Challenges in Massive Concurrency

Luca Cardelli
Microsoft Research & University of Oxford

Oxford, 2014-01-15

Outline

- Computational Models
  - The ‘massive concurrency’ of molecular soups

- Discrete-state Molecular Systems
  - Combinatorial verification of (DNA) Chemical Reaction Networks

- Continuous-state Molecular Systems
  - Morphisms of Chemical Reaction Networks that preserve kinetics

Computational Models

A computational model

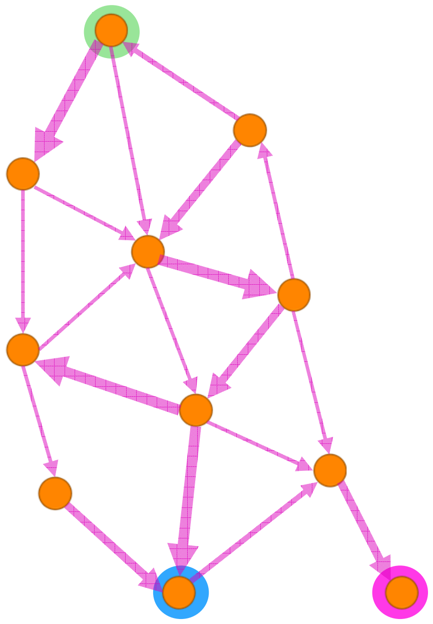
- Molecular ‘Soups’
  - Molecules randomly collide and can change state or composition.
    - Can we compute with that?
  - Based on the classical atomic theory of matter
    - probability of collision independent of location (“well-mixed” / “totally connected”)
- Related to:
  - For “small number of agents” (macroscopic systems):
    - Process Algebra, Petri Nets
  - For “large numbers of agents” (microscopic systems):
    - Population Protocols [Angluin et al.], Amorphous Computing [Abelson et al.] Swarm Intelligence – Ant Colonies, Epidemiology, Morphology, Chemistry

A notion of algorithm

- Data as populations
  - Inputs and outputs are composed of uniform *populations* of agents that do *not* have an identity
  - Algorithms emerge from the 'dumb' interactions of 'simple' agents
- In computing
  - Mostly assuming discrete or nondeterministic time
- In science and nature
  - Mostly assuming stochastic or continuous time
  - Stochastic because interactions typically correspond to random collisions or chance meetings

A mathematical model

- Continuous-Time (Discrete-Space) Markov Chains
  - Also underlies chemistry via the Chemical Master Equation (changes of probabilities of discrete states over continuous time).
  - In the limit of infinite molecules at finite concentration, it converges to the deterministic continuous-state continuous-time (ODE) model.
- NOT a probabilistic (-only) model
  - Probabilities emerge from the stochastic structure (the underlying DMC), but are not primary. We are in continuous time and we care about how long things take.
  - Non-determinism exists only in the form of 'quantitative races': who is faster is more likely to win. There is no speed-independent probability.
  - Interleaving holds by the Markov axiom: no two events ever happen at the same time.
- What can we compute in this model?



(finite or infinite)

Programming Languages

- Reaction-Based (A + B → C + D) (Chemical Reactions)
  - Finite set of species (no polymerization): finite Markov chains.
- Interaction-Based (A = !c. B) (Process Algebra)
  - Unbounded set of species: infinite Markov chains. Molecular state and identity.
  - Reduces combinatorial complexity of models by sharing *channels* between submodels.
- Rule-Based (A{-}:B{p} → A{p}:B{-}) (Logic, Graph Rewriting)
  - A *rule* is a reaction in a partially unspecified context.
  - Further reduces model complexity by abstracting over context.
  - Compatible with informal descriptions of biochemical events ("narratives").
- Relationships
  - The latter two can be translated to each other.
  - When they can be translated to the first, they may introduce an *extremely large* number of species.

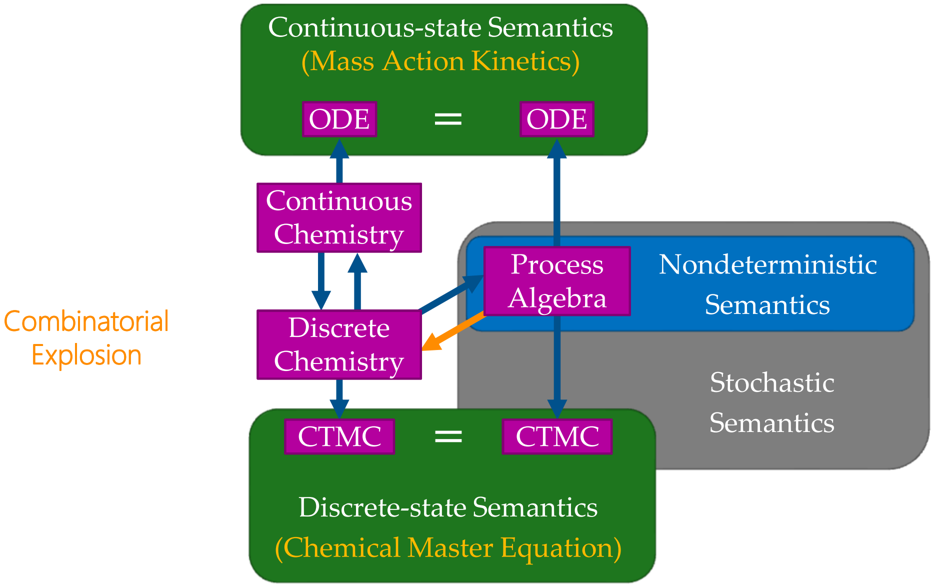
Basic Results

- The class of functions 'over individuals' that are computable
  - A finite number of chemical reactions can encode Turing machines (only) up to an arbitrarily small uniform error bound. "Approximately Turing-Complete". [1,2]
  - With polymerization, fully Turing completeness can be achieved.
  - But all these rely on 'single-molecule populations' that are difficult to achieve.
- The class of predicates 'over populations' that are 'stably computable' (population protocols)
  - Semi-linear predicates (first-order theory of (ℕ,+,<)). [3]
  - If you cannot distinguish individual molecules, you are much more restricted.

1. David Soloveichik, Matt Cook, Erik Winfree, Shuki Bruck, **Computation with Finite Stochastic Chemical Reaction Networks.** Natural Computing, 2008.
2. Luca Cardelli, Gianluigi Zavattaro. **Termination Problems in Chemical Kinetics.** CONCUR 2008.
3. Dana Angluin, James Aspnes, David Eisenstat, and Eric Ruppert. **The computational power of population protocols.** Distributed Computing, 2007.

Semantics of Chemistry (Chemical Kinetics)

- A connection with the theory of concurrency



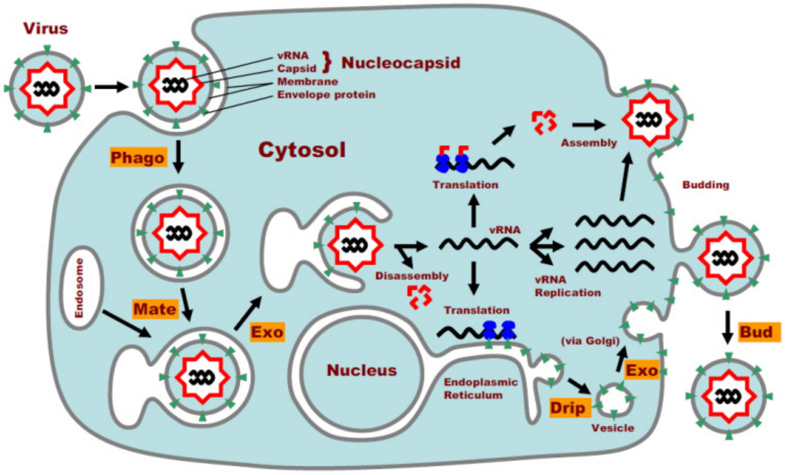
Luca Cardelli. **On Process Rate Semantics.**
Theoretical Computer Science 391(3) 190-215, Elsevier, 2008.

More Languages & Models

- Gene Networks
  - Synchronous Boolean networks
    - Stewart Kauffman, etc.
  - Asynchronous Boolean networks
    - René Thomas, etc.

- Protein Networks
  - Process Algebra (stochastic π-calculus etc.)
    - Priami, Regev-Shapiro, etc.
  - Graph Rewriting (kappa, BioNetGen etc.)
    - Danos-Laneve, Fontana & al., etc.

- Membrane Networks
  - Membrane Computing
    - Gheorghe Păun, etc.
  - Brane Calculi
    - Luca Cardelli, etc.



Challenges in Discrete-State Molecular Systems

In collaboration with:
Microsoft Biological Computation Group
U.Oxford PRISM group
U.Washington Seelig Lab

‘Writing’ Molecular Programs

- Chemistry is not a computational science
  - We can read (nature’s) molecular programs, but *we* cannot write them (in general)!
  - We cannot find molecules that do whatever *we* want them to do!
- But we can fake it (encode it)
  - Find some ‘universal molecules’ that *we* can build, and that can *do* what *all* other molecules, real or hypothetical, can do.
  - Ok, not quite ‘*do*’, but ‘*behave like*’ any other molecules.
- With DNA
  - These are molecules we can read *and write*! The folding problem for DNA/RNA is solvable, and they can be produced on industrial scale.

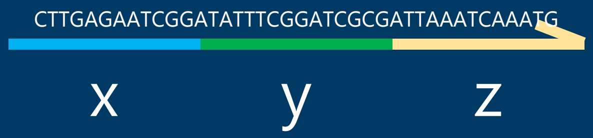
Soloveichik, D., Seelig, G., Winfree, E., **DNA as a Universal Substrate for Chemical Kinetics.** PNAS, 2010.

Why write molecular programs?

- Non-goals
  - Not to solve NP-complete problems with large vats of chemicals
    (even massive concurrency does not help!)
  - Not to replace silicon-based technology
    DNA is slow(er), but compatible with life processes
- Bootstrapping a programmable carbon-based technology
  - To precisely control the organization and dynamics
    of matter and information at the molecular level
    - Nanotechnology
    - Medicine and Biology
  - DNA is “just” the most convenient material for the task
    - It is an information-bearing programmable material;
      other such materials are actively being developed

Domains

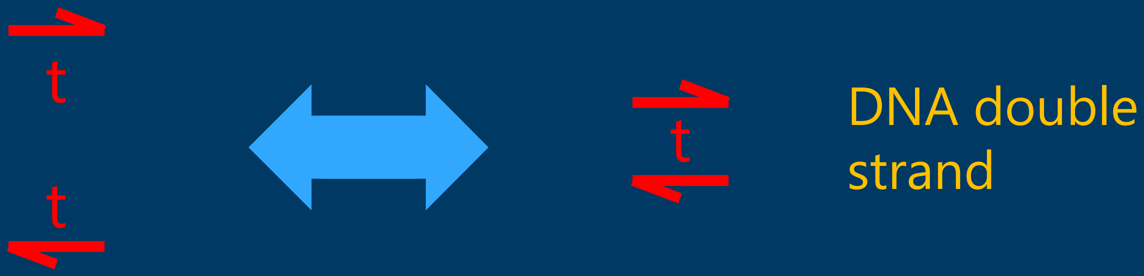
- Subsequences on a DNA strand are called domains
  - *provided* they are "independent" of each other



oriented DNA
single strand

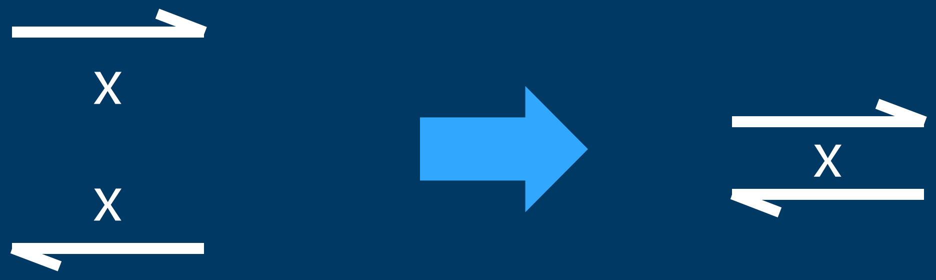
- Differently named domains must not hybridize
  - With each other, with each other's complement, with subsequences of each other, with concatenations of other domains (or their complements), etc.

Short Domains



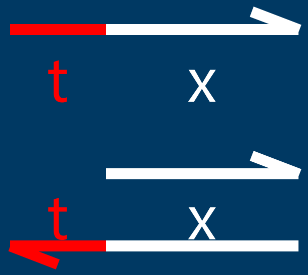
Reversible Hybridization

Long Domains



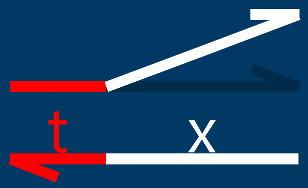
Irreversible Hybridization

Strand Displacement



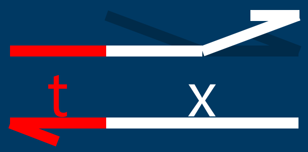
"Toehold Mediated"

Strand Displacement



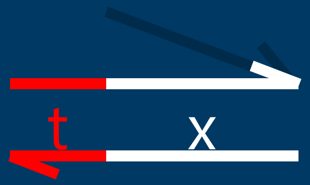
Toehold Binding

Strand Displacement



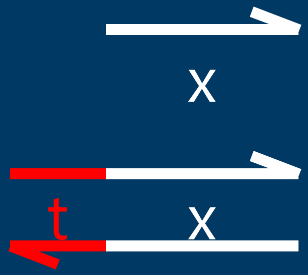
Branch Migration

Strand Displacement



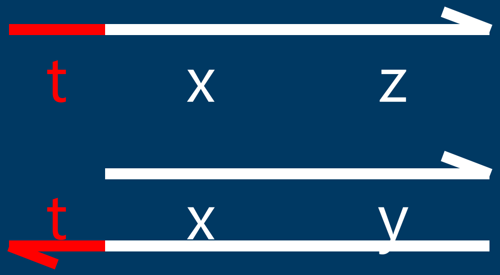
Displacement

Strand Displacement

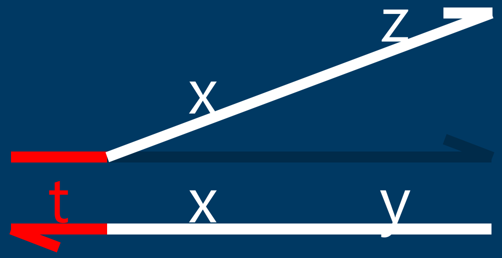


Irreversible release

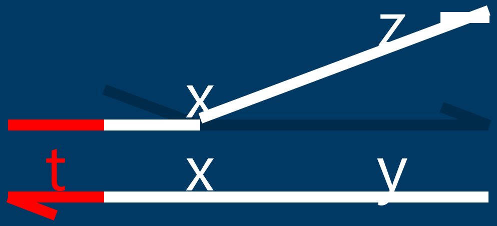
Bad Match



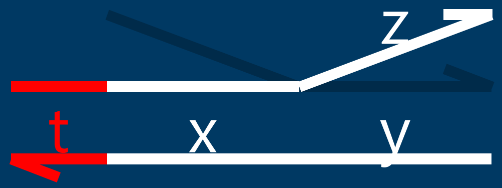
Bad Match



Bad Match



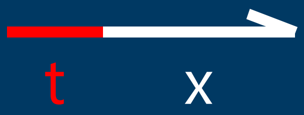
Bad Match



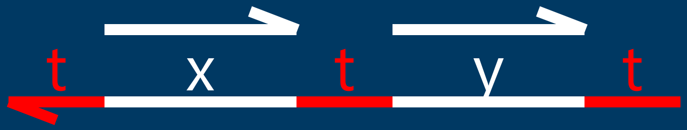
Cannot proceed
Hence will undo

Two-Domain Architecture

- Signals: 1 toehold + 1 recognition region



- Gates: "top-nicked double strands" with open toeholds



Garbage collection "built into" the gate operation

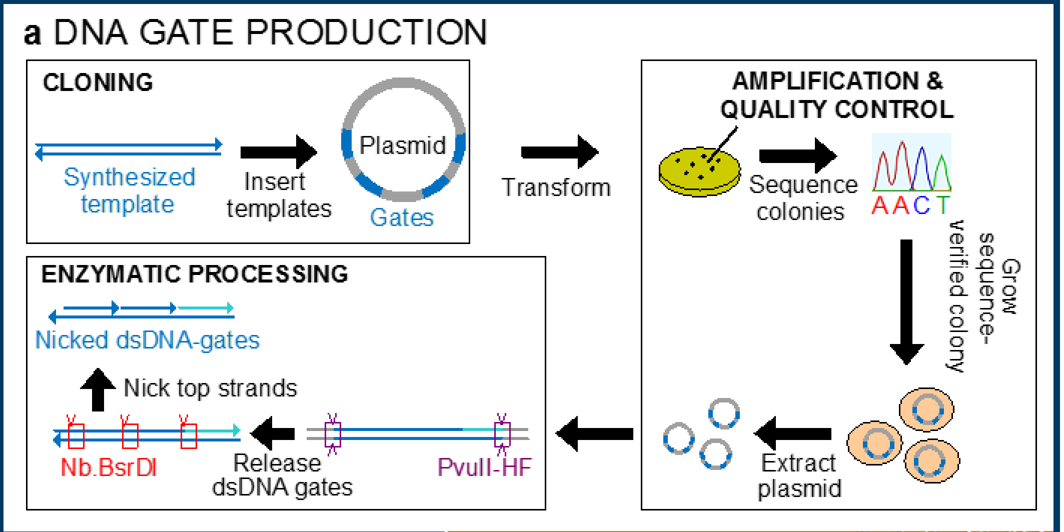
Two-Domain DNA Strand Displacement

Luca Cardelli

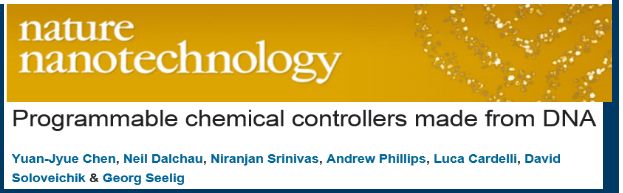
In S. B. Cooper, E. Kashefi, P. Panangaden (Eds.):
Developments in Computational Models (DCM 2010).
EPTCS 25, 2010, pp. 33-47. May 2010.

Plasmidic Gate Technology

- Synthetic DNA is length-limited
  - Finite error probability at each nucleotide addition, hence ~ 200nt max
- Bacteria can replicate plasmids for us
  - Loops of DNA 1000's nt, with extremely high fidelity
  - Practically no structural limitations on fan-in/fan-out



Only possible with two-domain architecture

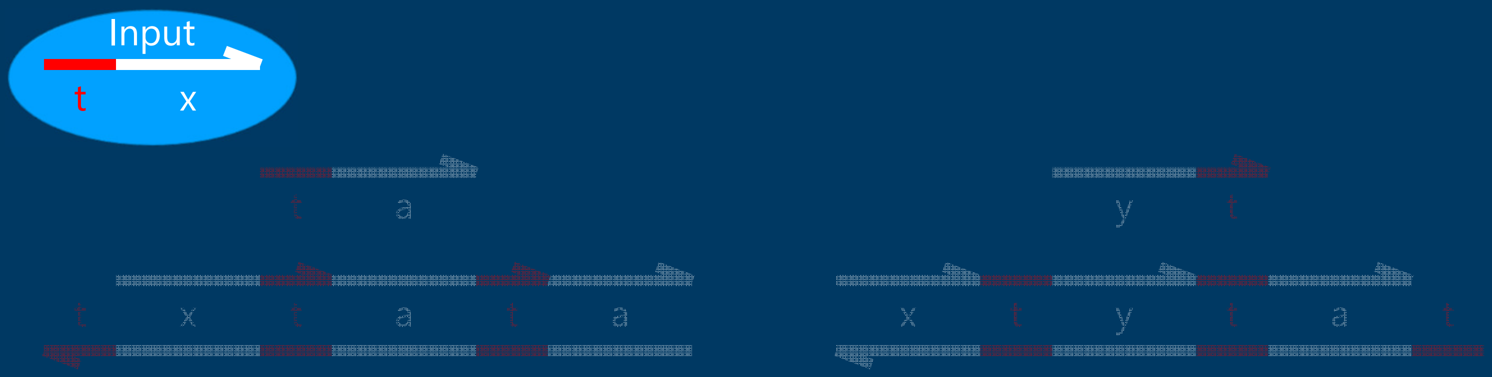

nature nanotechnology

Programmable chemical controllers made from DNA

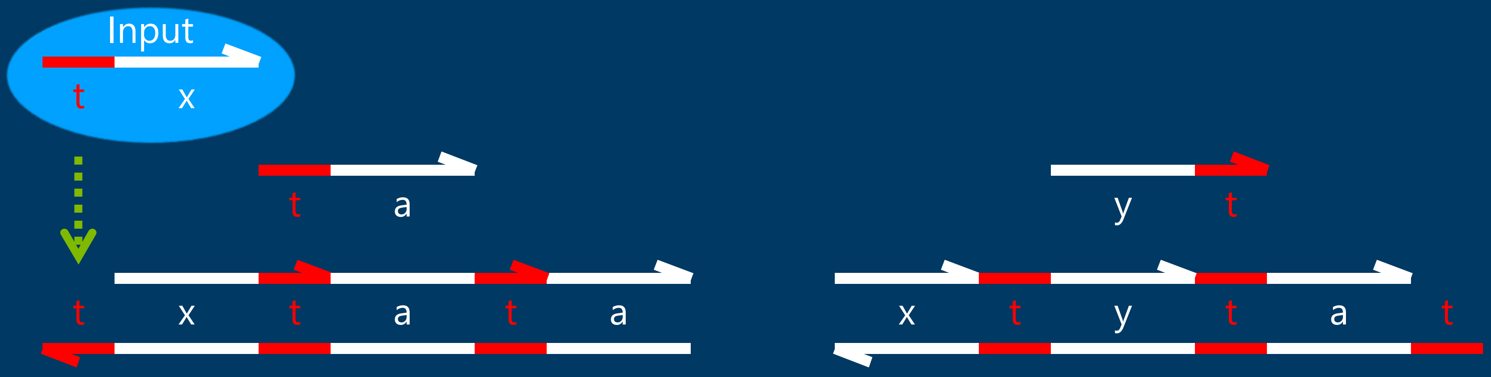
Yuan-Jyue Chen, Neil Dalchau, Niranjan Srinivas, Andrew Phillips, Luca Cardelli, David Soloveichik & Georg Seelig

Transducer

Transducer x⟶y



Transducer x→y

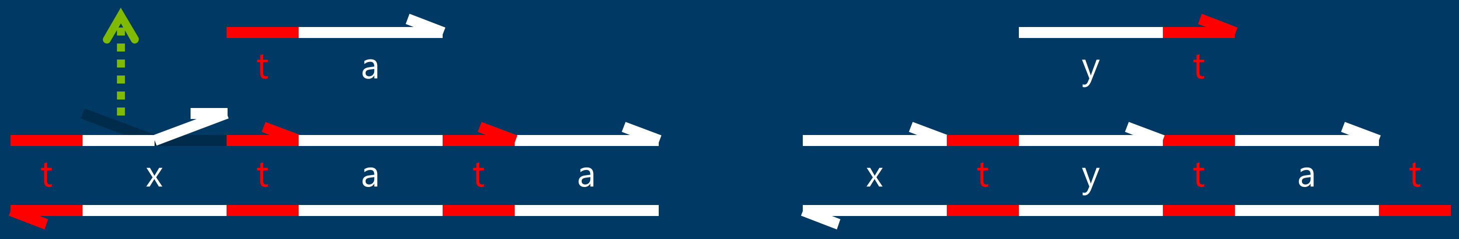


Join half

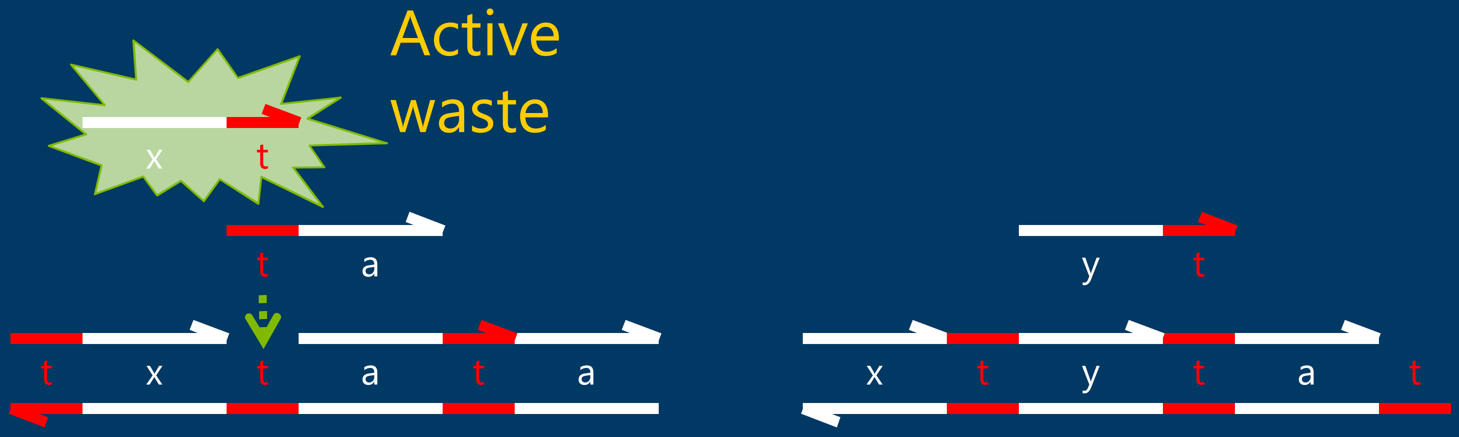
Fork half

ta is a *private* signal (a different ‘a’ for each xy pair)

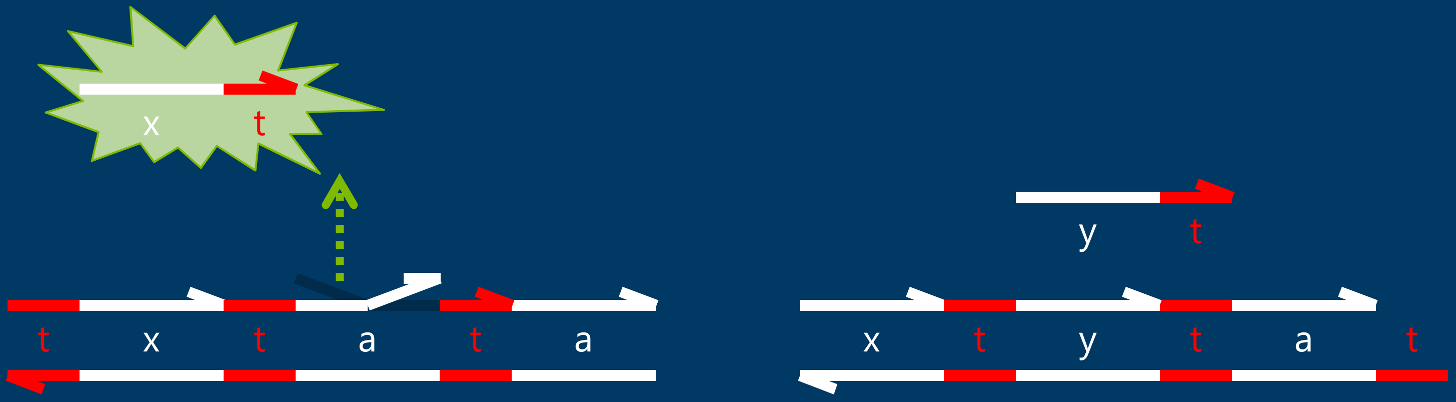
Transducer x→y



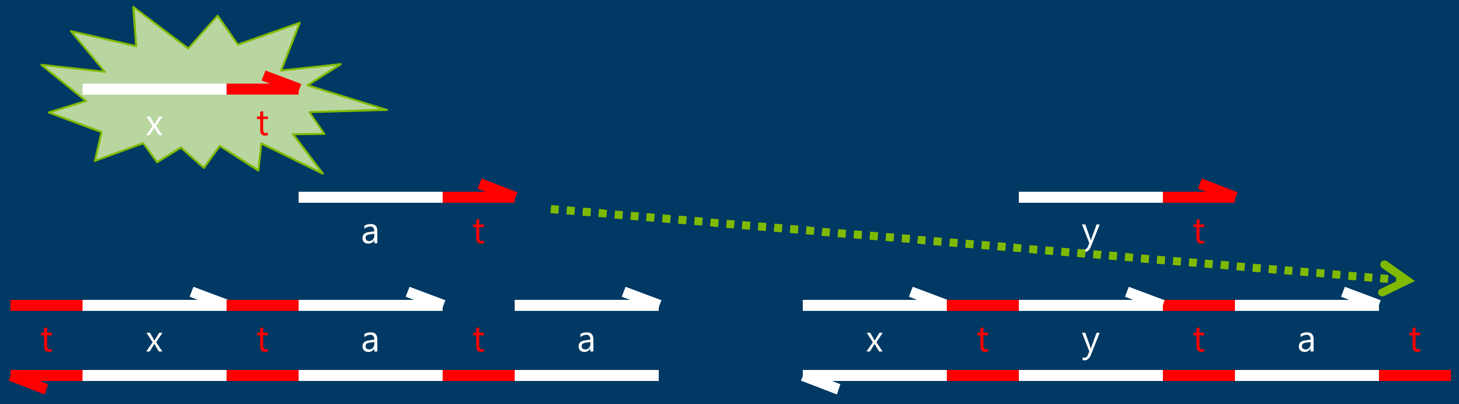
Transducer x→y



Transducer x→y

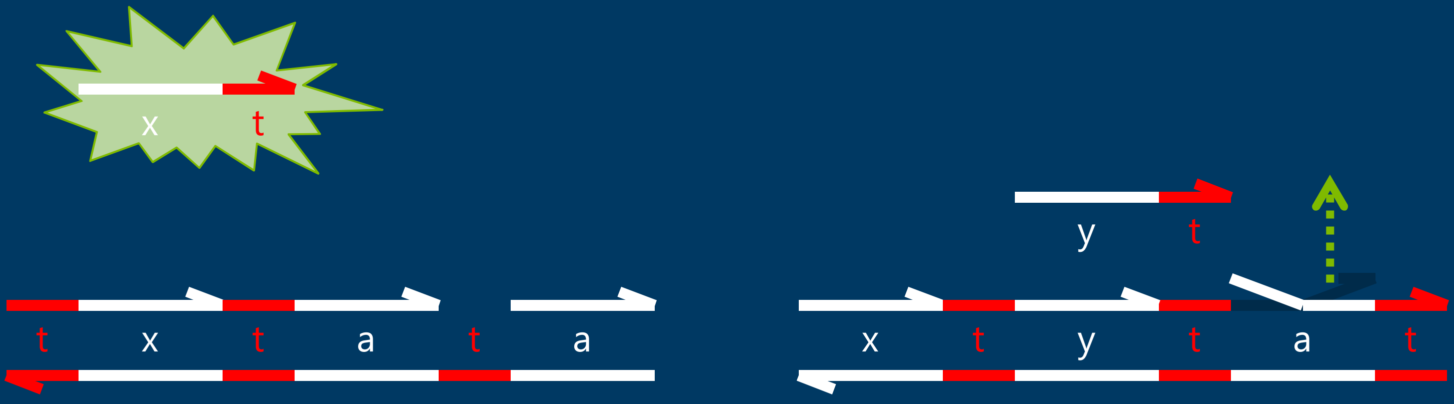


Transducer x→y

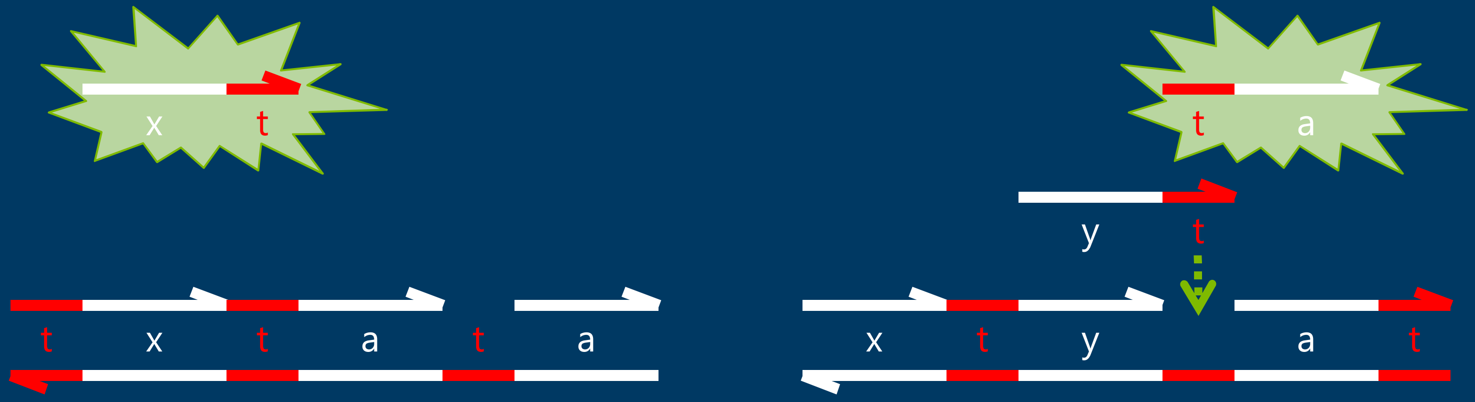


So far, a **tx** *signal* has produced an **at** *cosignal*.
But we want signals as output, not cosignals.

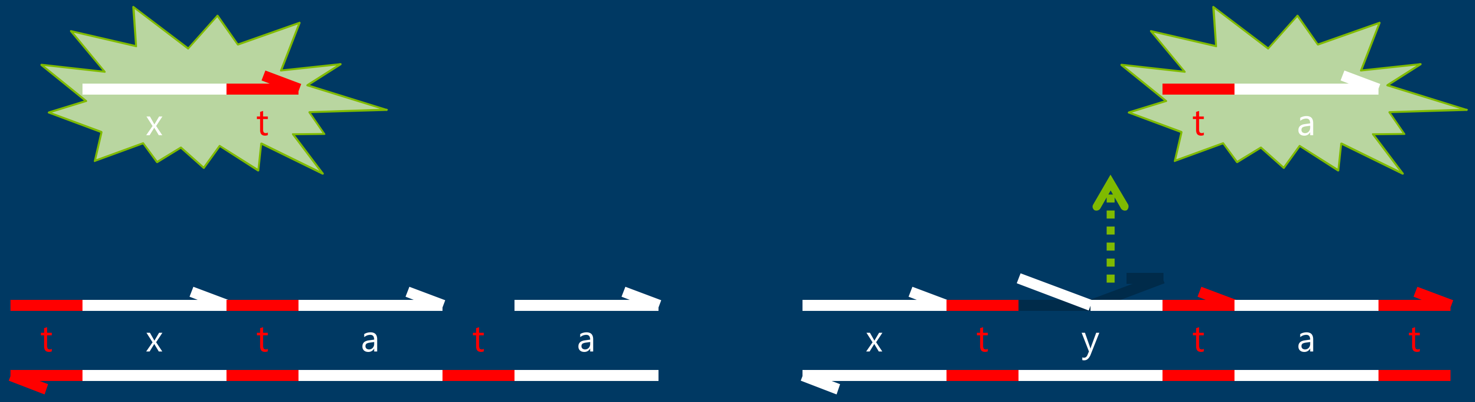
Transducer x⟶y



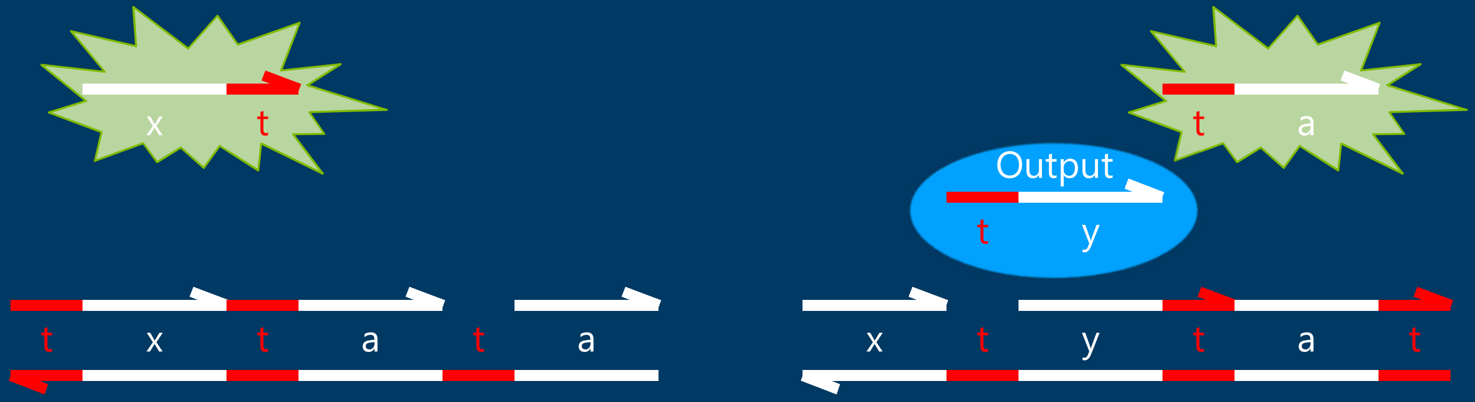
Transducer x⟶y



Transducer x→y



Transducer x⟶y

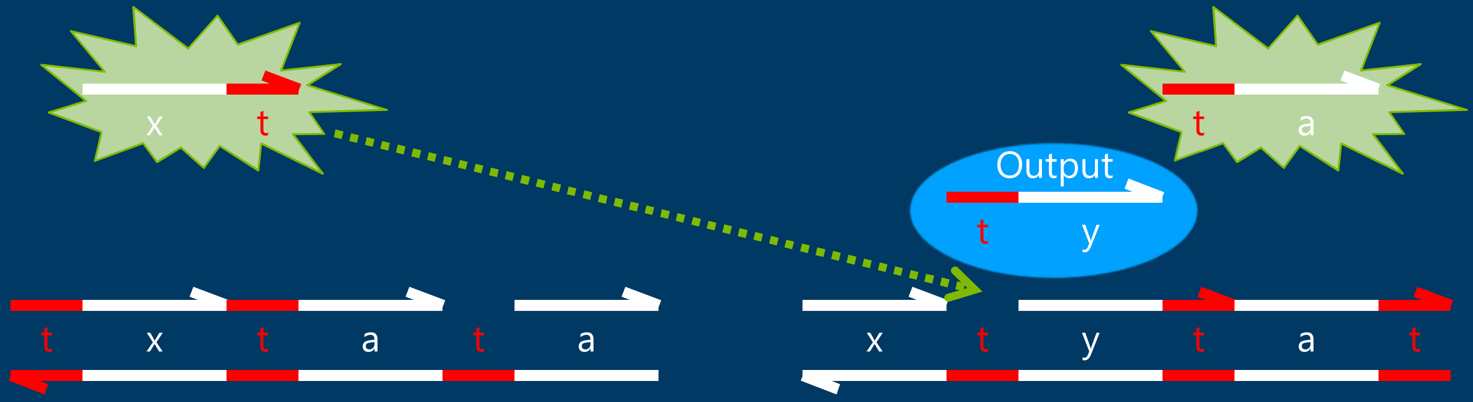


Here is our output **ty** *signal*.
But we are not done yet:

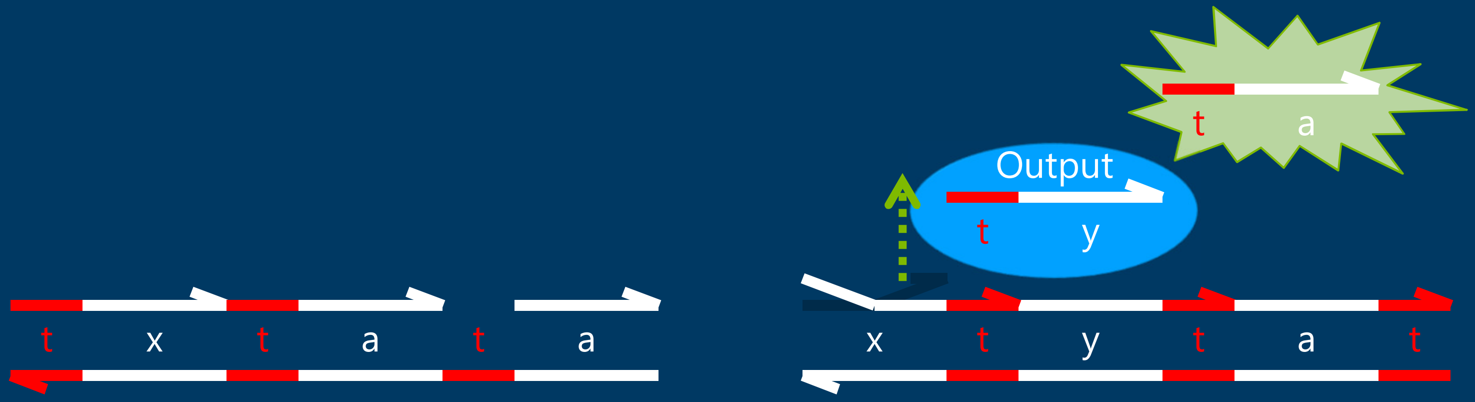
1) We need to make the output irreversible.
2) We need to remove the garbage.

We can use (2) to achieve (1).

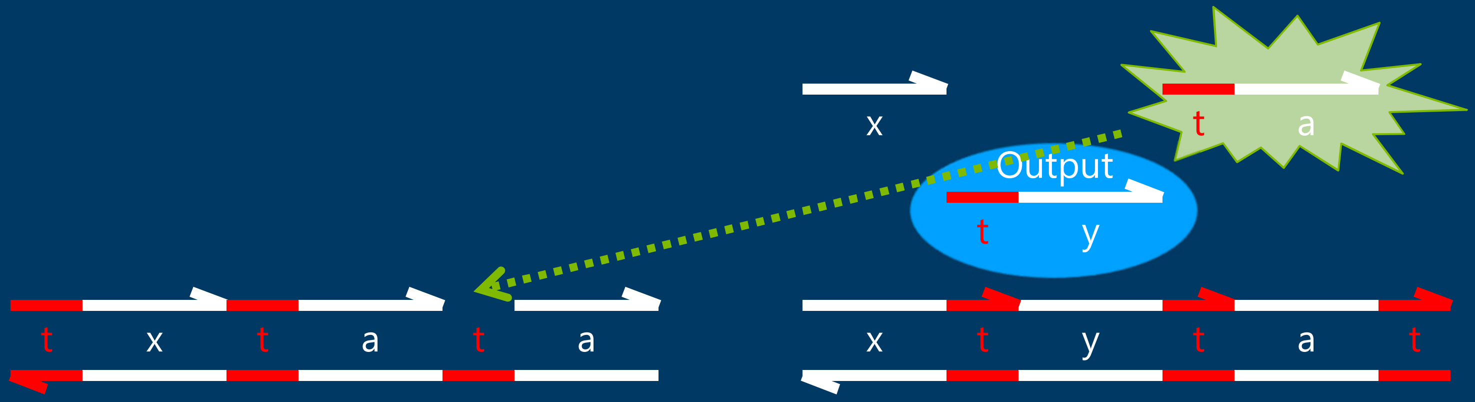
Transducer x→y



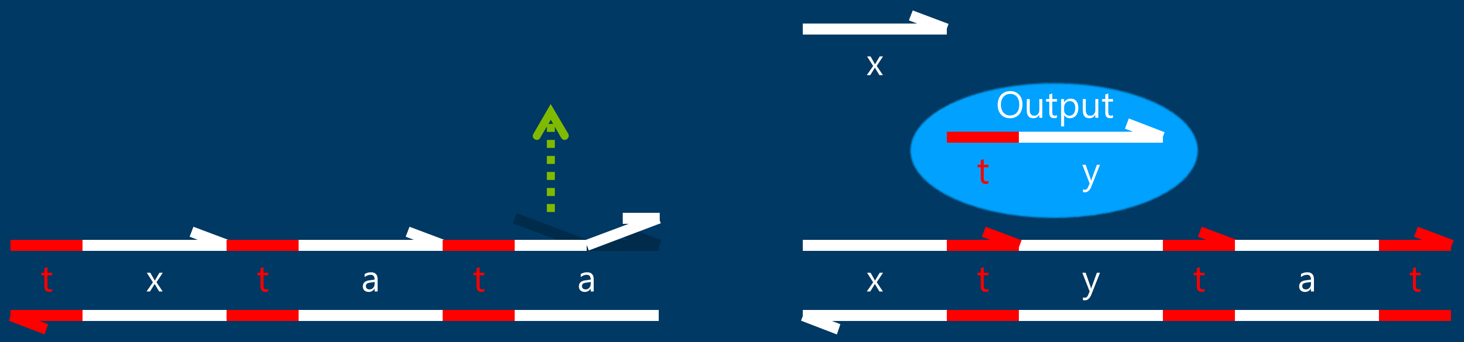
Transducer x→y



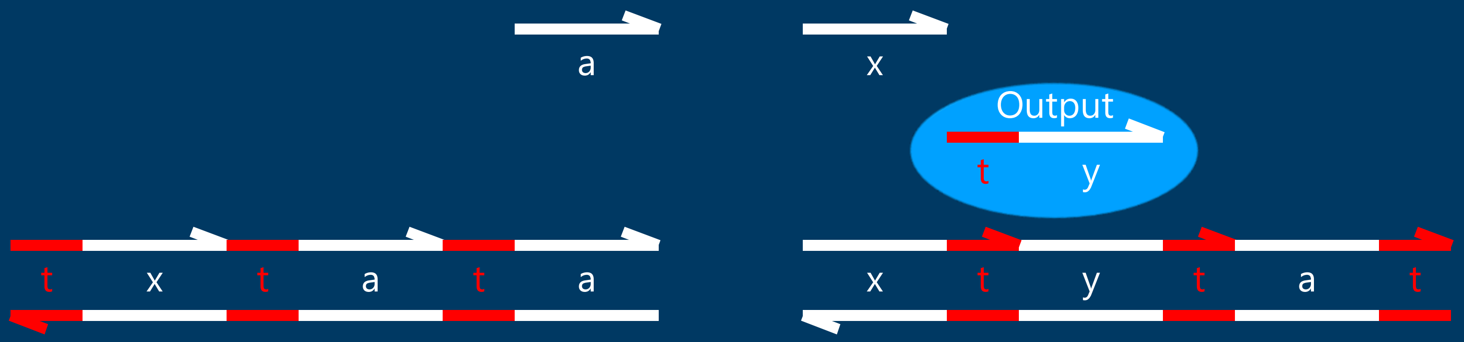
Transducer x→y



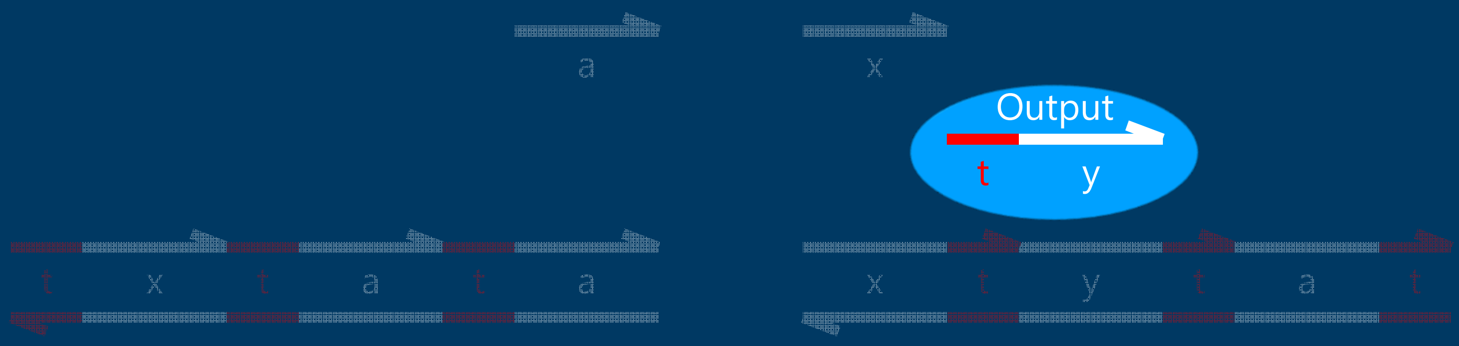
Transducer x⟶y



Transducer x→y



Transducer x→y



Done.

N.B. the gate is consumed: it is the energy source

(no proteins, no enzymes, no heat-cycling, etc.; just DNA in salty water)

General n×m Join-Fork = $A_1+\ldots+A_n \rightarrow B_1+\ldots+B_m$

- Easily generalized to 2+ inputs (with 1+ collectors).
- Easily generalized to 2+ outputs.

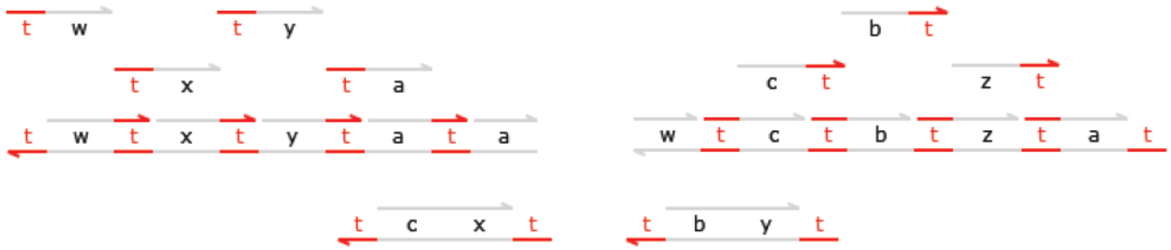
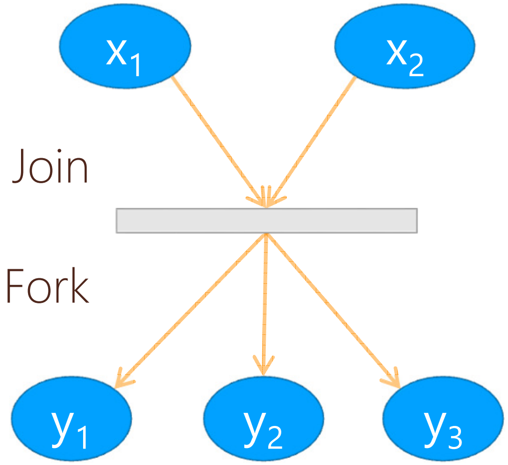


Figure 9: 3-Join J_{wxyz} | tw | tx | $ty \longrightarrow tz$: initial state plus inputs tw, tx, ty.

With that, we can ‘implement chemistry’

- That is, we can implement *arbitrary* chemistry …
  - … by using *specific* (DNA) chemistry
  - … up to an equivalence (same approximate kinetics, up to time dilation)
- Computing power equivalent to Stochastic Petri Nets
  - Not Turing complete, but as good as chemistry itself.
  - The correspondence is not completely trivial: gates are consumed by activation, hence a persistent Petri net transition requires a stable population of gates.
  - Many other mechanisms are expressible with Petri Nets like Boolean networks and state machines



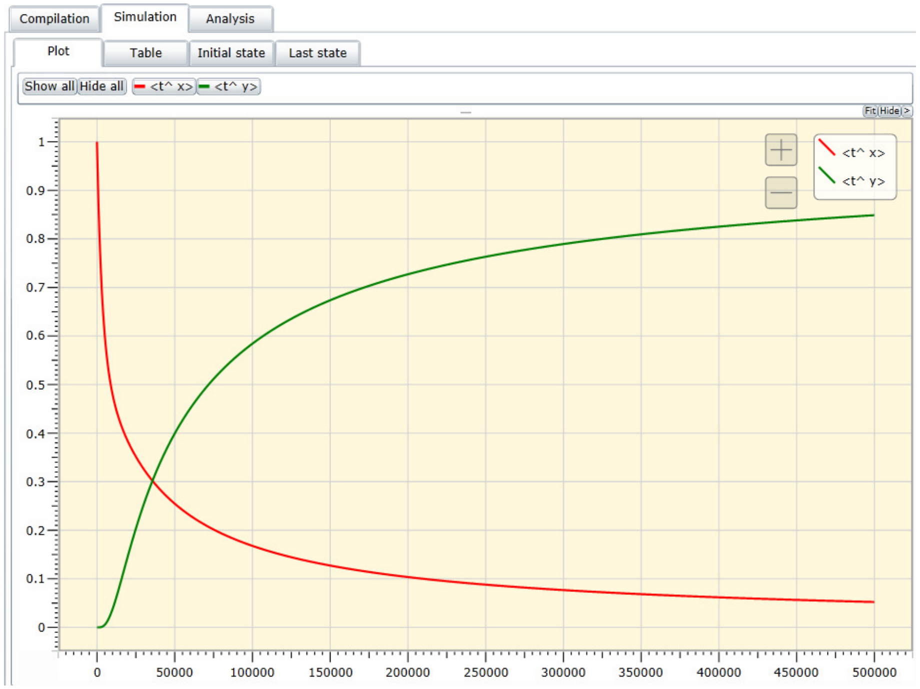
Challenges of Correct Design: Proofs

- Does the two-domain architecture correctly implement Stochastic Petri Nets (and chemistry)?
  - A rather difficult problem (which I left open). By modelchecking we can verify specific constructions, but only for limited range of inputs.
  - This was only recently settled using techniques form the theory of concurrency (serializability):

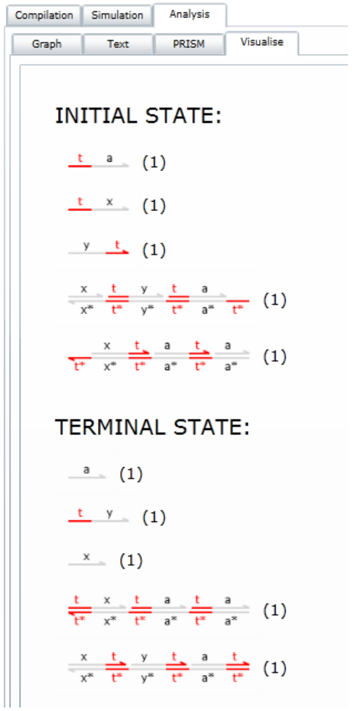
Matthew R. Lakin, Andrew Phillips, and Darko Stefanovic, Modular verification of DNA strand displacement networks via serializability analysis, in *International Conference on DNA Computing and Molecular Programming*, Springer Verlag, September 2013

Simulation

- Stochastic
- Deterministic

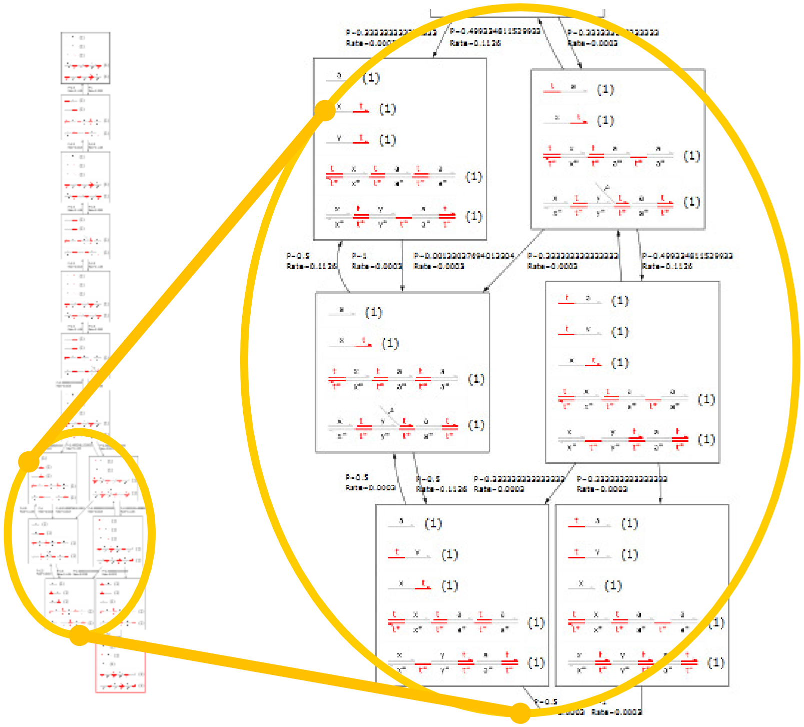


State Space Analysis



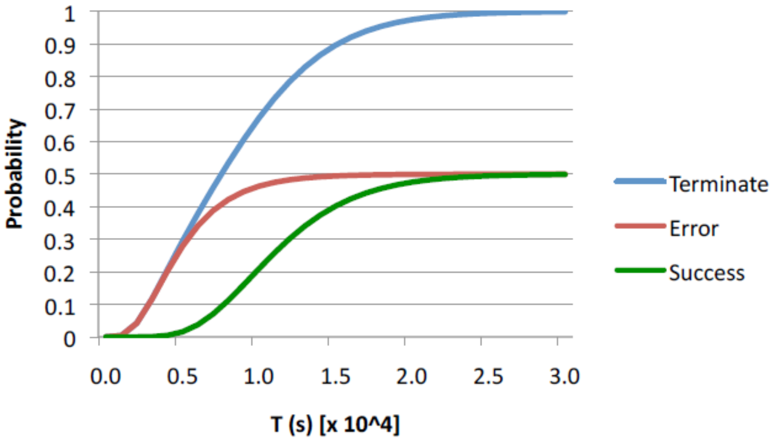
INITIAL STATE:

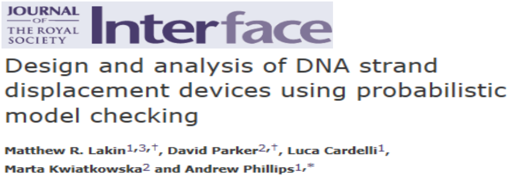
TERMINAL STATE:



Modelchecking

- PRISM probabilistic modelchecker





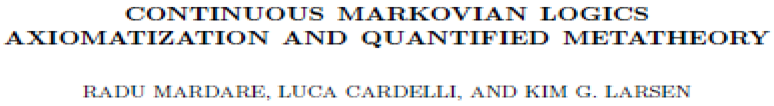
JOURNAL OF THE ROYAL SOCIETY Interface

Design and analysis of DNA strand displacement devices using probabilistic model checking

Matthew R. Lakin[1,3,†], David Parker[2,†], Luca Cardelli[1], Marta Kwiatkowska[2] and Andrew Phillips[1,*]

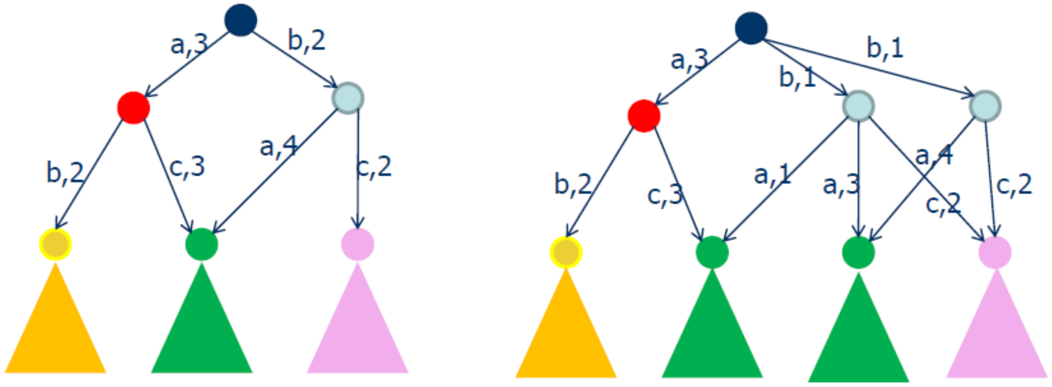
Verification

- Quantitative theories of system equivalence and approximation.



CONTINUOUS MARKOVIAN LOGICS
AXIOMATIZATION AND QUANTIFIED METATHEORY

RADU MARDARE, LUCA CARDELLI, AND KIM G. LARSEN

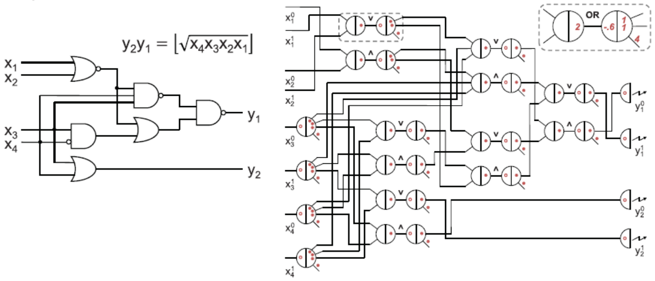


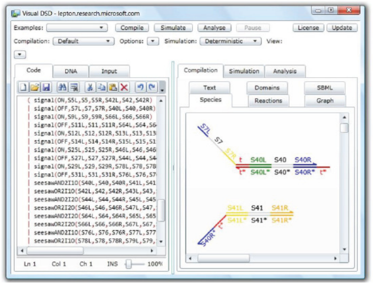
Scaling Up DNA Circuits

- Can verification catch up?

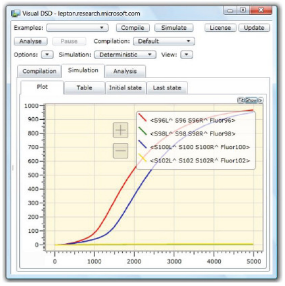
Scaling Up Digital Circuit Computation with DNA Strand Displacement Cascades

Lulu Qian[1] and Erik Winfree[1,2,3*]





```
( signal(ON,S5L,S5,S5R,S42L,S42,S42R)
| signal(OFF,S7L,S7,S7R,S40L,S40,S40R)
| signal(ON,S9L,S9,S9R,S66L,S66,S66R)
| signal(OFF,S11L,S11,S11R,S64L,S64,S6
| signal(ON,S12L,S12,S12R,S13L,S13,S13
| signal(OFF,S14L,S14,S14R,S15L,S15,S1
| signal(ON,S25L,S25,S25R,S46L,S46,S46
| signal(OFF,S27L,S27,S27R,S44L,S44,S4
| signal(ON,S29L,S29,S29R,S78L,S78,S78
| signal(OFF,S31L,S31,S31R,S76L,S76,S7
| seesawAND2I1O(S40L,S40,S40R,S41L,S41
| seesawOR2I1O(S42L,S42,S42R,S43L,S43,
| seesawAND2I2O(S44L,S44,S44R,S45L,S45
| seesawOR2I2O(S46L,S46,S46R,S47L,S47,
| seesawAND2I2O(S64L,S64,S64R,S65L,S65
| seesawOR2I2O(S66L,S66,S66R,S67L,S67,
| seesawAND2I1O(S76L,S76,S76R,S77L,S77
| seesawOR2I1O(S78L,S78,S78R,S79L,S79,
```



3 JUNE 2011 VOL 332 **SCIENCE**

Scaling Up DNA Computation

John H. Reif

“In addition to biochemistry laboratory techniques, computer science techniques were essential.”

“Computer simulations of seesaw gate circuitry optimized the design and correlated experimental data.”

3 JUNE 2011 VOL 332 **SCIENCE**

Challenges
in Continuous-State
Molecular Systems

In collaboration with:
Attila Csikász-Nagy
and thanks to David Soloveichik

Networks

- Informal ideas in Biology
  - Usually communicated by some kind of network or graph
  - These networks are often at best ambiguous [Kitano]
- Many kinds of networks, including:
  - Chemical Reaction Networks (species A becomes species B and C)
  - Influence Networks (species A promotes or inhibits species B)
- Networks convey meaning
  - Can network relationships convey meaning too?

Mutual Inhibition

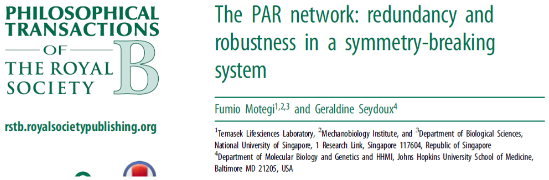
- A recent paper suggests that all cellular switches in all phases of the cell cycle follow (abstractly) a mutual inhibition pattern:

Molecular mechanisms creating bistable switches at cell cycle transitions

Anael Verdugo, P. K. Vinod, John J. Tyson and Bela Novak

Open Biol. 2013 **3**, 120179, published 13 March 2013

- Also found in other areas (cell polarity establishment):



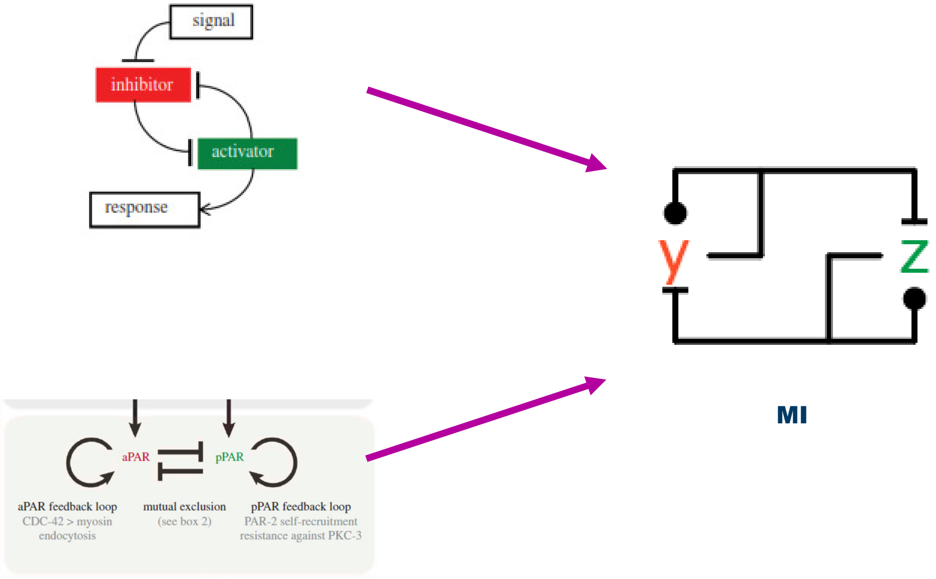
PHILOSOPHICAL TRANSACTIONS OF THE ROYAL SOCIETY B

rstb.royalsocietypublishing.org

The PAR network: redundancy and robustness in a symmetry-breaking system

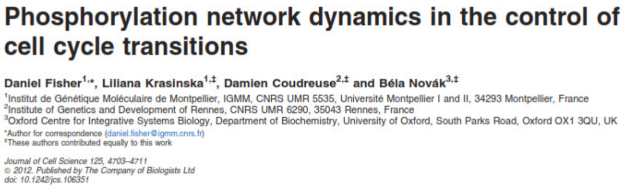
Fumio Motegi[1,2,3] and Geraldine Seydoux[4]

[1]Temasek Lifesciences Laboratory, [2]Mechanobiology Institute, and [3]Department of Biological Sciences, National University of Singapore, 1 Research Link, Singapore 117604, Republic of Singapore
[4]Department of Molecular Biology and Genetics and HHMI, Johns Hopkins University School of Medicine, Baltimore MD 21205, USA



Cell Cycle Switch Network

- A recent paper presents a more complete view of the classical cell cycle switch

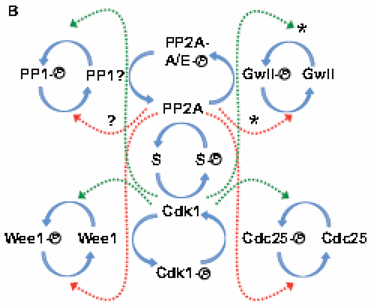


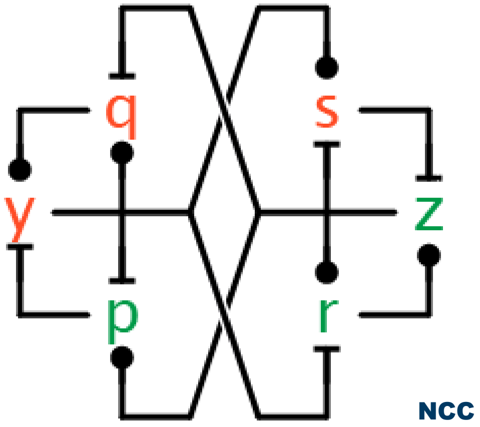
Phosphorylation network dynamics in the control of cell cycle transitions

Daniel Fisher[1,*], Liliana Krasinska[1,‡], Damien Coudreuse[2,‡] and Béla Novák[3,‡]

[1] Institut de Génétique Moléculaire de Montpellier, IGMM, CNRS UMR 5535, Université Montpellier I and II, 34293 Montpellier, France
[2] Institute of Genetics and Development of Rennes, CNRS UMR 6290, 35043 Rennes, France
[3] Oxford Centre for Integrative Systems Biology, Department of Biochemistry, University of Oxford, South Parks Road, Oxford OX1 3QU, UK
[*] Author for correspondence (daniel.fisher@igmm.cnrs.fr)
[‡] These authors contributed equally to this work

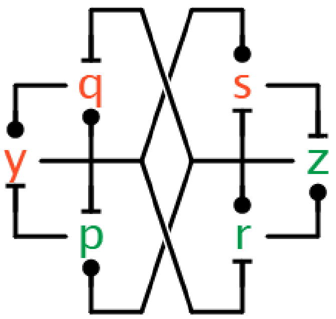
Journal of Cell Science 125, 4703–4711
© 2012. Published by The Company of Biologists Ltd
doi: 10.1242/jcs.106351

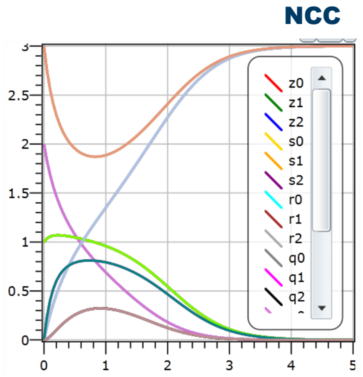




Network Emulation: NCC to MI

- For *any* initial state of MI we can find *some* initial state of NCC (actually by *copying* the state of MI) such that NCC *exactly* emulates MI

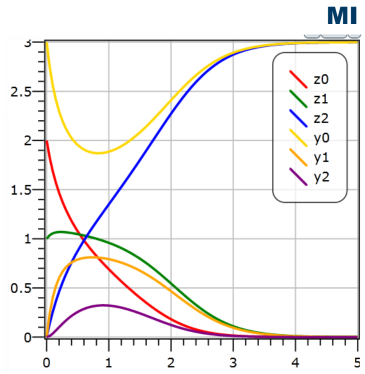




(18 species on 6 trajectories)

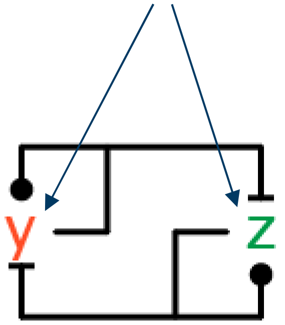
z,r,p ⇢ z

y,q,s ⇢ y



(6 species on 6 trajectories)

(3 species each)



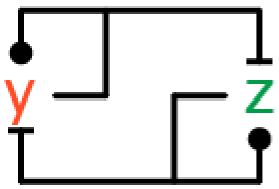
initialize z,r,p,
identically to z;

initialize y,q,s
identically to y

- Why does this work so well?

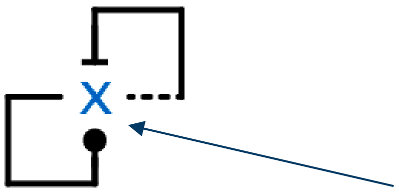
Network Emulation: MI to AM

- For chosen initial conditions of MI, the (6) trajectories of MI emulate those (3) of AM:



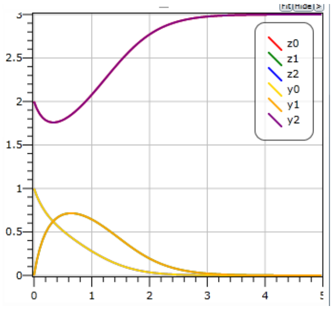
MI

~y,z⇢ x

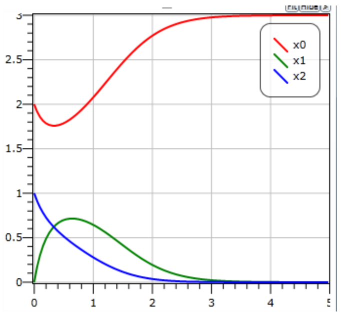


(3 species)

AM



(6 species on 3 trajectories)

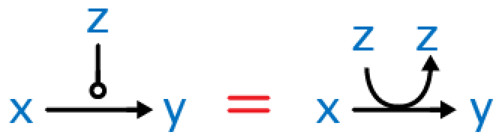


(3 species on 3 trajectories)

initialize ~y,z,
identically to x

Influence Network Notation

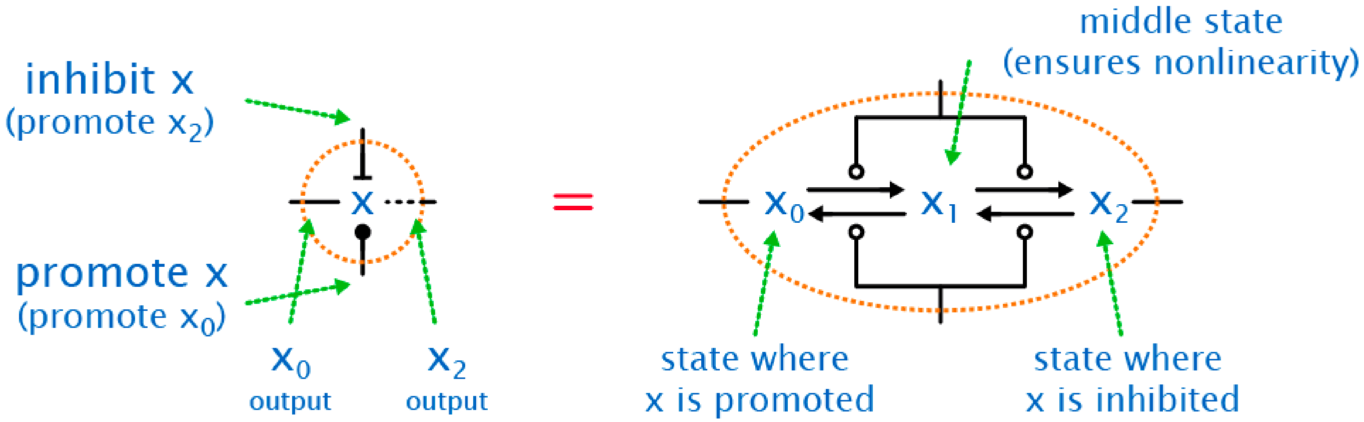
- Catalytic reaction



z is the catalyst

$x + z \rightarrow z + y$

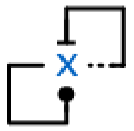
- ‘Double kinase-phosphatase’ motif



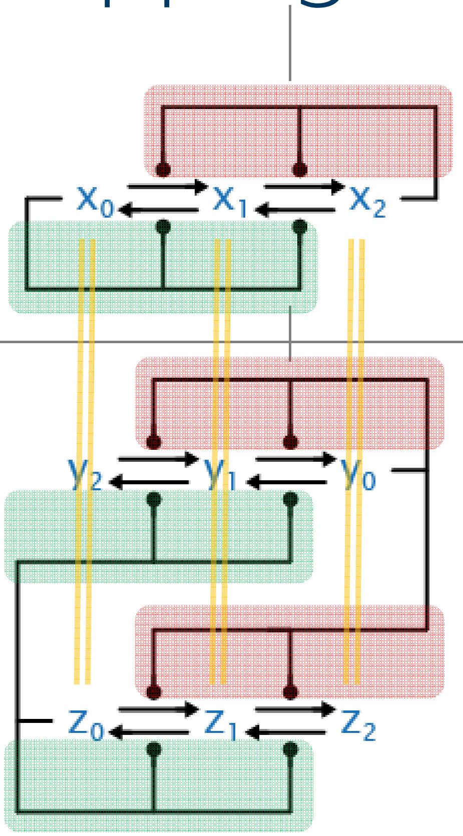
influence node

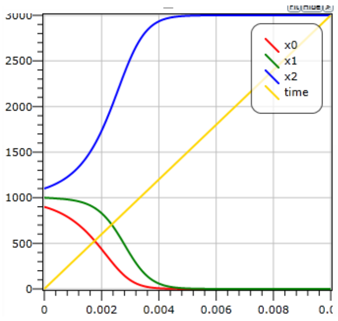
catalytic node

MI to AM mapping in detail



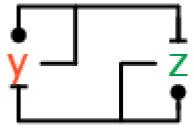
AM



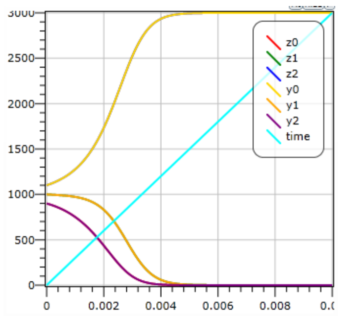


any initial conditions

homomorphic mapping



MI

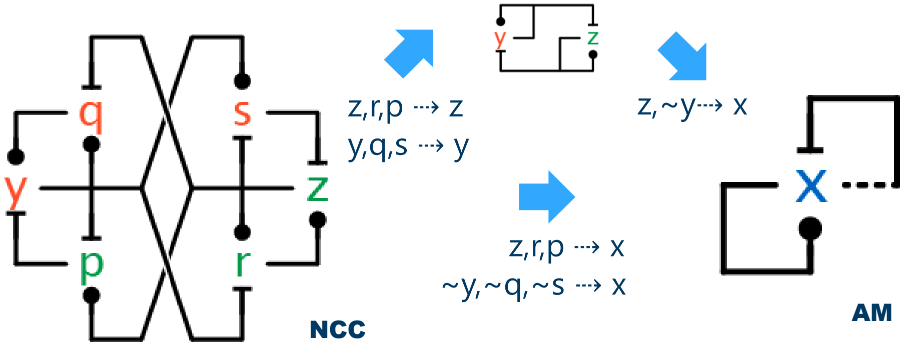


initial conditions:

$z_0 = y_2 = x_0$
$z_1 = y_1 = x_1$
$z_2 = y_0 = x_2$

Network Emulation Composes: NCC to AM

- For chosen initial conditions of NCC, the (18) trajectories of NCC emulate those (3) of AM

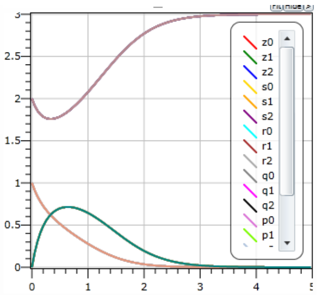


z,r,p ⇢ z
y,q,s ⇢ y

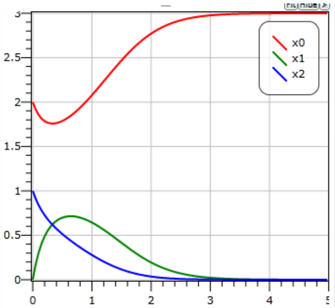
z,~y⇢ x

z,r,p ⇢ x
~y,~q,~s ⇢ x

This works also for GW, but not for the original CC.



(18 species on 3 trajectories)



(3 species on 3 trajectories)

The new cell cycle switch can emulate AM *exactly*.

For *any* initial conditions of AM.

And for *any* rates of AM.

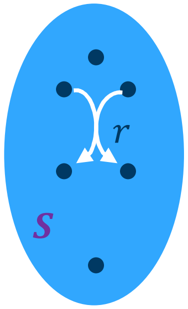
Why?

An Analytical Theory of Network Emulation

- An emulation is an "implementation"
  - "for every input produces the same output" → "for every initial conditions produces the same trajectories"
  - A refined network that works just as well as the coarser network *in the context* of the inputs of the coarser network (not arbitrary inputs)

- When can a network emulate another one?
  - Theories of behavioral equivalence and behavioral approximation, e.g. like in process algebra, are still lacking in this quantitative field
  - So we look at the continuous-state semantics of these networks, and see what we can do there

Chemical Reaction Networks



- A CRN is a pair (S, R) where
  - $S = \{s_1, \dots, s_n\}$ is a finite set of *species*
  - $R = \{r_1, \dots, r_m\}$ is a finite set of *reactions* over S
- Reactions $r = (\rho, \pi, k)$ written $\Sigma_{s \in S}\ \rho_s \cdot s \rightarrow^k \Sigma_{s \in S}\ \pi_s \cdot s$
  - Ex.: $r \;=\; 2A + B \rightarrow^k A + 3C$
  - *with* $\rho_A = 2,\ \rho_B = 1,\ \rho_C = 0$ *reactant stoichiometric numbers*
    $\pi_A = 1,\ \pi_B = 0,\ \pi_C = 3$ *product stoichiometric numbers*
- The *stoichiometry* of a species s in a reaction r is:

$\eta(s, (\rho, \pi, k)) = \pi_s - \rho_s$ *net stoichiometry* $\eta(A, r) = -1$

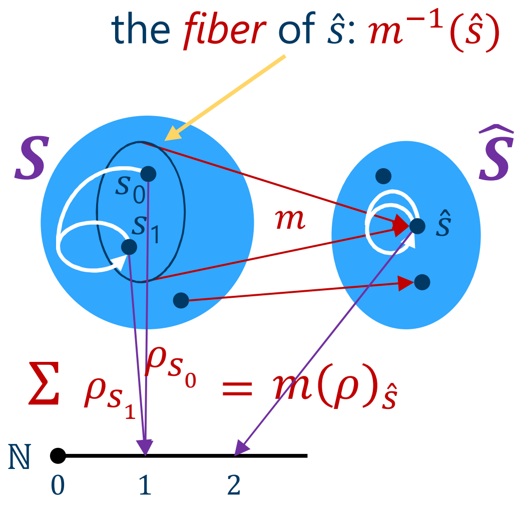
$\varphi(s, (\rho, \pi, k)) = k \cdot (\pi_s - \rho_s)$ *(instantaneous) stoichiometry* $\varphi(A, r) = -k$

Species Maps and Reaction Maps

- A *species map* is a map $\boldsymbol{m} \in \boldsymbol{S} \to \hat{\boldsymbol{S}}$
  - Ex: $m(s_0) = m(s_1) = \hat{s}$
- It induces a canonical *reaction map* $\boldsymbol{R} \to \hat{\boldsymbol{R}}$
  - Ex: $m(s_0 + s_1 \to^1 s_1) = 2\hat{s} \to^1 \hat{s}$
- Where $\boldsymbol{m}(\boldsymbol{\rho}, \boldsymbol{\pi}, \boldsymbol{k}) = (\boldsymbol{m}(\boldsymbol{\rho}), \boldsymbol{m}(\boldsymbol{\pi}), \boldsymbol{k})$
- And $\boldsymbol{m}(\boldsymbol{\rho})$ (similarly $\boldsymbol{m}(\boldsymbol{\pi})$) is the sum over fibers:

$$m(\rho)_{\hat{s}} = \Sigma_{s \in m^{-1}(\hat{s})}\, \rho_s$$

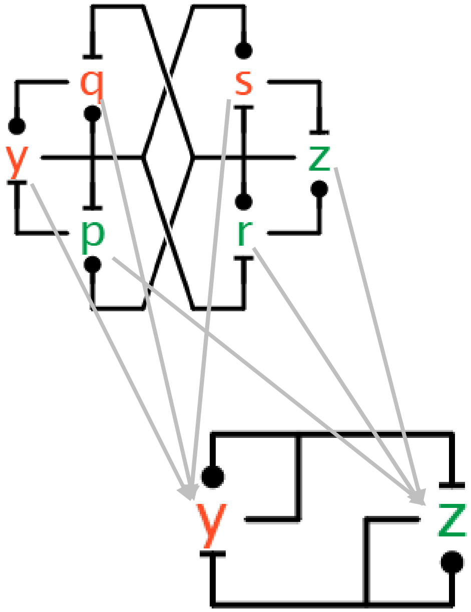
in case two species in the same reaction are mapped to the same species.



the *fiber* of $\hat{s}$: $m^{-1}(\hat{s})$

CRN Morphisms

- A *CRN morphism* is a map $m \in (S, R) \to (\hat{S}, \hat{R}) = (m_{\mathcal{S}}, m_{\mathcal{R}})$ with $m_{\mathcal{S}} \in S \to \hat{S}$ and $m_{\mathcal{R}} \in R \to \hat{R}$.
  - We are interested in morphisms that are *not* injective, that represent *implementations*



CRN Homomorphisms

- $m \in (S,R) \rightarrow (\hat{S},\hat{R})$ is a *CRN homomorphism* if $\boldsymbol{m_{\mathcal{R}}}$ is determined by $\boldsymbol{m_{\mathcal{S}}}$:

$$m_{\mathcal{R}}(\rho,\pi,k) = (m_{\mathcal{S}}(\rho), m_{\mathcal{S}}(\pi), k)$$

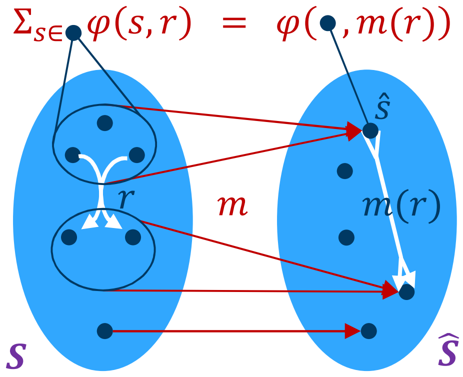
- Ex:

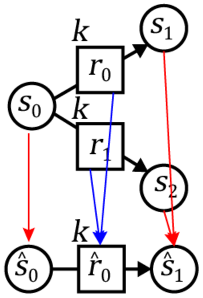
$$\begin{aligned} r_0\!: \quad & m_{\mathcal{R}}(s_0,s_1,k) = (\hat{s}_0,\hat{s}_1,k) = (m_{\mathcal{S}}(s_0), m_{\mathcal{S}}(s_1), k) \\ r_1\!: \quad & m_{\mathcal{R}}(s_0,s_2,k) = (\hat{s}_0,\hat{s}_1,k) = (m_{\mathcal{S}}(s_0), m_{\mathcal{S}}(s_2), k) \end{aligned}$$

- It implies that *for each reaction it preserves stoichiometry summed over species fibers*

$$\forall \hat{s} \in \hat{S}.\ \ \forall r \in R.\ \ \Sigma_{s \in m^{-1}(\hat{s})}\, \varphi(s,r) = \varphi(\hat{s}, m(r))$$

- But $\varphi(s_0,r_0) + \varphi(s_0,r_1) = -2k \neq -1k = \varphi(\hat{s}_0,\hat{r}_0)$ (see next slide)





Homomorphism
(but not *stoichiomorphism)*

CRN Stoichiomorphisms

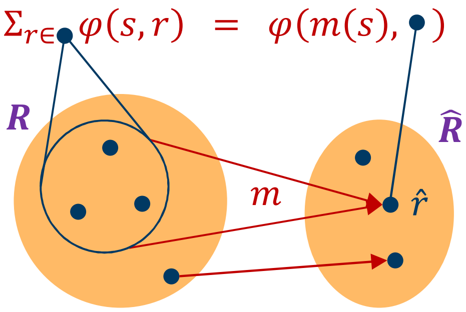
- $m \in (S,R) \to (\hat{S},\hat{R})$ is a *CRN stoichiomorphism* if <u>*for each species it preserves stoichiometry summed over reaction fibers*</u>

$$\forall s \in S.\ \ \forall \hat{r} \in \hat{R}.\ \ \Sigma_{r \in m^{-1}(\hat{r})}\, \varphi(s,r) = \varphi(m(s),\hat{r})$$

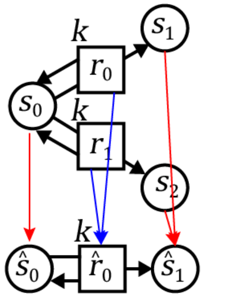
- This condition can be checked over the *syntax* of CRNs, without any consideration of their kinetics
  - Ex:

$$\begin{aligned}
s_0,\hat{r}_0:&\quad \varphi(s_0,r_0) + \varphi(s_0,r_1) = 0 = \ \varphi(\hat{s}_0,\hat{r}_0)\\
s_1,\hat{r}_0:&\quad \varphi(s_1,r_0) + \varphi(s_1,r_1) = 1k = \ \varphi(\hat{s}_1,\hat{r}_0)\\
s_2,\hat{r}_0:&\quad \varphi(s_2,r_0) + \varphi(s_2,r_1) = 1k = \ \varphi(\hat{s}_1,\hat{r}_0)
\end{aligned}$$

- We will show that existence of a stoichiomorphism implies identical network kinetics (in certain conditions).



$\Sigma_{r\in\bullet}\,\varphi(s,r) \ = \ \varphi(m(s),\bullet)$
R
$\hat{R}$
m
$\hat{r}$



Homomorphism and stoichiomorphism.

CRN Morphism Conditions

- Homomorphism consequence:

$$\forall \hat{s} \in \hat{S}.\ \ \forall r \in R.\ \ \Sigma_{s \in m^{-1}(\hat{s})}\, \varphi(s,r) = \varphi(\hat{s}, m(r))$$

- Stoichiomorphism condition:

$$\forall s \in S.\ \ \forall \hat{r} \in \hat{R}.\ \ \Sigma_{r \in m^{-1}(\hat{r})}\, \varphi(s,r) = \varphi(m(s), \hat{r})$$

- If m is an isomorphism (injective and surjective, with singleton fibers) then they both reduce to the simple property:

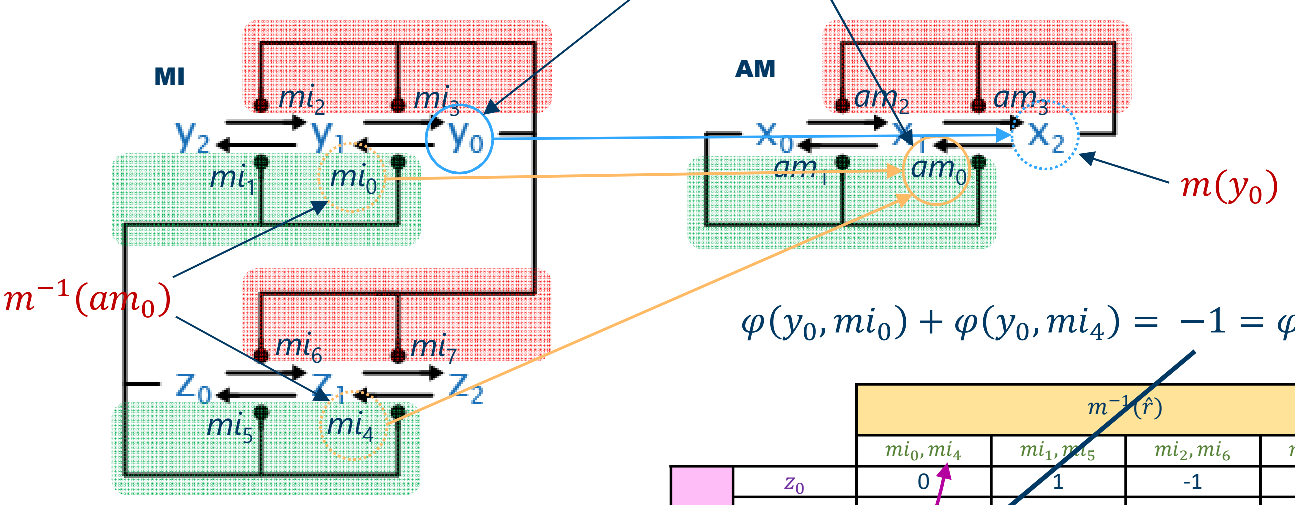
$$\forall s \in S.\ \ \forall r \in R.\ \ \varphi(s,r) = \varphi\big(m(s), m(r)\big)$$

- The above are generalization for when m is not injective.

Checking the Stoichiomorphism Condition

$m \in \mathrm{MI} \to \mathrm{AM}$

$$\forall s \in S.\quad \forall \hat{r} \in \hat{R}.\ \Sigma_{r \in m^{-1}(\hat{r})} \varphi(s,r) = \varphi(m(s),\hat{r})$$



$m(y_0)$

$m^{-1}(am_0)$

$$\varphi(y_0, mi_0) + \varphi(y_0, mi_4) = -1 = \varphi(x_2, am_0)$$

All unit rates (for simplicity)

		$m^{-1}(\hat{r})$				
		mi_0, mi_4	mi_1, mi_5	mi_2, mi_6	mi_3, mi_7	
$\forall s \in \mathrm{MI}$	z_0	0	1	-1	0	x_0
	z_1	1	-1	1	-1	x_1
	z_2	-1	0	0	1	x_2
	y_0	-1	0	0	1	x_2
	y_1	1	-1	1	-1	x_1
	y_2	0	1	-1	0	x_0
		am_0	am_1	am_2	am_3	$m(s)$
		$\forall \hat{r} \in \mathrm{AM}$				

This is both a homomorphism and a stoichiomorphism

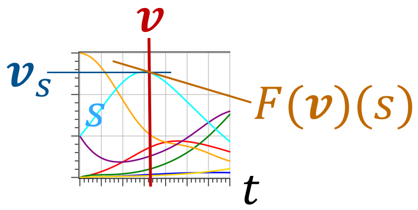
CRN Kinetics

- A *state* of a CRN (S,R) is a vector of concentrations for each species: $\boldsymbol{v} \in {\mathbb{R}^{+}}^{S}$.
- The *mass action* $[r] \in {\mathbb{R}^{+}}^{S} \to \mathbb{R}^{+}$ of a reaction $\boldsymbol{r} \in \boldsymbol{R}$ is:

$$[r]_{\boldsymbol{v}} = [(\rho,\pi,k)]_{\boldsymbol{v}} = \Pi_{s\in S}\, {\boldsymbol{v}_s}^{\rho_s} = \boldsymbol{v}^{\rho}$$

- The *differential system* of a CRN (S,R) is the map $\boldsymbol{F} \in {\mathbb{R}^{+}}^{S} \to \mathbb{R}^{S}$
  (for each state, gives the differential of concentration for each species):

$$F(\boldsymbol{v})(s) = \Sigma_{r\in R}\ \varphi(s,r)\cdot[r]_{\boldsymbol{v}}$$



- Normally written as a system of concentration ODEs, integrated over time:

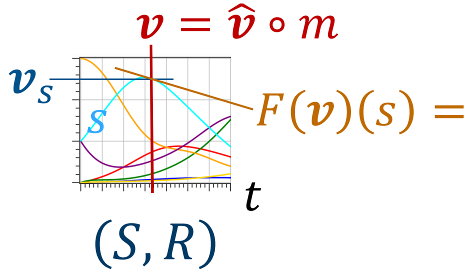
$$\frac{d\boldsymbol{v}_s}{dt} = F(\boldsymbol{v})(s) = \Sigma_{(\rho,\pi,k)\in R}\ k\cdot(\pi_s-\rho_s)\cdot\boldsymbol{v}^{\rho}$$

Kinetic Emulation

- A map $\boldsymbol{m} \in (S, R) \to (\hat{S}, \hat{R})$ is a *CRN emulation* if the following holds for the respective differential systems $\boldsymbol{F}, \hat{\boldsymbol{F}}$:

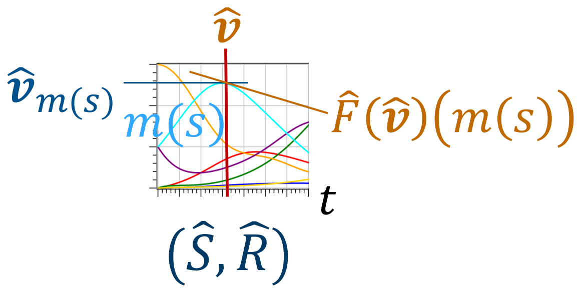
$$\forall \hat{\boldsymbol{v}} \in \mathbb{R}^{+\hat{S}}.\ \forall s \in S.\ F(\hat{\boldsymbol{v}} \circ m)(s) = \hat{F}(\hat{\boldsymbol{v}})\big(m(s)\big)$$

(the derivative of $\boldsymbol{s}$ in state $\hat{\boldsymbol{v}} \circ \boldsymbol{m}$ is equal to the derivative of $\boldsymbol{m}(\boldsymbol{s})$ in state $\hat{\boldsymbol{v}}$)



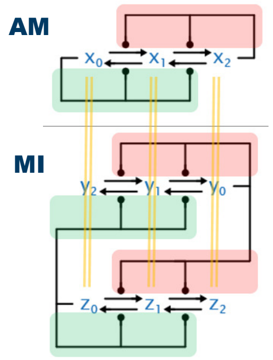
- It follows that for *any* initial state $\hat{\boldsymbol{v}}$ of $(\hat{S}, \hat{R})$ there is an initial state $\boldsymbol{v}$ (=$\hat{\boldsymbol{v}} \circ \boldsymbol{m}$) of (S, R) such that the trajectory of any $\boldsymbol{s}$ in (S, R) is identical to (*emulates*) the trajectory of $\boldsymbol{m}(\boldsymbol{s})$ in $(\hat{S}, \hat{R})$.

(With minor caveats if m is not surjective.)

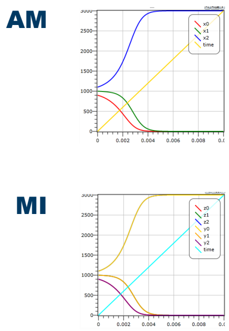


Emulation Theorem

- Theorem: If m is a CRN homomorphism and stoichiomorphism then it is a CRN emulation.



$\Rightarrow$



that is, for *any initial conditions* we can match trajectories.

- Actually, m need not be a homomorphism for this to hold: it is enough for m to be a *reactant morphism* and a stoichiomorphism. A reactant morphism agrees with the species map on the reactant species, but allows rates and product species to disagree. This allows a wider range of network mappings that preserve kinetics.

Change of Rates Theorem

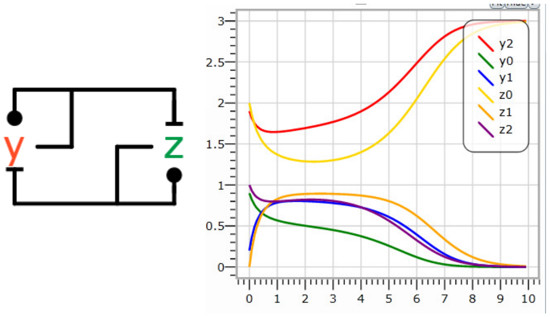
- A *change of rates* for (S, R) is bijection $\iota \in (S, R) \to (S, R')$ such that $\iota(S)$ is the identity and $\iota(\rho, \pi, k) = (\rho, \pi, k')$.

- Theorem: If $m \in (S, R) \to (\hat{S}, \hat{R})$ is a stoichiomorphism, then for *any* change of rates $\hat{\iota}$ of $(\hat{S}, \hat{R})$ there is a change of rates ι of (S, R) such that $\hat{\iota} \circ m \circ \iota^{-1}$ is a stoichiomorphism.
  - In fact, ι changes rates by the ratio with which $\hat{\iota}$ changes rates: $\iota(\rho, \pi, k) = \left(\rho, \pi, k \cdot \frac{\hat{k}'}{\hat{k}}\right)$ where $m(\rho, \pi, k) = (\hat{\rho}, \hat{\pi}, \hat{k})$ and $\hat{\iota}(\hat{\rho}, \hat{\pi}, \hat{k}) = (\hat{\rho}, \hat{\pi}, \hat{k}')$.

- Corollary: If $m \in (S, R) \to (\hat{S}, \hat{R})$ is a stoichiomorphism and homomorphism, then for *any change of rates* $\hat{\iota}$ of $(\hat{S}, \hat{R})$ there is a change of rates ι of (S, R) such that $\hat{\iota} \circ m \circ \iota^{-1}$ is an emulation.

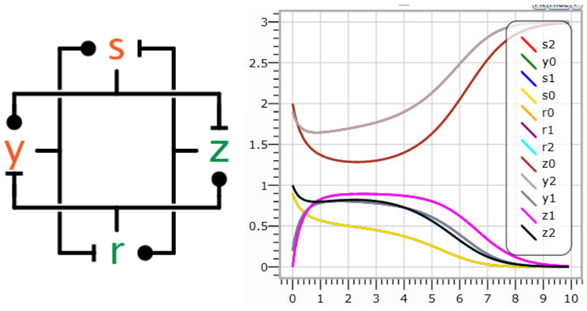
Any Rates, Any Initial Conditions

- A stoichiomorphism $\boldsymbol{m} \in (S, R) \to (\hat{S}, \hat{R})$ that is also a homomorphism, determines an emulation for any choice of rates of $(\hat{S}, \hat{R})$.
- Those emulations can match any initial conditions of any choice of rates of $(\hat{S}, \hat{R})$ with some initial conditions of some choice of rates of (S, R).



```
directive sample 10.0 100

init z0 2 |
init z1 0 |
init z2 1 |
init y0 0.9 |
init y1 0.2 |
init y2 1.9 |

y2 + y0 ->{0.9} y0 + y1 |
y1 + y0 ->{1.0} y0 + y0 |
z0 + y0 ->{1.1} y0 + z1 |
z1 + y0 ->{1.2} y0 + z2 |

y0 + z0 ->{0.7} z0 + y1 |
y1 + z0 ->{0.8} z0 + y2 |
z2 + z0 ->{0.5} z0 + z1 |
z1 + z0 ->{0.6} z0 + z0
```

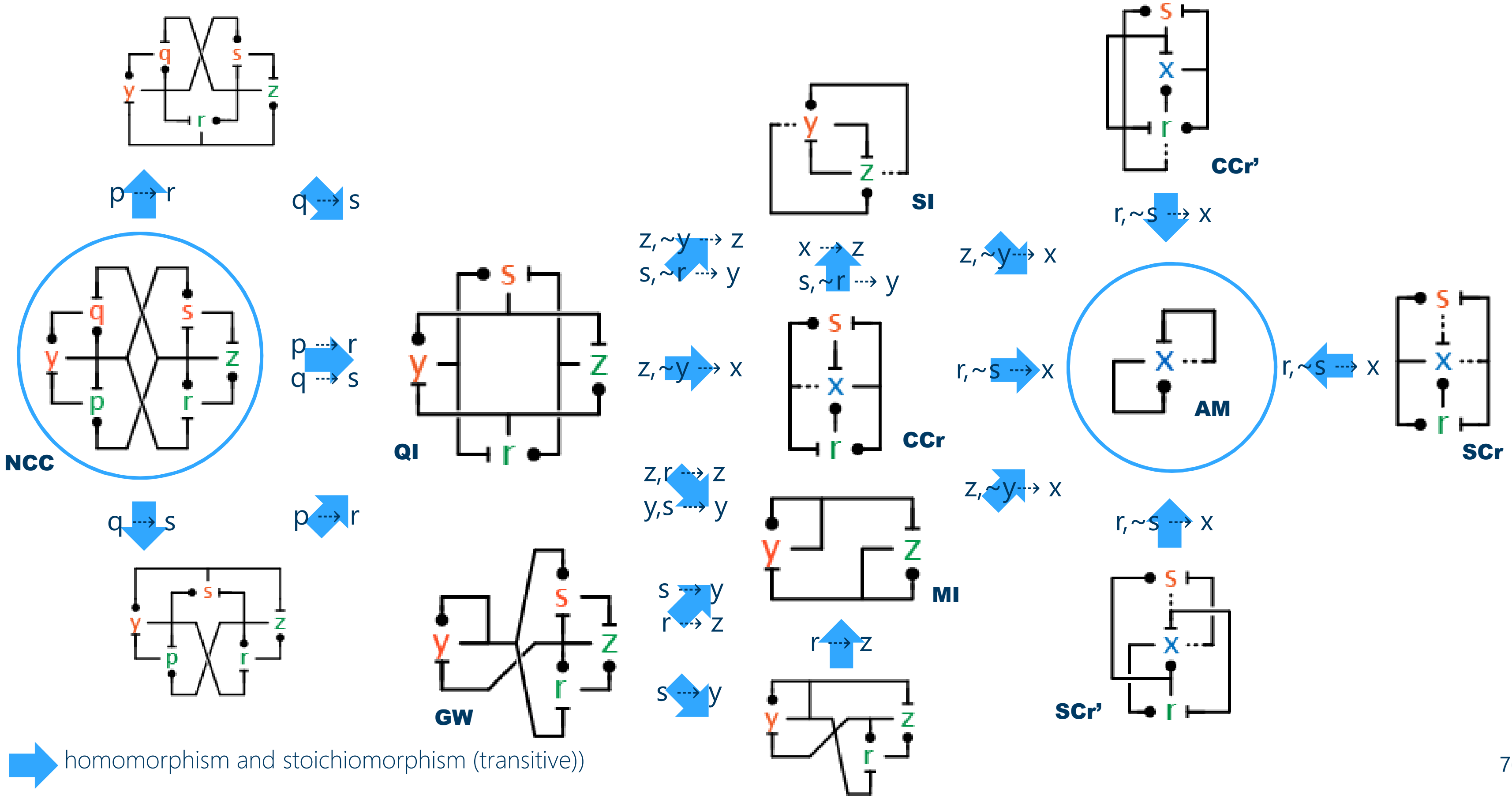
MI with completely heterogeneous rates and initial conditions



QI with matching rates and initial conditions

```
directive sample 10.0 100

init z0 2 |
init z1 0 |
init z2 1 |

init s0 0.9 |
init s1 0.2 |
init s2 1.9 |

init r0 2 |
init r1 0 |
init r2 1 |

init y0 0.9 |
init y1 0.2 |
init y2 1.9 |

s2 + y0 ->{0.9} y0 + s1 |
s1 + y0 ->{1.0} y0 + s0 |
r0 + y0 ->{1.1} y0 + r1 |
r1 + y0 ->{1.2} y0 + r2 |

s0 + z0 ->{0.7} z0 + s1 |
s1 + z0 ->{0.8} z0 + s2 |
r2 + z0 ->{0.5} z0 + r1 |
r1 + z0 ->{0.6} z0 + r0 |

y2 + s0 ->{0.9} s0 + y1 |
y1 + s0 ->{1.0} s0 + y0 |
z0 + s0 ->{1.1} s0 + z1 |
z1 + s0 ->{1.2} s0 + z2 |

y0 + r0 ->{0.7} r0 + y1 |
y1 + r0 ->{0.8} r0 + y2 |
z2 + r0 ->{0.5} r0 + z1 |
z1 + r0 ->{0.6} r0 + z0
```

Stoichiomorphism Zoo



(➡ homomorphism and stoichiomorphism (transitive))

Interpretation of Stoichiomorphism

- Ignorance about initial conditions
  - We may not know the concentrations of species in the more complex network, but at least we know that if they satisfy certain conditions, then it behaves like the simpler network.
- Neutral paths in network space (evolution)
  - If an evolutionary event happens to be a stoichiomorphism, or close to it, it will not be immediately selected against, because it is “kinetically neutral”.
  - This allows the network to increase its complexity without kinetic penalty.
  - Later, the extra degrees of freedom can lead to kinetic differentiation.
  - But meanwhile, the organism can explore variations of network structure.
- Relationship to abstraction / coarse-graining
  - Stoichiomorphism are not about abstractions that preserve behavior,
    on the contrary, they are about *concretions* that preserve behavior.
  - They describe *implementations* of abstract specs, where the specs themselves may not be (biologically) implementable because of excessive demands on individual species.

Conclusions

Conclusions

- The promise of nanotechnology
  - Controlling matter and information in detail at the molecular scale
    - This can only be achieved by *digital* (combinatorial) techniques
  - Interfacing to natural (biological) systems, which often have *analog* properties
    - This usually involves using continuous modeling/techniques
- Discrete systems are hard to engineer
  - We need combinatorial analysis techniques that scale up (massively!)
  - We need verification and approximation techniques for massive concurrency
- Continuous systems are hard to understand
  - Calculus is the weapon of choice, but even there *qualitative* understanding is king
  - We need quantitative methods that support qualitative reasoning