DNA Computation: Results, Trends, and Perspectives

Giuditta Franco

Department of Computer Science
University of Verona

Computing with Biomolecules - UC
Vienna, 27 August 2008

Talk Contents

1 Introduction to DNA Computing

Talk Contents

1. Introduction to DNA Computing
    - State of the Art, with fresh news from Prague (DNA14, Int. Meeting on DNA Computing, June 2-6)

Talk Contents

1. Introduction to DNA Computing
   - State of the Art, with fresh news from Prague (DNA14, Int. Meeting on DNA Computing, June 2-6)
2. XPCR Technique as a variant of PCR

Talk Contents

1. Introduction to DNA Computing
   - State of the Art, with fresh news from Prague (DNA14, Int. Meeting on DNA Computing, June 2-6)
2. XPCR Technique as a variant of PCR
   - XPCR-Based Recombination Algorithm

Talk Contents

1. Introduction to DNA Computing
    - State of the Art, with fresh news from Prague (DNA14, Int. Meeting on DNA Computing, June 2-6)
2. XPCR Technique as a variant of PCR
    - XPCR-Based Recombination Algorithm
    - Recombination Witness Strings

Talk Contents

1. Introduction to DNA Computing
   - State of the Art, with fresh news from Prague (DNA14, Int. Meeting on DNA Computing, June 2-6)
2. XPCR Technique as a variant of PCR
   - XPCR-Based Recombination Algorithm
   - Recombination Witness Strings
3. Conclusions and Open Problems

DNA Computation

Information processing requires a strategy of systematical data transformation by basic operations called computations.

DNA Computation

Information processing requires a strategy of systematical data transformation by basic operations called computations.

The definition of such operations depends on the type of data to which they apply, as well as on their organization within physical supports.

DNA Computation

Information processing requires a strategy of systematical data transformation by basic operations called computations.

The definition of such operations depends on the type of data to which they apply, as well as on their organization within physical supports.

$$\text{Input data} \overset{\textit{encoding}}{\longleftrightarrow} \text{DNA} \quad \overset{\textit{bio-steps}}{\rightsquigarrow} \quad \text{DNA} \overset{\textit{decoding}}{\longleftrightarrow} \text{Output data}$$

Information is stored in bio-polymers, enzymes (molecular biology and genetic engineering techniques) manipulate them in a massively parallel way, according to strategies producing universal operations [G.F., V. Manca, Soft Computing 2005].

DNA Algorithms

A feature common to molecular and quantum computations is the possibility to be observed and captured only at certain moments.

DNA Algorithms

A feature common to molecular and quantum computations is the possibility to be observed and captured only at certain moments. In both cases, the step by step exact effect of the operations is out of control, and one alters the system when observing the result.

DNA Algorithms

A feature common to molecular and quantum computations is the possibility to be observed and captured only at certain moments. In both cases, the step by step exact effect of the operations is out of control, and one alters the system when observing the result.

Despite this kind of approximation, bio-algorithms driving system behaviours can be written such that the final pool of molecules contains the desired solution.

DNA Algorithms

A feature common to molecular and quantum computations is the possibility to be observed and captured only at certain moments. In both cases, the step by step exact effect of the operations is out of control, and one alters the system when observing the result.

Despite this kind of approximation, bio-algorithms driving system behaviours can be written such that the final pool of molecules contains the desired solution.

DNA algorithms deterministically transform macro-states of DNA pools (where population properties are controlled by macro-steps), while the individual nondeterminism is left unaffected [V. Manca, G.F., Math. Biosci. 2008].

Importance of Experimentation

"No amount of experimentation can ever prove me right; a single experiment can prove me wrong." A. Einstein

Importance of Experimentation

"No amount of experimentation can ever prove me right; a single experiment can prove me wrong." A. Einstein

Laboratory experiments are crucial for DNA computing: the algorithms need to be validated by the scientific method of comparison with experimental results.

Importance of Experimentation

"No amount of experimentation can ever prove me right; a single experiment can prove me wrong." A. Einstein

Laboratory experiments are crucial for DNA computing: the algorithms need to be validated by the scientific method of comparison with experimental results.

An attempt to formalize laboratory experience can be found in [G.F., PhD Thesis, 2006] - what a mathematician should know to design a DNA computing experiment, about encoding and mismatching, DNA quantities, melting temperature.

Main goals of DNA Computing

1. Theoretical models of DNA computation

Main goals of DNA Computing

1. Theoretical models of DNA computation

2. DNA Self-assembly process

Main goals of DNA Computing

1. Theoretical models of DNA computation
2. DNA Self-assembly process
3. Efficiently solving NP-Complete problems

Main goals of DNA Computing

1. Theoretical models of DNA computation
2. DNA Self-assembly process
3. Efficiently solving NP-Complete problems
4. Encoding issues, mainly generated by experimental mismatch problems

Main goals of DNA Computing

1. Theoretical models of DNA computation
2. DNA Self-assembly process
3. Efficiently solving NP-Complete problems
4. Encoding issues, mainly generated by experimental mismatch problems
5. Search of innovative procedures, namely for biological applications.

A few models of DNA Computing

1. Splicing Systems $\mathcal{S} = (\Sigma, A, R)$ (T. Head, Bull. of Math.Biology '87) - In R: $\underline{wp}qx,\ yu\underline{vz} \xrightarrow{r_{(p,q,u,v)}} \underline{wpvz},\ yuqx,$ with $w, p, q, x \in \Sigma^\star$.

   Null context splicing rule r_p has $p = u$ and $qv = \lambda$:
   $\underline{\phi p}\psi,\ \delta\underline{p\eta} \xrightarrow{r_p} \underline{\phi p\eta},\ \delta p\psi,$ where $\phi, p, \psi, \delta, \eta \in \Sigma^\star$.

A few models of DNA Computing

1. Splicing Systems $\mathcal{S} = (\Sigma, A, R)$ (T. Head, Bull. of Math.Biology '87) - In R: $\underline{wp}qx,\ yu\underline{vz} \xrightarrow{r_{(p,q,u,v)}} \underline{wpvz},\ yuqx,$ with $w, p, q, x \in \Sigma^\star$.

   Null context splicing rule r_p has $p = u$ and $qv = \lambda$:
   $\underline{\phi p}\psi,\ \delta\underline{p\eta} \xrightarrow{r_p} \underline{\phi p\eta},\ \delta p\psi,$ where $\phi, p, \psi, \delta, \eta \in \Sigma^\star$.

2. Forbidding-Enforcing Systems [A. Ehrenfeucht and G. Rozenberg, TCS '03] simulating molecular reactions: everything that is not forbidden is allowed.

A few models of DNA Computing

1. Splicing Systems $\mathcal{S} = (\Sigma, A, R)$ (T. Head, Bull. of Math.Biology '87) - In R: $\underline{wp}qx,\ yu\underline{vz} \xrightarrow{r_{(p,q,u,v)}} \underline{wpvz},\ yuqx,$ with $w, p, q, x \in \Sigma^\star$.

   Null context splicing rule r_p has $p = u$ and $qv = \lambda$:
   $\underline{\phi p}\psi,\ \delta\underline{p\eta} \xrightarrow{r_p} \underline{\phi p\eta},\ \delta p\psi,$ where $\phi, p, \psi, \delta, \eta \in \Sigma^\star$.

2. Forbidding-Enforcing Systems [A. Ehrenfeucht and G. Rozenberg, TCS '03] simulating molecular reactions: everything that is not forbidden is allowed.

3. A variant of forbidding-enforcing systems for graphs which models DNA self-assembly was proposed in [G.F., N.Jonoska, Nanotechnology, Science and Computation, 105-118, 2005]

DNA Self-assembly

Self-assembly is a process in which substructures are spontaneously self-ordered into superstructures. It is driven by the selective affinity of the substructures.

DNA Self-assembly

Self-assembly is a process in which substructures are spontaneously self-ordered into superstructures. It is driven by the selective affinity of the substructures.

DNA substructures are formed by DNA filaments partially attached to each other by Watson-Crick complementarity between strands having opposite orientation.

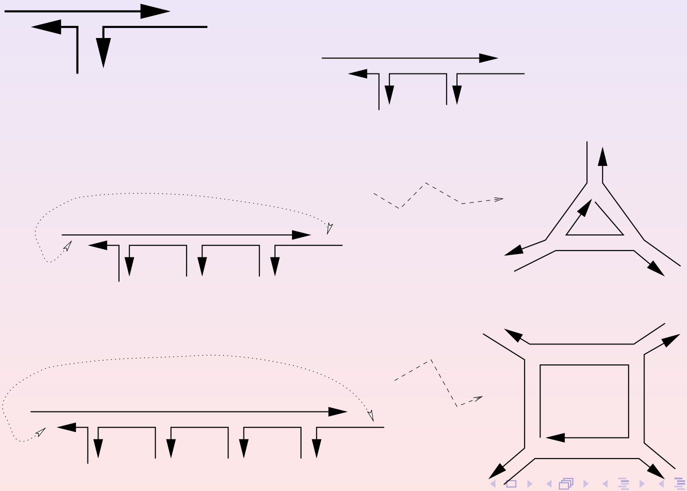
DNA Self-assembly

Self-assembly is a process in which substructures are spontaneously self-ordered into superstructures. It is driven by the selective affinity of the substructures.

DNA substructures are formed by DNA filaments partially attached to each other by Watson-Crick complementarity between strands having opposite orientation.

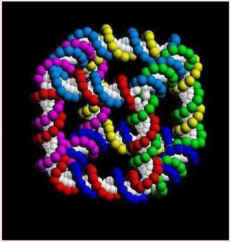
DNA Self-Assembly leads the formation of 3D structure, but DNA is used not only as a structural material, also as ’fuel’, for the construction of DNA machines, walkers, switchers (Nanoday, DNA14), DNA transducers [B. Chakraborty, N. Jonoska, N. Seeman, *Programmable Transducer by DNA Self-assembly*, DNA14].

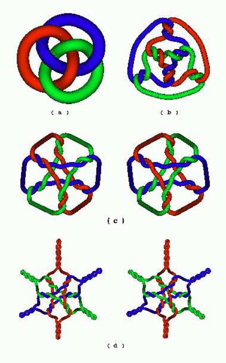
Self-Assembly Basic Step

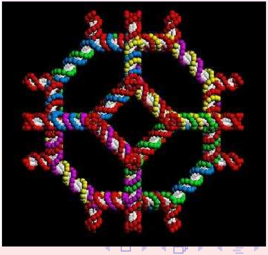


Self-assembly Examples

DNA cube [J.Chen, N. Seeman, Nature 1991], octahedron [Y.Zhang, N. Seeman, J.Am.Chem.Soc. 1994], Borromean rings [C.Mao, W.Sun, N. Seeman, Nature 1997], and Möbius strip (DNA14) were built up in laboratory.







Solving NP-complete problems

After the seminal Adleman's experiment ('94), where the solution of an instance of Hamiltonian Path Problem was found within DNA sequences, Lipton ('95) showed that SAT can be solved by using essentially the same bio-techniques.

Solving NP-complete problems

After the seminal Adleman's experiment ('94), where the solution of an instance of Hamiltonian Path Problem was found within DNA sequences, Lipton ('95) showed that SAT can be solved by using essentially the same bio-techniques.

The exponential amount of DNA in such an *extract model* (brute force search) was shown to be prohibitive to scale-up the algorithms for real problems.

Solving NP-complete problems

After the seminal Adleman's experiment ('94), where the solution of an instance of Hamiltonian Path Problem was found within DNA sequences, Lipton ('95) showed that SAT can be solved by using essentially the same bio-techniques.

The exponential amount of DNA in such an *extract model* (brute force search) was shown to be prohibitive to scale-up the algorithms for real problems.

There have been several attempts to reduce the space and/or the time complexity of DNA algorithms solving NP-complete problems, up to now [DNA14: J.Liu, K.Li, *A DNA-Based Solution to the SAT Problem Using Adleman-Lipton Model with Stickers*, and, X.Wang, *DNA Computing Solve the 3-SAT Problem with a Small Solution Space*].

NP-complete problems have been solved *linearly*

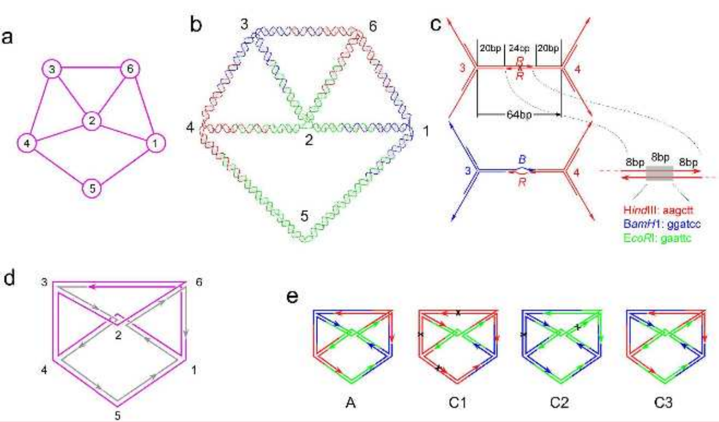
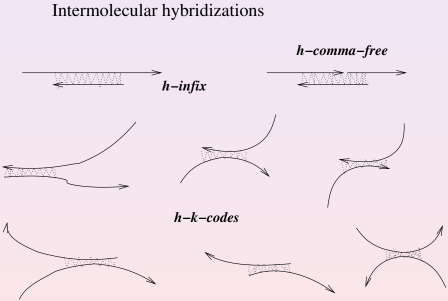


Fig: Courtesy of Natasha Jonoska and Ned Seeman [Personal Communication]

DNA Involution Codes - Examples

Involution codes to avoid mismatch cases arising from experiments.



Forbidden cases for h-infix, h-comma free, and h-k-codes

[N.Jonoska, K. Mahalingam, LNCS 2950, 2004]

Recent results for DNA encoding

- Origami method has been proposed to fold long single stranded DNA molecules into arbitrary two-dimensional shapes, by oligonucleotides 'staple strands' [P. Rothemund, *Folding DNA to create nanoscale shapes and patterns* Nature 2006].

  Free software on www.cdna.dk/origami (DNA14), which imports any shape as design object, automatically finds a folding path and generates the 3D atomic model.

Recent results for DNA encoding

- Origami method has been proposed to fold long single stranded DNA molecules into arbitrary two-dimensional shapes, by oligonucleotides 'staple strands' [P. Rothemund, *Folding DNA to create nanoscale shapes and patterns* Nature 2006].

  Free software on www.cdna.dk/origami (DNA14), which imports any shape as design object, automatically finds a folding path and generates the 3D atomic model.

- A longer double stranded DNA region is preferentially formed over one with a shorter double stranded region [B. Yurke, A. Turberfield, A. Mills, F. Simmel, J. Neumann, *A DNA-fuelled molecular machine made of DNA*, Nature 2000]. Principle used by F. Tanaka, T. Tsuda, M. Hagiya, to design a DNA concentration comparator, DNA14.

Improvements proposed for Whiplash PCR

Whiplash PCR allows a single strand state machine to perform a state transition by polymerase extension [Hagiya et al. '99].

Improvements proposed for Whiplash PCR

Whiplash PCR allows a single strand state machine to perform a state transition by polymerase extension [Hagiya et al. '99].

It is known to suffer from a serious efficiency problem called *back-hybridization*, and researchers are working to overcome it both by *introduction of primers as external signals* [K. Komiya, M. Yamamura, J.Rose, Experimental Validation of Signal Dependent Operation in Whiplash PCR, DNA14] and by exploiting *the polymerase ϕ29*, having an excellent displacement capability [J.Reif, U Majumder, *Isothermal Reactivating Whiplash PCR for Locally Programmable Molecular Computation*, DNA14].

XPCR as an innovative procedure

Novel XPCR-based recombination (extraction, mutagenesis, concatenation) methods have been proposed as combinatorial algorithms, and validated by experiments [G. F., C. Giagulli, C. Laudanna, V. Manca, LNCS 3384, ’05. G. F., V. Manca, C. Giagulli, C. Laudanna, LNCS 3892, ’06. V. Manca, G.F., (2008).Mathematical Biosciences, 211(2), ’08 Epub ’07].

XPCR as an innovative procedure

Novel XPCR-based recombination (extraction, mutagenesis, concatenation) methods have been proposed as combinatorial algorithms, and validated by experiments [G. F., C. Giagulli, C. Laudanna, V. Manca, LNCS 3384, ’05. G. F., V. Manca, C. Giagulli, C. Laudanna, LNCS 3892, ’06. V. Manca, G.F., (2008).Mathematical Biosciences, 211(2), ’08 Epub ’07].

XPCR-based procedure to generate large DNA libraries has been shown to be pretty convenient with respect other methods in literature.

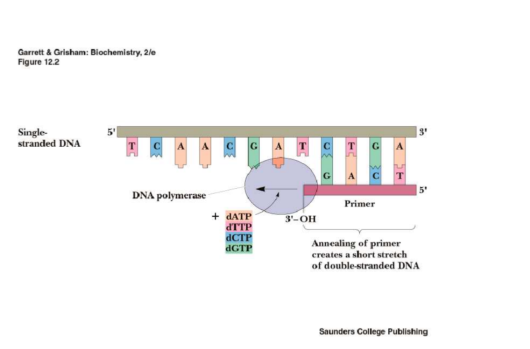
XPCR as an innovative procedure

Novel XPCR-based recombination (extraction, mutagenesis, concatenation) methods have been proposed as combinatorial algorithms, and validated by experiments [G. F., C. Giagulli, C. Laudanna, V. Manca, LNCS 3384, ’05. G. F., V. Manca, C. Giagulli, C. Laudanna, LNCS 3892, ’06. V. Manca, G.F., (2008).Mathematical Biosciences, 211(2), ’08 Epub ’07].

XPCR-based procedure to generate large DNA libraries has been shown to be pretty convenient with respect other methods in literature.Special strings, called *recombination witnesses*, were discovered, such that their presence in the pool guarantees that the whole library is there [G. F., Ph.D Thesis, CS Dept, Verona, ’06].

Polymerase Extension

Operations are performed over a **pool P** of DNA sequences: a multiset of strings over the alphabet $\Sigma = \{A, T, G, C\}$.



Garrett & Grisham: Biochemistry, 2/e
Figure 12.2

Saunders College Publishing

PCR$(\alpha, \overline{\beta})$(**P**) indicates a standard PCR with primers α and $\overline{\beta}$ applied on the pool P; it amplifies strings having α as a prefix

XPCR Procedure

Initial pool: heterogeneous pool of DNA double strands, all the same length *n*, prefix α, suffix β.

XPCR Procedure

Initial pool: heterogeneous pool of DNA double strands, all the same length *n*, prefix α, suffix β.

Given a string γ, $XPCR_\gamma$ recombines all the strings that contain γ as a substring, by implementing the “null context splicing rule”:

$$\underline{\alpha\phi\gamma}\psi\beta,\ \alpha\delta\underline{\gamma\eta\beta} \quad \xrightarrow{r_\gamma} \quad \underline{\alpha\phi\gamma\eta\beta},\ \alpha\delta\gamma\psi\beta,\ \alpha\phi\gamma\psi\beta,\ \alpha\delta\gamma\eta\beta$$

with $\alpha, \phi, \gamma, \psi, \delta, \eta, \beta$ strings on the alphabet.

$XPCR_\gamma$ Procedure Steps (1/2)

Input Pool P of α-prefixed and β-suffixed strings having length n

- **split** P into P_1 and P_2 (same approximate size)

$XPCR_\gamma$ Procedure Steps (1/2)

Input Pool P of α-prefixed and β-suffixed strings having length n

- **split** P into P_1 and P_2 (same approximate size)
- **apply** $PCR(\alpha, \overline{\gamma})$ to P_1 **and** $PCR(\gamma, \overline{\beta})$ to P_2

$XPCR_\gamma$ Procedure Steps (1/2)

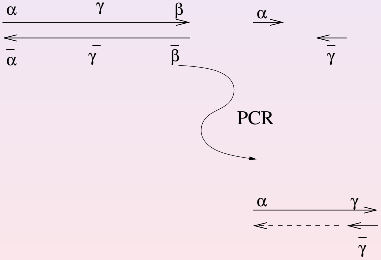
Input Pool P of α-prefixed and β-suffixed strings having length n

- **split** P into P_1 and P_2 (same approximate size)
- **apply** $PCR(\alpha, \overline{\gamma})$ to P_1 **and** $PCR(\gamma, \overline{\beta})$ to P_2

$XPCR_\gamma$ Procedure Steps (1/2)

Input Pool P of α-prefixed and β-suffixed strings having length *n*

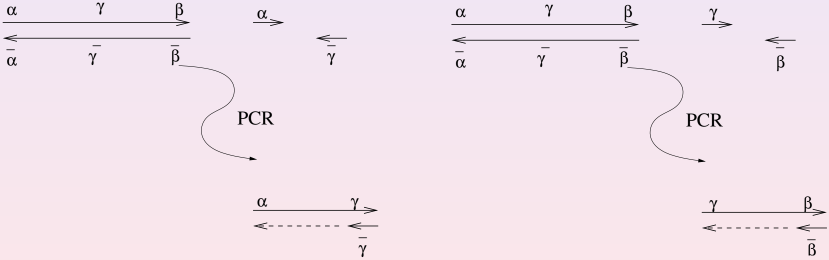
- **split** P into P_1 and P_2 (same approximate size)
- **apply** $PCR(\alpha, \overline{\gamma})$ to P_1 **and** $PCR(\gamma, \overline{\beta})$ to P_2



$XPCR_\gamma$ Procedure Steps (1/2)

Input Pool P of α-prefixed and β-suffixed strings having length n

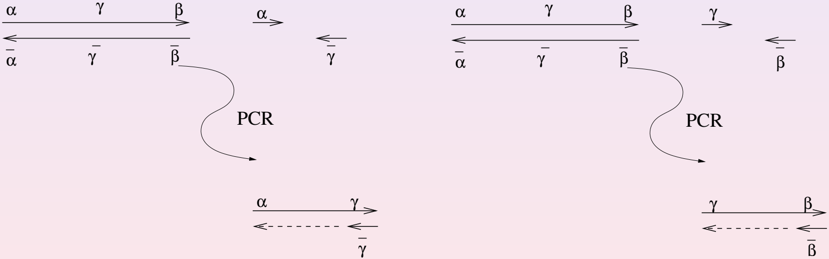
- **split** P into P_1 and P_2 (same approximate size)
- **apply** $PCR(\alpha, \overline{\gamma})$ to P_1 **and** $PCR(\gamma, \overline{\beta})$ to P_2



$XPCR_\gamma$ Procedure Steps (1/2)

Input Pool P of α-prefixed and β-suffixed strings having length *n*

- **split** P into P_1 and P_2 (same approximate size)
- **apply** $PCR(\alpha, \overline{\gamma})$ to P_1 **and** $PCR(\gamma, \overline{\beta})$ to P_2

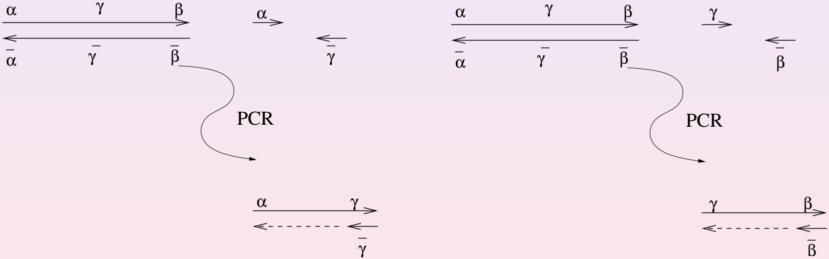


- perform **electrophoresis** on P_1 **and** on P_2 to eliminate the sequences of the initial length

$XPCR_\gamma$ Procedure Steps (1/2)

Input Pool P of α-prefixed and β-suffixed strings having length *n*

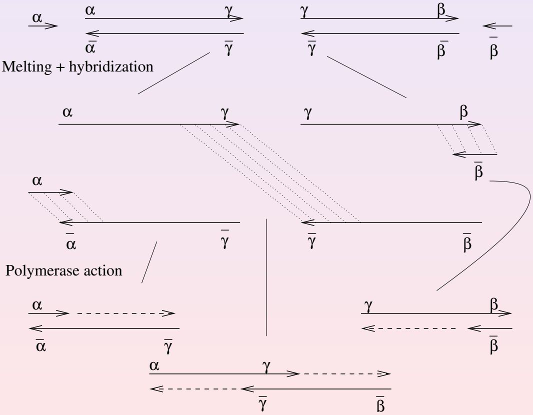
- **split** *P* into P_1 and P_2 (same approximate size)
- **apply** $PCR(\alpha, \overline{\gamma})$ to P_1 **and** $PCR(\gamma, \overline{\beta})$ to P_2



- perform **electrophoresis** on P_1 **and** on P_2 to eliminate the sequences of the initial length
- **mix** the two pools so obtained in a pool *P*

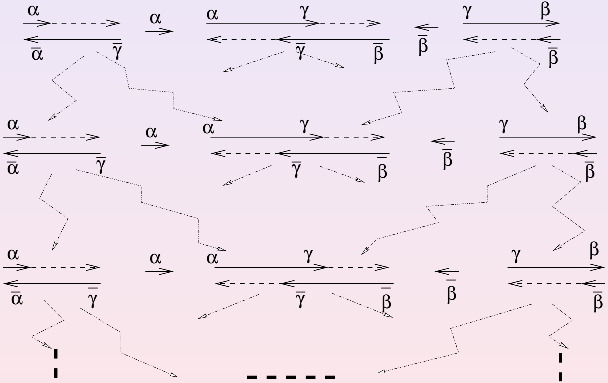
$XPCR_\gamma$ Procedure Steps (2/2)

- **apply** $PCR(\alpha, \overline{\beta})$ to P



Output The pool P resulting from the previous step.

XPCR Process



Long strands in the middle are seeds of exponential amplifications.

Generation of DNA Libraries

Goal: generating the combinatorial library of n binary numbers

$$\{\alpha_1 \cdots \alpha_n \mid \alpha_i \in \{X_i, Y_i\},\ i = 1, \ldots, n\}.$$

Generation of DNA Libraries

Goal: generating the combinatorial library of n binary numbers

$$\{\alpha_1 \cdots \alpha_n \mid \alpha_i \in \{X_i, Y_i\},\ i = 1, \ldots, n\}.$$

It has biological, technological, and algorithmic relevance.

Generation of DNA Libraries

Goal: generating the combinatorial library of n binary numbers

$$\{\alpha_1 \cdots \alpha_n \mid \alpha_i \in \{X_i, Y_i\},\ i = 1, \dots, n\}.$$

It has biological, technological, and algorithmic relevance.

A method to self-assembly multiple genes in all possible orders on operon structure was proposed in [Y.Nakagawa, K.Yugi, K.Tsuge, M.Itaya, J. Yanagawa, Y. Sakakibara, *Operon structure optimization by random self-assembly*, DNA14].

Generation of DNA Libraries

Goal: generating the combinatorial library of n binary numbers

$$\{\alpha_1 \cdots \alpha_n \mid \alpha_i \in \{X_i, Y_i\},\ i = 1, \ldots, n\}.$$

It has biological, technological, and algorithmic relevance.

A method to self-assembly multiple genes in all possible orders on operon structure was proposed in [Y.Nakagawa, K.Yugi, K.Tsuge, M.Itaya, J. Yanagawa, Y. Sakakibara, *Operon structure optimization by random self-assembly*, DNA14].The optimal order and orientation to reconstitute a set of genes are not trivial and depend on several factors, where "optimal" means the efficiency of biosynthesis induced by transfered genes in metabolic pathways.

Quaternary Recombination Algorithm

This method starts from l_1, l_2, l_3 l_4 (extended by prefix α and suffix β), and generates the whole library, in linear time and by using essentially polymerase extension.

Quaternary Recombination Algorithm

This method starts from l_1, l_2, l_3 l_4 (extended by prefix α and suffix β), and generates the whole library, in linear time and by using essentially polymerase extension.

1. *Positive*: $l_1 = X_1X_2X_3X_4X_5X_6$,

Quaternary Recombination Algorithm

This method starts from I_1, I_2, I_3 I_4 (extended by prefix α and suffix β), and generates the whole library, in linear time and by using essentially polymerase extension.

1. *Positive*: $I_1 = X_1X_2X_3X_4X_5X_6$,
2. *Negative*: $I_2 = Y_1Y_2Y_3Y_4Y_5Y_6$,

Quaternary Recombination Algorithm

This method starts from I_1, I_2, I_3 I_4 (extended by prefix α and suffix β), and generates the whole library, in linear time and by using essentially polymerase extension.

1. *Positive*: $I_1 = X_1X_2X_3X_4X_5X_6$,
2. *Negative*: $I_2 = Y_1Y_2Y_3Y_4Y_5Y_6$,
3. *Positive-Negative*: $I_3 = X_1Y_2X_3Y_4X_5Y_6$,

Quaternary Recombination Algorithm

This method starts from I_1, I_2, I_3 I_4 (extended by prefix α and suffix β), and generates the whole library, in linear time and by using essentially polymerase extension.

1. *Positive*: $I_1 = X_1X_2X_3X_4X_5X_6$,
2. *Negative*: $I_2 = Y_1Y_2Y_3Y_4Y_5Y_6$,
3. *Positive-Negative*: $I_3 = X_1Y_2X_3Y_4X_5Y_6$,
4. *Negative-Positive*: $I_4 = Y_1X_2Y_3X_4Y_5X_6$.

Recombination Algorithm

- **Let** P_1 and P_2 be two copies of the pool

$$\{\alpha l_1\beta,\ \alpha l_2\beta,\ \alpha l_3\beta,\ \alpha l_4\beta\}$$

Example: $l_1,\ l_4 \xrightarrow{r_{X_2}} X_1X_2Y_3X_4Y_5X_6,\ Y_1X_2X_3X_4X_5X_6,$
$l_2,\ X_1X_2Y_3X_4Y_5X_6 \xrightarrow{r_{Y_5}} Y_1Y_2Y_3Y_4Y_5X_6,\ \mathbf{X_1X_2Y_3X_4Y_5Y_6}.$

Recombination Algorithm

- **Let** P_1 and P_2 be two copies of the pool

$$\{\alpha l_1\beta,\ \alpha l_2\beta,\ \alpha l_3\beta,\ \alpha l_4\beta\}$$

- **for** $i = 2, 3, 4, 5$ **do**

Example: $l_1,\ l_4 \xrightarrow{r_{X_2}} X_1X_2Y_3X_4Y_5X_6,\ Y_1X_2X_3X_4X_5X_6,$
$l_2,\ X_1X_2Y_3X_4Y_5X_6 \xrightarrow{r_{Y_5}} Y_1Y_2Y_3Y_4Y_5X_6,\ \mathbf{X_1X_2Y_3X_4Y_5Y_6}.$

Recombination Algorithm

- **Let** P_1 and P_2 be two copies of the pool

$$\{\alpha l_1\beta,\ \alpha l_2\beta,\ \alpha l_3\beta,\ \alpha l_4\beta\}$$

- **for** $i = 2, 3, 4, 5$ **do**
  - **perform** $XPCR_{X_i}$ on P_1 and $XPCR_{Y_i}$ on P_2

Example: $l_1,\ l_4 \xrightarrow{r_{X_2}} X_1X_2Y_3X_4Y_5X_6,\ Y_1X_2X_3X_4X_5X_6,$
$l_2,\ X_1X_2Y_3X_4Y_5X_6 \xrightarrow{r_{Y_5}} Y_1Y_2Y_3Y_4Y_5X_6,\ \mathbf{X_1X_2Y_3X_4Y_5Y_6}.$

Recombination Algorithm

- **Let** P_1 and P_2 be two copies of the pool

$$\{\alpha l_1\beta,\ \alpha l_2\beta,\ \alpha l_3\beta,\ \alpha l_4\beta\}$$

- **for** $i = 2, 3, 4, 5$ **do**
    - **perform** $XPCR_{X_i}$ on P_1 and $XPCR_{Y_i}$ on P_2
    - **mix** the two pools obtained in the previous step in a pool $P := P_1 \cup P_2$, then **split** P randomly in two new pools P_1 and P_2 (with the same approximate size)

Example: $l_1,\ l_4 \xrightarrow{r_{X_2}} X_1X_2Y_3X_4Y_5X_6,\ Y_1X_2X_3X_4X_5X_6,$
$l_2,\ X_1X_2Y_3X_4Y_5X_6 \xrightarrow{r_{Y_5}} Y_1Y_2Y_3Y_4Y_5X_6,\ \mathbf{X_1X_2Y_3X_4Y_5Y_6}.$

Algorithm Correctness

XPCR-Recombination proved to be good for generation speed, material waste and efficiency (error). It is cheap and scalable, and can be easily automatized. It requires only 4 initial strings.

Algorithm Correctness

XPCR-Recombination proved to be good for generation speed, material waste and efficiency (error). It is cheap and scalable, and can be easily automatized. It requires only 4 initial strings.

The correctness of XPCR recombination algorithm can be given by the following formulation:

Theorem

The n-dimensional library $\{\alpha_1 \cdots \alpha_n \mid \alpha_i \in \{X_i, Y_i\}, i = 1, \ldots, n\}$ *is the null context splicing language generated by the system* $\mathcal{N} = (\Sigma, A, R)$*, where* $\Sigma = \{A, T, C, G\}$*,* $A = \{I_1, I_2, I_3, I_4\}$*, and* $R = \{r_{X_2}, r_{Y_2}, \ldots, r_{X_{n-1}}, r_{Y_{n-1}}\}$.

Theory and Experimentation

As all the biotechniques, it should be equipped with a method to verify that the whole library is really present in the final pool, that is, whether all the splicing rules have worked as expected.

Theory and Experimentation

As all the biotechniques, it should be equipped with a method to verify that the whole library is really present in the final pool, that is, whether all the splicing rules have worked as expected.

Task. Assuming we have methods to check if a certain sequence is present in the pool, we want to minimize the number of sequences to check in order to verify the success of the experiment (i.e. that the whole library was generated).

Recombination Witnesses

It is proved that, for any dimension *n* of the library, two sequences exist such that their simultaneous presence in the pool guarantees that all the recombinations required by the XPCR recombination algorithm occurred.

Recombination Witnesses

It is proved that, for any dimension n of the library, two sequences exist such that their simultaneous presence in the pool guarantees that all the recombinations required by the XPCR recombination algorithm occurred.

They can be: $\{W_1, W_2\}$ or $\{T_1, T_2\}$, where ($n \equiv 0 \pmod 4$)

$$W_1 = X_1X_2\,Y_3Y_4\,X_5X_6\,Y_7\dots Y_n$$
$$W_2 = Y_1Y_2\,X_3X_4\,Y_5Y_6\,X_7\dots X_n$$

$$T_1 = X_1\,Y_2Y_3\,X_4X_5\,Y_6Y_7\dots X_n$$
$$T_2 = Y_1\,X_2X_3\,Y_4Y_5\,X_6X_7\dots Y_n$$

Conclusions

It is yet clear that DNA computing is not competitive with *in silico* computers to solve NP-complete problems. Encoding problems due to mismatches are overcame as well.

Conclusions

It is yet clear that DNA computing is not competitive with *in silico* computers to solve NP-complete problems. Encoding problems due to mismatches are overcame as well.

Current trends focus on investigations on self-assembly phenomenon (namely construction of state machines, “DNA doctor”), as well as on improvements of bio-techniques (Whiplash PCR), and search of new procedures (XPCR, DNA concentration comparator).

Conclusions

It is yet clear that DNA computing is not competitive with *in silico* computers to solve NP-complete problems. Encoding problems due to mismatches are overcame as well.

Current trends focus on investigations on self-assembly phenomenon (namely construction of state machines, “DNA doctor”), as well as on improvements of bio-techniques (Whiplash PCR), and search of new procedures (XPCR, DNA concentration comparator).

New strategies of DNA computation (autonomous, autocatalytic, isothermal) were also addressed in DNA14 meeting.

A Few Open Problems

- How to efficiently quantify the different types of molecules of a pool? (A first attempt: F. Tanaka, T. Tsuda, M. Hagiya, *Towards DNA Comparator: the Machine That Compares DNA Concentrations*, DNA14)

A Few Open Problems

- How to efficiently quantify the different types of molecules of a pool? (A first attempt: F. Tanaka, T. Tsuda, M. Hagiya, *Towards DNA Comparator: the Machine That Compares DNA Concentrations*, DNA14)

- Is it experimentally possible to generate large libraries by logarithmic-time running XPCR recombination algorithm? (A Cooperint Project has been funded by Verona University to work in collaboration with Dept of Biological Sciences, University of Binghamton, NY)

A Few Open Problems

- How to efficiently quantify the different types of molecules of a pool? (A first attempt: F. Tanaka, T. Tsuda, M. Hagiya, *Towards DNA Comparator: the Machine That Compares DNA Concentrations*, DNA14)
- Is it experimentally possible to generate large libraries by logarithmic-time running XPCR recombination algorithm? (A Cooperint Project has been funded by Verona University to work in collaboration with Dept of Biological Sciences, University of Binghamton, NY)
- Any application of efficient recombination procedures to immunological contexts, to study rearrangements of antibody genes?

A Few Open Problems

- How to efficiently quantify the different types of molecules of a pool? (A first attempt: F. Tanaka, T. Tsuda, M. Hagiya, *Towards DNA Comparator: the Machine That Compares DNA Concentrations*, DNA14)
- Is it experimentally possible to generate large libraries by logarithmic-time running XPCR recombination algorithm? (A Cooperint Project has been funded by Verona University to work in collaboration with Dept of Biological Sciences, University of Binghamton, NY)
- Any application of efficient recombination procedures to immunological contexts, to study rearrangements of antibody genes?

A Few Open Problems

- How to efficiently quantify the different types of molecules of a pool? (A first attempt: F. Tanaka, T. Tsuda, M. Hagiya, *Towards DNA Comparator: the Machine That Compares DNA Concentrations*, DNA14)
- Is it experimentally possible to generate large libraries by logarithmic-time running XPCR recombination algorithm? (A Cooperint Project has been funded by Verona University to work in collaboration with Dept of Biological Sciences, University of Binghamton, NY)
- Any application of efficient recombination procedures to immunological contexts, to study rearrangements of antibody genes? Or to synthetic biology, for the optimization problem of operon structure?

The research group to which I belong

Prof. Vincenzo Manca, three Ph.D. students, a post-doc fellow, and a couple of master students.

We have collaborations with:

- Computational Biomedical Center, Department of General Pathology, Scientific and Biotechnological Department, University of Verona (*Analysis of new DNA protocols, Theory and Applications of Metabolic P systems*).

The research group to which I belong

Prof. Vincenzo Manca, three Ph.D. students, a post-doc fellow, and a couple of master students.

We have collaborations with:

- Computational Biomedical Center, Department of General Pathology, Scientific and Biotechnological Department, University of Verona (*Analysis of new DNA protocols, Theory and Applications of Metabolic P systems*).
- Department of Mathematics and Medical School, USF (*Self-assembly, Modeling by membrane computing*).

The research group to which I belong

Prof. Vincenzo Manca, three Ph.D. students, a post-doc fellow, and a couple of master students.

We have collaborations with:

- Computational Biomedical Center, Department of General Pathology, Scientific and Biotechnological Department, University of Verona (*Analysis of new DNA protocols, Theory and Applications of Metabolic P systems*).
- Department of Mathematics and Medical School, USF (*Self-assembly, Modeling by membrane computing*).
- LIACS, Leiden (*Self-assembly models*).

An announcement for interested researchers

For young researchers interested to work on *biological information modelling, encoding, and processing*:

there is a *postdoctoral or junior research position* (for one year, extensible for up to five years) available (from Sept 2008) in Biocomputing Laboratory, at University of Western Ontario

http://www.csd.uwo.ca/ lila/postdoc.html

It is funded by the Canada Research Chair in Biocomputing awarded to Professor Lila Kari.

Thank you!