Natural Computing

Lecture 16

Michael Herrmann
mherrman@inf.ed.ac.uk
phone: 0131 6 517177
Informatics Forum 1.42

11/11/2011

Molecular Computing

Overview

- Molecular computing
- Membrane computing

Video to last lecture:
http://www.youtube.com/watch?v=4PKjF7OumYo

Current developments

Synthetic biology: Creating new biological functions and systems for computation and medical applications

Self-assembly: On the molecular, supermolcular, as well as on the macroscopic scale.

Gross, R.; Dorigo, M.; , "Self-Assembly at the Macroscopic Scale," Proceedings of the IEEE , vol.96, no.9, pp.1490-1508, Sept. 2008

Molecular nanomachines: MAYA II plays TicTacToe

J. Macdonald et al.: Medium Scale Integration of Molecular Logic Gates in an Automaton. NANO LETTERS 2006 Vol. 6, No. 11 2598-2603

Autonomous molecular computers

Y. Benenson et al. An autonomous molecular computer. Nature 2004

Yurke, B.; Turberfield, A. J.; Mills, A. P., Jr; Simmel, F. C. & Neumann, J. L. (2000). "A DNA-fuelled molecular machine made of DNA". Nature 406 (6796): 605–609.

How to design the DNA pieces needed in the machines?

T. B. Kurniawan et al. (2008) An Ant Colony System for DNA sequence design based on thermodynamics. Proc. 4th IASTED ACST '08, 144-149.

Genetic Algorithms in DNA Computing

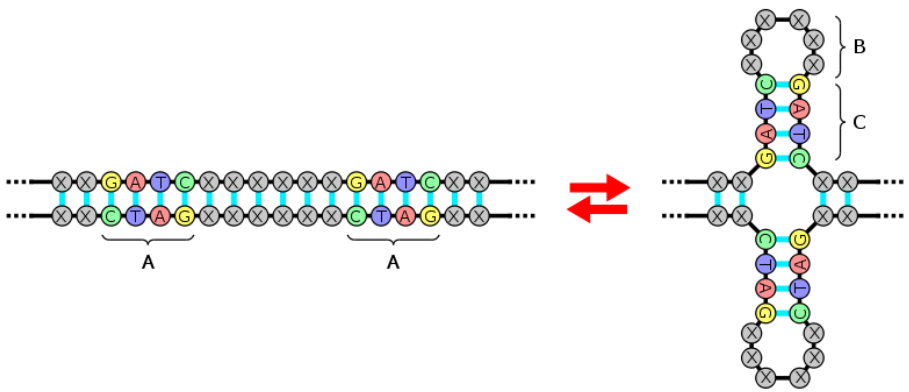
- Adleman's solution to the Hamiltonian path problem: checking all possible solutions (although "brute force", it was nevertheless a breakthrough in natural computing)
- New algorithms for a DNA computer?
- Life hasn't tested all possible combinations of genes, neither do GAs
- Introduce and exploit structure of the search space

$\Rightarrow$ DNA genetic algorithm

Z Ezziane: DNA computing: applications and challenges. Nanotechnology 17 (2006) R27–R39.

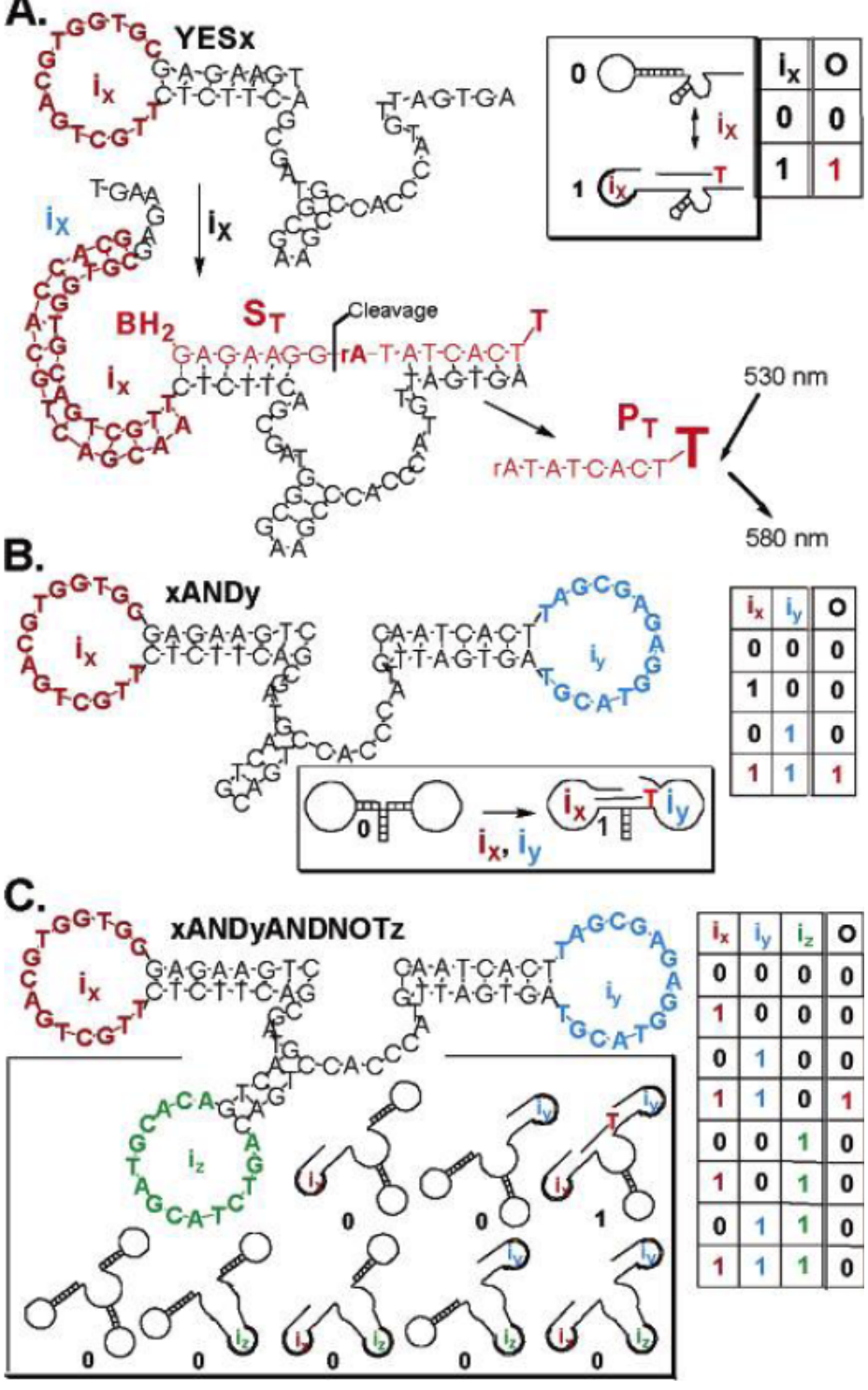
GA operators

- 100 different restriction enzymes are known, each of which cuts at its specific recognition site(s) often near "palindromes" (reverse complements) $\rightarrow$ 'hairpin' formation
- Mutation & recombination prepared restriction enzymes
- Self-replicating systems for producing offspring with shuffled components



- Selection based by affinity to an immobilizing motif (i.e. a protein encoding desired properties of the string)
- 30 nodes possible in the Hamiltonian path problem

J.A. ROSE et al.: A DNA-based in vitro Genetic Program. Journal of Biological Physics 28: 493–498, 2002



i_x	O
0	0
1	1

i_x	i_y	O
0	0	0
1	0	0
0	1	0
1	1	1

i_x	i_y	i_z	O
0	0	0	0
1	0	0	0
0	1	0	0
1	1	0	1
0	0	1	0
1	0	1	0
0	1	1	0
1	1	1	0

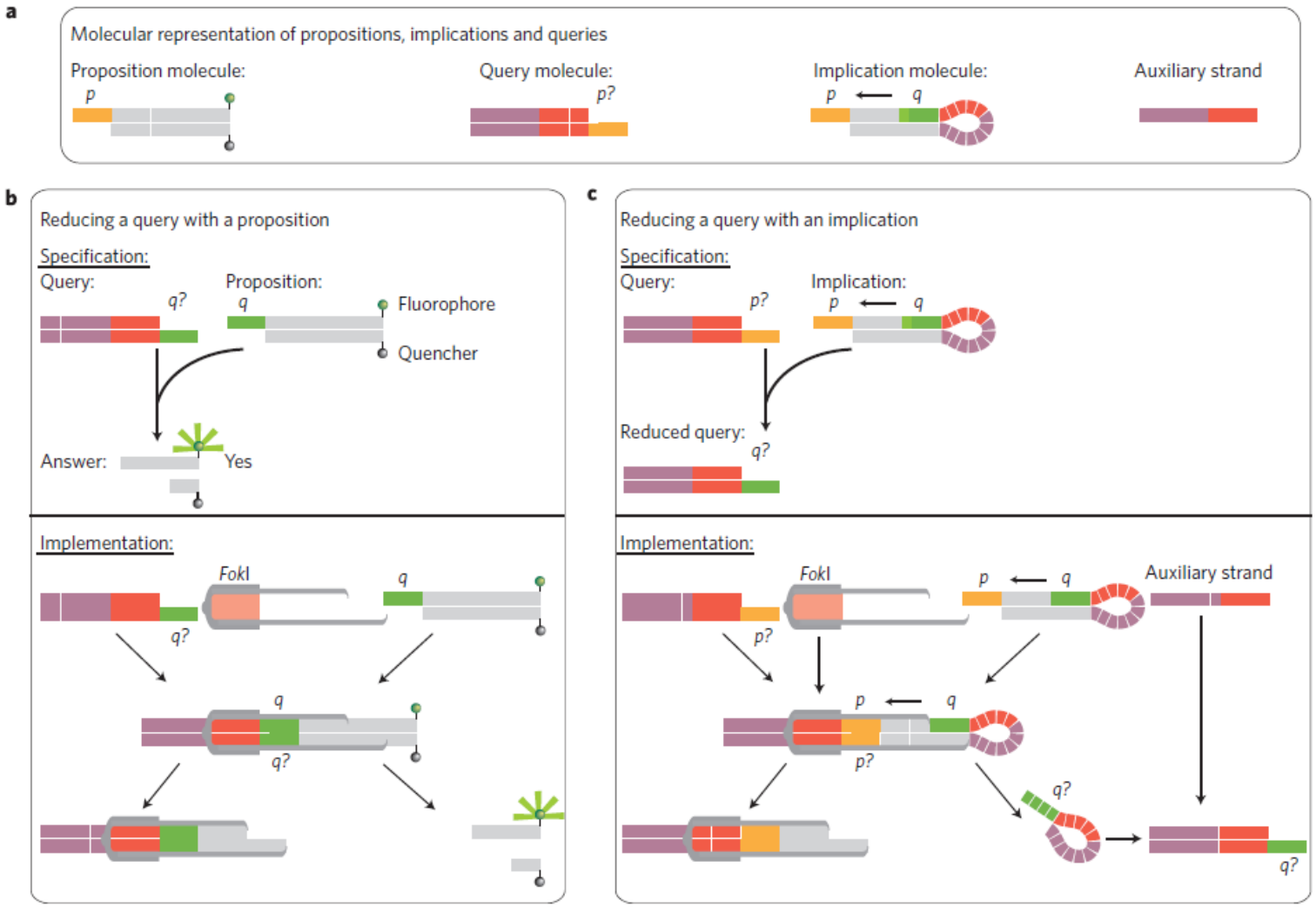
J. Macdonald et al.: Medium Scale Integration of Molecular Logic Gates in an Automaton. NANO LETTERS 2006 Vol. 6, No. 11 2598-2603

Molecular implementation of simple logic programs

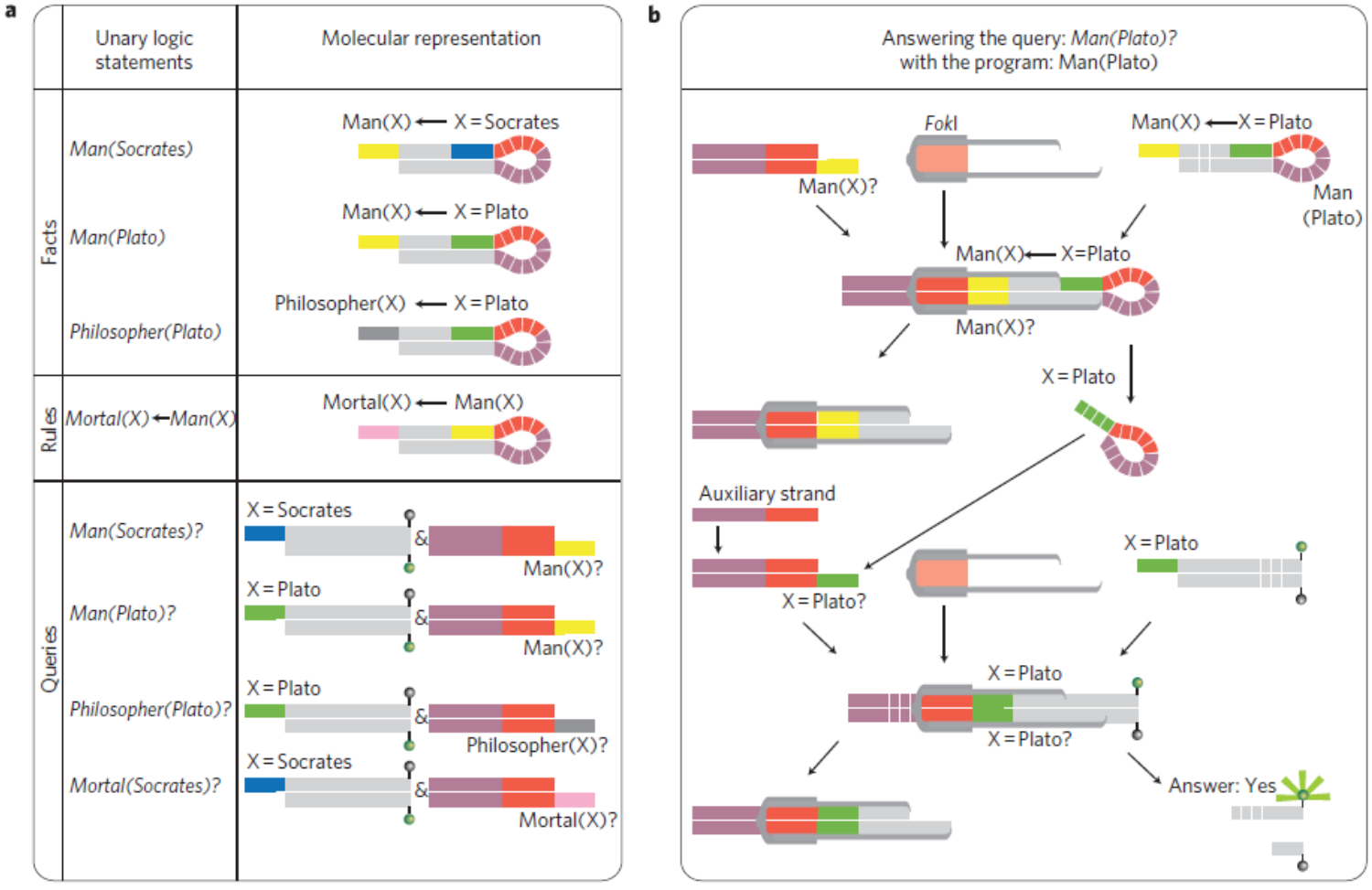
Programming using DNA molecules, e.g.

- Facts: Man(Socrates)
- Rules such as
  - Mortal(X)
  - Man(X)
  - (Every Man is Mortal)
- Answer queries such as
  - Mortal(Socrates)?
  - (Is Socrates Mortal?)
  - Mortal(X)?
  - (Who is Mortal?)

T. Ran, S. Kaplan & E. Shapiro: Molecular implementation of simple logic programs. Nature Nanotechnology 4, 642 - 648, 2009.



T. Ran, S. Kaplan & E. Shapiro: Molecular implementation of simple logic programs. Nature Nanotechnology 4, 642 - 648, 2009.

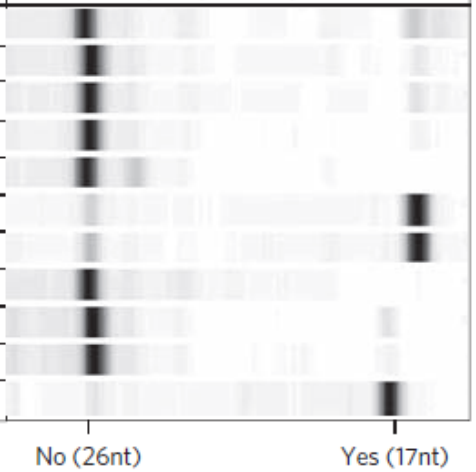


T. Ran, S. Kaplan & E. Shapiro: Molecular implementation of simple logic programs. Nature Nanotechnology 4, 642 - 648, 2009.

Molecular implementation of simple logic programs

b

	Program	Query	Capillary electrophoresis
1	Man(X) ← Greek(X), Greek(Socrates)	Mortal(Aristo)?	
2	Man(X) ← Greek(X), Greek(Plato)	Mortal(Plato)?	
3	Mortal(X) ← Man(X), Greek(Socrates)	Mortal(Aristo)?	
4	Mortal(X) ← Man(X), Greek(Plato)	Mortal(Plato)?	
5	Mortal(X) ← Man(X), Man(X) ← Greek(X), Greek (Plato)	Mortal(Aristo)?	
6	Mortal(X) ← Man(X), Man(X) ← Greek(X), Greek (Plato)	Mortal(Plato)?	
7	Mortal(X) ← Man(X), Man(X) ← Greek(X), Greek (Socrates)	Mortal(Aristo)?	
8	Mortal(X) ← Man(X), Man(X) ← Greek(X), Greek(Socrates)	Mortal(Plato)?	
9	Mortal(X) ← Man(X), Man(X) ← Greek(X), God(Zeus)	Mortal(Zeus)?	
10	God(Zeus)	Immortal(Zeus)?	
11	Immortal(X) ← God(X), God(Zeus)	Immortal(Zeus)?	



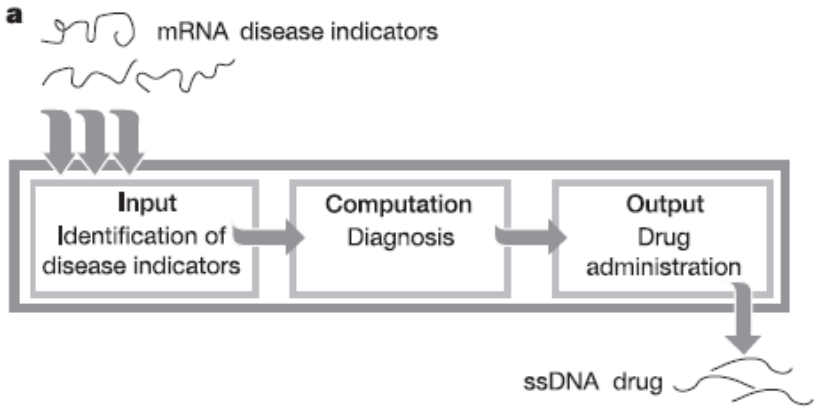
- All answers correct for the example of the previous page.
- The chart here shows that if some implication rules or facts are missing, answer is “no” (lanes 1,2,3,4,10)
- Reaction time (electrophoresis) is above one minute

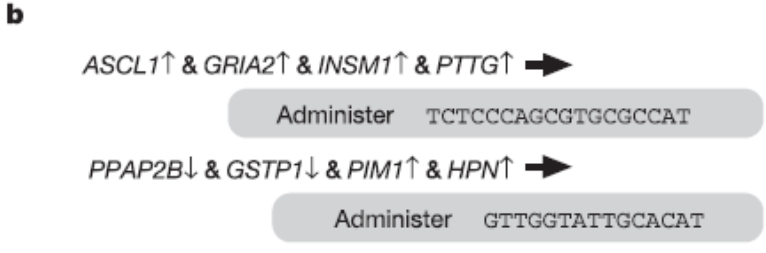
T. Ran, S. Kaplan & E. Shapiro: Molecular implementation of simple logic programs. Nature Nanotechnology 4, 642 - 648, 2009.

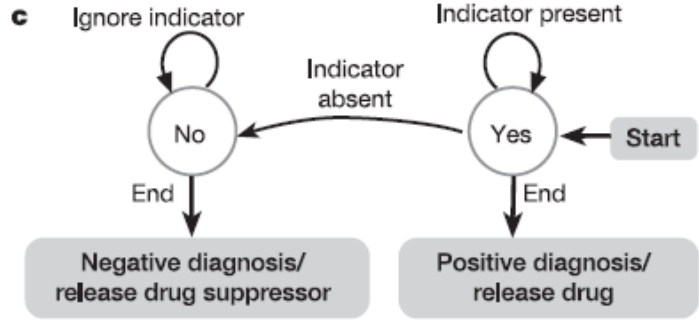
Applications of molecular computers

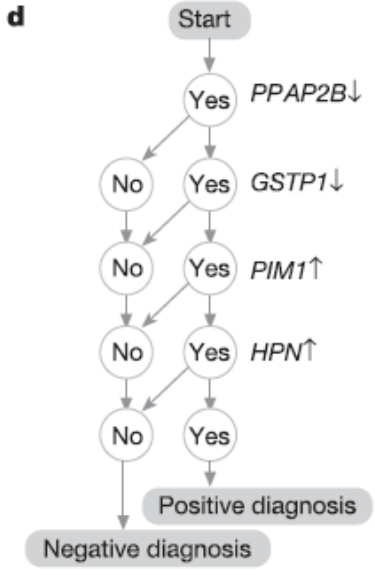
- Early biomolecular computers were human-operated for complex computational problems
- Here: input and output information in molecular form, a trillion computers per microlitre
- System for 'logical' control of biological processes: analyses the levels of messenger RNA species, and in response produces a molecule capable of affecting levels of gene expression.
- consists of three programmable modules:
  - computation module: a stochastic molecular automaton
  - input module, by which specific mRNA levels or point mutations regulate software molecule concentrations, and hence automaton transition probabilities
  - output module for controlled release of a short single-stranded DNA molecule

Y. Benenson et al. An autonomous molecular computer. Nature 2004


a
mRNA disease indicators
Input
Identification of disease indicators
Computation
Diagnosis
Output
Drug administration
ssDNA drug


b
ASCL1↑ & GRIA2↑ & INSM1↑ & PTTG↑
Administer TCTCCCAGCGTGCGCCAT
PPAP2B↓ & GSTP1↓ & PIM1↑ & HPN↑
Administer GTTGGTATTGCACAT


c
Ignore indicator
Indicator present
Indicator absent
No
Yes
Start
End
End
Negative diagnosis/ release drug suppressor
Positive diagnosis/ release drug


d
Start
Yes
PPAP2B↓
No
Yes
GSTP1↓
No
Yes
PIM1↑
No
Yes
HPN↑
No
Yes
Positive diagnosis
Negative diagnosis

An autonomous molecular computer

Y. Benenson et al. Nature 2004

- How does it work? States can be represented by DNA which can be "active" (single stranded) or "passive" (double stranded).
- Active strings can trigger other strings to become active by attaching to an overhang and then supersede and strip off one of the strands. The stripped-off strand is now active and may have a different overhang, i.e. a "computation" has taken place.
- Realised example: identify and analyse mRNA of disease-related genes associated with models of small-cell lung cancer and prostate cancer, and produce a single-stranded DNA molecule modelled after an anticancer drug
- Application: in vivo to biochemical sensing, genetic engineering and even medical diagnosis and treatment.

Synthetic Biology

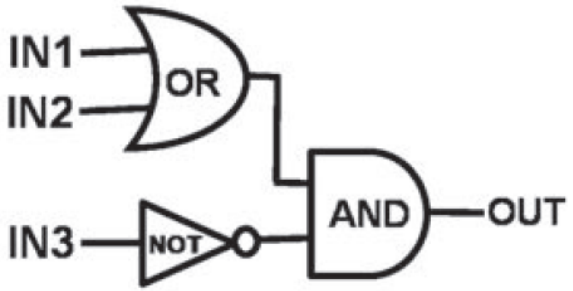
- Construction of molecular circuitry that can control biological systems or even diagnose and treat living cells from within.
- System of over 100 distinct DNA strands of 15 to 30 nucleotides whose binding and replication can be controlled to perform AND, OR, NOT, NAND, and NOR logic gates logic operations.
- The system allowed mathematical computation of square roots in a few hours
- Compiler could translate a logic circuit into its equivalent circuit built of DNA sequences
- The strategies used are scalable to build larger circuits, assure reliable digital behavior of the system, and suggest the possibility of embedding designed intelligent systems within biological systems.

Qian and Winfree (2011) Science 332:6034, 1196 (see also p. 1125)

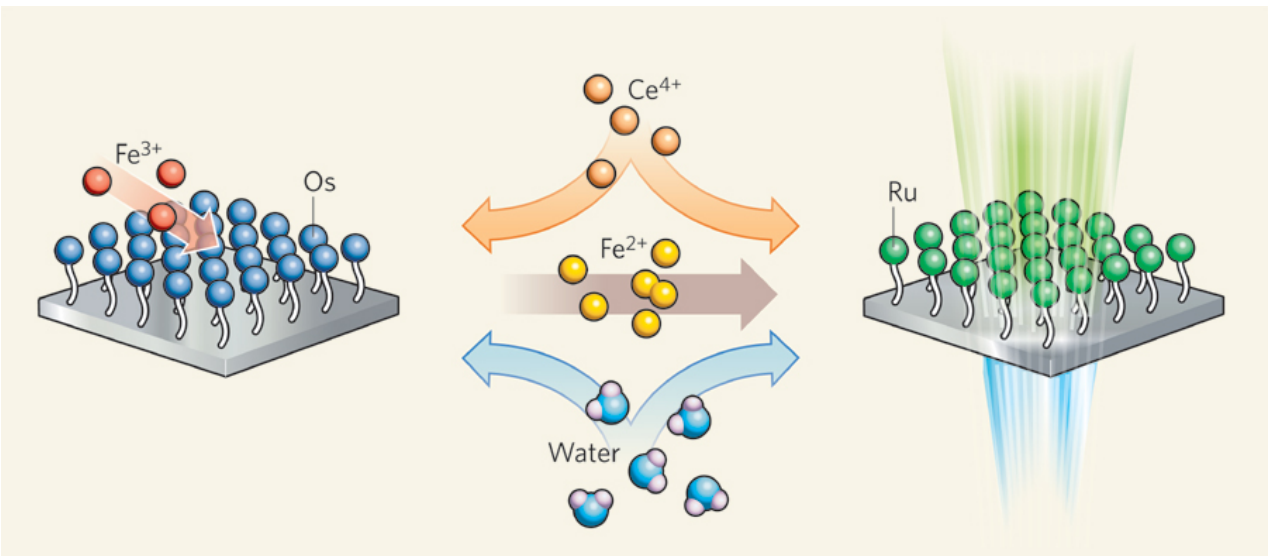
Watch video: http://www.youtube.com/watch?v=G2Ljgkh_v40

Molecular Computing: Chemical logic on a chip

- Computation on surfaces (mono- layers on glass surfaces)
- Concentrations of specific chemicals as inputs



- Ru is an AND gate for the presence of (Fe^{2+}or H_2O) and the absence of Ce^{4+}.
- Readout by difference in reflectance in the Ru-layer

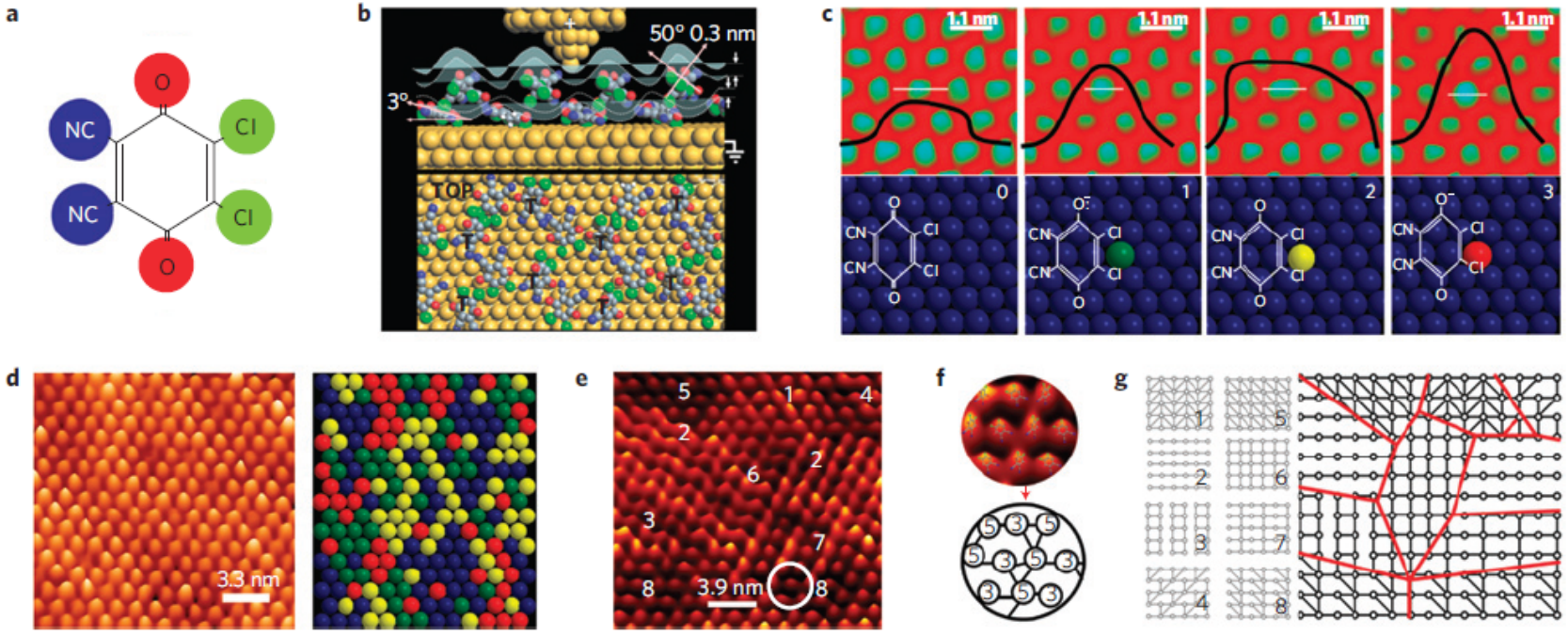


A. Prasanna de Silva: Molecular computing: A layer of logic. Nature 454, 417-418 (2008) reporting on Gupta, T. & van der Boom, M. E. Angew. Chem. Int. Edn 47, 5322–5325 (2008).

Computing on an organic molecular layer

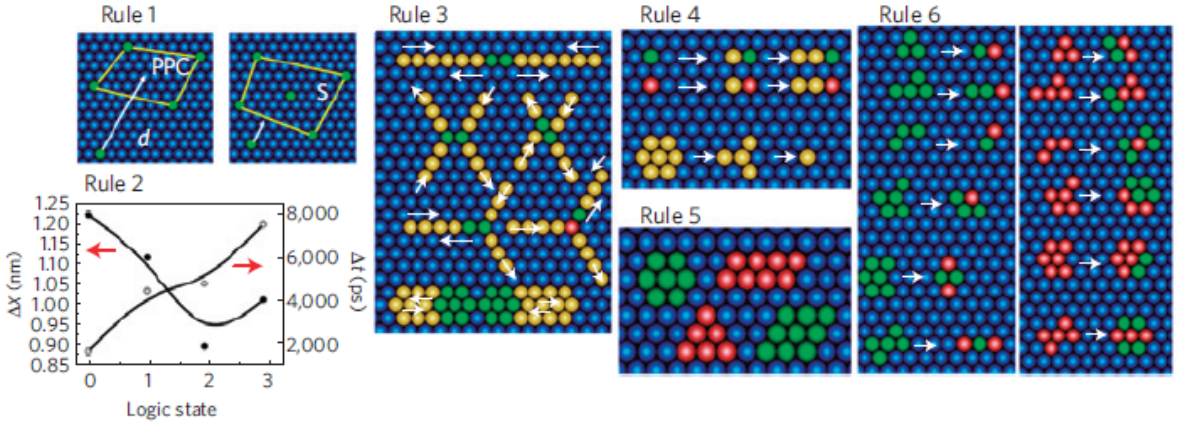
- Local interaction among the molecules of the layer represents a cellular automaton.
- Emerging patterns known since J. v. Neumann, now realizable in great variety in a molecular computer
- Massively parallel (300 dots at a time)
- Digital logic, calculating Voronoi diagrams, and simulating natural phenomena such as heat diffusion and cancer growth.

A. Bandyopadhyay e al.: Massively parallel computing on an organic molecular layer. NATURE PHYSICS, VOL 6, MAY 2010, 369-375.



- Switching among four conducting states by applying a voltage pulse via a the tip of a scanning tunnelling microscope (STM)
- Interaction depends on the state (8 different local circuits can be formed)
- Intralayer interaction is weak: Stable domains can be formed

A. Bandyopadhyay e al.: Massively parallel computing on an organic molecular layer. NATURE PHYSICS, VOL 6, MAY 2010, 369-375.

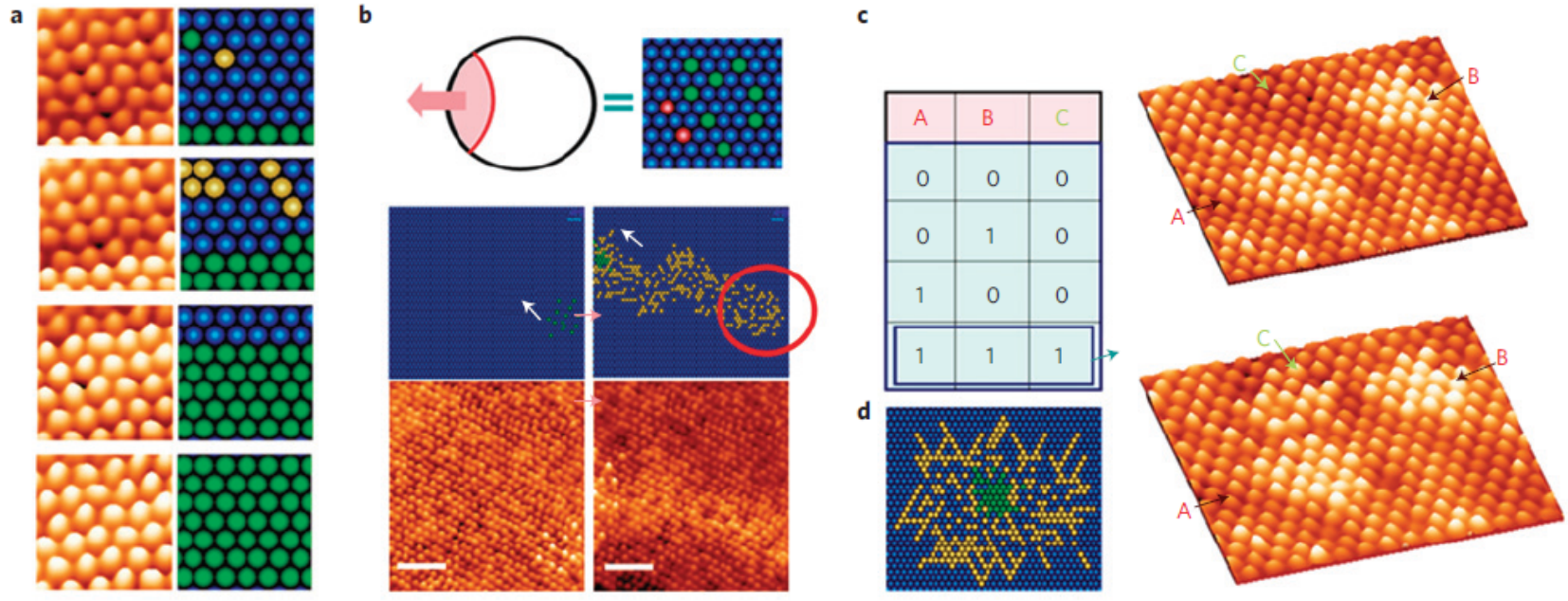


Rule 7

Input pattern (state 1s, and/or 3s)	Circuit created		Impact on rules	
	Between sub-patterns	Outside the pattern	Less dominating	More dominating
Linear	NA	Type 2	Rule 1	Rule 5
Parallel linear	Type 2	Type 2, 3	Rule 1	Rule 3, 5
Circular	NA	Type 2	Rule 3	Rule 1
Concentric circular	Type 6	Type 3	Rule 1, 3	Rule 4, 6
Solid triangle	NA	Type 7, 8	Rule 1, 6	Rule 3
Open triangle	Type 4	Type 8	Rule 4, 6	Rule 1, 3
Discrete states	NA	Type 7	None	Rule 1, 3

- Charges are attracted to "charged" zones on the layer. This enables a number of different state changes
- Transitions between states like in the Game of Life

A. Bandyopadhyay e al.: Massively parallel computing on an organic molecular layer. NATURE PHYSICS, VOL 6, MAY 2010, 369-375.

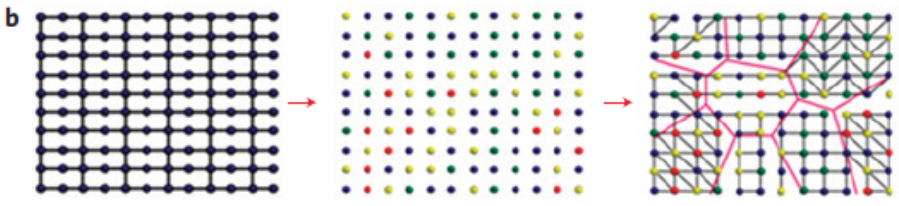


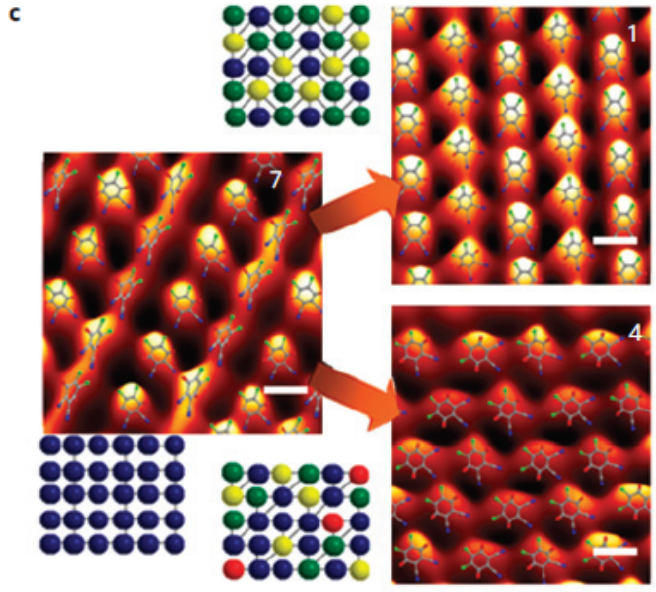
- Transport of "information packages"
- Realization of a AND gate

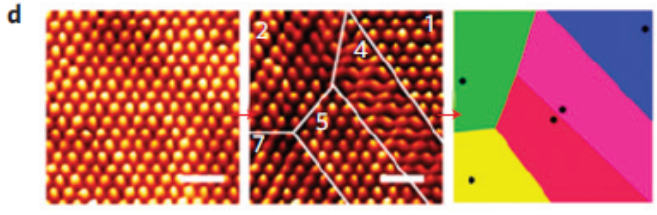
A. Bandyopadhyay e al.: Massively parallel computing on an organic molecular layer. NATURE PHYSICS, VOL 6, MAY 2010, 369-375.

a

0	1	2	3	Circuit type (wiring)	Dominating rules	No. of electrons/ area
25%	>50%	25%	0%	1 (6)	Rule 3	10/20 nm^2
30%	10%	>60%	10%	2 (2)	Rule 5	6/20 nm^2
40%	25%	30%	5%	3 (2)	Rule 5	6/20 nm^2
50%	25%	15%	10%	4 (3)	Rule 6	9/20 nm^2
35%	0%	35%	>30%	5 (6)	Rule 3	14/20 nm^2
50%	40%	10%	0%	6 (4)	Rule 6	8/20 nm^2
>60%	35%	5%	0%	7 (3)	Rule 3	7/20 nm^2
55%	20%	15%	10%	8 (3)	Rule 1	8/20 nm^2

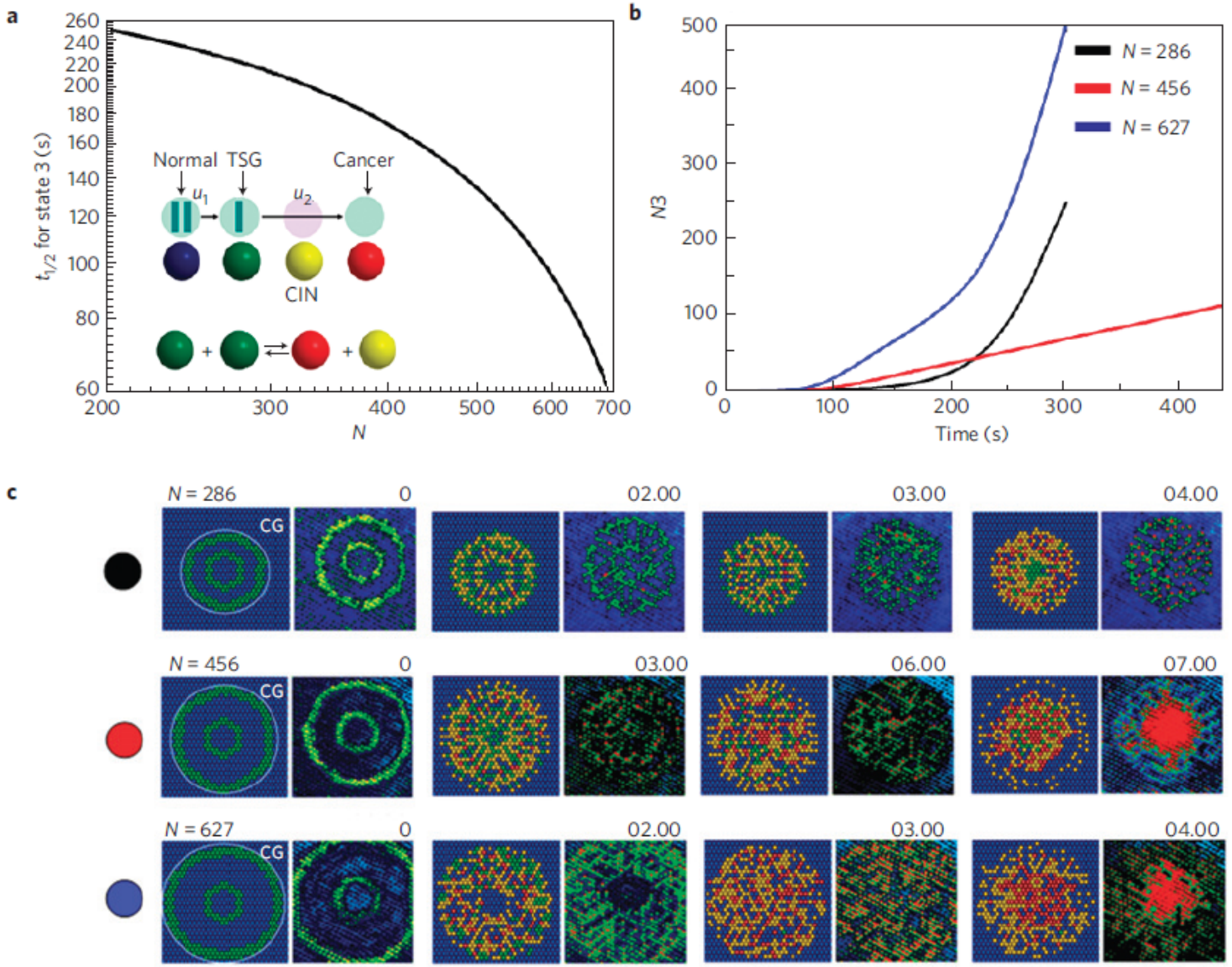






- Formation of a Voronoi diagram (represented by formation of different interaction patterns)

A. Bandyopadhyay e al.: Massively parallel computing on an organic molecular layer. NATURE PHYSICS, VOL 6, MAY 2010, 369-375.



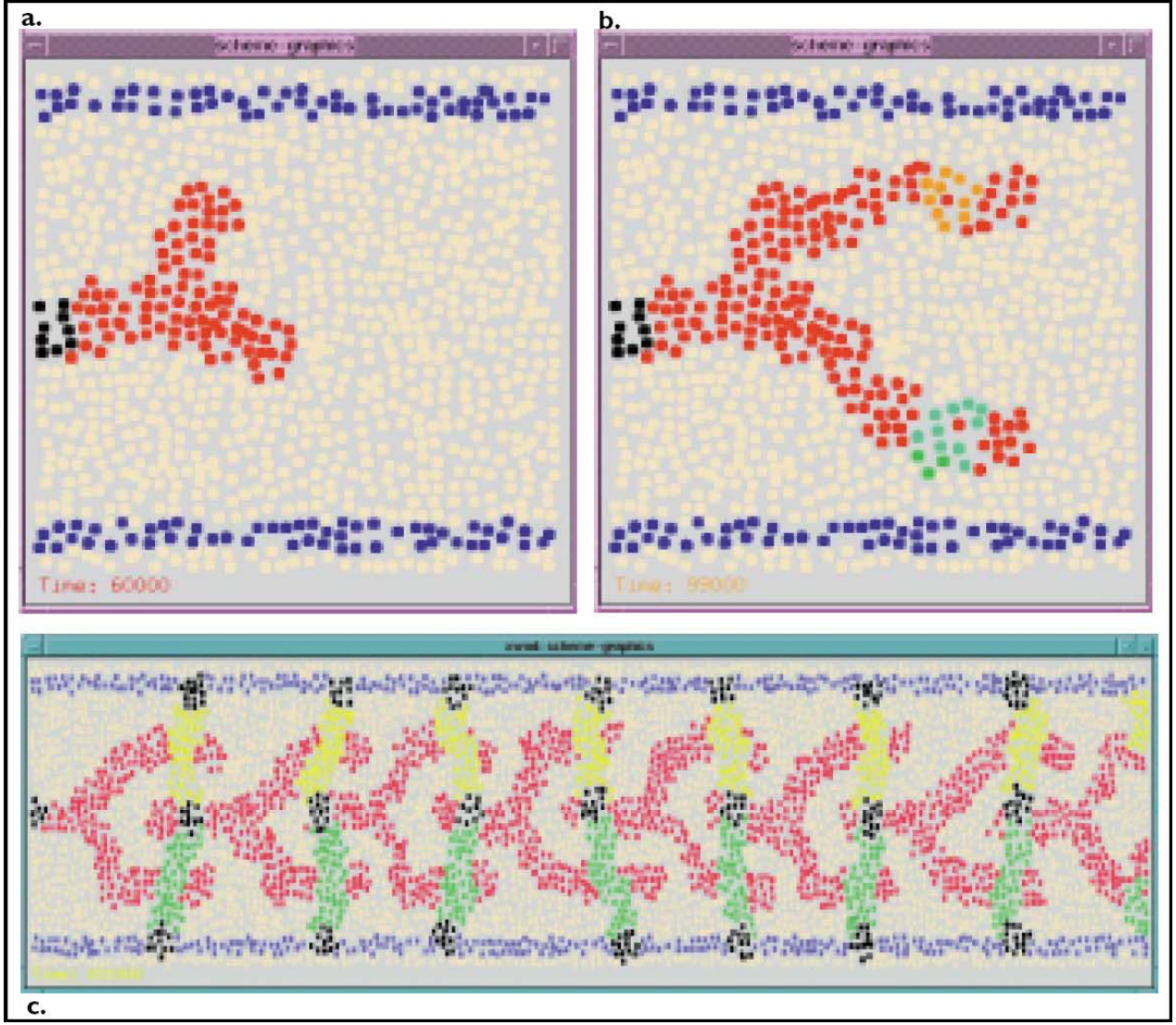
- Simulation of a biological process using the molecular computer (evolution of cancer cells)

A. Bandyopadhyay e al.: Massively parallel computing on an organic molecular layer. NATURE PHYSICS, VOL 6, MAY 2010, 369-375.

Amorphous Computing

- Implemented by redundant, potentially faulty, massively parallel devices (not necessarily molecules)
- Devices having limited memory and computational abilities.
- Devices being asynchronous.
- Devices having no a priori knowledge of their location.
- Devices communicating only locally.
- Exhibit emergent or self-organizational behavior (patterns or states larger than an individual device).
- Fault-tolerant, especially to the occasional malformed device or state perturbation.

Figure 2. Evolution of a complex design—the connection graph of a chain of CMOS inverters—being generated by a program in Coore's growing-point language. (a) An initial "poly" growth divides to form two branches growing toward "Vdd" and "ground." (b) The branches start moving horizontally and sprout pieces of "diffusion." (c) The completed chain of inverters.

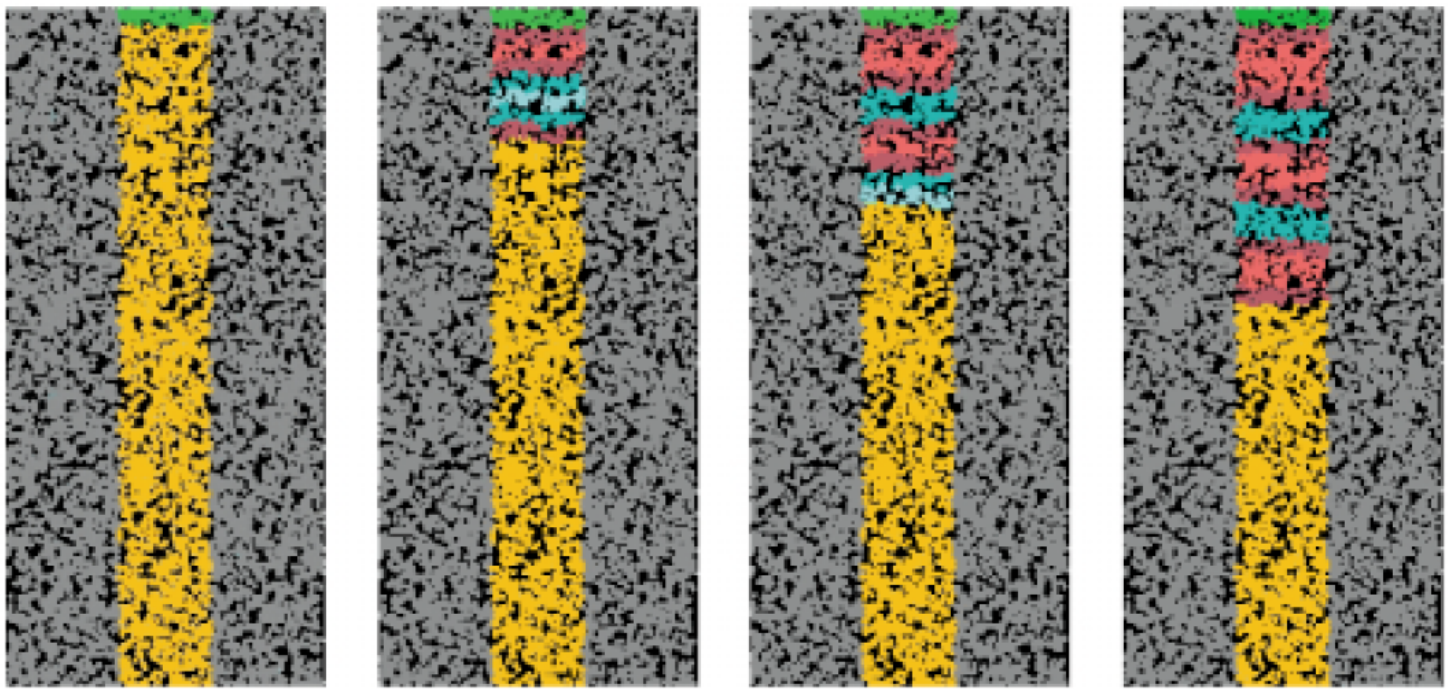


Amourphous computing with simple interacting processors as a metaphor of molecular computing.

www.eecs.harvard.edu/ssr/papers/cacm00-abelson.pdf

Amorphous Computing

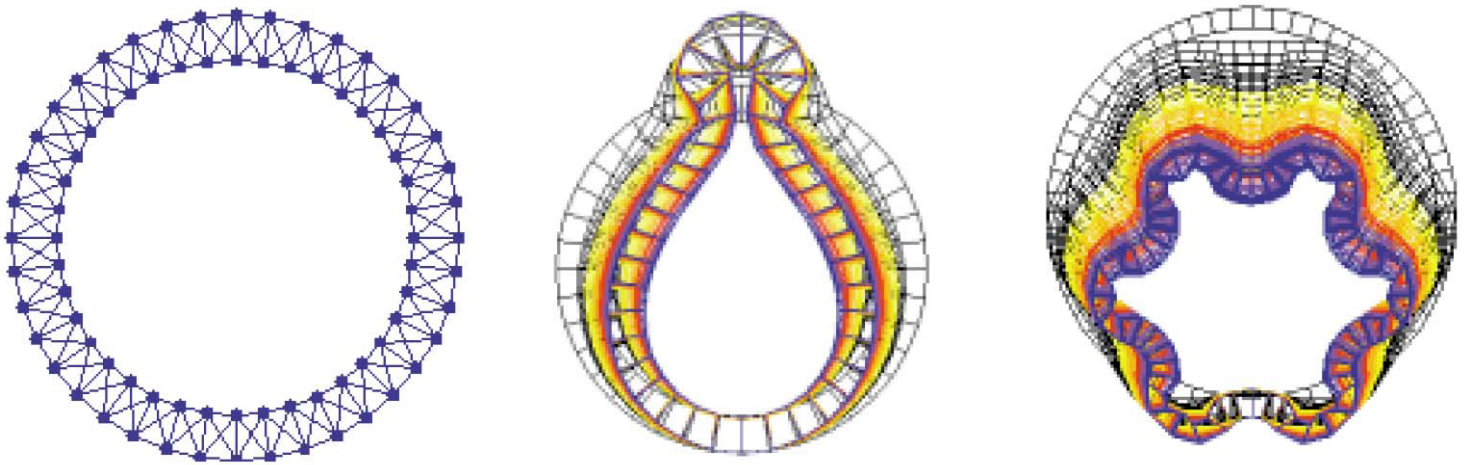
Figure 3. A pattern of alternating bands produced by marker propagation with the aid of Weiss's programming model.



www.eecs.harvard.edu/ssr/papers/cacm00-abelson.pdf

Amorphous Computing: Smart matter

Figure 4. Control of shape changes in a ring of cells, based on the mechanical cell models of [10]. Each cell has a simple programmed behavior and reacts to stresses in its neighbors.



Changing material properties by interacting microactuators.

www.eecs.harvard.edu/ssr/papers/cacm00-abelson.pdf

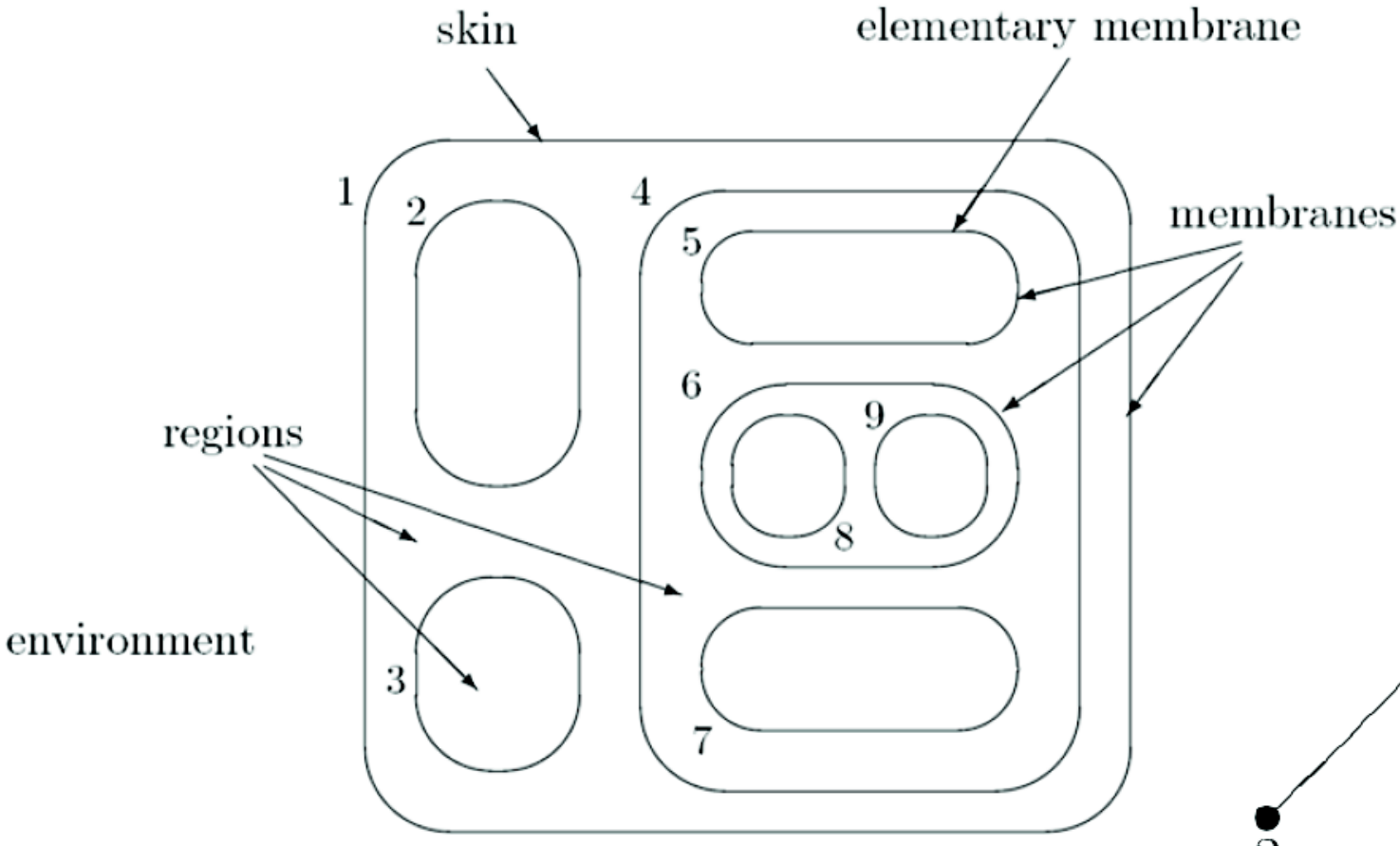
Perspectives

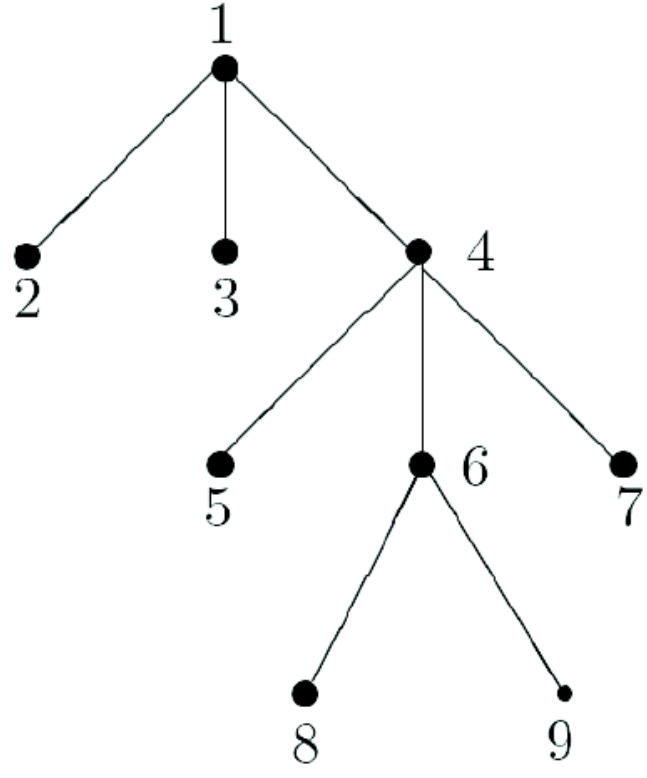
- Nanotechnology: Self-assembly, micromanipulation
- Synthetic biology
- Swarm intelligence in nano-robots
- Computing using designed molecules to carry out elementary computations
- Computation on surfaces, gels, networks
- Quantum computing: first universal 2-qubit quantum computer in 2009 (79% accuracy), 3 qubit (2010)
    - in future also in combination with molecular computing

Membrane Computing

- An area that seeks to discover new computational models from the study of the cellular membranes.
- It not so much the task of creating a cellular model but to derive a computational mechanism from processes that are know to proceed in a cell.
- Deals with distributed and parallel computing models, processing multi-sets of symbol objects
- The various types of membrane systems have been formalized as P systems.

Gheorghe Păun: Introduction to Membrane Computing





$$[_1\ [_2\ ]_2\ [_3\ ]_3\ [_4\ [_5\ ]_5\ [_6\ [_8\ ]_8\ [_9\ ]_9\ ]_6\ [_7\ ]_7\ ]_4\ ]_1$$

$$=$$

$$[_1\ [_3\ ]_3\ [_4\ [_6\ [_8\ ]_8\ [_9\ ]_9\ ]_6\ [_7\ ]_7\ [_5\ ]_5\ ]_4\ [_2\ ]_2\ ]_1$$

P systems

- A computational rather than a biological model
- Membrane Computing looks at the whole cell structure and functioning as a computing device
- Membranes play a fundamental role in the cell as filters and separators
- Modeling the living cell is beyond the purpose of Membrane Computing

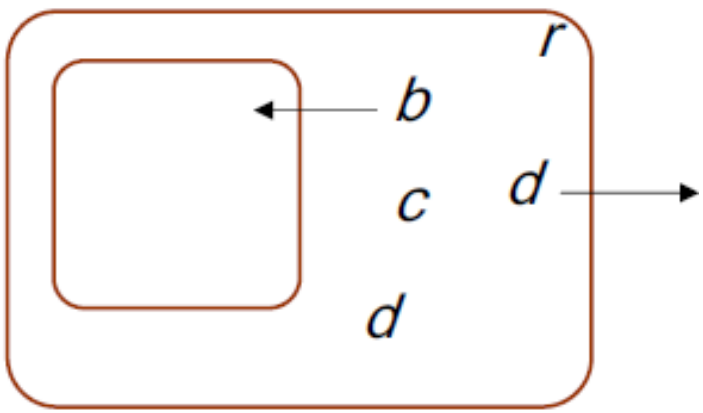
Gheorghe Păun: Computing with Membranes, (1998)

P systems

- A membrane structure formed by several membranes embedded in a unique main membrane (skin)
- Multi-sets of objects placed inside the regions delimited by the membranes (one per each region)
- The objects are represented as symbols of a given alphabet (each symbol denotes a different object)
- Sets of evolution rules associated with the regions (one per each region), which allow the system
  - to produce new objects starting from existing ones
  - to move objects from one region to another

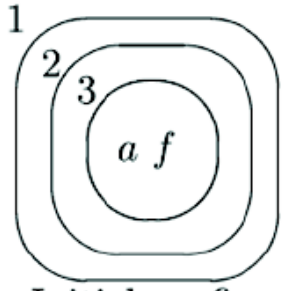
P systems

- Evolution rule of region
  $r : ca \rightarrow cb_{in}d_{out}d_{here}$
- "a copy of object a in the presence of a copy of the catalyst c is replaced by a copy of the object b and 2 copies of the object d" and
- "b enters the inner membrane of region r" and
- "one copy of d leaves region r" and
- "one copy of d remains in r."

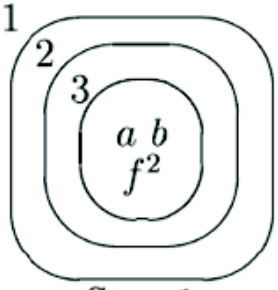


Computation in a P-system

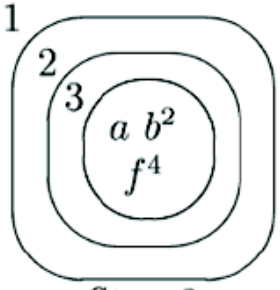
- Initial configuration: Construct a membrane structure and place an initial multi-set of objects inside the regions of the system.
- Apply the rules in a non-deterministic parallel manner: in each step, in each region, each object can be evolved according to some rule
- Halting if a configuration is reached where no rules can be applied
- The result is the multi-sets formed by the objects contained in a specific output membrane
- A non-halting computation yields no result
- Example: Assume the environment is reduced to some specific input objects. Now if the system halts in a final configuration, the system has "recognized"this input.



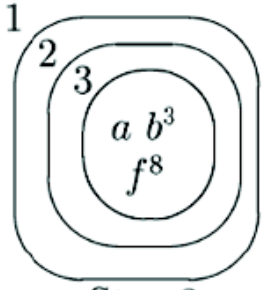
Initial config.



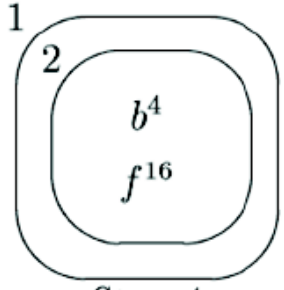
Step 1



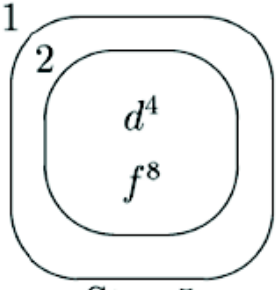
Step 2



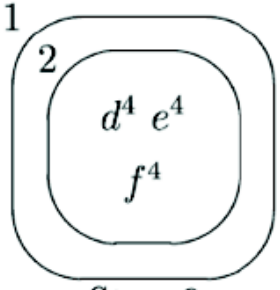
Step 3



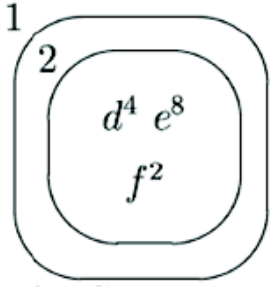
Step 4

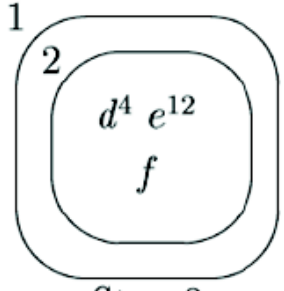


Step 5

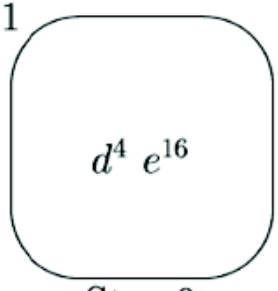


Step 6

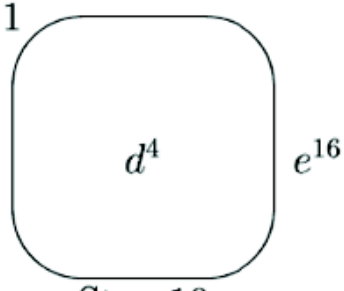




Step 8



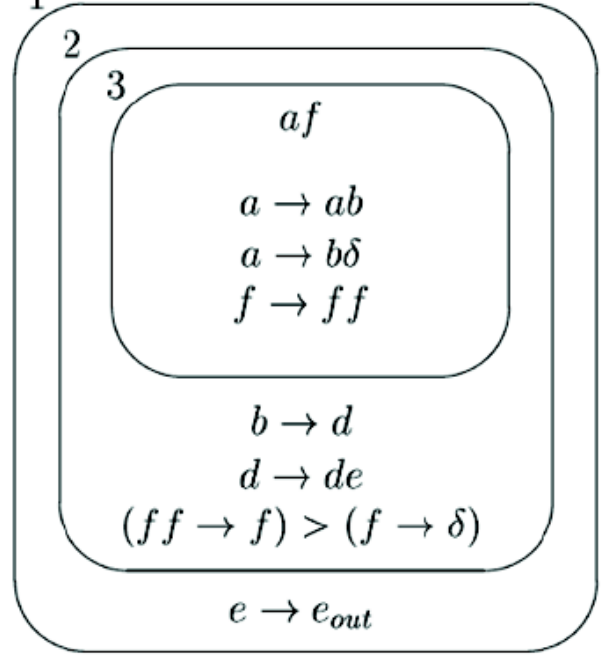
Step 9



Step 10

rules:

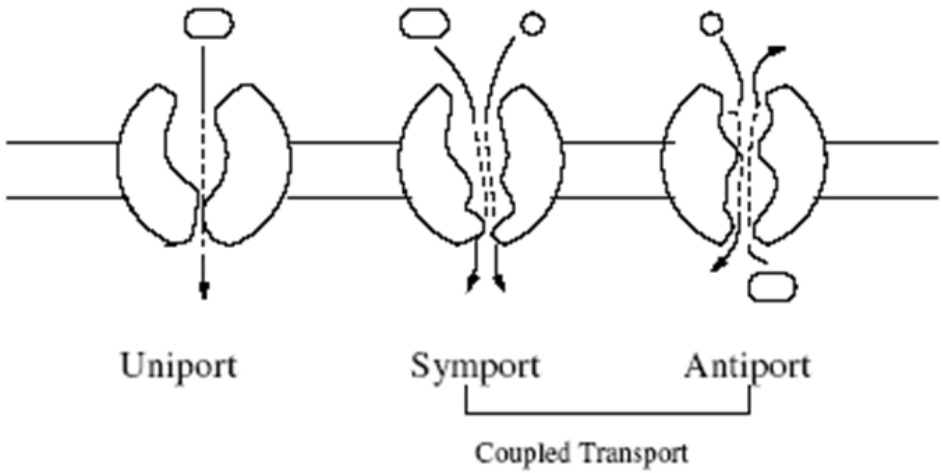
δ: dissolutor



More operators

- Membrane dissolution
- Priorities, reaction rates
- Catalysts
- Bi-stable catalysts
- Membrane permeability
- Active membranes: creation, deletion, duplication
- P-systems with catalysts are computationally universal

Communicative *P*-systems



P-systems with symport/antiport are computationally universal (even without chemical reactions)

Natural computing with P systems

- Complex dynamics of a bio-physical system
- Parallel by nature
- Largely non-deterministic
- Encoding/readout problems must be solved in applications to practical problems
- Computation works well in many problems but may be ineffective or inefficient on some problems
- In natural computing, good solutions emerge or are discovered rather than being designed, although capabilities of design are presently improving in a very impressive way.