

Course: Natural Computing

\*10. Biological Aspects of Natural Computing



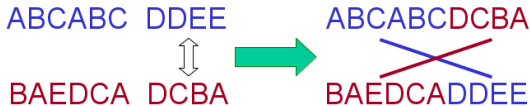
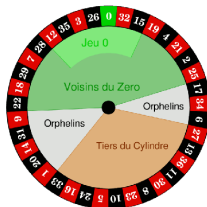
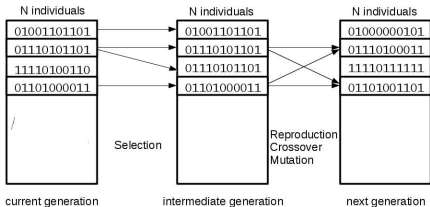
J. Michael Herrmann  
School of Informatics, University of Edinburgh

michael.herrmann@ed.ac.uk, +44 131 6 517177

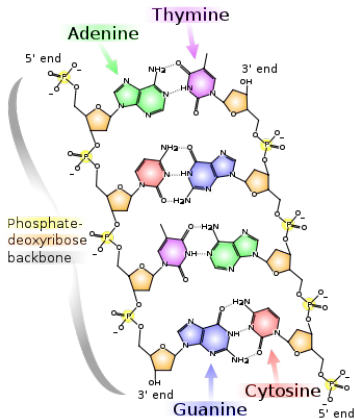
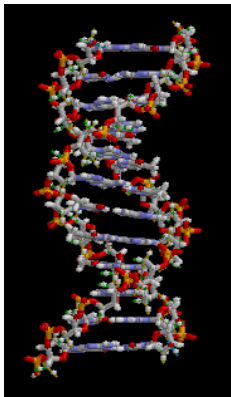
## Overview

- Algorithms inspired by nature
- Nature as seen with the eyes of a metaheuristicist
- Realisations of computational paradigms in nature

# Genetic Algorithms: Retrospective



# The Genetic Code



James Watson,  
Francis Crick,  
Maurice Wilkins  
and Rosalind  
Franklin:  
DNA structure  
(hypothesis 1953,  
Nobel Prize 1962)

## The Genetic Code

DNA = deoxyribonucleic acid

DNA is made up of a chain of simple molecular units. Each unit comprises a base, a sugar and a phosphate. The sugars and phosphates in many units link together in a chain with the bases sticking out. The bases in two chains attract one another resulting in a double helix structure.

There are just 4 kinds of **base** in DNA, labelled A, C, G and T (adenine, cytosine, guanine, thymine). C and G pair up, as do A and T.

... GATTACCA ...  
... CTAATGGT ...

George Gamow (1950s): Triplets as the elementary units of the genetic code (codons) [he wrongly assumed ambiguity: Is GGAC read as GGA+C or as G+GAC? Nature had solved this problem.]

## Encoding Proteins

How does this work?

Sections of chromosome contain the instructions for building chains of amino acids – proteins. The proteins are the building blocks, regulation units and manufacturing units of the body:

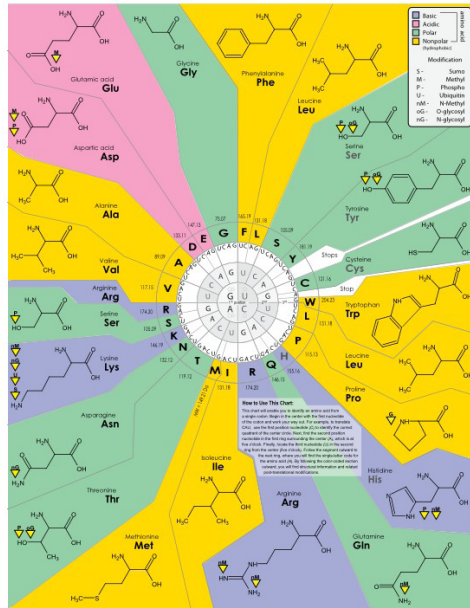
e.g. lactase (enzyme), collagen (structure), haemoglobin (oxygen transport), actin (muscle contractions), CLOCK protein (circadian rhythm regulation).

Encoding: 3 DNA bases → 1 amino acid      AAA = lysine  
64 combinations → 20 amino acids      CCC = proline  
– some redundancy

A protein is made up of many amino acids strung together and folded up.

# Coding Principle

$4^3 = 64$   
combinations  
from  
3 base pairs  
Encoding  
20 amino acids



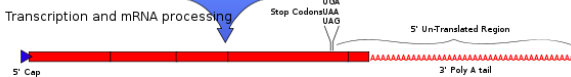
# Central Dogma of Molecular Biology

## Central Dogma of Molecular Biology : Eukaryotic Model

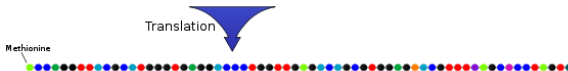
DNA



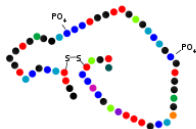
mRNA



Protein



Post-Translational Modification



Active Protein

Original work by Mike Jones for Wikipedia.



# Central Dogma of Molecular Biology

- Enunciated by Francis Crick in 1958 (Nature 1970)
- “Information cannot be transferred back from protein to either protein or nucleic acid.”
- In other words, “once information gets into protein, it can’t flow back to nucleic acid.”

From: To	DNA	RNA	Protein
DNA	replication	reverse transcription	?
RNA	transcription	RNA replication	?
Protein	direct translation	translation	prions?

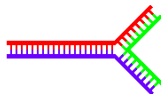
general transfer, special transfer, unknown

- Two chains of polymers, the nucleotides: Adenine (A), Cytosine (C), Guanine (G) and Thymine (T) or Uracil (U)
- Backbones made of sugars and phosphate groups joined by ester bonds
- **Exploit DNA as programmable matter to do computations as determined by material properties (i.e. by nucleotide sequence)**
- Overhangs can be used to attach a strand to a specific end of another string

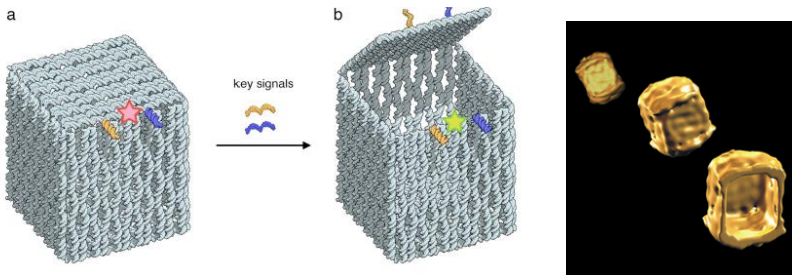
ATCTGACT      GATGCGTATGCT  
TAGACTGACTACG      CATACGA

↙

- Branches: if the strings do not match or if branching is triggered by a third molecule



# Self-Assembly of a Nano-Size DNA Box



Purposes for the box:

- calculator or logic gate
- controlled release, for example of drugs, in response to external stimuli
- sensor - where the thing you are sensing causes the box to open or close and give a readout

E S Andersen ... and J Kjems, Nature 2009 DOI:10.1038/nature07971

# DNA: Computing by Molecules

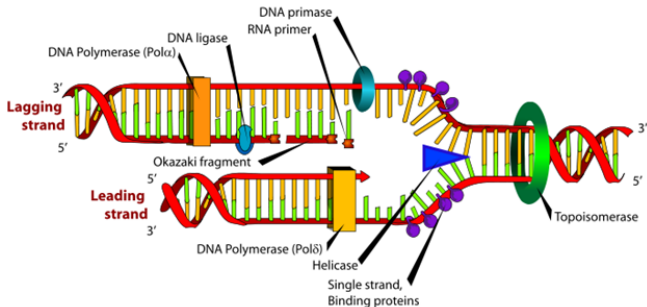
## DNA computing in numbers

- Energy consumption  $2 * 10^{19}$  operations/Joule (a billion times better than classical computers)
- 5 grams of DNA contain  $10^{21}$  bases (Zetta Bytes) [the size of the Internet is still measured in Exa Bytes ( $10^{18}$ )]
- Each DNA strand represents a processor (300 trillion)
- Relatively slow speed: 500-5000 base pairs a second
- Still the fastest and most economical among the existing computing mechanisms

- 7-point Hamiltonian path problem (L. Adleman, 1994)
- Programmable molecular computing machine (E. Shapiro, 2002)
- DNA computer (E. Shapiro, 2004) capable of diagnosing cancerous activity within a cell and releasing an anti-cancer drug upon diagnosis.
- “DNA origami” use a raster to impose precision and use “genetic operators” (here in a different sense: cut, insert, splice etc.) to fix the structure (see S. Dey et al. (2021) DNA origami. *Nature Reviews Methods Primers*, 1:1, 1-24)
- In theory, DNA computers can emulate Turing machines

# DNA Computing

- Parallelism by (typically) trillions of “processors”
- Complementarity makes DNA unique  $\Rightarrow$  error correction (improving upon 1 transcription error per every 100,000 bases)
- Basic suite of operations: AND, OR, NOT in a CPU must be represented by cutting, linking, pasting, amplifying or repairing DNA



Basic operations can be carried out on DNA sequences by commercial available enzymes:

- Cutting. An enzyme restriction endonuclease permits to recognize a small portion of the DNA. Any double helix that contains it can be cut in that exact place.
- Linking. An enzyme DNA ligase allows to join the end of a DNA sequence with the beginning of another one.
- Replication. An enzyme DNA polymerase makes possible the DNA replication.
- Destruction. With the enzyme exonuclease, it is possible to eliminate certain DNA subsequences.

C.A. Alonso Sanches, N.Y. Soma / Applied Mathematics and Computation 215 (2009) 2055–2062

# Seminal Example: Adleman's solution of the Hamiltonian Directed Path Problem (HDPP)

Algorithm to be implemented in the DNA computer:

- 1 Generate random paths
- 2 From all paths created in step 1, keep only those that start at  $s$  and end at  $t$ .
- 3 From all remaining paths, keep only those that visit exactly  $n$  vertices.
- 4 From all remaining paths, keep only those that visit each vertex at least once.
- 5 If any path remains, return "yes" otherwise, return "no"

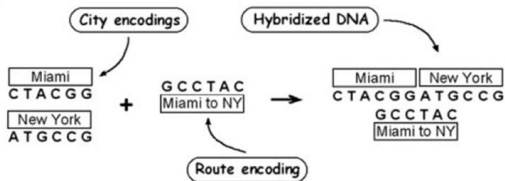


# Example with 5 cities: 1. Encoding and path construction

Encoding of nodes and edges using artificial gene synthesis

city	code
Los Angeles	GCTACG
Chicago	CTAGTA
Dallas	TCGTAC
Miami	CTACGG
New York	ATGCCG

route	(anti-)code
Los Angeles-Chicago	TGCGAT
Chicago-Los Angeles	CATAGC
Dallas-Miami	ATGGAT
Miami-New York	GCCTAC
...	... ..



L.A -> Chicago -> Dallas -> Miami -> New York would simply be  
GCTACGCTAGTATCGTACCTACGGATGCCG

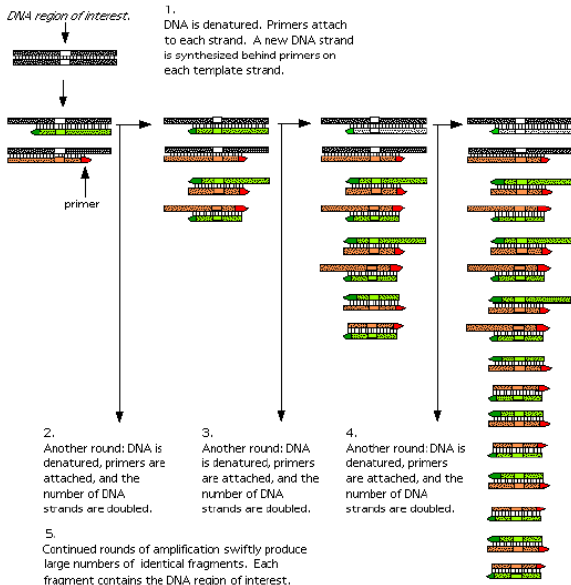
# Polymerase Chain Reaction (PCR)

PCR: One way to amplify DNA. (Kary Mullis, 1985)

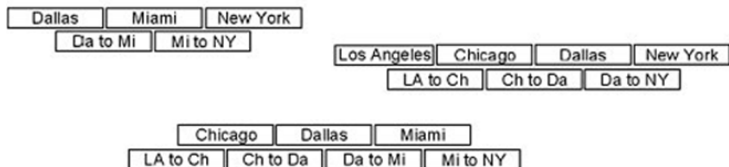
PCR alternates between two phases:

- separate DNA into single strands using heat
- convert into double strands using primer and polymerase reaction

PCR rapidly amplifies a single DNA molecule into billions of molecules



## Step 2: PCR selecting paths with correct start and end

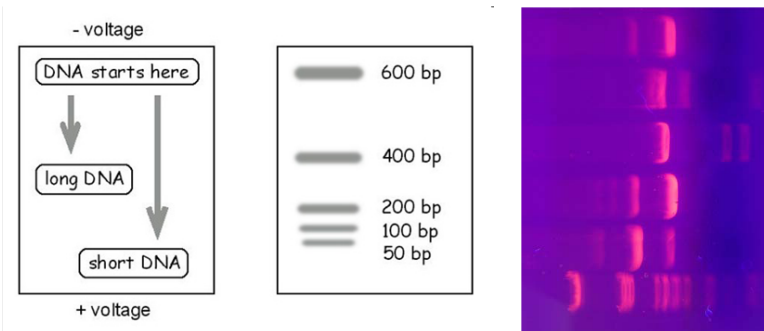


(a billion copies of each)

- Starting a PCR with the starting city as a primer (i.e. the primer is the complement of the city code)
- and a primer for termination corresponding to the goal city.
- produces lots of strands with the correct start and end (but with variable lengths and with possible repetitions)

## Step 3: Gel electrophoresis for measuring lengths

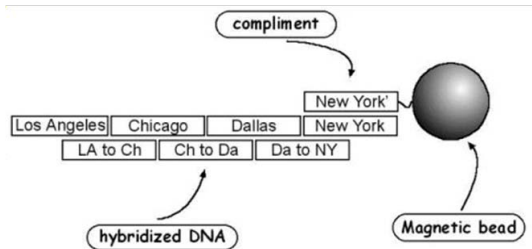
- Used to measure the length of a DNA molecule.
- Smaller DNA molecules travel faster in an electric field (for same charge)



- Keep only those paths that visit exactly  $n$  vertexes.
- Isolate the DNA if 30 base pairs long (5 cities  $\times$  6 base pairs).

## Step 4: Affinity purification for selecting admissible paths

- Select itineraries that have a complete set of cities
- Sequentially affinity-purify fives times, using a different city complement for each run.



- We are left with itineraries that start in LA, visit each city once, and end in NY.

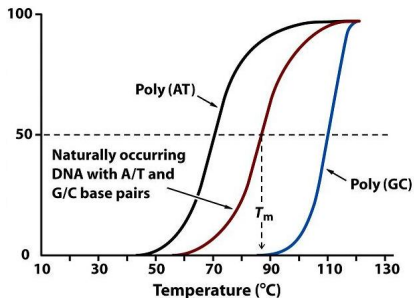
<http://arstechnica.com/reviews/2q00/dna>

## Step 5: Reading out the answer

- Result can be obtained by sequencing the DNA strands
- More effectively by using **graduated PCR**:
  - A series of PCR amplifications using the different primer for each city in succession.
  - Measuring the various lengths of DNA for each PCR product reveals the final sequence of cities.
  - For example, starting with LA gives 30 base pairs. If LA and Dallas primers give 24 base pairs, Dallas is the fourth city in the itinerary.

# Extension to TSP

Use differences in melting temperature (depends on the fraction of AT vs. GC) to encode real numbers. Note that the representation is not neutral to the selection anymore.



**Step 1:** Generation of answer pool  
[Hybridization & Ligation]

**Step 2:** Selection of paths satisfying the conditions of TSP [PCR with primer  $v_{out}$  and  $v_{in}$ , and affinity-separation]

**Step 3:** More amplification of the more economical paths [DTG-PCR]

**Step 4:** Separation of the most economical path among the candidate paths [TGGE]

**Step 5:** Readout of the final path  
[Cloning and sequencing]

Ji Youn Lee (2004) Solving traveling salesman problems with DNA molecules encoding numerical values. *BioSystems* 78 (2004) 39–47

<http://sandwalk.blogspot.com/2007/12/dna-denaturation-and-renaturation-and.html>

Natural Computing 24/25, week \*10, Michael Herrmann, School of Informatics, University of Edinburgh

- Adleman solved a seven city problem. What about more cities?
- The complexity of the problem still increases exponentially.
- For Adleman's method the amount of DNA scales exponentially: to solve a 200 city HP problem would take an amount of DNA that weighed more than the earth
- Each step contains statistical errors which grow if the strands become longer, more operations are being performed and less DNA is used per potential solution
- Cost: For a computation with  $n = 100$ , one would need to purchase at least 300 strands (cost are  $< \$100$ ).

Shrenik Shahy: DNA Computation and Algorithm Design. Harvard University 2009 Cambridge, MA 02138 (student article)



# Solved/solvable problems

- Ravinderjit S. Braich, Nickolas Chelyapov, Cliff Johnson, Paul W. K. Rothemund, Leonard Adleman: Solution of a 20-Variable 3-SAT Problem on a DNA Computer. *Science* 296, 19 April 2002
- S. Paul. G. Sahoo: Procedure for Multiplication based on DNA Computing 2009 Int. Conf. on Adv. in Computing, Control, and Telecommunication Technologies
- C. A. A. Sanches , N. Y. Soma: A polynomial-time DNA computing solution for the Bin-Packing Problem. *Applied Mathematics and Computation* 215 (2009) 2055–2062
- t.b.c.

# Adleman–Lipton model

From chemistry to language

A “test tube” language describing DNA computation on a multi-set of finite strings over the alphabet  $\{A; C; G; T\}$ .

- Amplify. Given a test tube  $T$ ;  $amplify(T; T_1; T_2)$  produces two new test tubes  $T_1$  and  $T_2$  that are identical copies of  $T$ , and this latter one becomes empty.
- Merge. Given two test tubes  $T_1$  and  $T_2$ , this operation generates a new tube with the content of both, that is, it is equivalent to decant the contents of the  $T_1$  and  $T_2$  into a third one without modifying no molecule.
- Append. Given a test tube  $T$  and a sequence  $S$ ;  $append(T; S)$  affixes  $S$  at the end of each sequence in  $T$ .
- Extract. Given a test tube  $T$  and a sequence  $S$ , generates two tubes:  $+(T; S)$  with all the sequences in  $T$  that had  $S$  as a subsequence, and  $-(T; S)$  with the remaining sequences of  $T$ .
- Detect. Given a tube  $T$ , this operation returns the logic value *yes* if there is at least a DNA molecule in it, and *no* otherwise. Discard.  
Given a test tube  $T$ , this operation simply discards it.

# Seminal Example: Adleman's solution of the Hamiltonian Directed Path Problem (HDPP)

Algorithm to be implemented in the DNA computer:

- 1 Generate random paths (**Append**)
- 2 From all paths created in step 1, keep only those that start at  $s$  and end at  $t$ . (**Extract**)
- 3 From all remaining paths, keep only those that visit exactly  $n$  vertexes. (**Extract**)
- 4 From all remaining paths, keep only those that visit each vertex at least once. (**Extract**)
- 5 If any path remains, return “yes” otherwise, return “no” (**Detect**)

# Sticker model enhances the Adleman–Lipton model

An enhanced language on a multi-set of finite strings over  $\{A; C; G; T\}$ .

- Combine: Given two test tubes  $T_1$  and  $T_2$  filled with DNA's sequences, this operation gives a third test tube  $T_3$  with a union of the first two. The operation corresponds to a merge to the original model.  
Notation :  $T_3 \leftarrow T_1 \cup T_2$
- Separate: Given a test tube  $T$  and a given bit in position  $i$ , this operation separates DNA's sequences into two groups: in test tube  $T_1$ , those with a value 1 in that position, and in test tube  $T_0$  the remaining ones. Moreover, the entire content of  $T$  is discarded.  
Notation :  $(T_0; T_1) \leftarrow (T; i)$
- Set: Given a test tube  $T$  and a particular bit at position  $i$ , all the DNA's sequences receive a sticker that corresponds to that bit.  
Notation :  $set(T; i)$
- Clear: Given a test tube  $T$  and a particular bit at position  $i$ , all the DNA's sequences with eventual stickers are removed in such a way that there is no match at that position.  
Notation :  $clear(T; i)$

# DNA algorithms

- The language is amended by initialisers and iterators
- Combinations of the models (e.g. Stickers plus amplify, append and detect) can be shown to solve NP-hard problems in polynomial time (assuming an exponential amount of DNA)
- A more formal approach is taken in  $H$ -systems (splicing systems) which have (for a finite set of slicing rules) the computing power as finite automata.
- Universal computation can be achieved by extended  $H$ -systems with permitting contexts which compute at the level of Turing machines
- Algorithms are obviously quite complex and unrealistic from a practical point of view.
- Note: algorithms run on multi-sets (i.e. set which may have more than one element of a kind)

[http://www.scholarpedia.org/article/Splicing\\_systems#Formal\\_Models\\_of\\_DNA\\_recombination](http://www.scholarpedia.org/article/Splicing_systems#Formal_Models_of_DNA_recombination)

# Conclusion on DNA computing

- Rather unsophisticated w.r.t. the computational problem, possibly to become more efficient (other molecules?)
- Can be performed automatically
- Most genetic operations are relatively cheap today, they become faster and smaller, improving in price/performance exponentially
- DNA chips (microarrays) are in operation (for diagnosis and research)
- Micro-electromechanical systems (MEMS) are there to operate them

L.M. Adleman: Molecular computation of solutions to combinatorial problems. *Science* 226 (1994), 1021-1024. G. Paun: *Computing with Bio-Molecules*, Springer-Verlag, 1998.

# Developments beyond Adleman's DNA algorithm

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

Self-assembly: On the molecular, supermolecular, 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 nanomachines?

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

# Developments beyond Adleman's DNA algorithm

- Zhang, C., Zhao, Y., Xu, X., Xu, R., Li, H., Teng, X., Du, Y., Miao, Y., Lin, H.C. and Han, D. (2020) Cancer diagnosis with DNA molecular computation. *Nature nanotechnology*, **15**:8, 709-715.



# Genetic Algorithms in DNA Computing

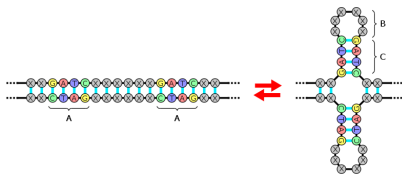
- Adleman's solution to the Hamiltonian path problem: checking all possible solutions (although "brute force", it was nevertheless a breakthrough in natural computing)
- New algorithms for a DNA computer?
- Life hasn't tested all possible combinations of genes, neither do GAs
- Introduce and exploit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

⇒ DNA genetic algorithm

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

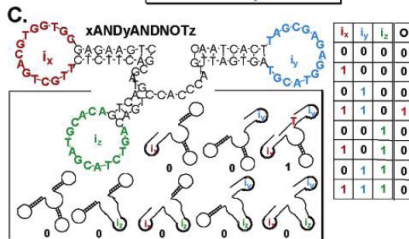
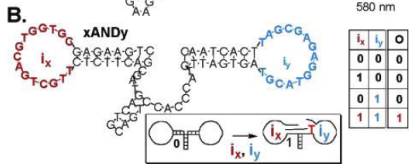
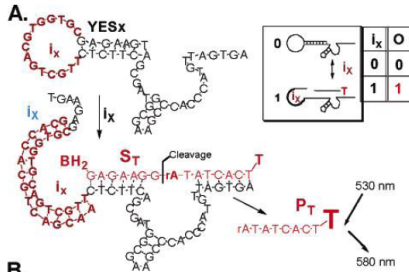
# GA operators

- 100 different restriction enzymes are known, each of which cuts at its specific recognition site(s) often near “palindromes” (reverse complements) → ‘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. J. Biological Physics 28: 493–498, 2002.  
[http://en.wikipedia.org/wiki/Palindromic\\_sequence](http://en.wikipedia.org/wiki/Palindromic_sequence)



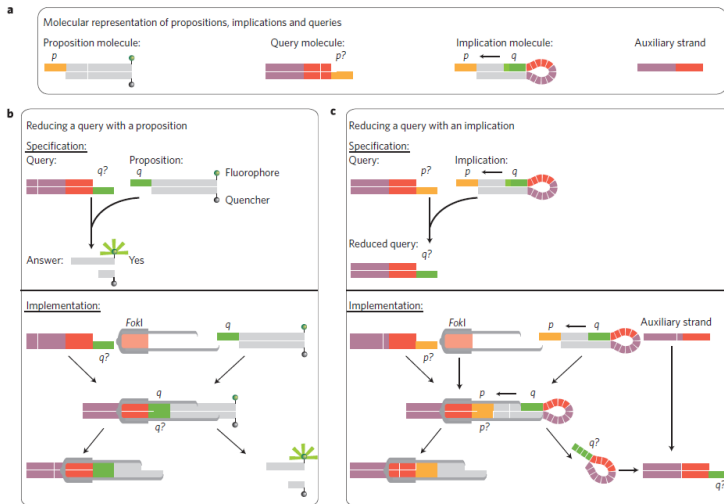
Programming using DNA molecules, e.g.

- 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)?

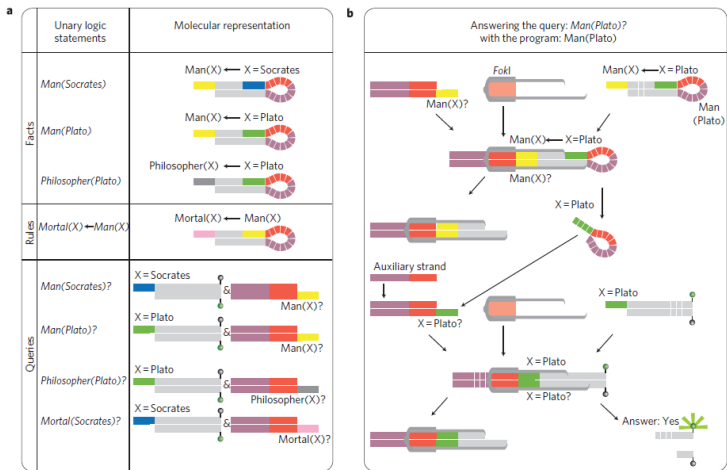
(Who is Mortal?)

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

# Molecular implementation of simple logic programs

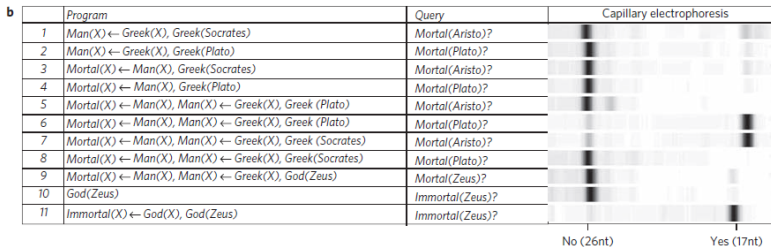


T. Ran, S. Kaplan & E. Shapiro: Molecular implementation of simple logic programs. *Nature Nanotechnology* 4, 642 - 648, 2009.



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

# Molecular implementation of simple logic programs



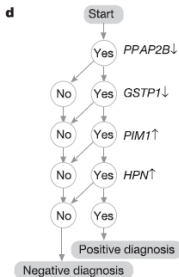
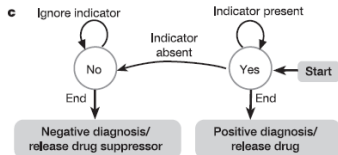
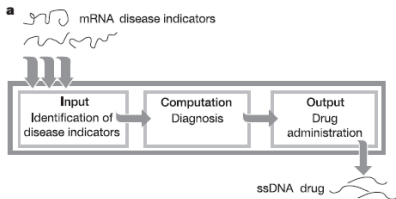
- All answers correct for the example of the previous page.
- The chart here shows that if some implication rules or facts are missing, answer is “no” (lanes 1,2,3,4,10)
- Reaction time (electrophoresis) is above one minute

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





# 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.

Hu, M et al.: An all-in-one homogeneous DNA walking nanomachine and its application for intracellular analysis of miRNA. *Theranostics* **9**:20 (2019) 5914-5923.

Zhang, C. et al.: Cancer diagnosis with DNA molecular computation. *Nature Nanotechnology* **15**:8 (2020) 709-715.

Chatterjee, G. et al.: A spatially localized architecture for fast and modular DNA computing. *Nature nanotechnology* **12**:9 (2017) 920-927.

Joesaar, A.: DNA-based communication in populations of synthetic protocells. *Nature Nanotechnology* **14** (2019) 369–378.

Estevez-Torres, A. & Rondelez, Y.: Spatially localized DNA domino. *Nature Nanotechnology* **12** (2017) 842–843.

# Synthetic Biology

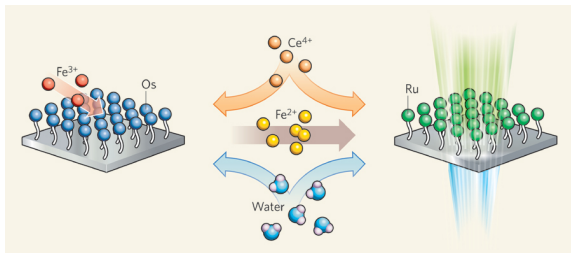
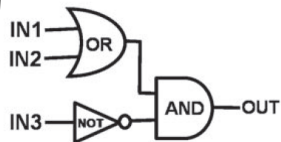
- Construction of molecular circuitry that can control biological systems or even diagnose and treat living cells from within.
- System of over 100 distinct DNA strands of 15 to 30 nucleotides whose binding and replication can be controlled to perform AND, OR, NOT, NAND, and NOR logic gates logic operations.
- The system allowed mathematical computation of a square root within 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 behaviour 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](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 ( $\text{Fe}^{2+}$  or  $\text{H}_2\text{O}$ ) and the absence of  $\text{Ce}^{4+}$ .
- Readout by difference in reflectance in the Ru-layer



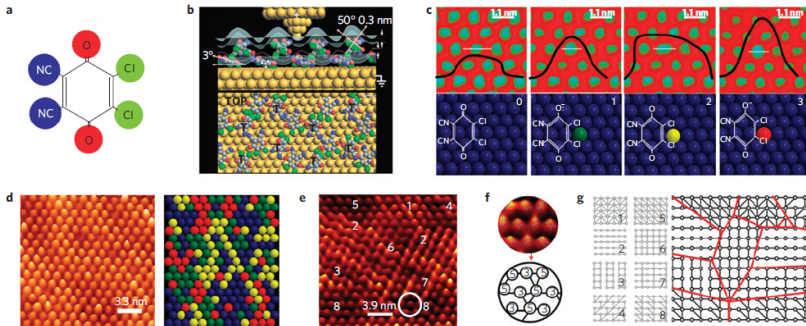
A. Prasanna de Silva: Molecular computing: A layer of logic. *Nature* 454, 417-418 (2008) reporting on Gupta, T. & van der Boom, M. E. *Angew. Chem. Int. Edn* 47, 5322-5325 (2008).

Natural Computing 24/25, week \*10, Michael Herrmann, School of Informatics, University of Edinburgh

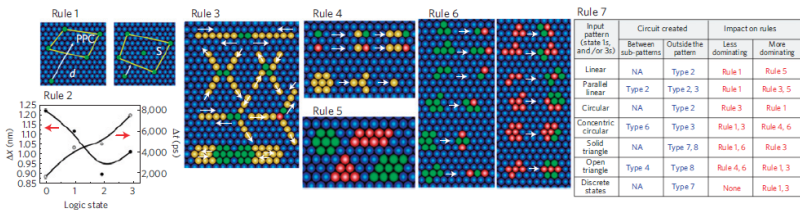
# Computing on an organic molecular layer

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

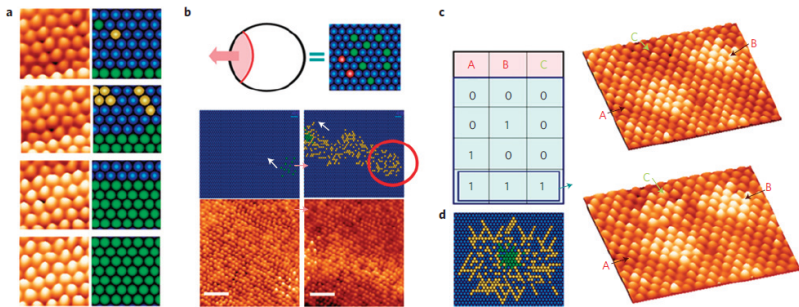


- Switching among four conducting states by applying a voltage pulse via the tip of a scanning tunnelling microscope (STM)
- Interaction depends on the state (8 different local circuits can be formed)
- Intralayer interaction is weak: Stable domains can be formed



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







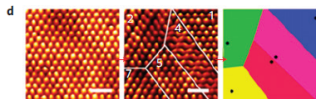
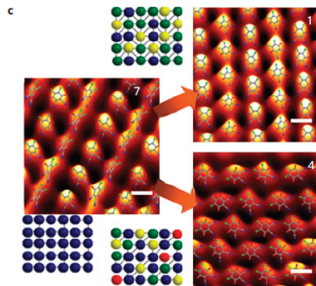
- Transport of “information packages”
- Realisation of a AND gate

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

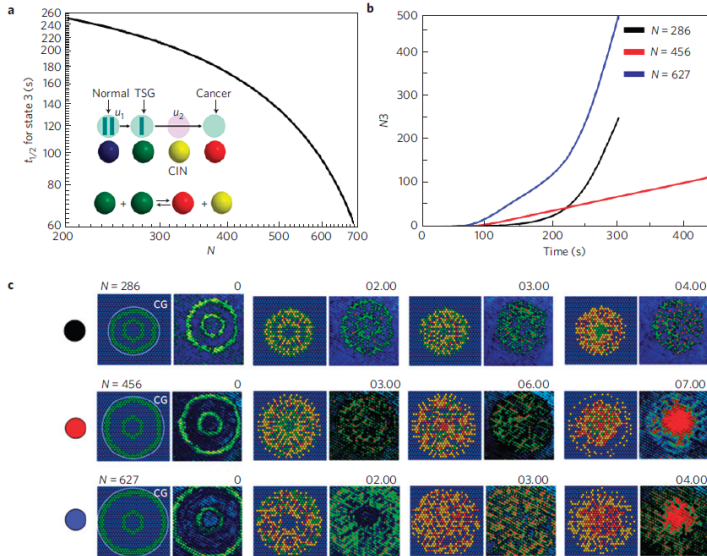


**a**

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



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



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

- $P$  systems,  $L$  systems,  $H$  systems: Representation vs. implementation?
- Design problem: A domain for MHO?
- Robustness?
- Autonomy?
- Risk-control?

# Natural Computing: A Formal Perspective

- Studying of models of computation inspired by biological systems
- Some approaches in Natural Computing use the methods of formal language theory
  - L systems: Development of multicellular organisms (plants), (Aristid Lindenmayer, 1968)
  - Cellular Automata (Stephen Wolfram, 1983; based on work by v. Neumann, Hedlund, Conway et al.)
  - H systems: DNA (Tom Head, 1987)
  - P systems: Membranes (Gheorghe Păun, 1998)

- 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
- Quantum computing in combination with molecular computing
- Membrane computing