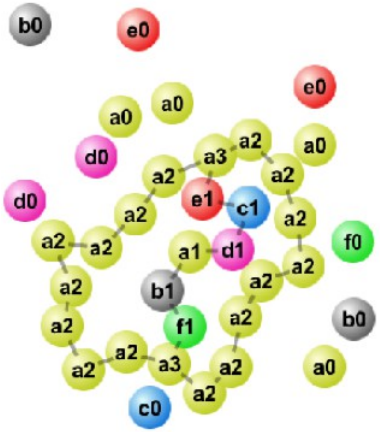
Artificial Chemistries: Overview and Computing Aspects



Lidia Yamamoto

(KULeuven, Belgium)

This talk has been prerecorded!

watch the corresponding video on YouTube:

`https://youtu.be/-JDFdIztGk0`

presenting joint work with

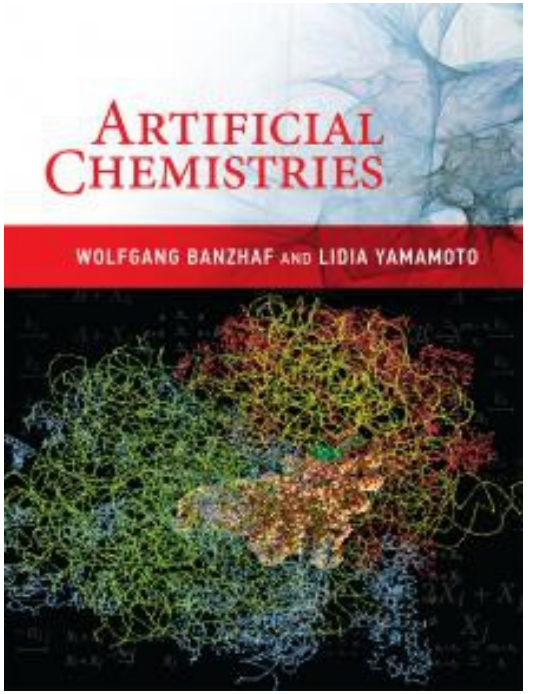
Wolfgang Banzhaf

(Memorial University of Newfoundland, Canada)

Engineering and Control of Self-Organization, e-track CS-DC 2015

Contents

- Artificial Chemistries (ACs) in a Nutshell
  - Wet (*"in vitro", "in vivo"*) vs. virtual (*"in silico"*) ACs
  - Well-mixed vs. spatial ACs
  - Constructive vs. nonconstructive ACs
  - Simulation algorithms
- Computing with Artificial Chemistries
  - In silico, in vitro and in vivo chemical computing
  - Distributed computing and self-organization
- PyCellChemistry software package
  - `www.artificial-chemistries.org`
- Summary and Outlook



MIT Press, Summer 2015 (571 pages)
`https://mitpress.mit.edu/books/artificial-chemistries`

Artificial Chemistries (ACs)

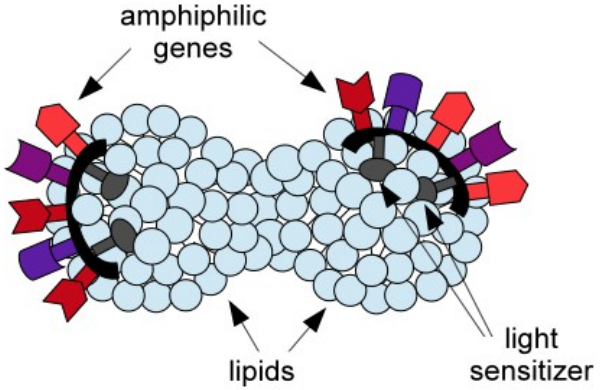
- Man-made virtual or physical systems where objects are transformed in interactions, like molecules in chemical reactions

$$abc + d \rightarrow abcd \quad , \quad 0101100 \rightarrow 01 + 1100 \quad , \quad 40 + 10 \rightarrow 10 + 4$$

- Spin-off of Artificial Life:
  - from *"life as it could be"* to *"chemistry as it could be (imagined)?"*
- Goals:
  - understand phenomena leading to the emergence of life
  - create new forms of synthetic life from the bottom up
    - *"in vitro", "in vivo"*: "Wet" ACs in the laboratory
    - *"in silico"*: computational systems
      - high-level modelling and simulation of (real) chemistry and biology
      - chemistry as a metaphor for distributed and parallel computer algorithms
      - chemistry as a general model for interacting systems of objects: nuclear physics, language, music, economies

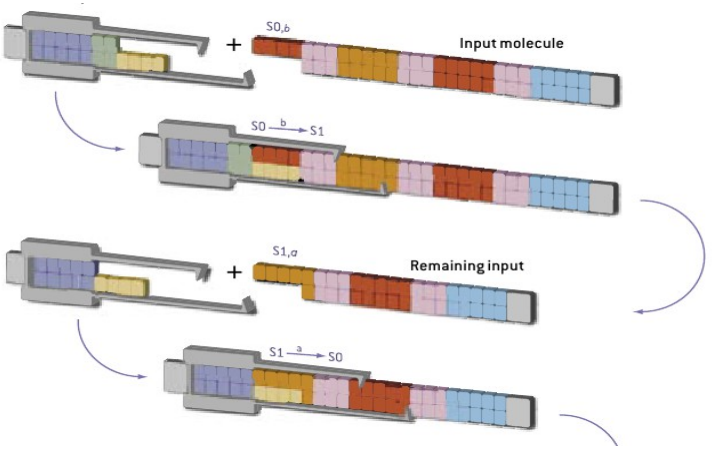
Wet ACs

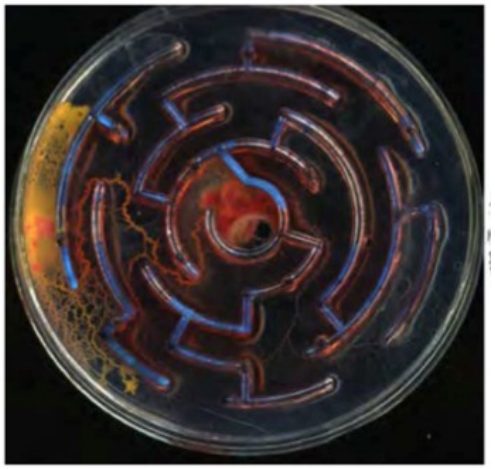
- DNA computing
- Reaction-diffusion computers
- Synthetic life and protocells
- Computing with bacteria, slime mold, ...



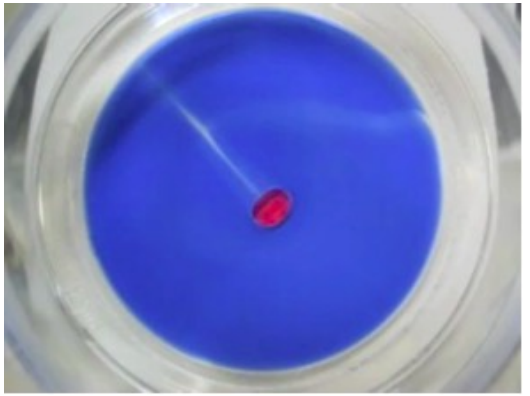
Los Alamos Bug
[Rasmussen2003]

Molecular Automaton
[Benenson2003][Shapiro2006]





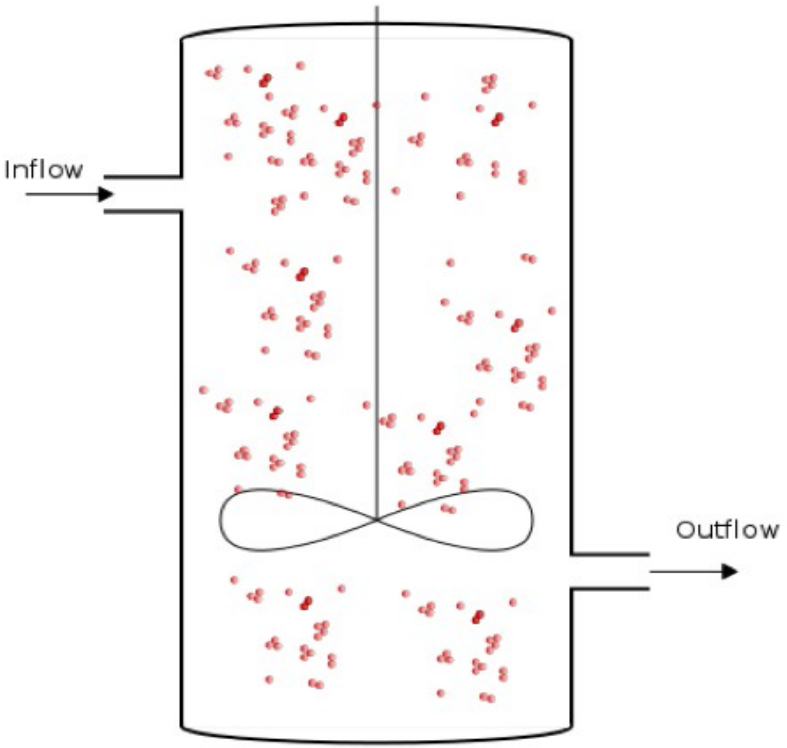
slime mold maze solver
[Adamatzky2010,2012]



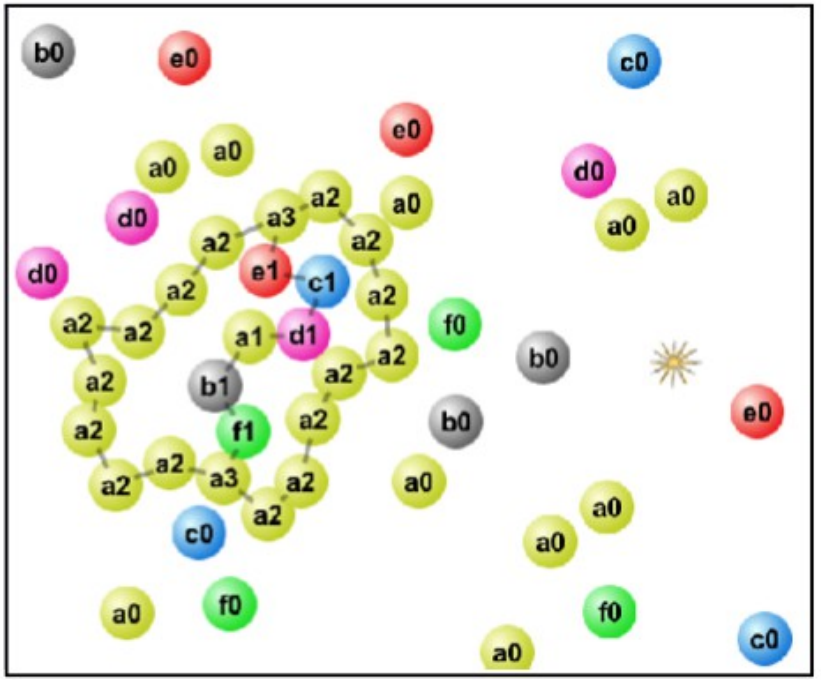
self-propelled oil droplet
[Hanczyc2010]

Artificial Chemistries "in silico"

- Virtual, abstract ACs:
  - well-stirred: molecules as a "gas" or dissolved in well-mixed reactor
  - spatially-resolved: molecules move in 2D or 3D space
  - compartmentalized: molecules inside various (nested) containers



well-stirred AC

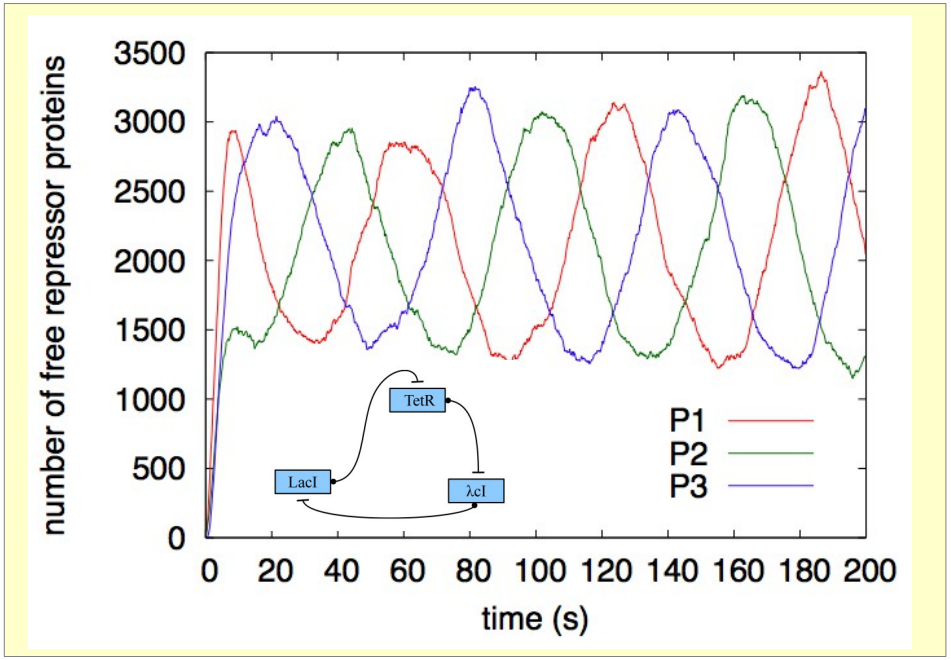


example of spatial AC:
Organic Builder [Hutton2009]

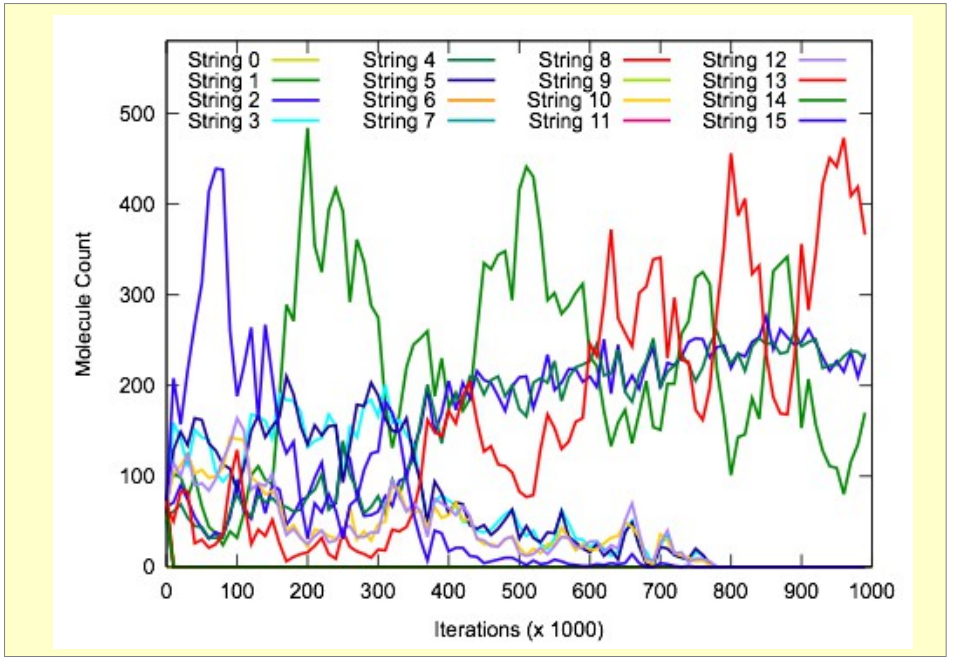
Constructive vs. Nonconstructive ACs

- N = total number of possible molecular species
- M = number of species present in the reactor at a given moment
- Nonconstructive: M = N or close: fixed set of molecules
- Constructive: M << N
  - new molecules may be created, with potentially new interactions

example of nonconstructive AC:
the Repressilator [Elowitz2000]



example of constructive AC:
the Matrix Chemistry [Banzhaf1993]



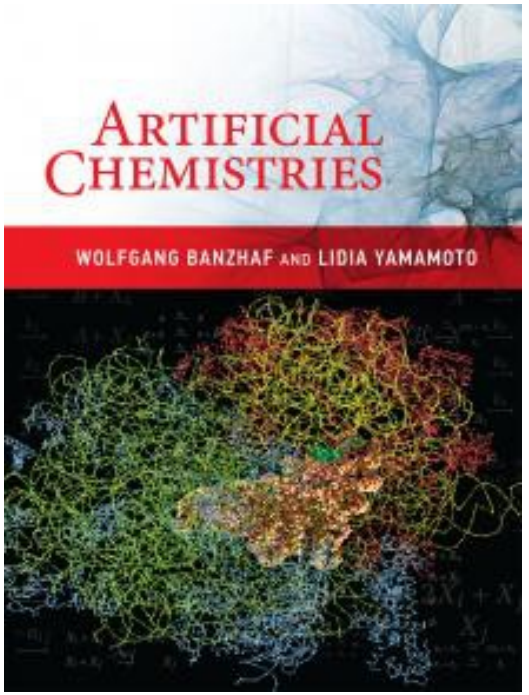
Components of an Artificial Chemistry

- Triple (S,R,A)
  - S = set of molecules
  - R = set of reaction rules
  - A = algorithm that applies rules to molecules
- Some algorithms: (see book ch. 4 or [Yamamoto2013] for a survey)

granularity	well mixed	spatial, compartmental
individual molecules, single reactions	random molecular collisions: effective or elastic	move, collide, react (gas vs. fluid dynamics, lattice systems, crowding)
molecular species, effective reactions	reaction probability proportional to propensity (Gillespie SSA, next reaction method)	next subvolume method, multicompartment Gillespie
groups of molecules and reactions	fire groups of reactions together within interval tau (tau leaping)	spatial tau-leaping
concentration changes	numerical ODE integration	PDE integration

Contents

- Artificial Chemistries (ACs) in a Nutshell
  - Wet (*"in vitro", "in vivo"*) vs. virtual (*"in silico"*) ACs
  - Well-mixed vs. spatial ACs
  - Constructive vs. nonconstructive ACs
  - Simulation algorithms
- **Computing with Artificial Chemistries**
  - In silico, in vitro and in vivo chemical computing
  - Distributed computing and self-organization
- PyCellChemistry software package
- Summary and Outlook

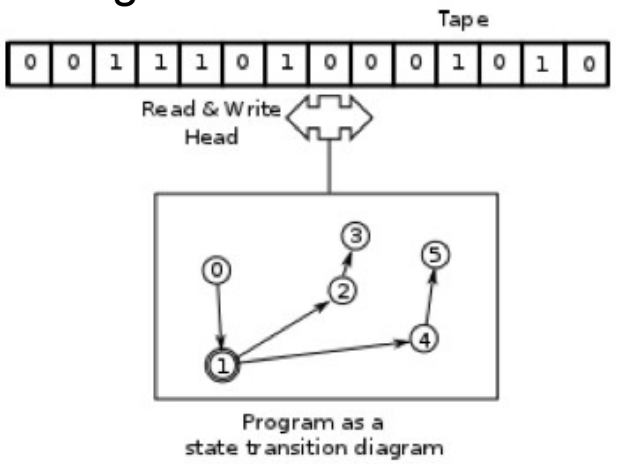


cover image: molecular model of a ribosome [theasis,iStockphotoLP]

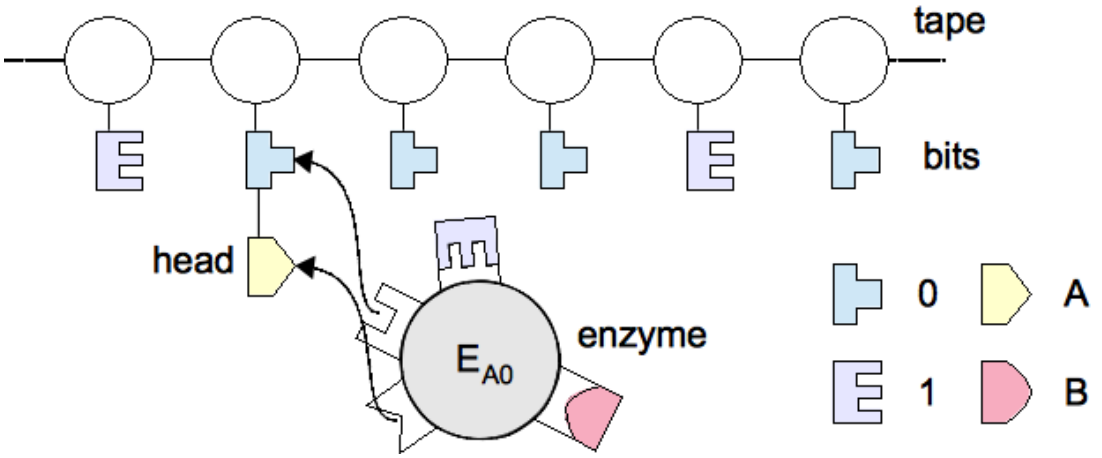
Molecular Machines and Turing Tapes

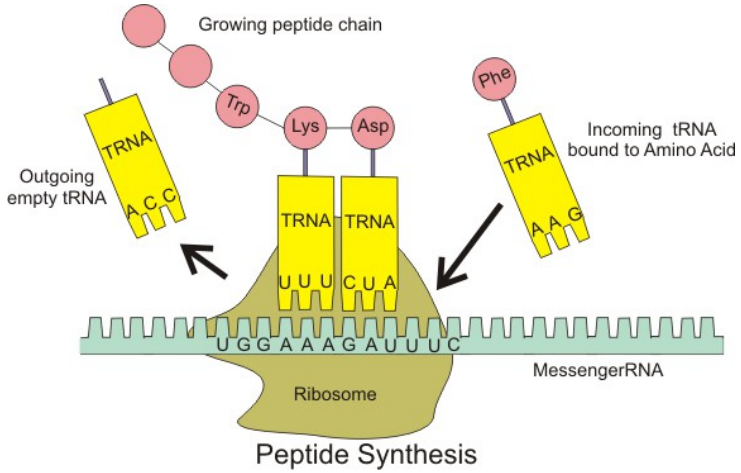
- Natural and synthetic information processing on multiple substrates

Turing Machine

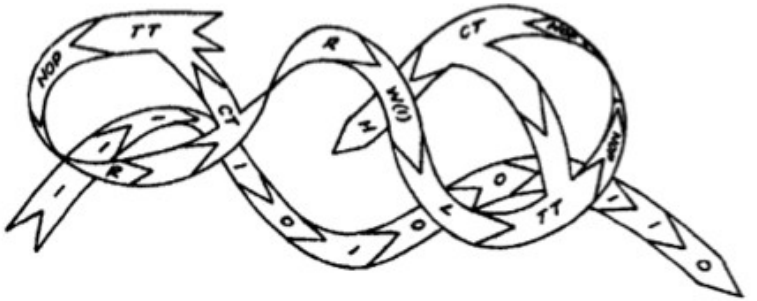


Enzymatic Turing Machine [Bennett1985]



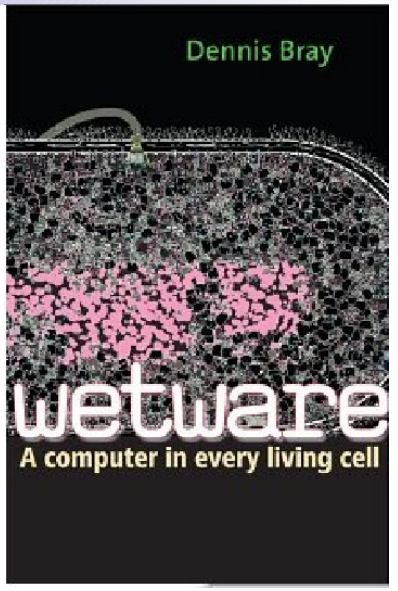


mRNA translation in a ribosome [Wikipedia]



Artificial Molecular Machine [Laing1975]

Computing with Artificial Chemistries

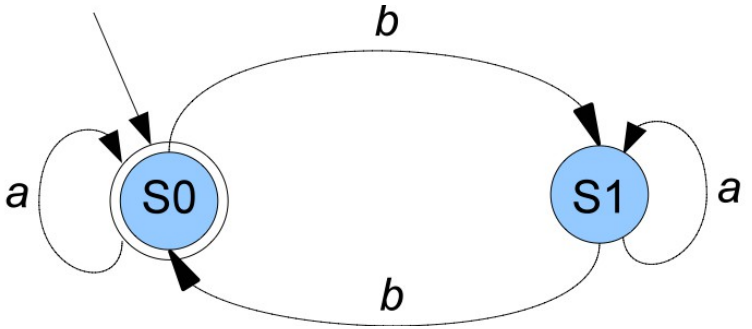


- Information processing occurs in nature, in a self-organizing way
  - How to harness these natural processes for our benefit: **engineer and control self-organization**
- In silico, in vitro and in vivo chemical computing
  - ACs for the modelling and simulation of wet biochemical computers
  - ACs as a metaphor for distributed and parallel computing
- Chemical computing inherently faces emergent phenomena
  - **Emergent computing** [Banzhaf1996][Forrest1990]
  - Properties at the higher levels emerge from interactions at the lower levels: molecular collisions, reactions, cell-to-cell communication...
  - Less intuitive for computer scientists: *"think chemically"*
    - **parallelism** at the microscopic scale (reactions in parallel)
    - **dynamical system** behavior at the macroscopic scale [Lones2014][Stepney2012]
    - program and data encoded in molecules
    - communication via molecules: cell signalling networks
    - compute by concentration changes, steer the flow of molecules

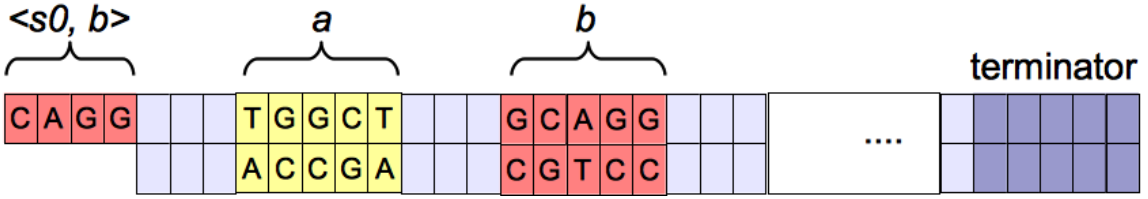
Molecular Automaton

[Benenson2003]
[Shapiro2006]

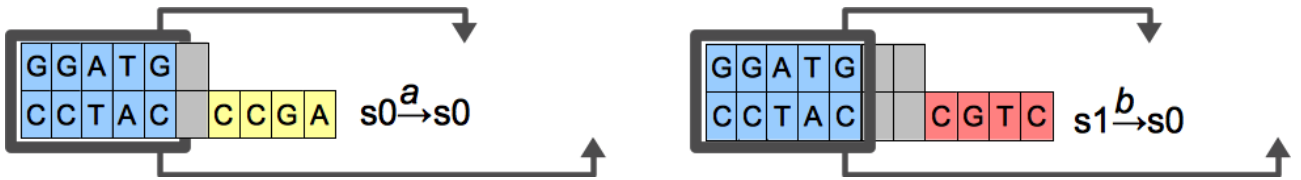
- Implementation of a Finite State Machine using DNA and enzymes
- Proof-of-concept: simple FSM with 2 states and 2 possible input symbols



- Ingredients:
  - DNA "tape" with input symbols encoded as short segments



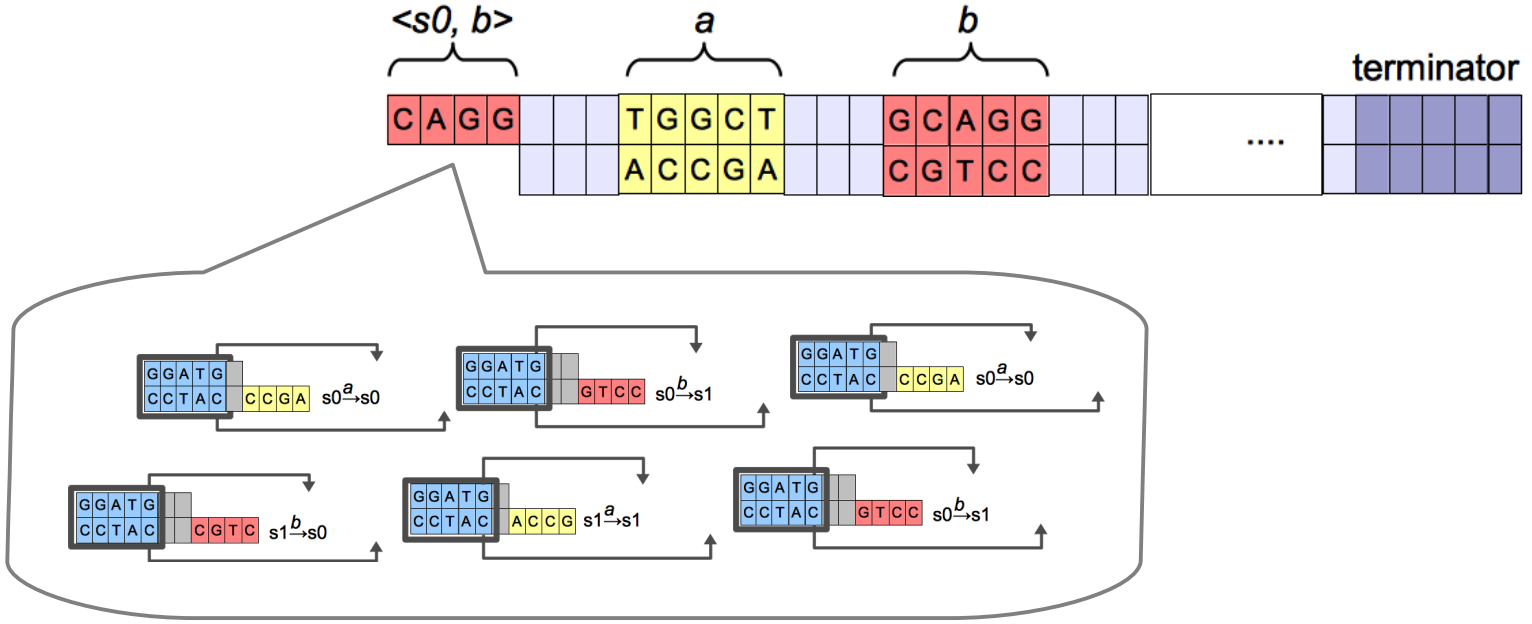
  - FokI enzyme loaded with DNA fragments:
    "cuts tape" to expose next state and symbol



Molecular Automaton

[Benenson2003]
[Shapiro2006]

- Example operation of enzyme on DNA input “tape”
  - DNA sticky end: current state and input symbol combination

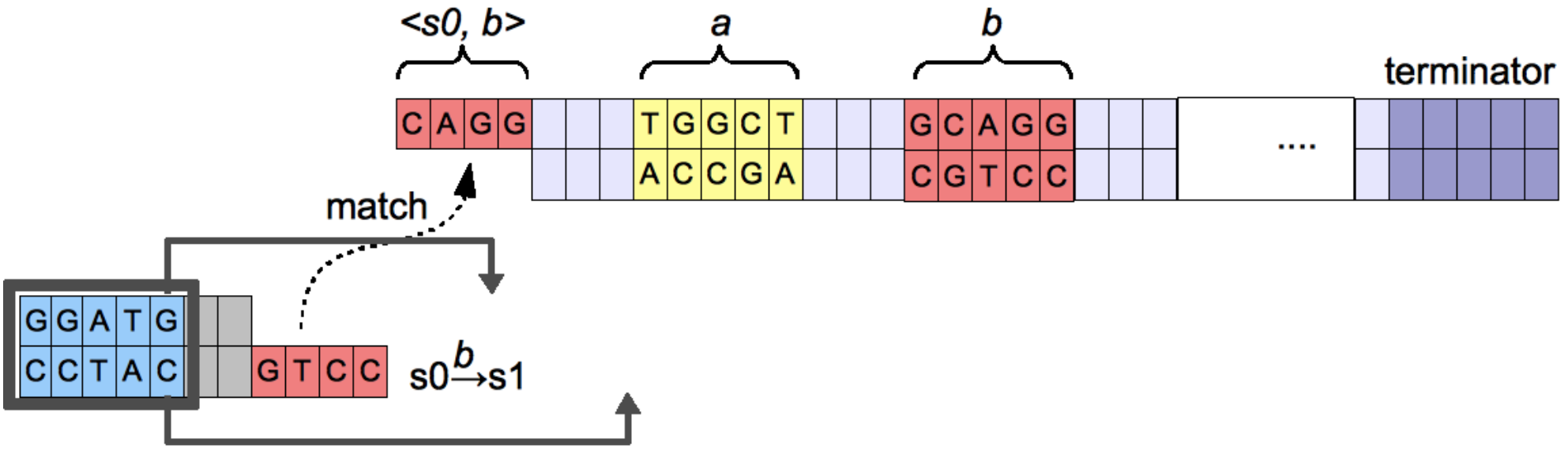


  - several enzyme-DNA complexes compete to bind to the exposed DNA sticky end
  - only those complementary to the single-stranded sticky end can bind in a stable way

Molecular Automaton

[Benenson2003]
[Shapiro2006]

- Example operation of enzyme on DNA input “tape”

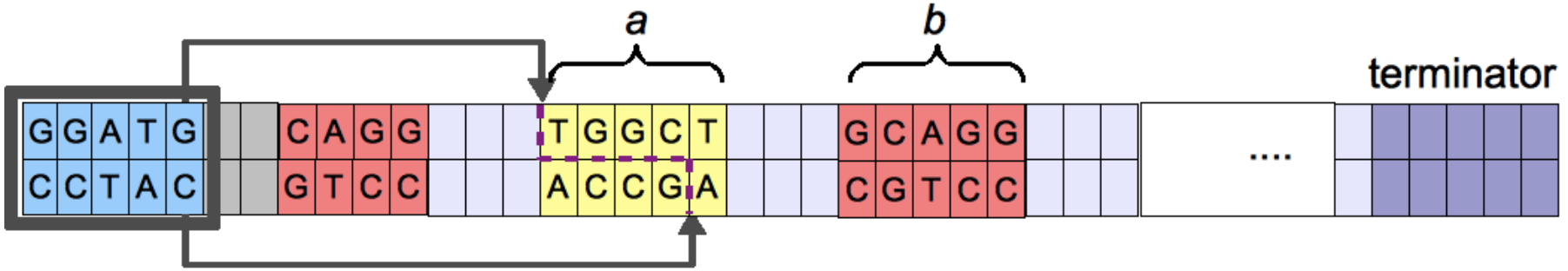


“winning” enzyme-DNA complex binds to complementary DNA fragment representing input symbol

Molecular Automaton

[Benenson2003]
[Shapiro2006]

- Example operation of enzyme on DNA input “tape”

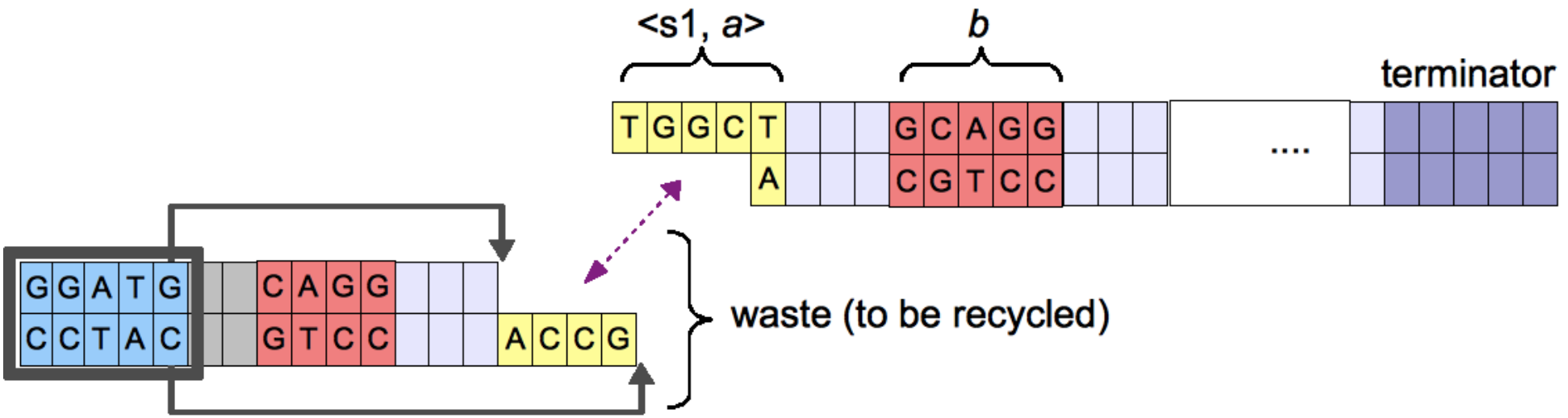


enzyme cleaves DNA at “scissor” position
(transition to the next state starts)

Molecular Automaton

[Benenson2003]
[Shapiro2006]

- Example operation of enzyme on DNA input “tape”

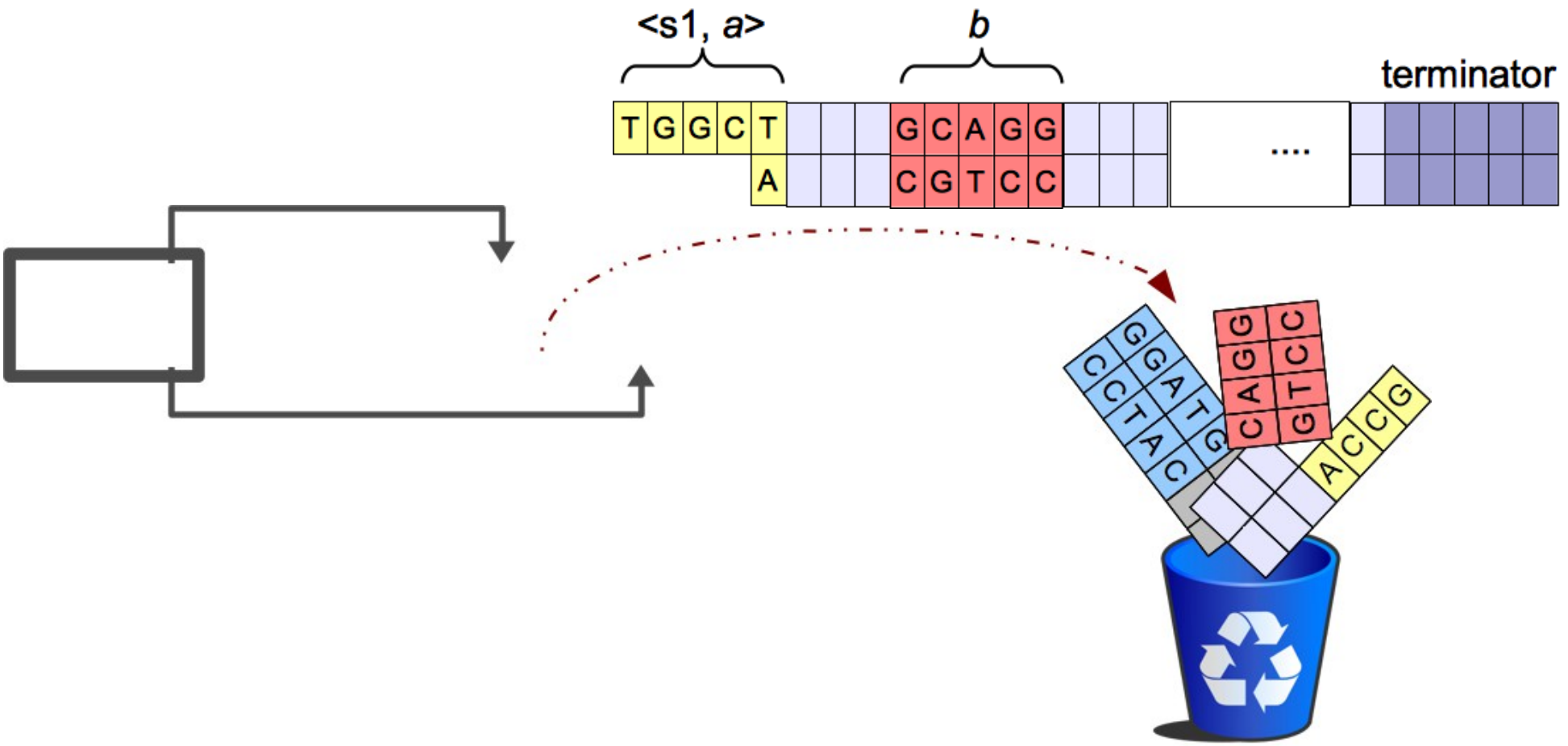


cleaved portion is removed from input tape and discarded:
next input symbol is exposed (transition to the next state is complete)

Molecular Automaton

[Benenson2003]
[Shapiro2006]

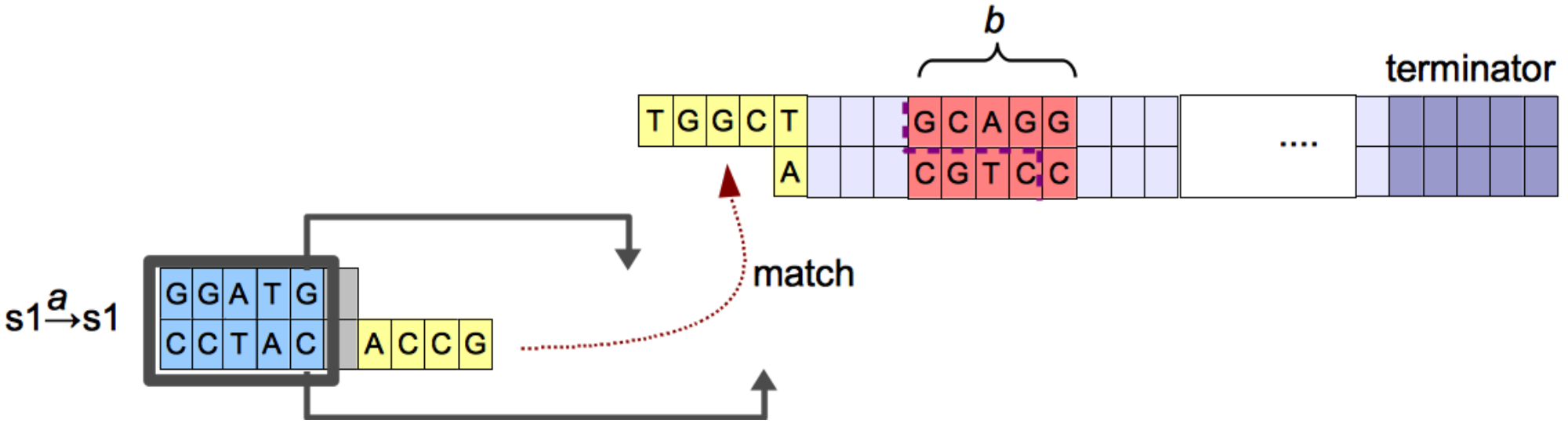
- Example operation of enzyme on DNA input “tape”



Molecular Automaton

[Benenson2003]
[Shapiro2006]

- Example operation of enzyme on DNA input “tape”

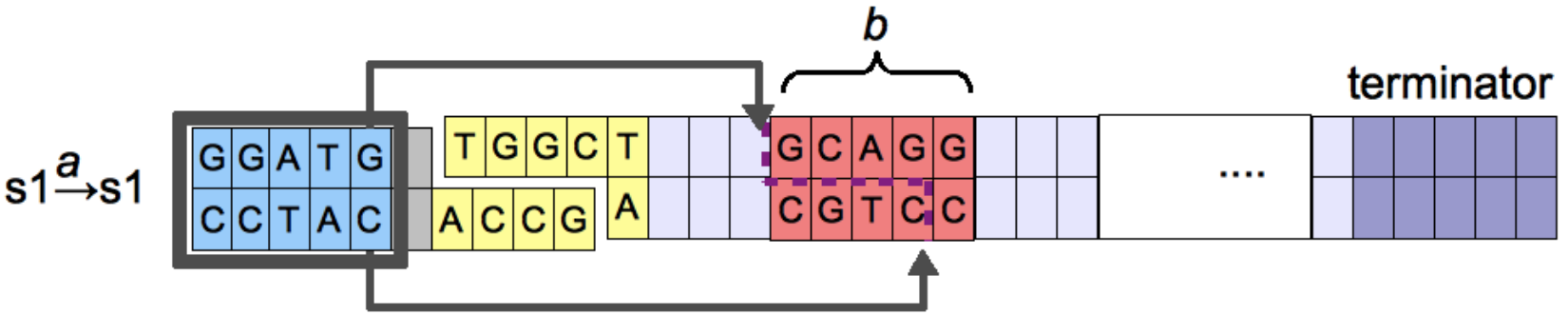


another enzyme-DNA complex binds to exposed input symbol, and cleaves the DNA input strand at the marked positions

Molecular Automaton

[Benenson2003]
[Shapiro2006]

- Example operation of enzyme on DNA input “tape”

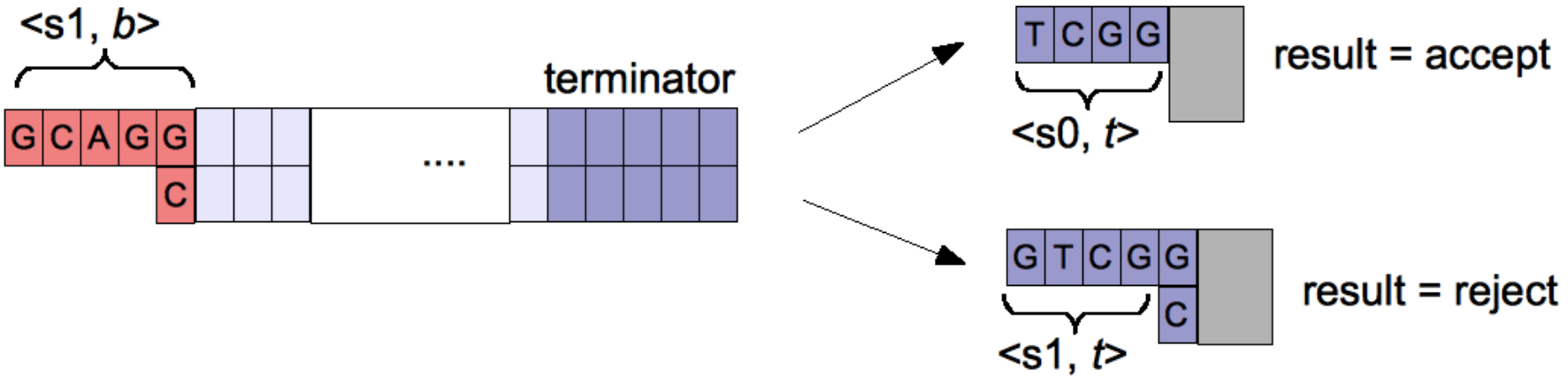


another enzyme-DNA complex binds to exposed input symbol, and cleaves the DNA input strand at the marked positions

Molecular Automaton

[Benenson2003]
[Shapiro2006]

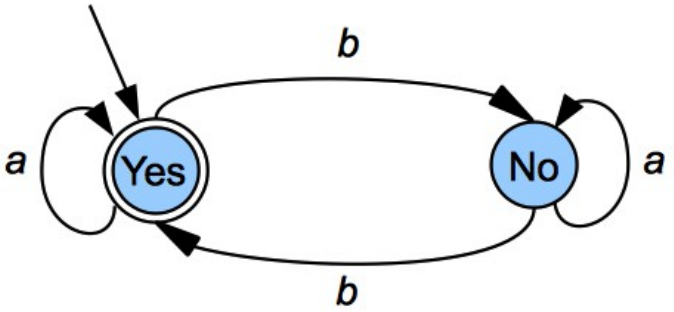
- Example operation of enzyme on DNA input “tape”



the computation proceeds until a terminator symbol is found:
the configuration of the terminator strand determines the output

Molecular Automaton

- Potential application: “DNA doctor in a cell” [Shapiro2006]
  - disease diagnosis: probabilistic operation due to competing biochemical pathways inside a cell, and fluctuating concentrations of molecules that trigger state transitions



- Computer simulation using an Artificial Chemistry based on pattern matching and recombination [Tominaga2007]

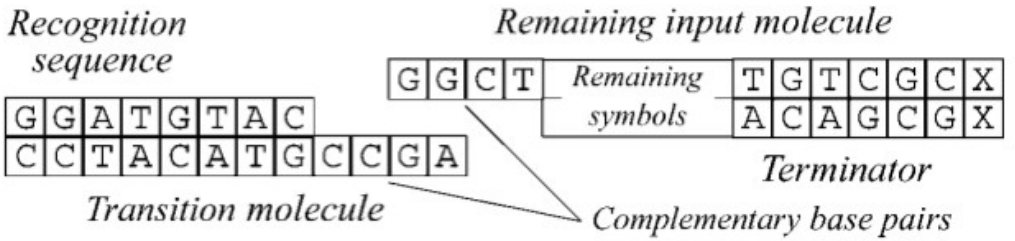
$S_0 \xrightarrow{a} S_0$: 0#GGATGTAC/0#CCTACATGCCGA/

$S_0 \xrightarrow{b} S_1$: 0#GGATGACGAC/0#CCTACTGCTGGTCG/

$S_1 \xrightarrow{a} S_1$: 0#GGATGTCG/0#CCTACAGCGACC/

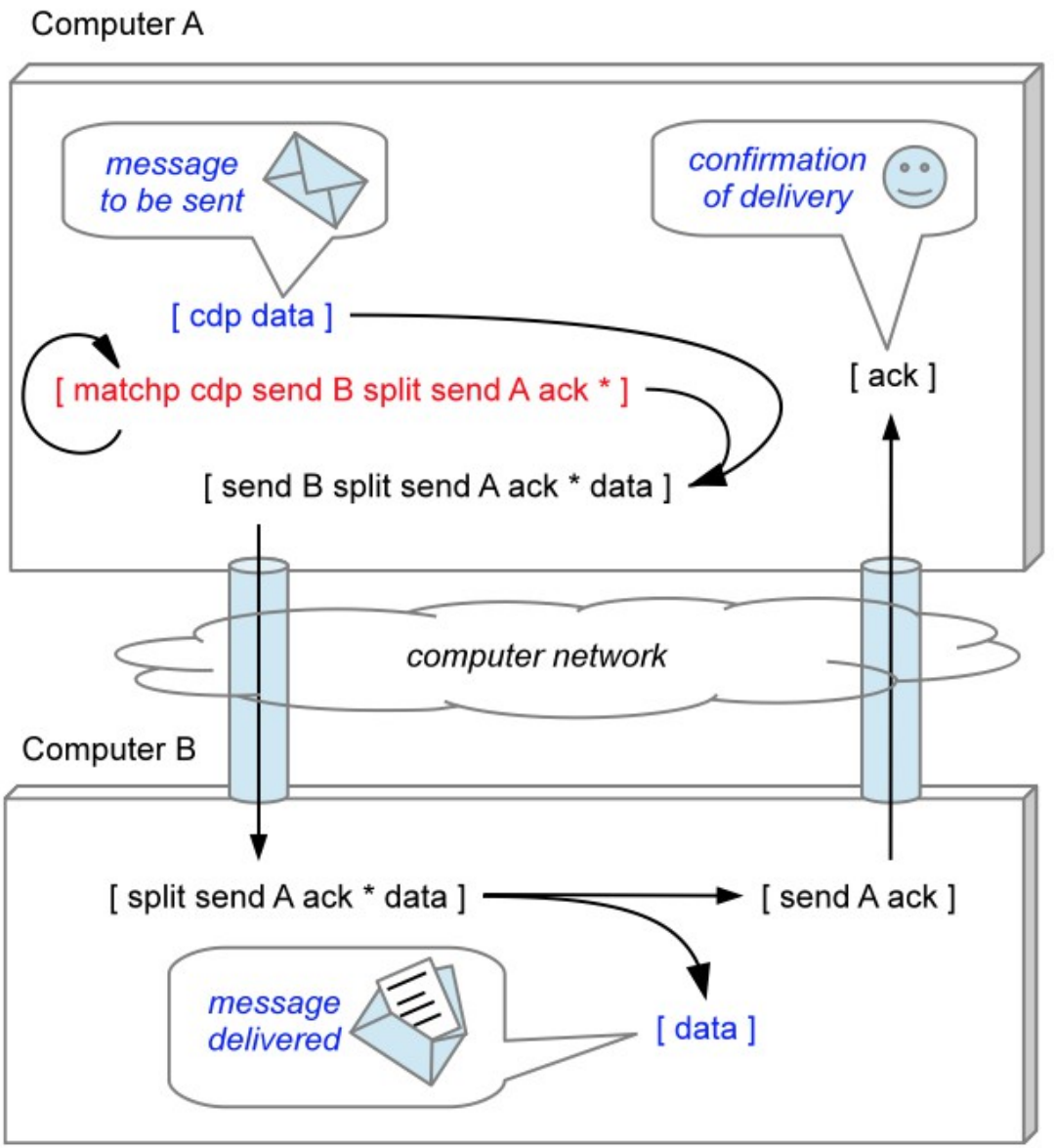
$S_1 \xrightarrow{b} S_0$: 0#GGATGG/0#CCTACCGCGT/

0#*0/0#*1CCGA/ + 0#GGCT2*/4#3*/ → 0#*0GGCT2*/0#*1CCGA3*/



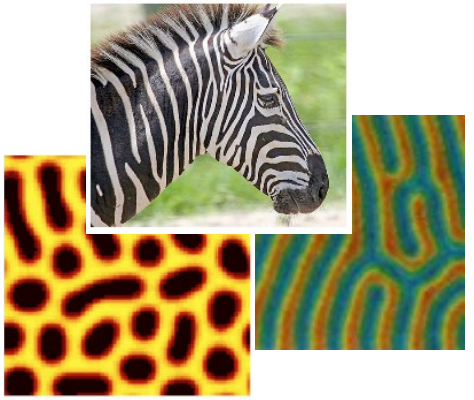
Fraglets: An AC for Computer Networks

- Fraglets programming language [Tschudin2003]
  - fraglet = computation fragment
    molecule = string of symbols = set of instructions and data
  - chemical reaction = bind to matching tag (head symbol), consume it, expose next symbol
  - constructive AC
- Automatic evolution of communication protocols for computer networks [Yamamoto2005]

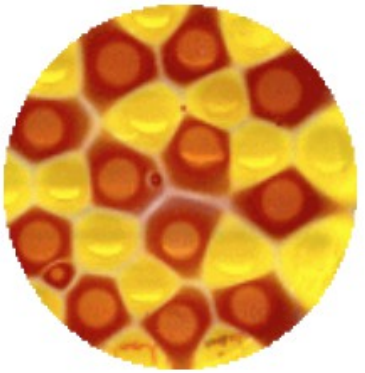


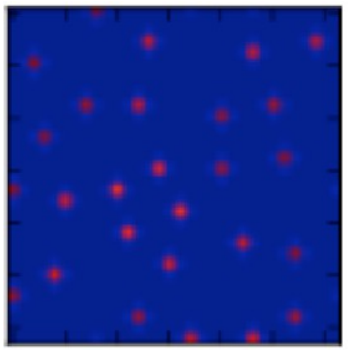
Computing with Reaction-Diffusion

- Turing patterns [Turing1952]
  - Morphogens: chemicals that diffuse and react through tissue
  - Equilibrium instability leads to pattern formation: spots, stripes, waves
- Spatial AC, non-constructive
- Applications:
  - Reaction-diffusion computers [Adamatzky2005]
  - Models of distributed computation inspired by chemistry



Voronoi diagrams
[Adamatzky2005]





evolution of decentralized
cluster head election
in sensor networks,
on GPU hardware
[Yamamoto2011]

BZ reaction for
image processing
[Kuhnert1989]

Contents

- Artificial Chemistries (ACs) in a Nutshell
  - Wet (*"in vitro", "in vivo"*) vs. virtual (*"in silico"*) ACs
  - Well-mixed vs. spatial ACs
  - Constructive vs. nonconstructive ACs
  - Simulation algorithms
- Computing with Artificial Chemistries
  - In silico, in vitro and in vivo chemical computing
  - Distributed computing and self-organization
- **PyCellChemistry software package**
  - `www.artificial-chemistries.org`
- Summary and Outlook

An Artificial Chemistry in Python

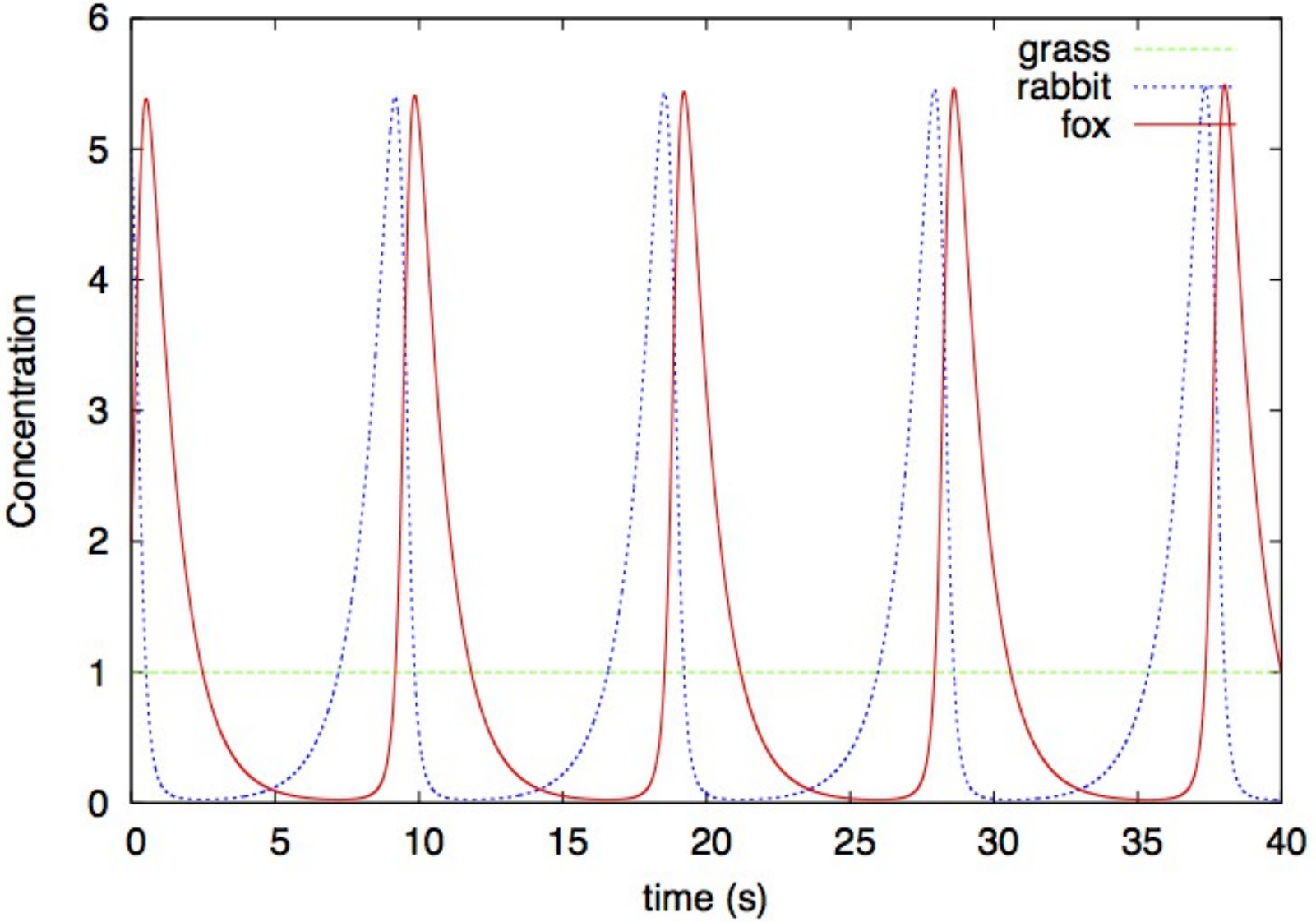
- **PyCellChemistry**: Python package to let users program their own ACs
  - **`www.artificial-chemistries.org`**
- Basic system:
  - multisets (bags) of molecules
  - chemical reactions
  - conversion from chemical reactions to ODE/PDE and Gillespie SSA
  - hierarchical cell compartments
- Example ACs:
  - basic: chameleons, prime number chemistry, matrix chemistry
  - biochemical circuits: dimerization, logistic growth, repressilator
  - ecology and evolution: Lotka-Volterra, quasispecies, NK landscapes
  - distributed & parallel computing: molecular TSP, fraglets, disperser
  - spatial ACs: reaction-diffusion

A Non-Constructive AC: Lotka-Volterra

```
class LotkaVolterra:
    def __init__( self, usestoch ):
        reactionstrs = [
            "rabbit + grass --> 2 rabbit + grass , k=1",
            "fox + rabbit   --> 2 fox             , k=1",
            "fox            -->                   , k=1" ]
        if usestoch:
            self.reactor = GillespieVessel(nav=40)
        else:
            self.reactor = WellStirredVessel()
        self.reactor.parse(reactionstrs)
        self.reactor.deposit('rabbit', 5.0)
    ...
    def run( self ):
        while (not self.extinct() and not self.exploded() and \
               self.reactor.vtime() <= 40.0):
            self.reactor.integrate(dt=0.001)
```

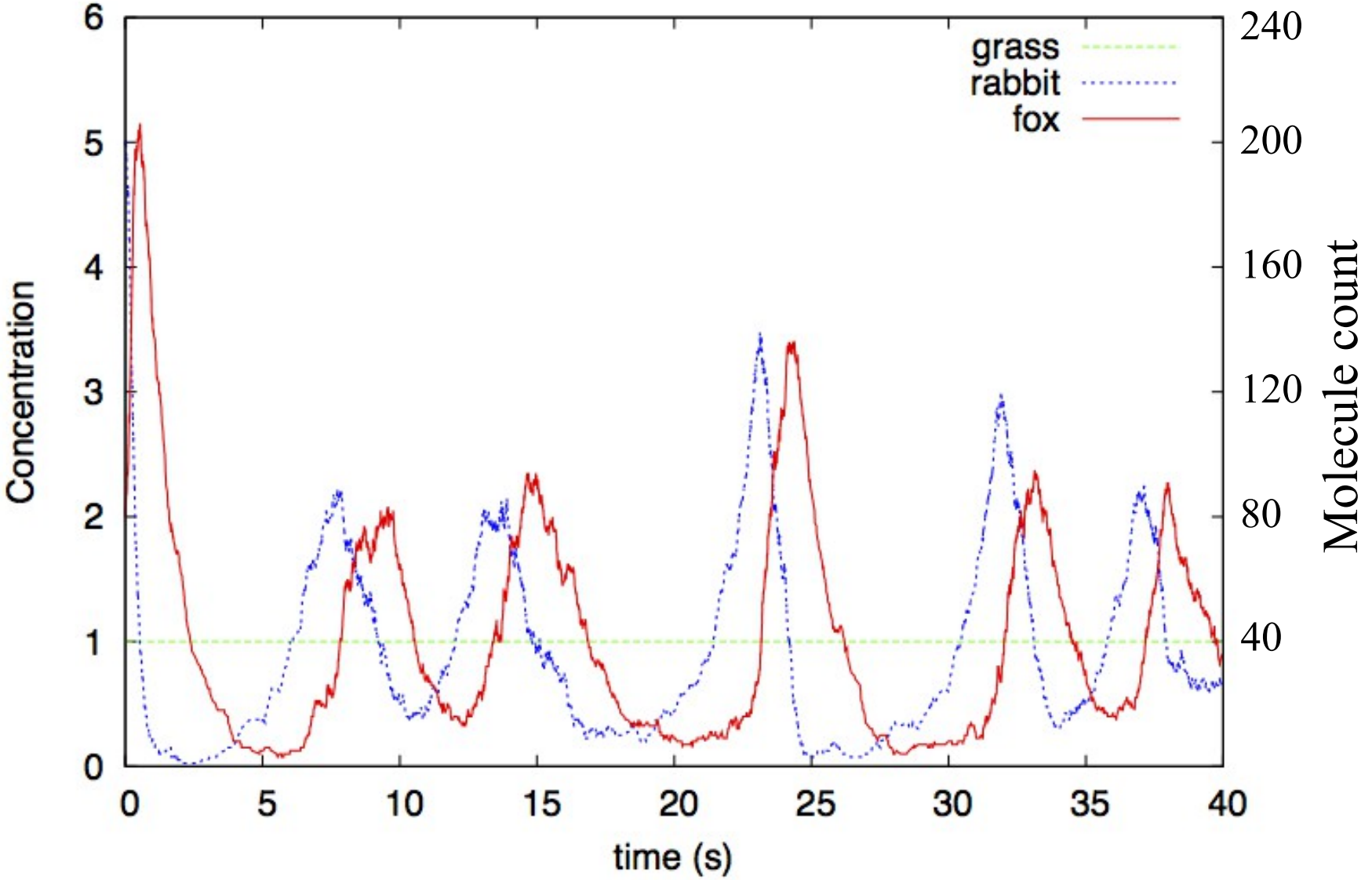
Lotka-Volterra: Deterministic vs. Stochastic

- Deterministic simulation via ODE integration:



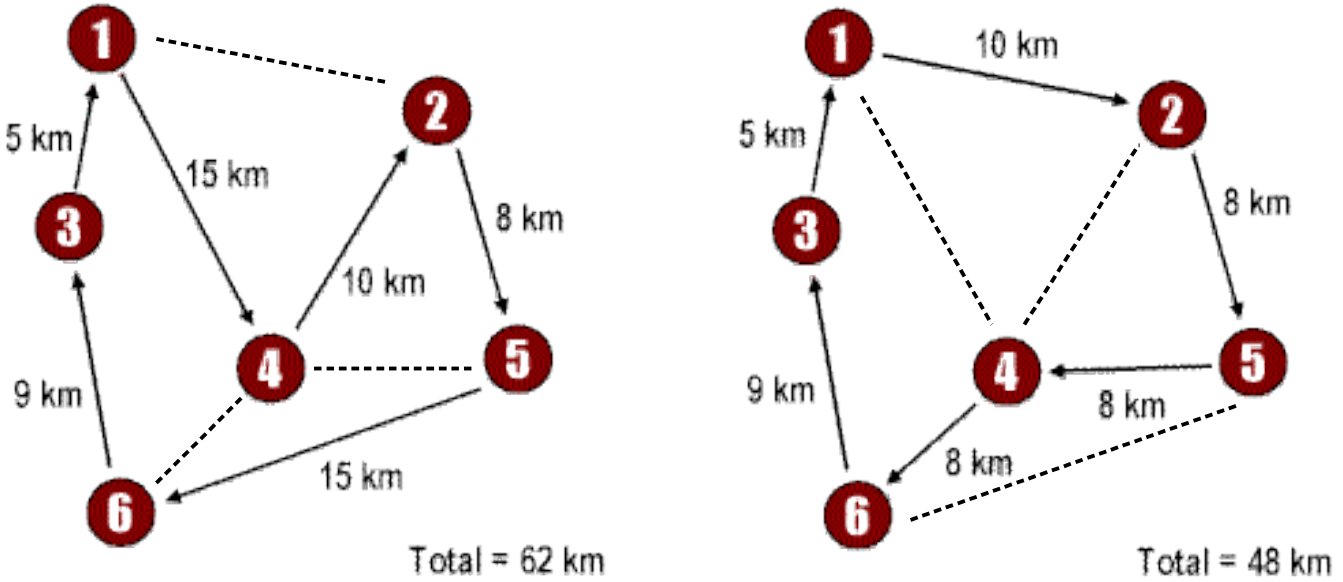
Lotka-Volterra: Deterministic vs. Stochastic

- Stochastic simulation via Gillespie SSA for $V = 40 / N_A$:



A Constructive AC: The Molecular TSP

- Traveling Salesman Problem (TSP):
  - find the tour of minimum cost that visits all the cities on a map
  - use only the available roads
  - visit each city only once
  - known to be NP-hard:
    - cannot be solved in general within a polynomial number of operations
    - typically heuristic algorithms are used: find approximate solutions

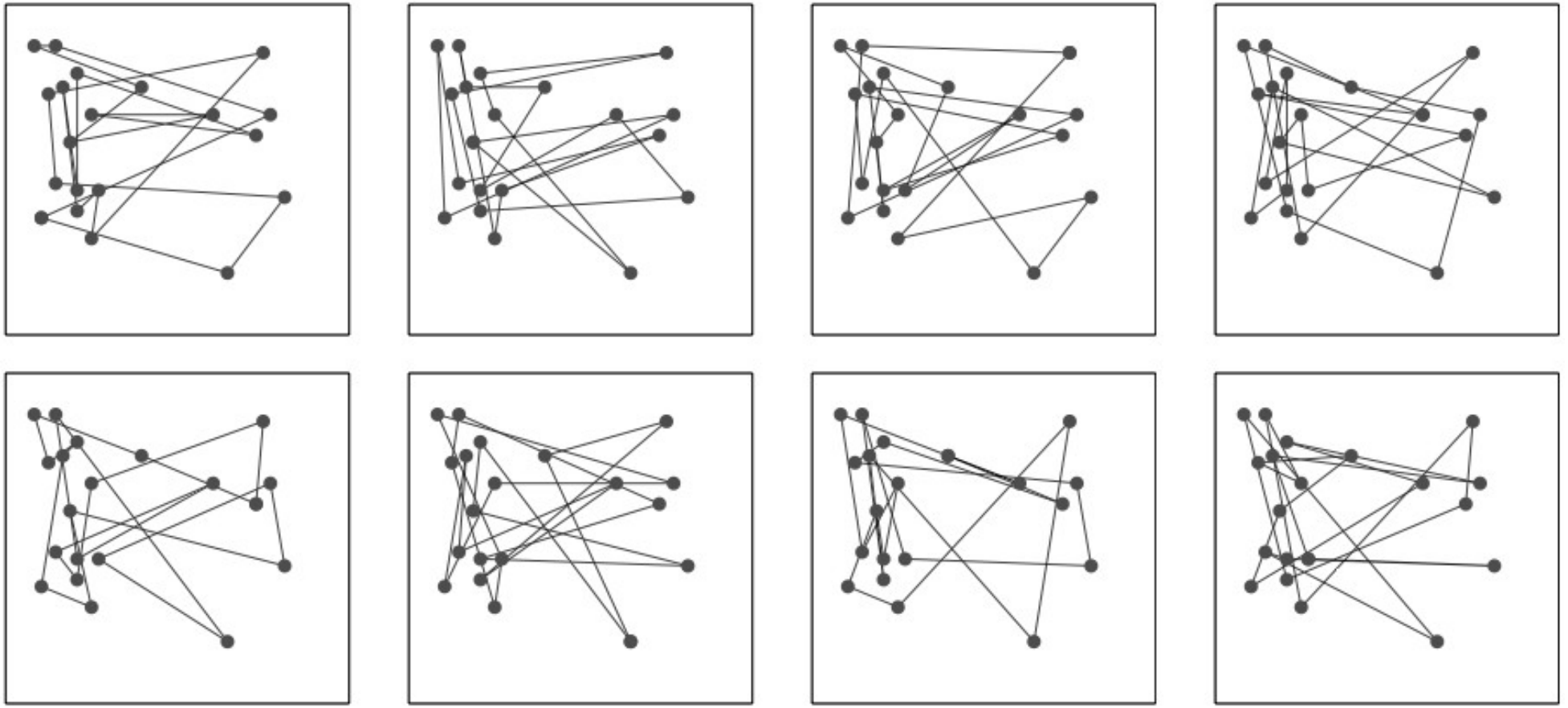


A Constructive AC: The Molecular TSP

- **Molecular TSP** [Banzhaf1990]: TSP heuristic inspired by chemistry
  - 2 types of molecules: machines and tours
    - tour: list of cities in the order they are visited, e.g. [1 2 5 4 6 3 1]
  - Machines ("enzymes") operate on tours ("substrates")
    - E-machine: swaps two random cities in a tour
    - C-machine: cuts a tour segment and pastes it elsewhere in tour
    - I-machine: cuts and inverts the segment before pasting it
    - R-machine: recombination (crossover) between 2 tours
  - Start with a "chemical soup" of random tours
  - Machines operate on tours independently (potentially in parallel)
    - draw 1 random molecule (2 for R-machine), perform operation
    - evaluate cost of each tour (educts and products)
    - inject best tour (2 best for R-machine) into soup, discard rest
  - Result: progressive selection of best tours

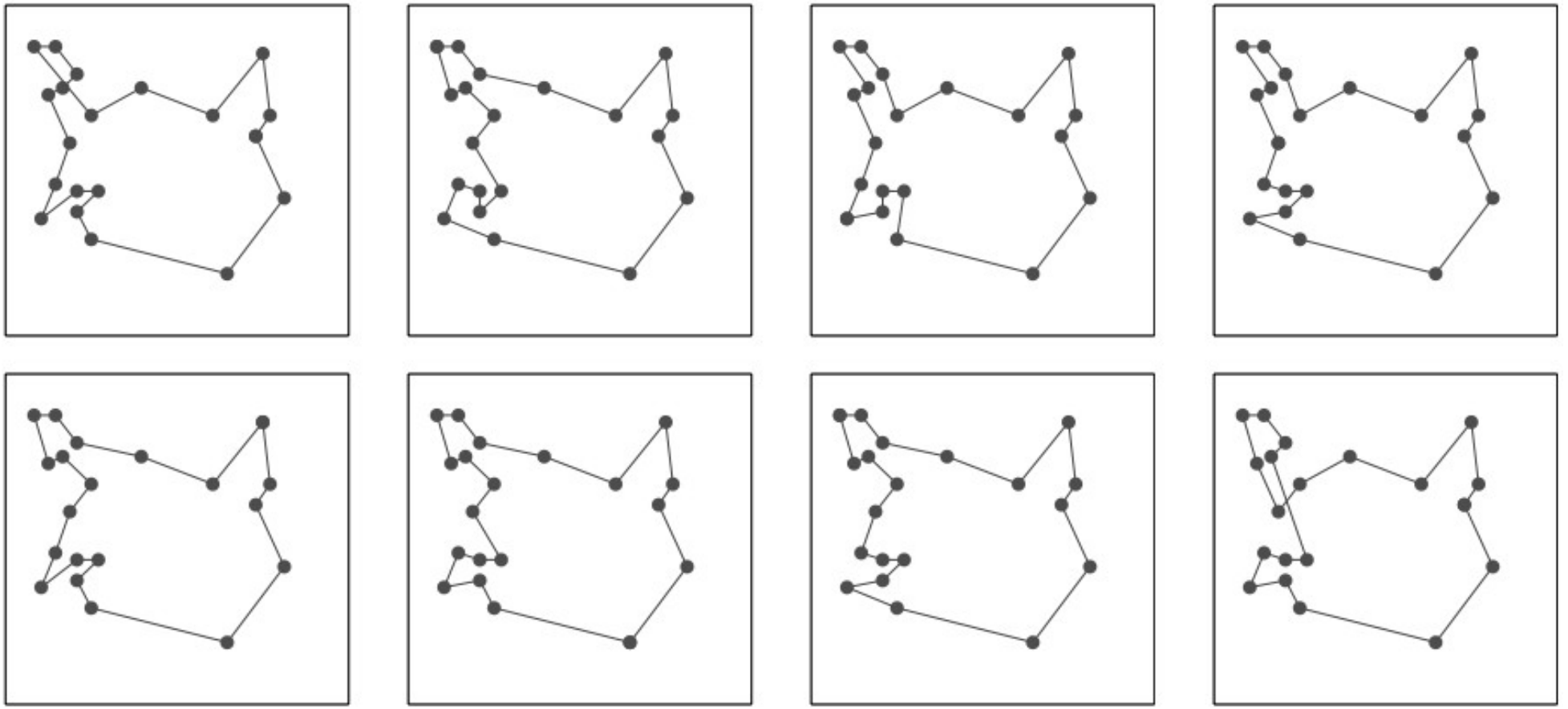
Molecular TSP: Initial Population

- Some random tours selected out of a population of 100 molecules:



Molecular TSP: After 4000 Generations

- Random tours selected out of the final population of 100 molecules:

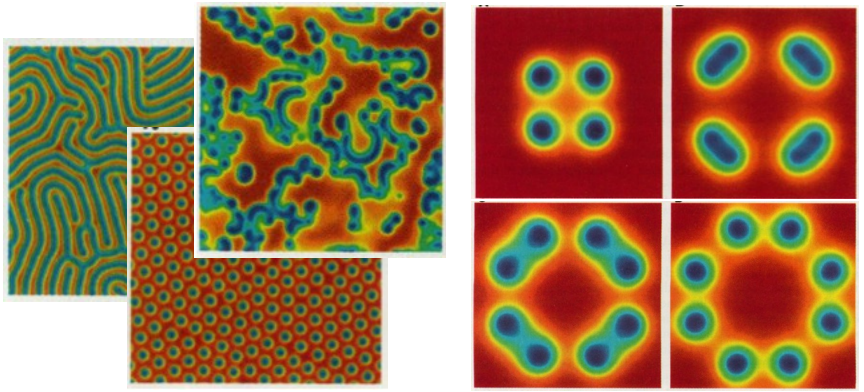


Molecular TSP in PyCellChemistry

```
class MolecularTSP( HighOrderChem ):
    def __init__( self, ncities ): ...
        tsp = TSPgraph(ncities, ...)  # create road map;
        for i in range(popsize):      # produce random tours:
            mol = self.randomMolecule()  # each tour is a molecule
            self.mset.inject(mol)        # injected in reactor;
        rule = 'self.exchangeMachine(%s)'   # machines are reaction
        self.rset.inject(rule, count)       # rules in same reactor
        rule = 'self.cutMachine(%s)'
        self.rset.inject(rule, count)
    ...
    def run( self ): ...
        for gen in range(self.maxgen):
            for j in range(genops):
                self.iterate() # pick rules and tours for reaction
            (bfit, bmol) = self.bestMolecule()  # best of generation
```

Spatial ACs: Reaction Diffusion Demos

- Gray-Scott [Gray1990][Pearson1993]



V: activator (autocatalyst) U: substrate

$$U + 2V \xrightarrow{k_1} 3V \qquad \emptyset \xrightarrow{F} U$$

$$V \xrightarrow{k_2} \emptyset \qquad U \xrightarrow{F} \emptyset$$

$$\frac{\partial u}{\partial t} = -uv^2 - Fu + F + D_u \nabla^2 u$$

$$\frac{\partial v}{\partial t} = uv^2 - (F + K)v + D_v \nabla^2 v$$

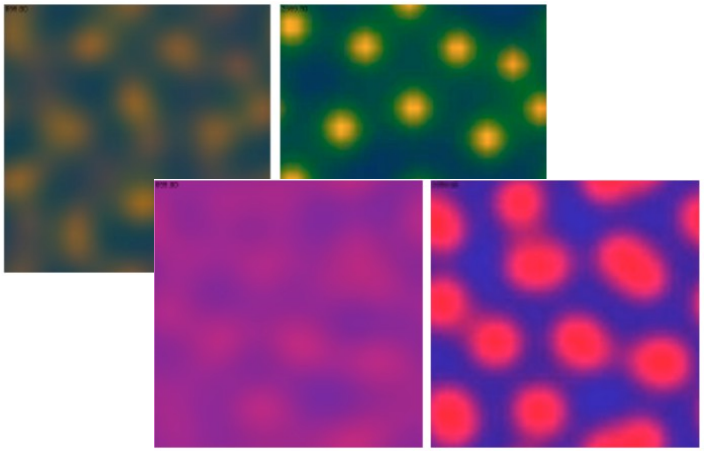
injection of U

reactions

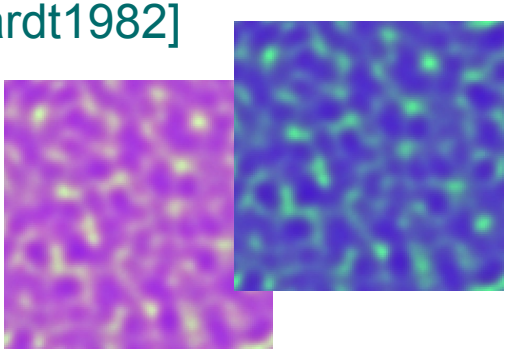
diffusion

Other demos:

- Activator-Inhibitor [Koch1994]
- Activator-Substrate [Meinhardt1982]

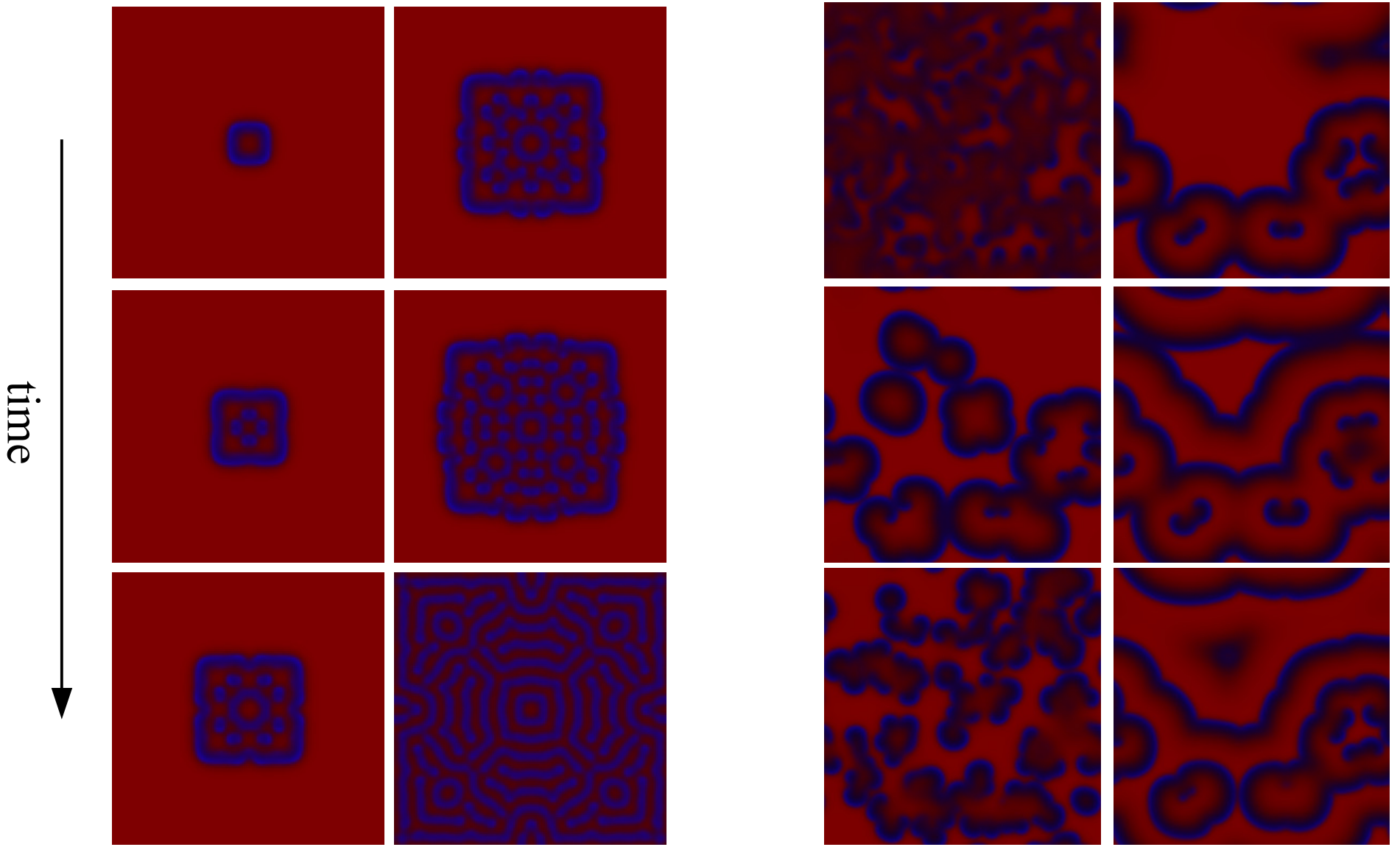


- Dichotomous branching [Meinhardt1982]



The Gray-Scott Demo

- Some example of patterns:



The Gray-Scott Demo

```
from ReactionDiffusion import *
class GrayScottDemo():
    def __init__( self ):
        reactionstrs = [ "U + 2 V --> 3 V",
                         "      V -->    ",
                         "        --> U  ",
                         "      U -->    "  ]
        self.rsys = ReactionDiffusionSystem(sizex, sizey, dx)
        self.rsys.parse(reactionstrs)
        self.rsys.set_coefficient(1, F+K)  ....  # kinetic coefs.
        self.rsys.set_diffcoef('U', DU)    ....  # diffusion coefs.
        self.rsys.deposit('V', initconc, posx, posy)  # initial cond.
    def run( self, finalvt=2000.0, dt=0.1 ):
        while (self.rsys.vtime() <= finalvt):
            self.rsys.integrate(dt)    # numerical PDE integration
            self.rsys.animate(...)     # animation in VPython
```

Summary and Outlook

- Brief overview of artificial chemistries with a few examples
  - Focus on computing applications
  - Many more ACs exist (our book contains almost 1000 citations)
- What can we learn from ACs? Are they just toy chemistries?
  - Engineering approach: learn how things work by building them: build complexity starting from the bottom up
    - PyCellChemistry as a software tool to facilitate learning, practice and experimentation with various ACs
  - Natural computing and emergent computation: computation is embedded in the chemical system
    - ACs make such tight association more clear
  - Understand emergent phenomena through mathematical analysis:
    - formalizing ACs: Chemical Organization Theory, RAF theory (reflexively autocatalytic sets), Chemical Reaction Automata (DNA computing), P systems, Brane calculi, ...

Summary and Outlook

- Towards a discipline of AC: challenges
  - AC field not mature yet:
    - borders still not clearly delimited, no coherent big picture
  - Barely scratching the surface of
    - commonalities among emergent phenomena (shared challenge with complex systems research)
    - computing with self-organization and emergence
    - how to move upwards in complexity, encapsulating the acquired emergent properties
- Future:
  - Tigher interdisciplinarity and integration between wet and virtual
  - Fuzzy line between virtual and real, more and more hybrid systems
  - Seamless programming: compile chemistry? chemical computers?

References

- [Adamatzky2005] A. Adamatzky, B. D. L. Costello, and T. Asai. Reaction-Diffusion Computers. Elsevier Science, 2005.
- [Adamatzky2010] A. Adamatzky. Physarum Machines: Computers from Slime Mould. World Scientific Series on Non-linear Science Series A, 2010.
- [Adamatzky2012] A. Adamatzky. Slime mold solves maze in one pass, assisted by gradient of chemo-attractants. IEEE Transactions on NanoBioscience, 11(2):131–134, June 2012.
- [Banzhaf1990] W. Banzhaf. The “molecular” traveling salesman. Biol. Cybern. 64:7-14, 1990.
- [Banzhaf1993] W. Banzhaf, Self-replicating sequences of binary numbers. Foundations I: General, Biological Cybernetics, August 1993, Volume 69, Issue 4, pp 269-274.
- [Banzhaf1996] W. Banzhaf, P. Dittrich, and H. Rauhe. Emergent computation by catalytic reactions. Nanotechnology, 7:307-314, 1996.
- [Benenson2003] Y. Benenson, R. Adar, T. Paz-Elizur, Z. Livneh, and E. Shapiro, “DNA molecule provides a computing machine with both data and fuel”, PNAS, 100 (2003), pp. 2191-2196.
- [Bennett1985] C. H. Bennett and R.Landauer. “The fundamental physical limits of computation”, Scientific American, 253(1):48–56, 1985.
- [Elowitz2000] M. B. Elowitz and S. Leibler. A synthetic oscillatory network of transcriptional regulators. Nature, 403:335–338, Jan. 2000.
- [Forrest1990] S. Forrest. Emergent computation: Self-organizing, collective, and cooperative phenomena in natural and artificial computing networks. Physica D: Nonlinear Phenomena, 42(1):1–11, 1990.

References

- [Gillespie1977] D. T. Gillespie: “Exact Stochastic Simulation of Coupled Chemical Reactions”. Journal of Physical Chemistry 81(25) (1977) 2340–2361.
- [Gray1990] P. Gray and S.K. Scott, "Chemical Oscillations and Instabilities: Nonlinear Chemical Kinetics", Oxford Science Publications, 1990.
- [Hanczyc2010] M. M. Hanczyc and T. Ikegami. Chemical basis for minimal cognition. Artificial Life, 16(3):233–243, 2010.
- [Hutton2009] T. J. Hutton. The organic builder: A public experiment in artificial chemistries and sel-replication. Artificial Life, 15(1):21–28, 2009.
- [Koch1994] A. J. Koch and H. Meinhardt, "Biological Pattern Formation: From Basic Mechanisms to Complex Structures", Reviews of Modern Physics, 66(4), 1994.
- [Kuhnert1989] L. Kuhnert, K. Agladze, and V. I. Krinsky. Image processing using light-sensitive chemical waves. Nature, 337:244–247, 1989.
- [Laing1975] R. Laing, “Some alternative reproductive strategies in artificial molecular machines”, Journal of Theoretical Biology, 54:63–84, 1975.
- [Lones2014] M. A. Lones et al. Artificial biochemical networks: Evolving dynamical systems to control dynamical systems. IEEE Trans. Evolutionary Computation, 18(2):145-166, 2014.
- [Meinhardt1982] H. Meinhardt, "Models of Biological Pattern Formation", Academic Press, 1982.
- [Meinhardt2003] H. Meinhardt, "The Algorithmic Beauty of Sea Shells", Springer, 2003.
- [Pearson1993] J. E. Pearson, "Complex Patterns in a Simple System", Science, volume 261, number 5118, July 1993, pages 189-192.

References

- [Rasmussen2003] S. Rasmussen, L. Chen, M. Nilsson, and S. Abe. Bridging nonliving and living matter. Artificial Life, 9(3):269–316, 2003.
- [Shapiro2006] E.Shapiro, Y.Benenson, “Bringing DNA Computers to Life, Scientific American, 2006.
- [Stepney2012] S. Stepney. Nonclassical computation: A dynamical systems perspective. In G. Rozenberg, T. Baeck and J. N. Kok (editors), Handbook of Natural Computing, chapter 59, pages 1979-2025. Springer, 2012.
- [Tominaga2007] K. Tominaga, T. Watanabe, K. Kobayashi, M. Nakamura, K. Kishi, and M.Kazuno. Modeling molecular computing systems by an artificial chemistry — Its expressive power and application. Artificial Life, 13(3):223–247, 2007.
- [Tschudin2003] C. Tschudin: "Fraglets - a Metabolistic Execution Model for Communication Protocols”, Proc. AINS, Menlo Park, USA, July 2003.
- [Turing1952] A. Turing, “The Chemical Basis of Morphogenesis”, Phil. Trans. Royal Society (B), 237:37-72, 1952.
- [Yamamoto2011] L. Yamamoto, P. Collet, W. Banzhaf, "Evolving Reaction-Diffusion Systems on GPU", Progress in Artificial Intelligence, Proc 15th EPIA, 2011 L. Antunes and H. Sofia Pinto (editors), Springer LNCS 7026, 2011, pp. 208-223.
- [Yamamoto2013] L. Yamamoto, P. Collet, and W. Banzhaf, "Artificial Chemistries on GPU" In: S. Tsutsui and P. Collet (editors), "Massively Parallel Evolutionary Computation on GPGPUs", Springer, pages 389-419, 2013.