PLM: A Model for Probabilistic Learning and Inference Based on DNA Computing*

Spring Conference of Korea Fuzzy and Intelligent Systems Society
April 30, 2005

Byoung-Tak Zhang

Biointelligence Laboratory
School of Computer Science and Engineering
Seoul National University
Seoul 151-742, Korea

btzhang@cse.snu.ac.kr

* Based on the material for GECCO-2005 Tutorial June 26, 2005, Washington, D.C.

Natural Computation

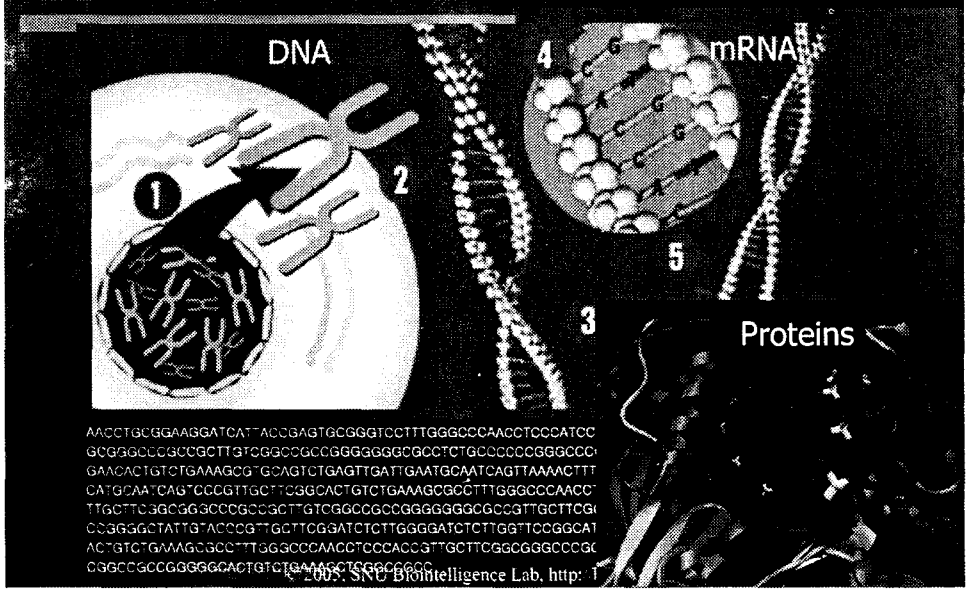
- Neural Computation
  - A network of neurons
- Evolutionary Computation
  - A population of chromosomes
- Molecular Computation
  - A test tube of molecules
- Molecular Evolutionary Computation ← This talk
  - A test tube of "evolving" molecules
  - "In vitro molecular evolution"

Talk Outline

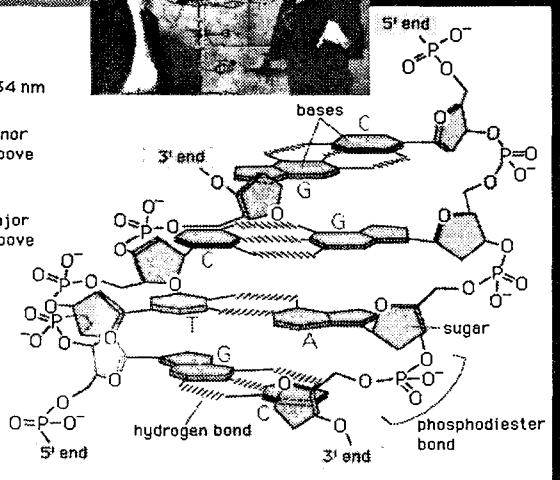
- Molecular Computation
- Learning and Inference with DNA Molecules: The PLM Model
- Molecular Programming (MP): The PLM in Practice
- Books and Web Sites

Molecular Computation

Biomolecular Information Processing



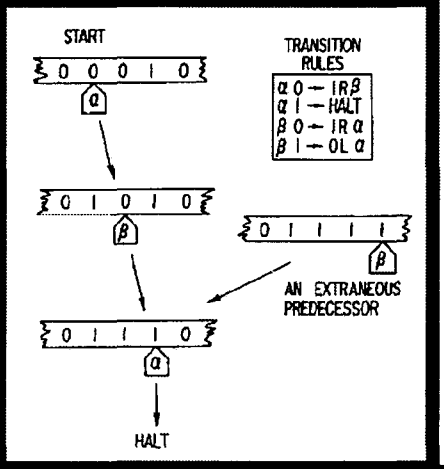
DNA

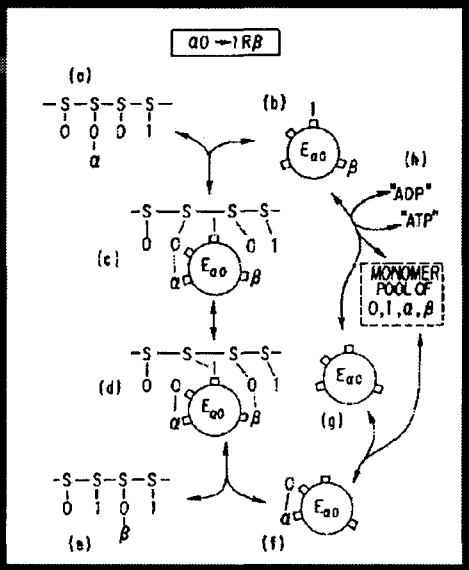


Molecular Computing: Brief History

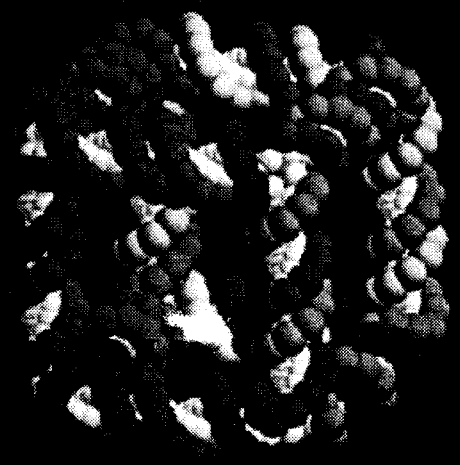
- Feynman (1959)
  - Potential of molecules
- Benett (1982)
  - DNA and thermodynamic computation
- Seeman (1991)
  - Self-assembly of a DNA cube
- Conrad (1992)
  - Lock-and-key paradigm for molecular computing
- Adleman (1994)
  - Experimental demonstration of DNA computing

Benett (1982)

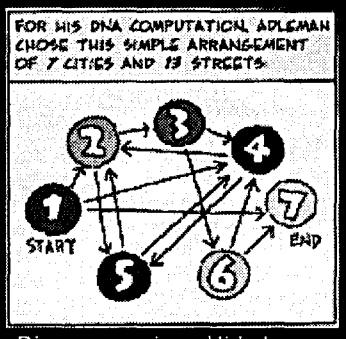




Seeman (1991)



Adleman (1994)



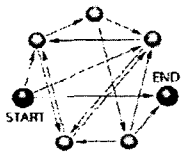
Discover magazine published an article in comic strip format about Leonard Adleman's discovery of DNA computation. Not only entertaining, but also the most understandable explanation of molecular computation I have ever seen.

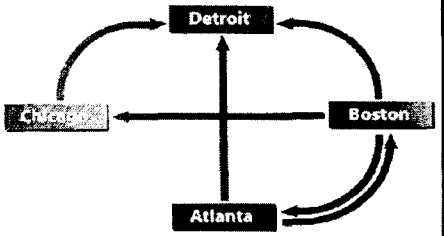
An Example Problem Illustrated

Hamiltonian Path Problem

Consider a map of cities connected by certain nonstop flights (*top right*). For instance, in the example shown here, it is possible to travel directly from Boston to Detroit but not vice versa. The goal is to determine whether a path exists that will commence at the start city (Atlanta), finish at the end city (Detroit) and pass through each of the remaining cities exactly once. In DNA computation, each city is assigned a DNA sequence (*ACTTGCAG* for Atlanta) that can be thought of as a first name (*ACTT*) followed by a last name (*GCAG*). DNA flight numbers can then be defined by concatenating the last name of the city of origin with the first name of the city of destination (*bottom right*).

The complementary DNA city names are the Watson-Crick complements of the DNA city names in which every C is replaced by a *G*, every *G* by a *C*, every *A* by a *T*, and every *T* by an *A*. (To simplify the discussion here, details of the 3′ versus 5′ ends of the DNA molecules have been omitted.) For this particular problem, only one Hamiltonian path exists, and it passes through Atlanta, Boston, Chicago and Detroit in that order. In the computation, this path is represented by *GCAGTCGGACTGGGCTATGTCCGA*, a DNA sequence of length 24. Shown at the left is the map with seven cities and 14 nonstop flights used in the actual experiment. —*L.M.A.*



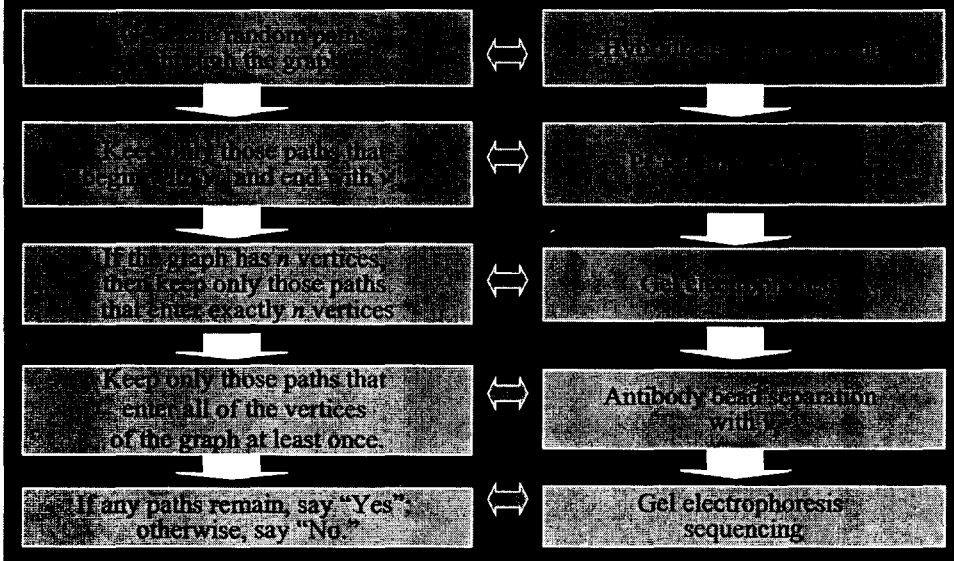


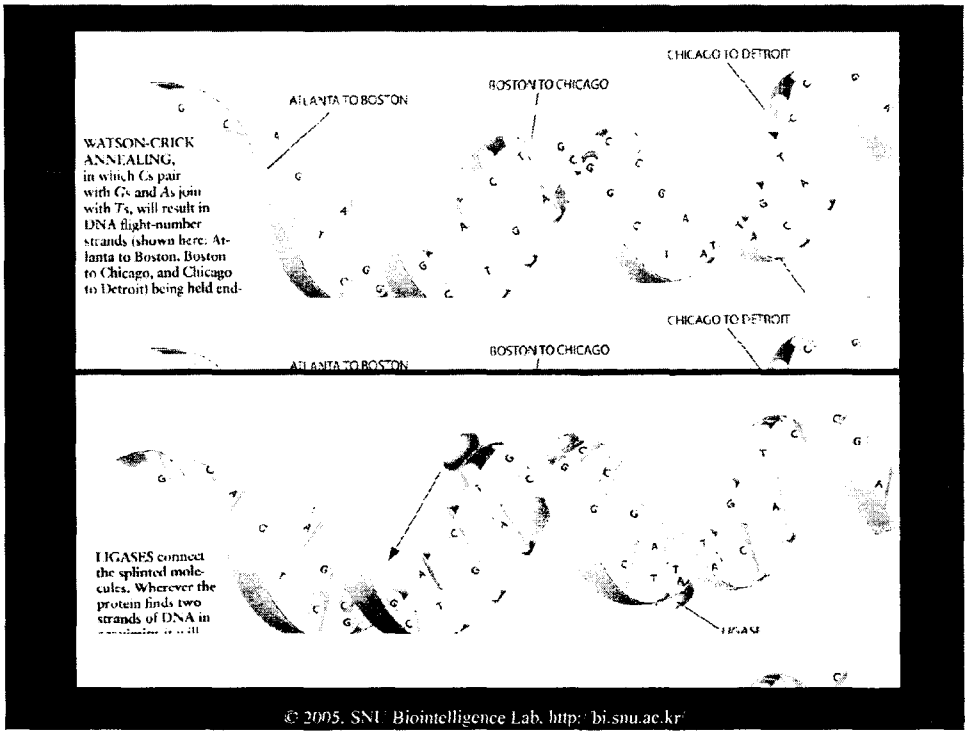
CITY	DNA NAME	COMPLEMENT
ATLANTA	ACTTGCAG	TGAACGTC
BOSTON	TCGGACTG	AGCCTGAC
CHICAGO	GGCTATGT	CCGATACA
DETROIT	CCGAGCAA	GGCTCGTT

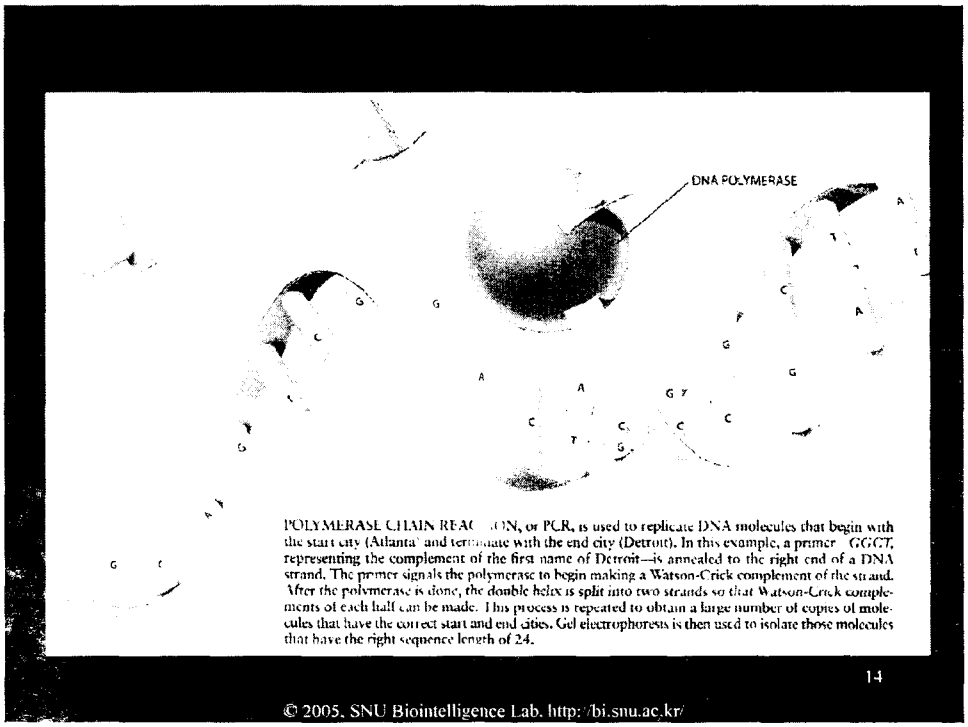
FLIGHT	DNA FLIGHT NUMBER
ATLANTA - BOSTON	GCAGTCGG
ATLANTA - DETROIT	GCAGCCGA
BOSTON - CHICAGO	ACTGGGCT
BOSTON - DETROIT	ACTGCCGA
BOSTON - ATLANTA	ACTGACTT
CHICAGO - DETROIT	ATGTCCGA

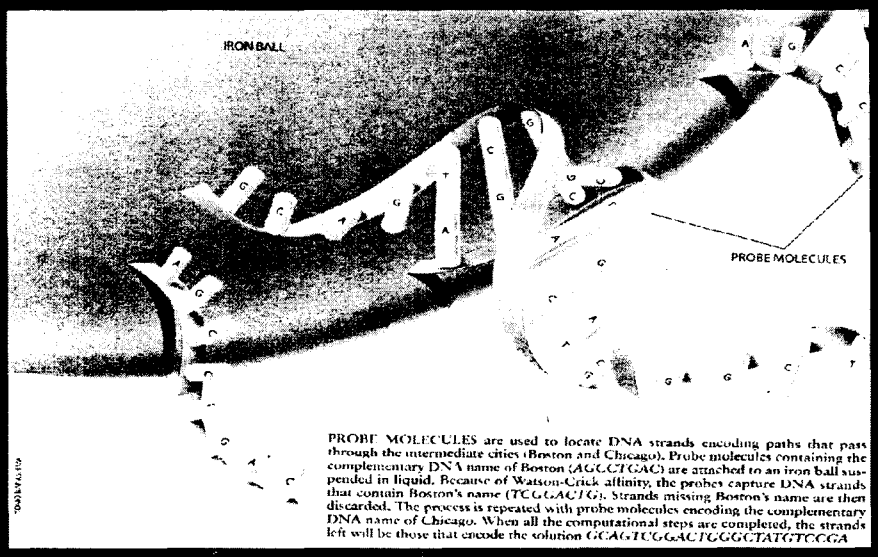
[Adleman, *Scientific American*, 1998]

Bio-Lab Procedure




ATLANTA TO BOSTON
BOSTON TO CHICAGO
CHICAGO TO DETROIT
WATSON-CRICK ANNEALING, in which Cs pair with Gs and As join with Ts, will result in DNA flight-number strands (shown here: Atlanta to Boston, Boston to Chicago, and Chicago to Detroit) being held end-
ATLANTA TO BOSTON
BOSTON TO CHICAGO
CHICAGO TO DETROIT
LIGASES connect the splinted molecules. Wherever the protein finds two strands of DNA in
© 2005, SNU Biointelligence Lab, http://bi.snu.ac.kr/


DNA POLYMERASE
POLYMERASE CHAIN REACTION, or PCR, is used to replicate DNA molecules that begin with the start city (Atlanta) and terminate with the end city (Detroit). In this example, a primer—GGGT, representing the complement of the first name of Detroit—is annealed to the right end of a DNA strand. The primer signals the polymerase to begin making a Watson-Crick complement of the strand. After the polymerase is done, the double helix is split into two strands so that Watson-Crick complements of each half can be made. This process is repeated to obtain a large number of copies of molecules that have the correct start and end cities. Gel electrophoresis is then used to isolate those molecules that have the right sequence length of 24.
14
© 2005, SNU Biointelligence Lab, http://bi.snu.ac.kr/



PROBE MOLECULES are used to locate DNA strands encoding paths that pass through the intermediate cities (Boston and Chicago). Probe molecules containing the complementary DNA name of Boston (*AGCCTGAC*) are attached to an iron ball suspended in liquid. Because of Watson-Crick affinity, the probes capture DNA strands that contain Boston's name (*TCGGACTG*). Strands missing Boston's name are then discarded. The process is repeated with probe molecules encoding the complementary DNA name of Chicago. When all the computational steps are completed, the strands left will be those that encode the solution *GCAGTCGGACTGGGCTATGTCGGA*

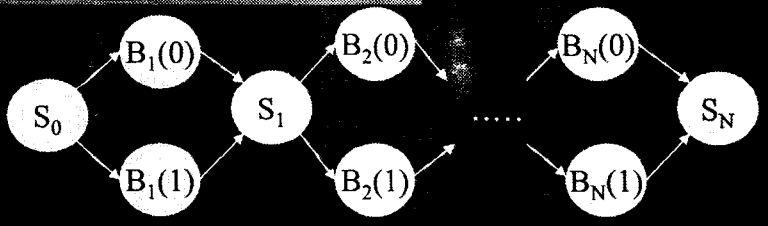
Basic Ideas in DNA Computing

- Exhaustive search
- Parallelism
- Density
- Miniaturization
- Energy efficiency

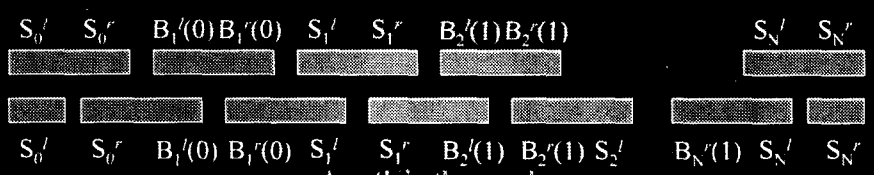
Recent Applications

- Computational
  - Cryptography (Boneh et al., 1995)
  - Chess (Landweber et al., *PNAS* 2000)
  - 20-var 3-SAT (Adleman, *Science* 2002)
  - Tic-Tac-Toe (Stojanovic, *Nature Biotech* 2004)
- Biology and Medicine
  - Genetic switch (Weiss et al., *PNAS* 2002)
  - Gene control (Benenson et al., *Nature* 2004)
- Nanotechnology
  - DNA crystals (Winfree & Seeman et al., *Nature* 1998)
  - Molecular tweezer (Yurke & Turberfield et al., *Nature* 2000)
  - TX complexes (Reif & Seeman et al, *Nature* 2000)
  - Tiles (LaBean & Reif, 2003)

Breaking DES



DES circuit initial graph



A path in the graph

[Boneh et al., 1995]

Self-Assembly of DNA Crystals

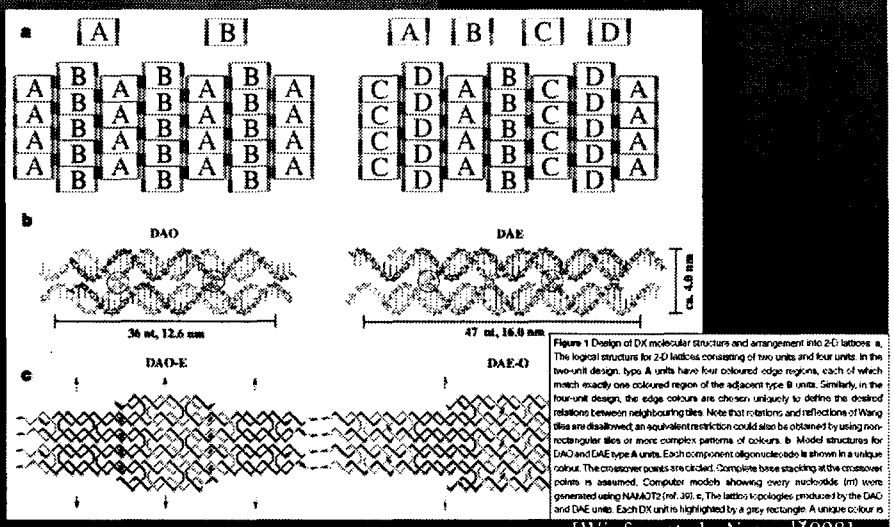


Figure 1 Design of DX molecular structure and arrangement into 2-D lattices. a, The logical structure for 2-D lattices consisting of two units and four units. In the two-unit design, type A units have four coloured edge regions, each of which match exactly one coloured region of the adjacent type B units. Similarly, in the four-unit design, the edge colours are chosen uniquely to define the desired relations between neighbouring tiles. Note that rotations and reflections of Wang tiles are disallowed; an equivalent restriction could also be obtained by using non-rectangular tiles or more complex patterns of colours. b, Model structures for DAO and DAE type A units. Each component oligonucleotide is shown in a unique colour. The crossover points are circled. Complete base stacking at the crossover points is assumed. Computer models showing every nucleotide (m) were generated using NAMOT2 (ref. 39). c, The lattice topologies produced by the DAO and DAE units. Each DX unit is highlighted by a grey rectangle. A unique colour is

[Winfree et al., *Nature* 1998]

RNA Solution to a Chess Problem

$$((\neg h \wedge \neg f) \vee \neg e) \wedge (\neg g \wedge \neg f) \vee \neg b) \wedge ((\neg d \wedge \neg h) \vee \neg c) \wedge ((\neg c \wedge \neg f) \vee \neg d) \wedge (\neg a \wedge \neg g) \vee \neg f) \wedge (\neg b \wedge \neg f) \vee \neg g) \wedge ((\neg a \wedge \neg c) \vee \neg h) \wedge (\neg d \wedge \neg b) \vee \neg f)$$

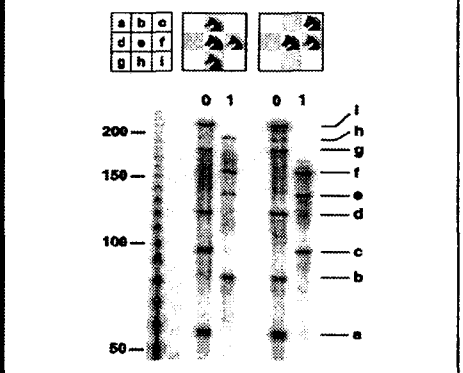
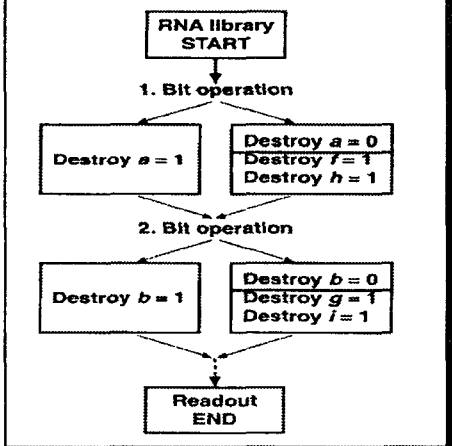


Fig. 3. Multiplex colony PCR readout. Colony PCR (10) allowed rapid screening and recovery of individual library strands. The boards represented by two clones are shown above their respective readouts on a 5% polyacrylamide gel. These clones represent the strings 010 011 010 and 001 011 000. Left lane, 10-bp ladder.



[Faulhammer et al., *PNAS* 2000]

Solving a 20-var 3-CNF Problem

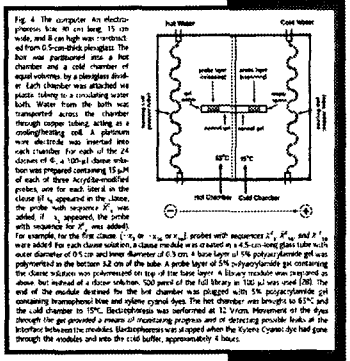
A

$\Phi = (\sim x_3 \text{ or } \sim x_{16} \text{ or } x_{18})$ *and* $(x_5 \text{ or } x_{12} \text{ or } \sim x_9)$ *and* $(\sim x_{13} \text{ or } \sim x_2 \text{ or } x_{20})$ *and* $(x_{12} \text{ or } \sim x_9 \text{ or } \sim x_5)$ *and* $(x_{19} \text{ or } \sim x_4 \text{ or } x_6)$ *and* $(x_9 \text{ or } x_{12} \text{ or } \sim x_5)$ *and* $(\sim x_1 \text{ or } x_4 \text{ or } \sim x_{11})$ *and* $(x_{13} \text{ or } \sim x_2 \text{ or } \sim x_{19})$ *and* $(x_5 \text{ or } x_{17} \text{ or } x_9)$ *and* $(x_{15} \text{ or } x_9 \text{ or } \sim x_{17})$ *and* $(\sim x_5 \text{ or } \sim x_9 \text{ or } \sim x_{12})$ *and* $(x_6 \text{ or } x_{11} \text{ or } x_4)$ *and* $(\sim x_{15} \text{ or } \sim x_{17} \text{ or } x_7)$ *and* $(\sim x_6 \text{ or } x_{18} \text{ or } x_{13})$ *and* $(\sim x_{12} \text{ or } \sim x_9 \text{ or } x_5)$ *and* $(x_{12} \text{ or } x_1 \text{ or } x_{14})$ *and* $(x_{20} \text{ or } x_3 \text{ or } x_2)$ *and* $(x_{10} \text{ or } \sim x_7 \text{ or } \sim x_8)$ *and* $(\sim x_5 \text{ or } x_9 \text{ or } \sim x_{12})$ *and* $(x_{18} \text{ or } \sim x_{20} \text{ or } x_3)$ *and* $(\sim x_{10} \text{ or } \sim x_{18} \text{ or } \sim x_{16})$ *and* $(x_1 \text{ or } \sim x_{11} \text{ or } \sim x_{14})$ *and* $(x_8 \text{ or } \sim x_7 \text{ or } \sim x_{15})$ *and* $(\sim x_8 \text{ or } x_{16} \text{ or } \sim x_{10})$

B

x_1=F, x_2=T, x_3=F, x_4=F, x_5=F, x_6=F, x_7=T, x_8=T, x_9=F, x_{10}=T, x_{11}=T, x_{12}=T, x_{13}=F, x_{14}=F, x_{15}=T, x_{16}=T, x_{17}=T, x_{18}=F, x_{19}=F, x_{20}=F

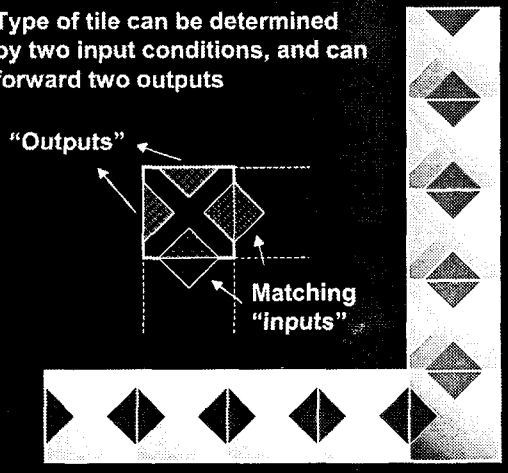
Fig. 1. The computational problem. (A) 20-variable 3-CNF Boolean formula Φ. The symbol "~" indicates "not." (B) The unique truth assignment satisfying Φ.



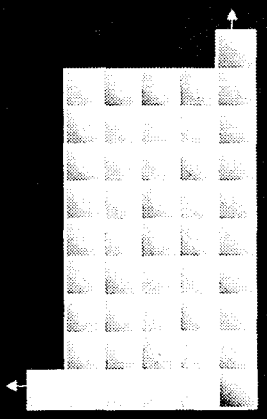
[Braich et al., *Science* 2002]

Binary Counter

Type of tile can be determined by two input conditions, and can forward two outputs

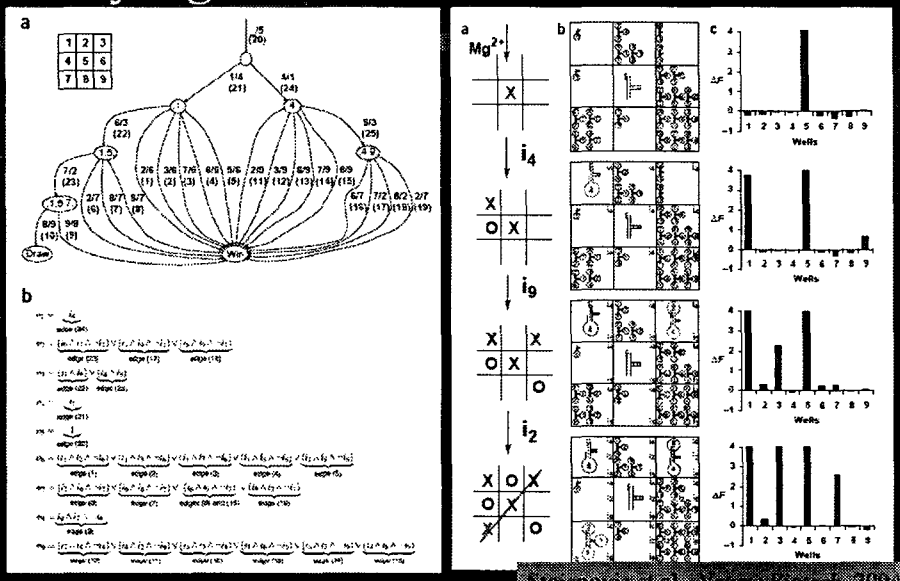


Assembly grows in this direction



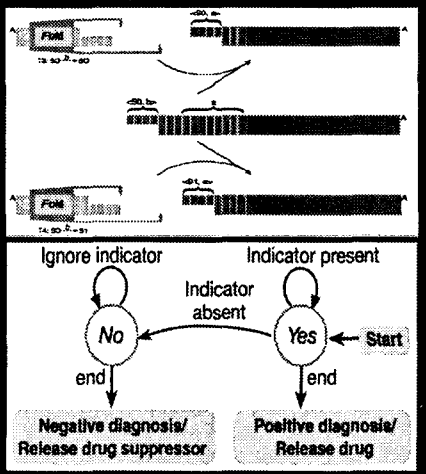
[Winfree et al., *DNAC* 2003]

Playing a Tic-Tac-Toe Game



[Stojanovic et al., *Nature Biotech*, 2004]

DNA as Smart Drugs

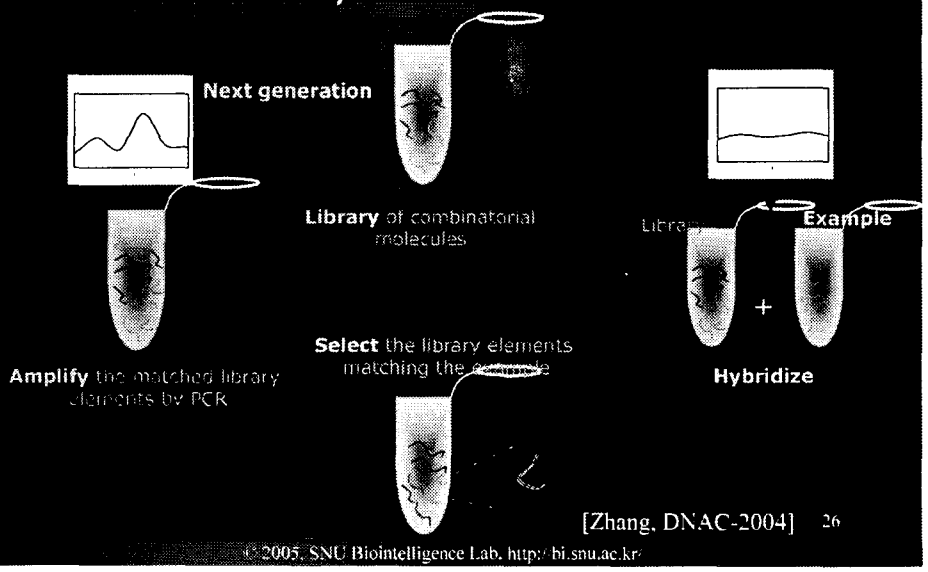


[Benenson *et al.*, *Nature* 2001 & *Nature*, 2004]

PPAP2B↓ & GSTP1↓ & PIM1↑ & HEPSIN↑ → Administer GTTGGTATTGCACAT

The Probabilistic Library Model (PLM)

PLM: Using Molecules to Represent the Probability Distribution



[Zhang, DNAC-2004]

The Probabilistic Library Model (PLM)

- A library of DNA molecules represents the empirical distribution of data variables.
- Each library element consists of n variables, $X_1,\ldots, X_n$.
- A big number of molecules are maintained in the library.
  - $L = \{ x_i \mid i = 1,\ldots,N \}$
  - N: typically 10^{15} with 10 nM
- Duplications of elements are allowed. And the number of duplications is proportional to the strength of the element.
- The library is so maintained that it represents the joint probability distribution of the data variables.
  - $P(X) = P(X_1,\ldots, X_n)$ [Zhang. DNAC-2004]

The PLM (cont'd)

- The probability of variable X_k having x_k is computed chemically by putting in the library the complementary sequence - x_k of the query sequence x_k and extracting the hybridized sequences followed by normalization.
  - $P(X_k=x_k) \sim c(x_k)/|L|$
- Conditional probabilities can be computed by the relative frequencies of the molecules.
  - $P(X_i X_k) = P(X_i, X_k) / P(X_k)$
  - Here $P(X_i=x_i, X_k=x_k) \sim c(x_i x_k)/|L|$ and $P(X_k=x_k) \sim c(x_k)/|L|$
- The library as an ensemble
- Probabilistic computation
- Massively parallel computation of probabilities

The PLM as a Pattern Classifier

- Assume L contains sequence patterns x_i with known labels y_i (training set)
  - $L = \{ (x_i, y_i) \mid i = 1,...,N \}$
  - $x_i = \{A,T,G,C\}^m$: observable input, e.g. DNA sequence
  - $y_i = \{0, 1\}$, observable output, e.g. cancer or normal

$$P(X,Y)$$

- Given a query sequence x_q
  - Put $-x_q$ into the test tube -x: complementary to x

$$P(Y \mid X)$$

- Find the correct class y_q for x_q (classification)
  - $y_q = 1$: cancer
  - $y_q = 0$: normal

Classification Decision: Probabilistic Formulation

- $P(X)$: Probability of observing protein sequence X
- $P(X,Y)$: Probability of sequence X being in class Y
- $P(X,Y,Z)$: Probability of sequence X being in class Y with some parameter Z
- $P(Y|X)$: Conditional probability of class Y given X

$$P(X) = \sum_Y P(X,Y)$$

$$P(X,Y) = \sum_Z P(X,Y,Z)$$

$$P(Y \mid X) = \frac{P(X,Y)}{P(X)} = \frac{P(X,Y)}{\sum_Y P(X,Y)}$$

Classification Learning: In Vitro Evolution

1. Start with library L of random samples (molecules)
2. Given a training sample $s = (x, y)$
3. Classify s using L
   - Extract x → $N(x) := P(x)$
   - Extract Y → $N(x,Y) := P(x,Y)$
   - $y^* = \text{argmax}_Y N(x,Y)$
4. Update L
   - If $y^*=y$, $P(y^*|x) \leftarrow d\,N(y^*|x)$ with $d>1$
   - Otherwise, $P(y^*|x) \leftarrow d\,N(y^*|x)$ with $d<1$

$$P(Y = y \mid X = x) = \frac{P(x,y)}{P(x)}$$

$$P(y^* \mid x) \Leftarrow P'(y^* \mid x) \cong N'(y^* \mid x)$$

$$N'(y^* \mid x) = \delta N(y^* \mid x) = \sum_{z \in Z} \delta N(y^*, z \mid x)$$

$$= \sum_{z \in Z} \beta N\ y\ (*\ z)(\ )\alpha N\ z\ \ x$$

$$\text{with} \begin{cases} \delta,\alpha,\beta \geq 1.0 \text{ if } y^*=y \\ \delta,\alpha,\beta < 1.0 \text{ if } y^* \neq y \end{cases}$$

The Learning Rule Leads to Bayesian Update

$$N'(y^* \mid x) = \sum_{z \in Z} \beta N(y^* \mid z,x)\alpha N(z \mid x)$$

$$\sum_{z \in Z} N'(y^* \mid z,x) N'(z \mid x)$$

$$P'(z \mid y,x) = \frac{P'(y \mid z,x)P'(z \mid x)}{P'(y \mid x)}$$

$$\frac{P'(y \mid z,x)P'(z \mid x)}{\sum_{z \in Z} P'(y \mid z,x)P'(z \mid x)}$$

$$\cong \frac{N'(y \mid z,x)N'(z \mid x)}{\sum_{z \in Z} N'(y \mid z,x)N'(z \mid x)}$$

Update of $N(y^*|x)$ leads to update of the posterior probability distribution $P(z|y,x)$, resulting in a Bayesian learning rule for classification learning with DNA computing

$$Q\ \ P(y \mid x) = \sum_{z \in Z} P(y,z \mid x)$$

$$= \sum_{z \in Z} P\ y\ \ z\ x\ P(z\ \ x\ |$$

[Zhang, DNAC-2004]

PLM vs. Probabilistic Model-Building GAs (or EDAs)

- Some recent genetic and evolutionary algorithms build explicit probabilistic models for the population.
- These distribution-estimation algorithms (EDAs) generate offspring by sampling from the probabilistic model rather than using crossover and mutation.
- Like EDA, the probabilistic library model (PLM) generates the offspring by sampling from a probabilistic distribution.
- Unlike EDA, in PLM no extra probabilistic model is built. The PLM itself represents a probability distribution.
- The use of a huge number of molecules (10^{15} or more) enables the test tube to represent the empirical probability distribution.

Why Try Molecular EC?

- 6.022×10^{23} molecules / mole
- Massively Parallel Search
  - Desktop: 10^9 operations / sec
  - Supercomputer: 10^{12} operations / sec
  - 1 μmol of DNA: 10^{26} reactions
- Favorable Energetics: Gibbs Free Energy

  $\Delta G = -8kcal\ \mathrm{mol}^{-1}$
  - 1 J for 2×10^{19} operations
- Storage Capacity: 1 bit per cubic nanometer
- The fastest supercomputer vs. DNA computer
  - 10^6 op/sec vs. 10^{14} op/sec
  - 10^9 op/J vs. 10^{19} op/J (in ligation step)
  - 1bit per 10^{12} nm^3 vs. 1 bit per 1 nm^3 (video tape vs. molecules)

Molecular Programming (MP): In Vitro Evolution of Genetic Programs

Molecular Programming (MP): Evolving Genetic Programs in a Test Tube

- Theory
  - Bayesian evolution [Zhang, CEC-99; Zhang, Handbook-2003]
- Model
  - Probabilistic library model [Zhang, DNAC-2004 & 2005]
- Algorithm
  - Molecular algorithms [Zhang, GP-98]
- Representation
  - Decision lists [Zhang, GECCO-2005]
- Operators
  - Molecular operators for variation and selection [Zhang, GECCO-2005]

Molecular Programming of the PLM

1. Let the library L represent the current distribution $P(X,Y)$.
2. Get a training example $(\mathbf{x},y)$.
3. Classify $\mathbf{x}$ using L as follows
   - 3.1 Extract all molecules matching $\mathbf{x}$ into M.
   - 3.2 From M separate the molecules into classes:
     - Extract the molecules with label Y=0 into M^0
     - Extract the molecules with label Y=1 into M^1
   - 3.3 Compute $y^*=\text{argmax}_{Y\in\{0,1\}}|M^Y|/|M|$
4. Update L
   - If y^*=y, then $L_n \leftarrow L_{n-1}+\{\Delta c(\mathbf{u}, v)\}$ for $\mathbf{u}=\mathbf{x}$ and v=y for $(\mathbf{u}, v)\in L_{n-1}$.
   - If $y^*\neq$y, then $L_n \leftarrow L_{n-1}-\{\Delta c(\mathbf{u}, v)\}$ for $\mathbf{u}=\mathbf{x}$ and $v \neq y$ for $(\mathbf{u}, v)\in L_{n-1}$
5. Goto step 2 if not terminated.

[Zhang, DNAC-2004]

Step 1: Probability Distribution in the Library

$$D=\{(\mathbf{x}_i,y_i)\}_{i=1}^{K} \quad \mathbf{x}_i=(x_{i_1},x_{i_2},\cdots,x_{i_n})\in\{0,1\}^n, \quad y_i\in\{0,1\}$$

$$P(X,Y)\approx\frac{1}{|L|}\sum_{i=1}^{|L|}f_i^{(n)}(X_1,X_2,\ldots,X_n,Y)$$

Step 2: Presentation of an Example (or Query)

$$P(\mathbf{x}_i,y_i\mid\mathbf{x}_q,y_q)=\frac{\exp(-\Delta G(\mathbf{x}_i,y_i\mid\mathbf{x}_q,y_q))}{\sum_j\exp(-\Delta G(\mathbf{x}_j,y_i\mid\mathbf{x}_q,y_q))}$$

Step 3: Classify the Example (Inference)

$$y^* = \arg\max_{Y \in \{0,1\}} P(Y \mid \mathbf{x}) = \arg\max_{Y \in \{0,1\}} \frac{P(Y, \mathbf{x})}{P(\mathbf{x})}$$

$$c(\mathbf{x})/|L| = |M|/|L| \approx P(\mathbf{x})$$

$$y^* = \arg\max_{Y \in \{0,1\}} c(Y \mid \mathbf{x})/|M| = \arg\max_{Y \in \{0,1\}} c(Y \mid \mathbf{x}) \approx \arg\max_{Y \in \{0,1\}} P(Y \mid \mathbf{x})$$

$$c(Y \mid \mathbf{x})/|M| = |M^Y|/|M| \approx P(Y \mid \mathbf{x})$$

Step 4: Update the Library (Learning)

$$L \leftarrow L + \{(\mathbf{u}, v)\} \qquad L_v \leftarrow L_v - \{(\mathbf{u}, v)\}$$

$$P_n(X, Y \mid \mathbf{x}, y) = (1+\delta) P_{n-1}(X, Y \mid \mathbf{x}, y)$$

$$\delta = \frac{P(\mathbf{x}, y \mid X, Y) - P(\mathbf{x}, y)}{P(\mathbf{x}, y)}$$

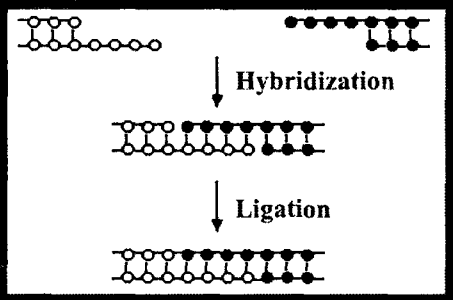
$$\delta = \frac{\Delta c(\mathbf{x}, y)}{c_{n-1}(\mathbf{x}, y)}$$

Molecular Operators

- Variation
  - Ligation
  - Restriction
  - Mutation (PCR)
- Selection
  - Gel electrophoresis
  - Affinity separation (beads)
  - Capillary electrophoresis
- Amplification
  - Polymerase chain reaction (PCR)
  - Rolling circle amplification (RCA)

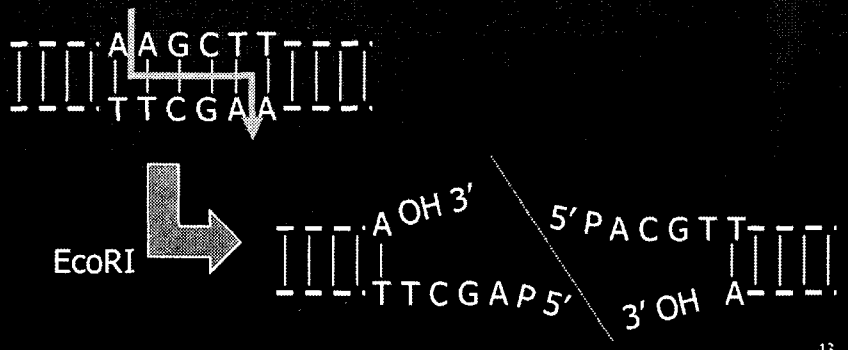
Variation: Hybridization & Ligation

- Hybridization
  - base-pairing between two complementary single-strand molecules to form a double stranded DNA molecule
- Ligation
  - Joining DNA molecules together
- Usually used for candidate solution generation.



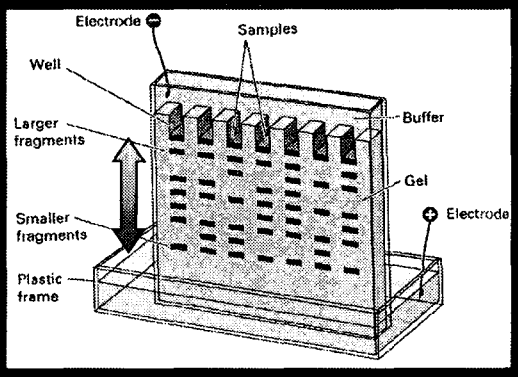
Variation: Restriction

- Cut the specific DNA site.
- Solution detection or filtering step

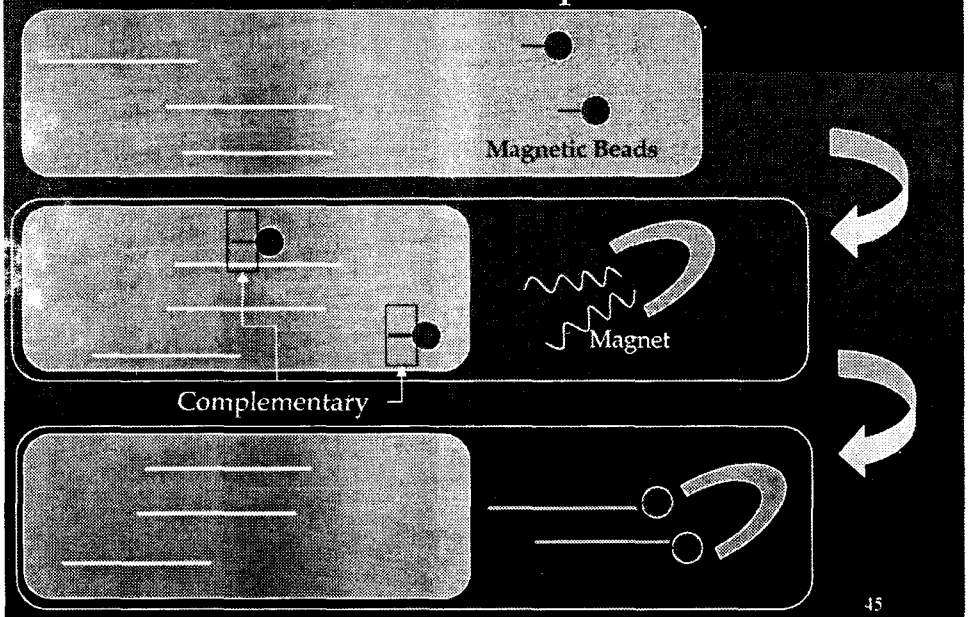


Selection: Gel Electrophoresis

- Detection desired solutions.
- Separate solution molecules by length



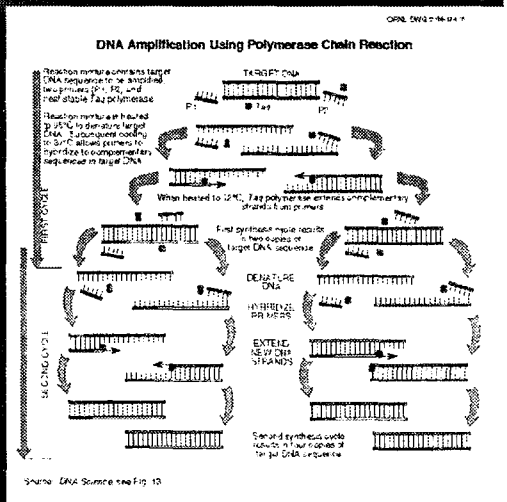
Selection: Bead Separation



Detect & separate the specific DNA

Amplification: PCR

- Polymerase chain reaction
- Amplifies (produces identical copies of) selected dsDNA molecules.
- Make 2^n copies (n : number of iteration)
- Used to filter solutions of detection.

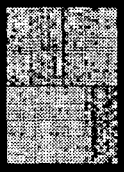


Application to Leukemia Diagnosis

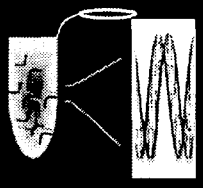
120 samples from
60 leukemia patients

Gene expression data

&



Class: ALL/AML



Training with
6-fold validation

Diagnosis

[Golub et al., Nature Genetics, 2003]

a) z_1 : (x_1=0, x_2=1, x_3=0, y=1)
z_2 : (x_1=0, x_2=0, x_3=1, x_4=0, x_5=0, y=0)
z_3 : (x_2=1, x_4=1, y=1)
z_4 : (x_2=1, x_3=0, x_4=1, y=0)

b) z_1 : AAAACCAATTGGAAGGCCATGCGG
z_2 : AAAACCAATTCCAAGGGGCCTTCCCCAACCATGCCC
z_3 : AATTGGCCTTGGATGCGG
z_4 : AATTGGAAGGCCCCTTGGATGCCC

where

AAAA	x_1	CCTT	x_4	CC	0
AATT	x_2	CCAA	x_5	GG	1
AAGG	x_3	ATGC	y		

Figure 1: Population of genetic programs in two different representations: (a) set of decision lists, (b) library of DNA molecules corresponding to (a). The DNA code shown are illustration-purposes only and this design does not fully reflect the biochemical properties of the sequences.

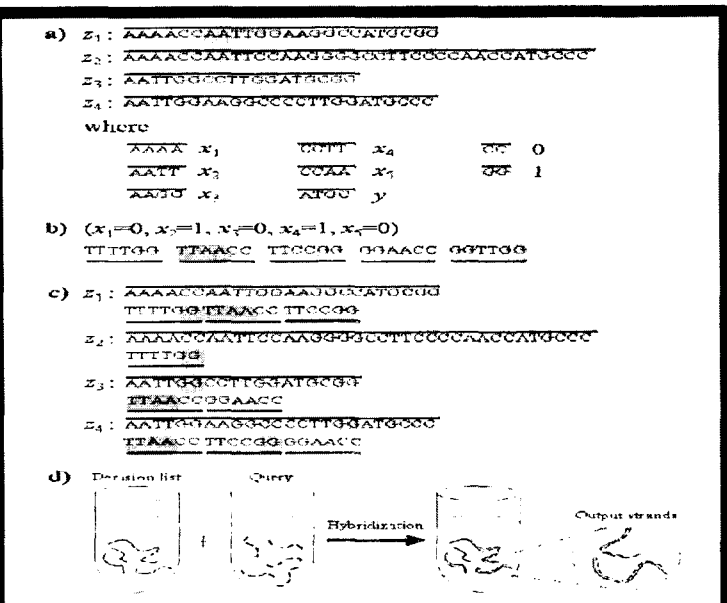


Figure 2: Illustration of the decision-making procedure using the population of DNA-encoded genetic programs: (a) Library of decision lists, (b) query sample (in multiple copies), (c) decision lists hybridized with query samples,(d) schematic for illustrating the whole decision procedure.

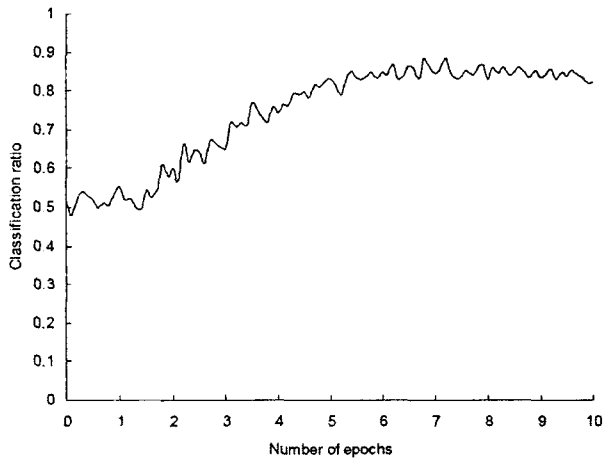


Figure 5: Fitness evolution of the population of molecular genetic programs. Though there are fluctuations the fitness values tend to converge 90 % accuracy. The reproduction rate was 0.01.

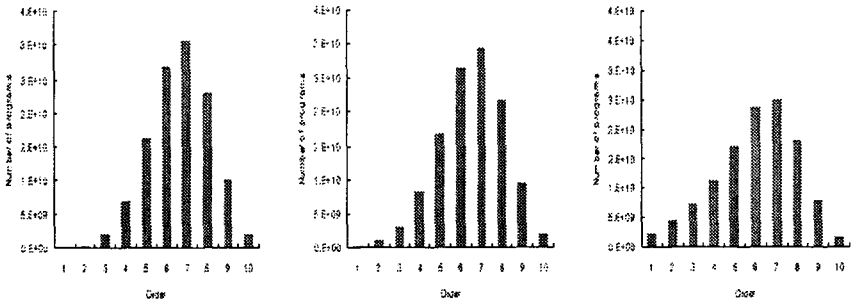


Figure 6: Distribution of the size of genetic programs. Shown are the number of programs of each size in the final population in a run. It shows the tendency that, as generation goes on, smaller programs are used more frequently than larger ones. The reproduction rate was 0.01.

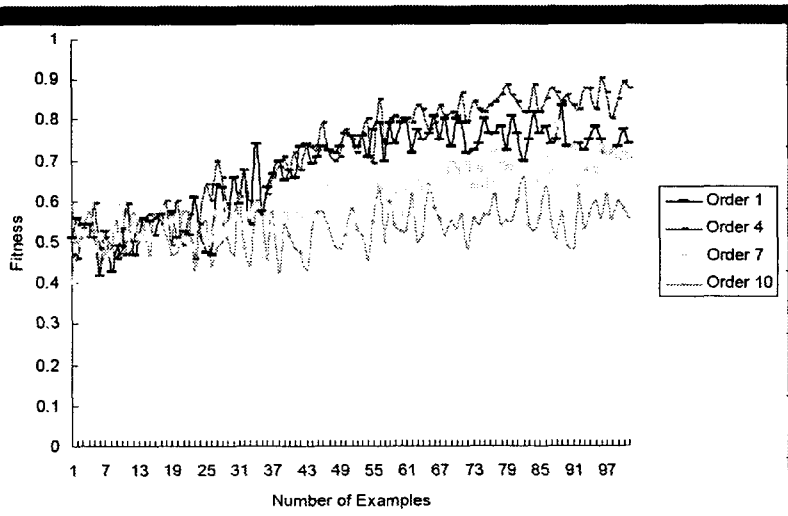


Figure 7: Fitness curve for runs with fixed-size programs. Shown are average fitness values for runs with programs of fixed-order 1, 4, 7, and 10.

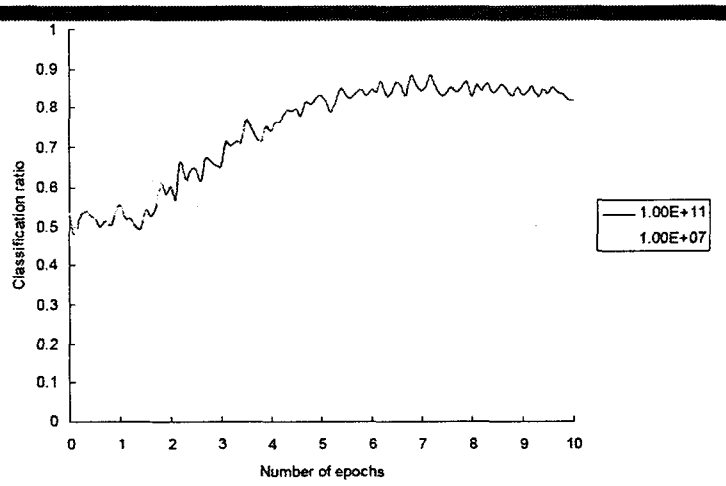
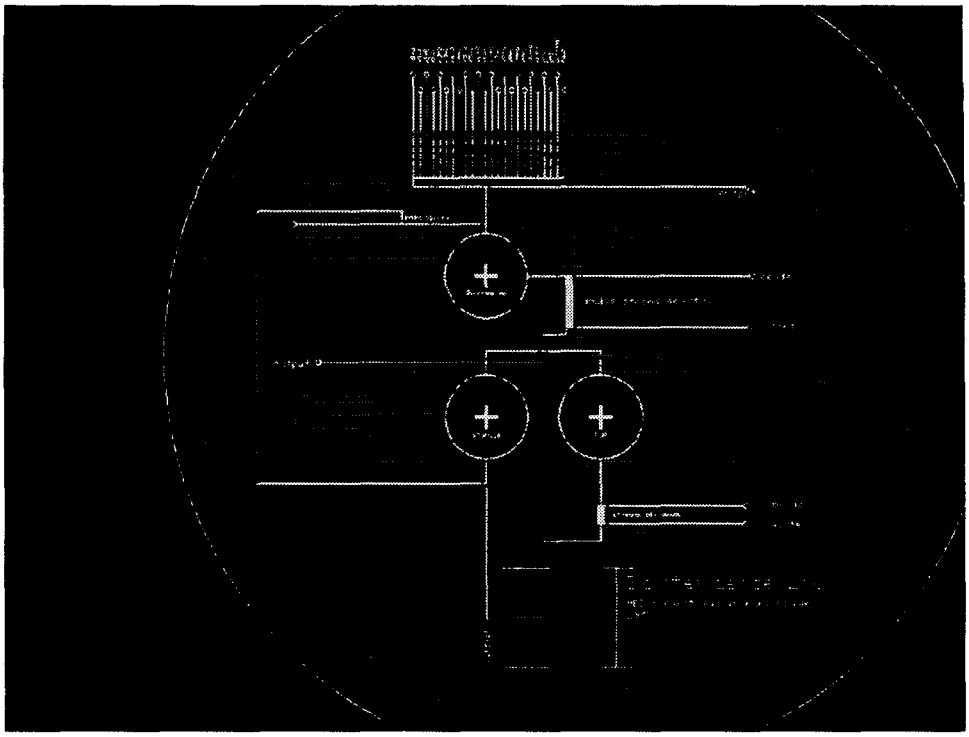


Figure 8: Effect of population size on ensemble performance. Shown are the best-fitness curves for population sizes of 10^{11} (in our experiments) 10^{7} and (subsampling case for testing). The results show that too much subsampling degrades the performance.

MP vs. GP

	Genetic Programming (GP)	Molecular Programming (MP)
Representation	Variable-size trees	Variable-length lists
Variation	Random xover, mutation	Combinatorial sampling
Selection	Proportional selection	Amplification (PCR)
Population size	~ $O(10^4)$	~ $O(10^{15})$
Parallelism	Can be parallelized	Inherently parallel
Solution	Single individual	Ensemble of individuals
Interaction	2D matrix	3D collision
Material	Silicon (dry, hard)	Carbon (wet, soft)



Discussion

Molecular Programming (MP) as a New Paradigm for Molecular Computing

- Scalability
  - *Problem:* For big problems, exhaustive search does not work.
  - *Solution:* Evolutionary search
- Reliability
  - *Problem:* DNA reaction is error-prone.
  - *Solution:* Probabilistic formulation
- Fault tolerance
  - *Problem:* What if a single molecule malfunctions?
  - *Solution:* Ensemble machine approach
- Design
  - *Problem:* How to design the decision (or diagnosis) rules?
  - *Solution:* Evolutionary learning from examples

In Vitro Evolution vs. In Silico Evolution

	In Vitro Evolution	In Silico Evolution
Processing	Ballistic	Hardwired
Medium	Liquid (wet)	Solid (dry)
Communication	3D collision	2D switching
Configuration	Amorphous (asynchronous)	Fixed (synchronous)
Parallelism	Massively parallel	Sequential
Speed	Fast (millisec)	Ultra-fast (nanosec)
Reliability	Low	High
Density	Ultrahigh	Very high
Reproducibility	Probabilistic	Deterministic

New Research Issues

- Representation
  - New representation schemes under molecular constraints
  - 2D and 3D structures for molecular genetic programs
  - Parsimony/bloat issues
- Operators
  - New molecular operators under thermodynamic constraints
  - Biochemical wet operators
  - Physical implementation of operators (e.g. physical simulated annealing)
- Theory
  - The role of a huge population size
  - Theory for guiding experimental procedures (e.g., SELEX)
  - EC theories of the origins of life
- Applications
  - Physical evolution
  - Bio, pharma, medicine
  - Nanotechnology
  - Molecular electronics
  - Molecular robotics

Acknowledgements

Collaborating Labs

- Biointelligence Laboratory, Seoul National University
- Biochemistry Lab, Seoul National Univ. Medical School
- Cell and Microbiology Lab, Seoul National University
- Advanced Proteomics Lab, Hanyang University
- DigitalGenomics, Inc.
- GenoProt, Inc.

Supported by

- National Research Lab Program of Min. of Sci. & Tech.
- Next Generation Tech. Program of Min. of Ind. & Comm.

More Information at

- http://bi.snu.ac.kr/MEC/
- http://cbit.snu.ac.kr/

Books and Web Sites

Books (General)

- Calude, C.S., Casti, J. and Dinneen, M.J. (Eds.) *Unconventional Models of Computation*, Springer, 1998.
- Eigen, M. and Winkler, R., *Laws of the Game: How the Principles of Nature Govern Chance*, Princeton University Press, 1993 (English translation).
- Kauffman, S.A., *The Origins of Order: Self-Organization and Selection in Evolution*, Oxford University Press, 1993.
- Kueppers, B.-O., *Molecular Theory of Evolution: Outline of a Physico-Chemical Theory of the Origin of Life*, Springer, 1983.
- Landweber, L.F., Winfree, E. (Eds.) *Evolution as Computation*, Springer, 2003.
- Page, R.D.M and Holmes, E.C., *Molecular Evolution: A Phylogenetic Approach*, Blackwell Science, 1998.
- Scheutz, M. (Ed.) *Computationalism: New Directions*, MIT Press, 2000.
- Sienko, T., Adamatzky, A., Rambidi, N.G., and Conrad, M. (Eds.) *Molecular Computing*, MIT Press, 2003.

Some References

- Adleman, L., "Computing with DNA," *Scientific American*, 34-41, 1998.
- Conrad, M., "Molecular computing: The lock-key paradigm," *IEEE Computer*, 25(1): 11-20, 1992.
- Joyce, G. F. "The antiquity of RNA-based evolution," *Nature*, 418: 214-221, 2002.
- Seeman, N. C., "Biochemistry and structural DNA nanotechnology: An evolving symbiotic relationship," *Biochemistry*, 42(24): 7259-7269, 2003.
- Tuerk, C. and Gold, L., "Systematic evolution of ligands by exponential enrichment: RNA ligands to bacteriophage T4 DNA polymerase," *Science*, 249(4968): 505-510, 1990.
- Wright, M. C. and Joyce, G. F., "Continuous in vitro evolution of catalytic function," *Science*, 276: 614-617, 1997.
- Zhang, B.-T. and Jang, H.-Y., Molecular programming: Evolving genetic programs in a test tube, *Proc. Genetic and Evolutionary Computation (GECCO-2005)*, Washington, D.C., 2005 (to appear)
- Zhang, B.-T. and Jang, H.-Y., A Bayesian algorithm for in vitro molecular evolution of pattern classifiers, *Proc. 10th Int. Conf. on DNA Computing*, LNCS 3384: pp. 458-467, 2005.

Web Sites

- California Inst. of Tech. (Erik Winfree)
- Duke Univ. (John Reif)
- Harvard Univ. (Jack Szostak)
- MIT (Tom Knight)
- New York Univ. (Ned Seeman)
- Scripps Res. Inst. (Gerald Joyce)
- Seoul National Univ. (Tak Zhang)
- Univ. of Bonn (Michael Famulok)
- Univ. of Southern California (Leon Adleman)
- Univ. of Tokyo (Masami Hagiya)
- Univ. of Vienna (Peter Schuster)
- Weizmann Inst. of Tech. (Ehud Shapiro)