Models of Distributed Computing in Molecular Programming

Nicholas Schiefer

This is a survey, with some informal statements.

Stop me at any time.

Prelude

Perspective on molecular programming

Molecular programming is…

computing with molecules

what do we mean by “computing”?

what kinds of problems can we solve?

what are “molecules”?

how do they interact with each other?

Three perspectives

Natural scientist

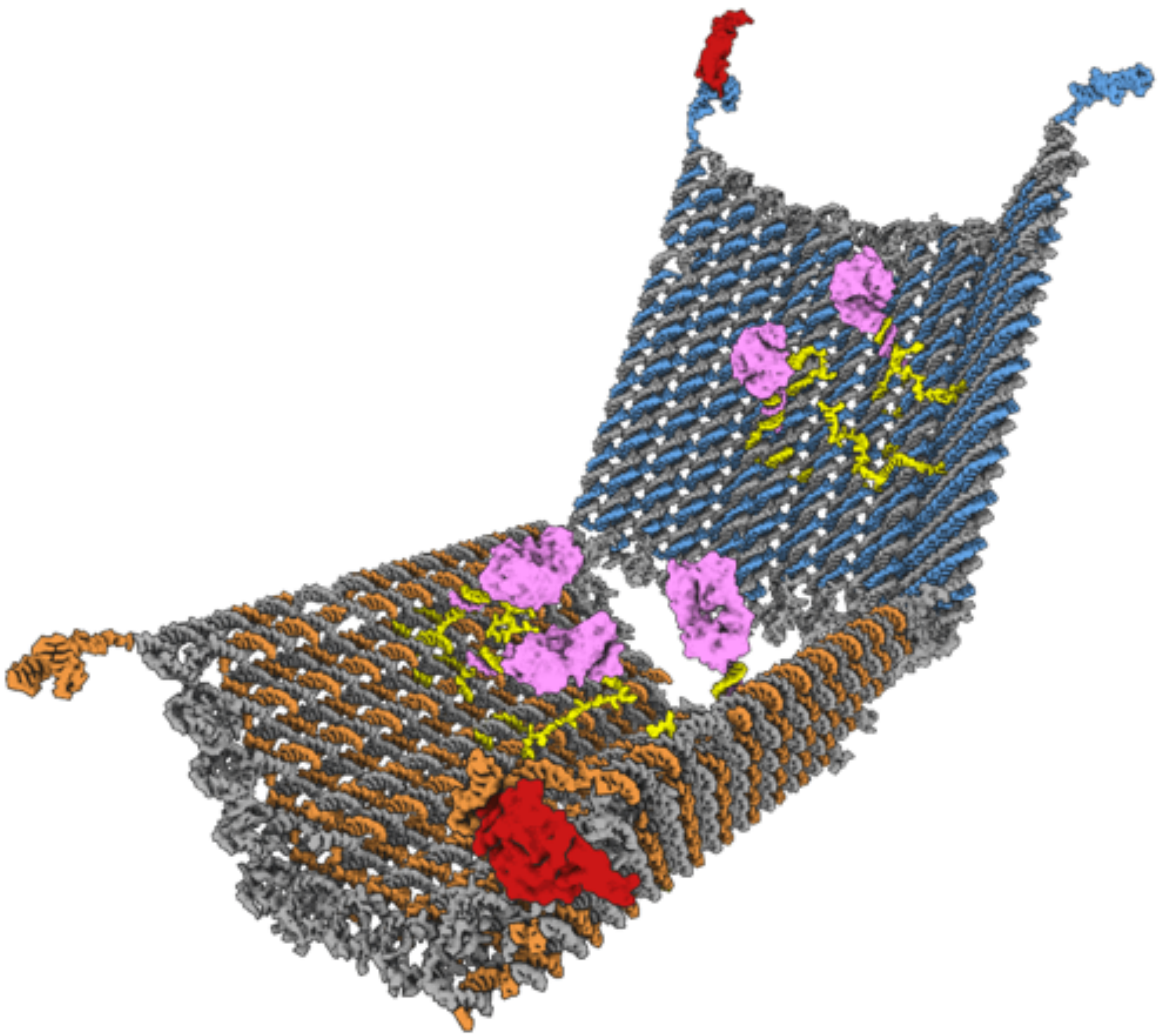
How does computation occur in the physical/biological world?

[Wikimedia]

Three perspectives

Engineer

How can use computational power in nanotechnology?

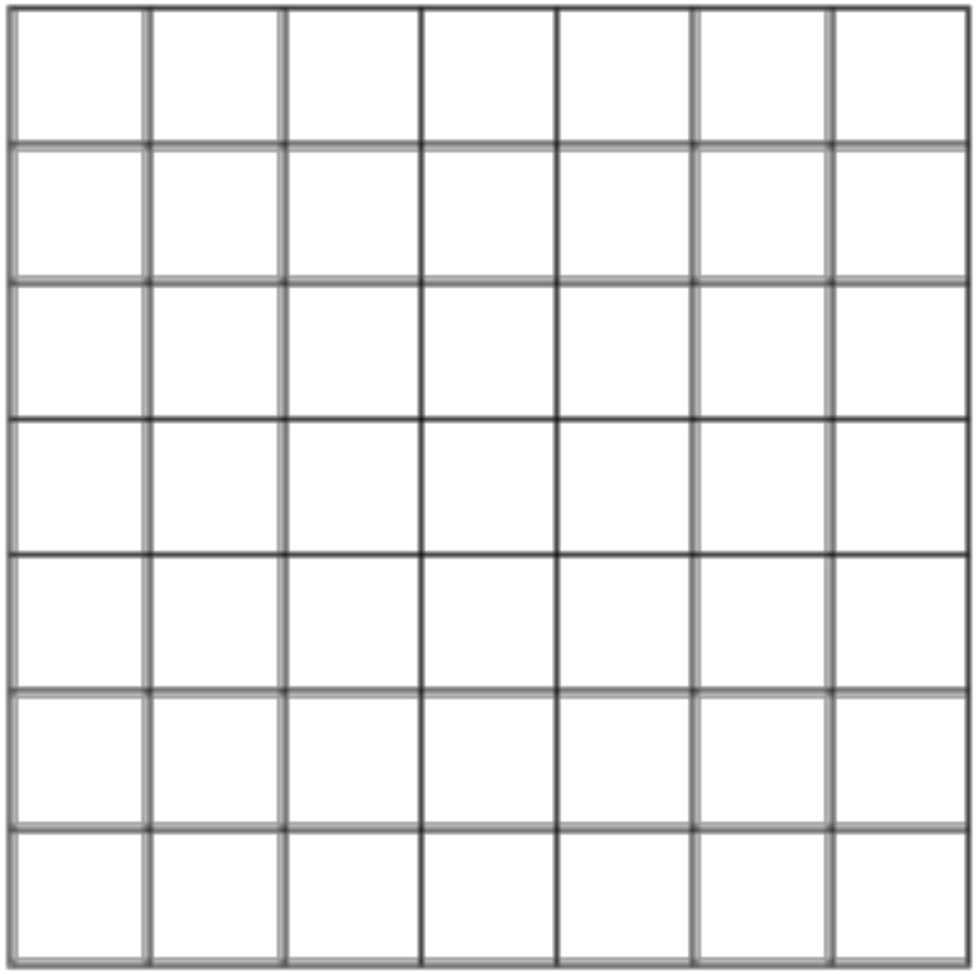


[Douglas/Wyss Institute]

Three perspectives

Computer scientist

How can I compute using extremely simple systems?



Three perspectives

Natural scientist

How does computation occur in the physical/biological world?

Engineer

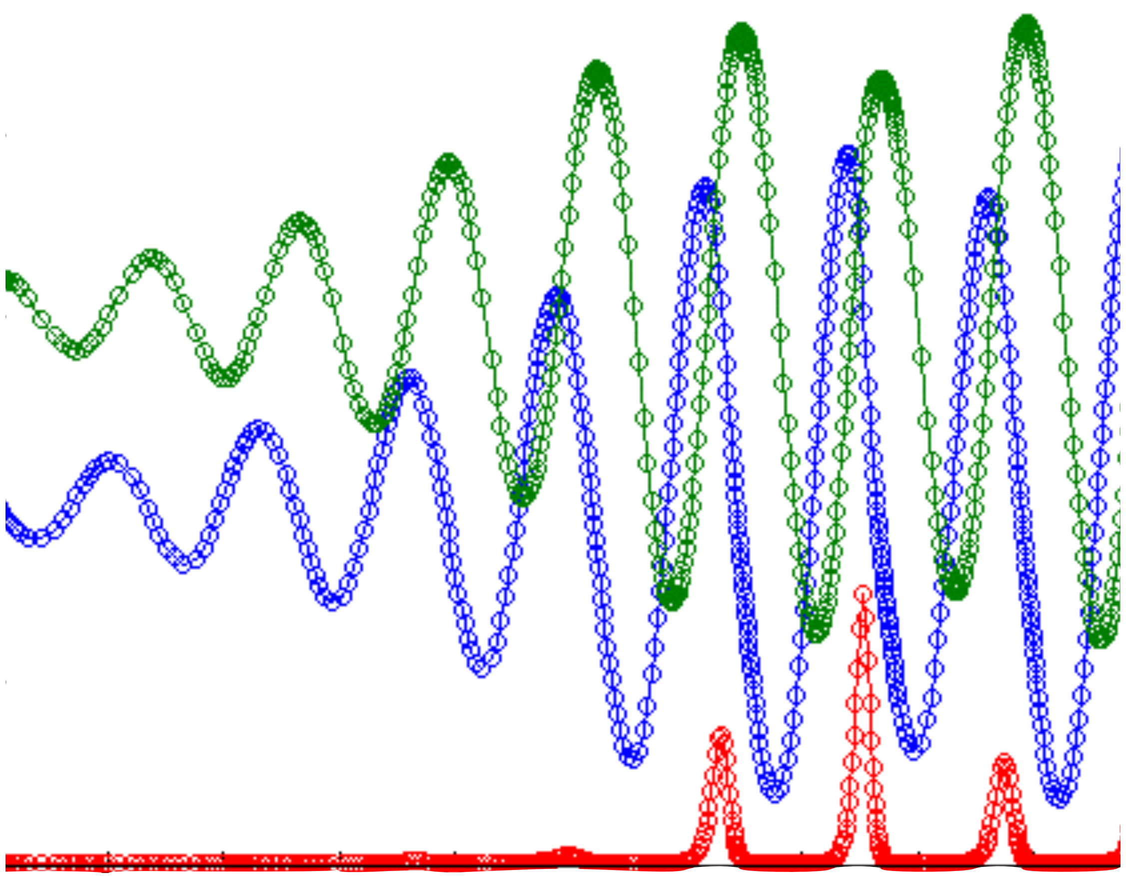
How can use computational power in nanotechnology?

Computer scientist

How can I compute using extremely simple systems?

Two things biology does…

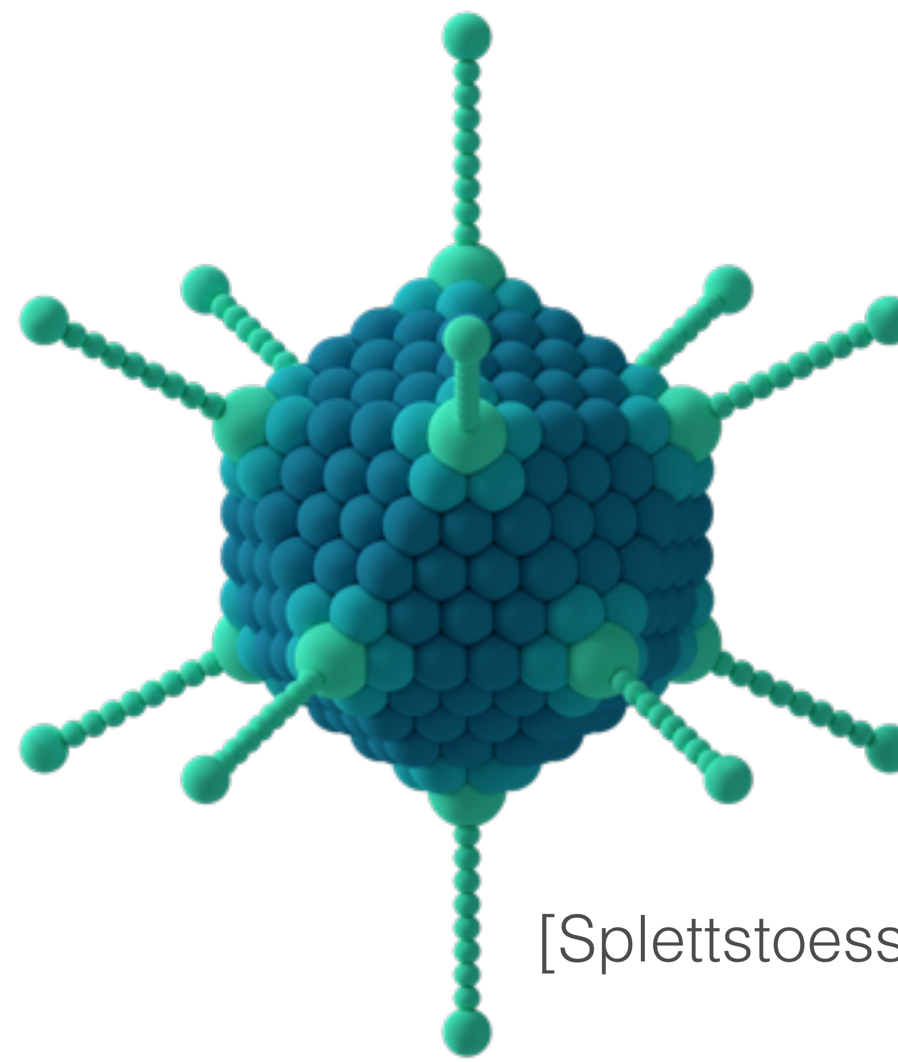
Control concentrations



Dynamical molecular programming

Chemical Reaction Networks

Build things



[Splettstoesser]

Structural molecular programming

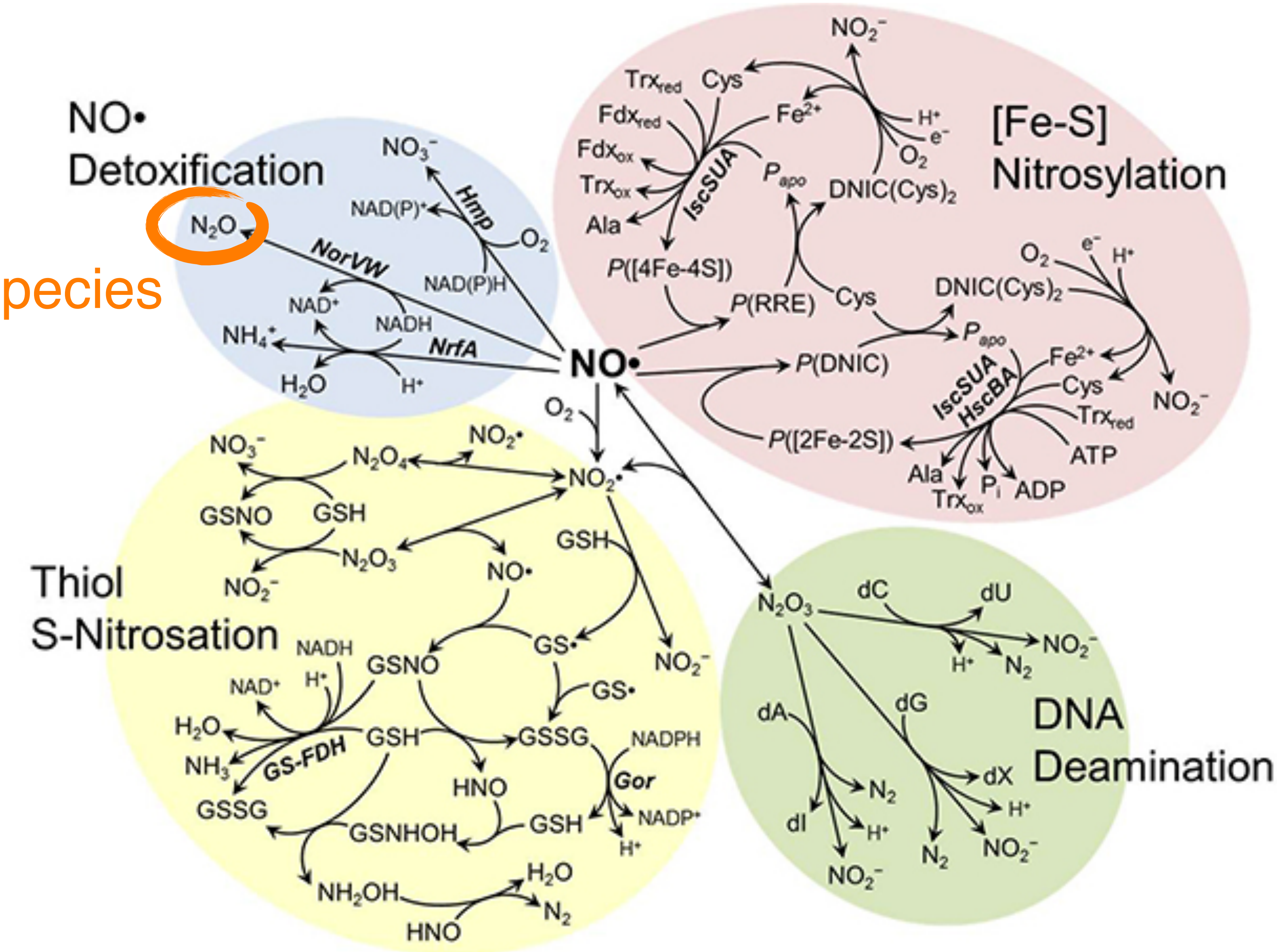
Tile Assembly Model

Part I

Chemical Reaction Networks

Chemical Reaction Networks

chemical species



[Brynildsen]

Chemical Reaction Networks

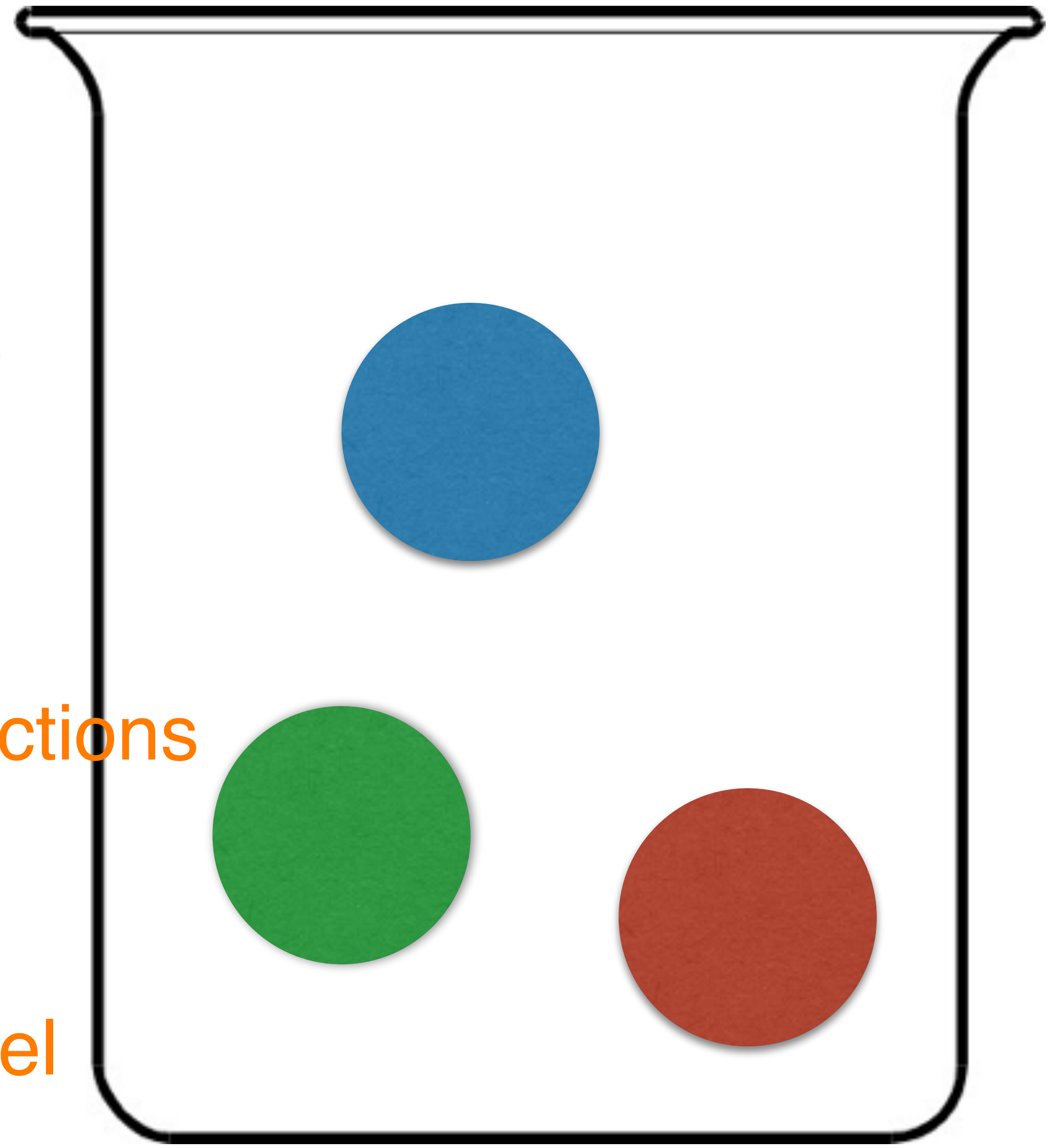
$$A, B, C, W, X, Y, Z,$$

Finite* set of species

$$A + B \to C$$
$$X + Y \to Z + W$$

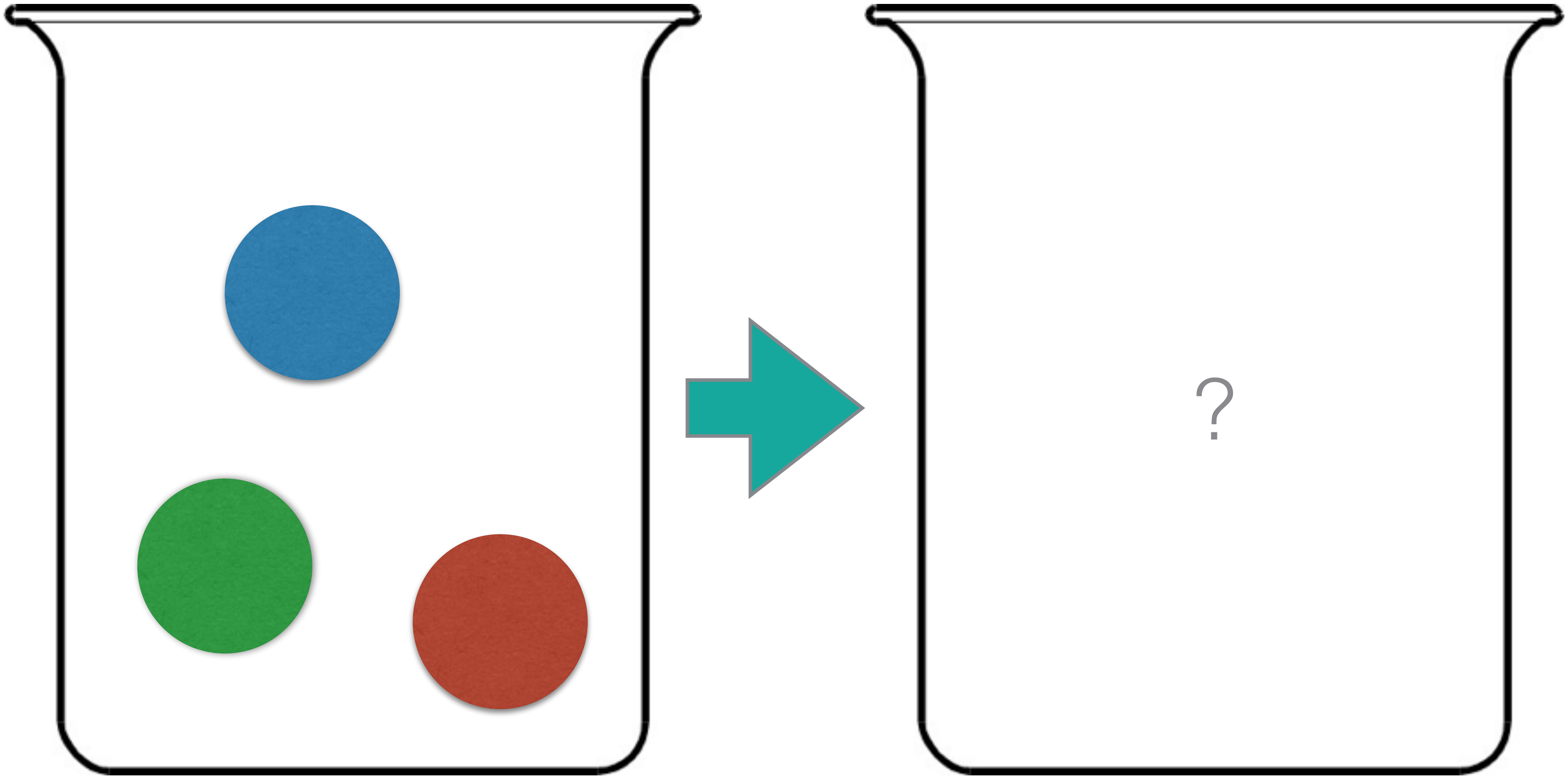
Finite* set of reactions

Well-mixed reaction vessel



*usually

CRN Dynamics



time t

time (t + ε)

CRN Dynamics

$$A + B \xrightarrow{k} C$$

"Mass action" regime

Concentrations

$[A], [B], [C] \in \mathbb{R}^+$

Described by ODE

$$\frac{d[A]}{dt} = -k[A][B]$$
$$\frac{d[B]}{dt} = -k[A][B]$$
$$\frac{d[C]}{dt} = k[A][B]$$

"Kinetic" regime

Counts

$\#(A), \#(B), \#(C) \in \mathbb{Z}^+$

State is a continuous time Markov chain ("Gillespie dynamics")

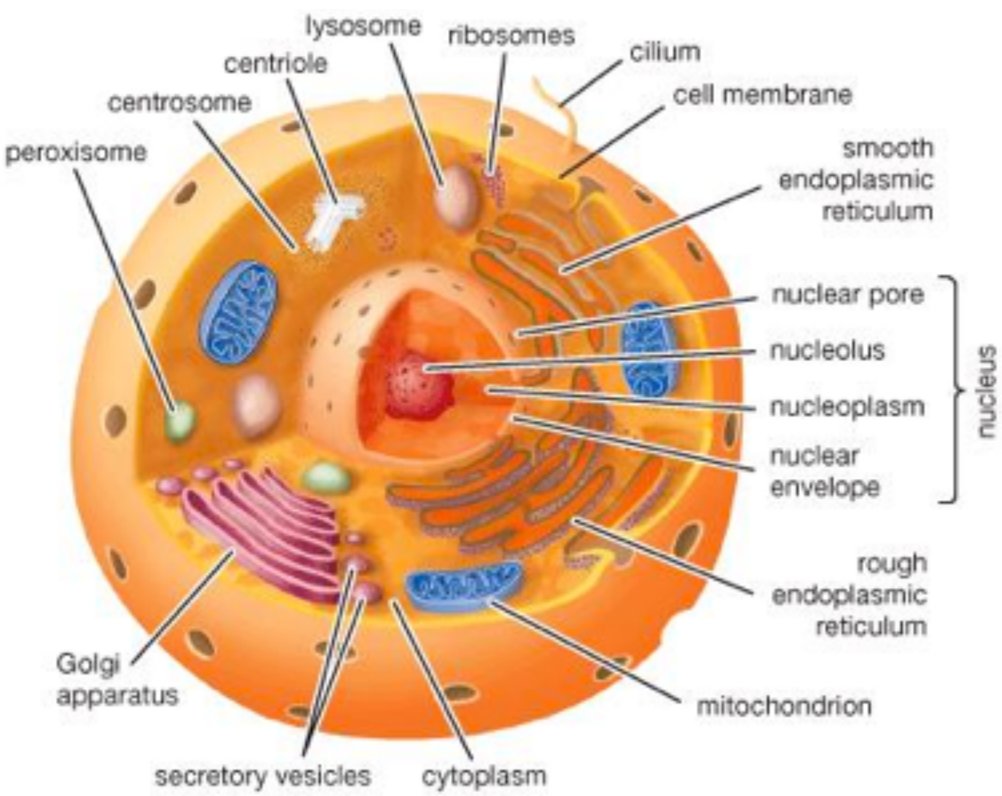
Equivalent to population protocols

CRN Dynamics

“Mass action” regime

“Kinetic” regime

vs.



Kinetic model approaches mass action model as counts grow towards infinity.

[Britannica]

Mass action CRNs

Theorem. Any polynomial ODE can be approximated arbitrarily well by a mass action CRN.

Key ideas

Proof due to [Berleant 2014], much older proof via Michaelis-Menten enzyme kinetics

For each variable x, have species X^+ and X^- that "cancel quickly" $X^+ + X^- \xrightarrow{\text{fast}} \emptyset$

Linear terms are easy

Mass action CRNs

Theorem. Any polynomial ODE can be approximated arbitrarily well by a mass action CRN.

Key ideas

To implement quadratic terms, use multiple reactants:

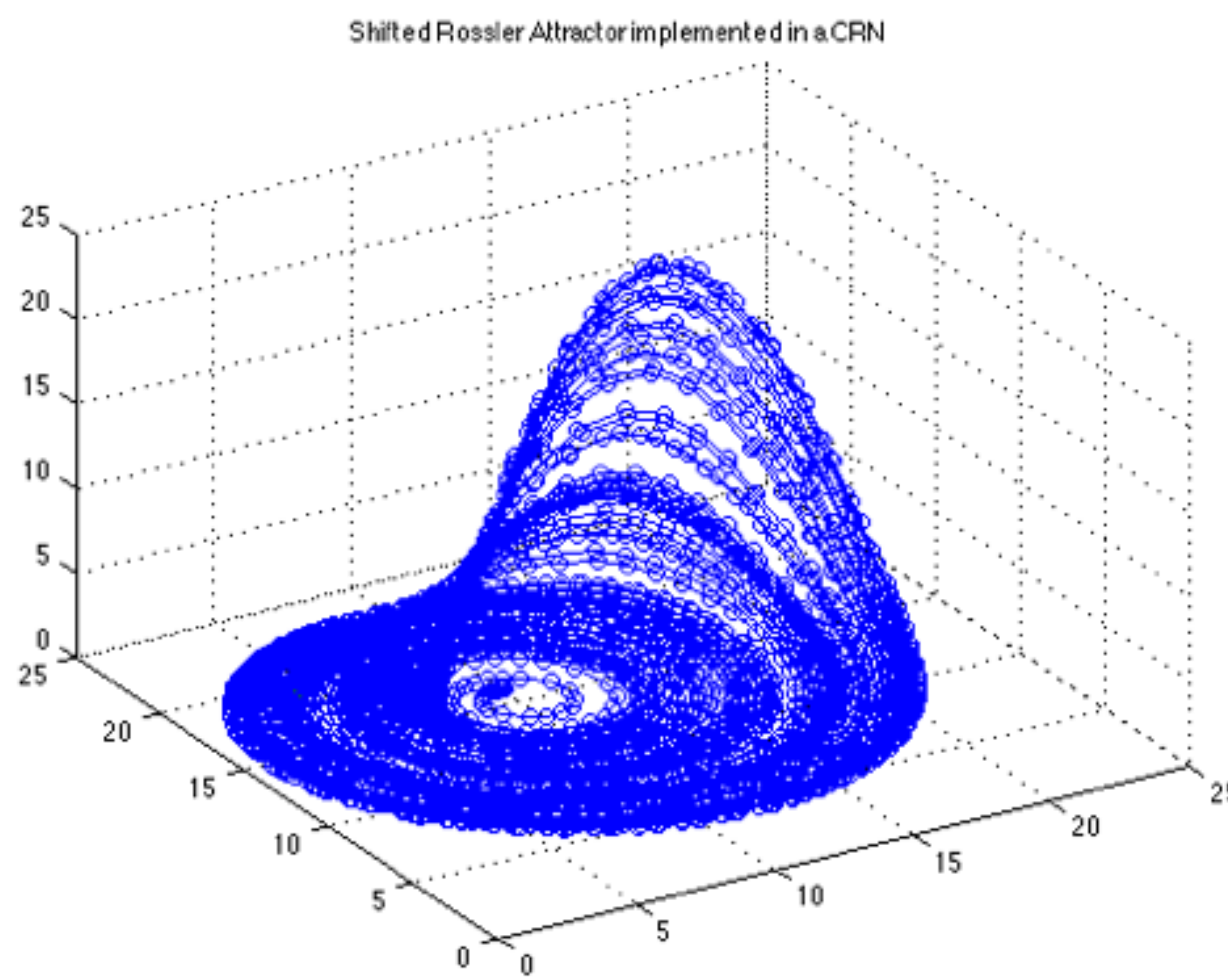
$$X + X \to Y \qquad \frac{d[Y]}{dt} = [X]^2 \qquad \frac{d[X]}{dt} = -[X]^2$$

Use special reactants to break higher degree terms into several parts

Mass action CRNs

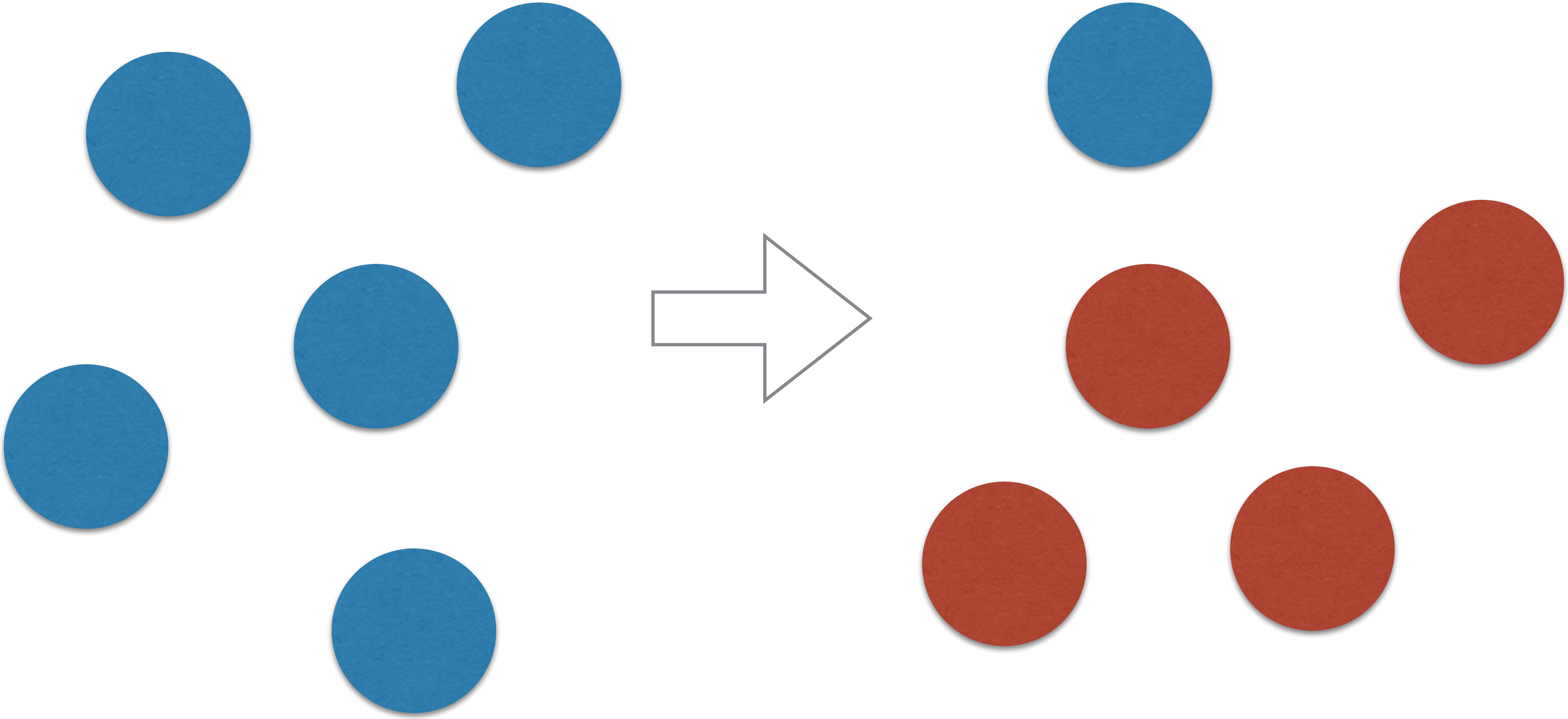
Theorem. Any polynomial ODE can be approximated arbitrarily well by a mass action CRN.

$$\begin{aligned}\frac{dx}{dt} &= -z - y\\ \frac{dy}{dt} &= x + ay\\ \frac{dz}{dt} &= b + z(x - c)\end{aligned}$$



Kinetic CRNs

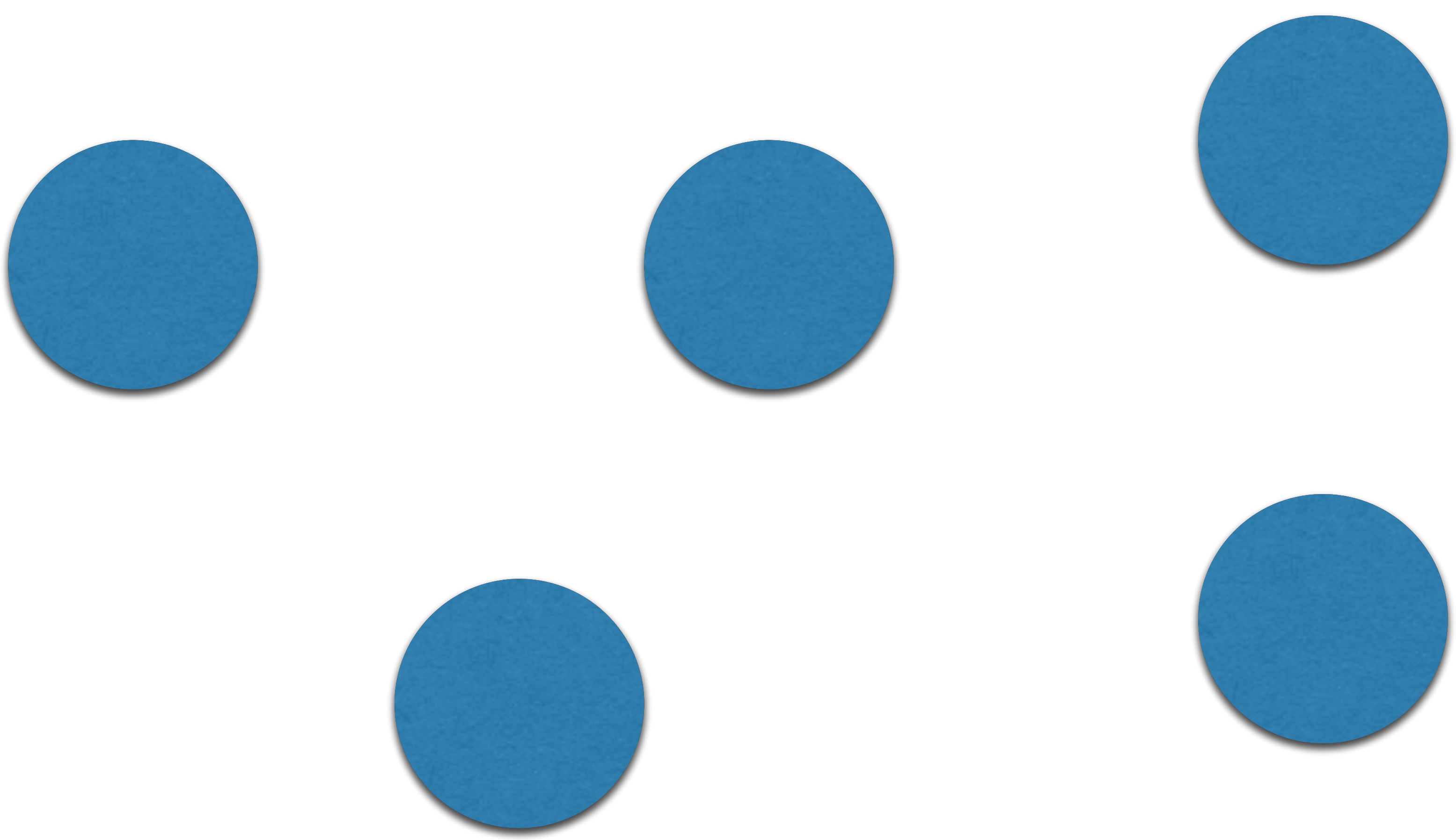
Stable leader election



Kinetic CRNs

Stable leader election

$A + A \to A + B$ solves leader election in $\mathcal{O}(n)$ expected time (equivalently $\mathcal{O}(n^2)$ pairwise interactions)



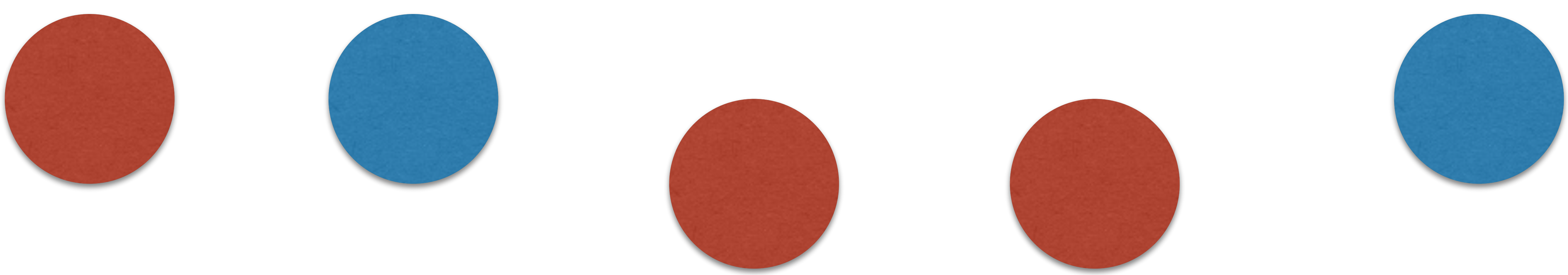
Kinetic CRNs

Stable leader election

Theorem. Stable leader election requires $\Omega(n^2)$ pairwise interactions.

[Doty and Soloveichik, 2015]

Intuition: No matter what, there is always a "bottleneck" somewhere, where two low concentration species need to find each other to know who will be the leader.

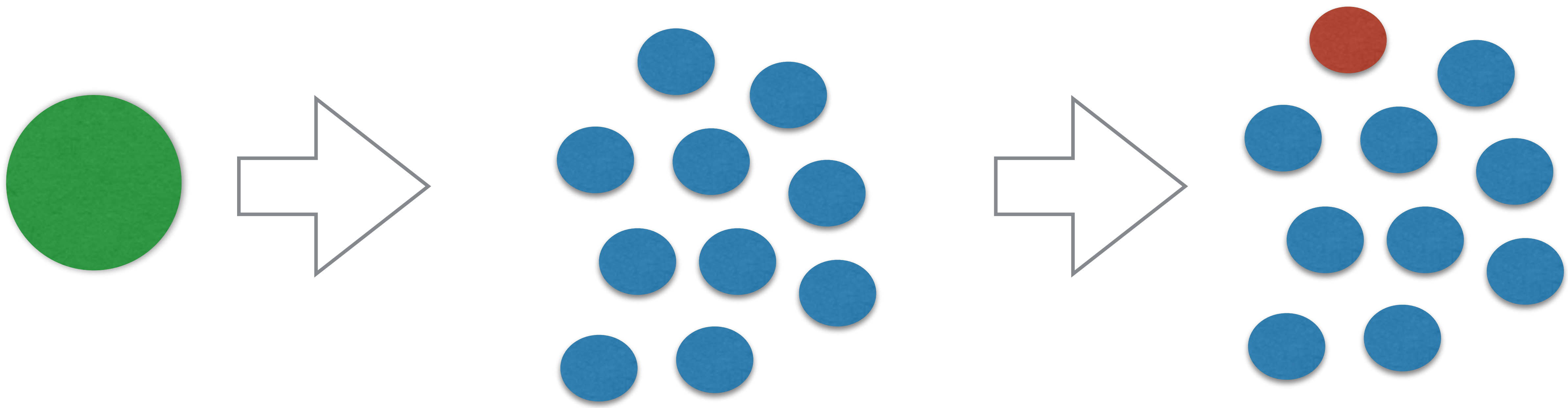


Extensions of kinetic CRNs

Stable leader election

Allowing a super-constant number of CRN species gives better protocols [Alistairh and Gelashvili, 2015], [Alistarh, Aspnes, Eisenstat, Gelashvili, Rivest, 2017], etc.

Can have substantial impact on *descriptive complexity* in some applications



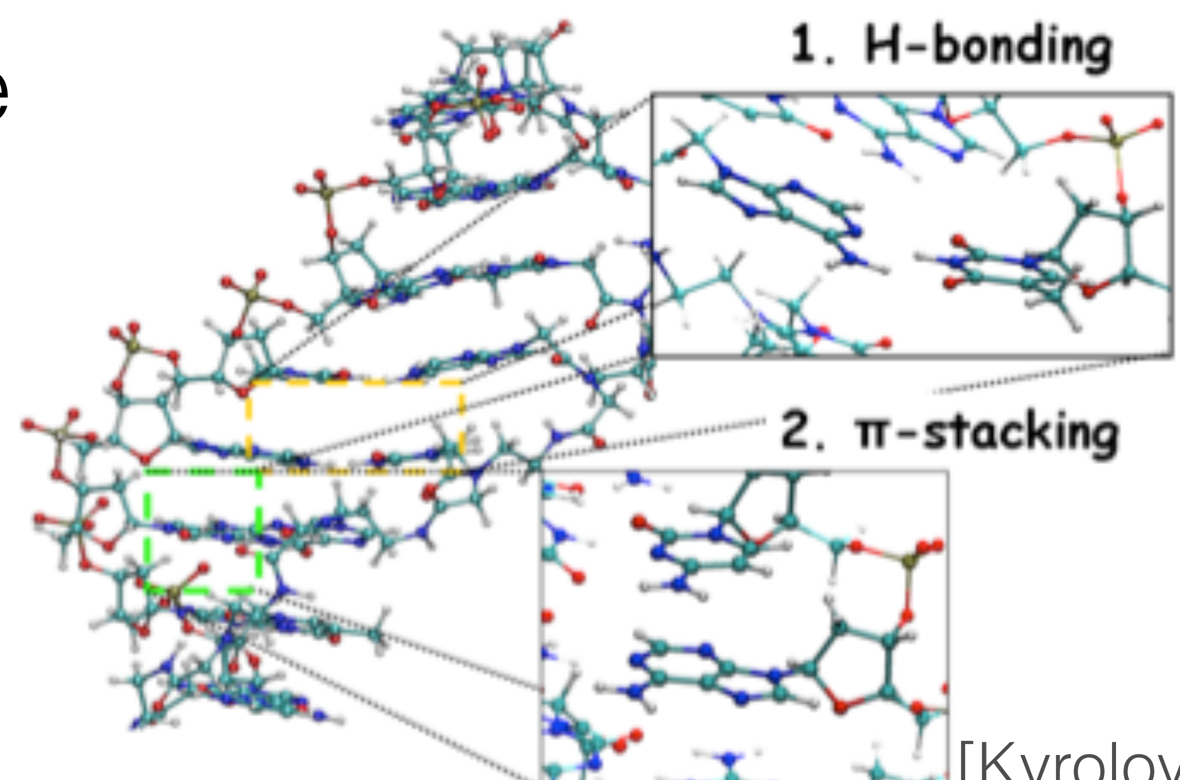
Implementing CRNs

DNA as a substrate for molecular engineering

Very predictable secondary structure due to Watson-Crick complementarity

Easy to filter out garbage because of pi-stacking

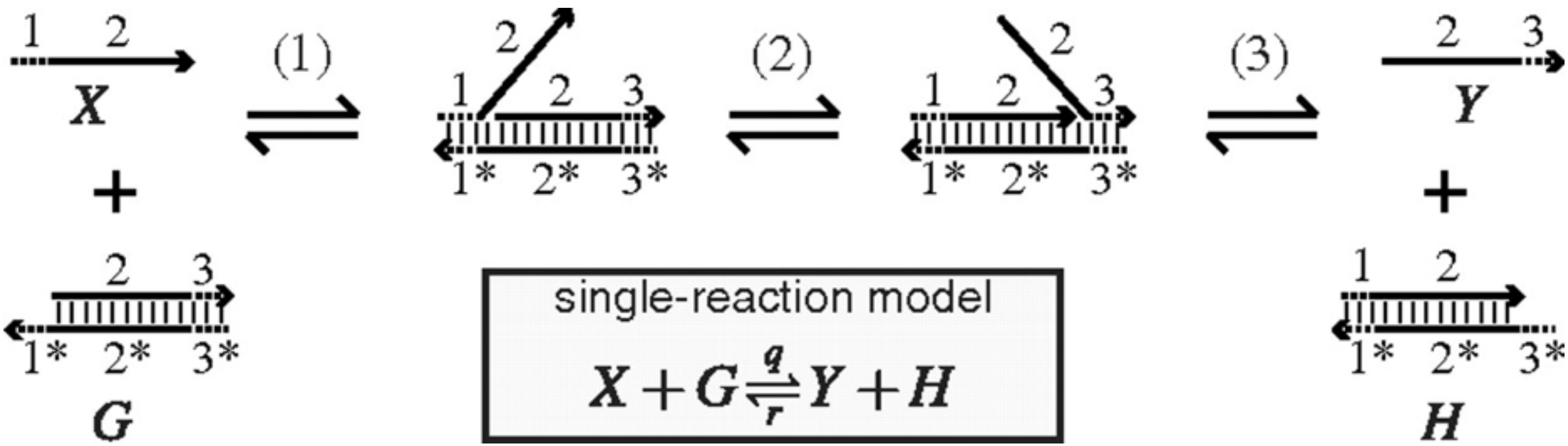
Cheap and easy to synthesize



[Kyrolov]

Implementing CRNs

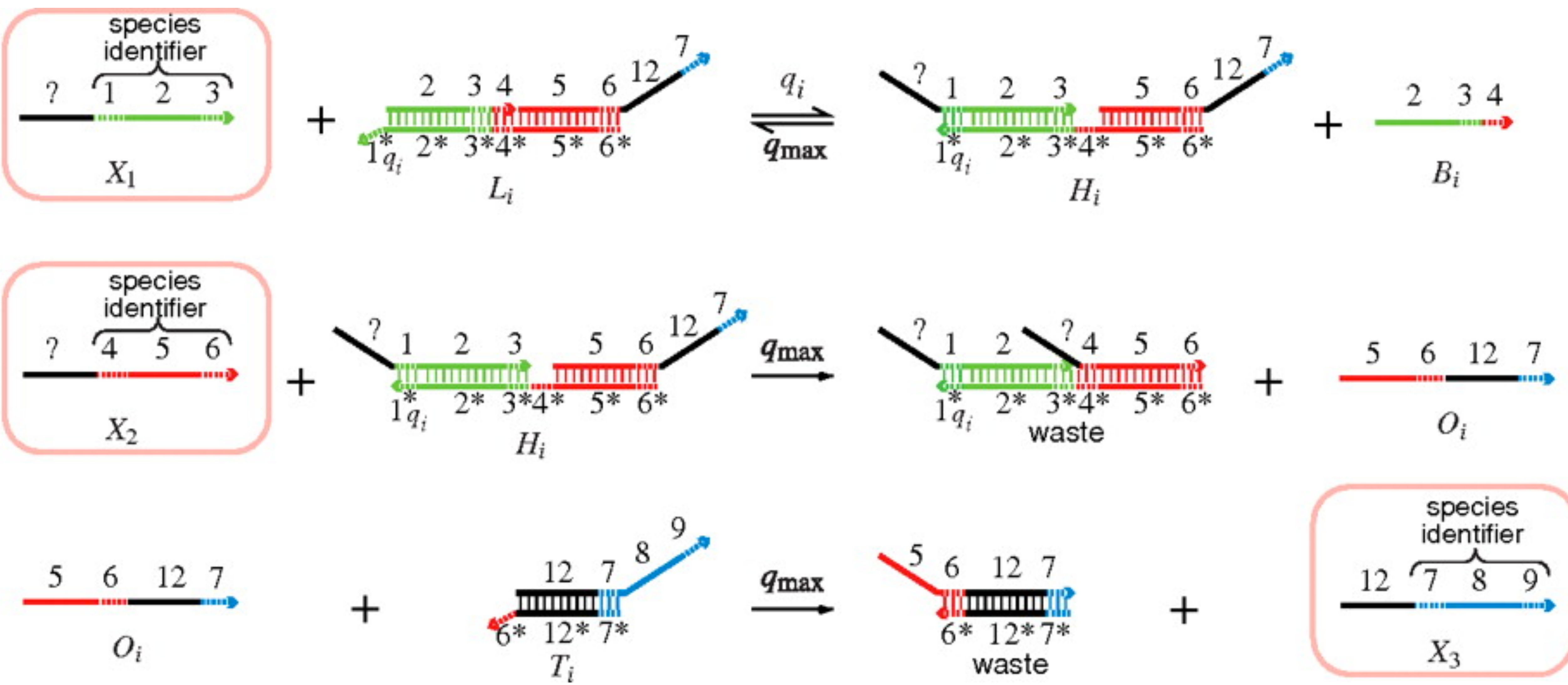
Toehold-mediated branch migration



[Soloveichik, Seelig, and Winfree , 2010]

Implementing CRNs

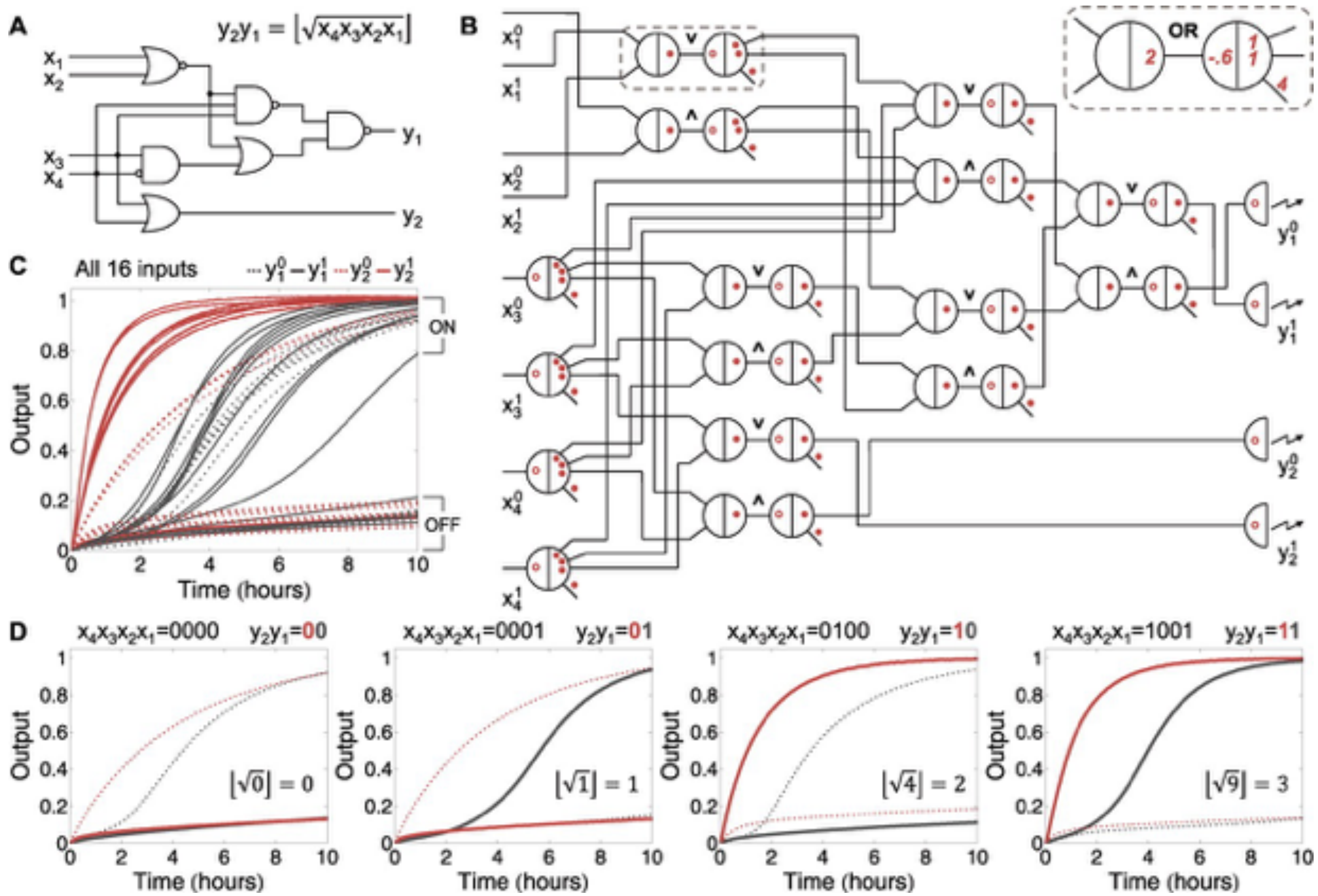
DNA strand displacement circuits



[Soloveichik, Seelig, and Winfree , 2010]

Implementing CRNs

Difficulty of design is related to # of species!

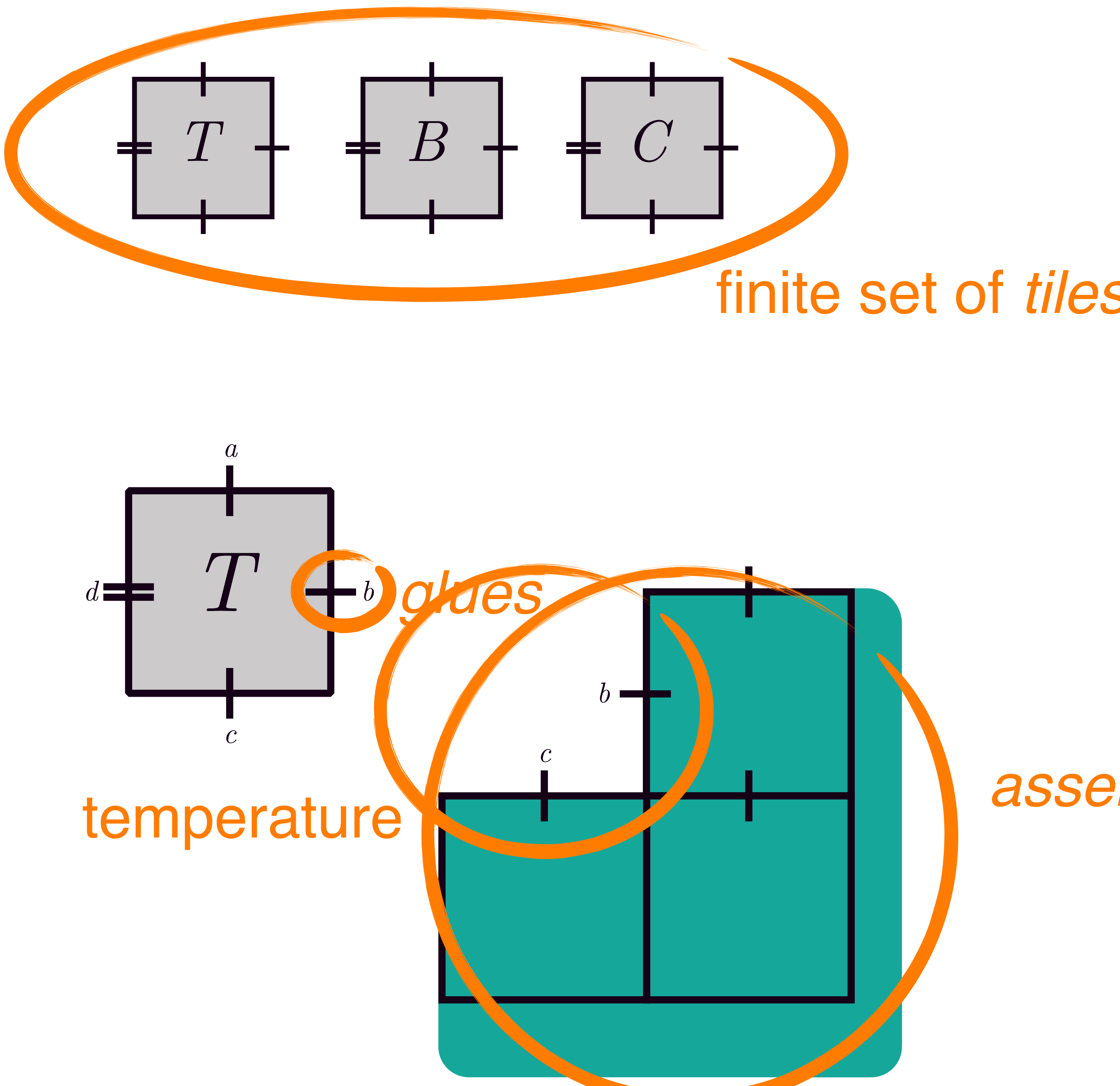


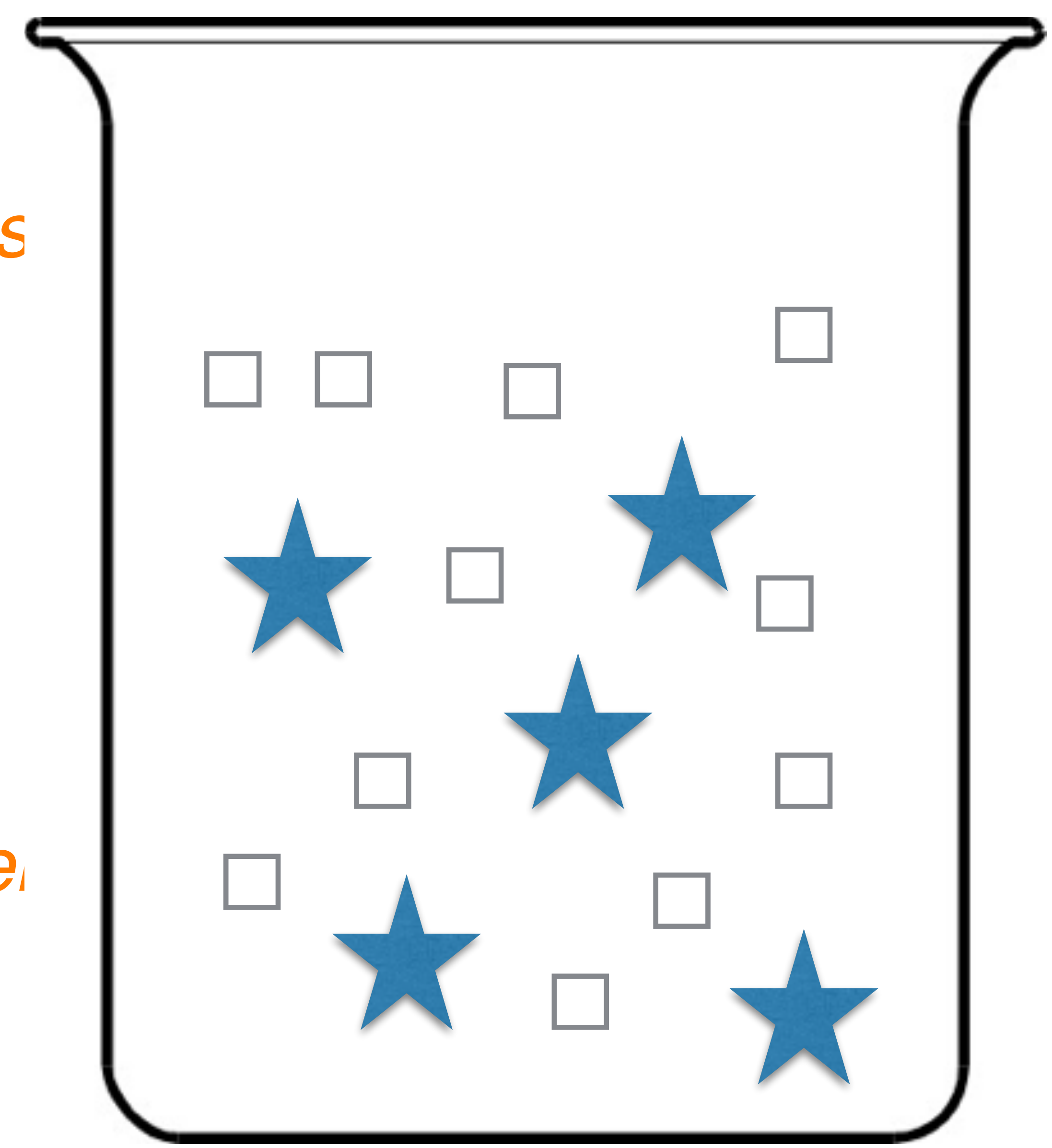
[Qian and Winfree, 2011]

Part II

Tile Assembly Models

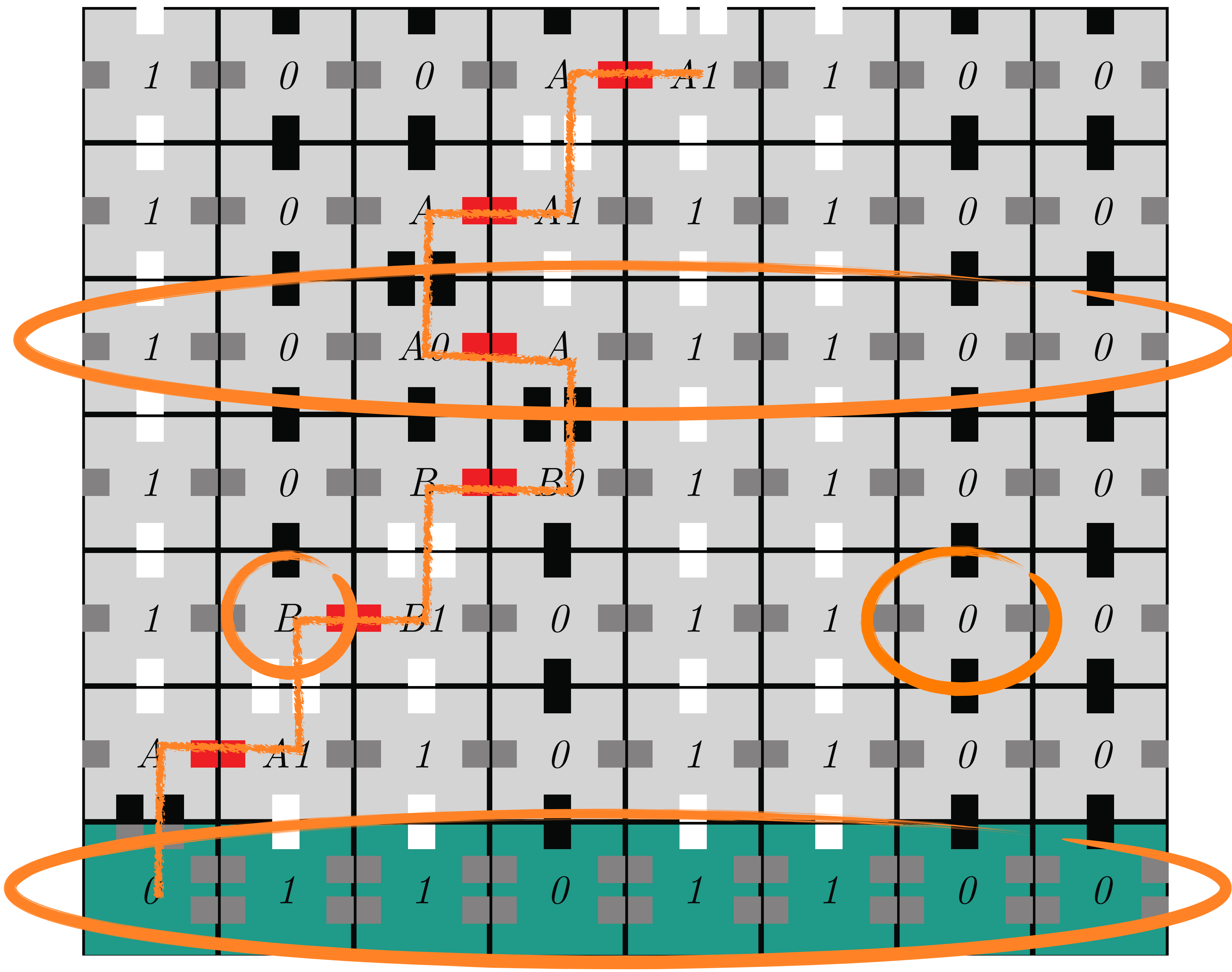
Abstract Tile Assembly Model





[Winfree, 1998]

Computation with Tiles



head moves left/right

each row represents a single Turing machine operation

position and state of head is recorded

copy tape from row to row

hard-coded seed tiles specify input

[adapted from a classic construction for Wang tiles; Winfree, 1996]

Temperature in the aTAM

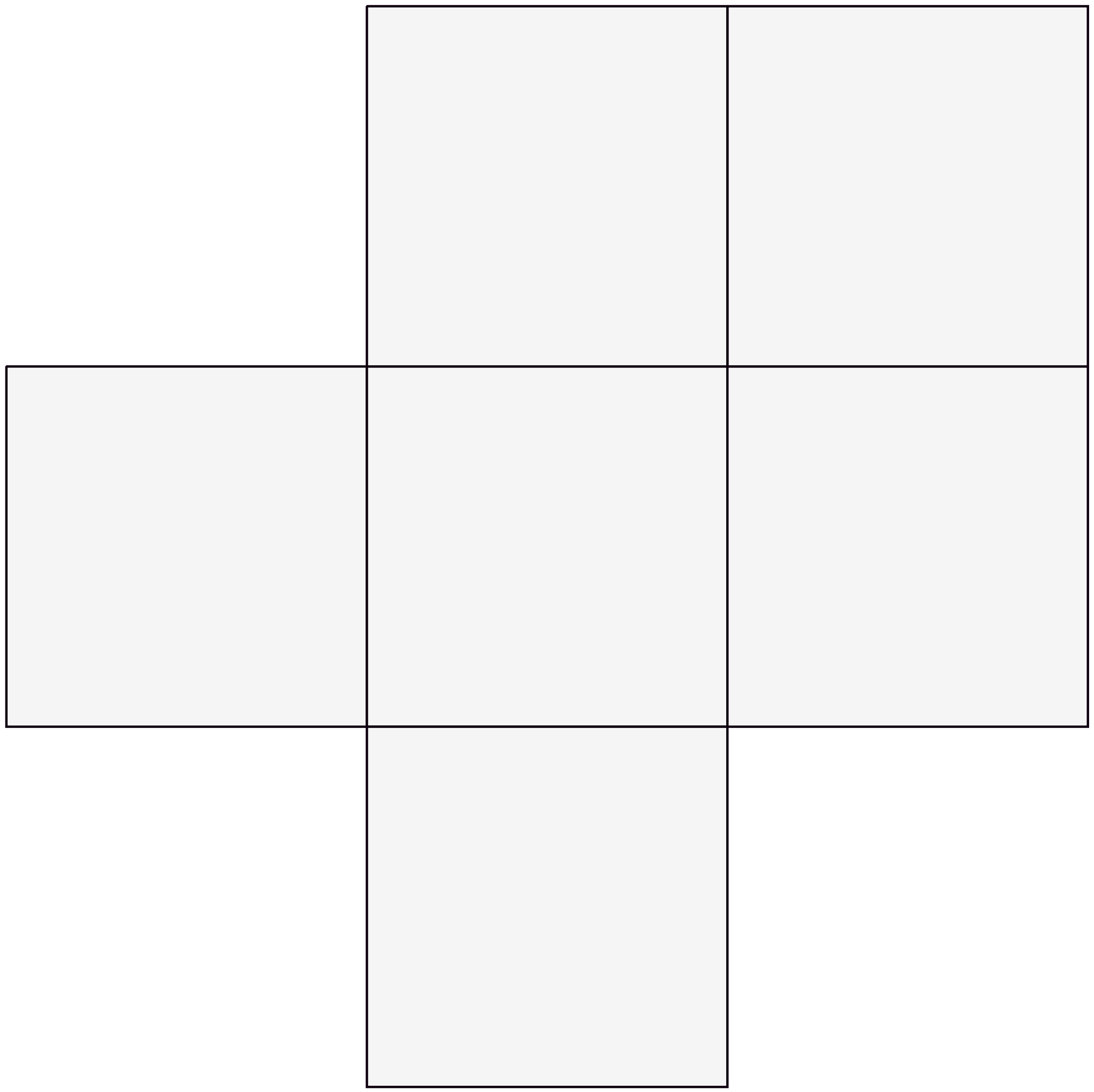
All of our proofs rely on temperature 2

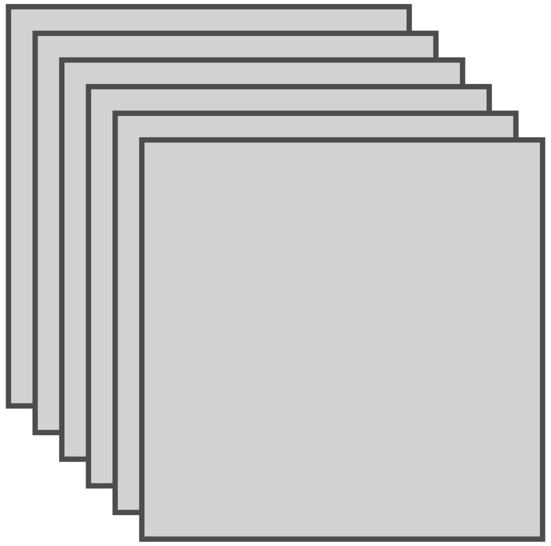
Allows “cooperative binding”, i.e. merging of information

The aTAM at temperature 1 is *believed* to be very weak…

Possibly the biggest open problem in the field!

Finite Shape Construction





Always possible with N tile types!

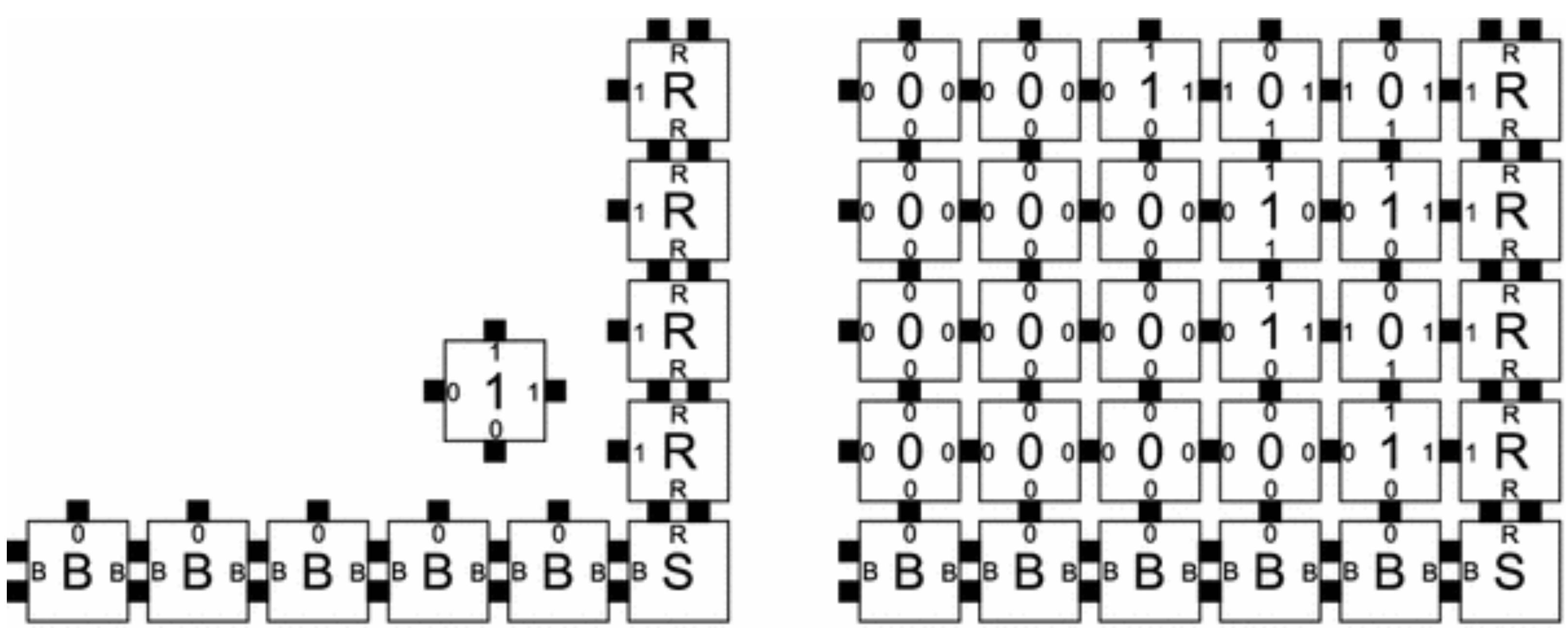
Constructing squares

> **Theorem.** An aTAM program with $\mathcal{O}(\log n)$ tile types can construct an $n \times n$ square.

[Rothemund and Winfree, 2000]

Proof idea:

Build a "binary counter" using $\mathcal{O}(1)$ tile types



(a) (b)

[image due to Patitz, 2014]

Constructing squares

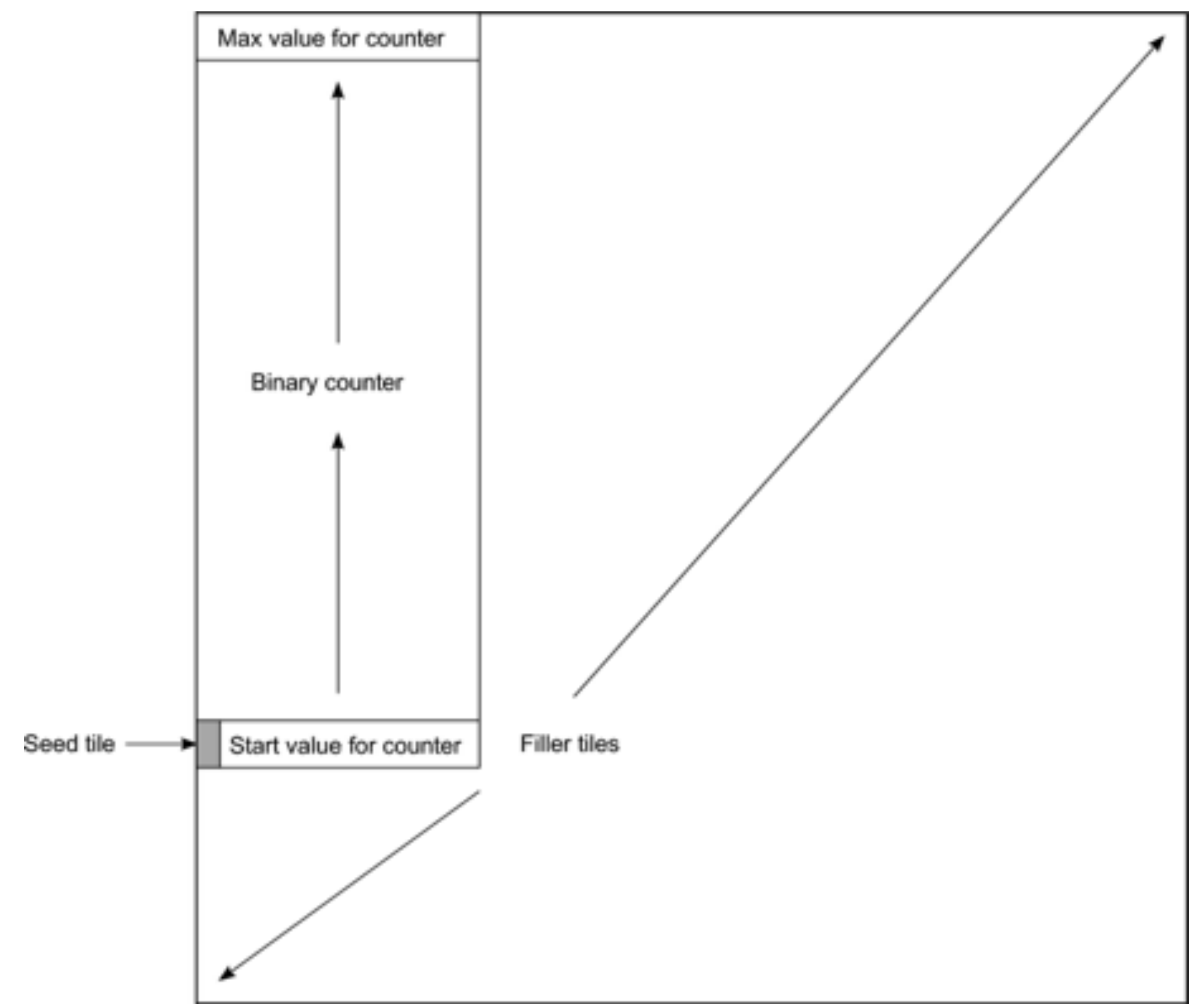
Theorem. An aTAM program with $\mathcal{O}(\log n)$ tile types can construct an $n \times n$ square.

[Rothemund and Winfree, 2000]

Proof idea:

Encode binary counter input in $\mathcal{O}(\log n)$ tile types.

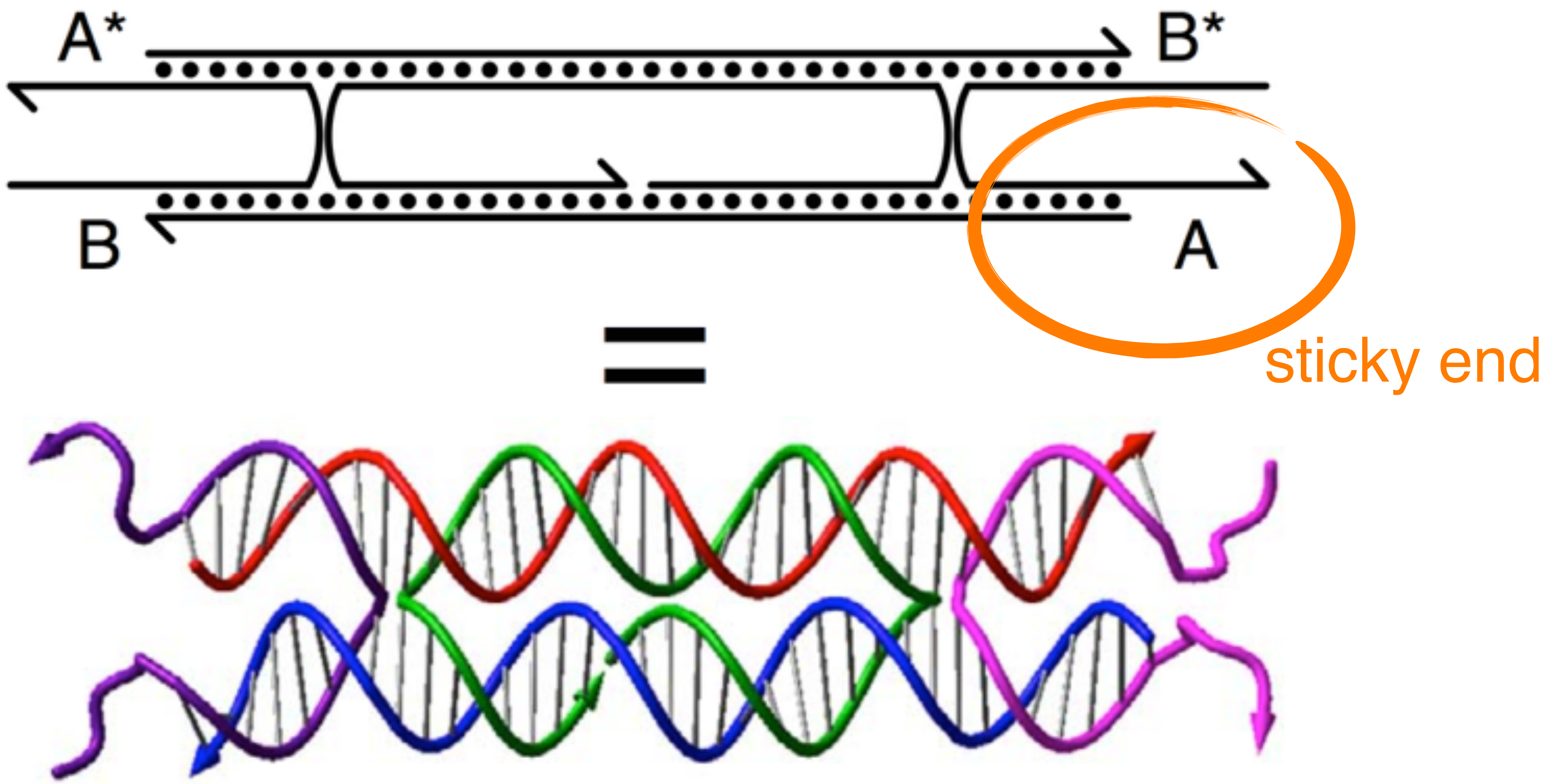
Fill in gaps with $\mathcal{O}(1)$ filler tile types



[image due to Patitz, 2014]

Tile Assembly Model

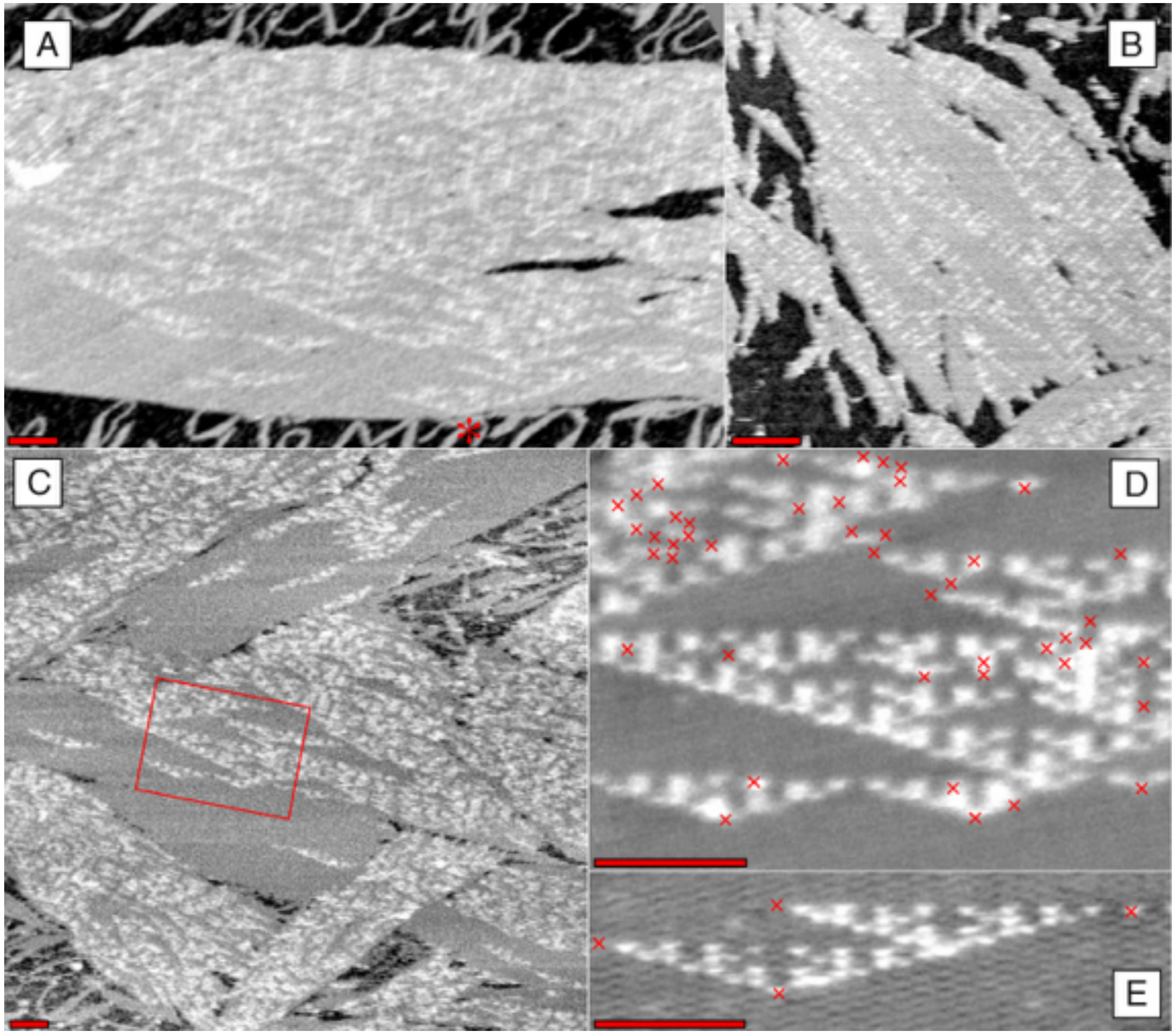
Double-crossover motif



[Fu and Seeman, 1993]

[Winfree, 1998]

Tile Assembly Model



[Rothemund, Papadakis, Winfree, 2004]

[Evans, 2014]

Constructing squares

Theorem. An aTAM program with $\mathcal{O}(\log n)$ tile types can construct an $n \times n$ square.

[Rothemund and Winfree, 2000]

Theorem. There is an aTAM program with $\mathcal{O}\left(\frac{\log n}{\log \log n}\right)$ tile types can construct a $n \times n$ square.

[Adleman, Cheng, Goel, and Huang, 2001]

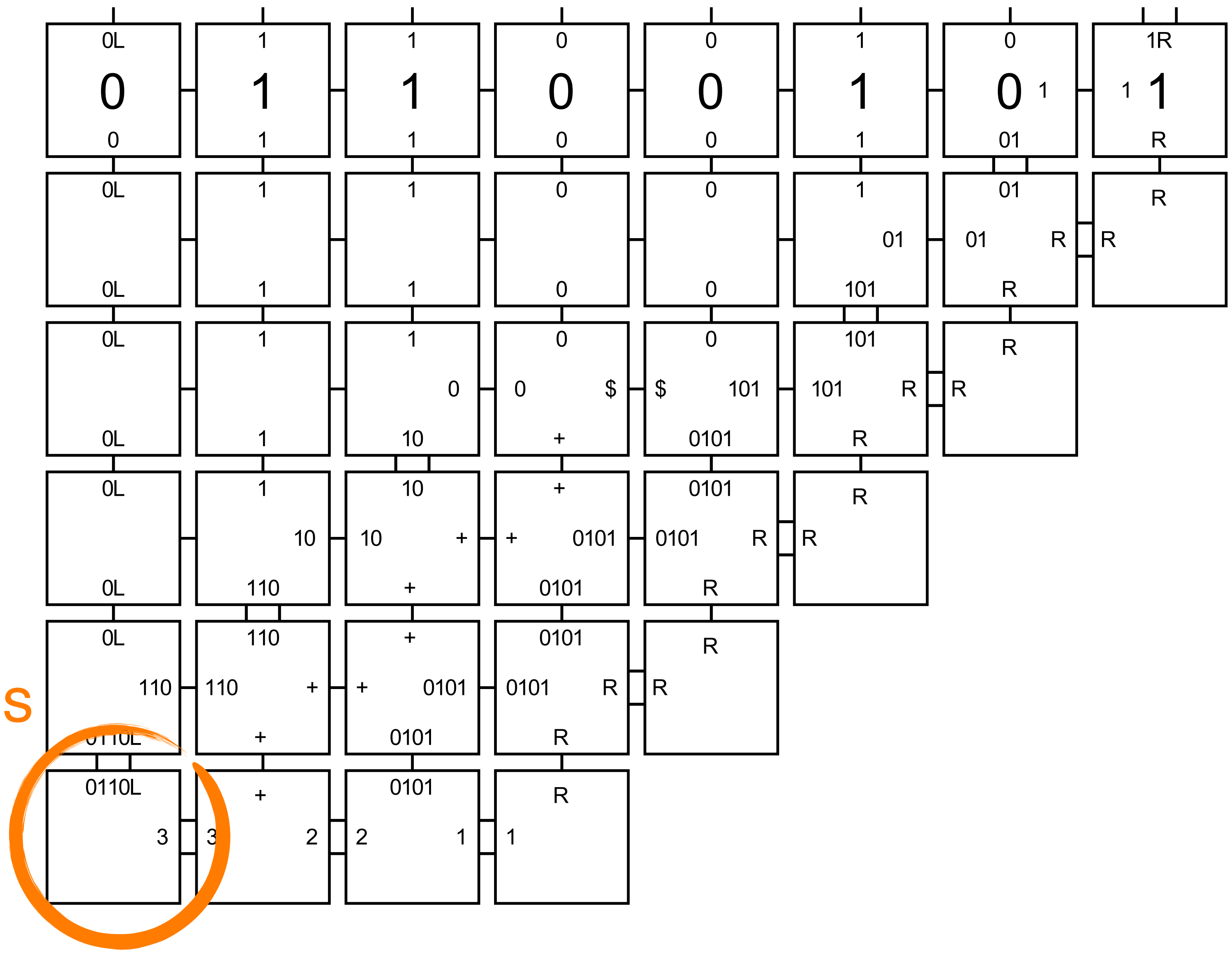
Better input encoding!

Optimal encoding of binary strings

n bit binary string to encode with tiles

each seed encodes

$$\mathcal{O}\left(\frac{\log n}{\log \log n}\right)$$



[image due to Patitz, 2014]

Optimal encoding of binary strings

n bit binary string to encode with tiles

$$\mathcal{O}\left(\frac{n}{\log n}\right)$$

seed tiles

$$\mathcal{O}\left(\frac{n}{\log n}\right)$$

unpacking tiles

Information-theoretic optimum!

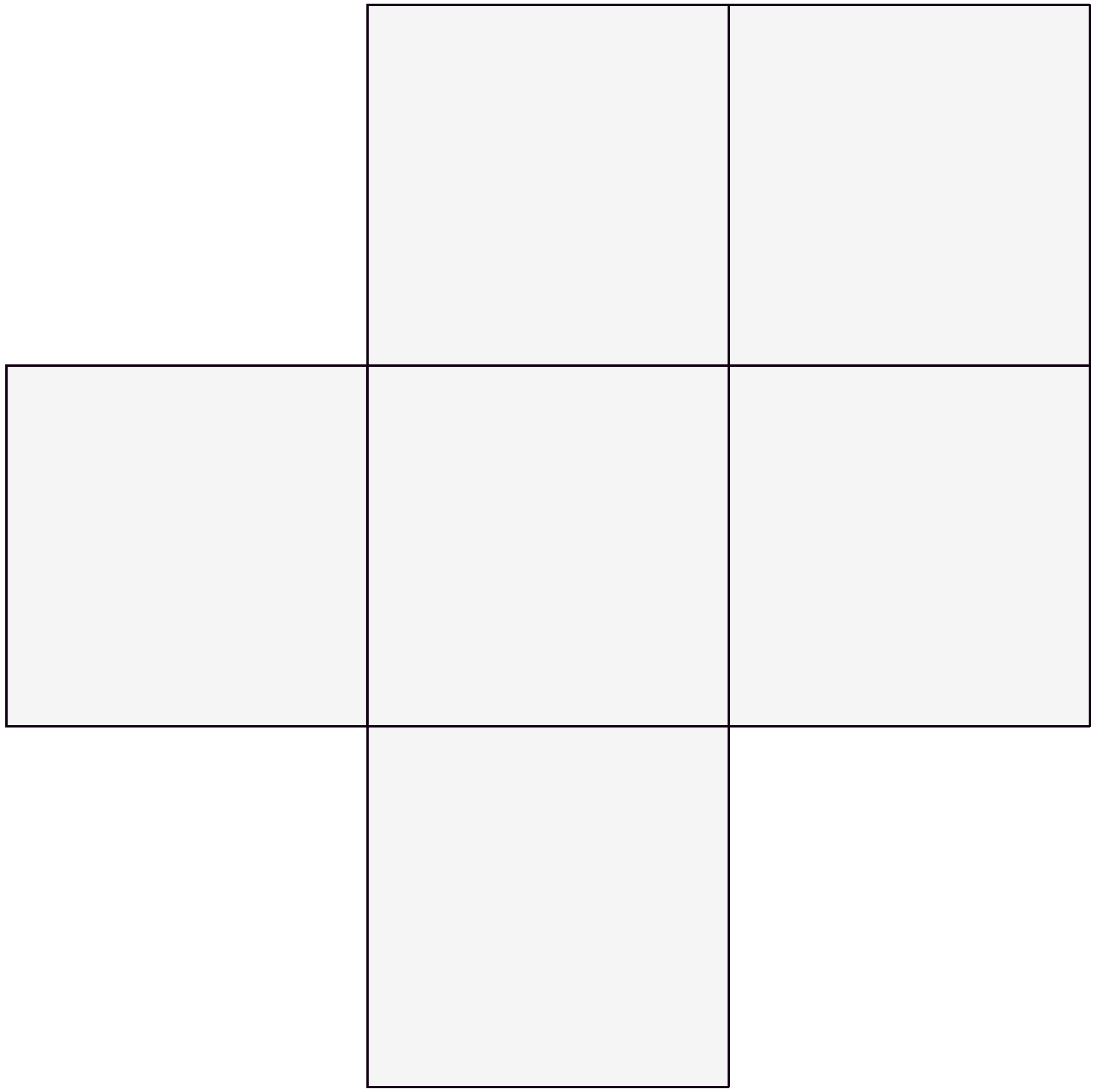
[image due to Patitz, 2014]

Finite Shape Construction

Kolmogorov complexity

$$K_U(\mathcal{S}) =$$

size of the smallest program for U that outputs $\mathcal{S}$



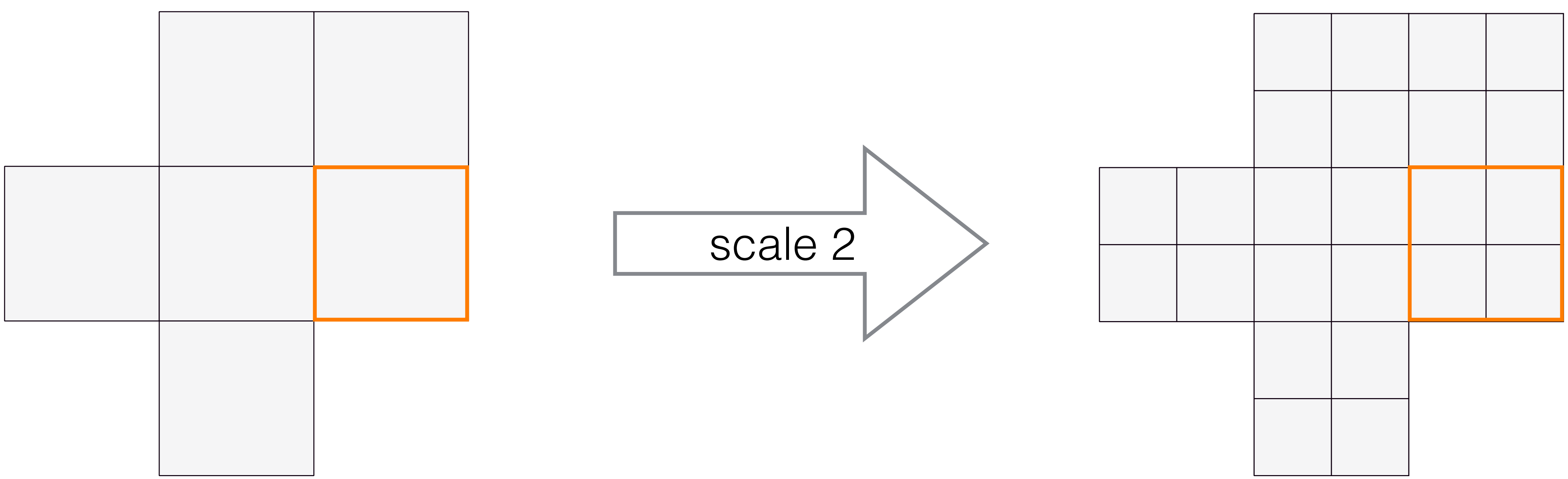
Shape Construction in the aTAM

Theorem. For any shape $\mathcal{S}$, there is an aTAM tile set T that constructs $\mathcal{S}$ at some scale such that

$$|T| \in \Theta\left(\frac{K_U(\mathcal{S})}{\log K_U(\mathcal{S})}\right)$$

[Soloveichik and Winfree, 2007]

optimal input encoding



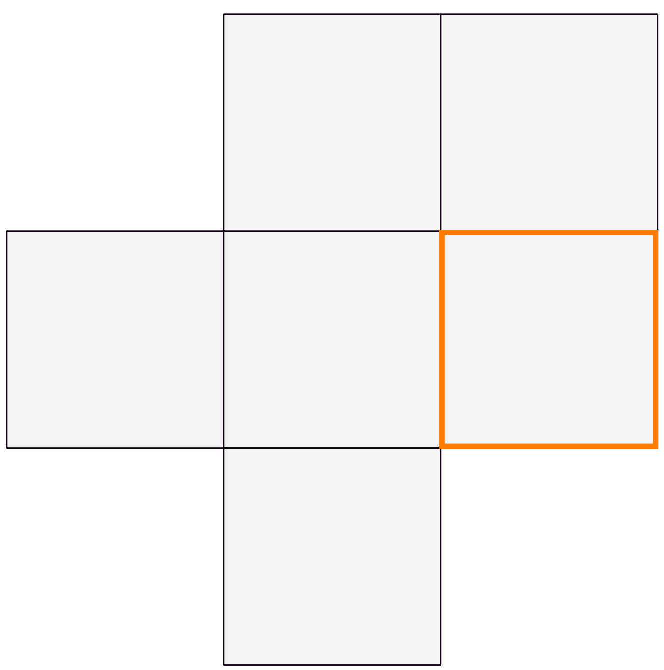
Shape Construction in the aTAM

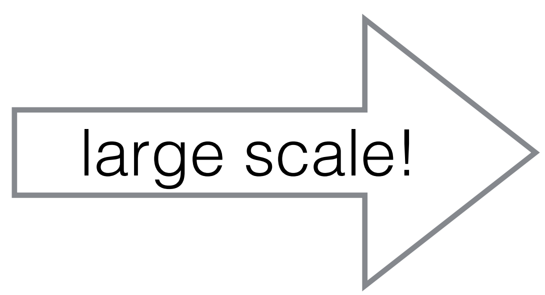
Theorem. For any shape $\mathcal{S}$, there is an aTAM tile set T that constructs $\mathcal{S}$ at some scale such that

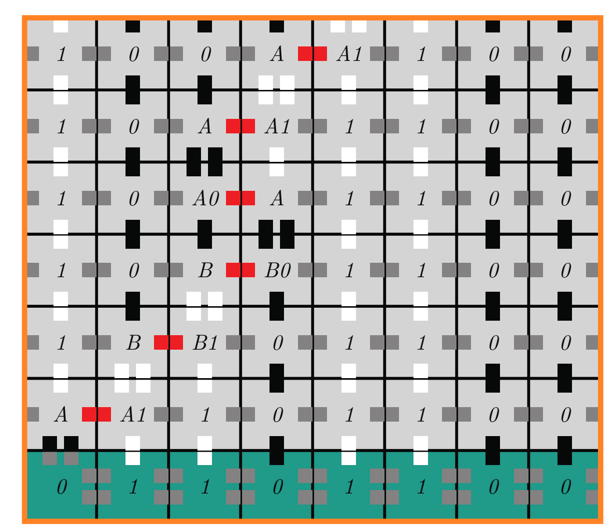
$$|T| \in \Theta\left(\frac{K_U(\mathcal{S})}{\log K_U(\mathcal{S})}\right)$$

[Soloveichik and Winfree, 2007]

optimal input encoding







But wait, there’s more!

aTAM is extremely well studied

aTAM is intrinsically universal [Doty, Lutz, Patitz, Schweller, Summers, Woods, 2012]

classification of shapes by difficulty [Aggarwal†, Cheng, Goldwasser, Kao, Moisset de Espanes, Schweller, 2005]

But wait, there's more!

Dozens of variants

2-handed tile assembly model

hierarchical tile assembly model

kinetic tile assembly model

polyomino tile assembly model

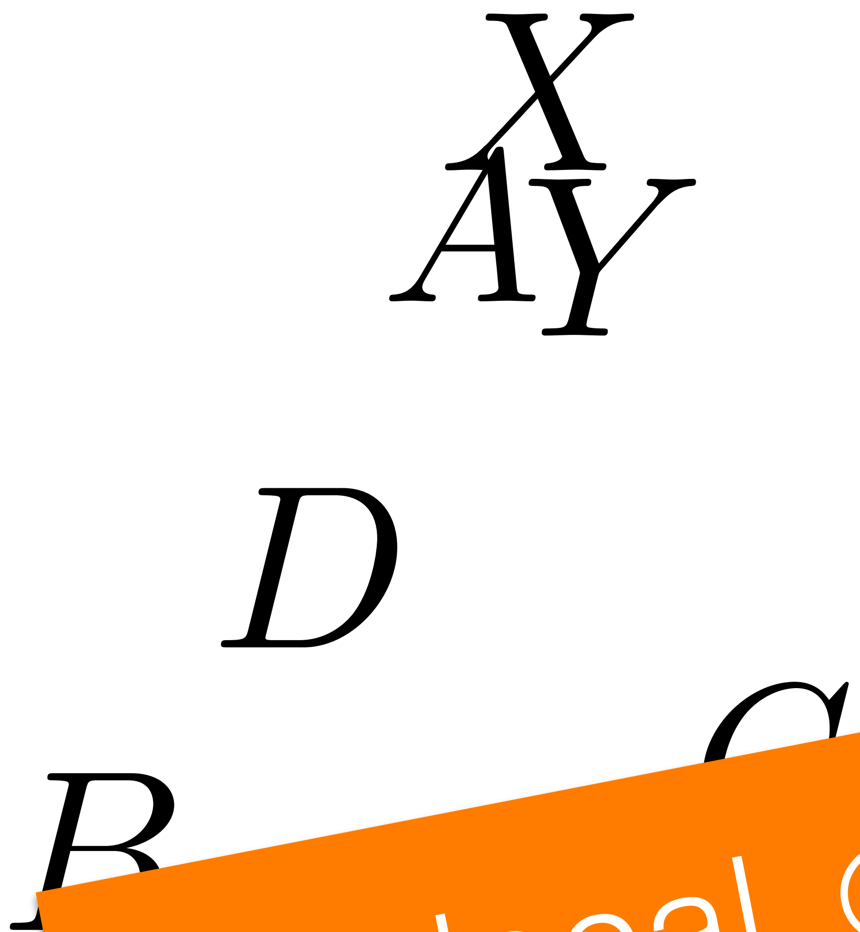
flipping tile assembly model

xTAM has been studied for many, many *x*

"One tile to rule them all" in non-square tile systems [Demaine, Demaine, Fekete, Patitz, Schweller, Winslow, Woods, 2012]

Locality: friend and foe

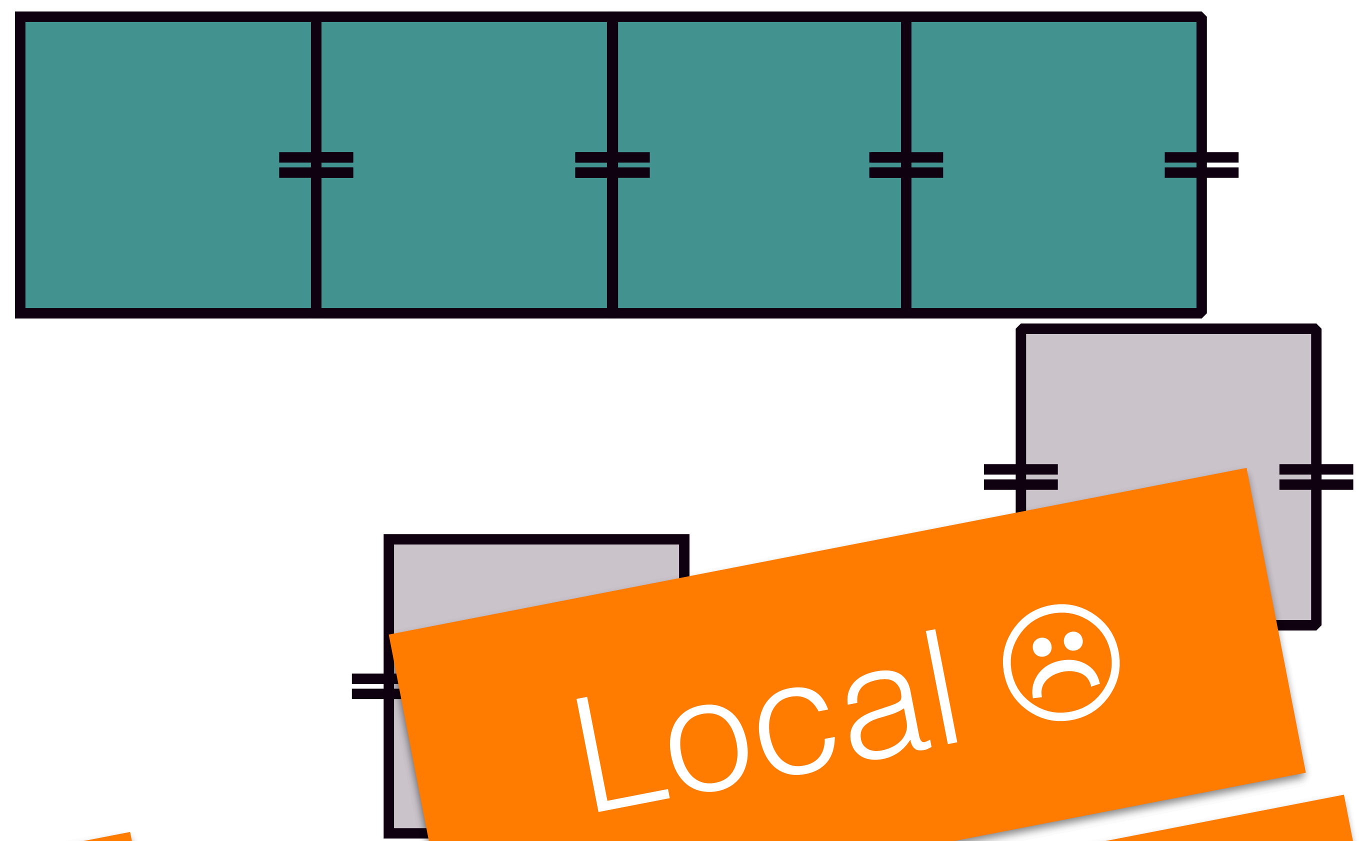
CRNs



Non-local ☺

No geometry ☹

Tile assembly model



Local ☹

Geometry ☺

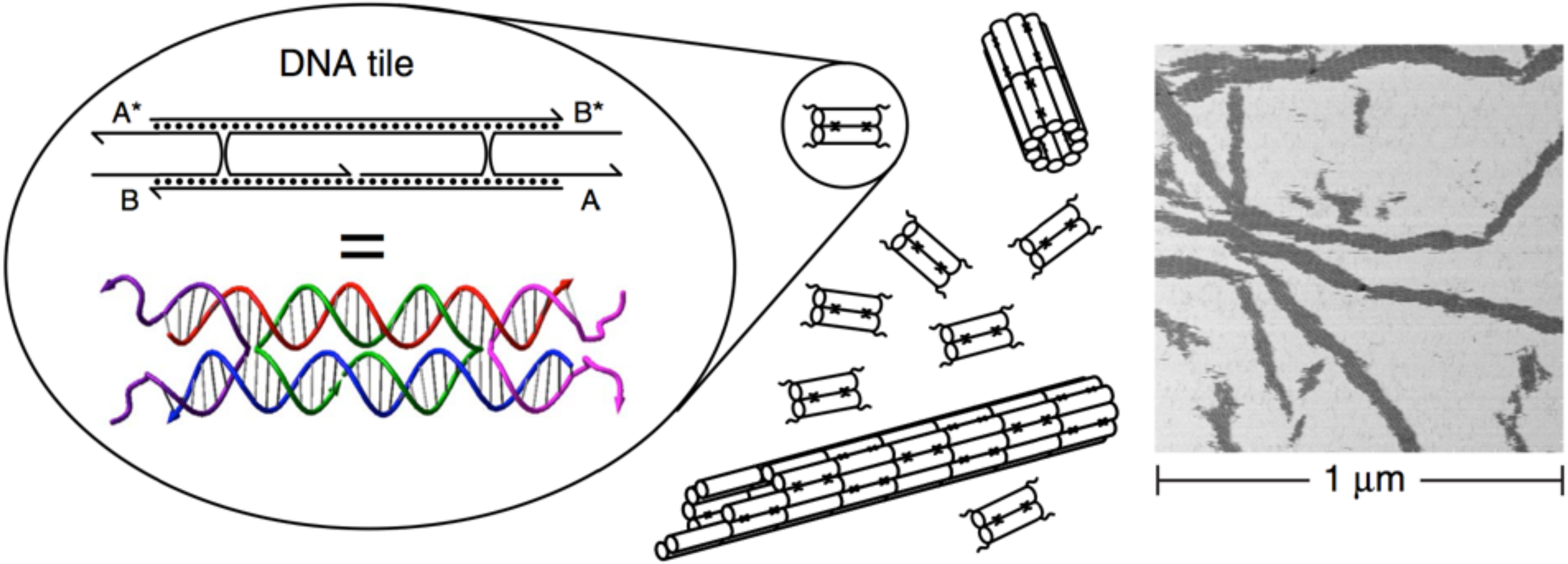
In nature…

$$
\begin{tikzcd}
\text{Dynamics} \arrow[rr, bend left] & & \text{Structure} \arrow[ll, bend left]
\end{tikzcd}
$$

Part III

Hybrid models of molecular programming

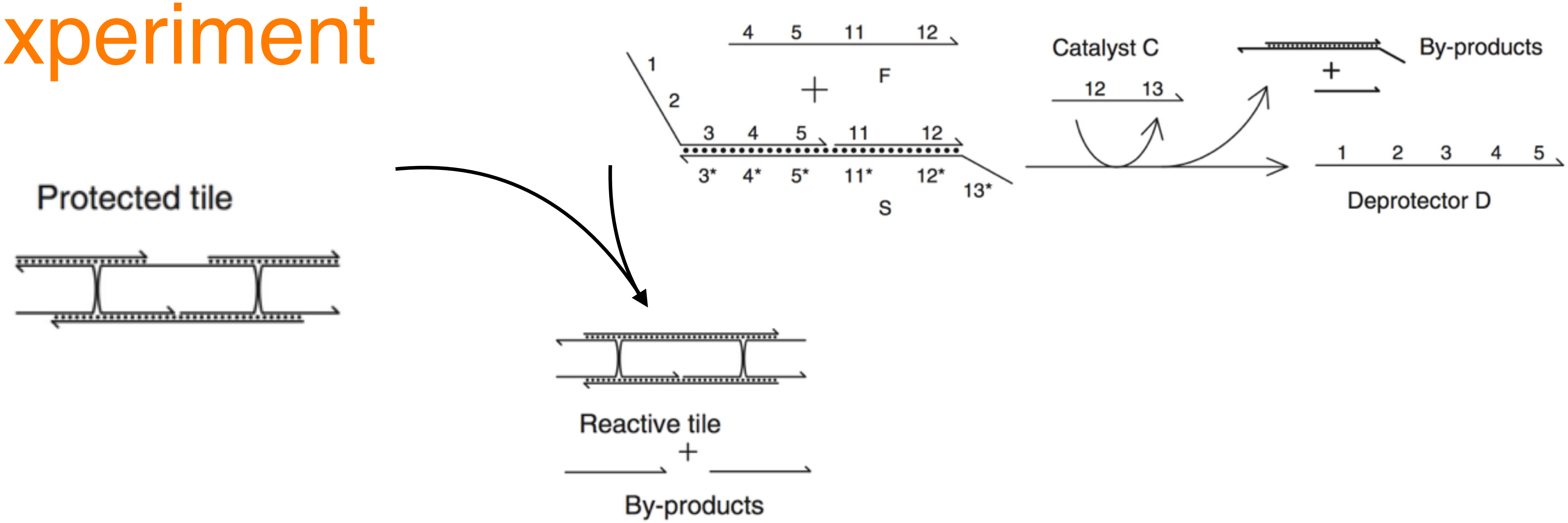
CRN Control of Tile Self-Assembly

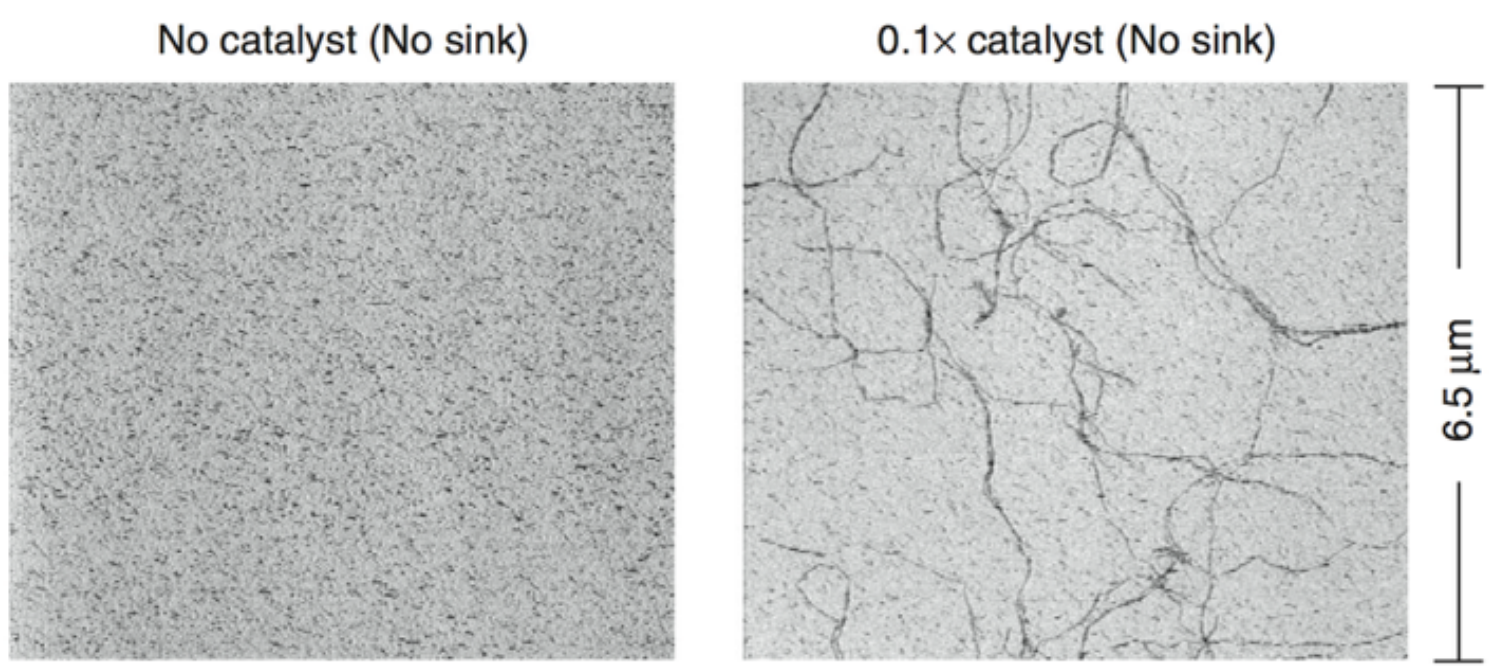


[Zhang, et al. 2013]

CRN Control of Tile Self-Assembly

Experiment





[Zhang, et al. 2013]

CRN Control of Tile Self-Assembly

Theory

?

CRN Control of Tile Self-Assembly

Theory

"A theoretical framework is needed to characterize what molecular behaviors can or cannot be achieved by integrated DNA tile and strand-displacement systems…."

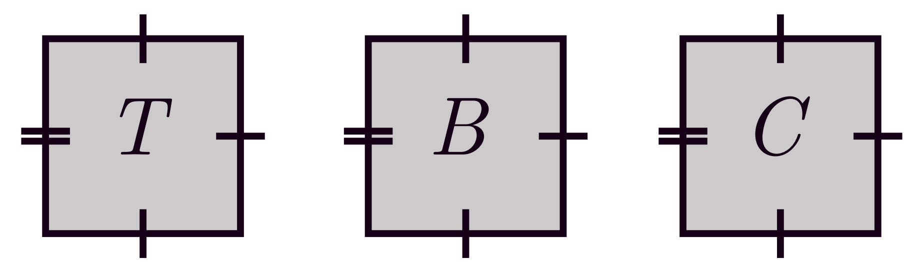
[D. Y. Zhang, R. F. Hariadi, H. M. T. Choi, and E. Winfree, "Integrating DNA strand-displacement circuitry with DNA tile self-assembly," Nat Commun, vol. 4, Jun. 2013.]

Chemical Reaction Network-Controlled Tile Assembly Model (**CRN-TAM**)

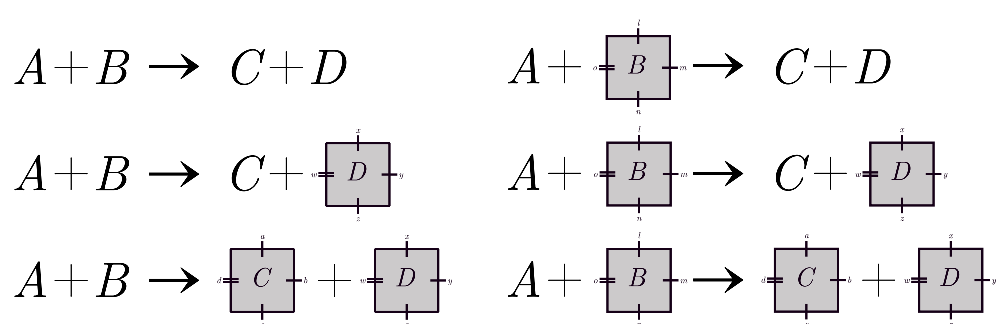
Program includes:

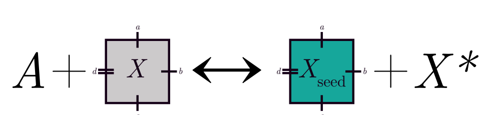
$$A, B, C, X, Y, Z, \ldots$$

finite set of *signals*

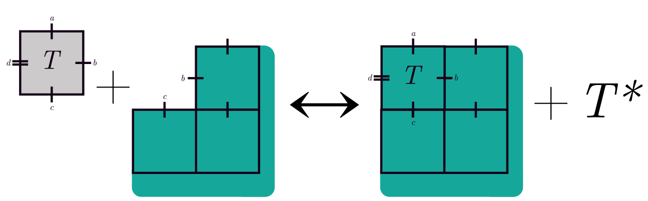


finite set of *tiles*

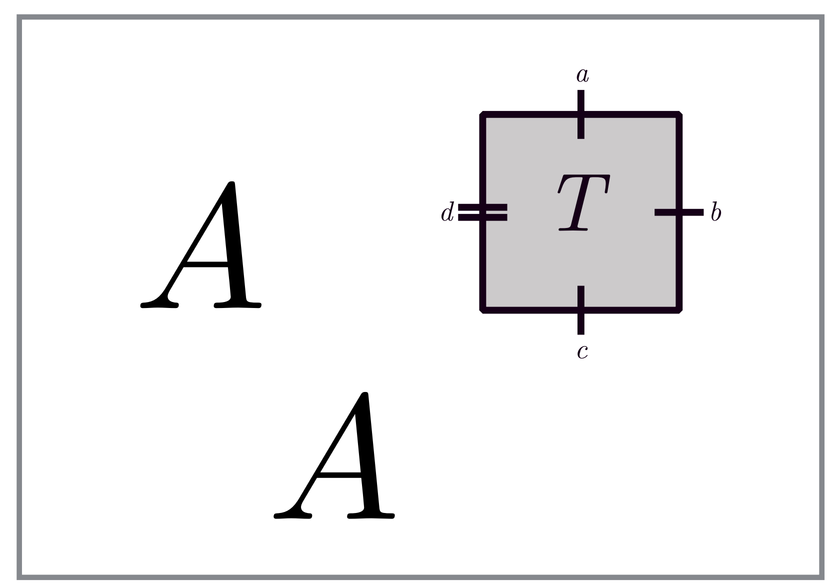




finite set of *reactions*



tile assembly reactions



initial state

[S and Winfree 2015]

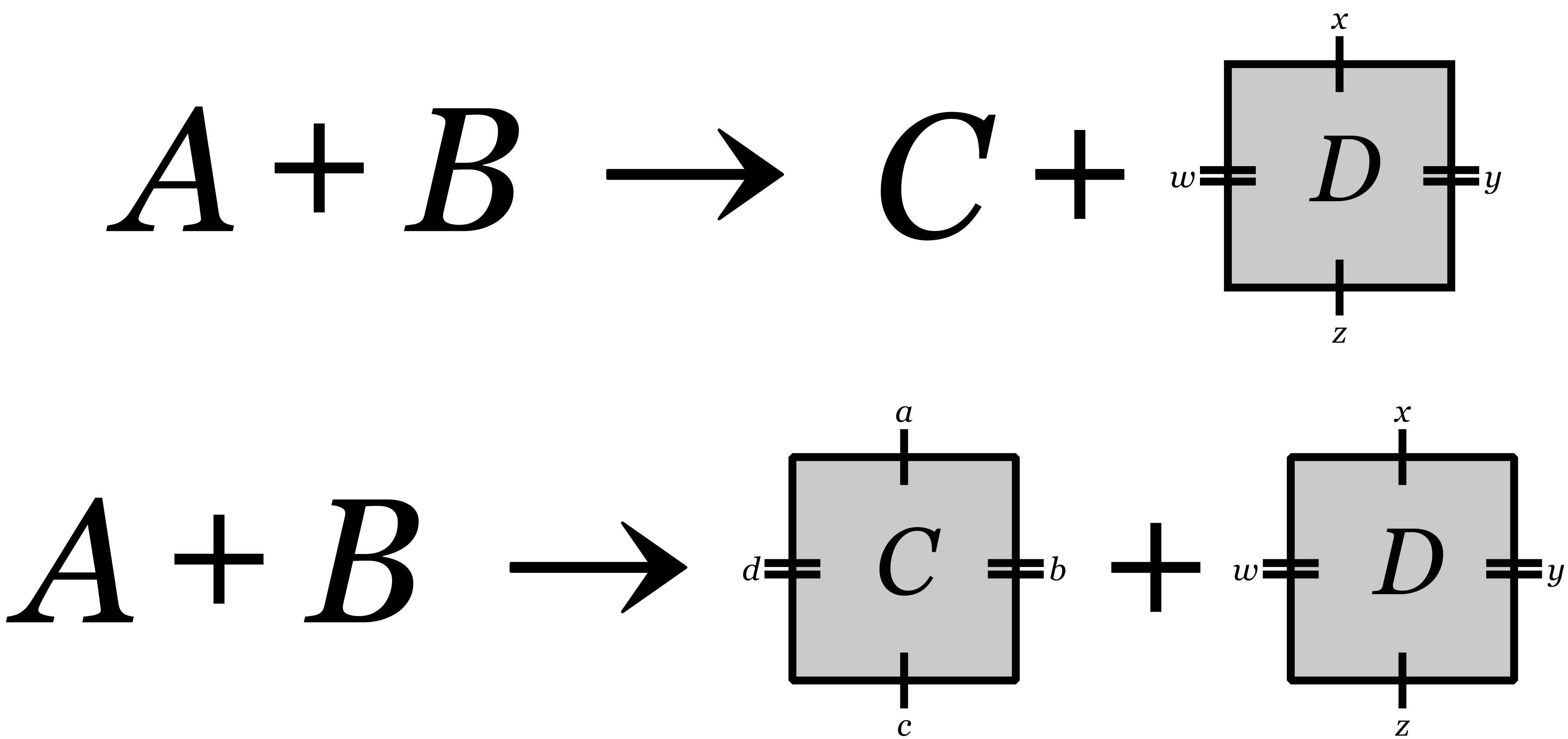
Chemical Reaction Network-Controlled Tile Assembly Model (**CRN-TAM**)

Normal CRN reactions

$$A + B \rightarrow C + D$$

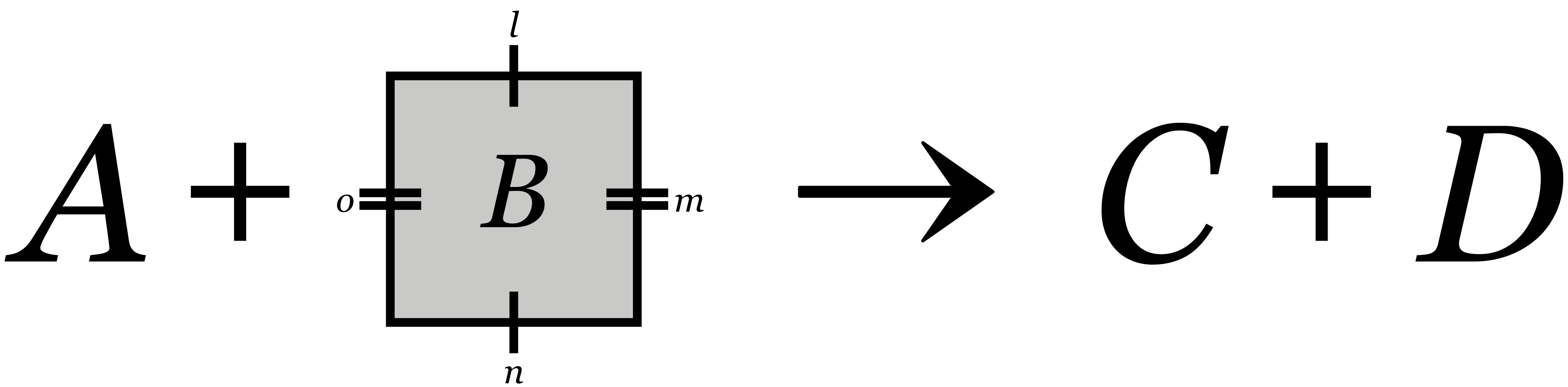
Chemical Reaction Network-Controlled Tile Assembly Model (**CRN-TAM**)

Creating tiles



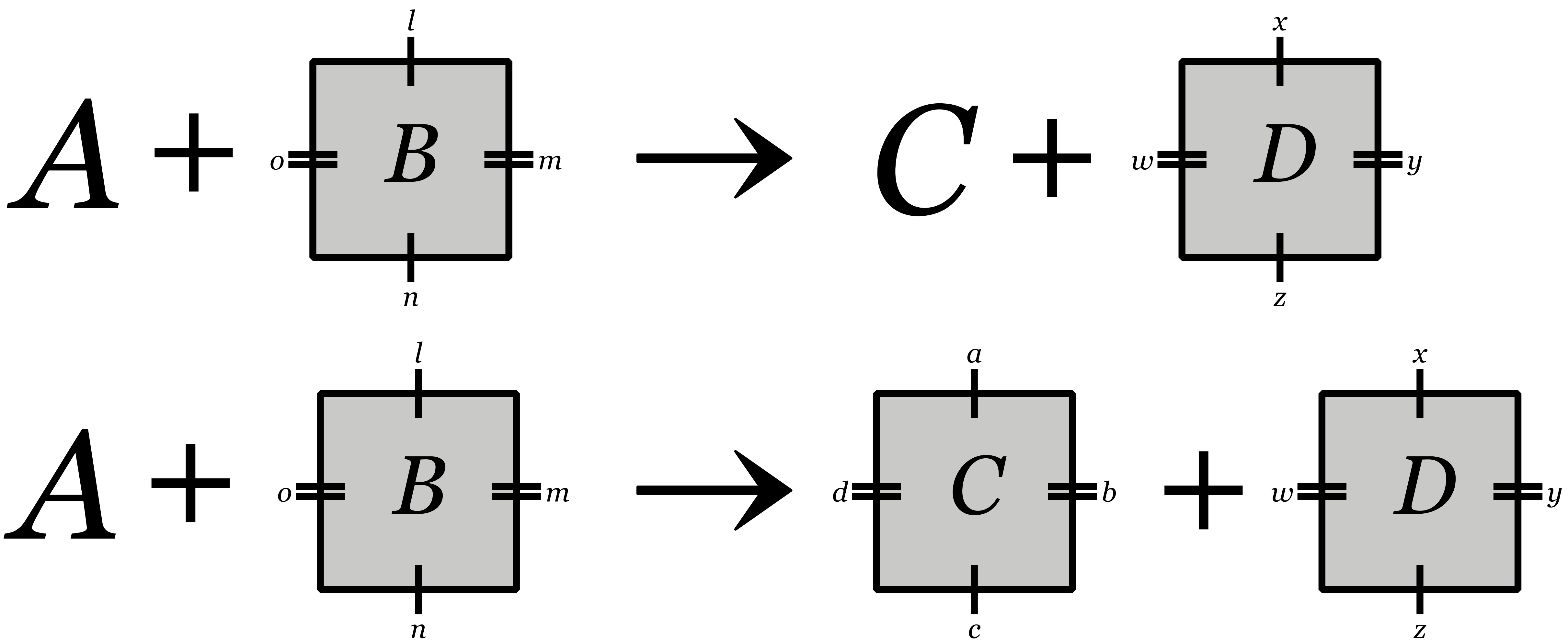
Chemical Reaction Network-Controlled Tile Assembly Model (**CRN-TAM**)

Deleting tiles



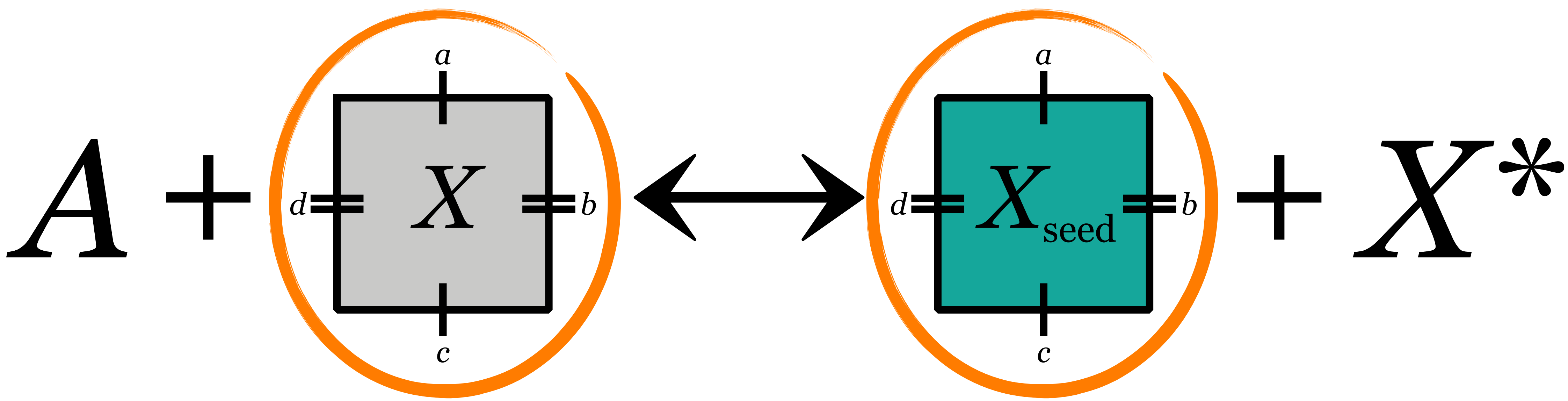
Chemical Reaction Network-Controlled Tile Assembly Model (**CRN-TAM**)

Relabelling tiles



Chemical Reaction Network-Controlled Tile Assembly Model (**CRN-TAM**)

Activating/deactivating tiles



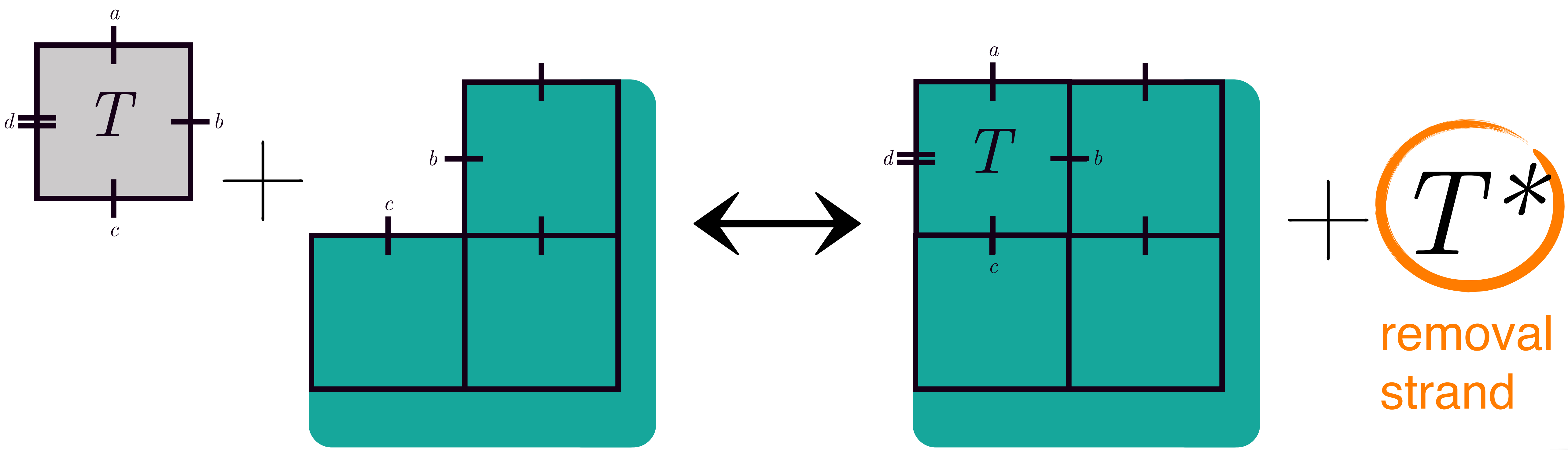
“inactive” tile

“active” seed assembly

Variable # of assemblies!

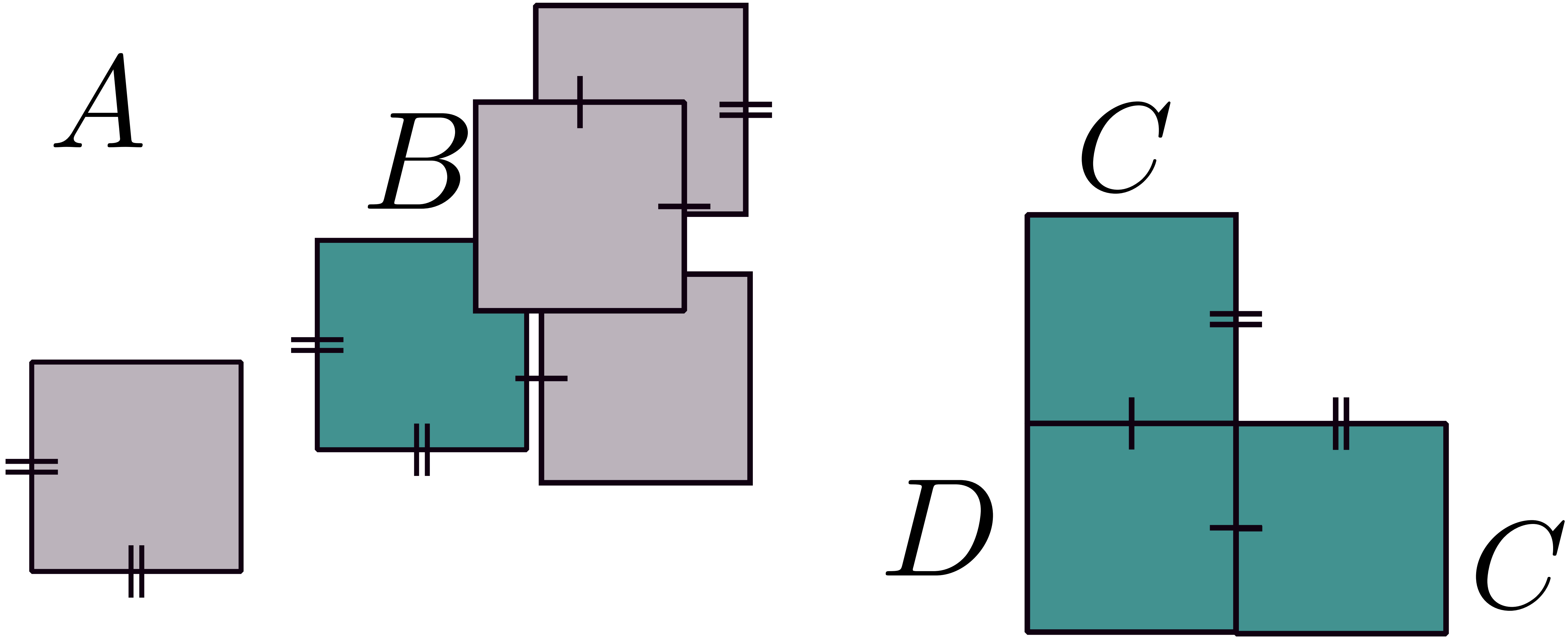
Chemical Reaction Network-Controlled Tile Assembly Model (**CRN-TAM**)

Tile attachment/detachment



Feedback from structure!

Example



Program Complexity

number of signals

number of variables

number of reactions

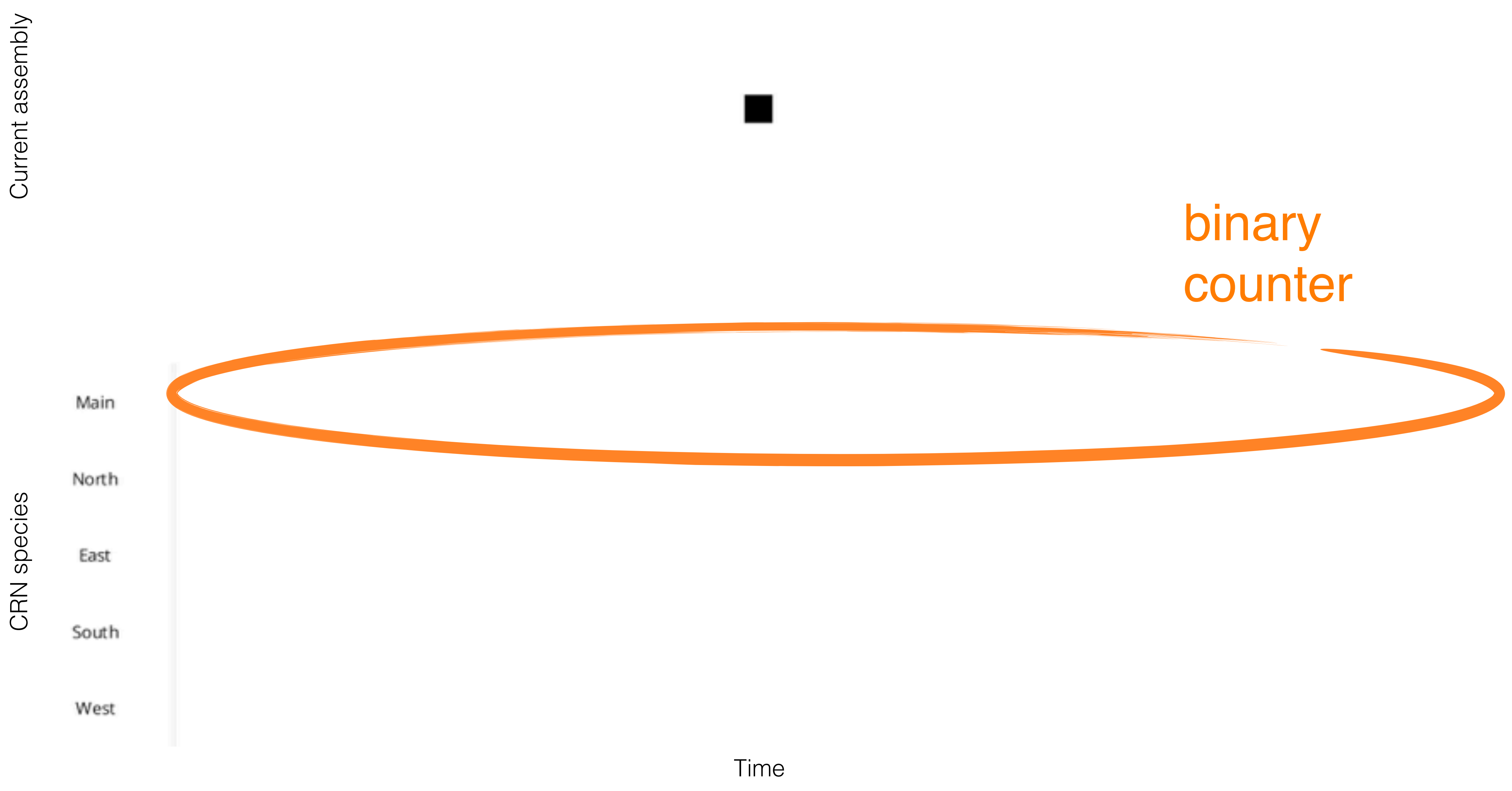
$$K^{\tau}_{\mathrm{CT}}(P) = |S| + |T| + |R| + |I|$$

size of initial state

$$\equiv |S| + |T| + |R| + \sum_{z \in (S \cup T)} \log(I(z) + 1)$$

Building skinny shapes

[pattern designed by Kevin Li, software by S]



Turing Machines

Theorem. The CRN-TAM is Turing-universal at temperature $\tau = 2$.

Proof. The aTAM is Turing-universal at temperature $\tau = 2$.

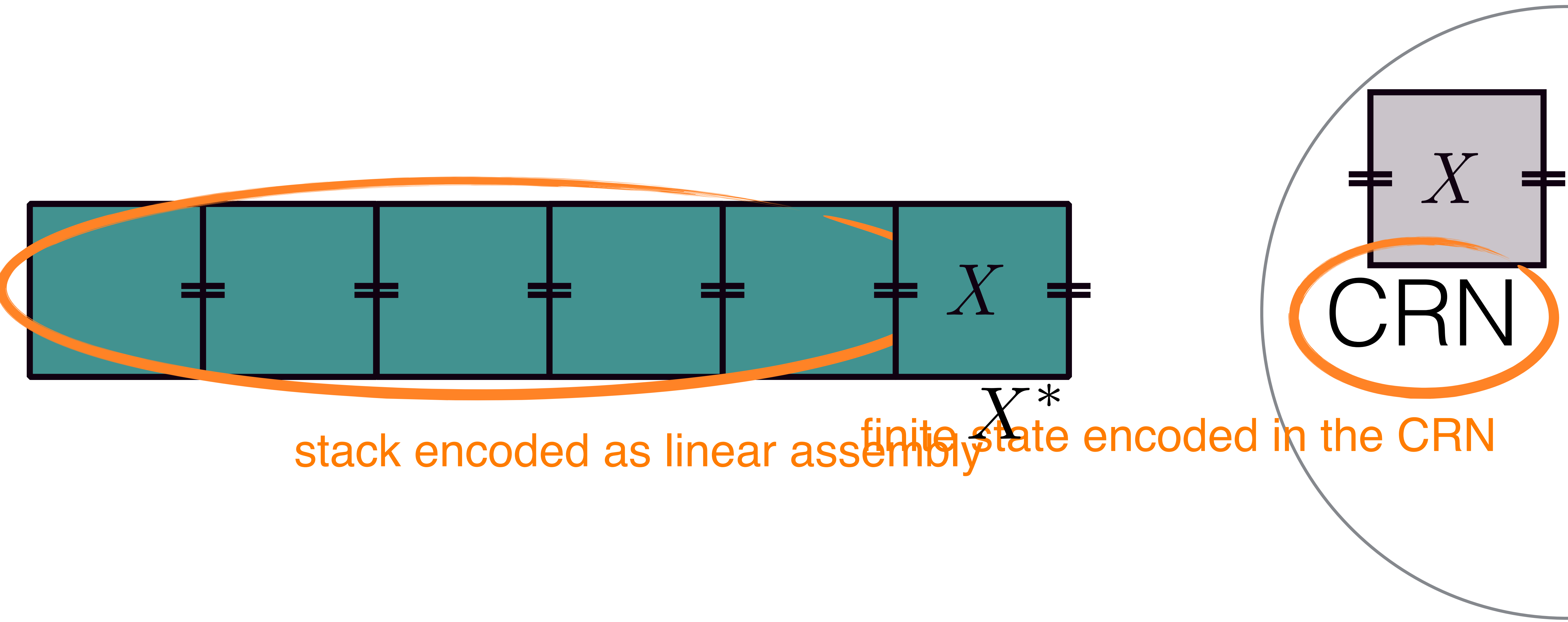
(Simulation of computation histories)

Requires $\Omega(s \times t)$ space!

(Probably) requires temp. 2!

Stack Machines

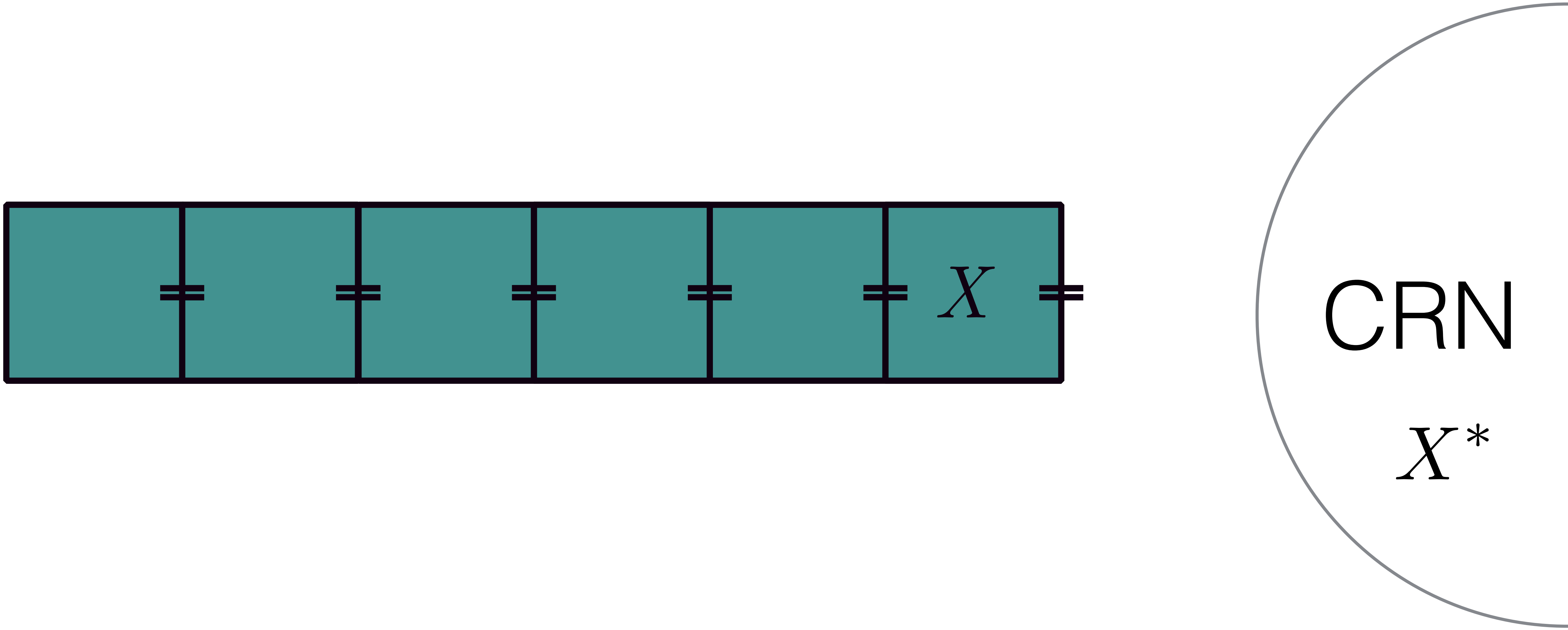
Push operation



Theorem. Turing machines $\Leftrightarrow$ multi-stack machines.

Stack Machines

Pop operation



Theorem. Turing machines $\Leftrightarrow$ multi-stack machines.

Stack Machines

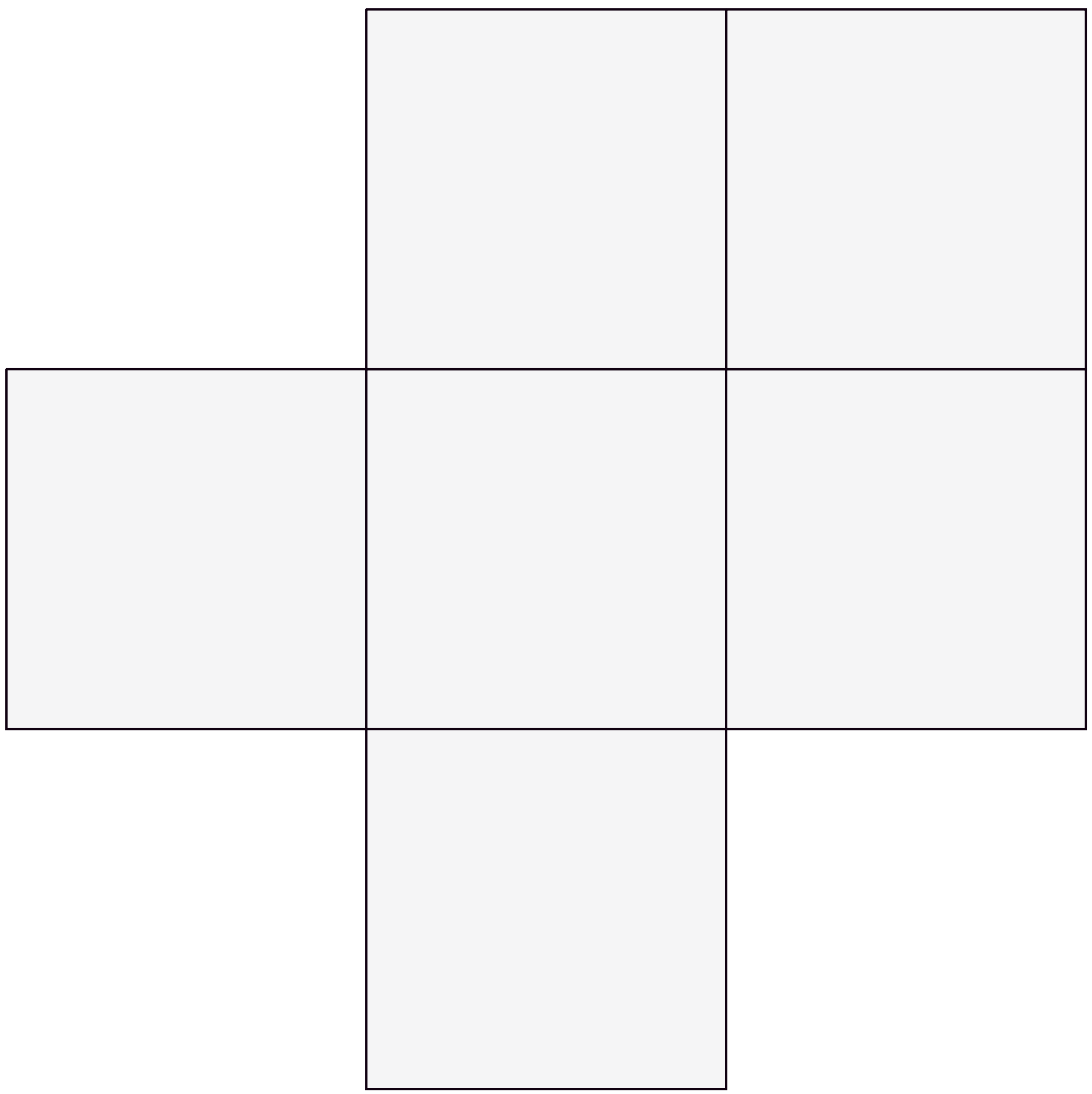
Theorem. Turing machines $\Leftrightarrow$ multi-stack machines.

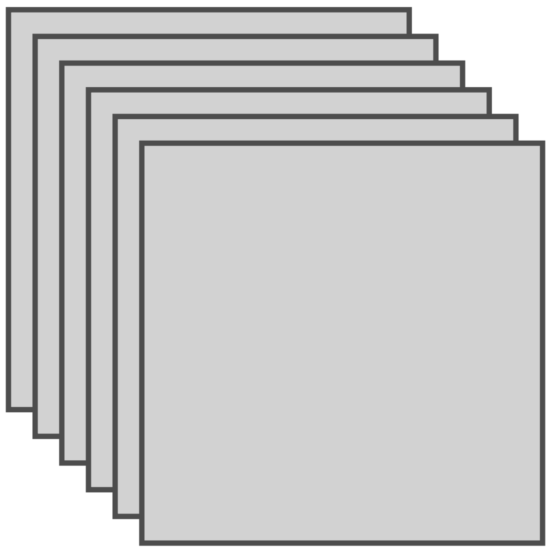
Theorem. The CRN-TAM is Turing universal at all temperatures $\tau \geq 1$ using $\mathcal{O}(1)$ space in one dimension.

Only $\Theta(s)$ space!

Temperature 1!

Finite Shape Construction





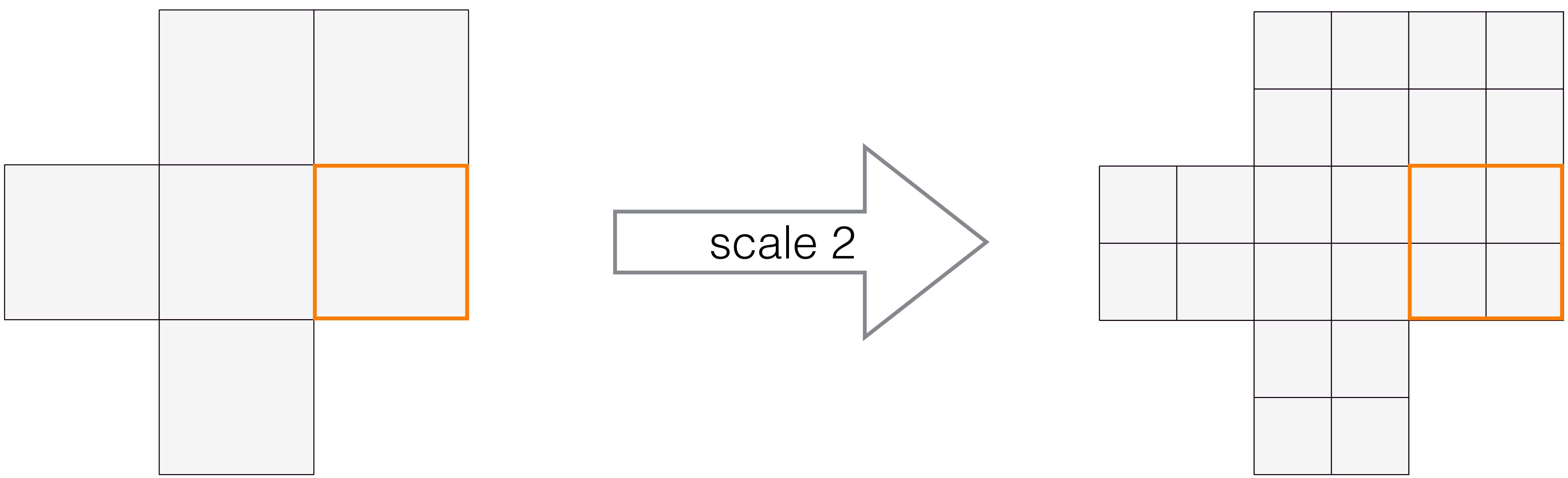
Shape Construction in the aTAM

Theorem. For any shape $\mathcal{S}$, there is an aTAM tile set T that constructs $\mathcal{S}$ at some scale such that

$$|T| \in \Theta\left(\frac{K_U(\mathcal{S})}{\log K_U(\mathcal{S})}\right)$$

[Soloveichik and Winfree, 2007]

conversion from “bits” to “tiles”



Shape Construction in the aTAM

Theorem. For any shape $\mathcal{S}$, there is an aTAM tile set T that constructs $\mathcal{S}$ at some scale such that

$$|T| \in \Theta\left(\frac{K_U(\mathcal{S})}{\log K_U(\mathcal{S})}\right)$$

[Soloveichik and Winfree, 2007]

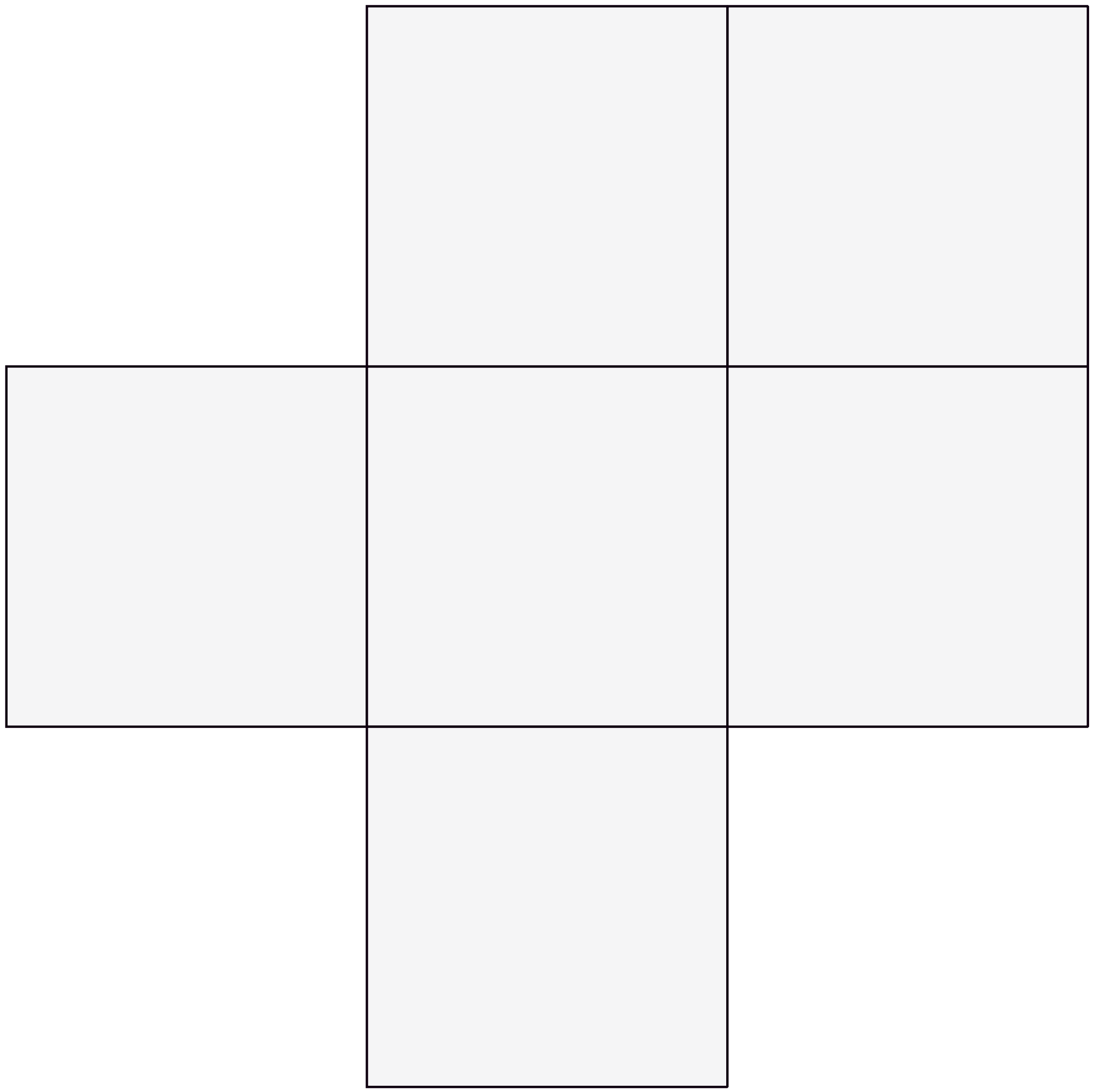
Huge scale = huge shapes!

Shape Construction in the CRN-TAM

CRN-TAM complexity

$$K^{\tau}_{\mathrm{CT}}(\mathcal{S}) =$$

size of the smallest CRN-TAM program that constructs $\mathcal{S}$



Shape Construction in the CRN-TAM

Theorem. For any shape $\mathcal{S}$, the CRN-TAM complexity of $\mathcal{S}$ at scale 2 and any temperature $\tau \geq 1$ is:

$$K_{\mathrm{CT}}^{\tau}(\mathrm{scale}_2(\mathcal{S})) \in \Theta\left(\frac{K_U(\mathcal{S})}{\log K_U(\mathcal{S})}\right)$$

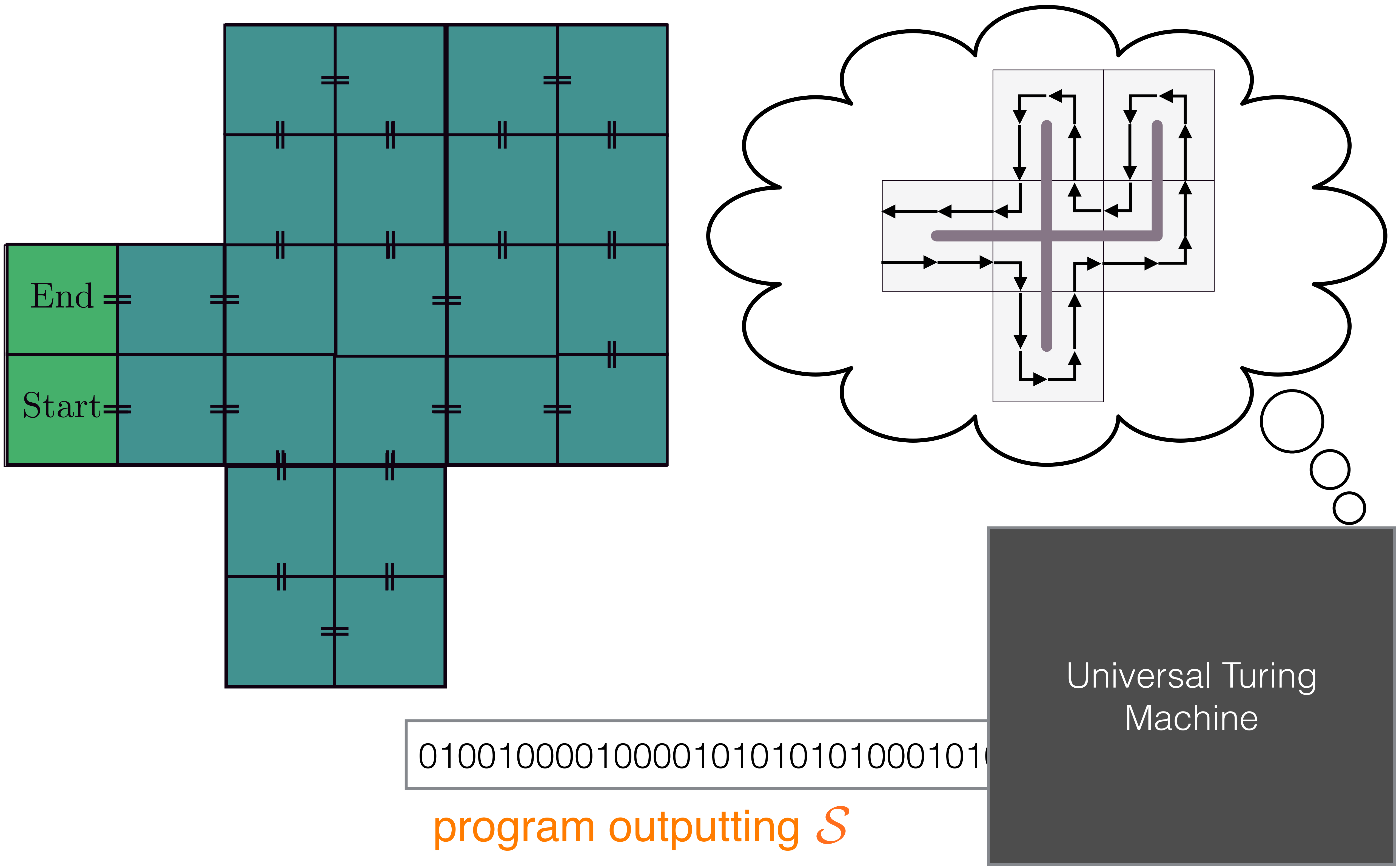
[S and Winfree, 2015]

same conversion from "bits" to "CRN-TAM complexity"

Constant scale!

Temperature 1!

Shape Construction in the CRN-TAM



Shape Construction in the CRN-TAM

Program components

Universal Turing machine (constant complexity)

Path tiles (constant complexity)

Depth-first search (constant complexity)

Universal Turing machine program ($K_U(\mathcal{S})$ bits)

Theorem. A binary string of length n can be encoded in a (temperature 1) CRN-TAM program with complexity $\Theta(n/\log n)$.

[S and Winfree, 2015]

Shape Construction in the CRN-TAM

Theorem. A binary string of length n can be encoded in a (temperature 1) CRN-TAM program with complexity $\Theta(n/\log n)$.

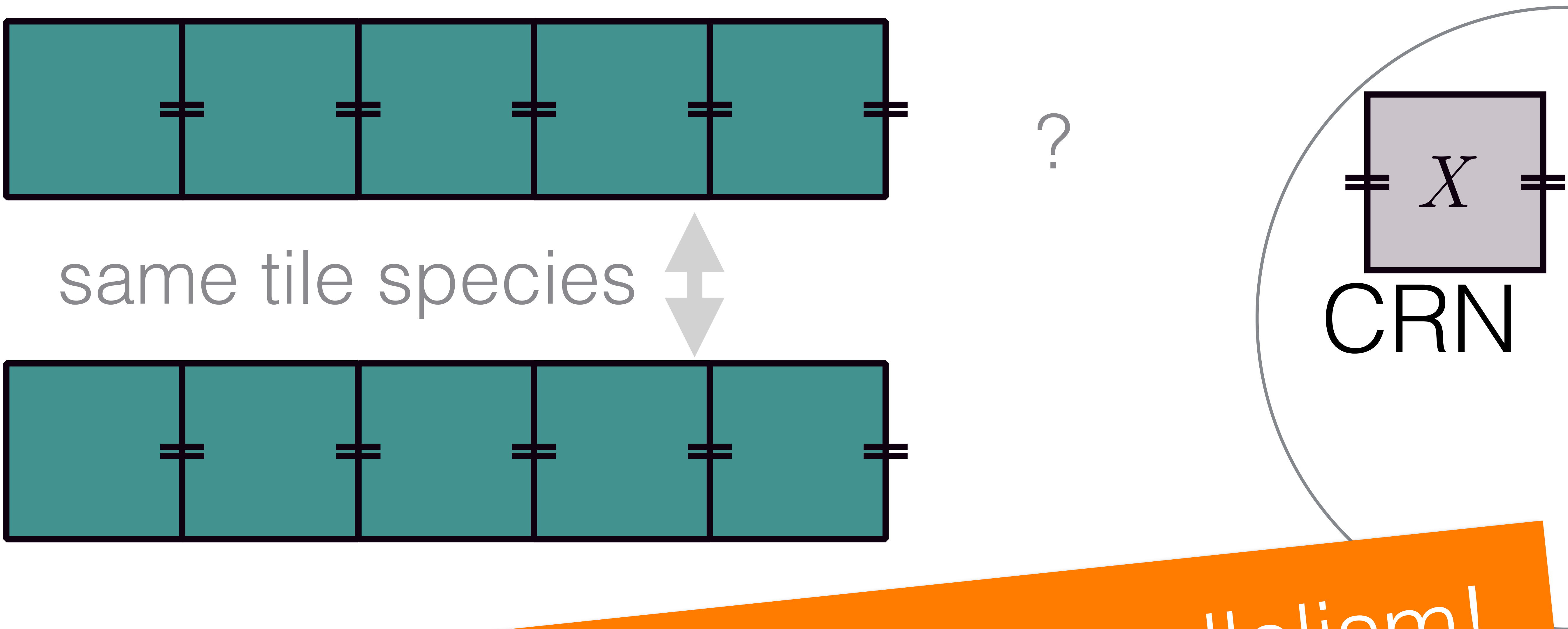
Theorem. For any shape $\mathcal{S}$, the CRN-TAM complexity of $\mathcal{S}$ at scale 2 and any temperature $\tau \geq 1$ is:

$$K^{\tau}_{\mathrm{CT}}(\mathrm{scale}_2(\mathcal{S})) \in \Theta\left(\frac{K_U(\mathcal{S})}{\log K_U(\mathcal{S})}\right)$$

Parallelism in the CRN-TAM

Previous constructions were carefully engineered to avoid parallelism!

A problem with stack machines...



Shared state = limited parallelism!

A problem with stack machines...

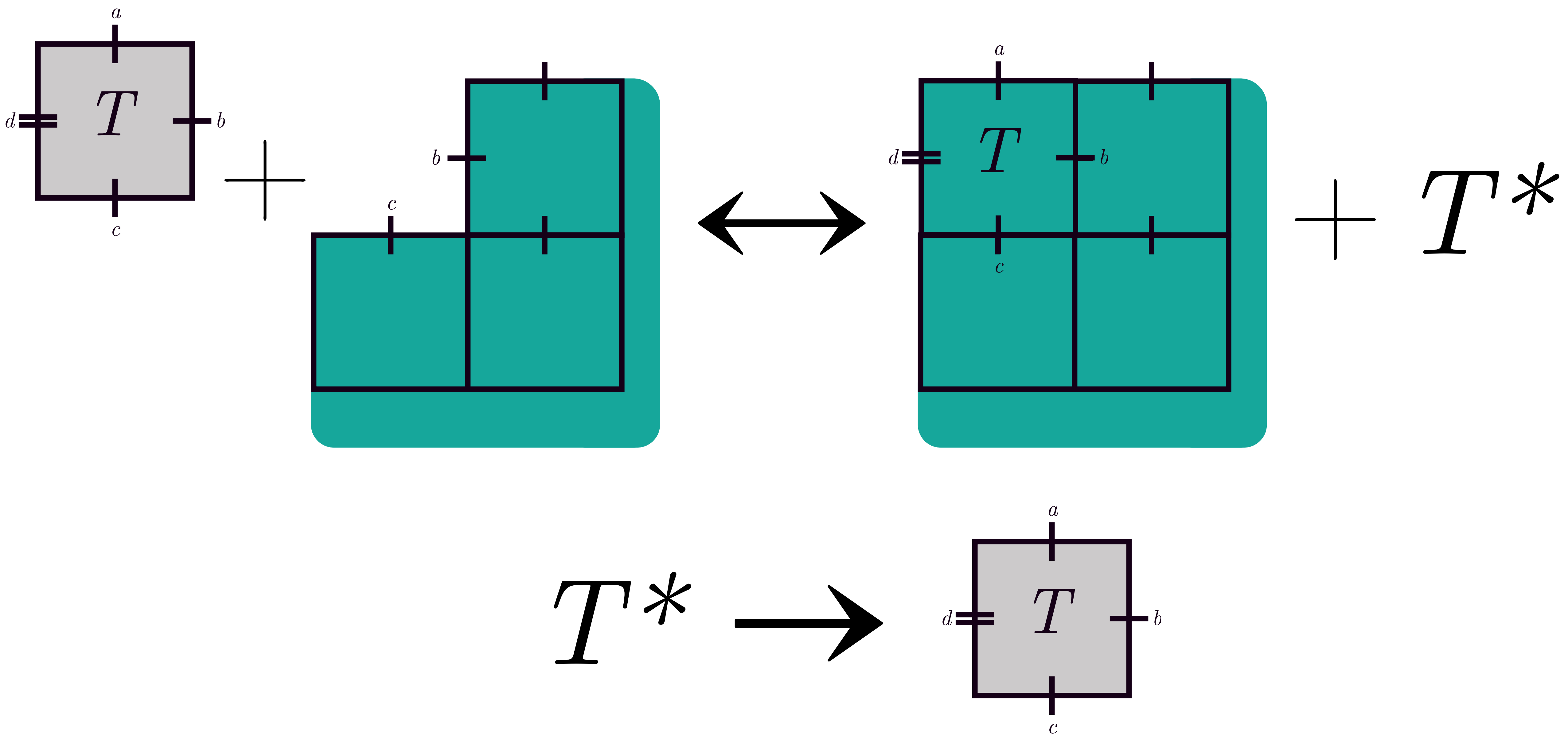
Definition. A CRN-TAM program that decides a language is *scalable* if it still works with arbitrarily many copies in the same reaction vessel.

Theorem. Every scalable CRN-TAM program uses $\Omega(t(n))$ space to simulate a Turing machine that takes time $t(n)$.

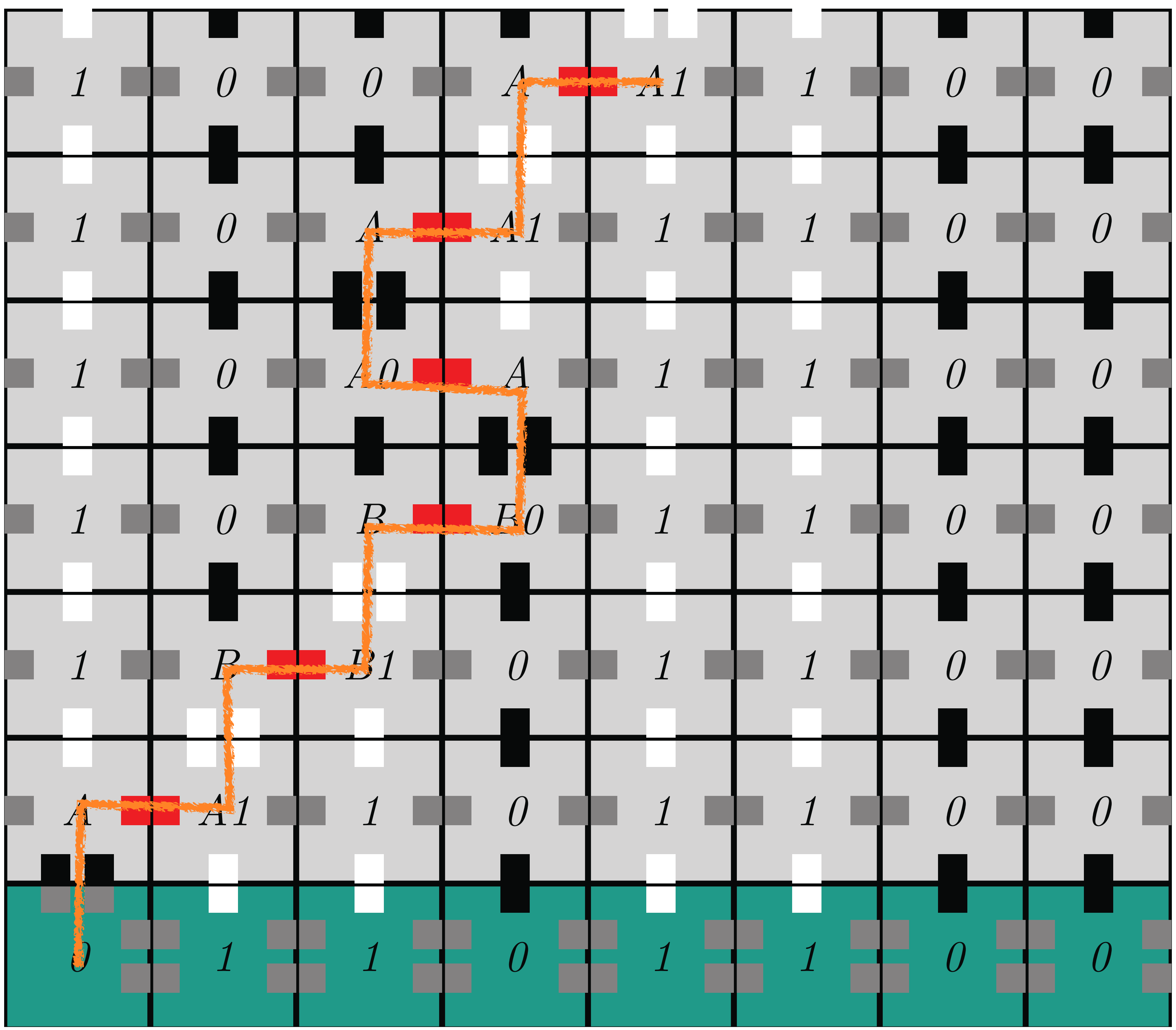
"Stack machines" are not scalable!

Computation with Tiles

Maintaining $\Theta(V)$ tiles of each type



Time Complexity of Tile Computation



Intuitively, takes as long as the "backbone" takes to build

($\Theta(t(n))$ time)

Time Complexity of Tile Computation

Seems straightforward to analyze, but…

Assembly process is asynchronous,so tiles could attach in any order!

Tile consumption and re-generation could be i … aved, so tile concentratio …

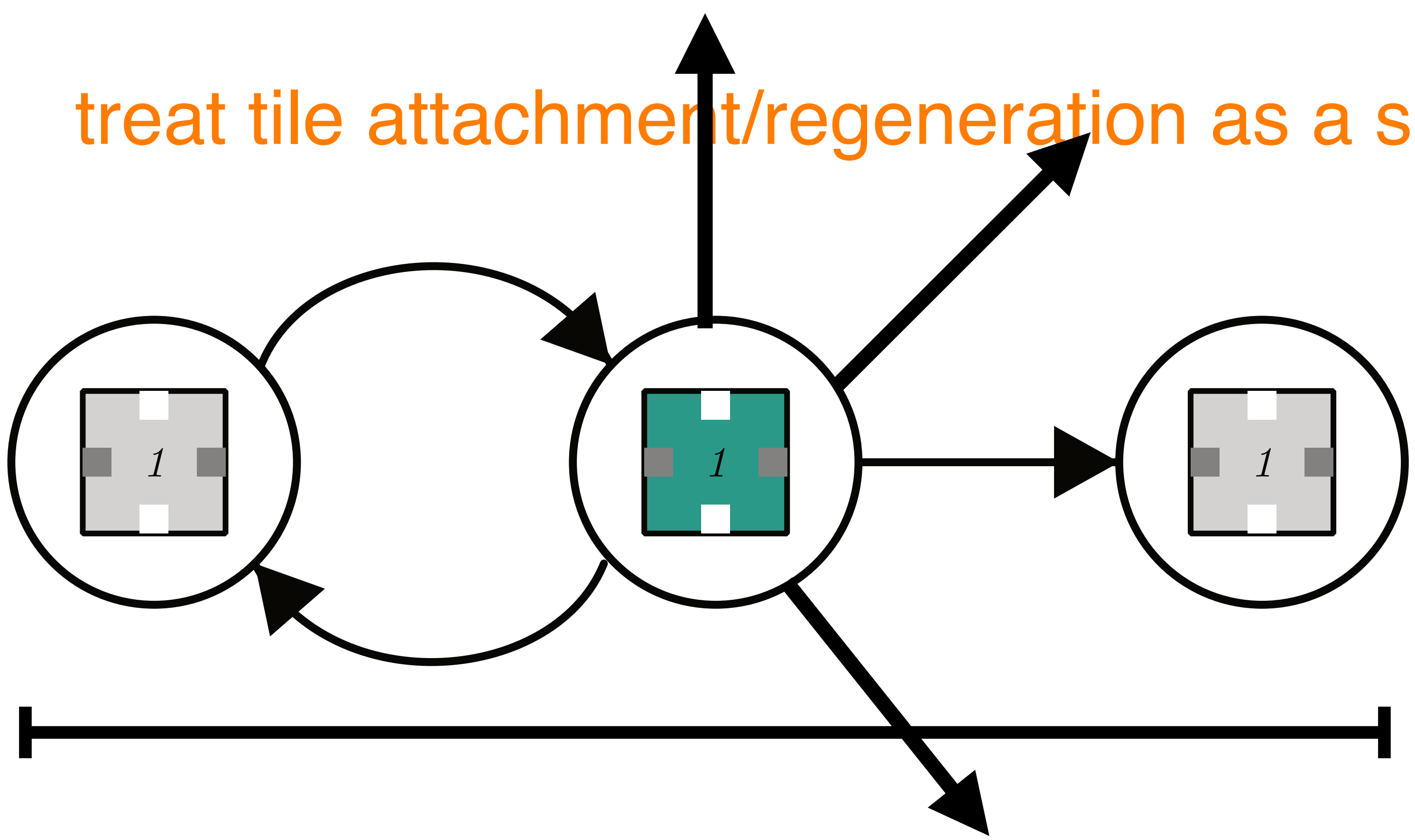
Don't … mbly to complete before knowi … answer (just the last piece of the "backbone")!

Absolute chaos!

Time Complexity of Tile Computation

Sentinel process

treat tile attachment/regeneration as a single process



transit time has phase-type distribution (well studied)

Restricting particular reactions only *increases* time complexity!

[similar to Adleman, Cheng, Goel, Huang, 2001]

Time Complexity of Tile Computation

Theorem. The sentinel process takes $\Theta(t(n))$ time.

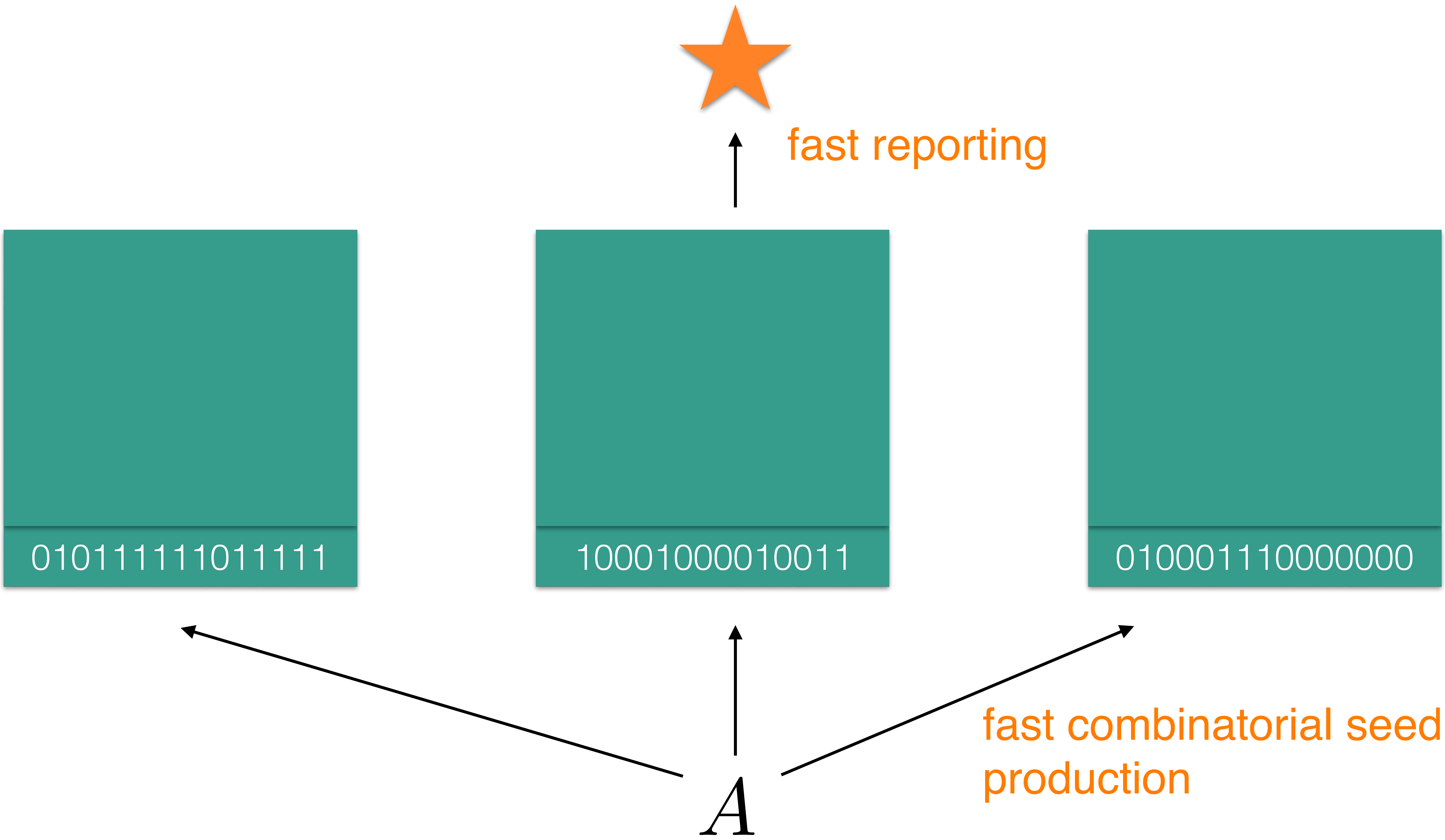
Theorem. CRN-TAM computation with tiles takes $\Theta(t(n))$ time.

[S and Winfree, 2016]

Same time as stacks!

Scalable!

Towards Parallel Computation



Limits and Open Questions

Message routing in a well-mixed CRN

A

What parallelism can/cannot be implemented in the CRN-TAM?

Three perspectives

Natural scientist

How does computation occur in the physical/biological world?

Engineer

How can use computational power in nanotechnology?

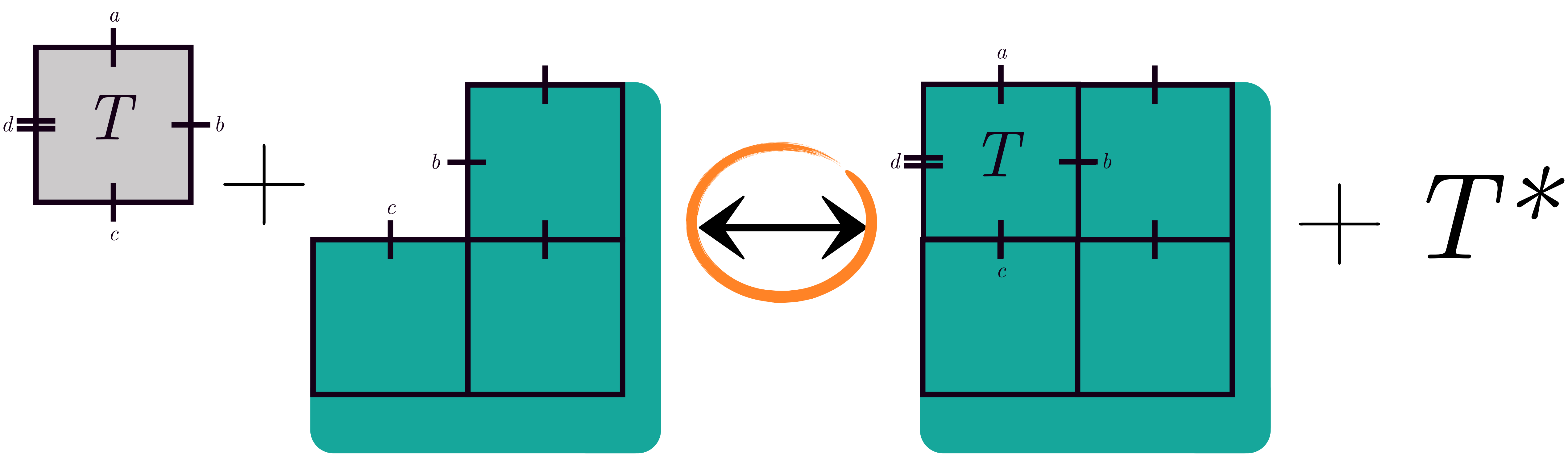
Computer scientist

How can I compute using extremely simple systems?

Thanks!

Chemical Reaction Network-Controlled Tile Assembly Model (**CRN-TAM**)

Tile attachment/detachment



"reversible" based on binding strength *b*

1) Attachment only if *b* ≥ τ
2) Detachment only if *b* = τ