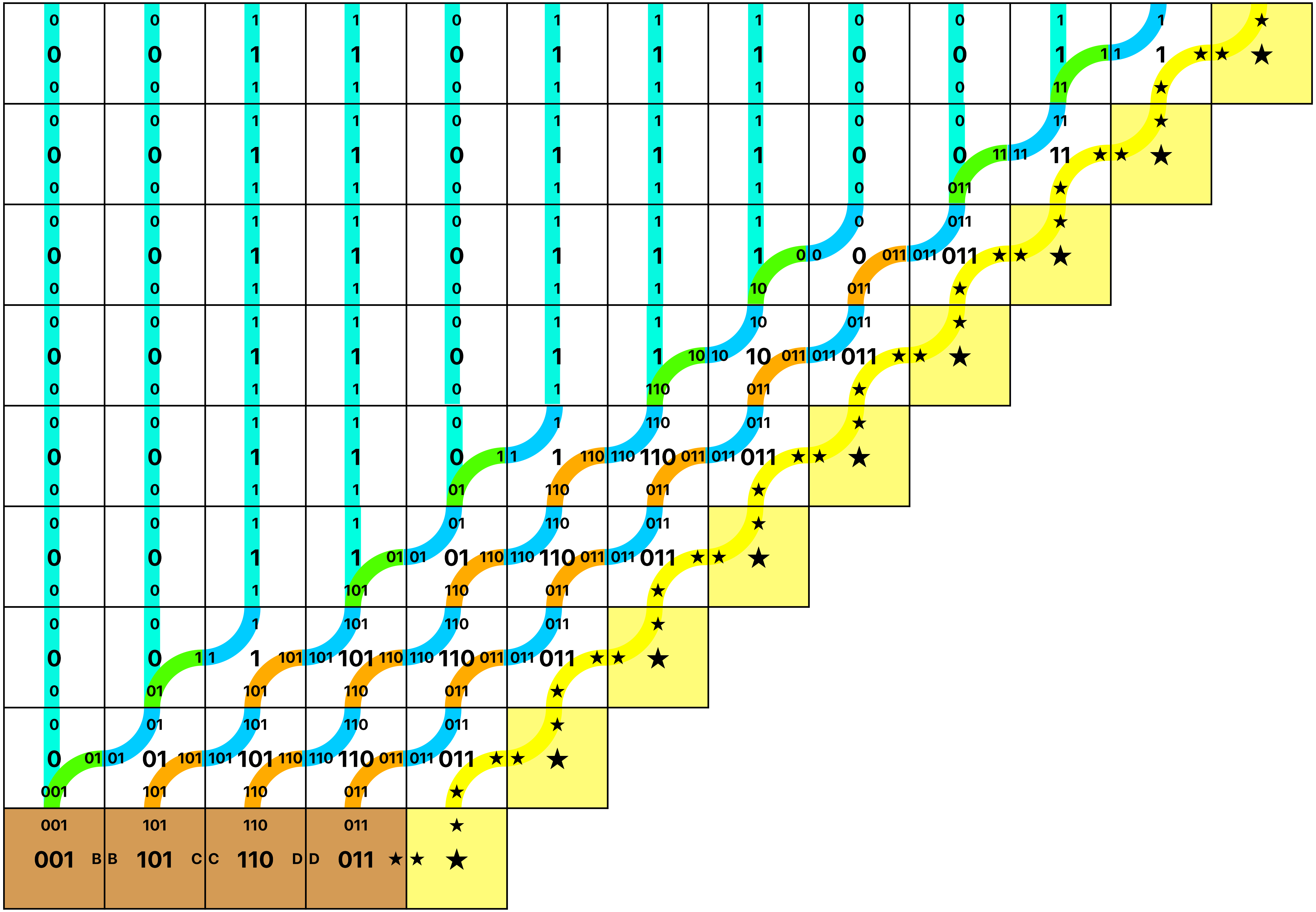
Unpacking a binary string

x unpacked	0	0	1	1	0	1	1	1	0	0	1	1	★
	0	0	1	1	0	1	1	1	0	0	11	★	
	0	0	1	1	0	1	1	1	0	011	★		
	0	0	1	1	0	1	1	10	011	★			
	0	0	1	1	0	1	110	011	★				
	0	0	1	1	01	110	011	★					
	0	0	1	101	110	011	★						
	0	01	101	110	011	★							
x packed	001	101	110	011	★								

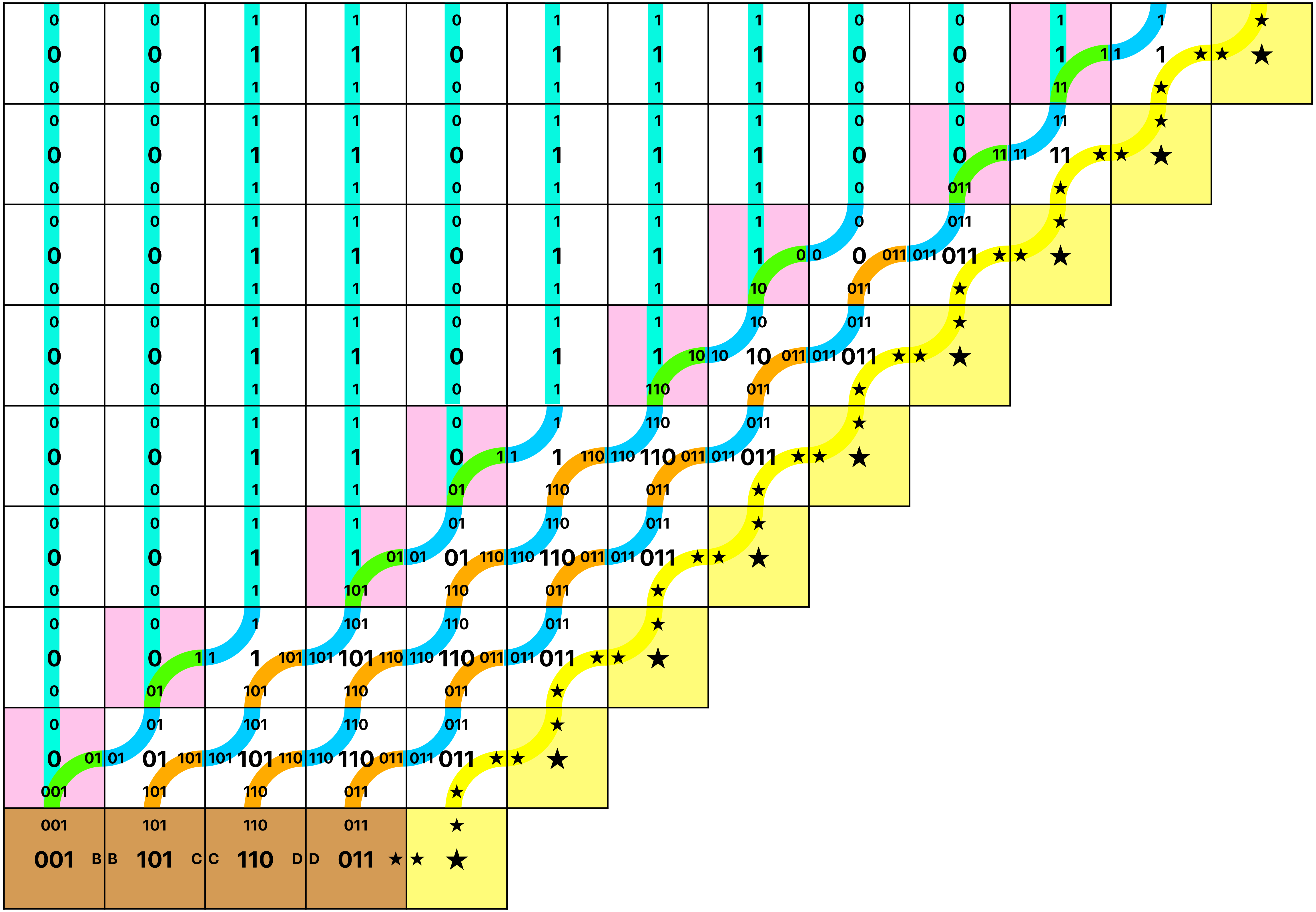
Determine the flow of information



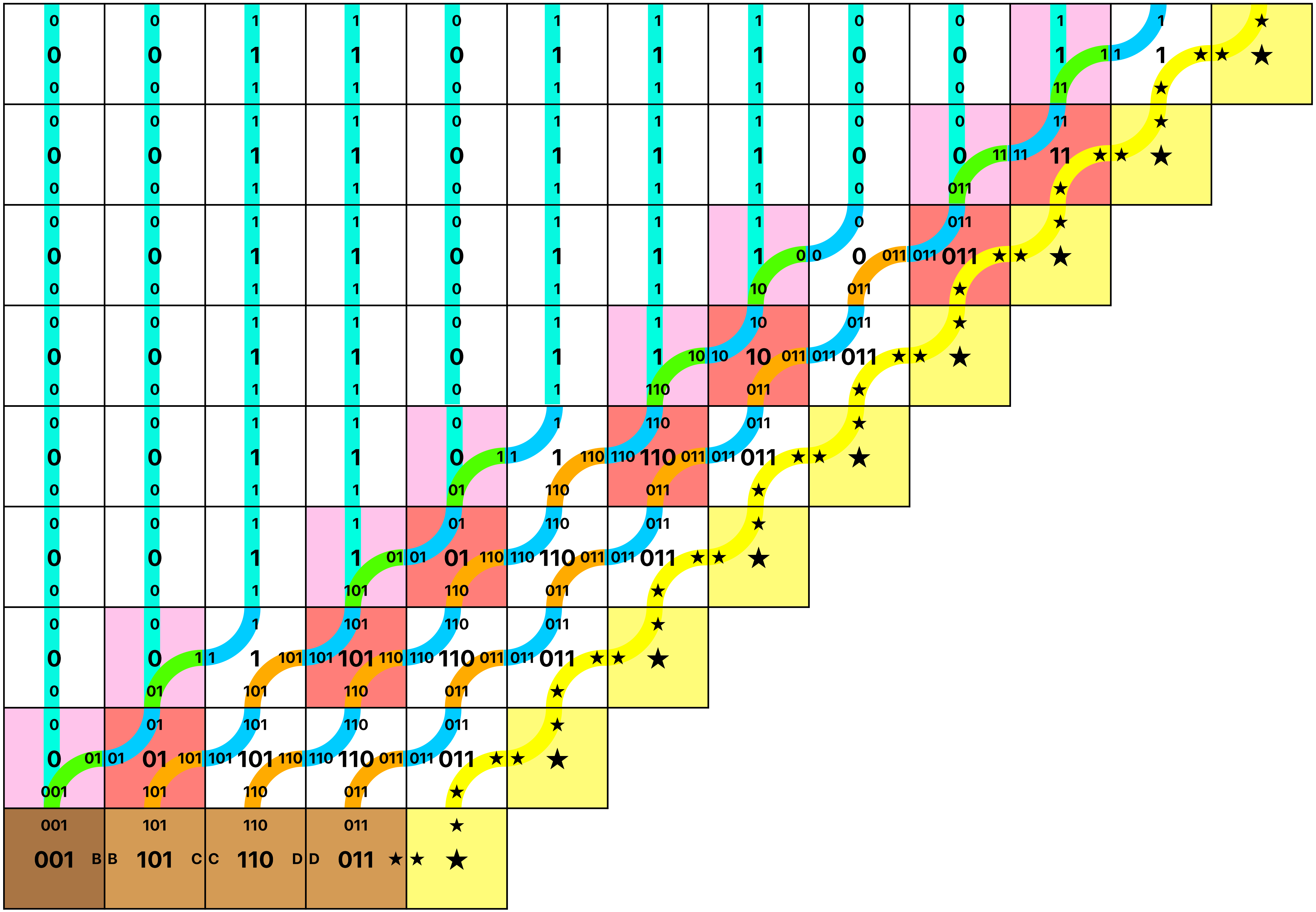
Determine the various tile types



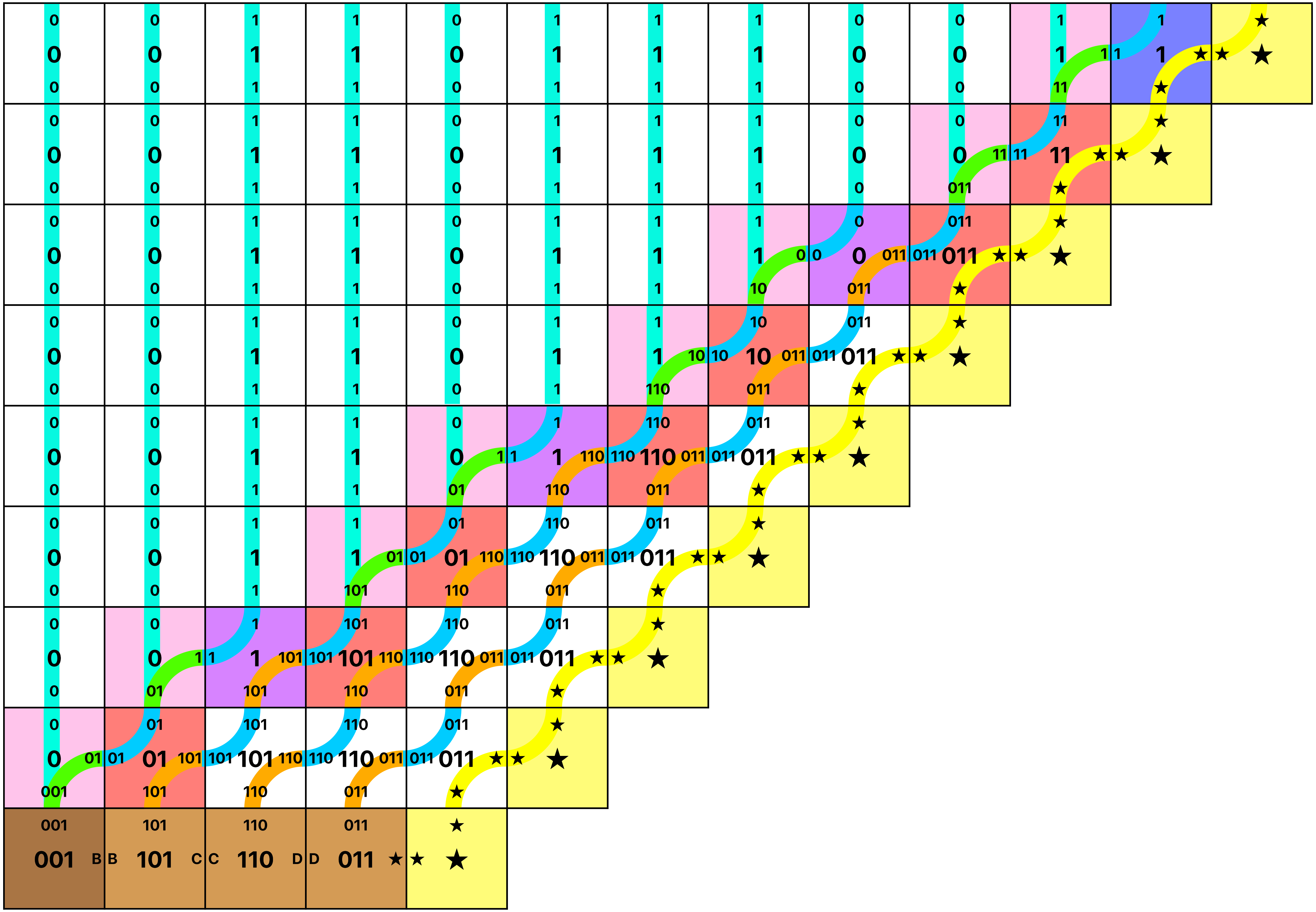
Determine the various tile types



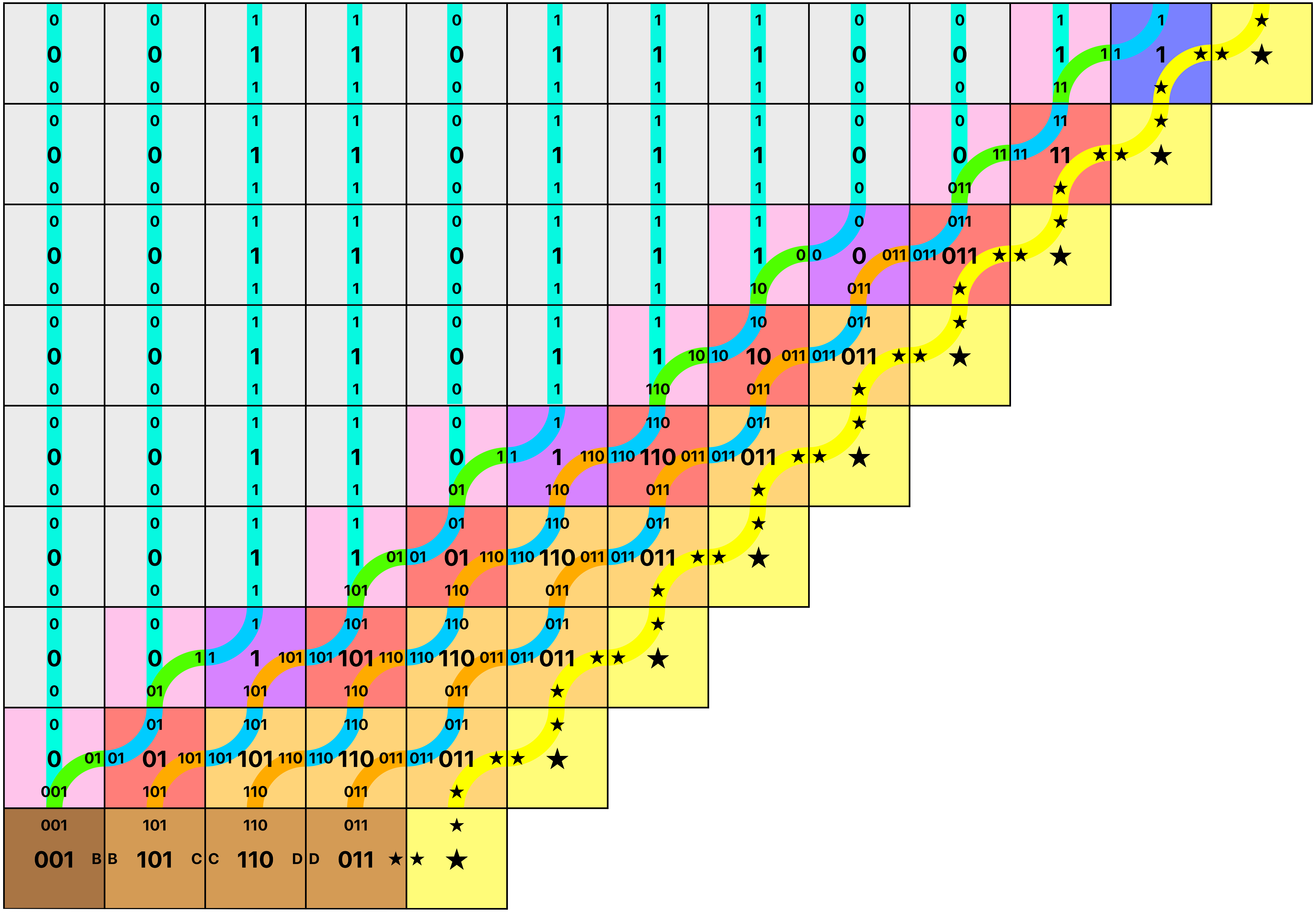
Determine the various tile types

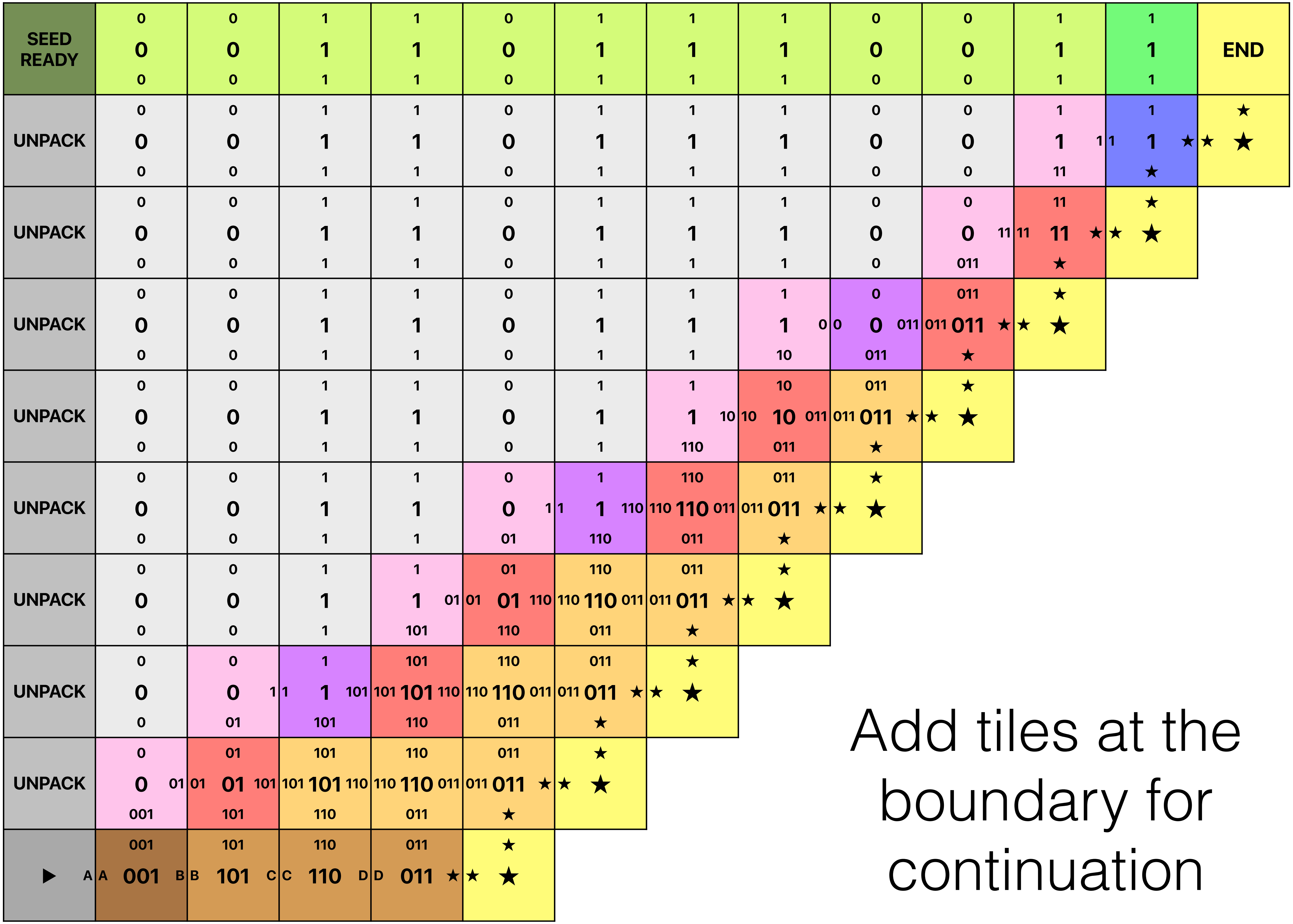


Determine the various tile types

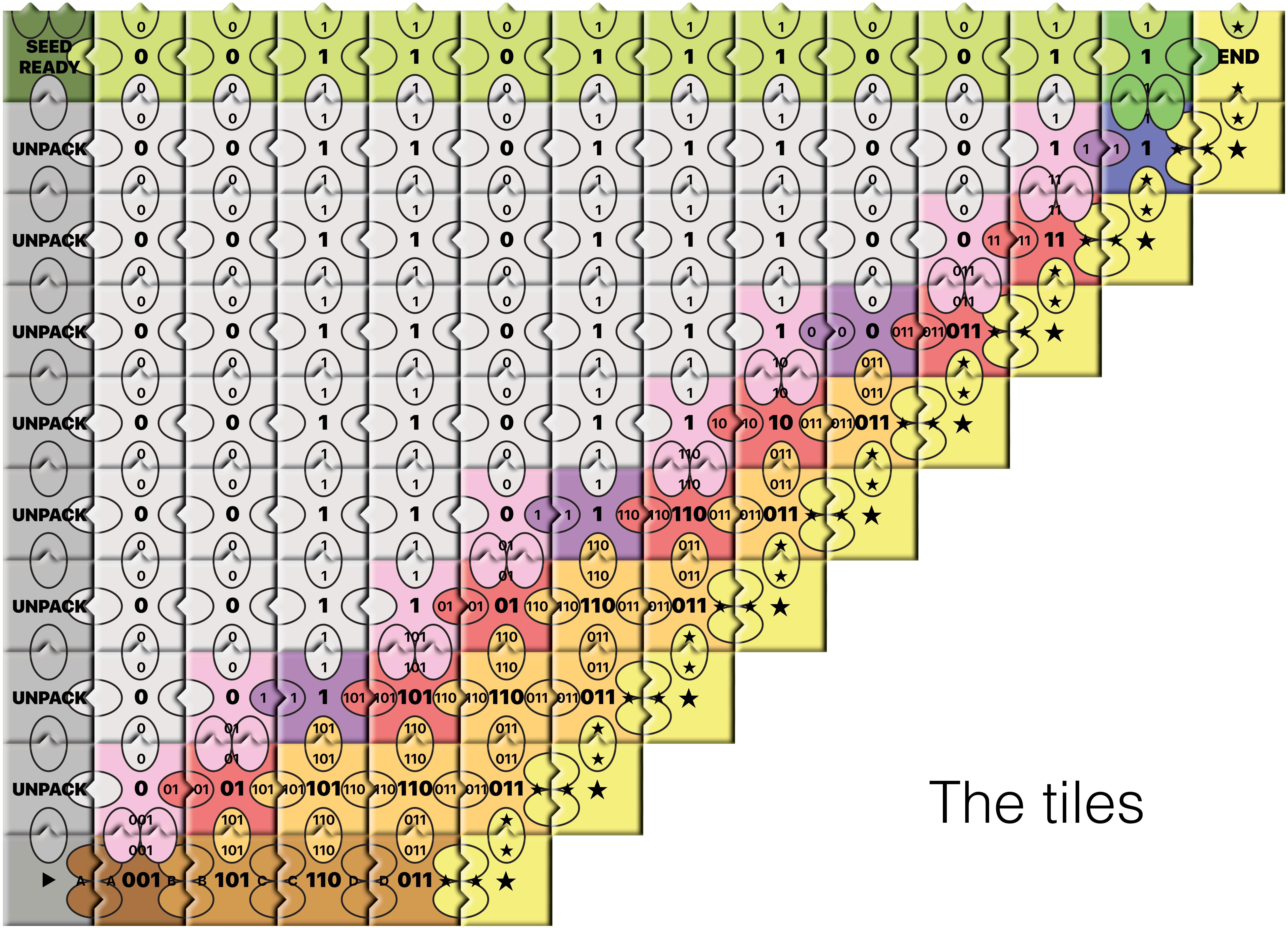


Determine the various tile types

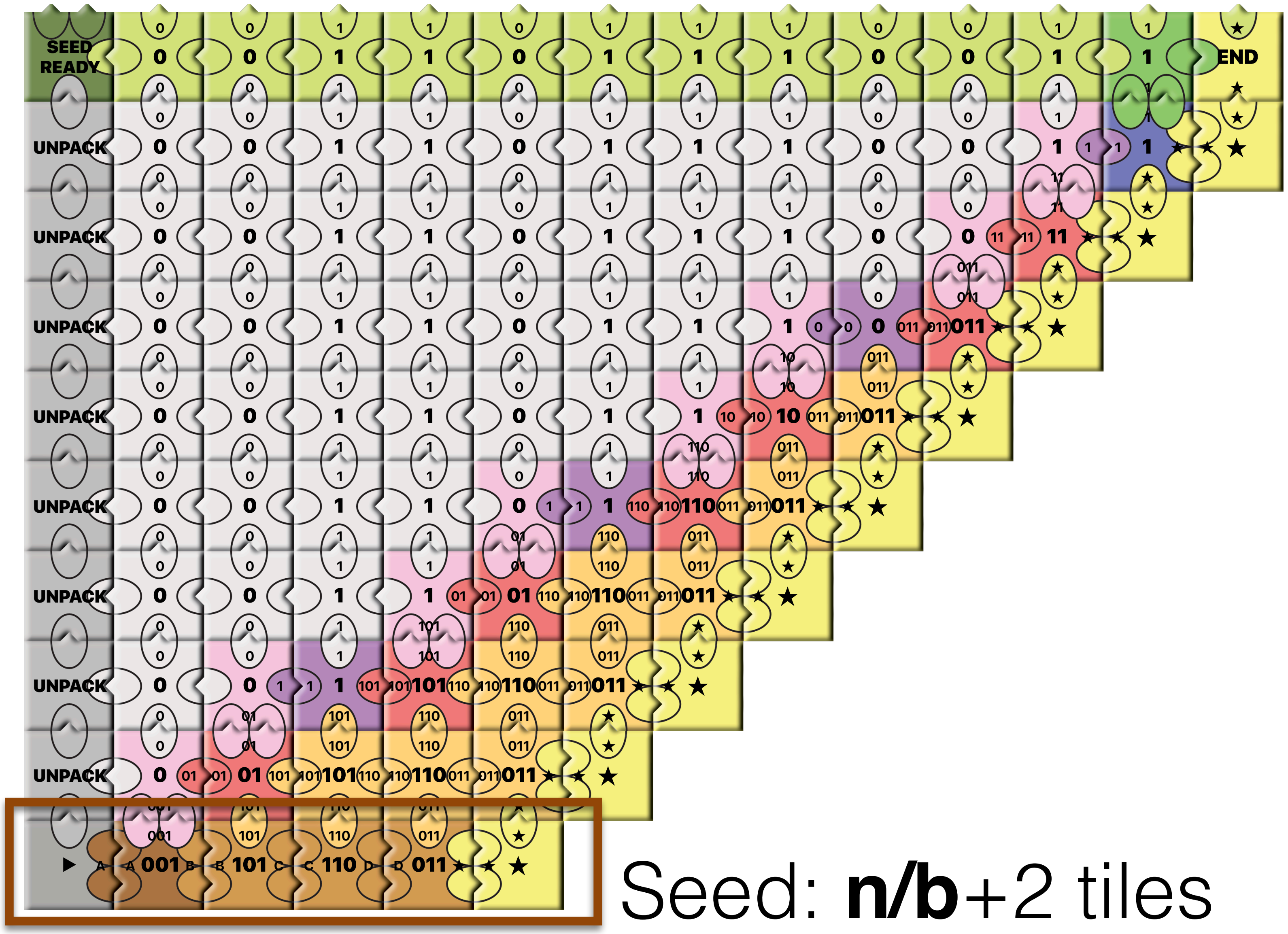




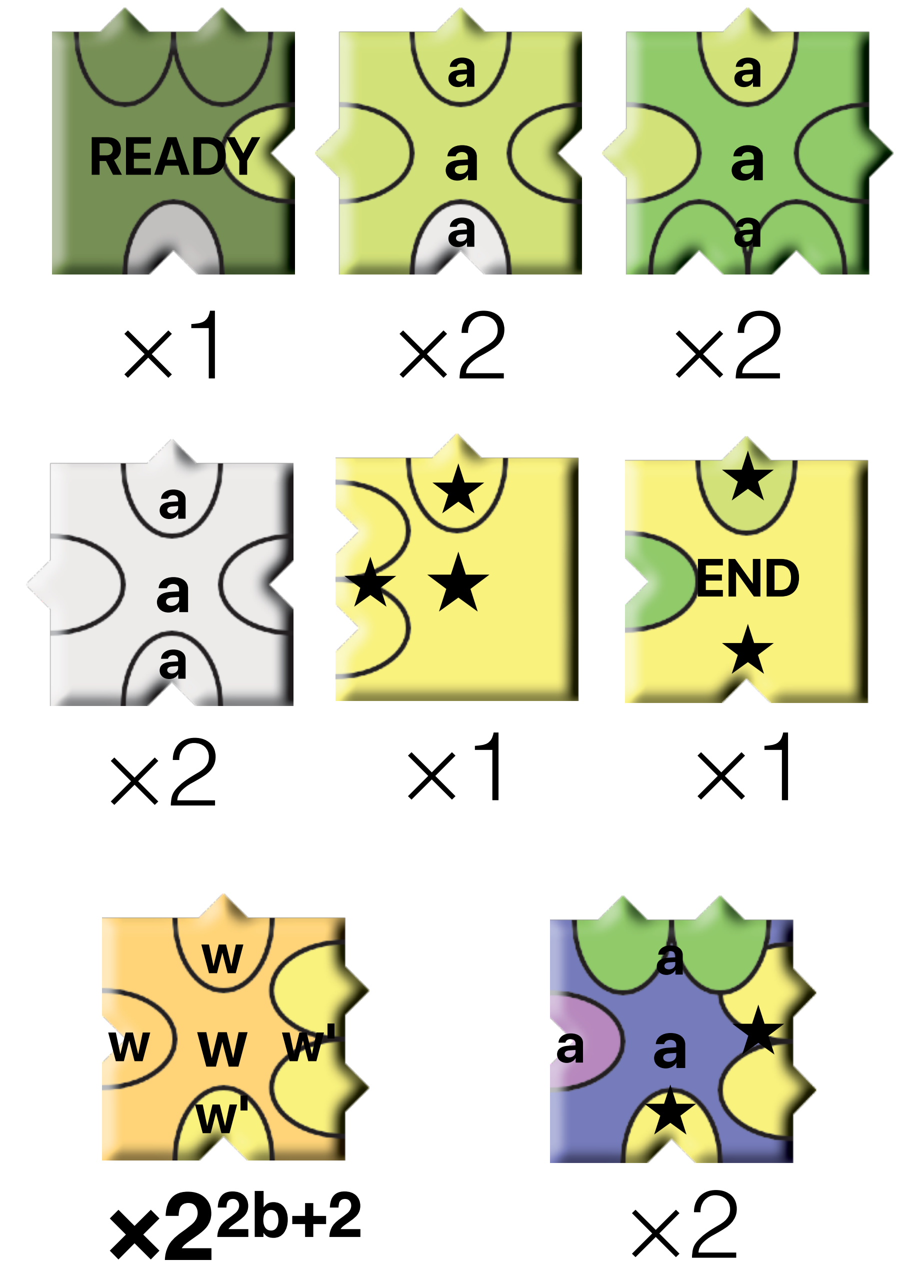
Add tiles at the boundary for continuation

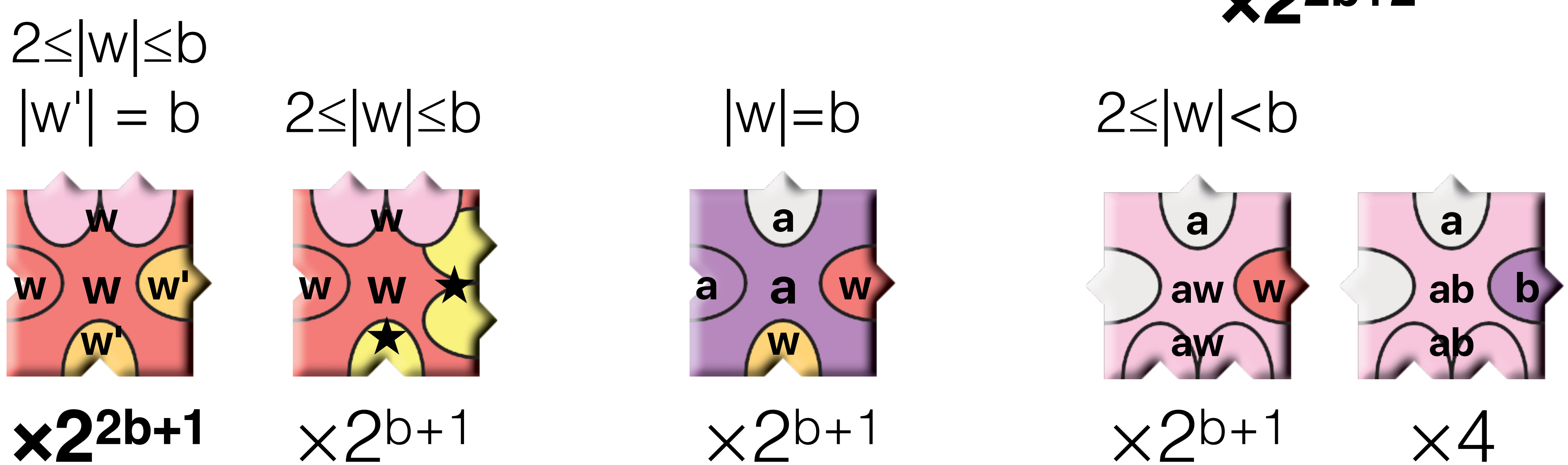


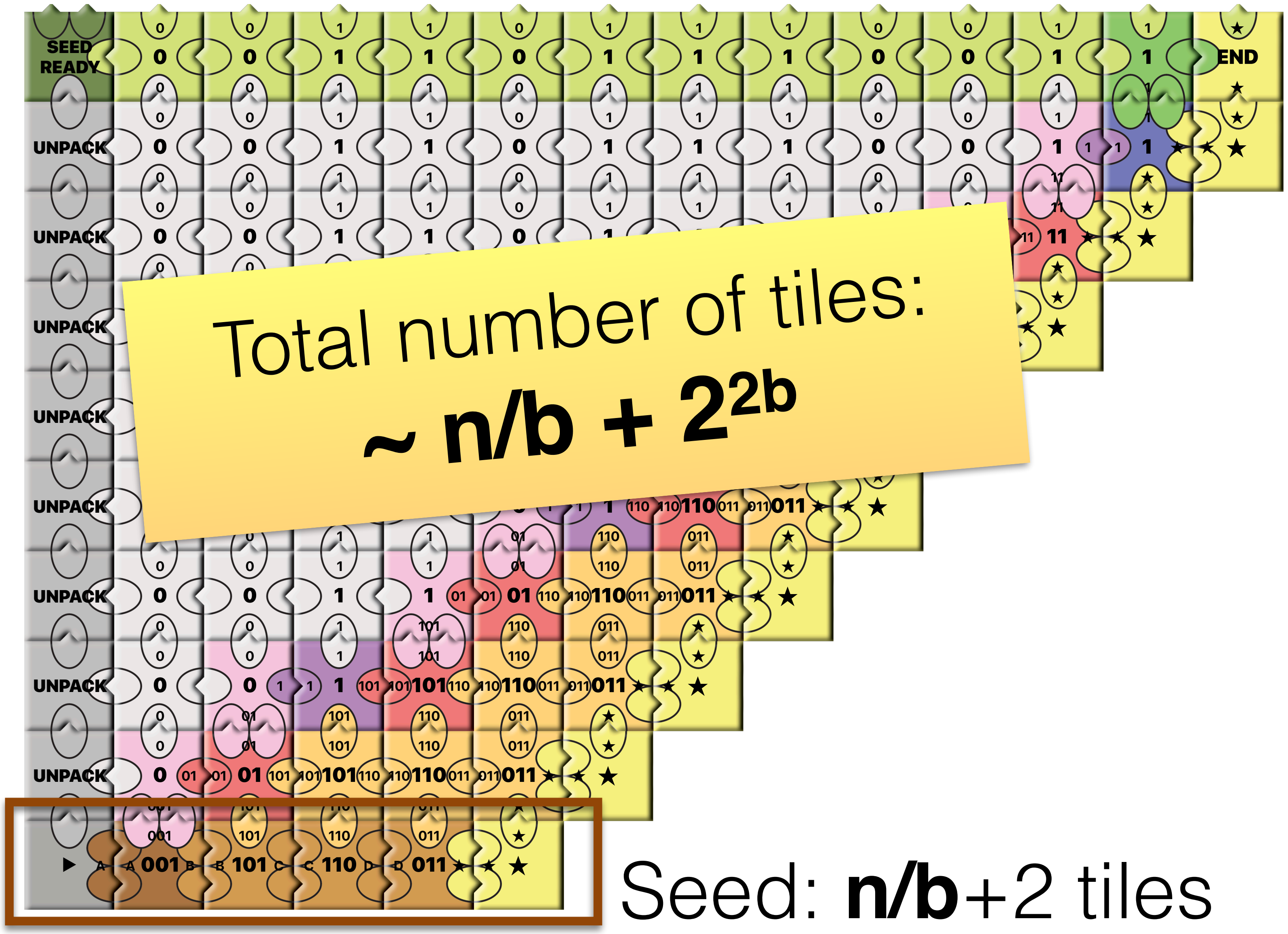
The tiles



Seed: **n/b**+2 tiles



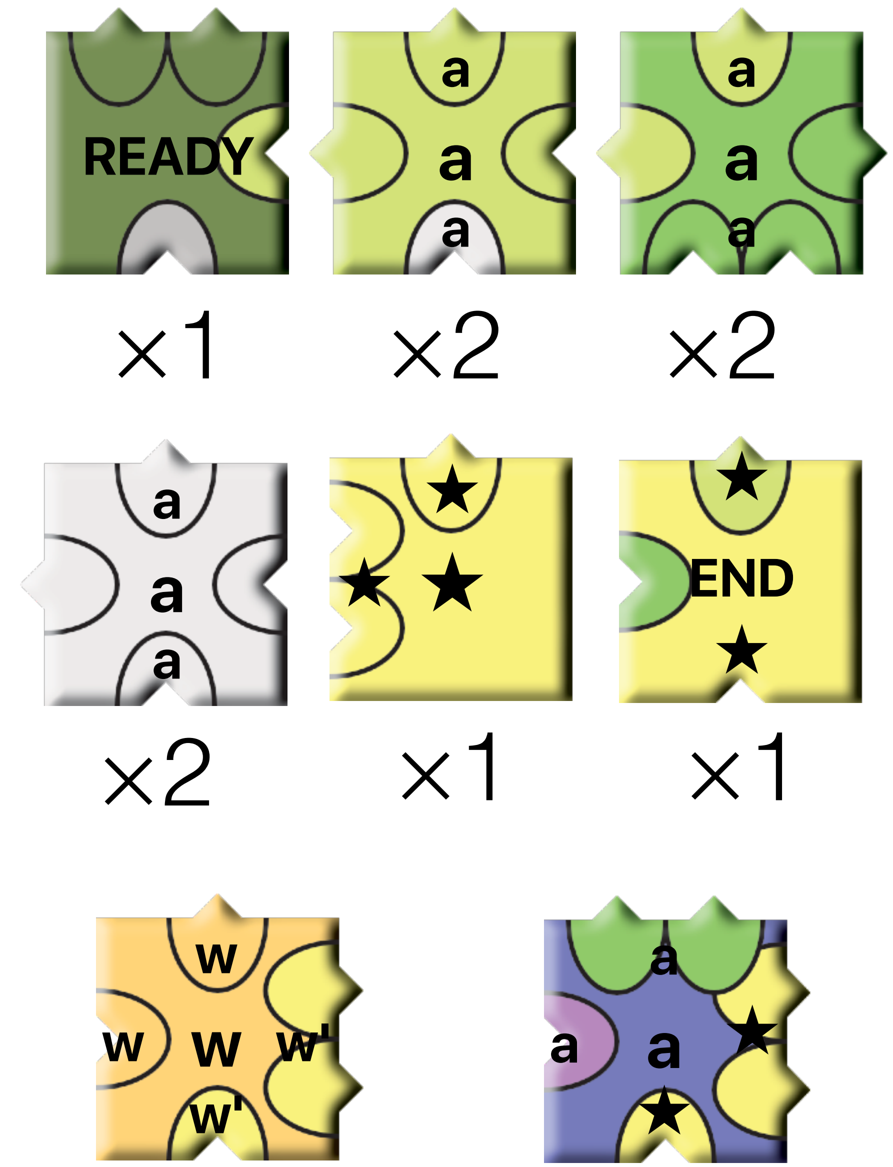




Total number of tiles:

$\sim n/b + 2^{2b}$

Seed: **n/b**+2 tiles

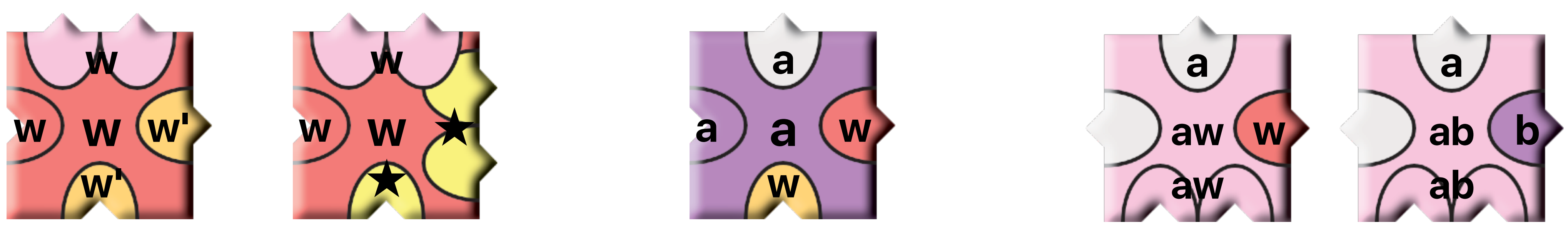


$\times 1$ $\times 2$ $\times 2$

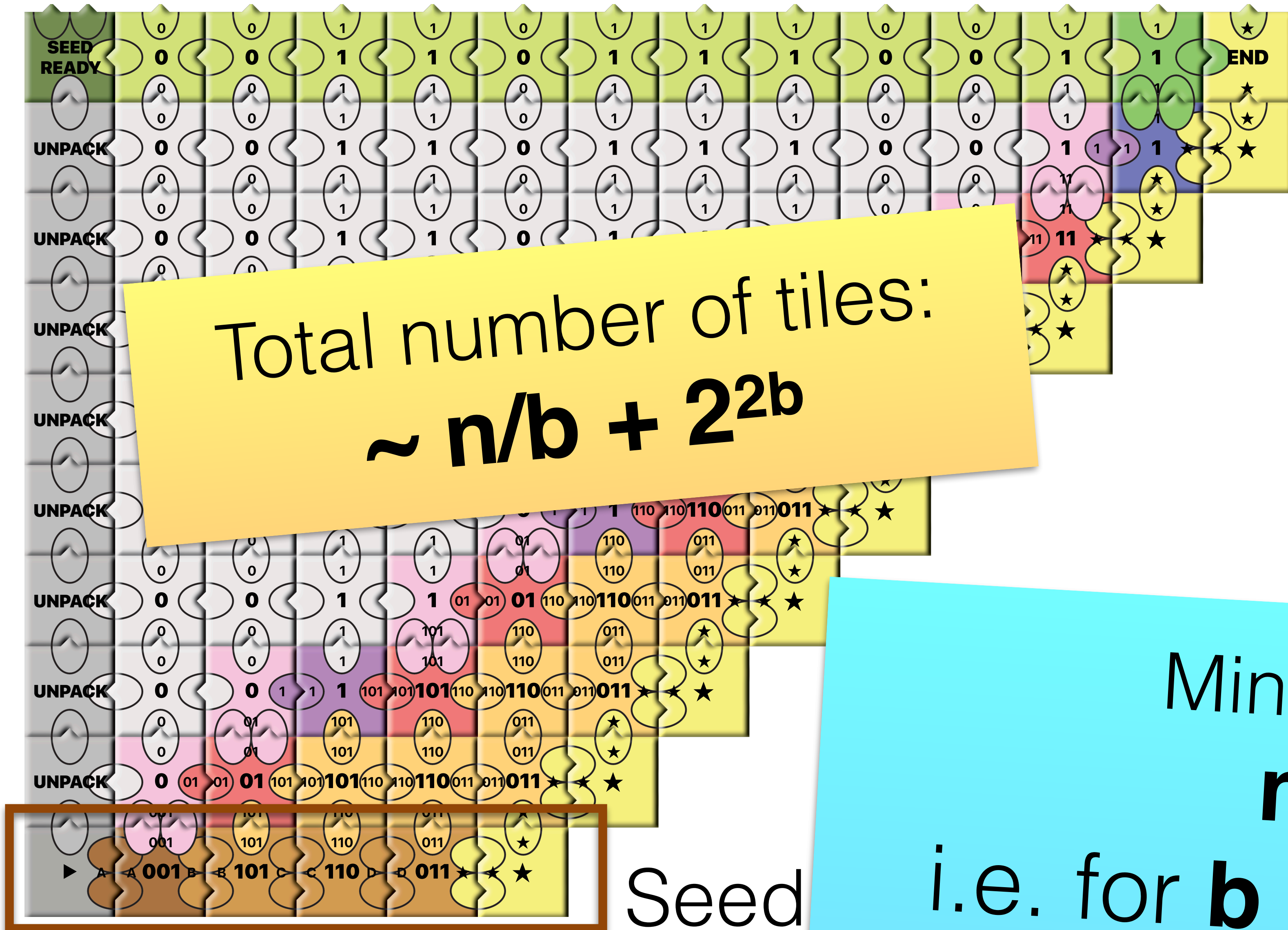
$\times 2$ $\times 1$ $\times 1$

$\mathbf{\times 2^{2b+2}}$ $\times 2$

$2 \leq |w| \leq b$, $|w'| = b$ | $2 \leq |w| \leq b$ | $|w| = b$ | $2 \leq |w| < b$



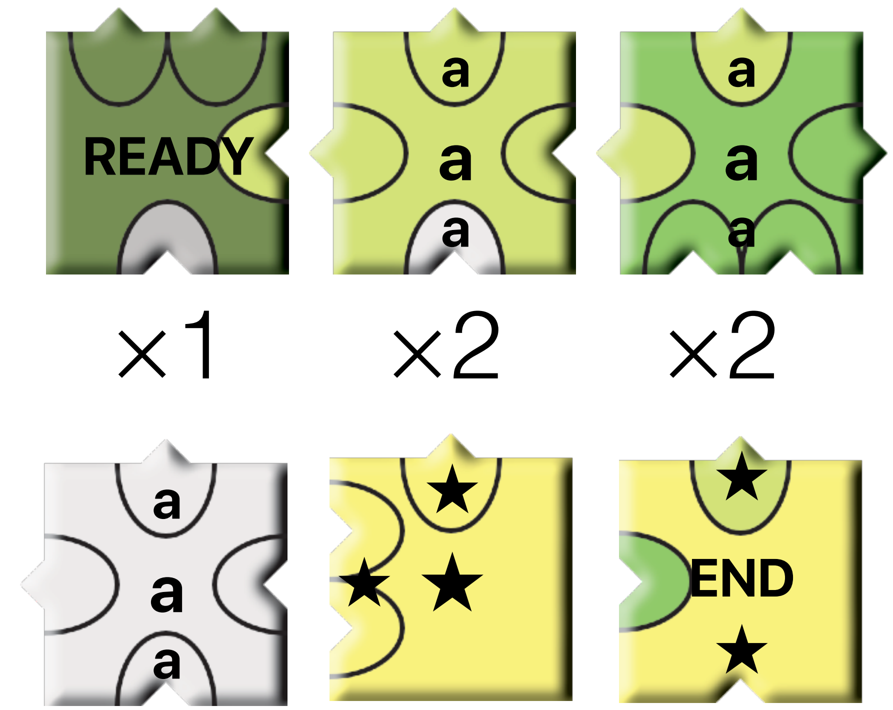
$\mathbf{\times 2^{2b+1}}$ $\times 2^{b+1}$ $\times 2^{b+1}$ $\times 2^{b+1}$ $\times 4$



Total number of tiles:

$\sim n/b + 2^{2b}$

Seed



$\times 1$ $\times 2$ $\times 2$

$\times 2$ $\times 1$ $\times 1$

Minimized when

$n/b = 2^{2b}$

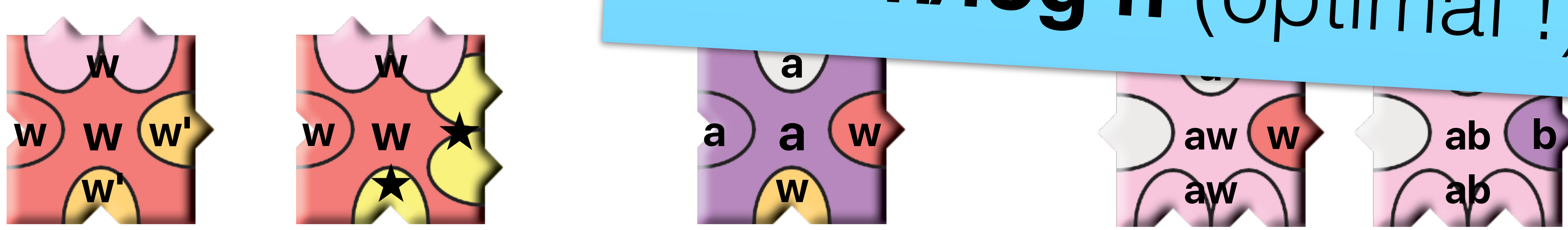
i.e. for **$b = \log(n/\log n)/2$**

The total number of tiles is:

$\sim n/\log n$ (optimal !)

$2 \leq |w| \leq b$
$|w'| = b$

$2 \leq |w| \leq b$

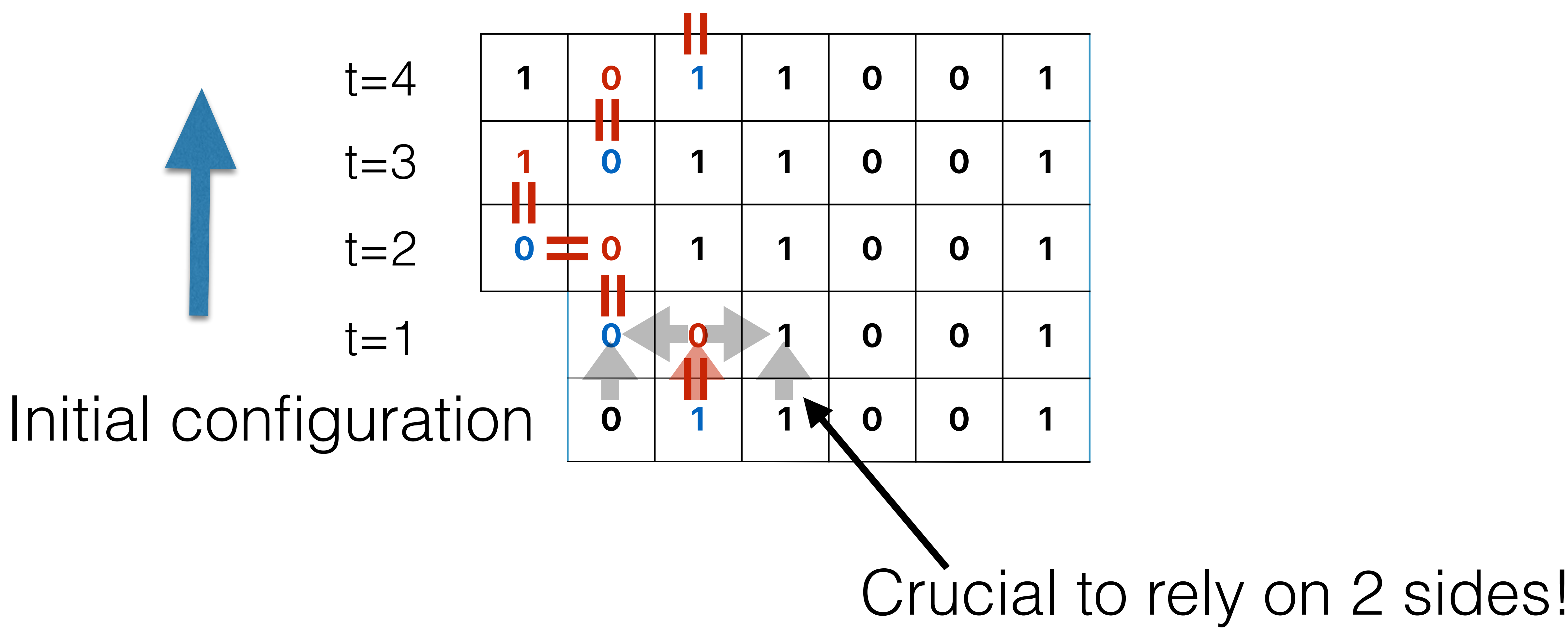


$\times 2^{2b+1}$ $\times 2^{b+1}$ $\times 2^{b+1}$ $\times 2^{b+1}$ $\times 4$

What about $T° = 1$?

Does algorithmic self-assembly simulate Turing machine at **T°1** in 2D?

- How to read and propagate the position of the head?



Does algorithmic self-assembly simulate Turing machine at **T°1** in 2D?

- **Theorem.** [Meunier, Regnault, 2015]
  Any deterministic T°1 tile set can be pumped outside a fixed radius in 2D

- **Corollary.** No T°1 tilesystem is Turing complete in 2D

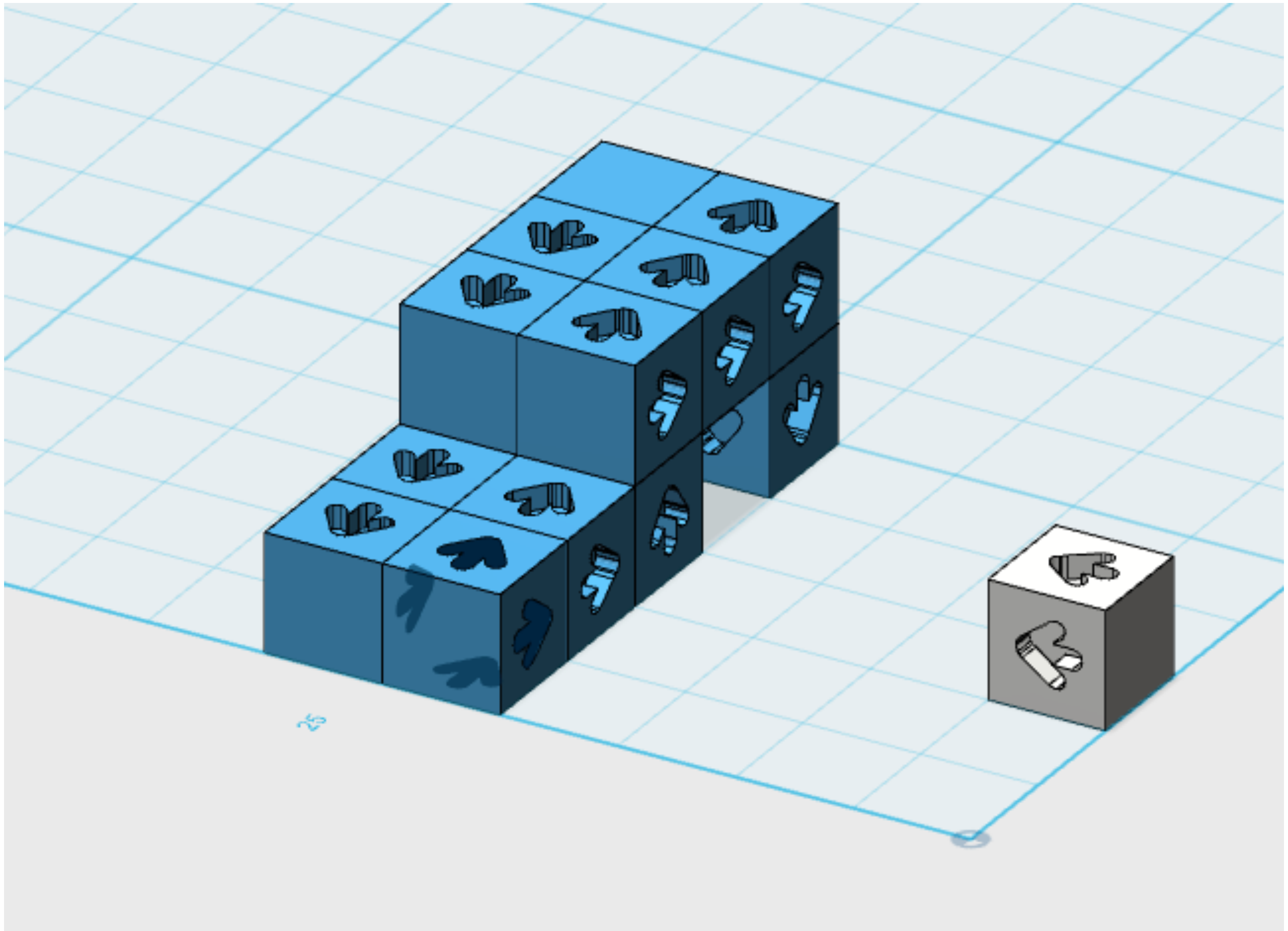
Algorithmic self-assembly simulates any Turing machine at T°1 in 3D

- **Theorem.** *[Cook, Fu, Schweller, 2011]*
  There is a T°1 tile set that simulates any Turing machine with 2-layers in 3D.

- *Key beautiful idea:* Blocking probe crystal

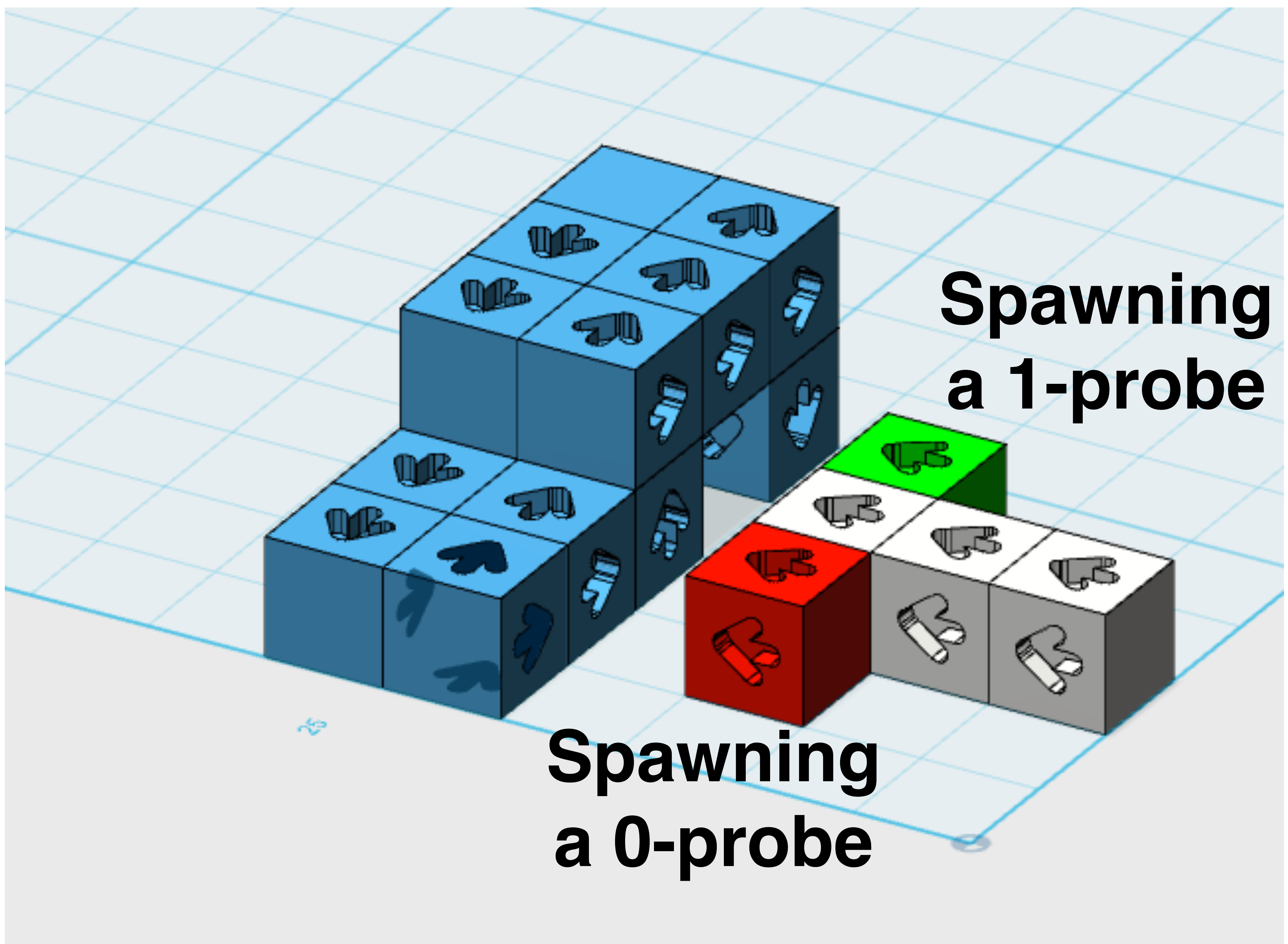
Algorithmic self-assembly simulates any Turing machine at T°1 in 3D

- *Key beautiful idea:* Blocking probe crystal



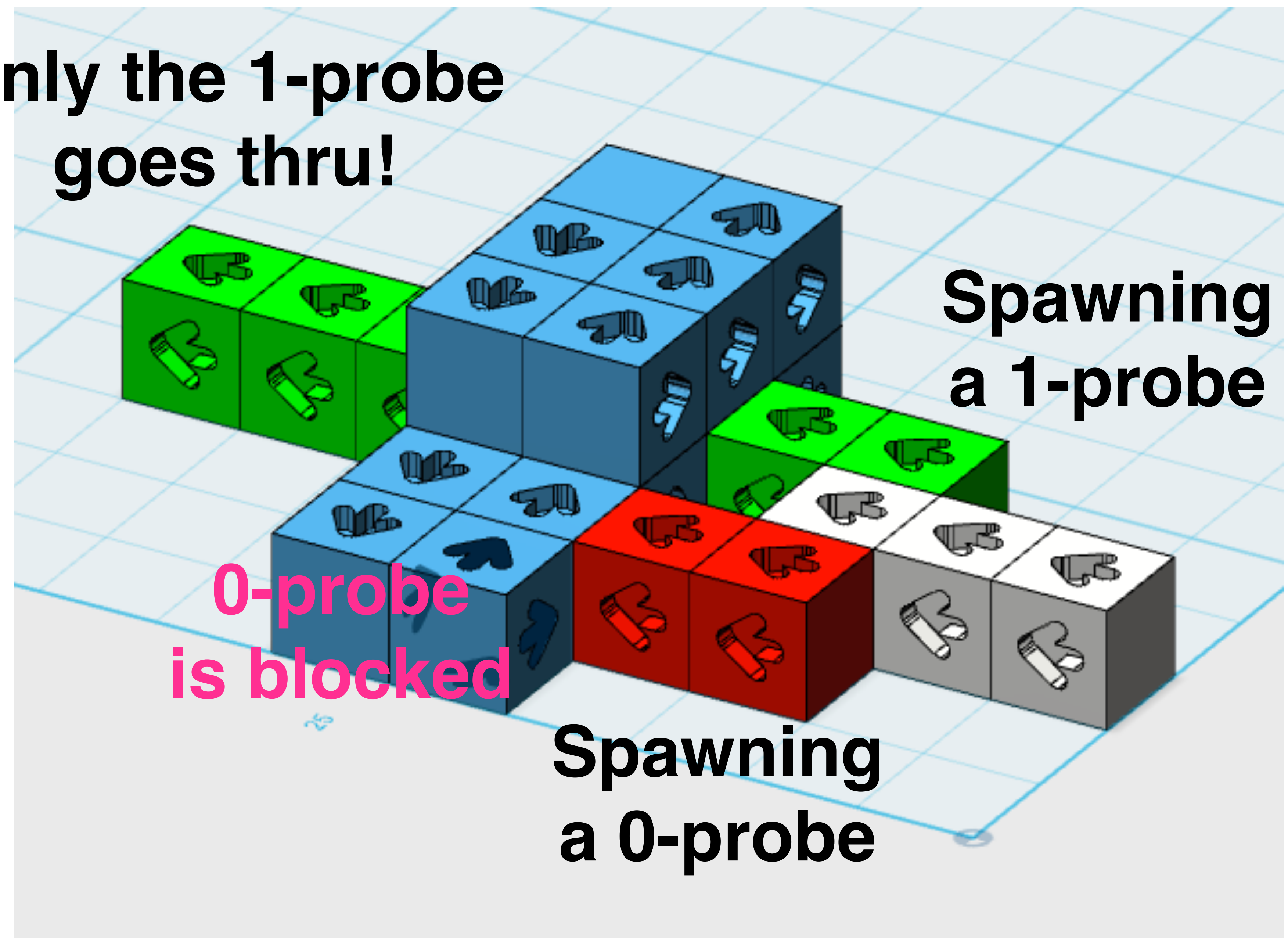
Algorithmic self-assembly simulates any Turing machine at T°1 in 3D

- *Key beautiful idea:* Blocking probe crystal



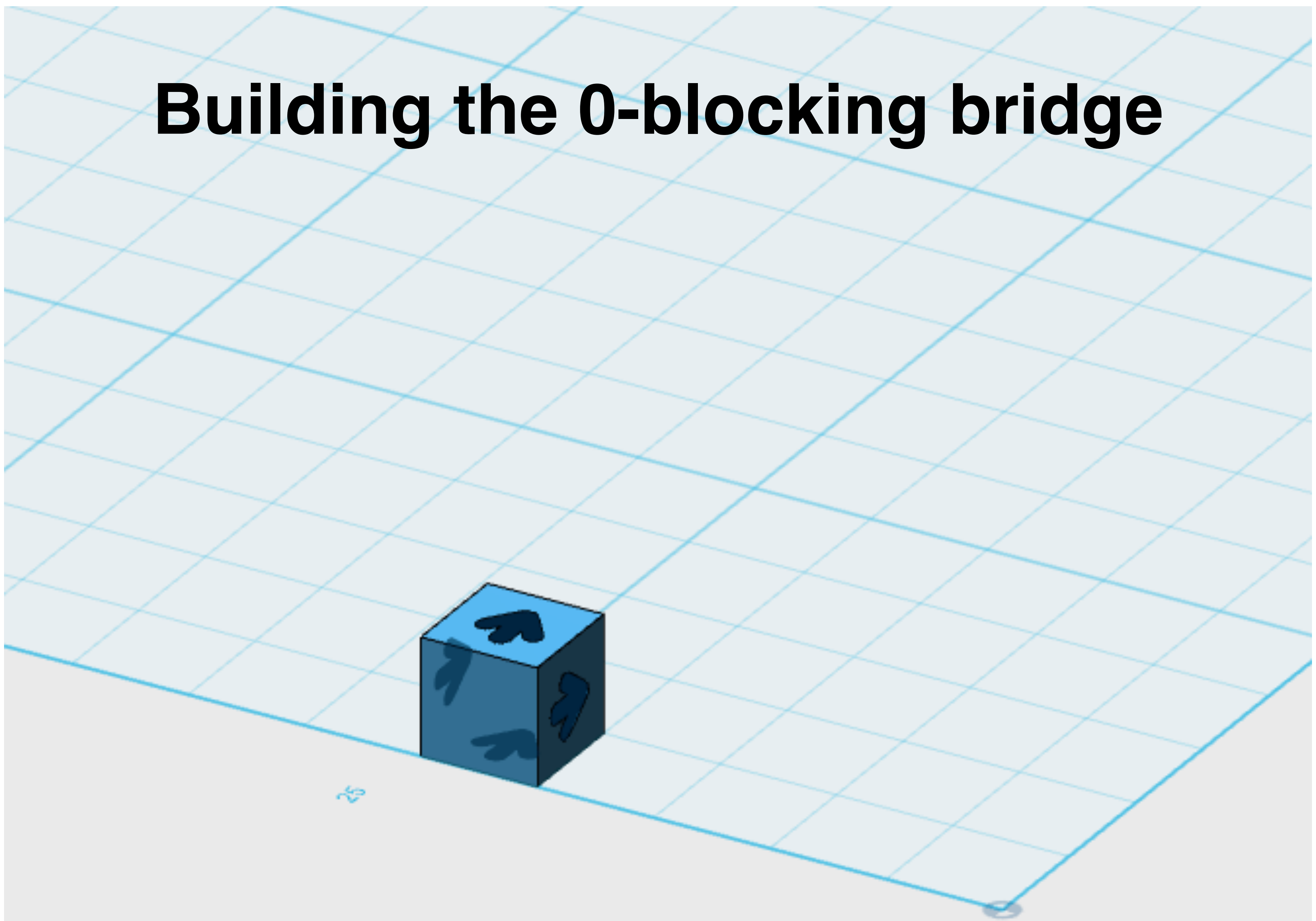
Algorithmic self-assembly simulates any Turing machine at T°1 in 3D

- *Key beautiful idea:* Blocking probe crystal



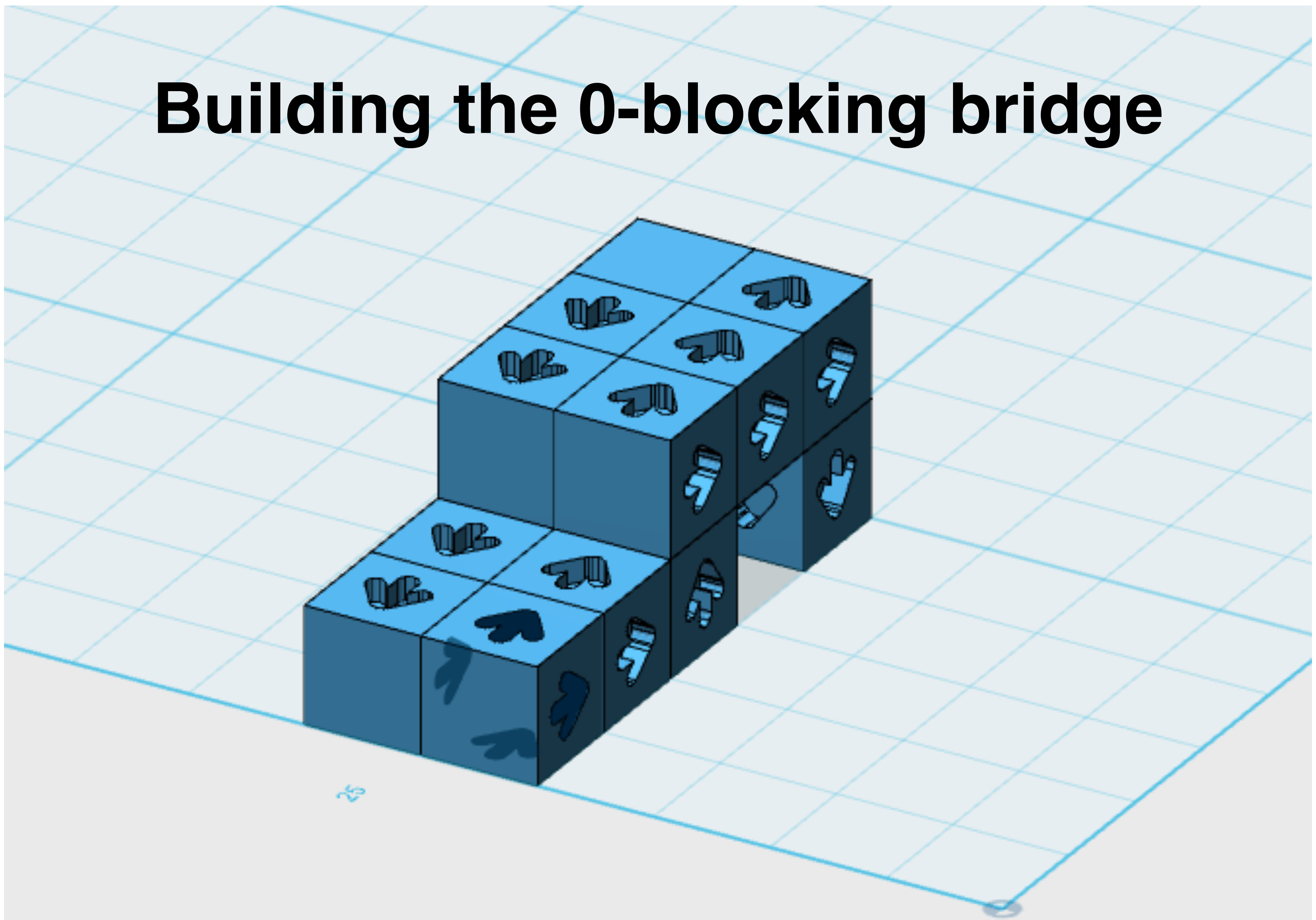
Algorithmic self-assembly simulates any Turing machine at T°1 in 3D

- *Key beautiful idea:* Blocking probe crystal



Algorithmic self-assembly simulates any Turing machine at T°1 in 3D

- *Key beautiful idea:* Blocking probe crystal



Algorithmic self-assembly simulates any Turing machine at T°1 in 3D

- *Key beautiful idea:* Blocking probe crystal

We can thus read and write 0 and 1 on the "next tape" !

Crystals are Turing complete !!!

An experimental realization of a universal computer

Conclusion

- A lot of new models
- A new kind of geometric algorithmic
- A lot of implication in biology and bio-engineering
- Lots of extensions: mixed dynamics, errors,…
  More on fixing errors at the last lecture