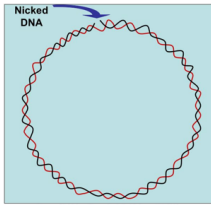
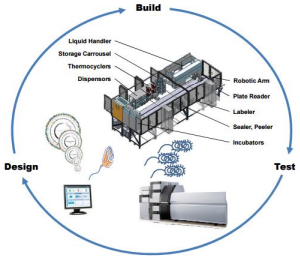
Technical summary: University of Illinois, Texas, Minnesota

Period of Performance: [January 1st, 2018 – Janury 1st, 2021]

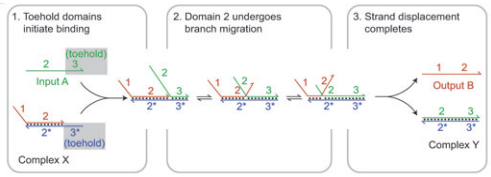
Native (naturally occurring) DNA



Positional encoding via DNA nicking



Process Automation



Nick-displacement based computing

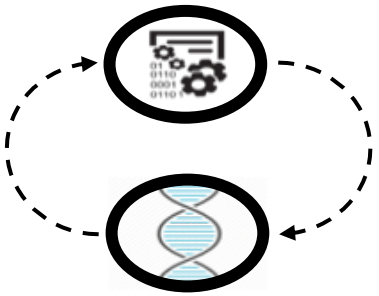
- High cost of synthesis calls for new information encoding methods. Propose to encode information by nicking native DNA at positions specified by integer-encoded user information. Goal is to significantly reduce system implementation cost.
- Nicked DNA may be used in a new computational paradigm, termed nick displacement, to implement counters and comparison units. Goal is to integrate storage and computing paradigms via nick information and perform in-memory computations.

Key innovations:

1) Novel encoding via positional information and native DNA strand nicking;
2) Novel error-control mechanisms for nick readouts;
3) Simple random access strategies;
4) Novel parallel, selective nicking strategies;
5) Register-based data formating/organization;
6) Novel nick displacement computing paradigm;
7) Implementation of counters and comparison units for in-memory computations.
8) Potential implementation of in-memory computation following Minsky's Register Machine.
9) Seamless integration of storage and computing.

Existing methods are exclusively built around synthetic DNA encoding information in the symbol values. They do not involve computational features.

Specialized reading and computing strategies via nicking of native DNA. Sequencing and decoding for positional information.



Specialized encoding for digital data into naturally occurring DNA. New computational paradigms based on positional encoding.

The Team

Alvaro Hernandez, co-PI, University of Illinois

Expertise: DNA sequencing.

Olgica Milenkovic, PI, University of Illinois

Expertise: Error-control and constraint coding for modern storage devices.

Marc Riedel, co-PI, University of Minnesota

Expertise: DNA computing, in memory computing.

David Soloveichik, co-PI, University of Texas, Austin

Expertise: DNA computing, molecular programming.

Huimin Zhao, co-PI, University of Illinois

Expertise: Chemistry, synthetic biology.

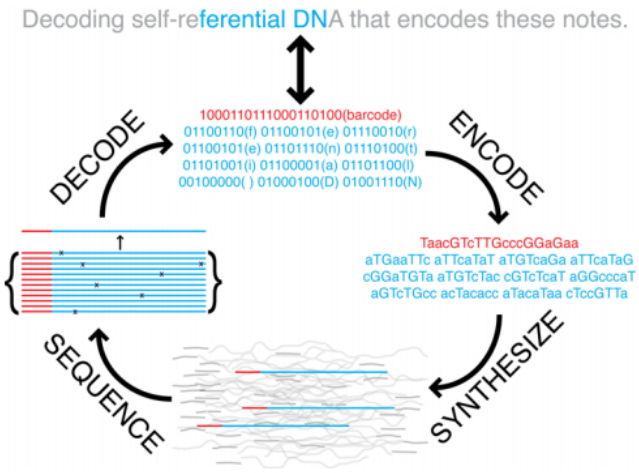
Start of the Art

Definition of SOA?

Conceptual simplicity?

Church et al, 2012
Goldman et al, 2013

Largest density?

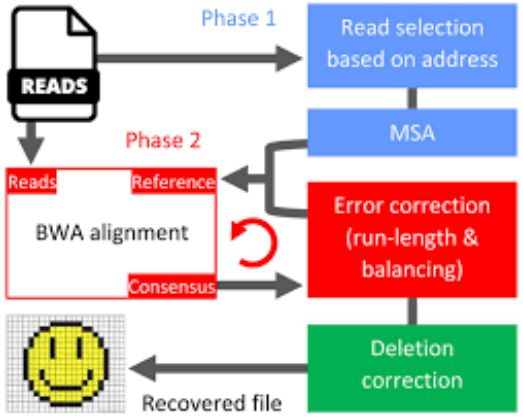


Fountains: Erlich et al, 2017; Yazdi et al, 2017 (UIUC team)
Microsoft Research, 2016

>215PB/g, file sizes >2MB

Access and rewriting, functional capabilities?

First PCR-based random access architecture
First portable architecture
Yazdi et al, 2017 (UIUC team)



Smallest cost per base? ???

Start of the Art

DNA data blocklengths: 100-1000bps

Costs?

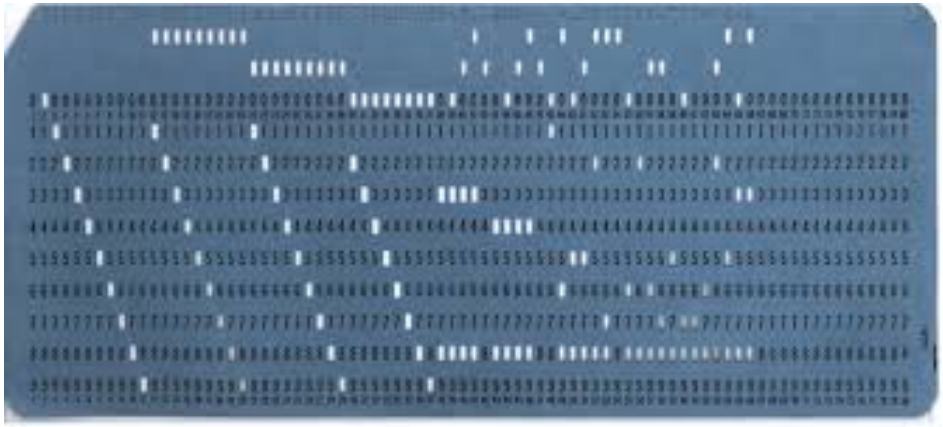
Company	Length (bp)	#Oligos	Concentration	Price (USD)
CustomArray	10-79	92,918	1 fmol	$4,000
Twist Bioscience	1-120	600,000	0.25 fmol	$59,160
ThermoFisher	5-40	24	25 nmol	$5.2
Operon	20	96	4 nmol	$115
IDT	20	1	25 nmol	$7.4

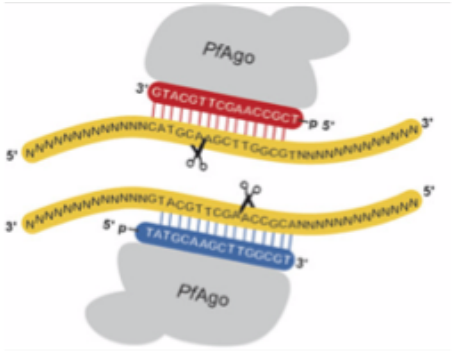
In house synthesizers (BioLytics, BioXP): ~20bps

Beyond the Start of the Art

Native DNA based storage: Bringing down the cost and in memory computing!

Our proposed methodology: "Teaching a new dog an old trick."





50nmols, oligolength ~20: >200,000 "punches" (bits), cost < 10 cents

Beyond the State of the Art

Why Does this Work?

1. Need to synthesize only one primer in sufficiently large concentration to introduce many nicks at same position in different registers.

2. Each nick encodes log(3) bits, expected number of nicks 200,000.

3. Short oligos (primers) also used needed for computing, may hence be used for nicking (storage), addressing (random access) and computing (nick displacement).

4. Why not store information in short ~20bps length oligos directly?

    - Problem: How does one store the primer orders?
    - mlog(m) bits needed to store order of m items.
    - Think of native strand serving as template for the order.

Risks

Challenges and risks:

1. Parallelization and scalability?
   - Recording time depends on number of edits that may be performed in parallel.
   - Multiple “data registers” needed – system maintenance challenges.

2. Readout architectures?
   - Immunoprecipitation-based and nanopore nick detection methods may be of insufficient accuracy.

3. Integration with displacement computing paradigms?
   - Precision of displacement strategies and computational delays.

Status and Outlook

Current Status/Timeline:

No adjustments made. Additional testing to be performed using nanopore sequencers.

Looking Forward/Next 6 months:

Goal 1: Data encoding, positional conversion, error-correction.
Goal 2: Native DNA sample preparation, replication, register organization.
Goal 2: Parallel register and positional DNA nicking experiments.
Goal 3. Mathematical modeling of DNA nicking-based computational paradigms.
Goal 4. Initial development of readout systems.

• Phase I Goals: Practical implementation of small scale storage architecture, theoretical development of in memory computational component.

Risks

Current and future strategies for risk mitigation:

1. Parallelization and scalability?

    - Replace and/or combine *PfAgo* editing with CRISPR technologies.
    - Combine nicking with other writing techniques such as chimeric synthesis.

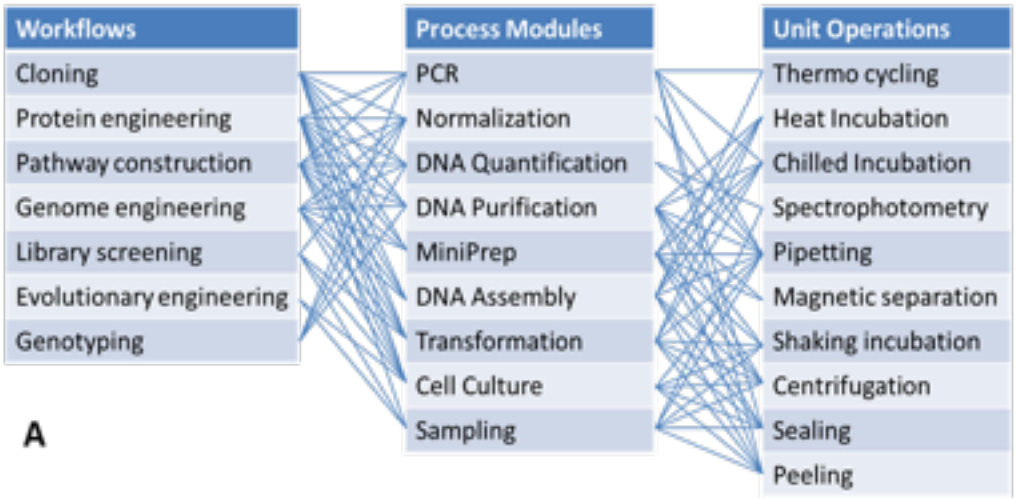
2. Readout architectures?

- Molecular simulations to predict technical obstacles in nanopore sequencing methods.

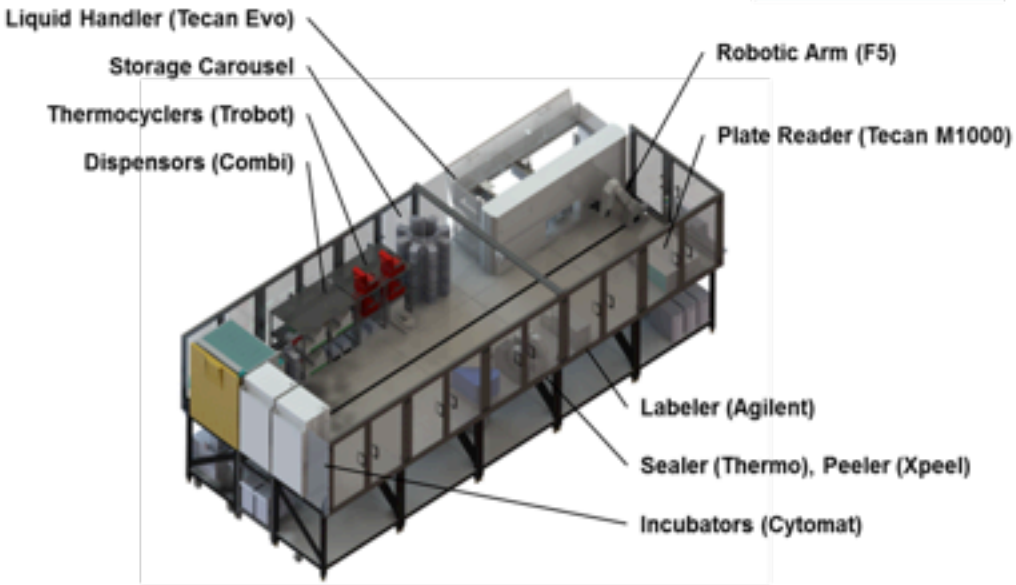
3. Integration with computing paradigms?

- Difficult to judge. In needed, restricted to counting and comparison.

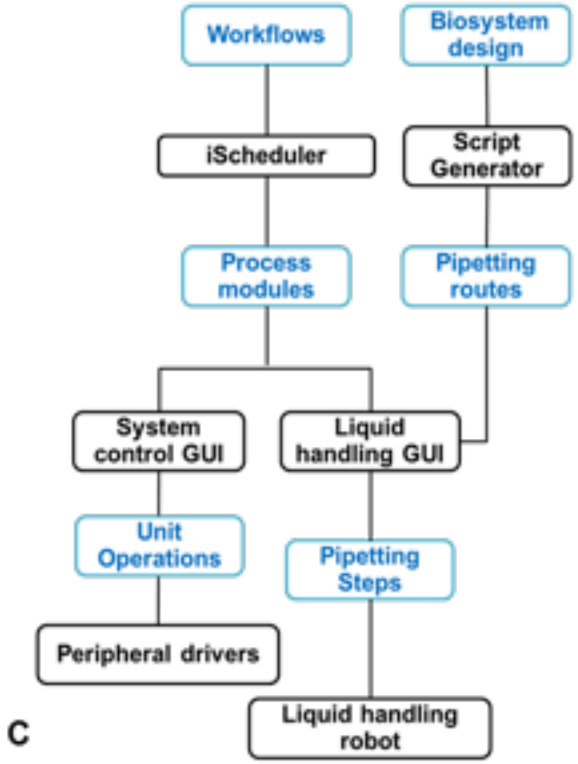
Lab Facilities



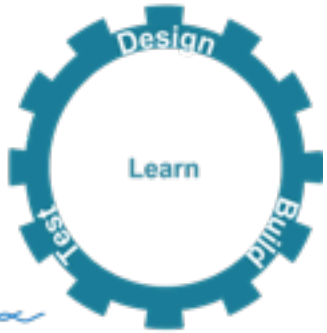
A



B



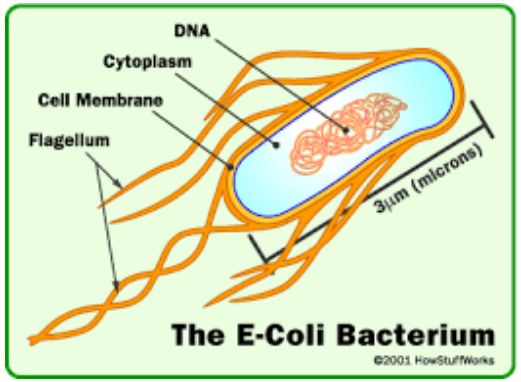
C

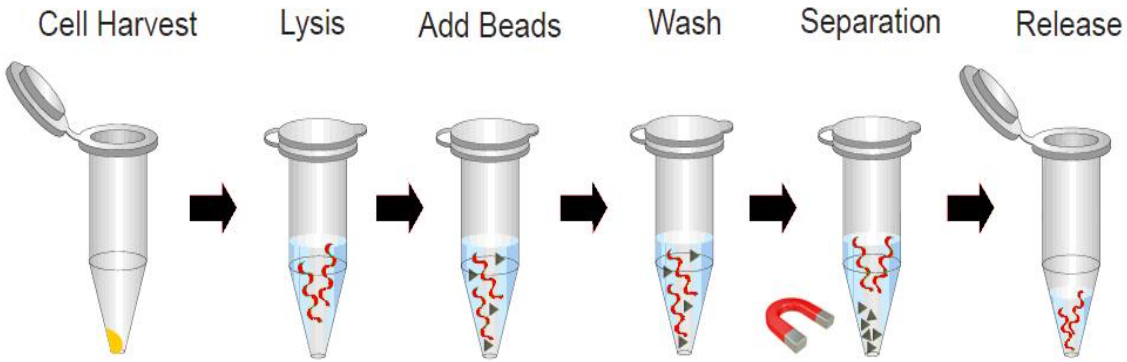


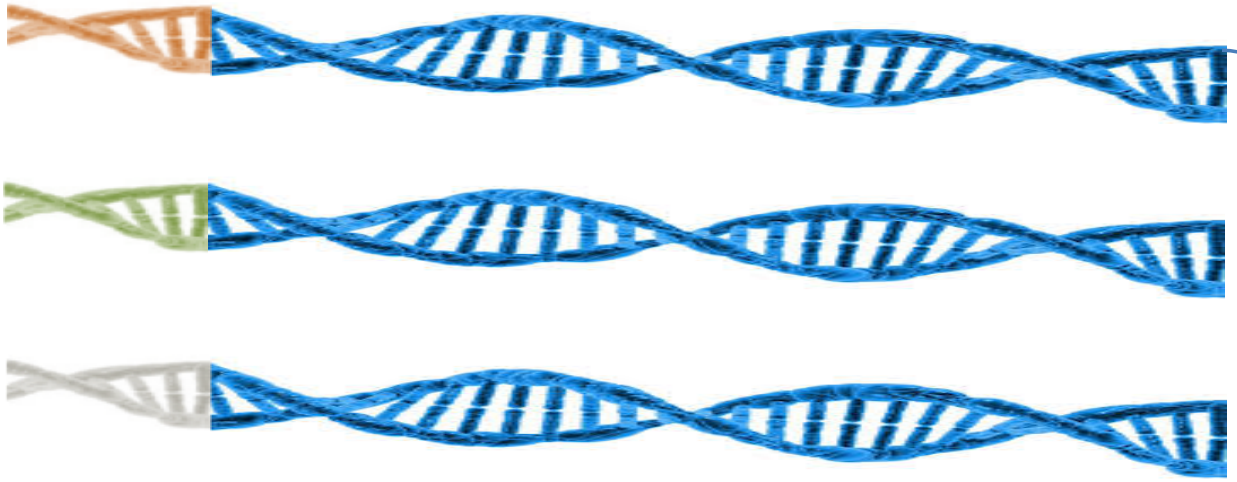
http://youtu.be/Hwb735qZ-IQ

The Project: Native DNA-Based Data Storage

The storage medium: E. coli K-12







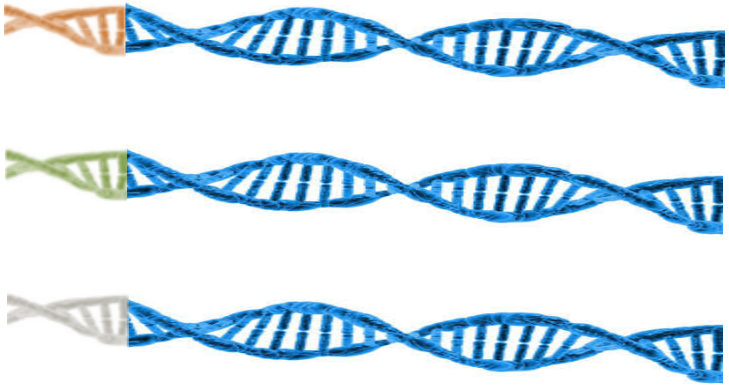
Addressable registers

Yazdi et al, 2015, 2017

The Project: Native DNA-Based Data Storage

Why E. coli K-12?

Length of each substring	# unique substrings	$\log_4$(# unique substrings)
1	4	1.0000
2	16	2.0000
3	64	3.0000
4	256	4.0000
5	1,024	5.0000
6	4,096	6.0000
7	16,383	7.0000
8	65,360	7.9981
9	256,527	8.9844
10	898,115	9.8883
11	2,196,861	10.5335
12	3,478,960	10.8651
13	4,170,362	10.9959



Genome: U00096.3 Escherichia coli str. K-12 substr. MG1655, complete genome

Length in bp: 4,641,652

Organism name	Length	M	m
Streptomyces coelicolor A3(2) (high GC Gram+)	9,054,847	5	2,621
Myxococcus xanthus DK 1622 (d-proteobacteria)	9,139,763	6	2,817
Nostoc punctiforme PCC 73102 (cyanobacteria)	9,059,191	5	5,569

The Project: Native DNA-Based Data Storage

Data Encoding

Ternary coding: Do not nick (0), nick sense (1), nick antisense (3)

Nicking window: 10-20 bps

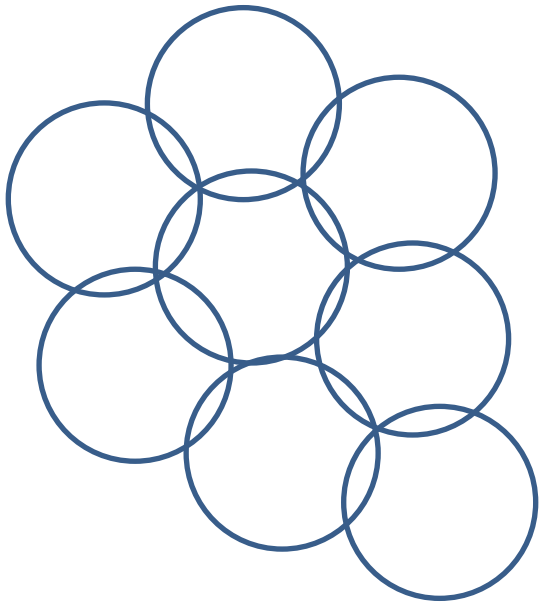
Set encoding: Convert binary string into positions of nicks. Allows for efficient error correction (Lex and Grey order).

123 (0)	00000	00000	124 (1)	00001	00001	125 (2)	00010	00011	126 (3)	00011	00010
127 (4)	00100	00110	134 (5)	00101	00100	135 (6)	00110	00101	136 (7)	00111	00111
137 (8)	01000	01111	145 (9)	01001	01110	146 (10)	01010	01100	147 (11)	01011	01101
156 (12)	01100	01001	157 (13)	01101	01011	167 (14)	01110	01010	234 (15)	01111	01000
235 (16)	10000	11000	236 (17)	10001	11001	237 (18)	10010	11011	245 (19)	10011	11010
246 (20)	10100	11110	247 (21)	10101	11100	256 (22)	10110	11101	257 (23)	10111	11111
267 (24)	11000	10111	345 (25)	11001	10110	346 (26)	11010	10100	347 (27)	11011	10101
356 (28)	11100	10001	357 (29)	11101	10011	367 (30)	11110	10010	456 (31)	11111	10000

The Project: Native DNA-Based Data Storage

Data Encoding

Set encoding: Convert binary string into sets of positions of nicks.
Error-control through controlled set intersection.



L-intersecting family of sets
(Babai & Frankl, Erdos-Ko-Rado type theorems)

L={1,2,..,s}, s << n

Bad pairs:
{1,3,5,12,46,…}
{1,3,6,13,46,...} …}

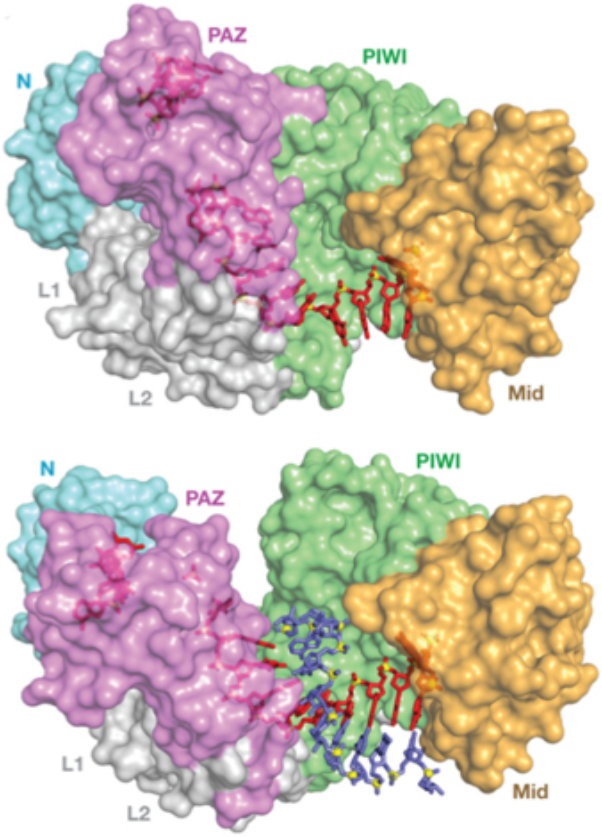
Good pairs:
{1,3,5,12,46,…}
{2,7,9,17,33,…}

Can handle: deletions, insertion, transposition errors (Gabry et al, 2017)

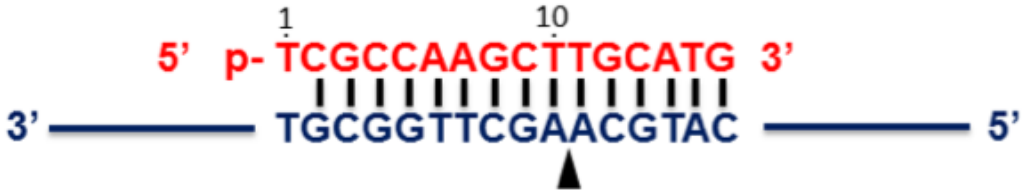
The Project: Native DNA-Based Data Storage

DNA Nicking

*Pf*Ago: An Ago protein from hyperthermophilic archaea *Pyrococcus furiosus*, is uses DNA as guide and cleave ssDNA target at temperatures up to 100 °C.



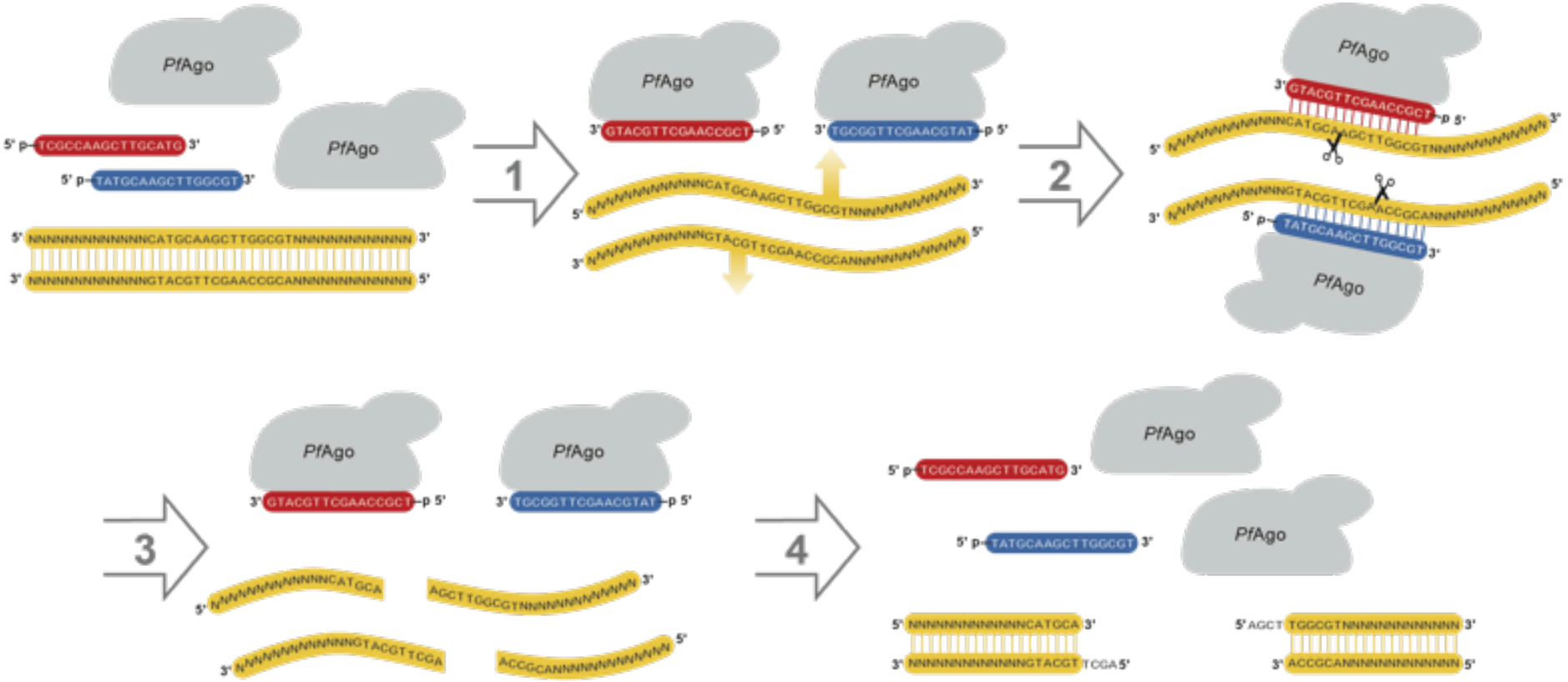
Cleavage of ssDNA target using ssDNA guide for *Pf*Ago



Swarts et al. *Nature* (2014)

The Project: Native DNA-Based Data Storage

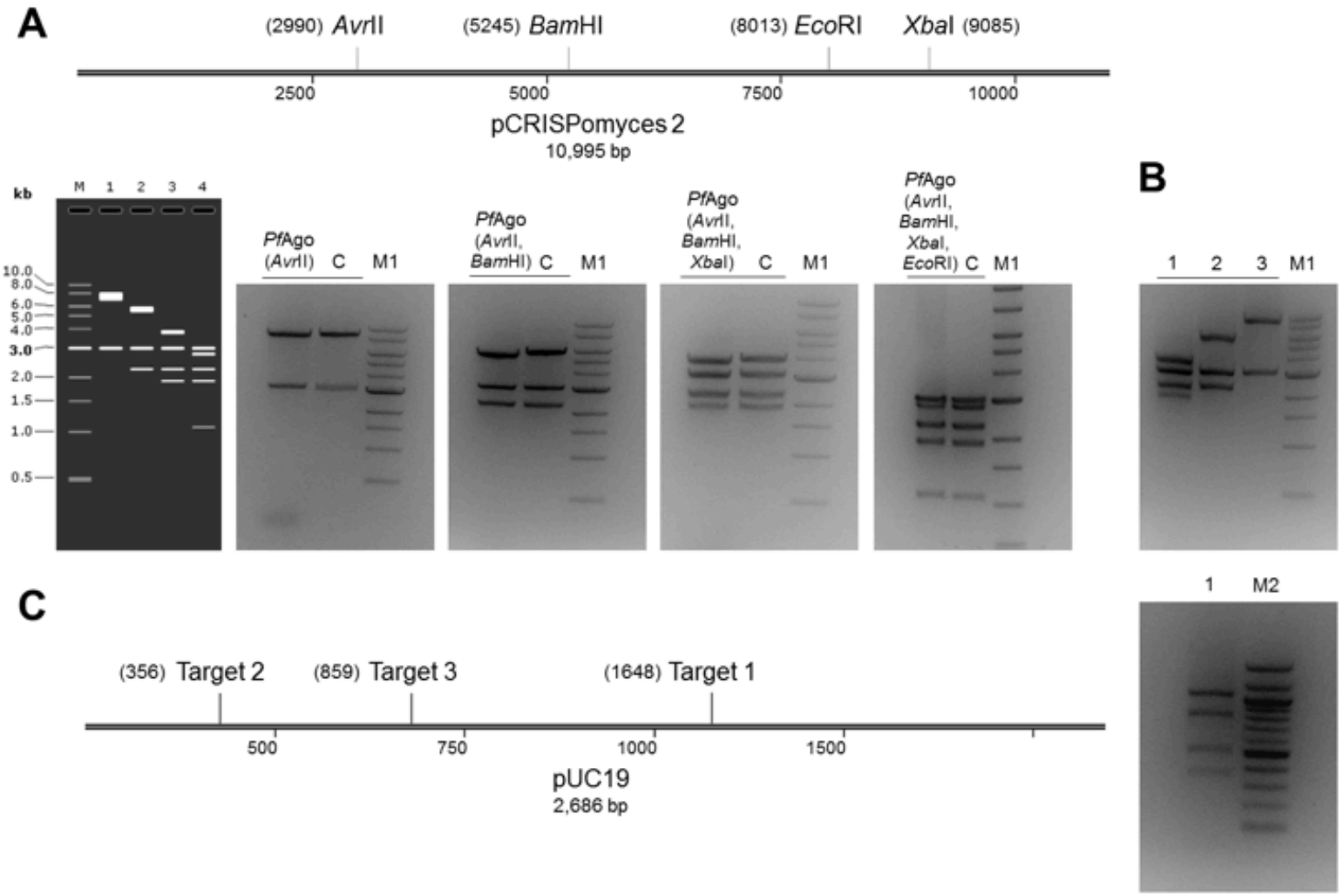
Programmable DNA-guided artificial restriction enzymes



Enghiad & Zhao. *ACS Synthetic Biology* 2017

The Project: Native DNA-Based Data Storage

Programmable DNA-guided artificial restriction enzymes



The Project: Native DNA-Based Data Storage

Programmable DNA-guided AREs

- Programmability and multiplexing
  - Can create extremely large number of AREs with virtually any sequence specificity and defined sticky ends of varying length.
  - The system can be used in parallel (multiplexed).
  - AREs nickases need only one guide.

- Accessibility
  - Low production cost.

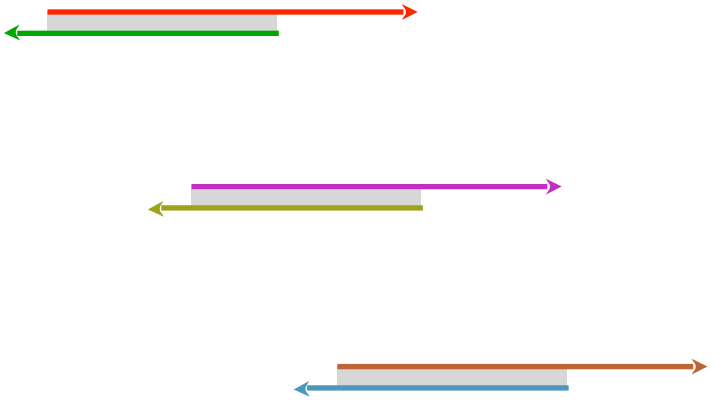
- High specificity and activity
  - Significantly better than the CRISPR-Cas9 for in vitro applications.

"Revolutionizing Biotechnology with Artificial Restriction Enzymes." *Genetic Engineering and Biotechnology News*, Feb. 10, 2017.

The Project: Native DNA-Based Data Storage

DNA-Computing: Strand Displacement

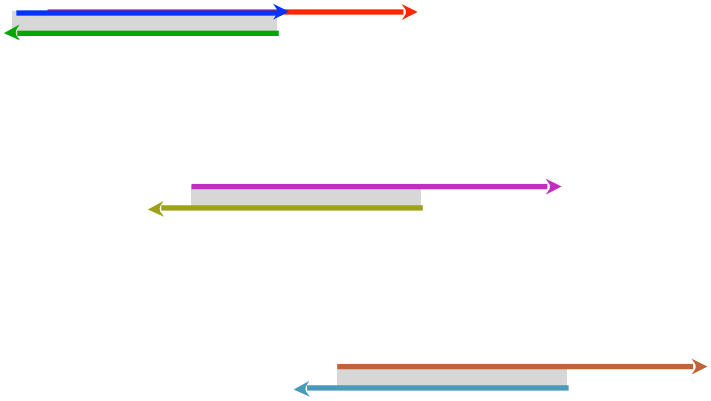
Input



Chen, Dalchau, Srinivas, Phillips, Cardelli, Soloveichik, Seelig, *Nature Nanotechnology* 2013

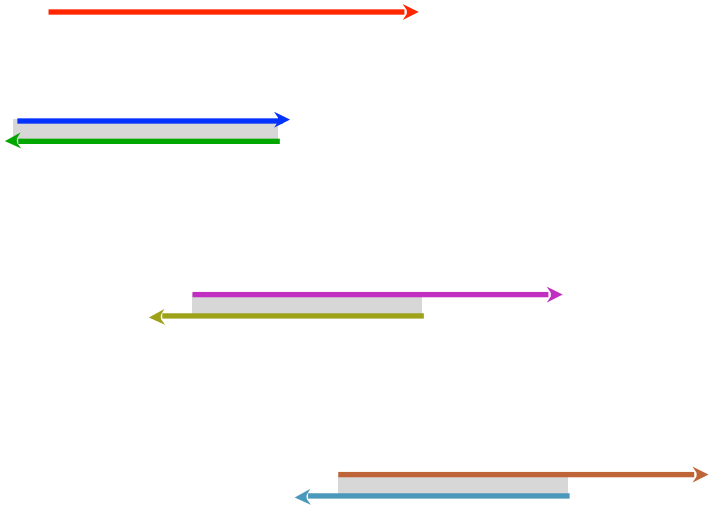
The Project: Native DNA-Based Data Storage

DNA-Computing: Strand Displacement



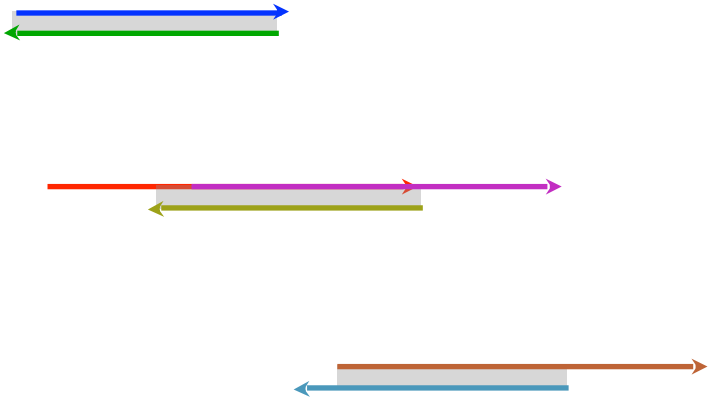
The Project: Native DNA-Based Data Storage

DNA-Computing: Strand Displacement



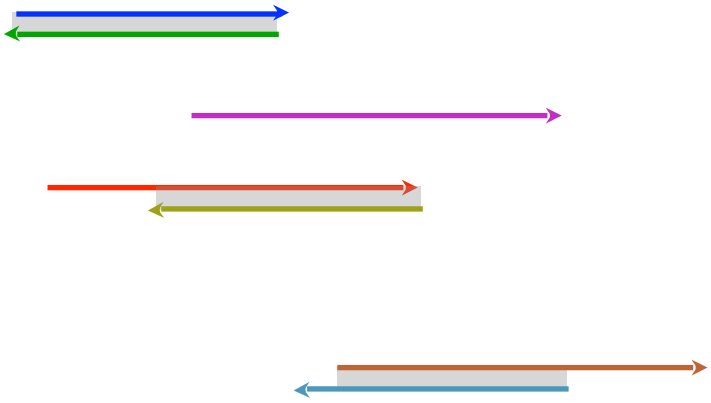
The Project: Native DNA-Based Data Storage

DNA-Computing: Strand Displacement



The Project: Native DNA-Based Data Storage

DNA-Computing: Strand Displacement

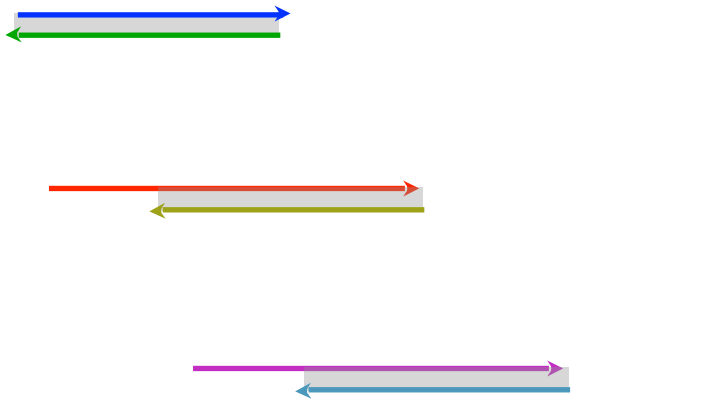


The Project: Native DNA-Based Data Storage

DNA-Computing: Strand Displacement

The Project: Native DNA-Based Data Storage

DNA-Computing: Strand Displacement

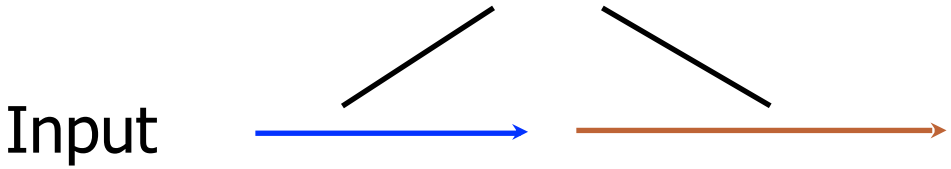


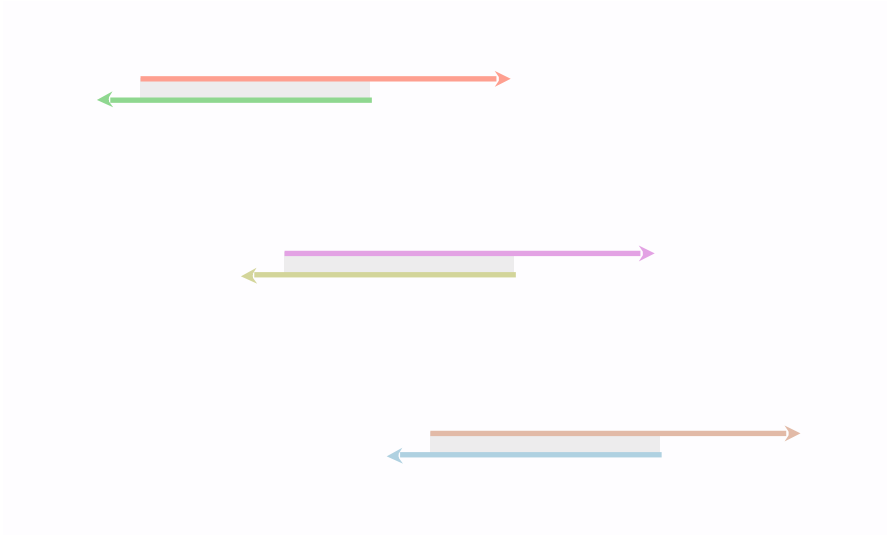
Output

The Project: Native DNA-Based Data Storage

DNA-Computing: Strand Displacement

Notice: No sequence overlap!

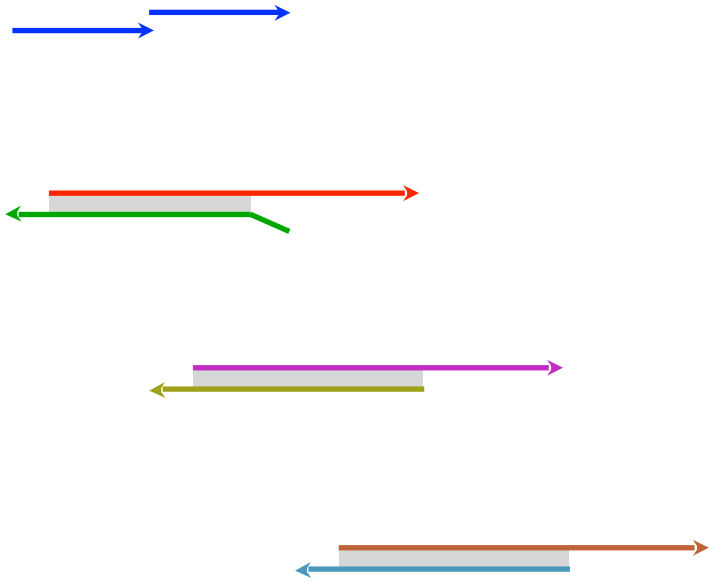




The Project: Native DNA-Based Data Storage

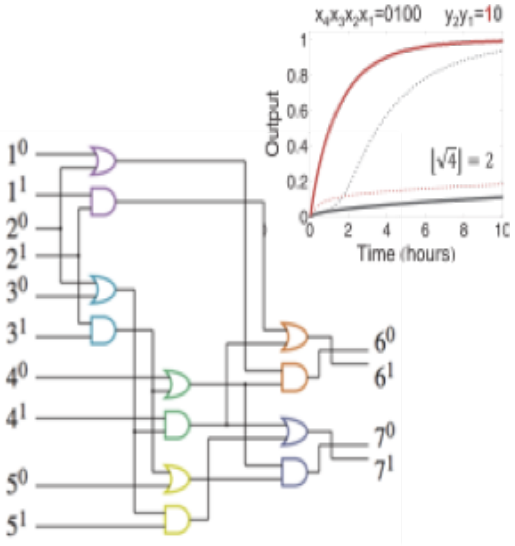
DNA-Computing: Strand Displacement

Input1 + Input2 → Output

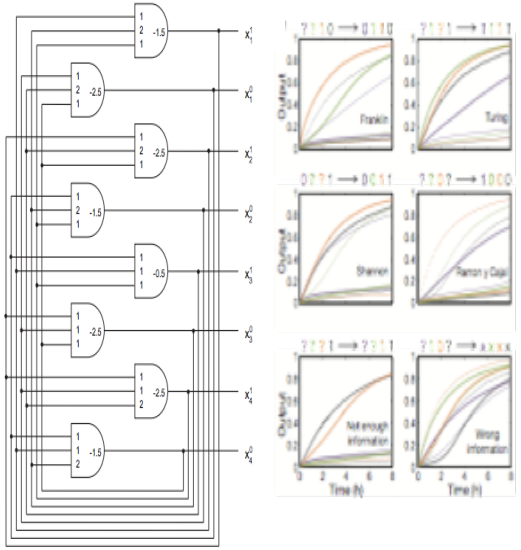


The Project: Native DNA-Based Data Storage

- Logic circuits [Seelig et al, *Science* 2006, Qian et al, *Science* 2011]
- Neural networks [Qian et al, *Nature* 2011]
- Distributed algorithms [Chen et al, *Nature Nanotechnology* 2013]
- Dynamical systems (oscillator) [Srinivas et al, Accepted to *Science*]

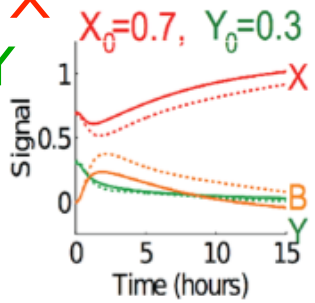


Logic circuit



Neural Network

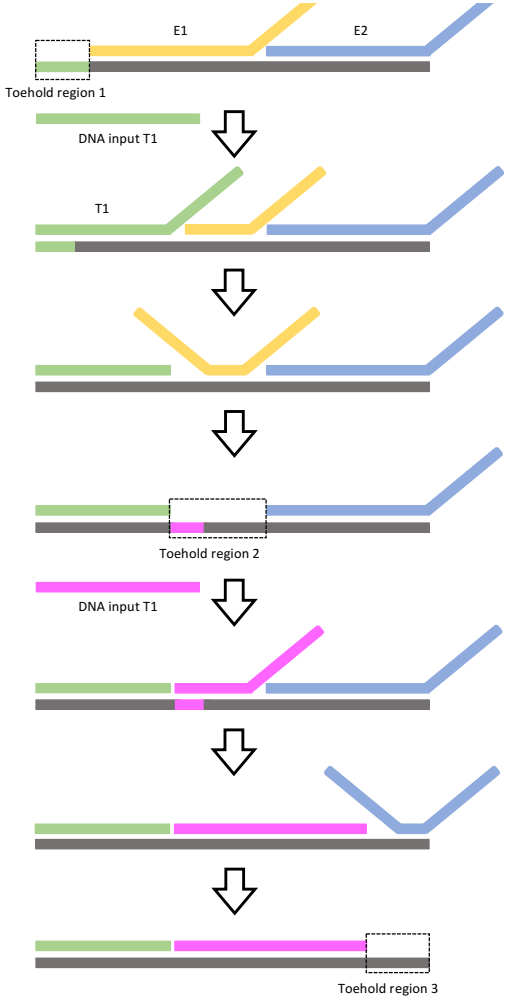
$$X + Y \rightarrow X + B$$
$$X + Y \rightarrow B + Y$$
$$X + B \rightarrow X + X$$
$$Y + B \rightarrow Y + Y$$

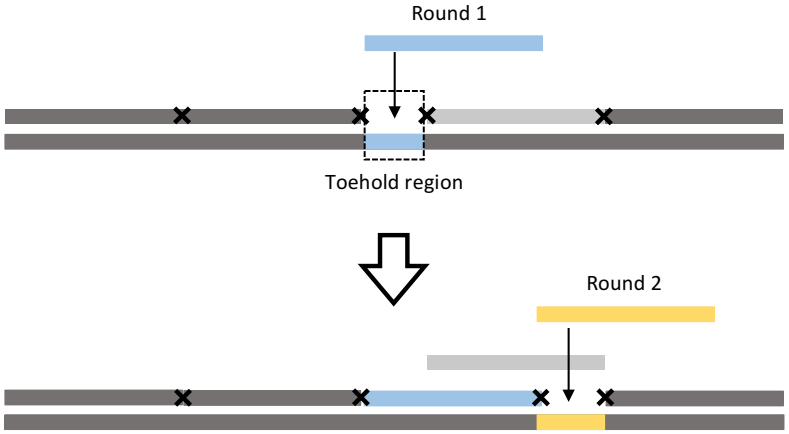


Distributed algorithm

The Project: Native DNA-Based Data Storage

DNA-Computing: Strand Displacement and Nick Displacement



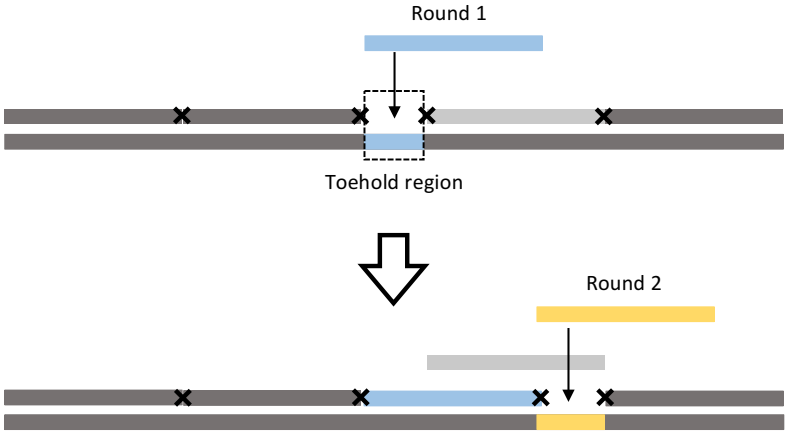


First strand displacement and new nick displacement paradigm built around native DNA.

Primers may be used to displace nicks by one position (for toehold length ~20).

The Project: Native DNA-Based Data Storage

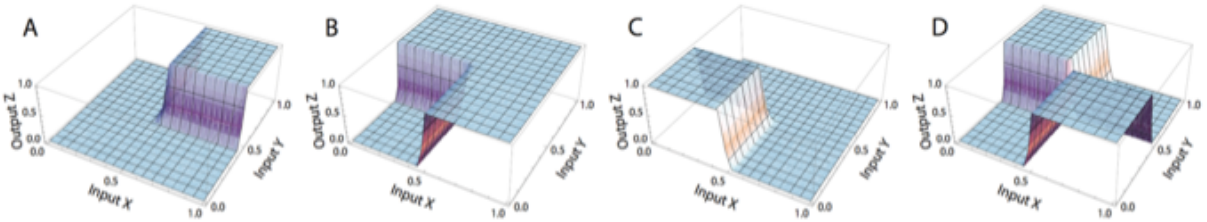
DNA-Computing: Strand Displacement and Nick Displacement



Counting: Adding and Subtracting

Displacing nicks left-right until carry over is encountered

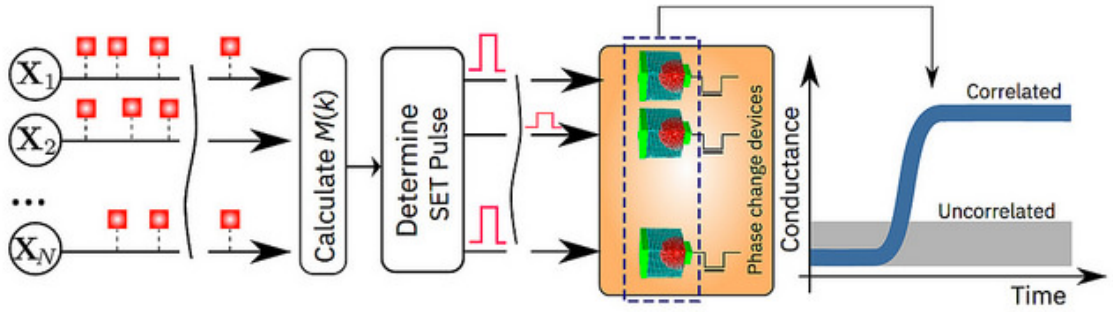
Comparison: Values in different registers



Example: Logical Computation with DNA Strand Displacement. Figure shows simulations of DNA-based implementations of standard logic gates: (A) AND, (B) OR, (C) NOR, (D) XOR. (Salehi, Riedel, Parhi, 2015.)

The Project: Native DNA-Based Data Storage

DNA-Computing: In Memory Computation



A schematic illustration of an in memory computing algorithm. Neural Computations are performed directly on data stored in memory. Credit: IBM Research

Computing with high-dimensional vectors (P. Kanerva, Cognitive Computation'09)

Advantages of in memory computations:

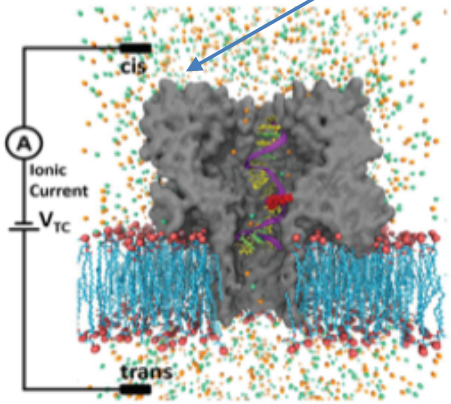
- General and scalable model of computing.
- Memory-centric with highly parallel operations.
- Extremely robust against most failure mechanisms and noise.

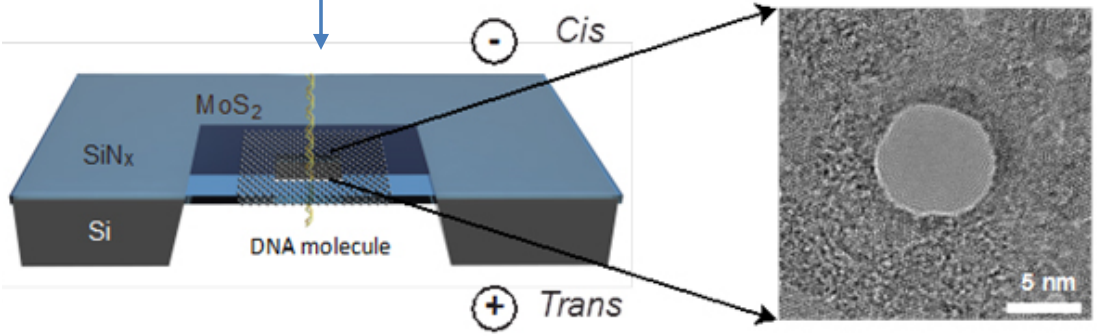
The Project: Native DNA-Based Data Storage

Readout Architectures

NGS

1. Nick detection via immunoprecipitation+NGS
2. Nick detection using protein and solid state nanopores (In collaboration with Radenovic lab, EPFL, Leburton lab UIUC)

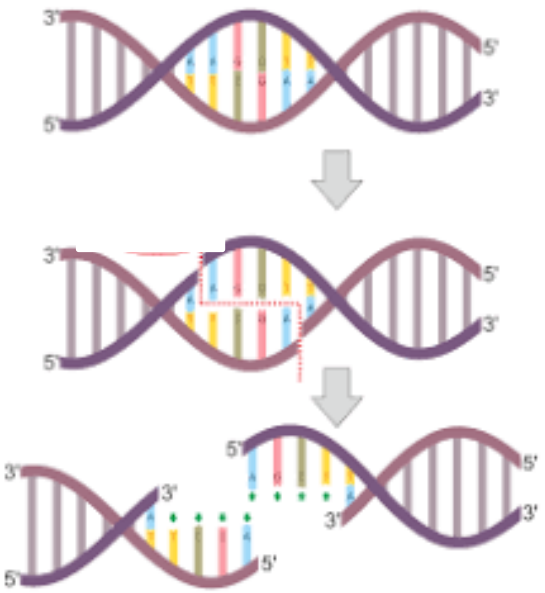


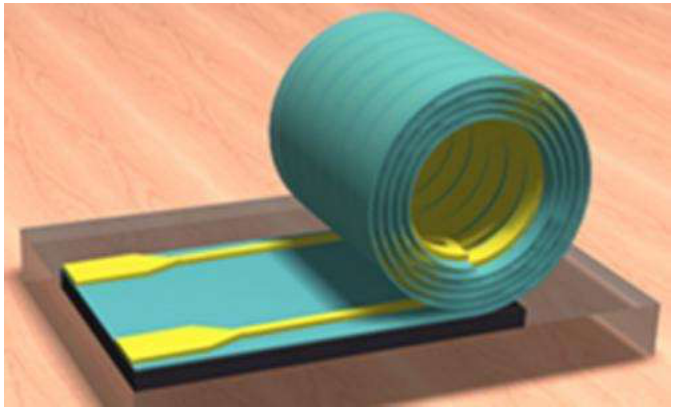


The Project: Native DNA-Based Data Storage

Nick-Encoding for Watermarking

Nicking → DSB+PCR





Random Access: via strand displacement, addressing or microtubes
(In collaboration with X. Li Lab, ECE UIUC)
Not funded by this project.

Closed Session

ROI

Number of people employed, by category:
3 graduate students, 2 postdoctoral fellows.
Number of newly trained scientists in this area: None
Resource Status: None
Number of PhD theses initiated based on this work: None yet
Discoveries utilized on other efforts: NSF CCF, CIA, SRI UIUC
Patents filed: 3
Papers published: For proposed topic material >5 journal papers
Presentations given: [please indicate the number of invited talks separately]
Technology licenses: None
New companies formed: On related subject - HelixVault
Venture capital: None
Follow-on funding: None