

50 years later


<10 iterations of Moore’s Law left!
The race is on for molecular scale integrated circuits.
Building The Smallest Things

- How do we build structures that are by definition smaller than your tools?
- Basic answer: you can’t. Structures (and tools) should build themselves!
- By *programmed self-assembly*.
Molecular IKEA

• Nature can self-assemble. Can we?

• “Dear IKEA, please send me a chest of drawers that assembles itself.”

• We need a magical material where the pieces are pre-programmed to fit into each other.

• At the molecular scale many such materials exist; let’s pick one...
DNA

GC Base Pair
Guanine–Cytosine

TA Base Pair
Thymine–Adenine

Interactive DNA Tutorial
(http://www.biosciences.bham.ac.uk/labs/minchin/tutorials/dna.html)

Sequence of Base Pairs (GACT alphabet)
Robust, and *Long*

- **DNA in each human cell:**
  - 3 billion base pairs
  - *2 meters long*, 2nm thick
  - folded into a 6µm ball
  - 750 MegaBytes

- **A huge amount for a cell**
  - Every time a cell replicates it has to copy *2 meters of DNA* reliably.
  - To get a feeling for the scale disparity, compute:

- **DNA in human body**
  - 10 trillion cells
  - 133 Astronomical Units long
  - 7.5 OctaBytes

- **DNA in human population**
  - 20 million light years long
Zipping Along

- DNA can support structural and computational complexity.

DNA replication in *real time*

- In Humans: 50 nucleotides/second
- Whole genome in a few hours (with parallel processing)
- In Bacteria: 1000 nucleotides/second (higher error rate)

DNA transcription in *real time*

RNA polymerase II: 15–30 base/second

Drew Berry
http://www.wehi.edu.au/wehi-tv
Nanoscale Engineering

- **Sensing**
  - Reacting to forces
  - Binding to molecules

- **Actuating**
  - Releasing molecules
  - Producing forces

- **Constructing**
  - Chassis
  - Growth

- **Computing**
  - Signal Processing
  - Decision Making

Nucleic Acids can do all this. And interface to biology.
Hybridization

- Strands with **opposite orientation and complementary base pairs** stick to each other (Watson–Crick duality).
- This is all we are going to use
  - We are not going to exploit DNA replication, transcription, translation, restriction and ligation enzymes, etc., which enable other classes of tricks.
Hybridization Tricks
Constructing
Crosslinking
Crosslinking
Crosslinking
Crosslinking
Crosslinking

In nature, crosslinking is deadly (blocks DNA replication).

In engineering, crosslinking is the key to using DNA as a construction material.
2D DNA Lattices

Chengde Mao
Purdue University, USA

N-point Stars
3D DNA Structures

Ned Seeman
NYU

3D Crystal

Andrew Tuberfield
Oxford

Tetrahedron
CADnano

S.M. Douglas, H. Dietz, T. Liedl, B. Högberg, F. Graf and W. M. Shih
Self-assembly of DNA into nanoscale three-dimensional shapes, Nature (2009)
Folding long (7000bp) naturally occurring (viral) ssDNA

By lots of short ‘staple’ strands that constrain it

Black: long viral strand
Color: short staple strands

Paul Rothemund’s “Disc with three holes” (2006)

This means we can already self-assemble meso-scale structures.
What we are really making are tiny DNA circuit boards that will be used to assemble other components.

Greg Wallraff, IBM
Sensing
Aptamers

- Artificially evolved DNA molecules that stick to anything you like (highly selectively).
Pathogen Spotlights

- DNA aptamer binds to:
  - A) a pathogen
  - B) a molecule our immune system already hates and immediately removes (eats) along with anything attached to it

- **Result: instant immunity**
  - Mice poisoned with Anthrax plus aptamer (100% survival)
  - Mice poinsoned with Anthrax (not so good)

Kary Mullis (incidentally, also Nobel prize for inventing the Polymerase Chain Reaction)
Computing

Basic Steps
Compositionality

- Sensors and Actuators at the 'edge' of the system
  - They can use disparate kinds of inputs (sensors) and outputs (actuators)

- The 'kernel' of the system computes
  - Must use uniform inputs and outputs

- Compositionality in the kernel
  - Supporting 'arbitrary' computing complexity
  - The output of each computing components must be the same kind of 'signal' as the input
    - If the inputs are voltages, the outputs must be voltages
    - If the inputs are DNA, the outputs must be DNA

- Central design question
  - What should our signals (not components!) be?
  - Then design components that manipulate those signals.
Rules of the Game

• Short complementary segments hybridize **reversibly**

• Long complementary segments hybridize **irreversibly**
DNA Strand Displacement

- **Short strand** (toehold): reversible binding
- **Long strand** (body): irreversible binding
Failed Strand Displacement

- What if the input does not match the gate?
Failed Strand Displacement
Failed Strand Displacement
Failed Strand Displacement
Failed Strand Displacement
Failed Strand Displacement
Failed Strand Displacement

• **Hence** an incorrect binding will undo
  o That’s why toeholds must bind reversibly

• **Matching depends on the long segment only**
  o Strand displacement succeeds iff the whole long segment matches
  o The address space is determined by the size of the long segment, which is unbounded (not by the size of the toehold)
  o The toehold is just a ‘cache’ of the address
Computing

Implementing “Arbitrary” Computing Functions
What does DNA Compute?

• Electronics has *electrons*
  o All electrons are the same: you can only count them
  o *Few* electrons = *False*; *lots* of electrons = *True*
  o But **Boolean Logic** is only a necessary evil to build symbolic computation

• DNA computing has *symbols* (DNA words)
  o DNA words are not all the same
  o **Symbolic computation on abstract signals** can be done *directly*
  o Signals are presented *concurrently* (in a soup)
  o No requirement to do Boolean Logic

• Then, what are our ‘*gates*’ (if not Boolean?)
  o **Theory of Concurrency**
  o **Process Algebra** as the “Boolean Algebra” of DNA Computing
Signals

• A signal is the representation of an abstract event
  o E.g. generated by a sensor
  o E.g. accepted by an effector
  o We are not limited to true/false

• 3–domain signals
  o $x_h$: hystory (ignore)
  o $x_t$: toehold (binding)
  o $x_b$: body (recognition)

• Signals (single stranded DNA) are prepared by (artificial) DNA synthesis
Gates

- Double-stranded structures with free toeholds

- Gates are prepared by self-assembly from single-stranded DNA that is synthesized
Fork Gate

- $x \rightarrow y + z$

- $x \rightarrow y + 0$ transform $x$ to $y$ (transducer)
- $x \rightarrow x + y$ linear production of $y$ (catalyst)
- $x \rightarrow x + x$ exponential production of $x$ (amplifier)
Fork Gate
Fork Gate
Fork Gate
Fork Gate
Fork Gate
Fork Gate
Fork Gate
Fork Gate
Fork Gate
Fork Gate
Fork Gate

This is Waste
Join Gate

\[ x + y \rightarrow z \]
Join Gate

This is the Join Gate structure
Join Gate
Join Gate
Join Gate
Join Gate
Join Gate
Join Gate
Join Gate
Join Gate
Join Gate
Join Gate
Join Gate
Join Gate
Join Gate
Join Gate

This is Waste
General n–Join/m–Fork Gate

Garbage collection

\[ x_{1h}, \ldots, x_{nh} \text{ generic} \]
\[ t, y_{1h}, \ldots, y_{nh} \text{ fresh} \]
\[ c_{2t}, c_{2b}, \ldots, c_{nt}, c_{nb} \text{ fresh} \]
Strand Algebra

\[ x_1 \mid .. \mid x_n \mid [x_1, \ldots, x_n].[y_1, \ldots, y_m] \rightarrow y_1 \mid .. \mid y_m \]

- Join + Fork + Populations = (Stochastic) Petri Nets
Gate Design Verification

• Active garbage
  o The active join residuals slow down the performance of following joins.
  o Add a garbage collector to remove the active residuals.

• Interference between gates
  o The join garbage collector interferes with the fork gate.
  o Modify the fork gate to remove the interference.

• What else could go wrong?
  o Endless possibilities.
  o Prove that the fork/join gate structures correctly implement fork/join in all larger circuits.
Actuating
Hybridization Chain Reaction

Stable mixture of two hairpins

Initiator

1. Chain reaction

3. Chain reaction

by hybridization

Robert M. Dirks and Niles A. Pierce

5
An autonomous polymerization motor powered by DNA hybridization

SUVR VENKATARAMAN, ROBERT M. DIRKS, PAUL W. K. ROTHEMUND, ERIK WINFREE, AND NILES A. PIERCE

Directional Actin Polymerization Associated with Spotted Fever Group Rickettsia Infection of Vero Cells

ROBERT A. VERNER, STANLEY F. BATES, MARLIE S. TEOCK, AND TED HOGARTH
Curing
A Doctor in Each Cell

Fig. 1 Medicine in 2050: “Doctor in a Cell”
Tools
So we can in principle work at this level.
Visual DSD
A Strand Displacement Simulator

Matthew Lakin, Simon Youssef, Andrew Phillips

http://lepton.research.microsoft.com/webdna/
A programming language for composable DNA circuits

Andrew Phillips* and Luca Cardelli

A. Syntax of DNA molecules $D$

Upper strand with sequence complementary to $S$

$S$

$<S>$

Molecule with segments $G_1, \ldots, G_k$

$G_1:G_2: \ldots :G_k$

Parallel molecules $D_1, \ldots, D_k$

$D_1 \mid D_2 \mid \ldots \mid D_k$

$D_1 \mid D_2 \mid \ldots \mid D_k$

Molecules $D$ with private domains $N_1, \ldots, N_k$

$(N_1, \ldots, N_k)$

new $(N_1, \ldots, N_k)$

B. Syntax of DNA segments $G$

Lower strand with toehold $N^2$

$N^2$

$N^2$

Double strand with sequence $S$ and overhangs $L$, $R$

$L \quad \quad S \quad \quad R$

$<L>[S]<R>$

C. Syntax of DNA sequences $S,L,R$

Sequence of domains $O_1, \ldots, O_k$

$O_1 \quad O_2 \quad \ldots \quad O_k$

$O_1 \quad O_2 \quad \ldots \quad O_k$
Dynamics

1. Toehold binding and unbinding

2. Strand displacement to the right

3. Strand displacement to the left

4. Branch migration
Initial Species
Reaction Graph
Simulation
Abstract Reactions
Detailed Reactions
Detailed Leak Reactions!
Just-in-Time Simulation
DNA Sequences

3^ --> TATCC
5^ --> GCTA
1 --> CCCTTTACATTACATAAACA
2 --> CCAAAAACAAAAACAAAAACAA
4 --> CCCTTTTCTAACTAAACAA
6 --> CCCTTATCATATCAATACAA
Final DNA Circuit
Next-Day Oligos!

SameDay® Oligo Service

Only €1.44 EUR / Base!

Current Time is 22:40 (GMT)

Specifications:

- Order online by 11:00 GMT
- SameDay® priority shipping for delivery by 10:30 GMT on second business day
- 2-OD minimum guarantee (sufficient for > 250 PCR reactions)
- 15-45 bases
- Shipped lyophilized in tubes
- Deprotected & desalted
- Unmodified

Place an Order Now
Place Order NOW!
DNA by Mail

Custom Oligonucleotide Synthesis

Innovation and Precision in Nucleic Acid Synthesis
It runs!

• (Nothing to do with us.)
DNA Compilation
Monolithic Compilers

Language Design #1
Boolean Networks
Language Implementation #1

Language Design #2
Petri Nets
Language Implementation #2

Language Design #3
... 
Language Implementation #3
Intermediate Languages

Front End

Intermediate Language

Back End

Boolean Networks

Petri Nets

Strand Algebra

The algebra of fork and join gates
Front Ends

Circuit Design

Intermediate Language

Boolean Networks

Petri Nets

Strand Algebra

…

Intermediate Language #2
Back Ends

- Intermediate Language
  - Gate Design
    - Structural Language
      - 4-domain Signals
      - 3-domain Signals
      - 2-domain Signals
  - Device Design
    - Strand Algebra
Compiling Abstract Machines
Boolean Networks

Boolean Networks to Strand Algebra

This encoding is *compositional*, and can encode *any* Boolean network:
- multi-stage networks can be assembled (*combinatorial logic*)
- network loops are allowed (*sequential logic*)
Petri Nets

Petri Nets to Strand Algebra

Transitions as Gates
Place markings as Signals

\[
\begin{align*}
\text{p}_1 & \quad \text{p}_2 \\
\text{p}_3 & \quad \text{p}_4
\end{align*}
\]

\[
([\text{p}_1,\text{p}_2].[\text{p}_3,\text{p}_4])^* \mid \text{p}_1 \mid \text{p}_1 \mid \text{p}_4
\]
Chemical Reaction Networks

Implementing an arbitrary finite chemical system in DNA with asymptotically correct kinetics
Soloveichick & al. DNA 15

Species become signals
Reactions become gates

\[ A + B \rightarrow C + D \\Rightarrow \ [A,B].[C,D] \]
Interacting Automata

This is a uniform population of identical automata, but heterogeneous populations of interacting automata can be similarly handled.

([A,B].[B,B])* | ([B,C].[C,C])* | ([C,A].[A,A])* | A | A | B | C
This is a uniform population of identical automata, but heterogeneous populations of interacting automata can be similarly handled.
Interacting Automata

This is a uniform population of identical automata, but heterogeneous populations of interacting automata can be similarly handled.
Interacting Automata

This is a uniform population of identical automata, but heterogeneous populations of interacting automata can be similarly handled.
Strand Algebra to DSD

\[ P ::= x : [x_1,..,x_n].[y_1,..,y_m] : 0 : P|P : P^* \quad n \geq 1, m \geq 0 \]

- \( \text{compile}(x) = \)
- \( \text{compile}([x_1,..,x_n].[y_1,..,y_m]) = \)
- \( \text{compile}(0) = \) empty solution
- \( \text{compile}(P | P') = \text{mix}(\text{compile}(P), \text{compile}(P')) \)
- \( \text{compile}(P^*) = \text{population}(\text{compile}(P)) \)
And finally...
Summary

• Abstract Machines to Strand Algebra
  o Or other intermediate language

• Strand Algebra to DSD
  o Or other structural language

• Simulation, analysis, etc.
  o Iterate a lot

• DSD to Sequences
  o E.g. NuPack, or pre-build strand libraries

• Sequences to DNA
  o Web order

• DNA experiments
  o Fairly basic wet lab

• Deployable Nanotech
Conclusions

• **Nucleic Acids**
  - Programmable matter

• **DNA Strand Displacement**
  - A computational mechanism at the molecular level

• **DNA as a Compilation Target for Abstract Machines**
  - Abstract Machines (Boolean Networks, Petri Nets, Interacting Automata)
  - Intermediate languages (Strand Algebra, Strand Displacement Language).
  - DNA sequence generation.

• **Tools**
  - Thermodynamic analysis.
  - Reaction graph generation.
  - Simulation.
  - Verification (not yet).
Acknowledgments

• Illustrations
  o John Reif, Duke
  o Ned Seeman, NYU
  o Erik Winfree, Caltech
  o Bernard Yurke, Boise State
  o Wikipedia
  o YouTube

• David Soloveichik