Deterministic Function Computation with Chemical Reaction Networks

David Doty (joint work with Ho-Lin Chen and David Soloveichik)
University of British Columbia
September 14, 2012

The programming language of chemical kinetics

Use the language of coupled chemical reactions *prescriptively* as a "programming language" for engineering new systems (rather than *descriptively* as a modeling language for existing systems)

Real programmers code in CHEMISTRY

Cells are smart: controlled by signaling and regulatory networks

Human white blood cell chasing a bacterium:

source: David Rogers, Vanderbilt University

Want to engineer embedded controllers for biochemical systems, “wet robots”, smart drugs, etc.

Need to understand theoretical principles of chemical computation

Chemical Reaction Networks (CRN)

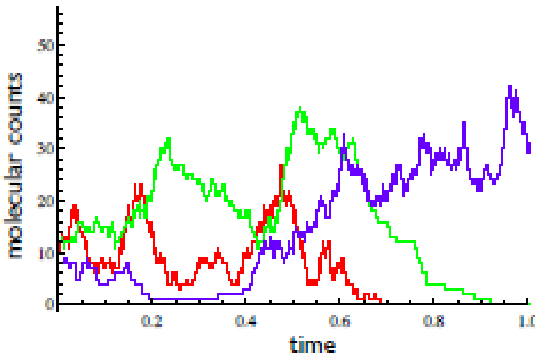
syntax:

$$
\begin{aligned}
A &\xrightarrow{0.03} 2A \\
2A &\xrightarrow{5\times10^4} A \\
B + A &\xrightarrow{10^5} 2B \\
B &\xrightarrow{0.01} \\
A + C &\xrightarrow{10^5} \\
C &\xrightarrow{0.0165} 2C \\
2C &\xrightarrow{5\times10^4} C
\end{aligned}
$$

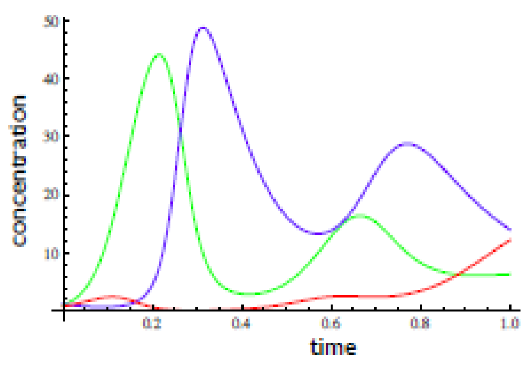
two possible semantics:

stochastic: discrete state space, continuous time Poisson process

we use only stochastic CRNs in this talk



mass-action: continuous ODEs



Discrete (Stochastic) CRN Model

- Finite set of **species** $\{X, Y, Z, \ldots\}$
- A **state** is a nonnegative integer vector **c** indicating the *count* (number of molecules) of each species: write counts as $\#_{\mathbf{c}}X$, $\#_{\mathbf{c}}Y$, …
- Finite set of **reactions**: e.g.

$$X \to W + Y + Z$$

$$A + B \to C$$

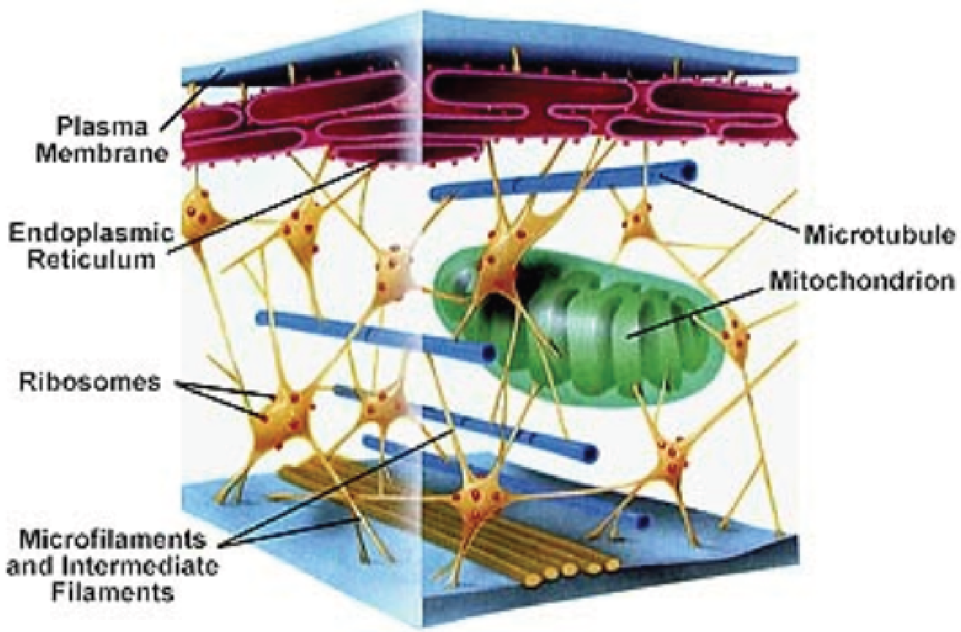
(in this talk, all rate constants are 1, and all reactions are unimolecular or bimolecular)

Objections?

What is not captured?

Biological examples

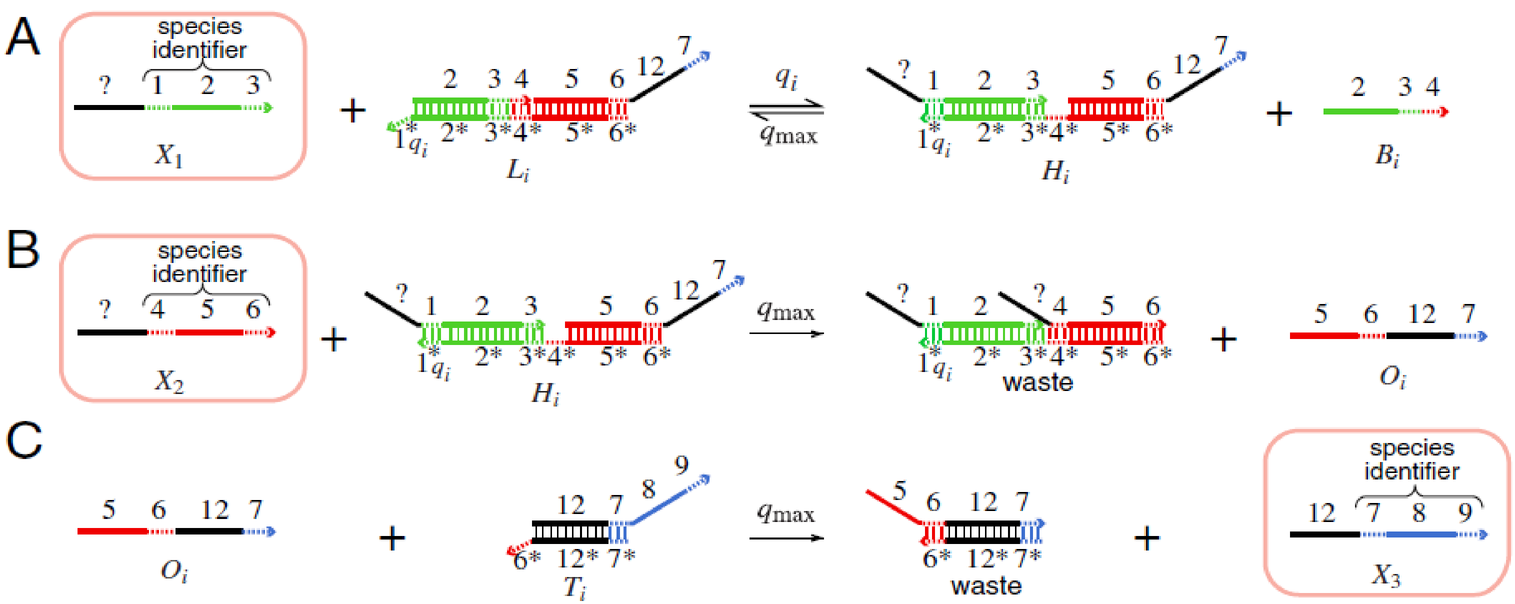
Localization, space, assembly/ disassembly, movement



Combinatorial species

Are CRNs an "implementable" programming language?

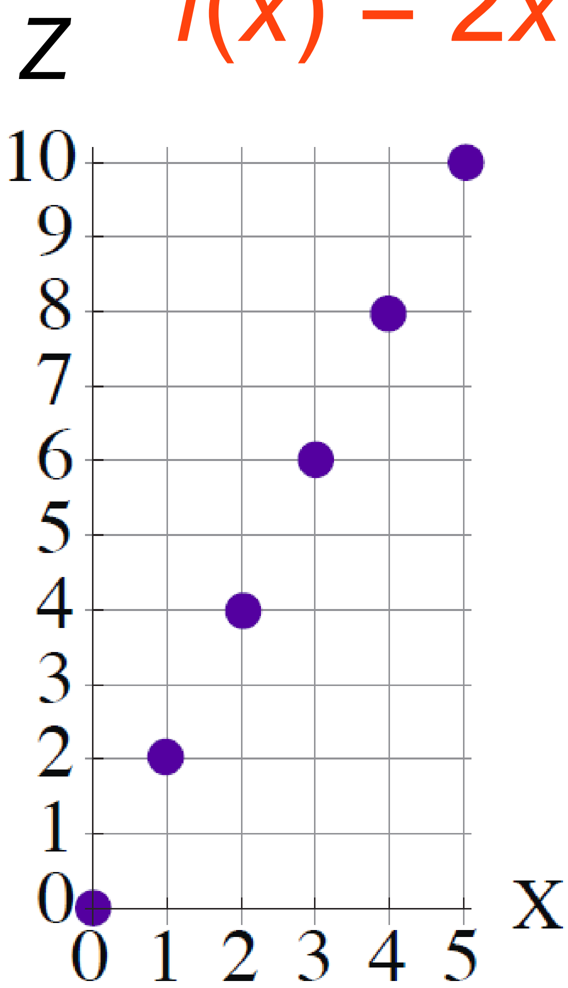
- "I don't believe that every crazy CRN you write down actually describes real chemicals!"
- **Response to objection**: Soloveichik, Seelig, Winfree [*PNAS* 2010] found a physical implementation (high-accuracy approximation) of any CRN, using *nucleic-acid strand displacement cascades*



Deterministic Function Computation with CRNs

Deterministic function computation with CRNs (example 1)

$f(x) = 2x$

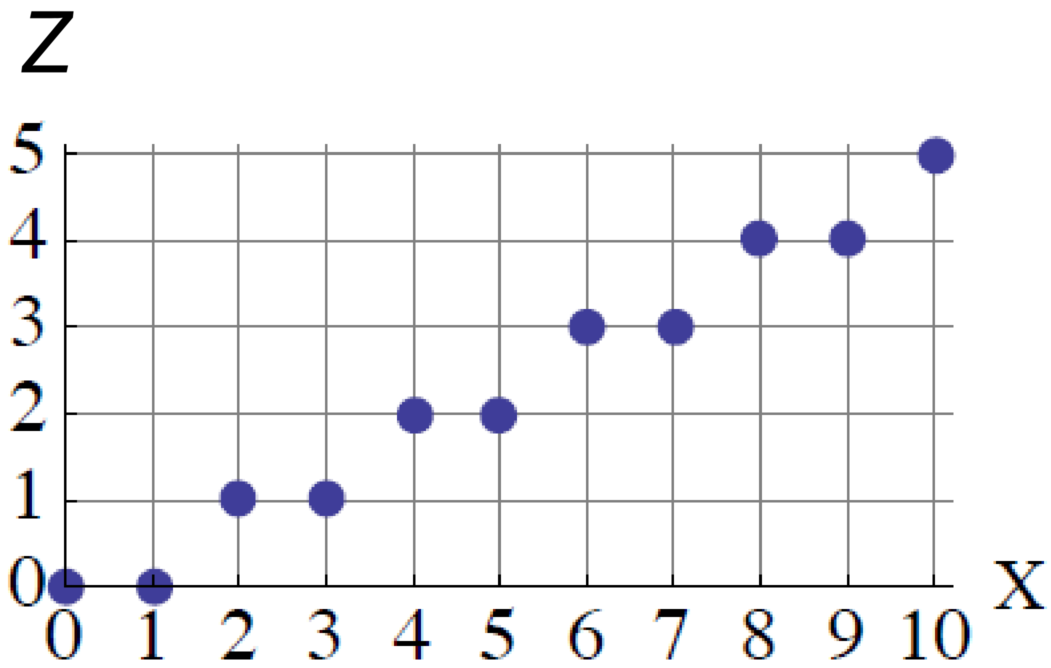


start with x (input amount) of X

$X \rightarrow Z + Z$

Deterministic function computation with CRNs (example 2)

$f(x) = \lfloor x/2 \rfloor$

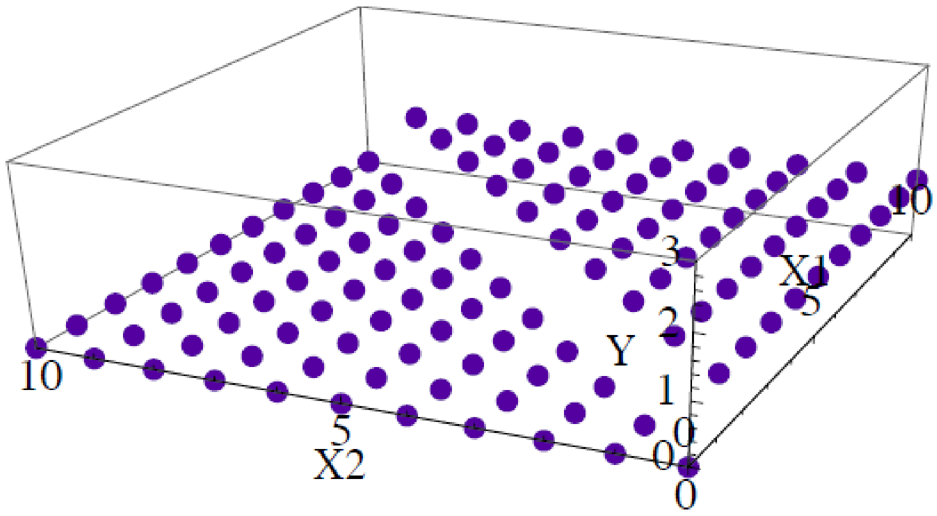


start with input amount of X

$X + X \rightarrow Z$

Deterministic function computation with CRNs (example 3)

$f(x_1,x_2)$ = if $x_1 > x_2$ then $y = 1$ else $y = 0$

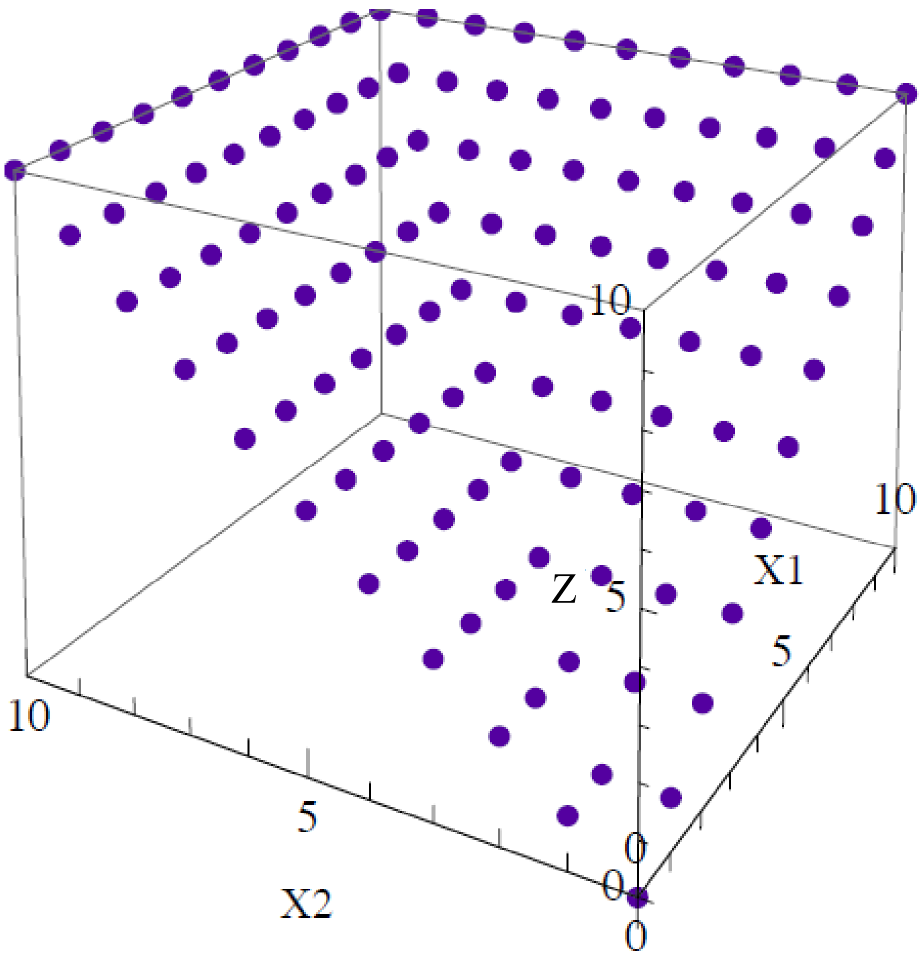


start with 1 N and input amounts of X_1,X_2

$$X_1 + N \rightarrow Y$$
$$X_2 + Y \rightarrow N$$

Deterministic function computation with CRNs (example 4)

$f(x_1,x_2) = \max\{x_1,x_2\}$



start with input amounts of X_1, X_2

$X_1 \to X_1' + Z$

$X_2 \to X_2' + Z$

$X_1' + X_2' \to K$

$K + Z \to \varnothing$

Other functions?

$f(x) = x^2$?

$f(x_1, x_2) = x_1 \cdot x_2$?

$f(x) = 2^x$?

Deterministic function computation with CRNs (definition)

task: compute function $\mathbf{z} = f(\mathbf{x})$ $(\mathbf{x} \in \mathbb{N}^k, \mathbf{z} \in \mathbb{N}^l)$

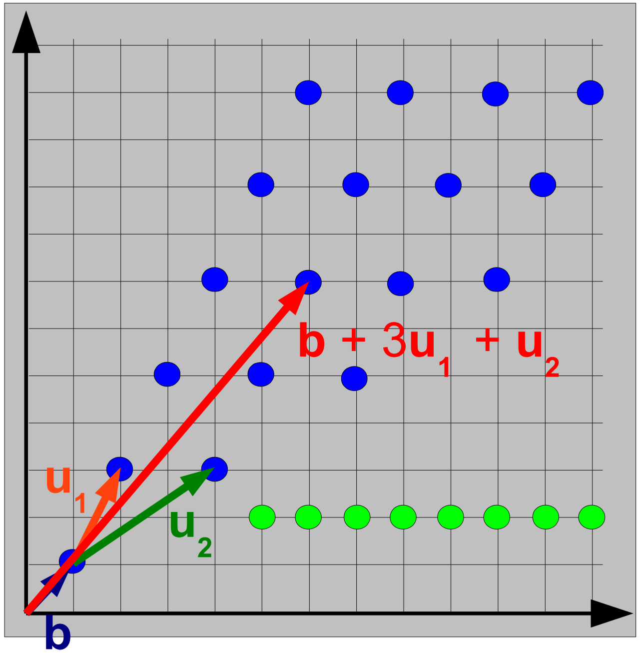
- **initial state**: input counts $X_1, X_2, \ldots, X_k$ (and fixed counts of non-input species)
- **output**: counts of $Z_1, Z_2, \ldots, Z_l$
- **output-stable state**: all states reachable from it have same counts of $Z_1, Z_2, \ldots, Z_l$
- **deterministic computation**: a correct output-stable state "reached with probability 1 in the limit $t \to \infty$"

Main result

Theorem: Functions f: $\mathbb{N}^k \to \mathbb{N}^l$ deterministically computable by CRNs are precisely those with a *semilinear graph*. graph(f) = $\{ (\mathbf{x},\mathbf{z}) \in \mathbb{N}^{k+l} \mid f(\mathbf{x})=\mathbf{z} \}$

$A \subseteq \mathbb{N}^{k+l}$ is linear if there are vectors $\mathbf{b}$, $\mathbf{u}_1$, …, $\mathbf{u}_p$ so that $A = \{ \mathbf{b} + n_1 \cdot \mathbf{u}_1 + \ldots + n_p \cdot \mathbf{u}_p \mid n_1,\ldots,n_p \in \mathbb{N} \}$

A is semilinear if it is a finite union of linear sets.

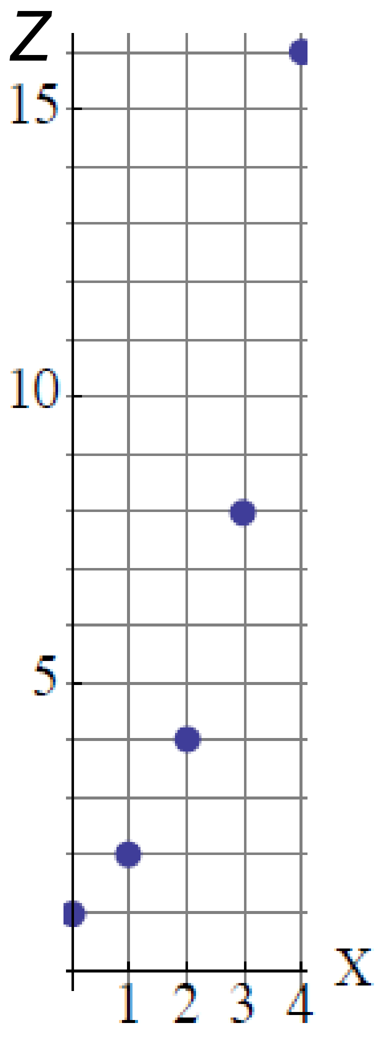


Intuition: linear sets are multi-dimensional "periodic" sets

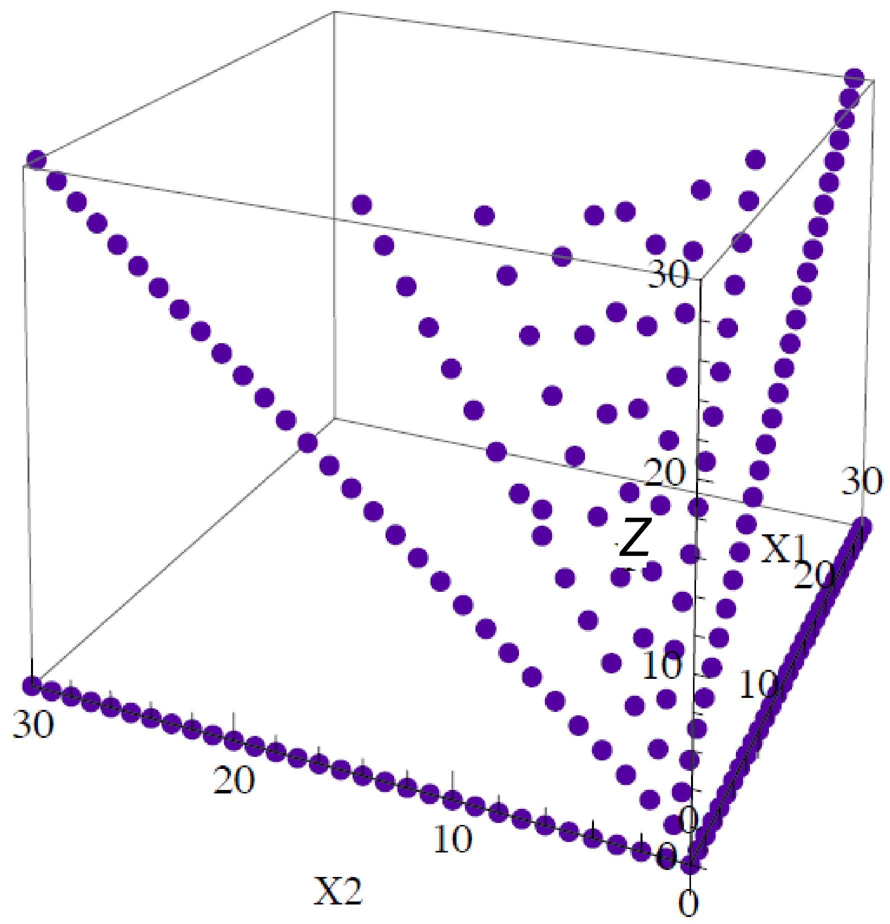
Semilinear functions are "piecewise linear functions" with a finite number of pieces

Non-semilinear examples

$f(x) = 2^x$

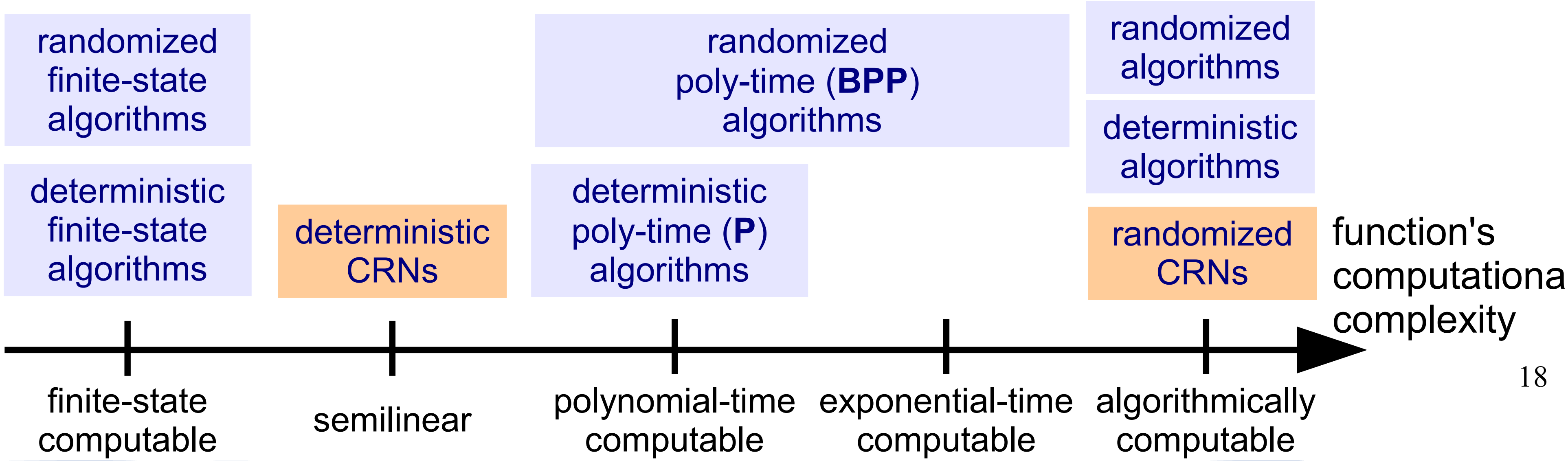


$f(x_1, x_2) = x_1 \cdot x_2$



What if we allow error?

- Any function computable by an algorithm is computable by a randomized CRN with arbitrarily small positive probability of error.
  - [Soloveichik, Cook, Winfree, Bruck, *Natural Computing* 2008]
  - [Angluin, Aspnes, Eisenstat, *DISC* 2006]
- **Lesson**: disallowing error hurts chemical algorithms much more than it hurts conventional algorithms



How do we show this?

Theorem [Angluin, Aspnes, Eisenstat, PODC 2006]: The *predicates* decidable by CRNs are precisely the semilinear predicates.

We connect computation of *functions* (integer output) to computation of *predicates* (YES/NO output)

Deterministic predicate computation with stochastic CRNs (definition)

task: decide predicate $b = \varphi(\mathbf{x})$ ($\mathbf{x} \in \mathbb{N}^k$, $b \in$ {yes,no})

- **initial state**: input counts X_1, X_2, …, X_k (and fixed counts of non-input species)
- **output**: either $\#Y > 0$ and $\#N = 0$ (yes)
  or $\#Y = 0$ and $\#N > 0$ (no)
- **output-stable state**: all states reachable from it have same yes/no answer
- **set decided by CRN**: $S_{yes} = \{ \mathbf{x} \in \mathbb{N}^k \mid \varphi(\mathbf{x}) = \text{yes} \}$

Two directions to proof

(reminder) **Theorem** [Angluin, Aspnes, Eisenstat, *PODC* 2006]: The sets decidable by CRNs are precisely the semilinear sets.

- Only semilinear functions can be computed:

  f computed by CRN $C \Rightarrow$ graph(f) decided by CRN D
- All semilinear functions can be computed:

  graph(f) decided by CRN $D \Rightarrow f$ computed by CRN C

  direct systematic construction to show computation of $f(\boldsymbol{x})$ can be done in expected time $O(\text{polylog } ||\boldsymbol{x}||)$

f computed by CRN $C \Rightarrow$ graph(f) decided by CRN D

- Want to decide, given input (x,z), is $f(x) = z$?
- Keep track of total number of Z's ever produced or consumed:

CRN C CRN D

$A + B \to Z + W$ becomes $A + B \to Z + W + Z_P$

$A + Z \to B$ becomes $A + Z \to B + Z_C$

monotonic production of Z_P and Z_C allows composition

- Initial state has z copies of Z_C

Eventually all Z_P and Z_C go away (if equal) or one is left over (if unequal)

$Z_P + Z_C \to Y$

$Y + N \to Y$

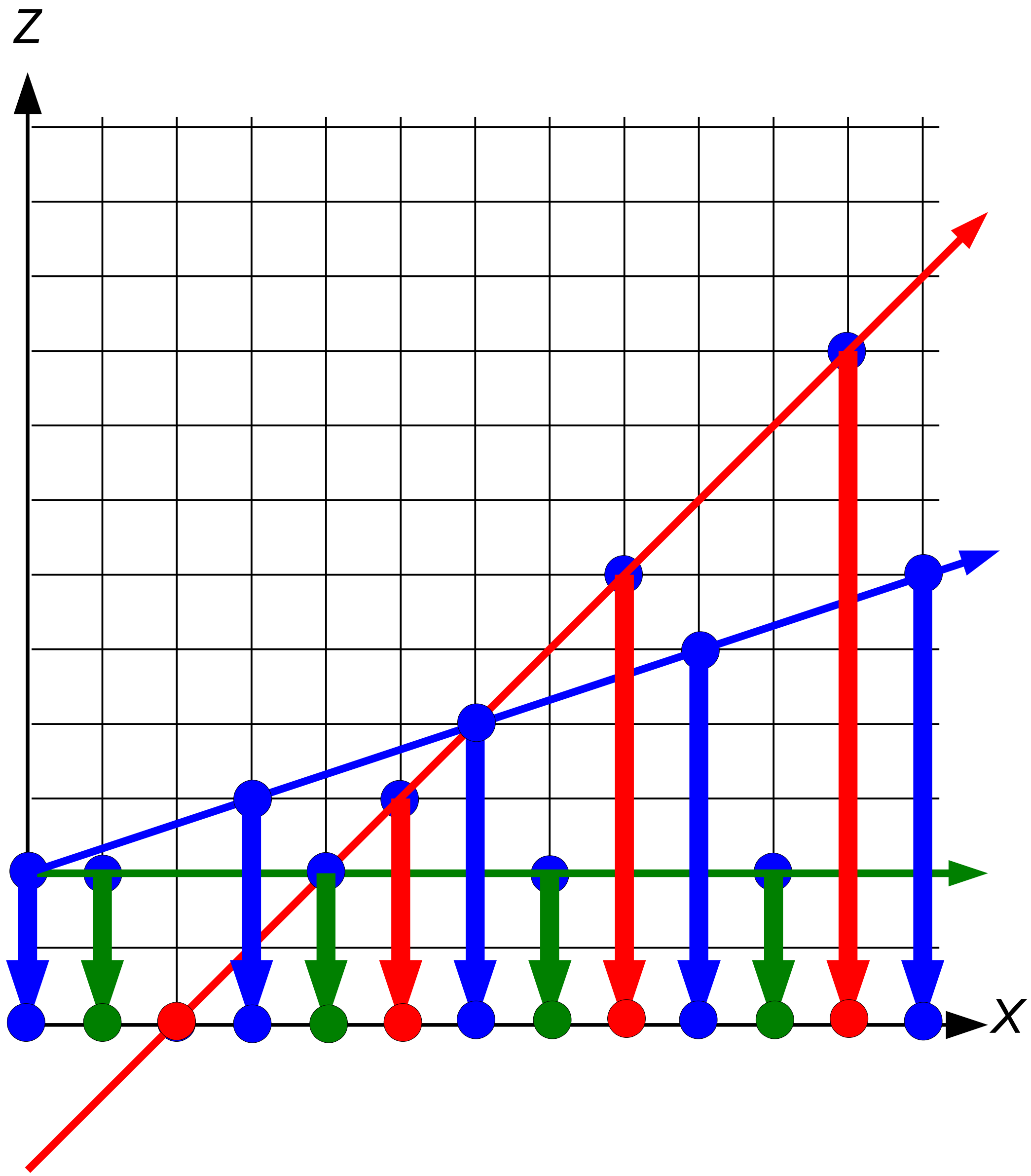
If neither is left over, change answer to YES

$Z_P + Y \to Z_P + N$

$Z_C + Y \to Z_C + N$

If Z_P or Z_C are left over, change answer to NO

Computing any semilinear *f*



Alternative characterization: partial linear functions

$f_1(x) = x/3 + 2$

start with 1 L_0, 2 Z

$L_0 + X \to L_1$

$L_1 + X \to L_2$

$L_2 + X \to L_0 + Z$

$f_2(x) = x - 2$

$f_3(x) = 2$

… with linear domains

Deciding which domain input *x* is in can be done with a CRN

How fast can semilinear functions be computed?

$||\mathbf{x}|| = \Sigma_i\, \mathbf{x}(i)$

Construction on previous slide takes O(||**x**|| log ||**x**||), but ...

Theorem: Every semilinear function *f* can be computed by a CRN on input **x** in expected time $O(\log^5 ||\mathbf{x}||)$.

i.e., in time $O(n^5)$, where *n* is the number of bits to write **x** in binary

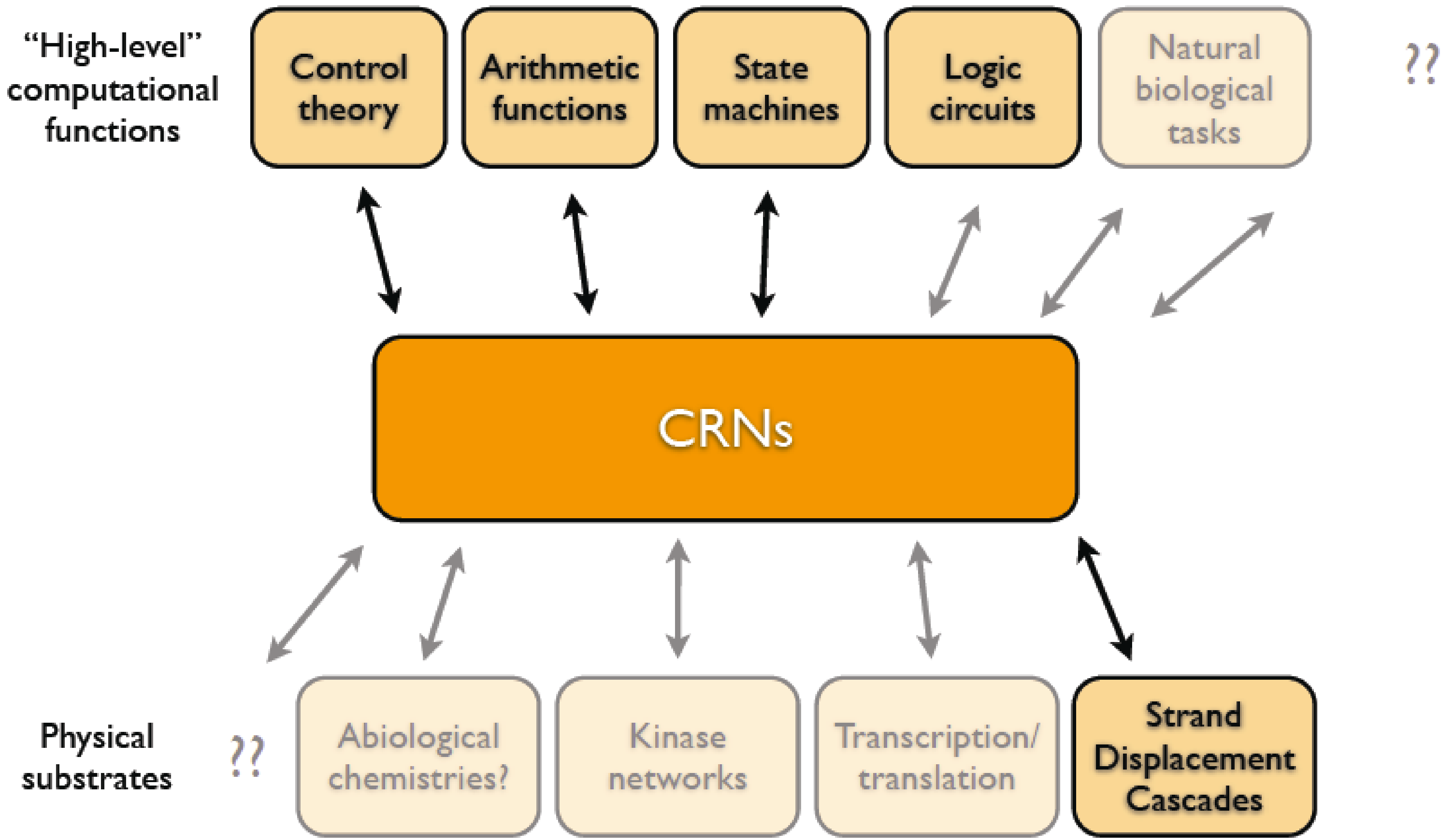
Proof: Combine slow deterministic construction with $O(\log^5 ||\mathbf{x}||)$ CRN computing *f* with arbitrarily small probability of error. [1,2]

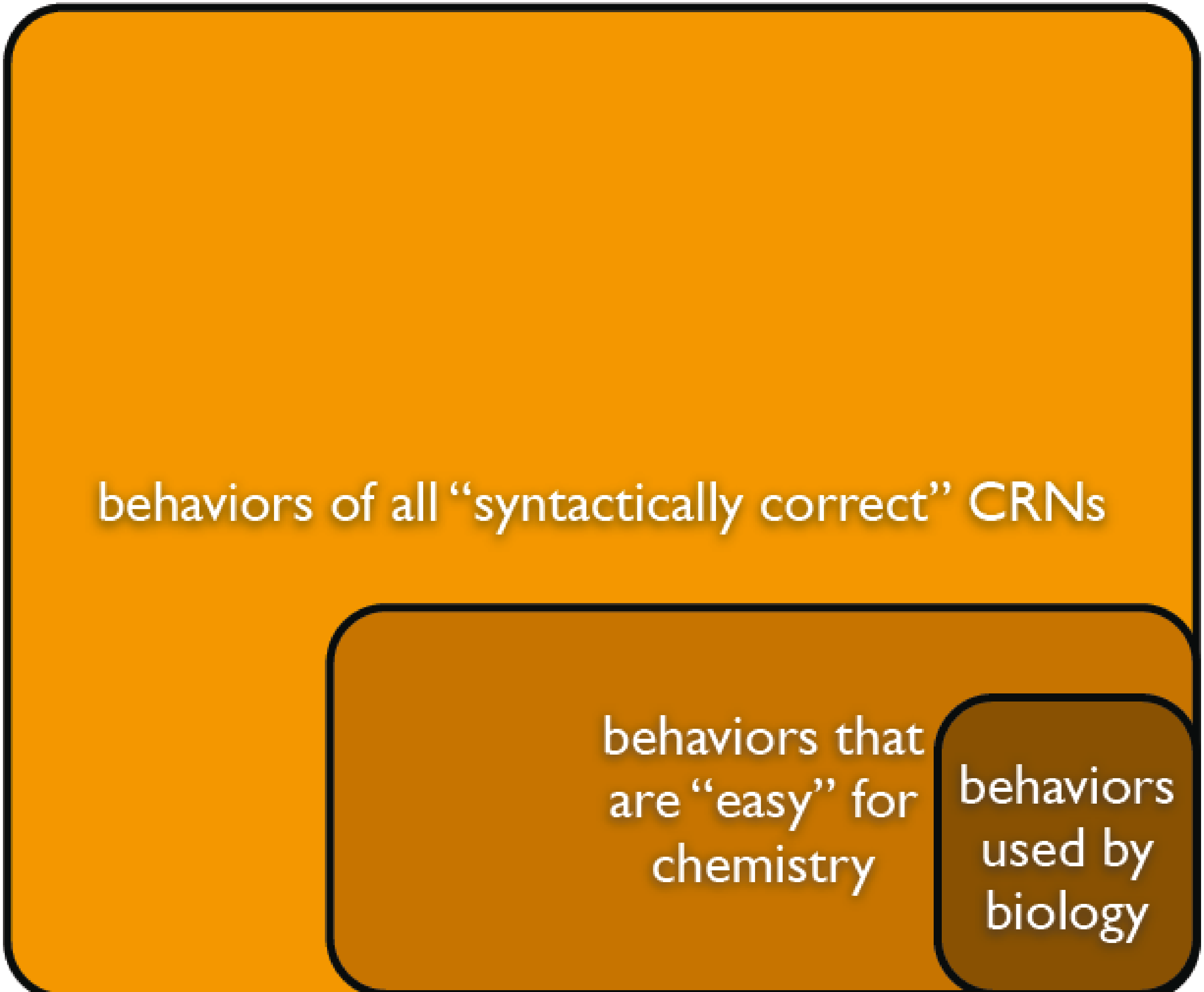
- If fast CRN is correct then output stabilizes quickly
- Otherwise, slow CRN compares and corrects the output

Error probability is small enough that total expected time to stabilize to correct answer remains $O(\log^5 ||\mathbf{x}||)$.

[1] Angluin, Aspnes, Eisenstat, "Fast computation by population protocols with a leader", *DISC* 2006

[2] Soloveichik, Cook, Winfree, Bruck, "Computation with finite stochastic chemical reaction networks", *Natural Computing* 2008


"High-level" computational functions
Control theory
Arithmetic functions
State machines
Logic circuits
Natural biological tasks
??
CRNs
Physical substrates
??
Abiological chemistries?
Kinase networks
Transcription/ translation
Strand Displacement Cascades


behaviors of all "syntactically correct" CRNs
behaviors that are "easy" for chemistry
behaviors used by biology

Acknowledgments

Ho-Lin Chen

David Soloveichik

UCSF
center for systems & synthetic biology
an NIGMS national systems biology center

Damien Woods

Niranjan Srinivas

Thank you!