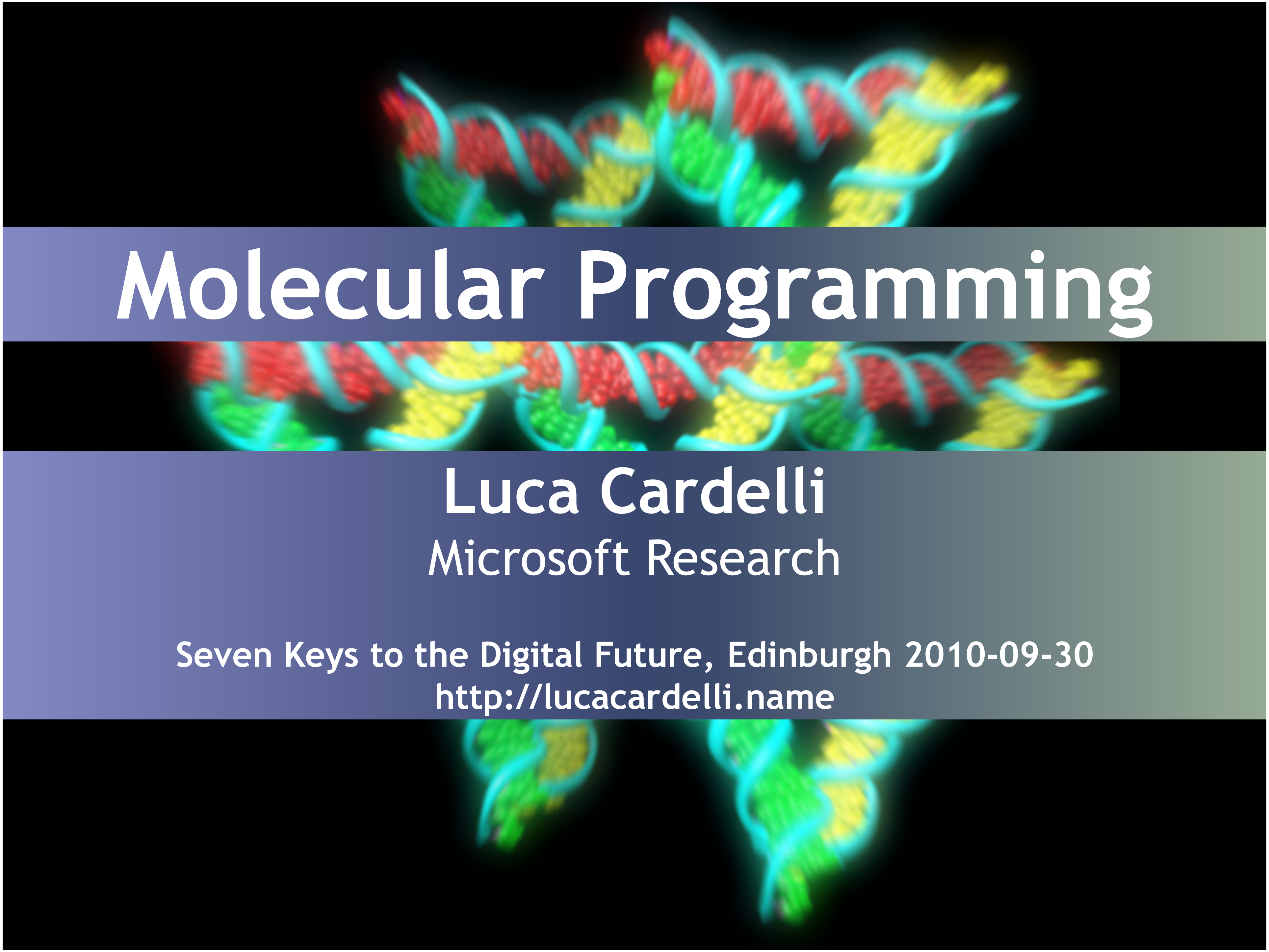


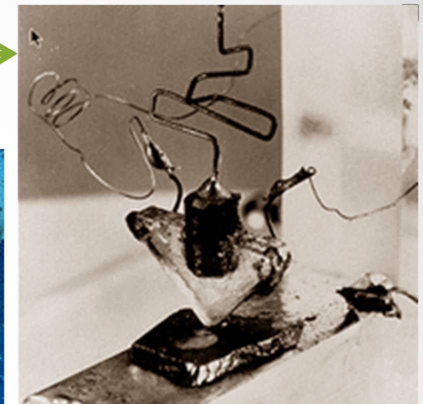
# Molecular Programming


## Luca Cardelli
Microsoft Research

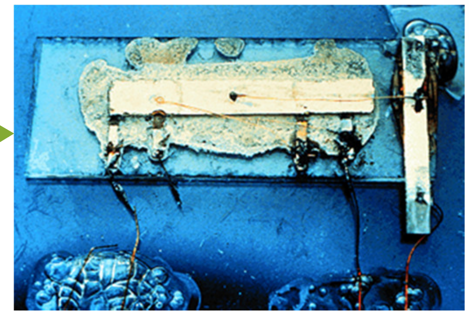

**Seven Keys to the Digital Future, Edinburgh 2010-09-30**
**http://lucacardelli.name**

# Smaller and Smaller

Dec. 23, 1947. John Bardeen and Walter Brattain show the first working transistor.

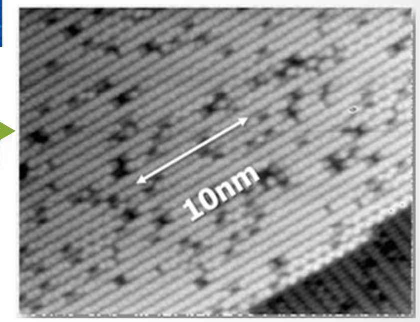


Sep. 1958. Jack Kilby builds the first integrated circuit.



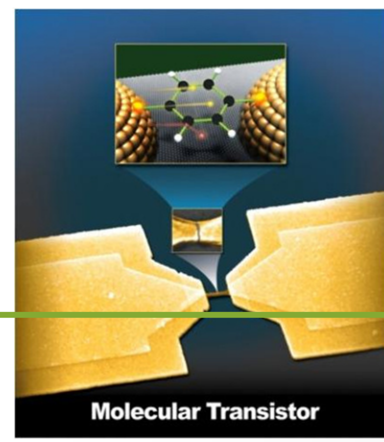
50 years later

Jan. 2010. Intel and Micron announce 25nm NAND flash.



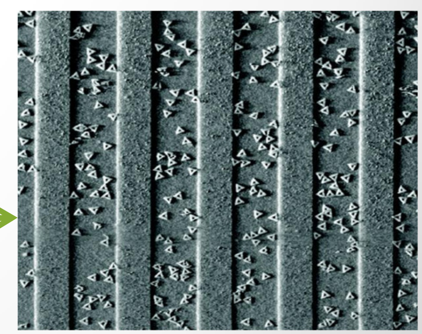

Scanning tunneling microscope image of a silicon surface showing 10nm is ~20 atoms across

Dec. 24, 2009. Working transistor made of a single molecule.




Observation of molecular orbital gating. *Nature*, 2009; 462 (7276): 1039

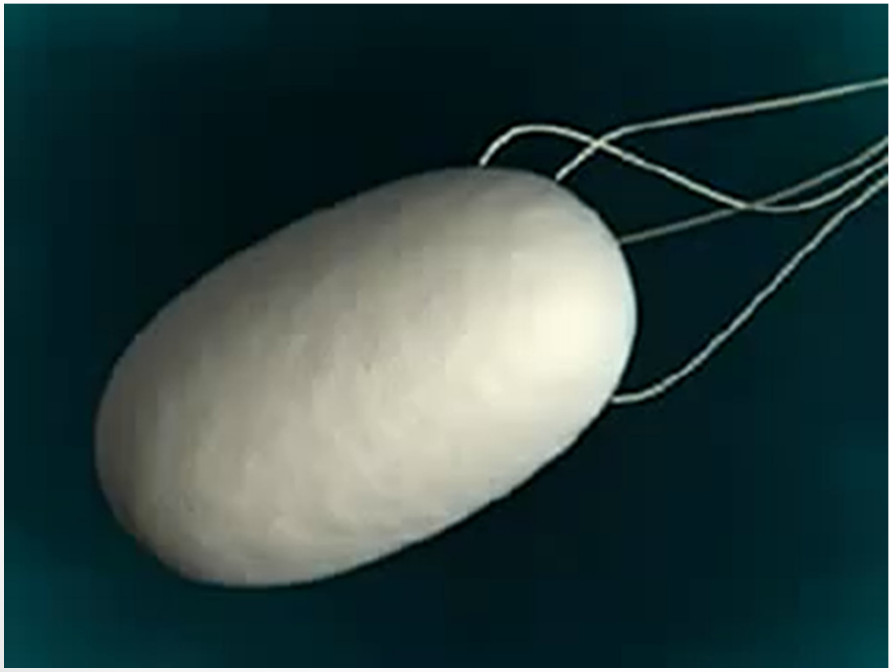
*<10* iterations of Moore's Law left!
The race is on for *molecular scale integrated circuits*.



Placement and orientation of individual DNA shapes on lithographically patterned surfaces.
Nature Nanotechnology 4, 557 – 561 (2009).

# Building The Small*est* 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*.



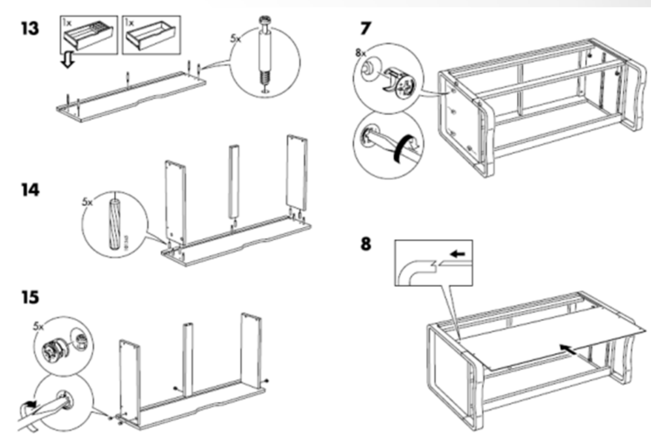
# Molecular IKEA

- Nature can self-assemble.
  Can we?

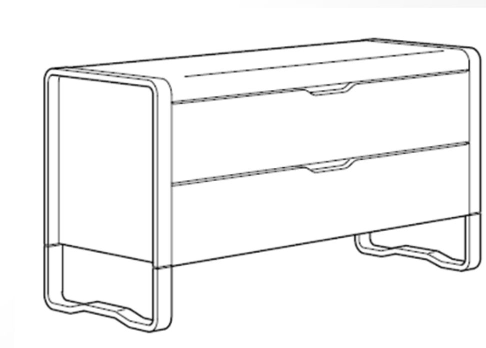
- *"Dear IKEA, please send me a chest of drawers that assembles itself."*

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

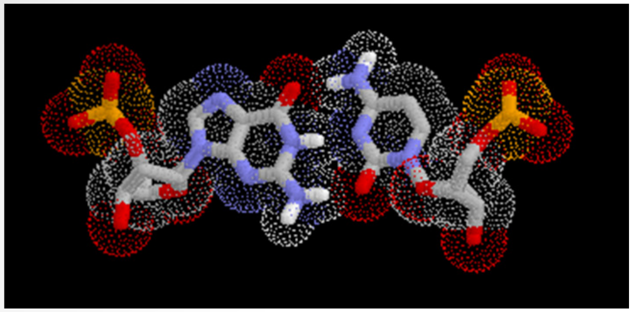
- At the molecular scale many such materials exist; let's pick one…



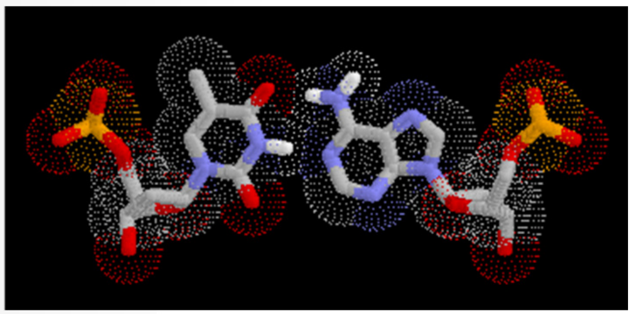

Add water



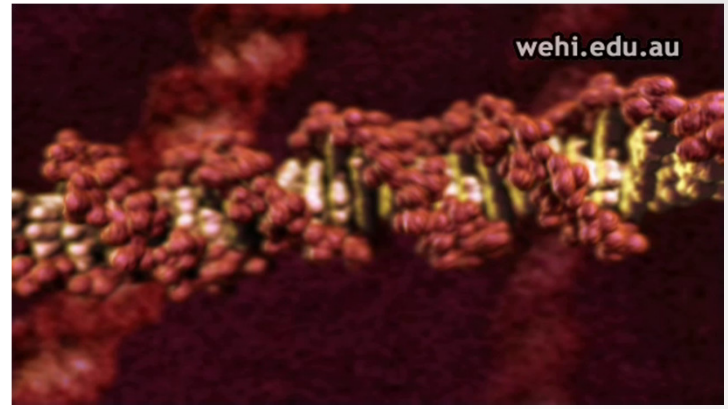
# DNA



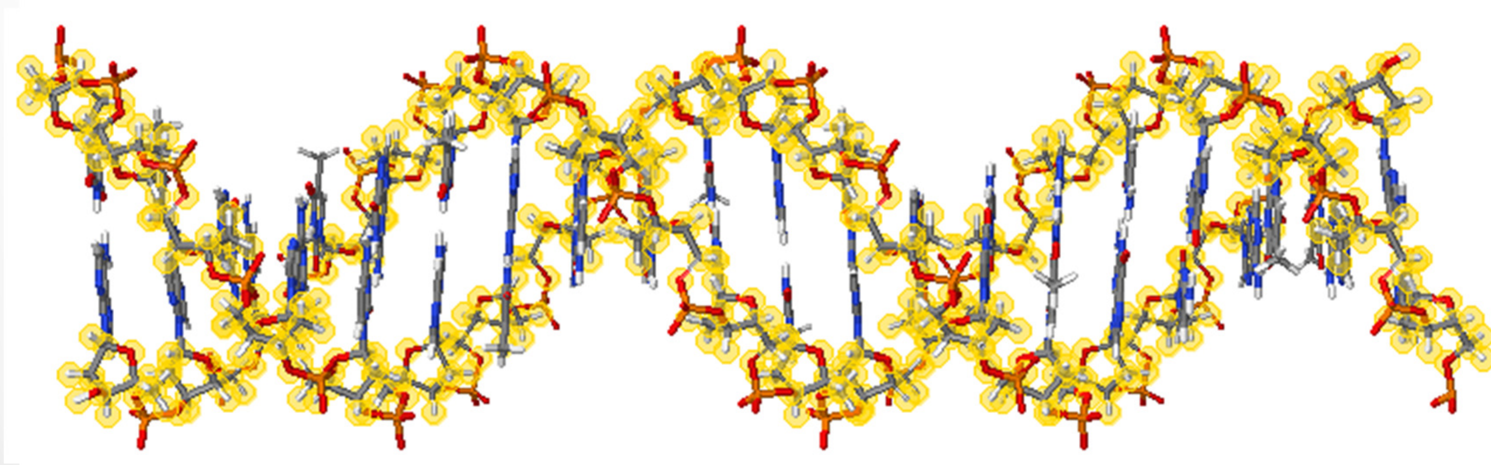
GC Base Pair
Guanine–Cytosine



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)

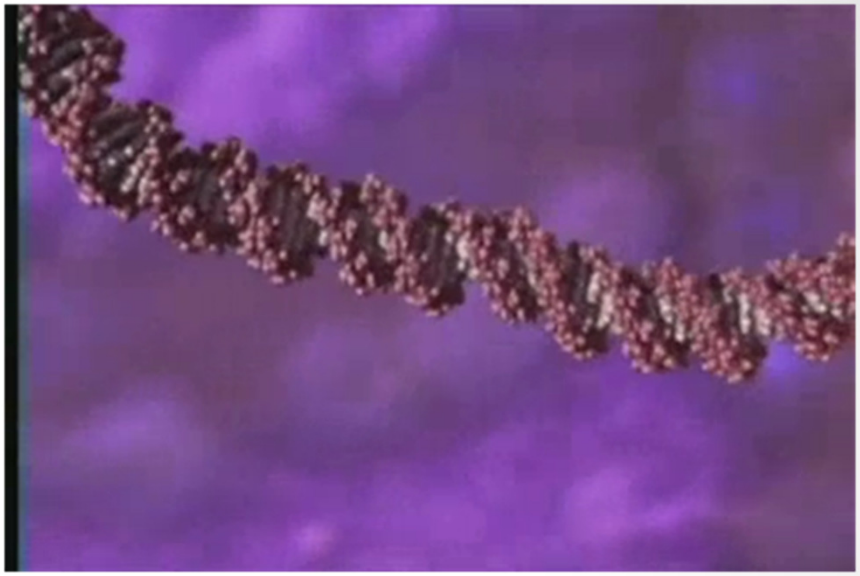
# Robust, and *Long*

- 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

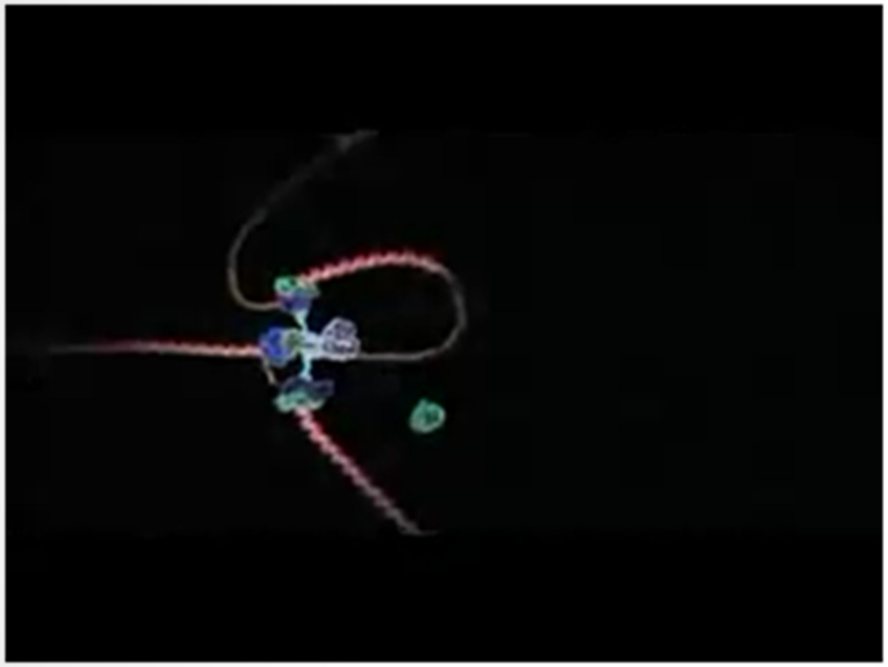


DNA wrapping into chromosomes

Andromeda Galaxy
2.5 million light years

# 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)

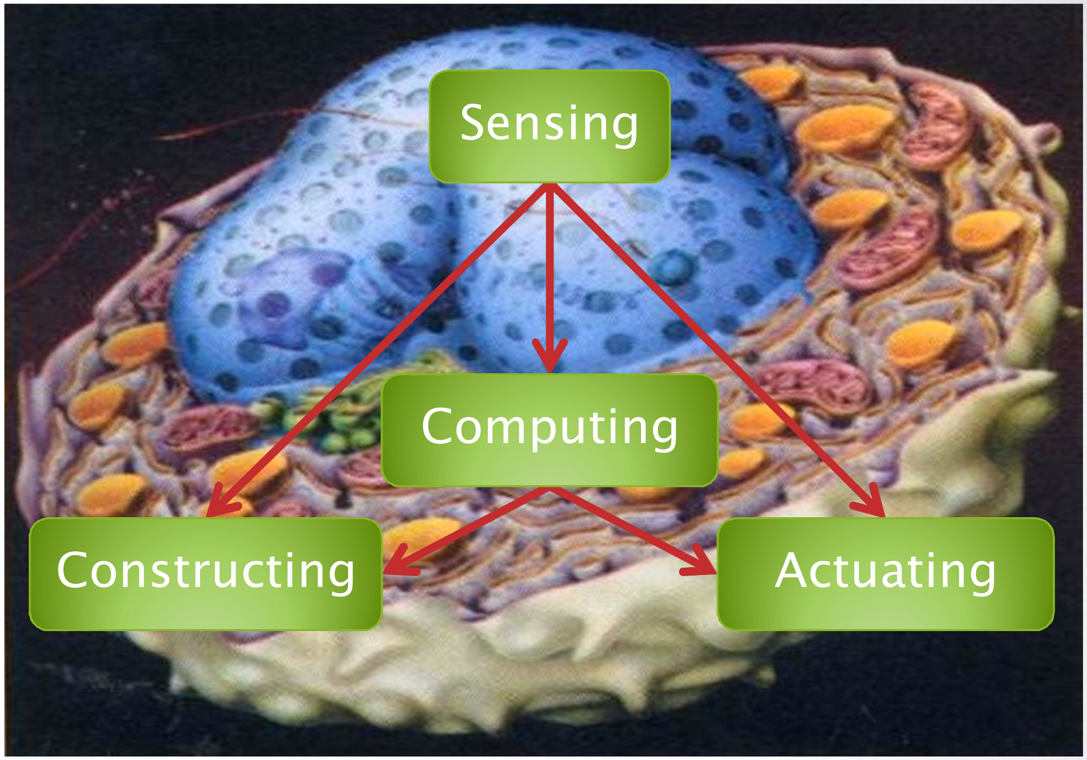
DNA transcription in *real time*

RNA polymerase II: 15-30 base/second

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

# Nanoscale Engineering

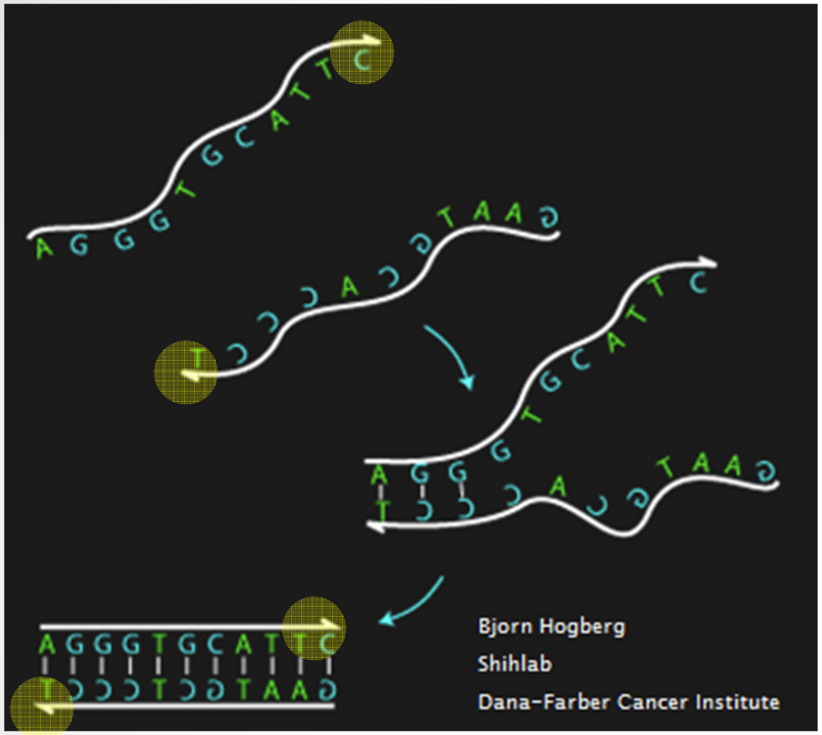
- Sensing
  - Reacting to forces
  - Binding to molecules
- Actuating
  - Releasing molecules
  - Producing forces
- Constructing
  - Chassis
  - Growth
- Computing
  - Signal Processing
  - Decision Making

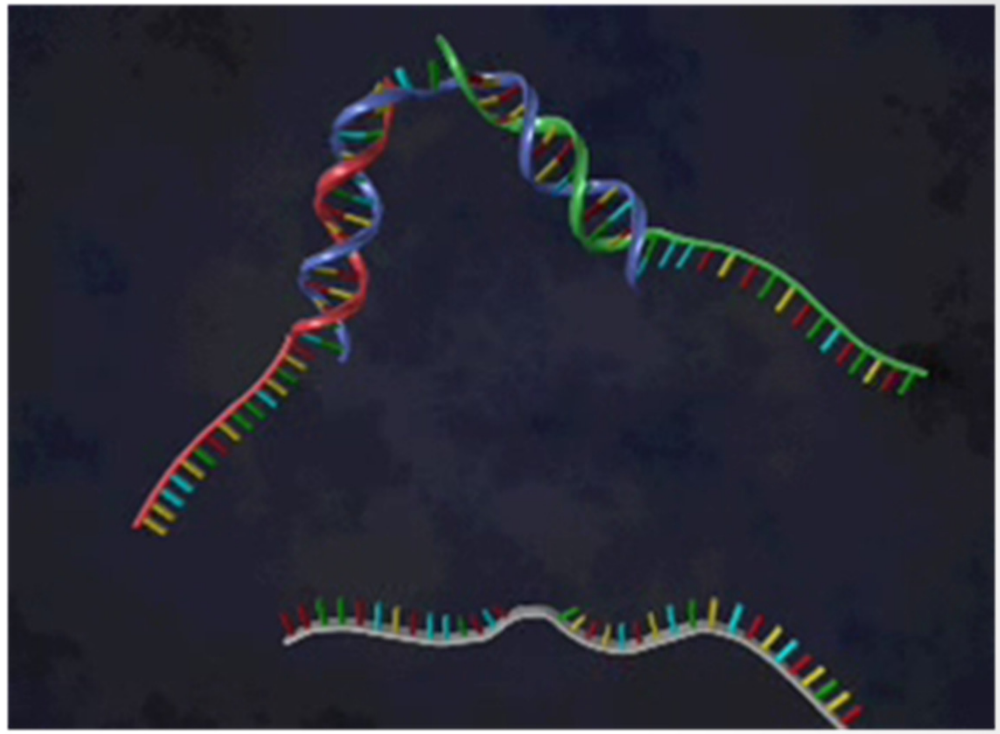



Nucleic Acids can do all this.

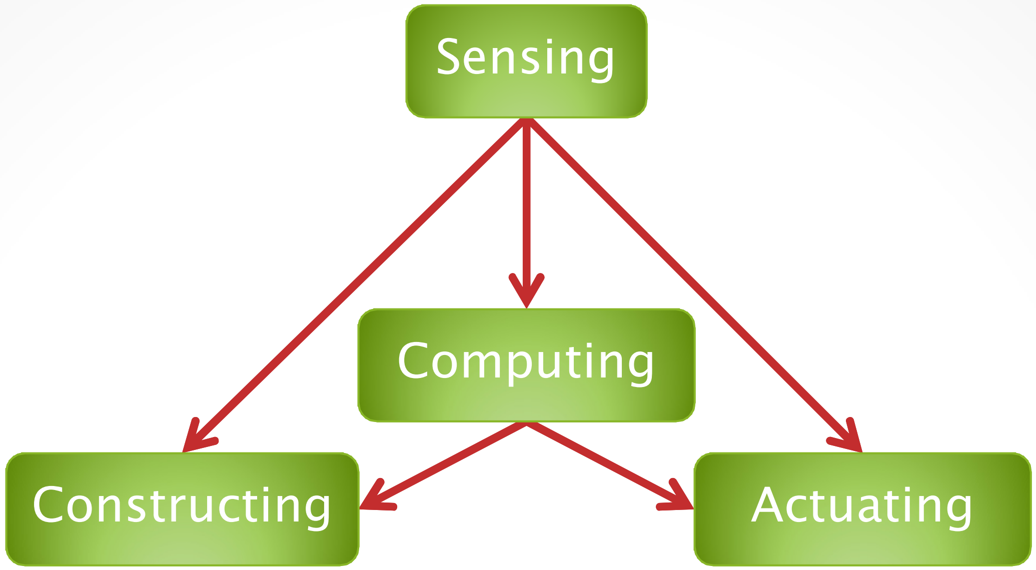
And interface to biology.

# Hybridization

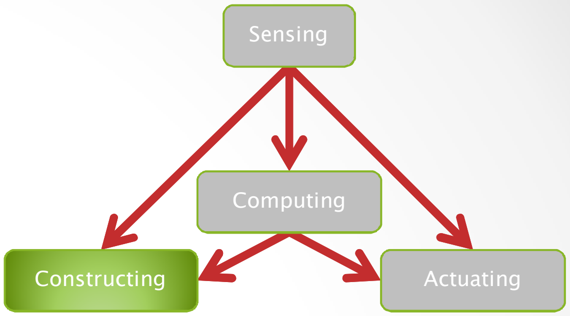


- Strands with **opposite orientation** and **complementary base pairs** stick to each other (Watson-Crick duality).
- This is all we are going to use
  - We are not going to exploit DNA replication, transcription, translation, **restriction and ligation enzymes**, etc., which enable other classes of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



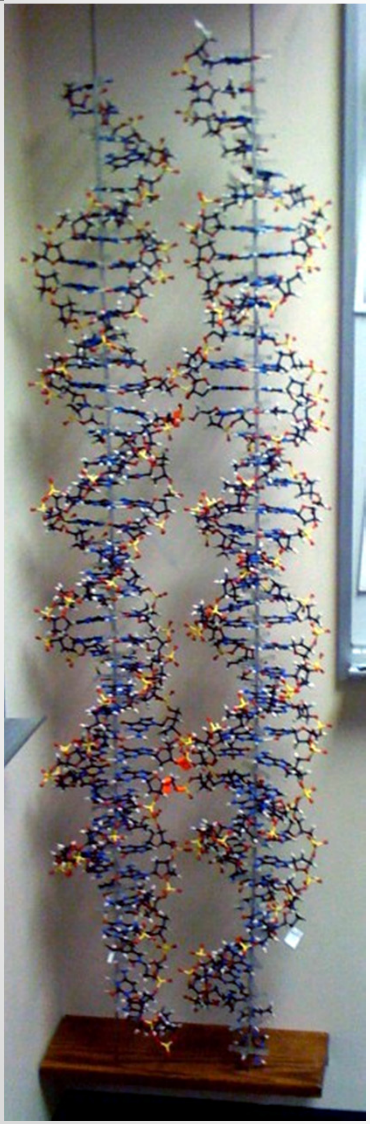

# Hybridization Tricks

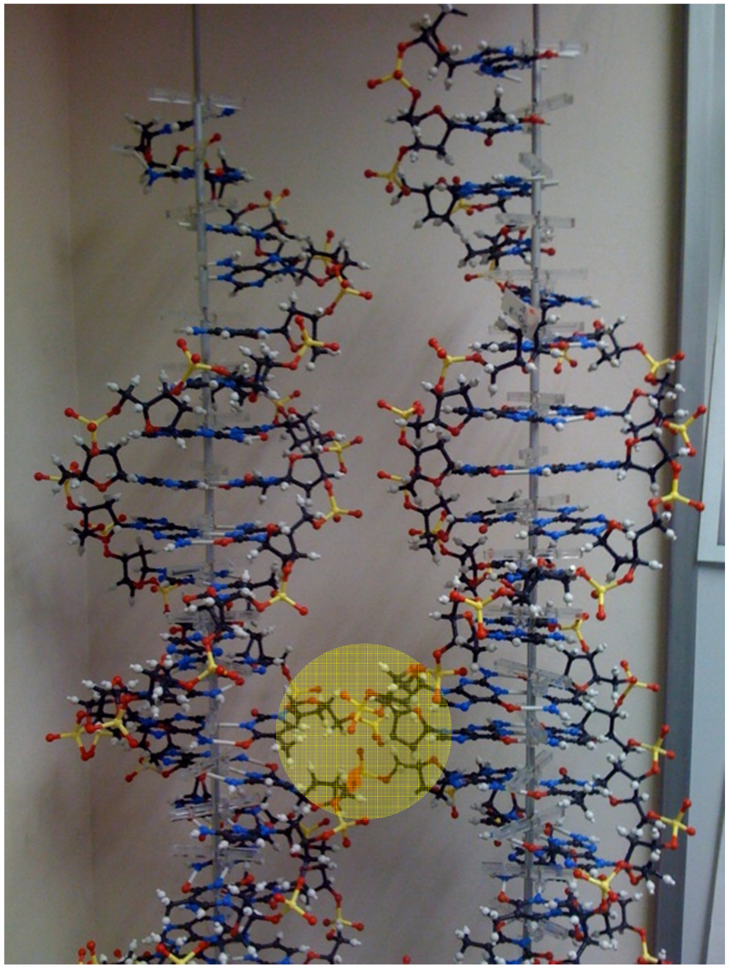



# Constructing

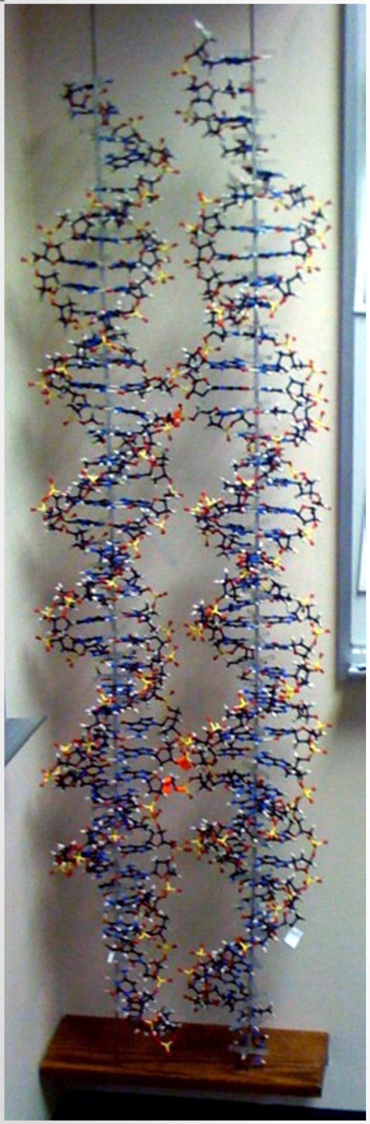
• • •

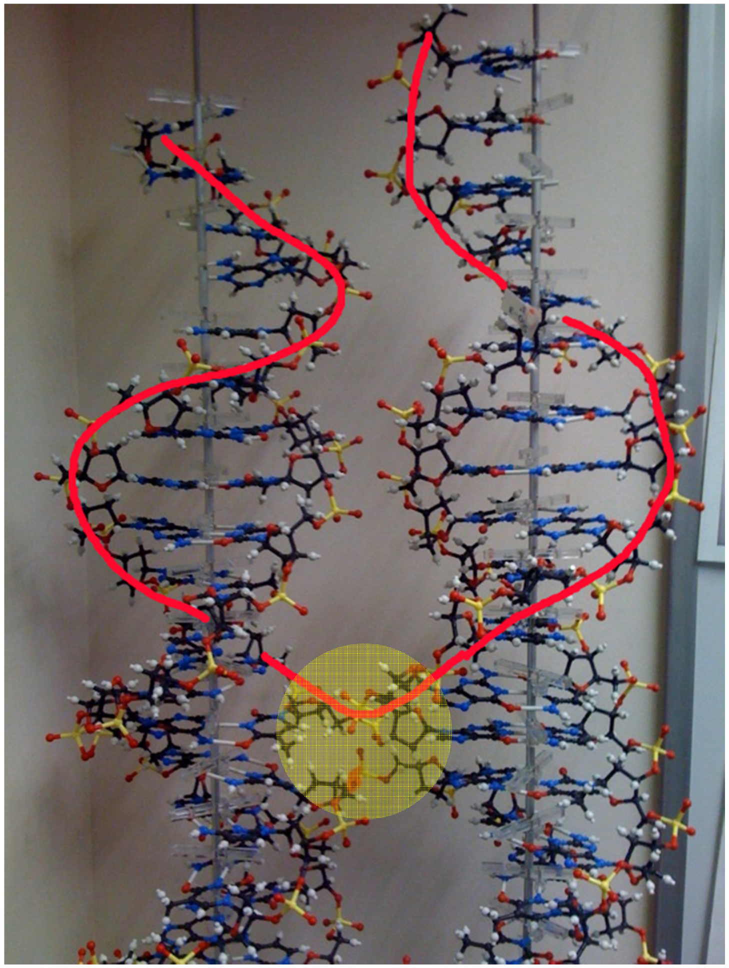
# Crosslinking



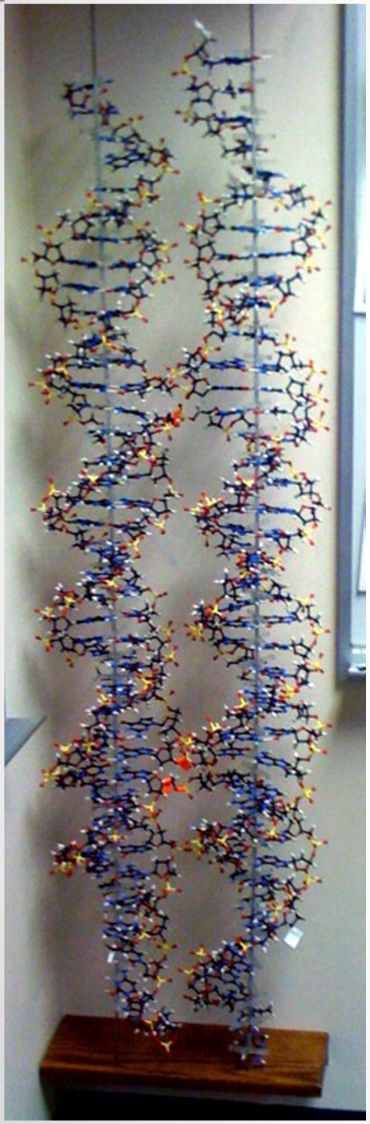


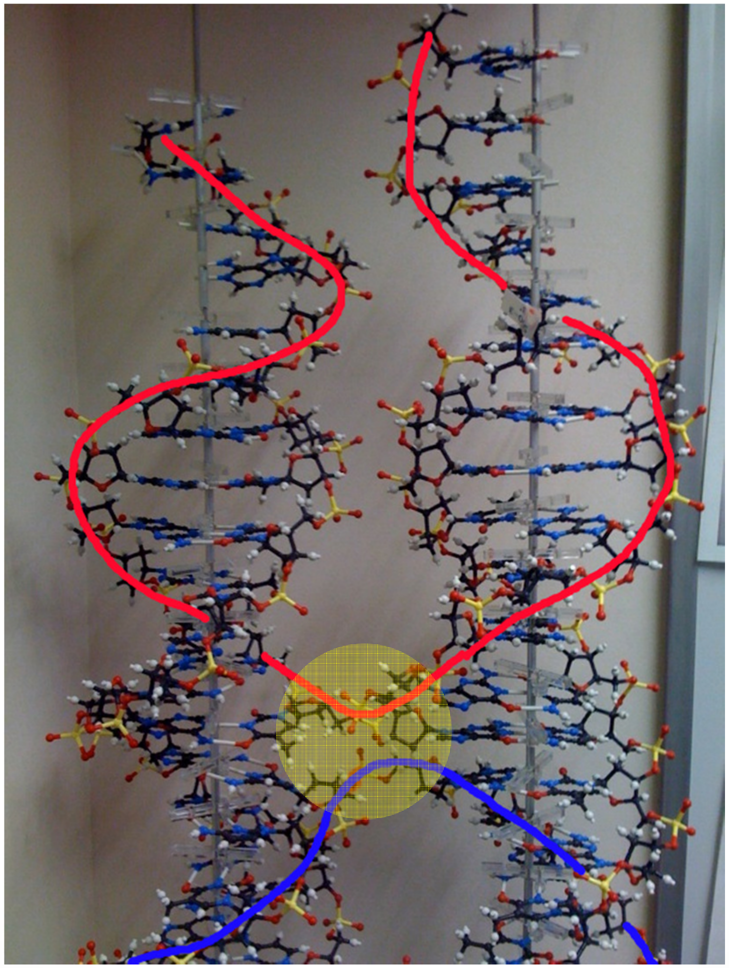
# Crosslinking



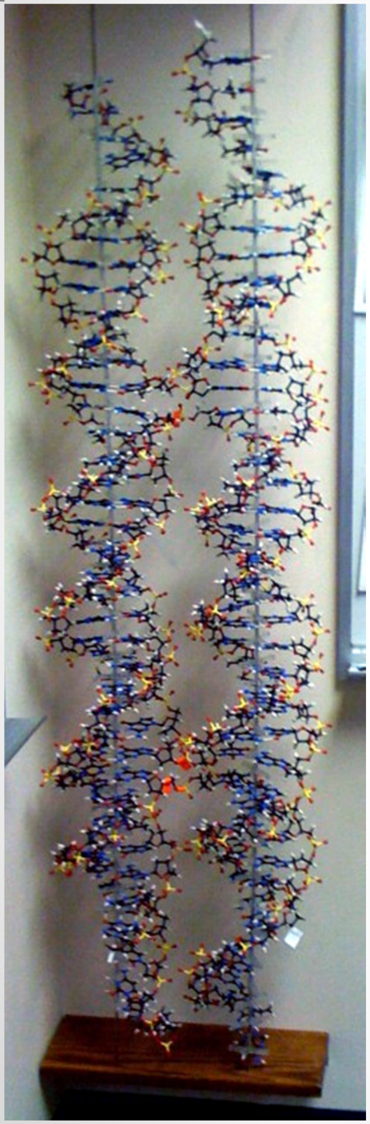


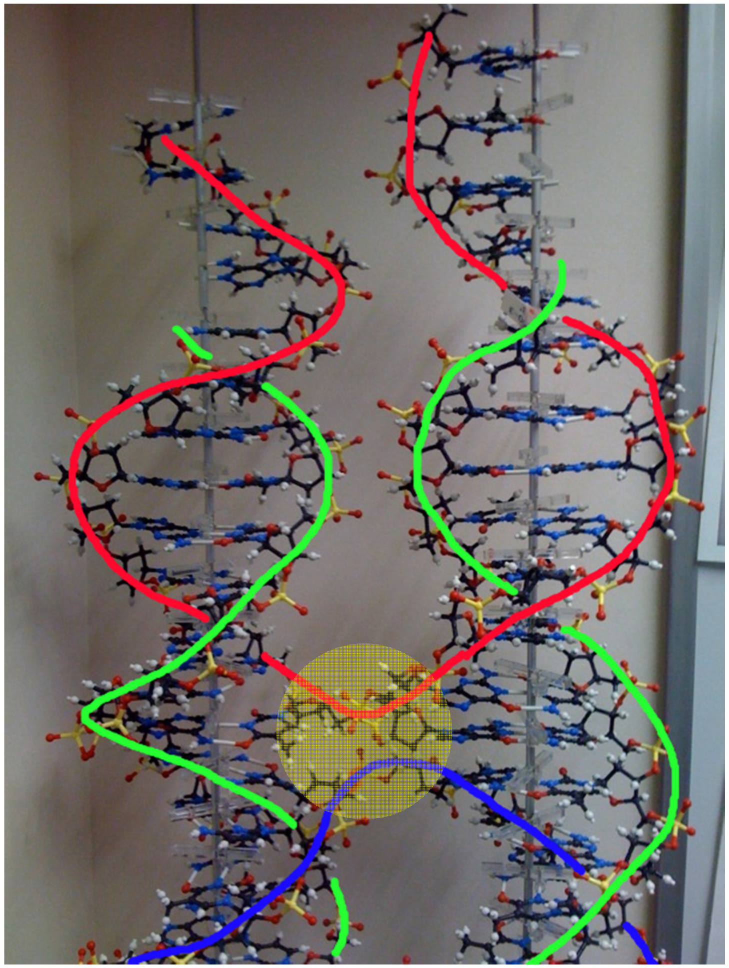
# Crosslinking



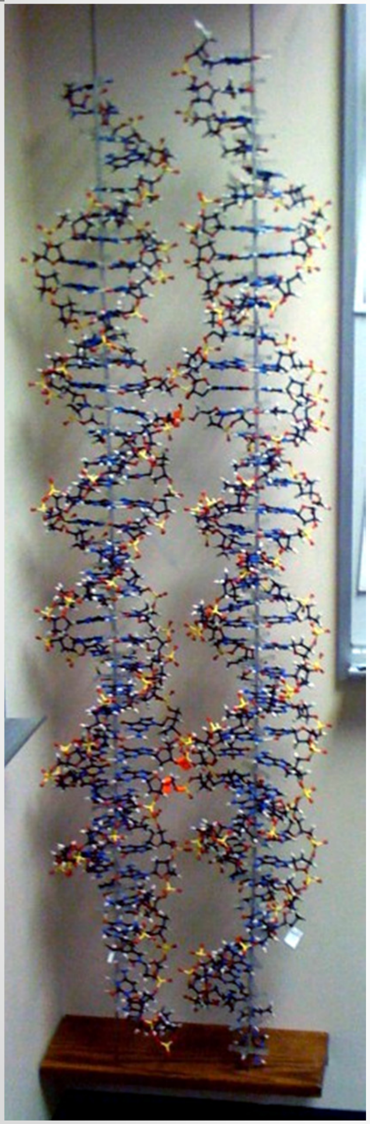


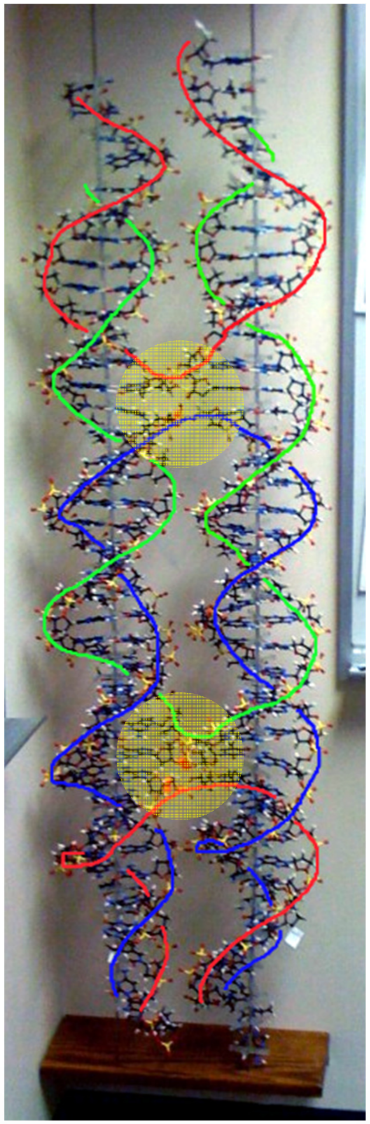
# Crosslinking



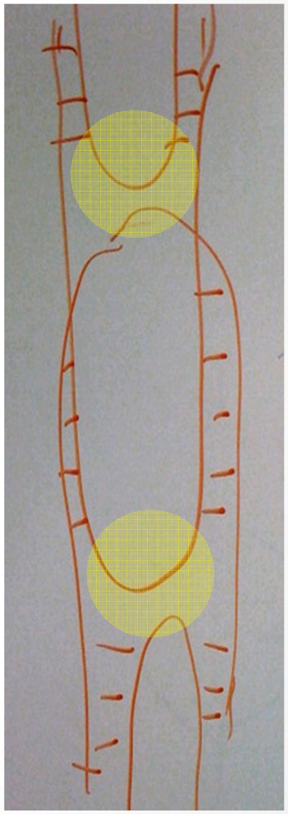


# Crosslinking



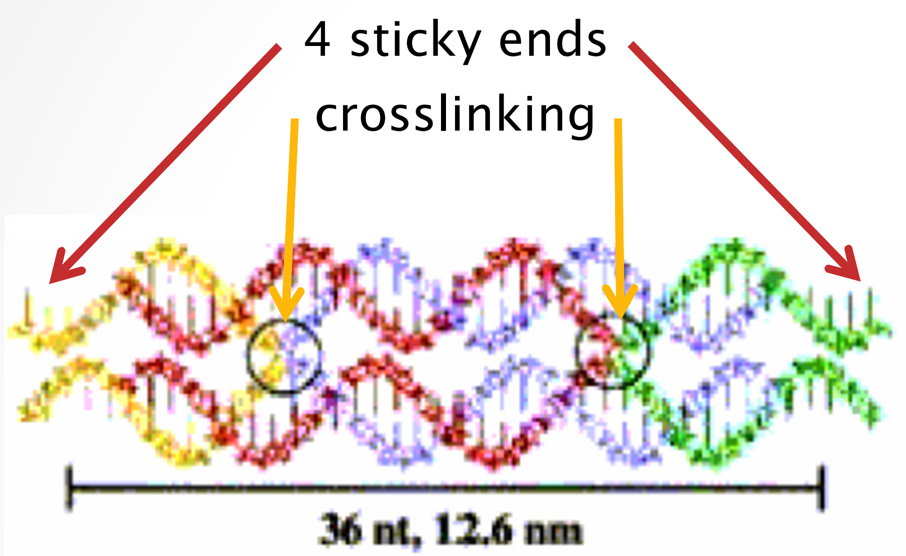


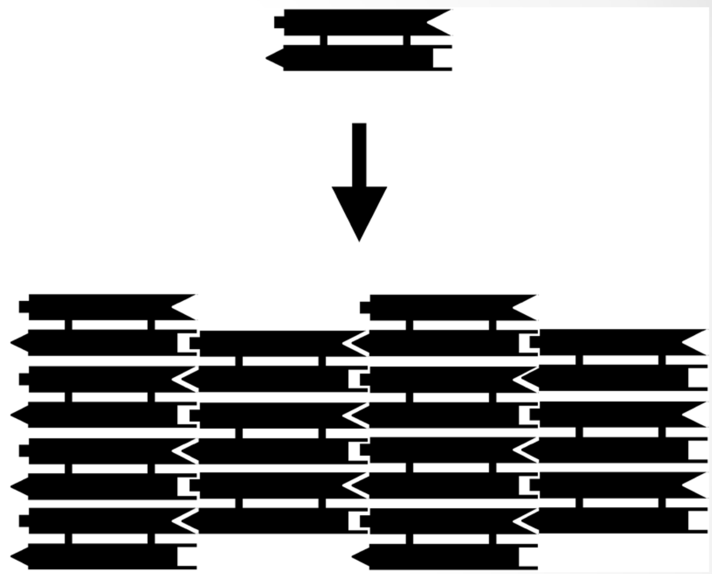
In nature, crosslinking is deadly (blocks DNA replication).

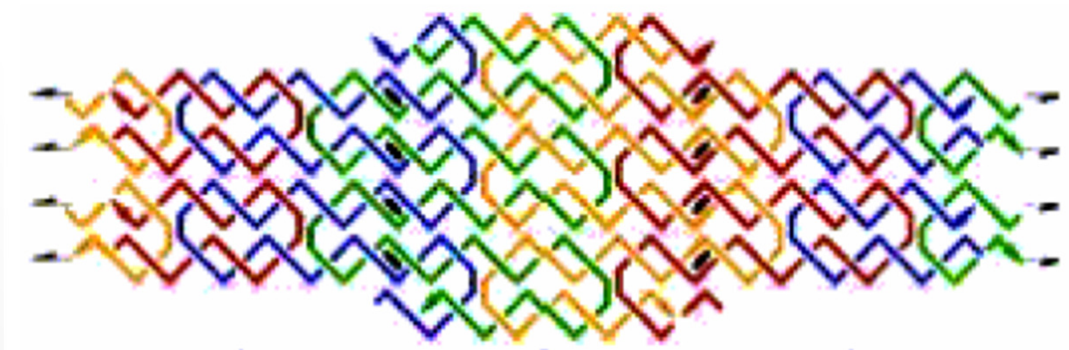


In engineering, crosslinking is the key to using DNA as a construction material.

# DNA Tiling



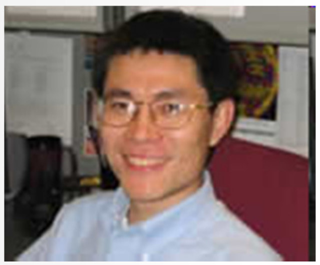





Construction and manipulation of DNA tiles in free space

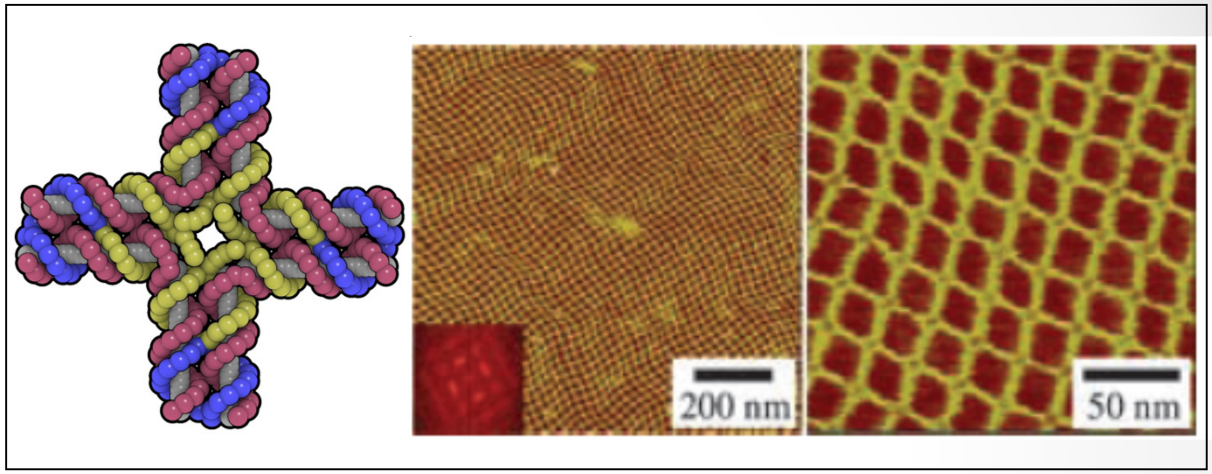
Pankhudi

# 2D DNA Lattices

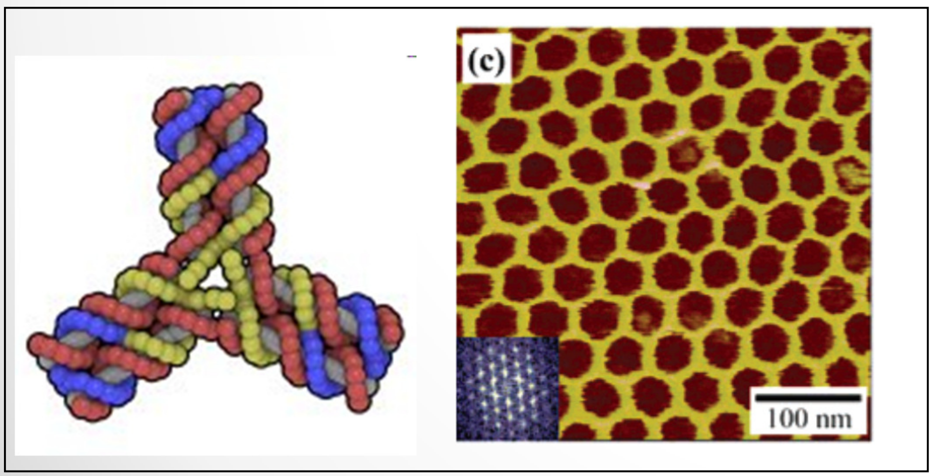


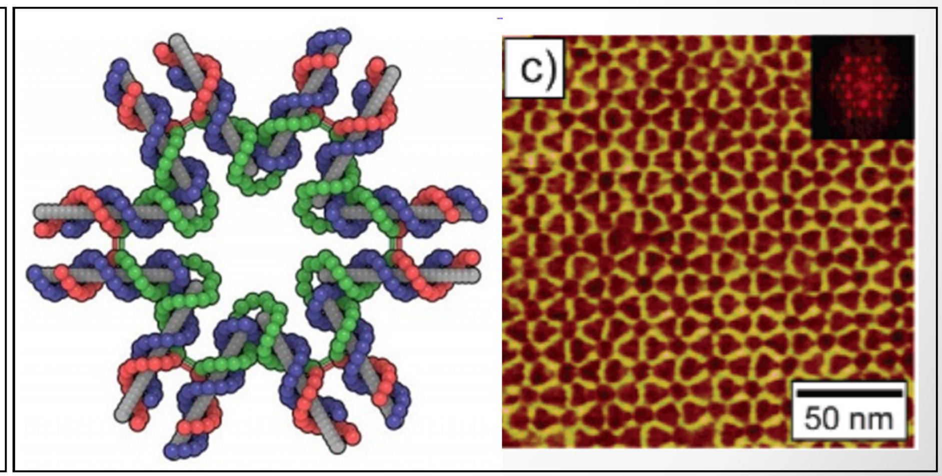
Chengde Mao

Purdue University, USA



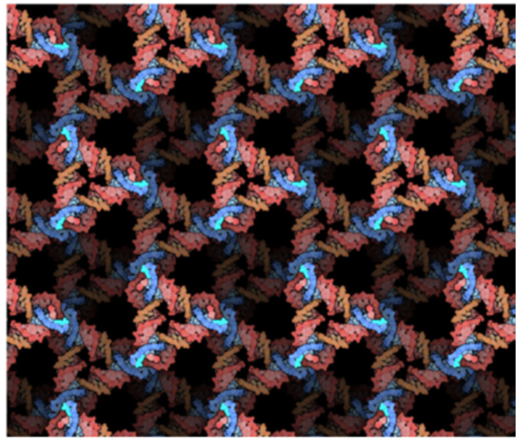

N-point Stars

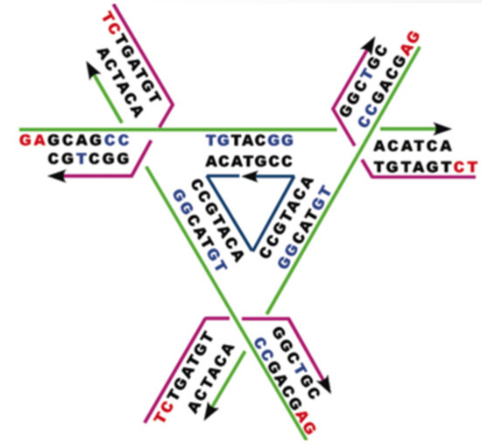


# 3D DNA Structures

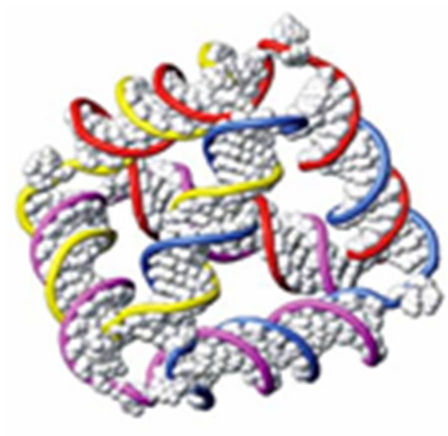
Ned Seeman
NYU

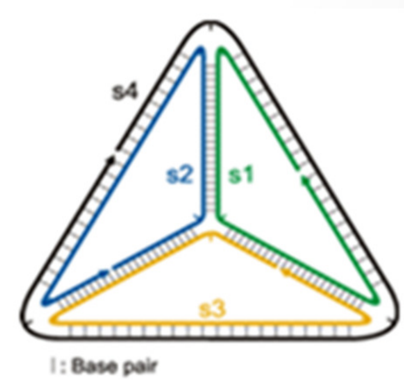





3D Cyrstal

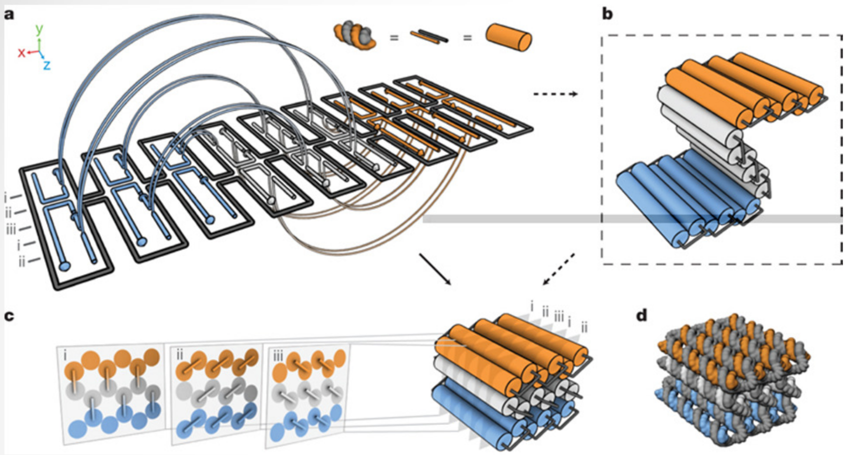
AndrewTuberfield
Oxford

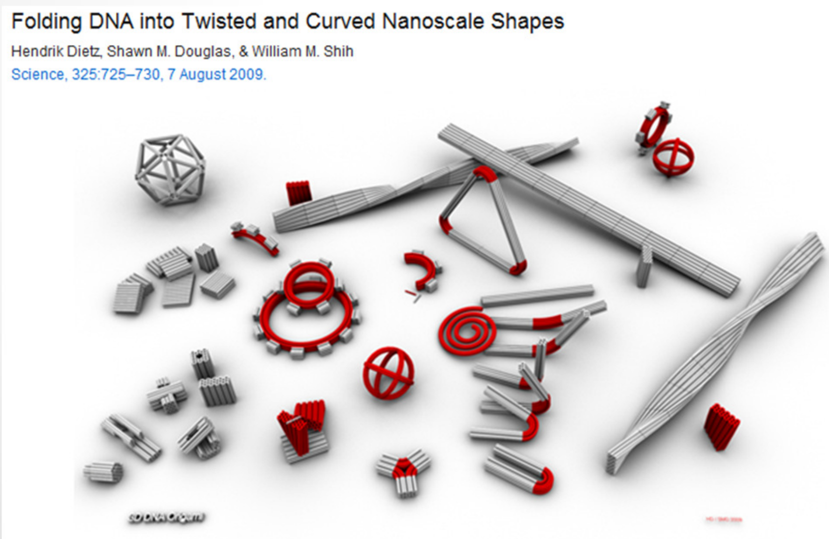


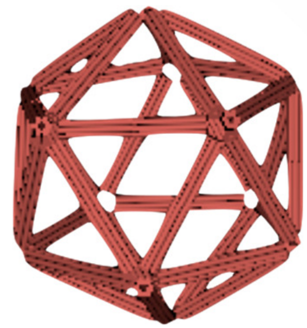



Tetrahedron

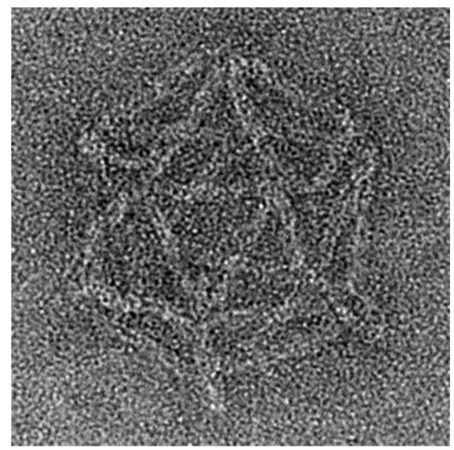
# CADnano



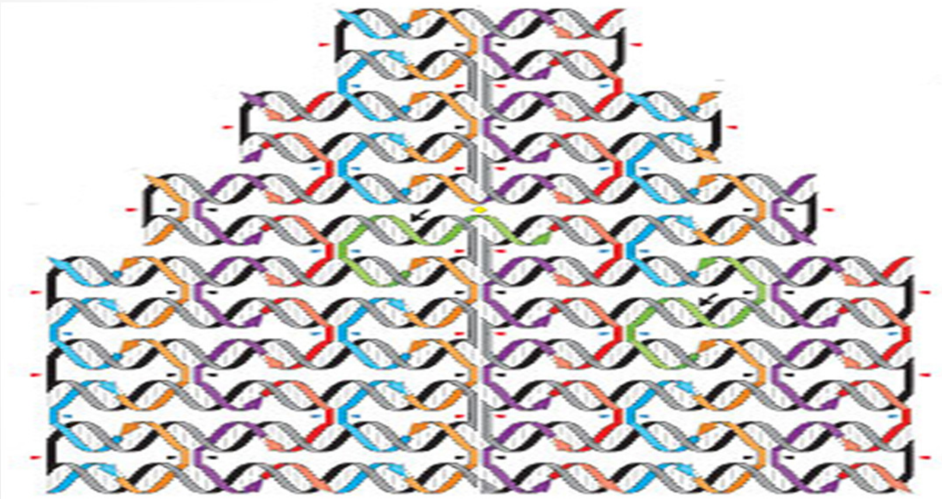
William Shih
Harvard



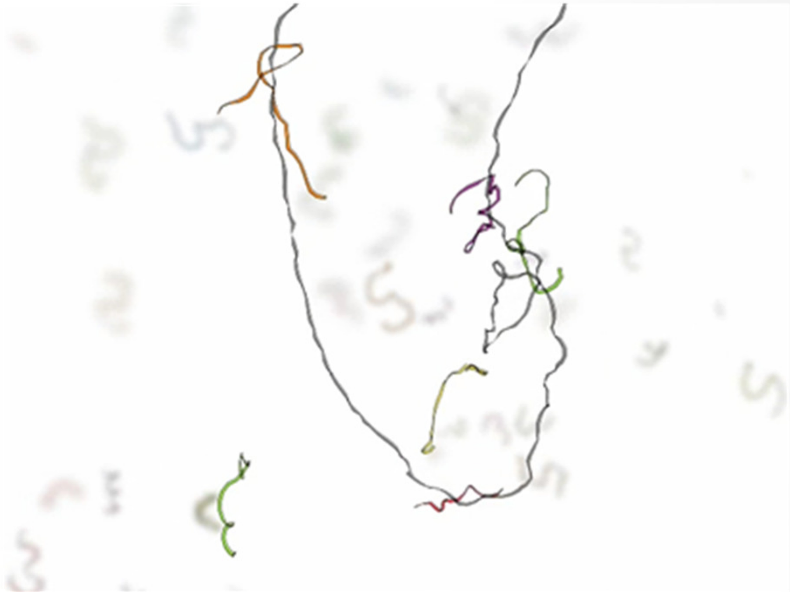
S.M. Douglas, H. Dietz, T. Liedl, B. Högberg, F. Graf and W. M. Shih
Self-assembly of DNA into nanoscale three-dimensional shapes, Nature (2009)

# DNA Origami

- *Folding* long (7000bp) naturally occurring (viral) ssDNA
- By lots of short 'staple' strands that constrain it

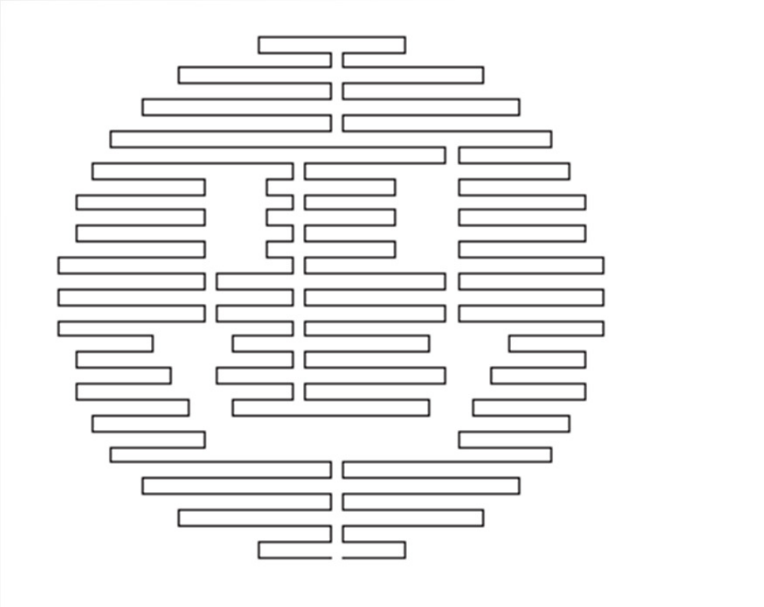


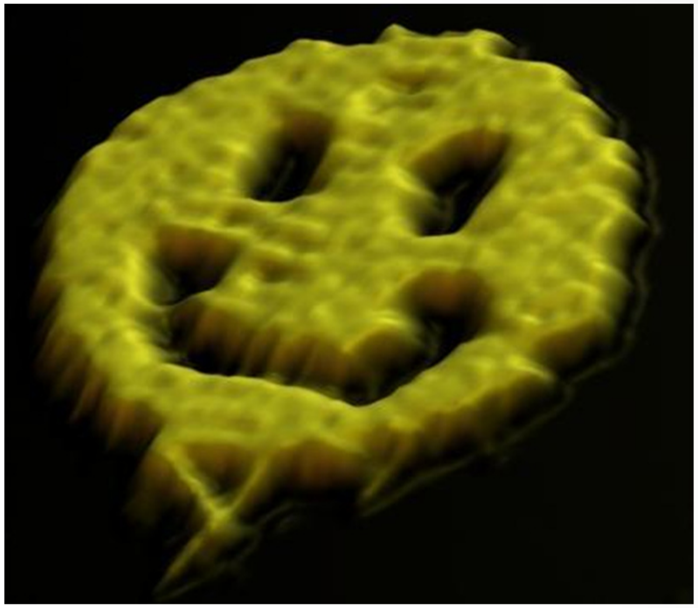
PWK Rothemund, *Nature* 440, 297 (2006)



Black: long viral strand
Color: short staple strands

# DNA Origami





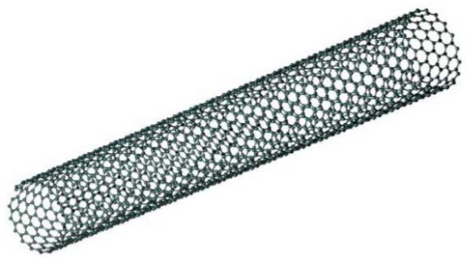
Paul Rothemund's "Disc with three holes" (2006)

Paul W K Rothemund
California Institute of Technology

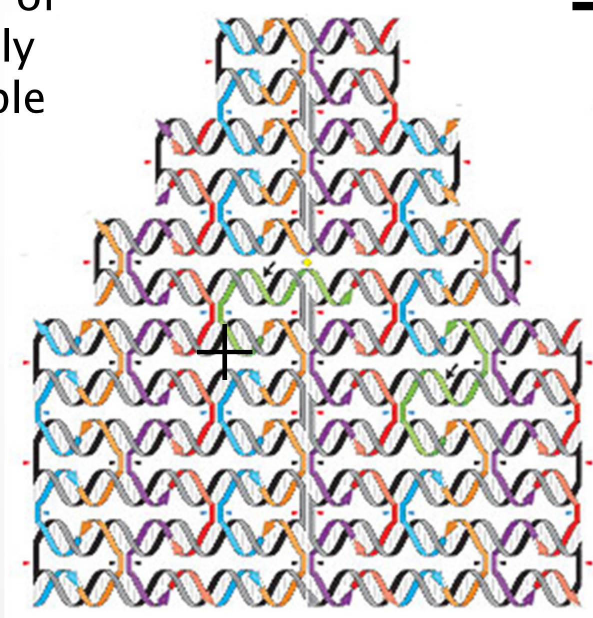
This means we can already self-assemble meso-scale structures.

# DNA Circuit Boards

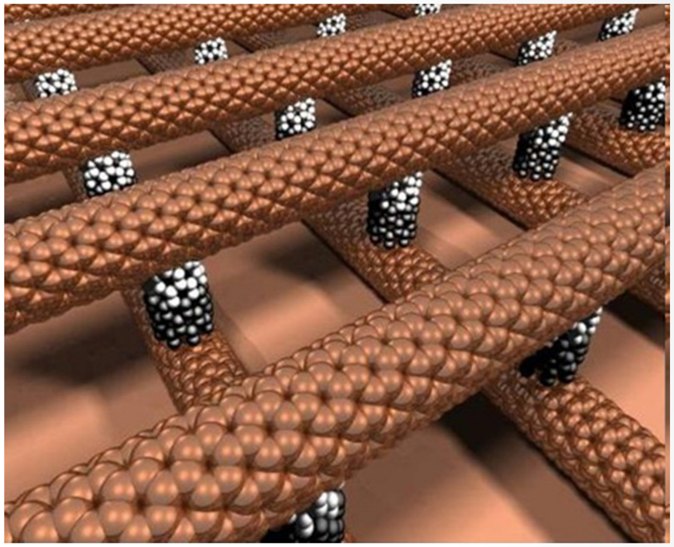
DNA-wrapped nanotubes



6 nm grid of individually addressable pixels

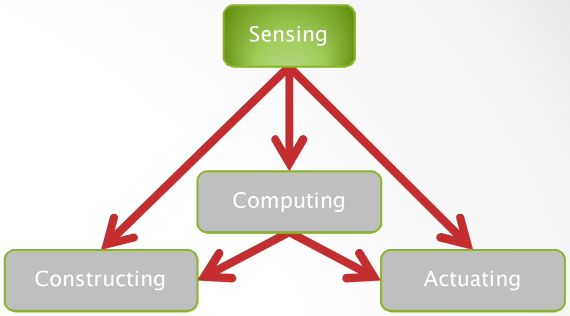


PWK Rothemund, *Nature* 440, 297 (2006)



European Nanoelectronics Initiative Advisory Council

"What we are really making are tiny DNA circuit boards that will be used to assemble other components."
*Greg Wallraff, IBM*

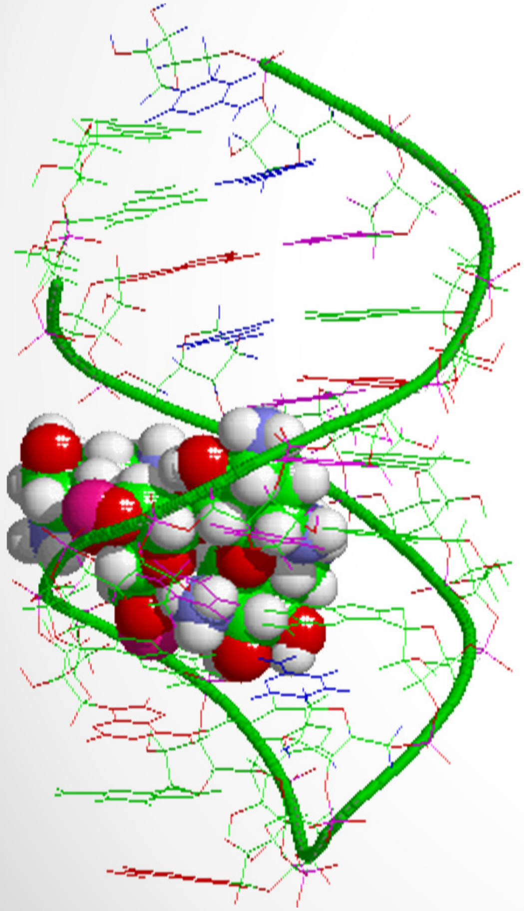


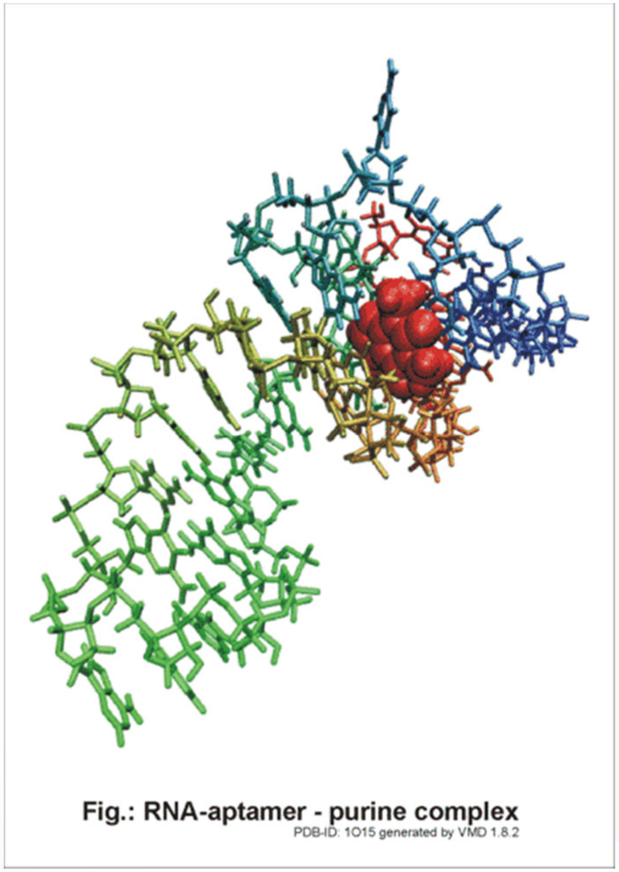

# Sensing

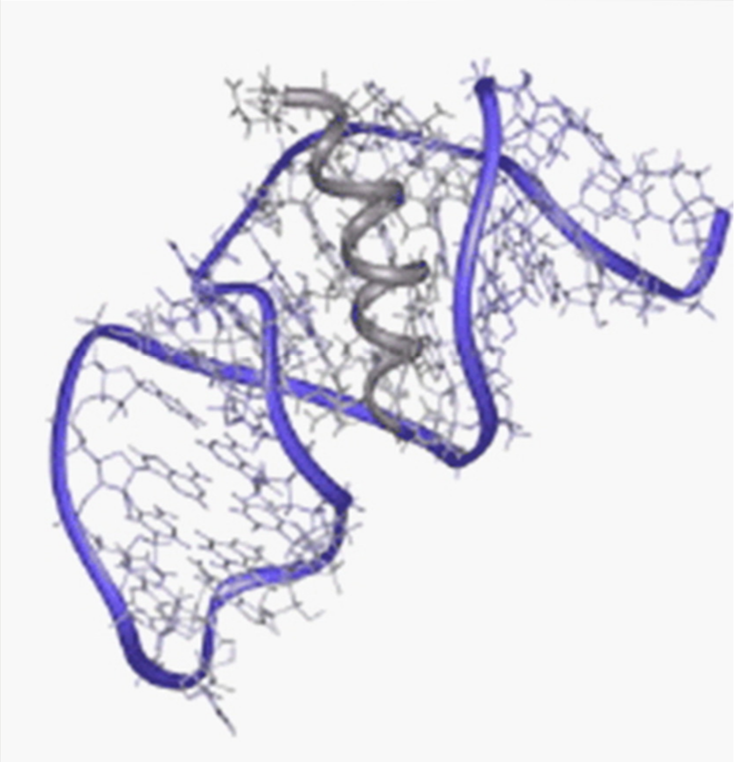
• • •

# Aptamers

- Artificially eveloved DNA molecules that stick to anything you like (highly selectively).

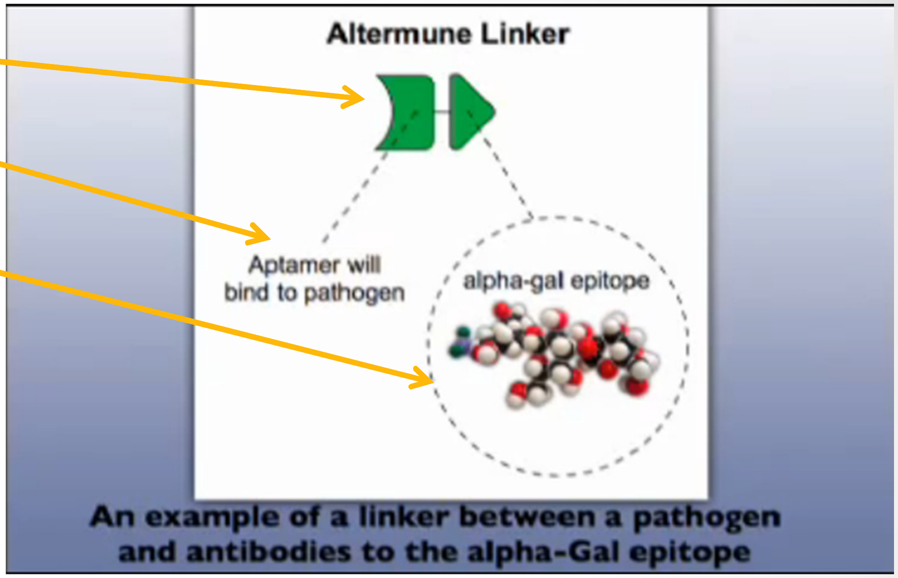




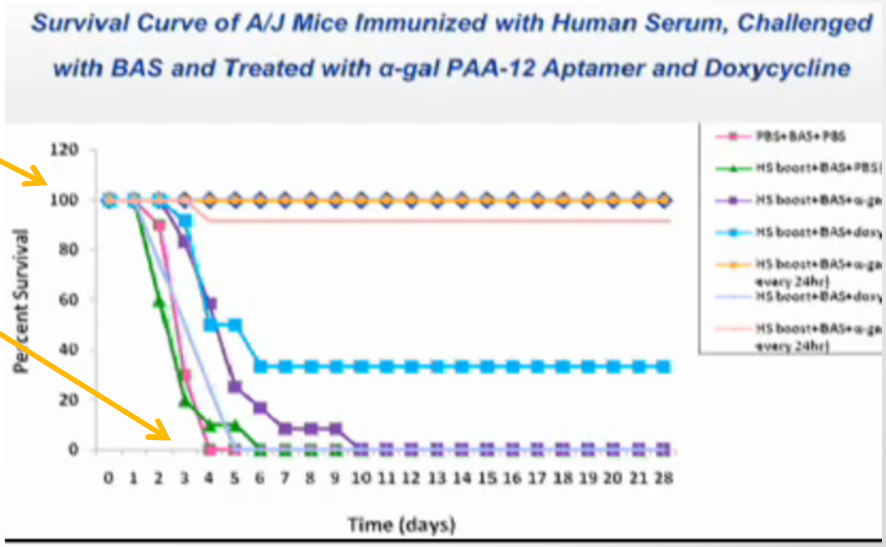



# Pathogen Spotlights

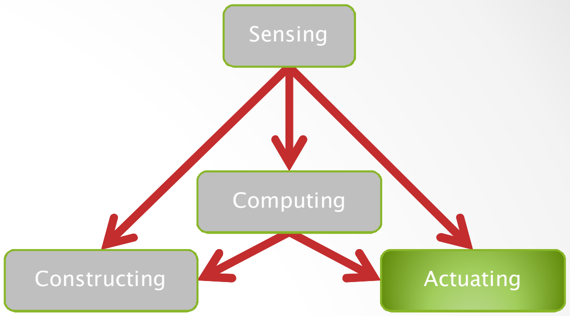
- DNA aptamer binds to:
  - A) a pathogen
  - B) a molecule our immune system already hates and immediately removes (eats) along with anything attached to it




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



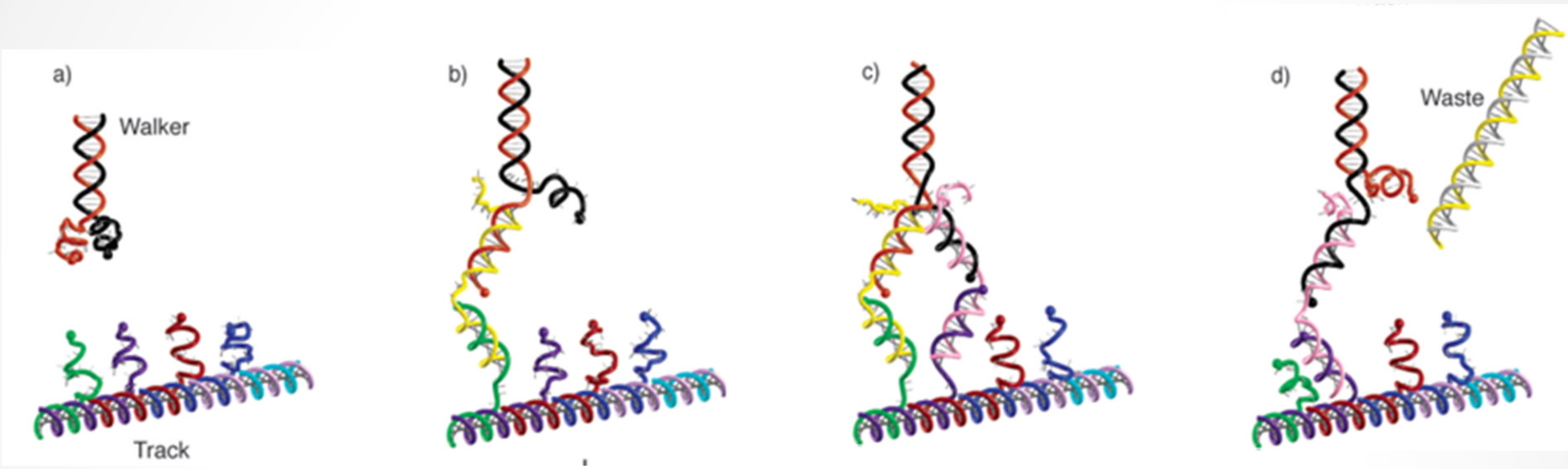

Kary Mullis (incidentally, also Nobel prize for inventing the Polymerase Chain Reaction)




# Actuating

• • •

# DNA Walkers

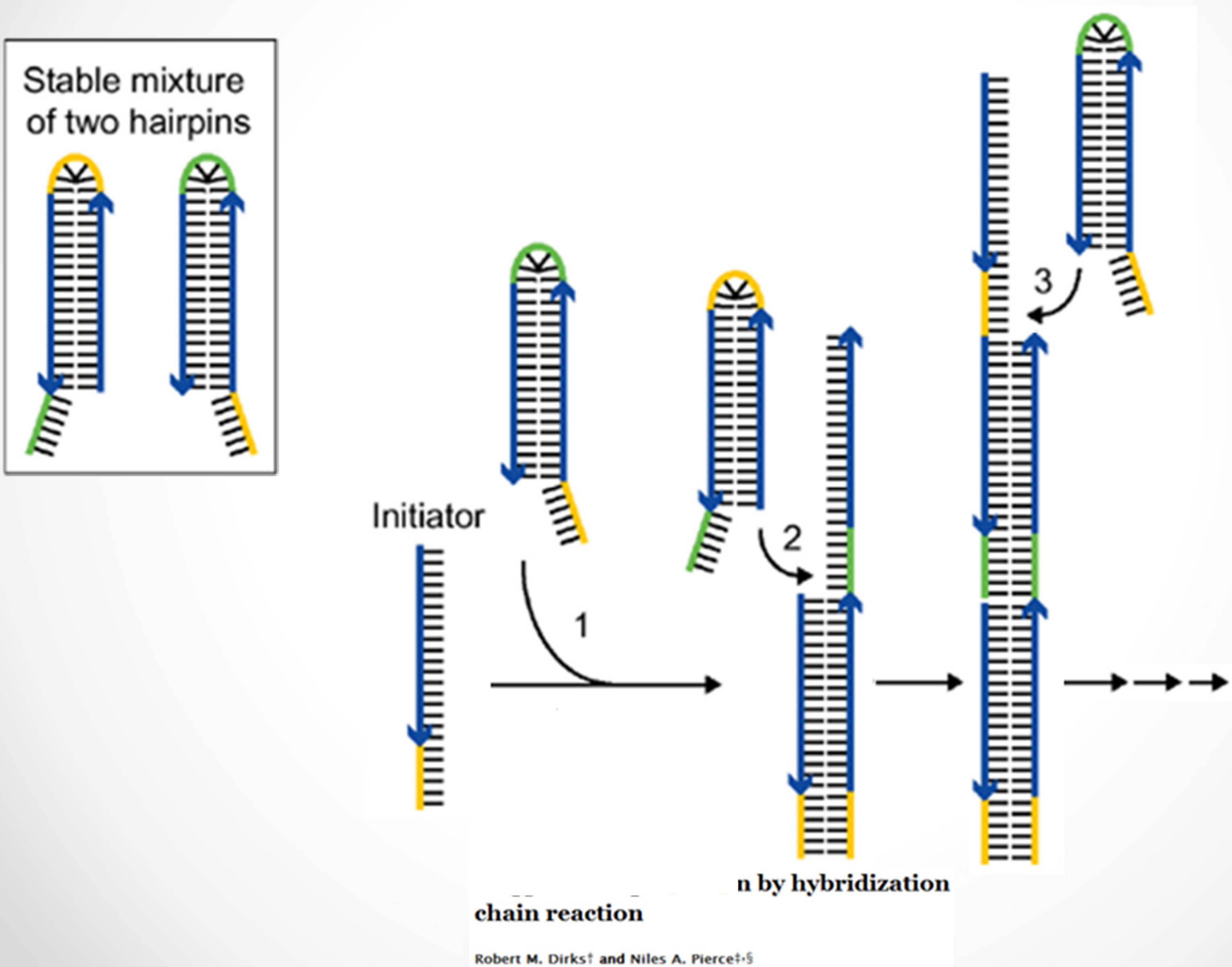



J|A|C|S COMMUNICATIONS
Published on Web 08/17/2004

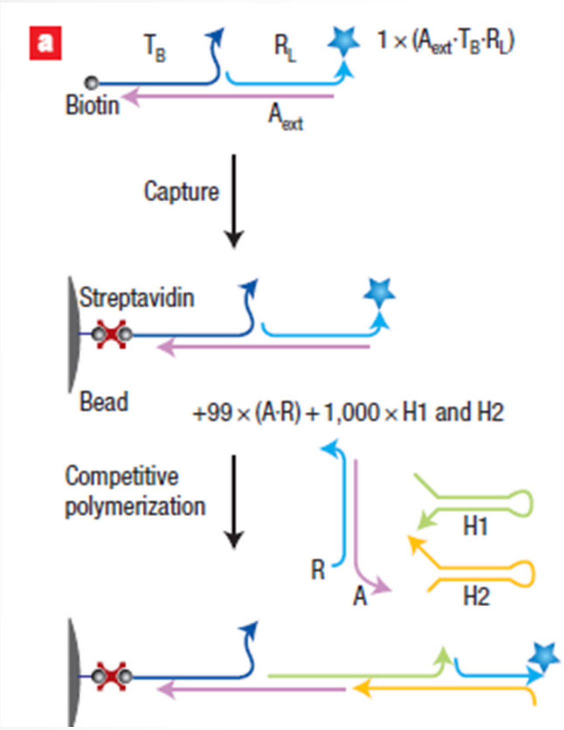
**A Synthetic DNA Walker for Molecular Transport**

Jong-Shik Shin[†] and Niles A. Pierce[*,†,‡]
*Departments of Bioengineering and Applied & Computational Mathematics, California Institute of Technology, Pasadena, California 91125*

# Hybridization Chain Reaction



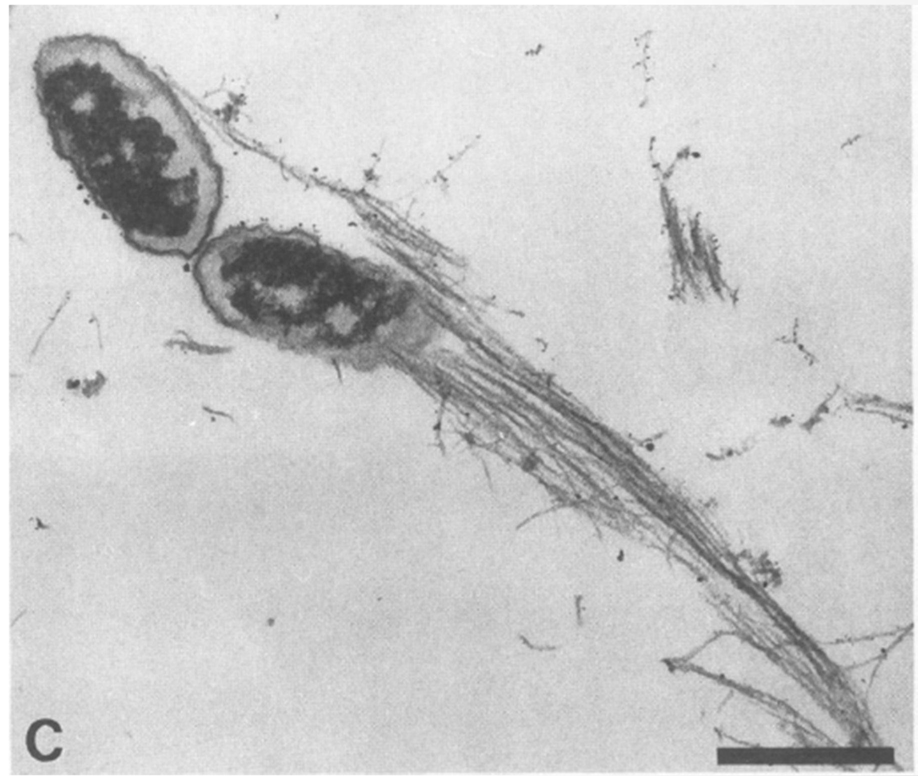
# Polymerization Motor




An autonomous polymerization motor powered by DNA hybridization

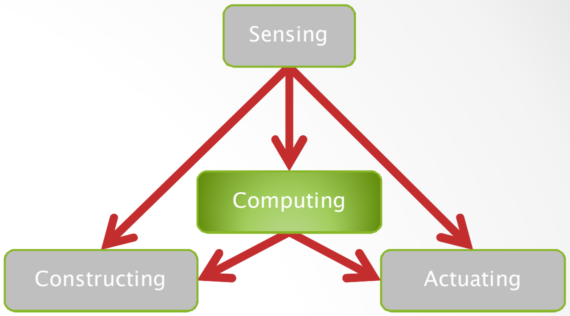
SUVIR VENKATARAMAN[1], ROBERT M. DIRKS[1], PAUL W. K. ROTHEMUND[2,3], ERIK WINFREE[2,3] AND NILES A. PIERCE[1,4]*

Rickettsia (spotted fever)




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

ROBERT A. HEINZEN, STANLEY F. HAYES, MARIUS G. PEACOCK, AND TED HACKSTADT*




# Computing

• • •

## Basic Notions

# 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
- sdf
  - 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.

# What does DNA Compute?

- Electronics has *electrons*
  - All electrons are the same: you can only count them
  - *Few* electrons = False; *lots* of electrons = True
  - But Boolean Logic is only a necessary evil to build symbolic computation
- DNA computing has *symbols* (DNA words)
  - DNA words are not all the same
  - Symbolic computation on abstract signals can be done *directly*
  - Signals are presented concurrently (in a soup)
  - No requirement to do Boolean Logic
- Then, what are our 'gates' (if not Boolean?)
  - Theory of Concurrency
  - Process Algebra as the "Boolean Algebra" of DNA Computing

# Why Compute with DNA?

- Not to solve NP-complete problems.
- Not to put Intel out of business.
- Not to orchestrate protein production.
- To precisely control the organization and dynamics of matter and information at the molecular level.
  - The use of DNA is “accidental”.
  - No genes involved.
  - In fact, no material of biological origin.

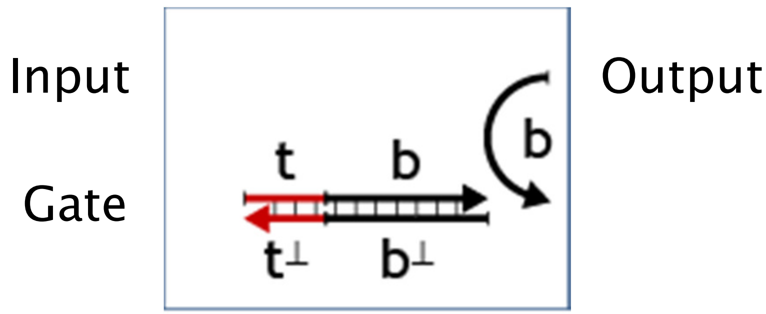
# Rules of the Game

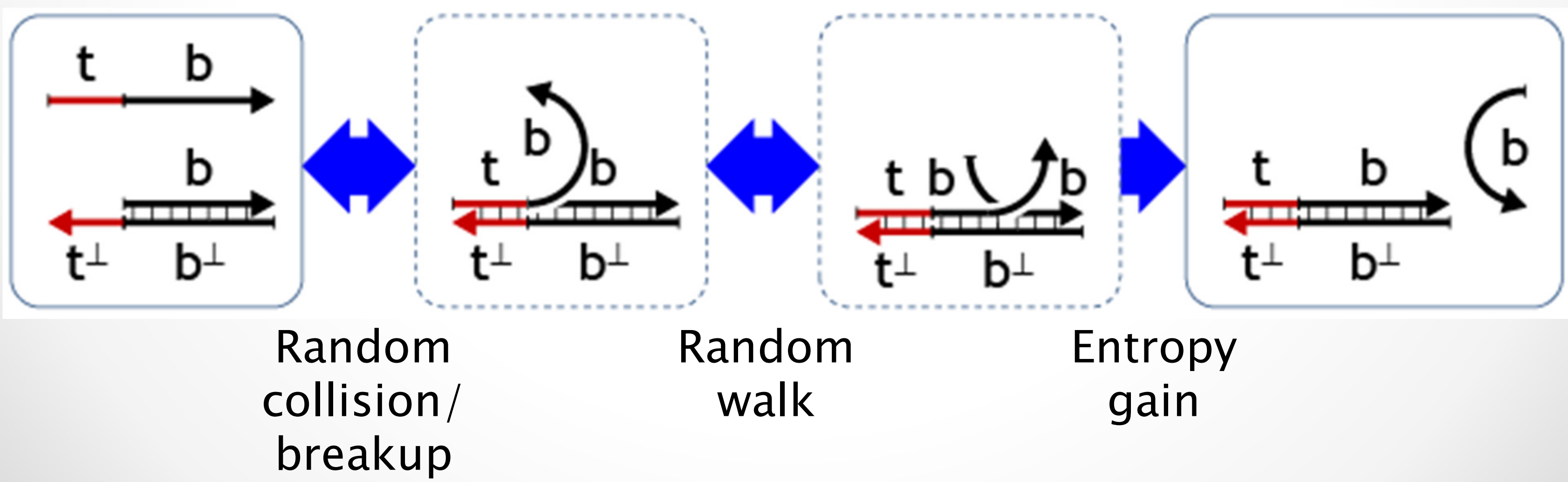
- Short complementary segments hybridize **reversibly**


- Long complementary segments hybridize **irreversibly**

# DNA Strand Displacement

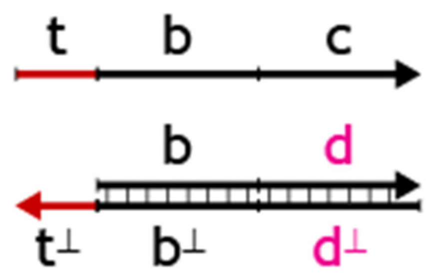
- Short strand (**toehold**): reversible binding
- Long strand (**body**): irreversible binding



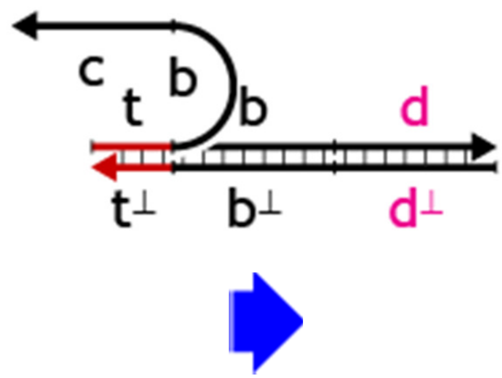



# Failed Strand Displacement

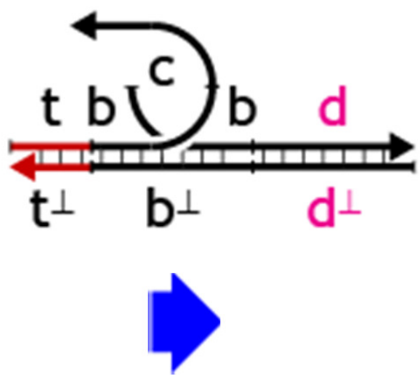
- What if the input does not match the gate?



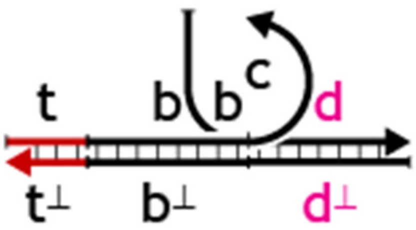
# Failed Strand Displacement



# Failed Strand Displacement

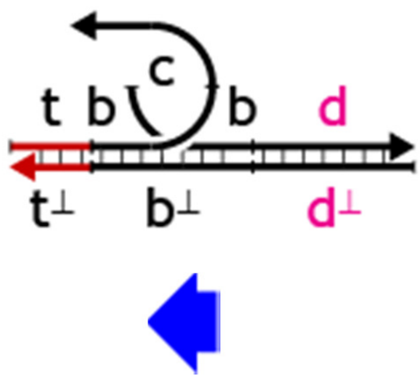


# Failed Strand Displacement

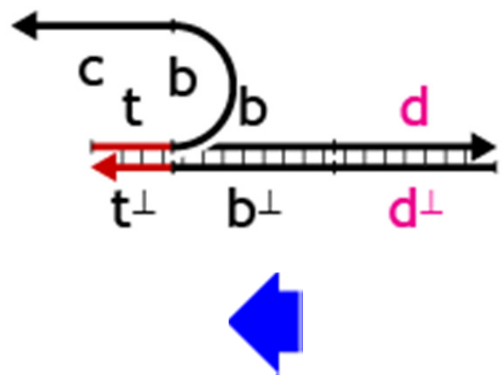



?

# Failed Strand Displacement

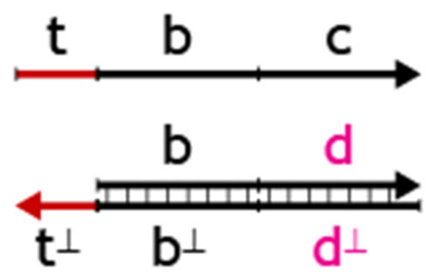


# Failed Strand Displacement

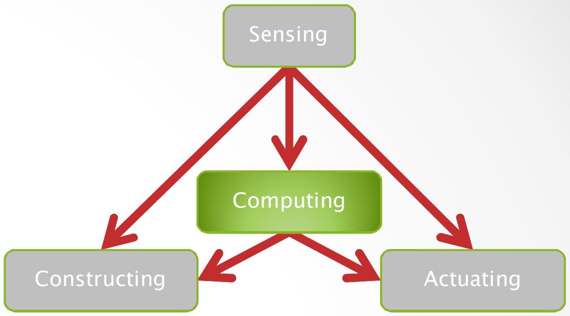


# Failed Strand Displacement

- Hence an incorrect binding will undo
  - That's why toeholds must bind reversibly




- Matching depends on the long segment only
  - Strand displacement succeeds iff the whole long segment matches
  - The address space is determined by the size of the long segment, which is unbounded (not by the size of the toehold)
  - The toehold is just a 'cache' of the address




# Computing

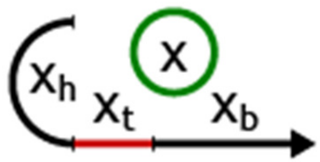
• • •

## Implementing "Arbitrary" Computing Functions

# Signals

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

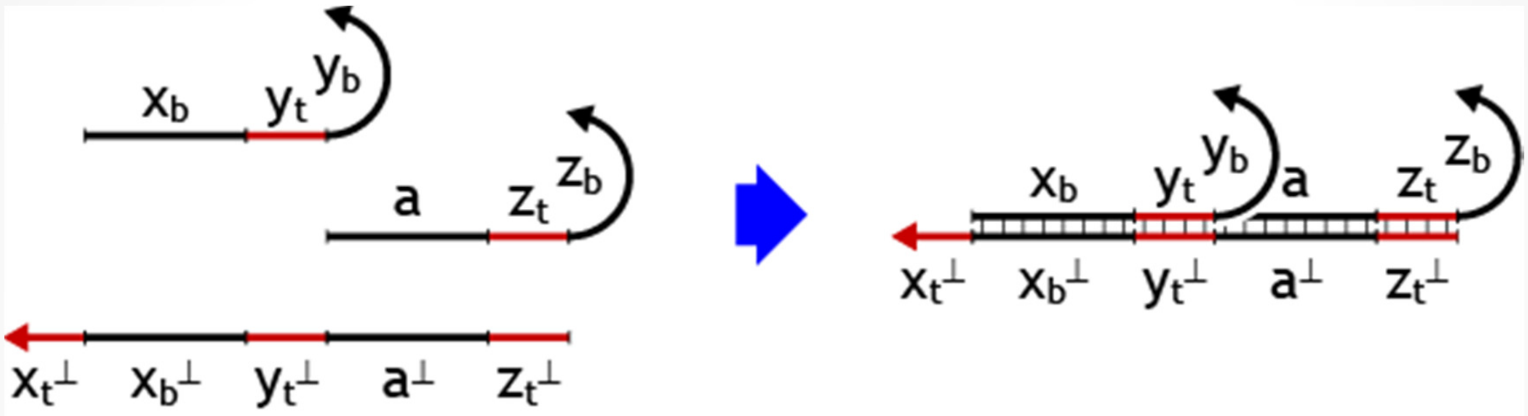
- 3-domain signals
  - $x_h$: hystory (ignore)
  - $x_t$: toehold (binding)
  - $x_b$: body (recognition)




- Signals (single stranded DNA) are prepared by (artificial) DNA synthesis

# Gates

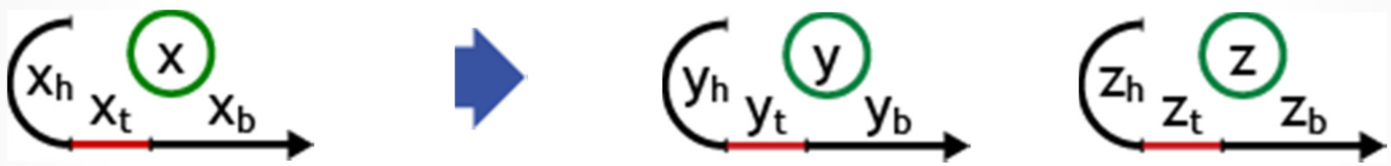
- Double-stranded structures with free toeholds




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

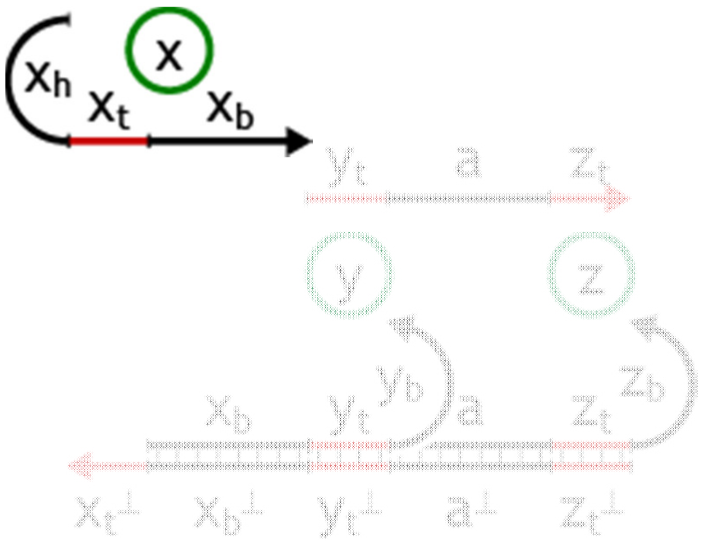
# Fork Gate

- $x \rightarrow y + z$



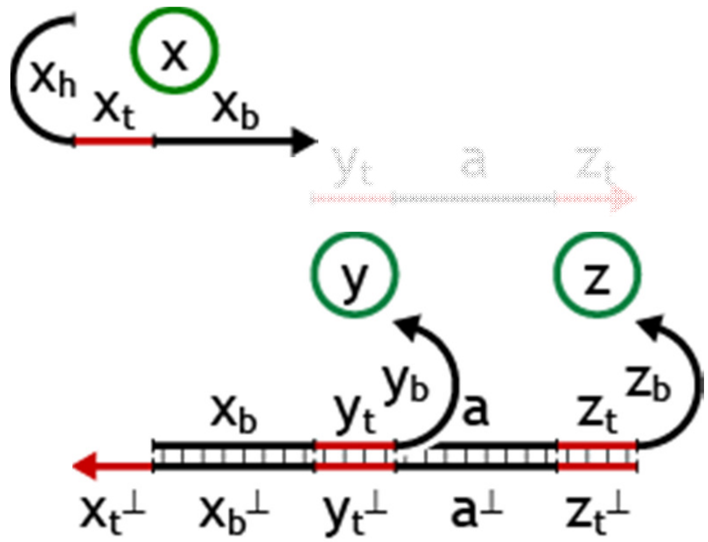

- $x \rightarrow y + 0$ transform x to y (transducer)
- $x \rightarrow x + y$ linear production of y (catalyst)
- $x \rightarrow x + x$ exponential production of x (amplifier)

# Fork Gate



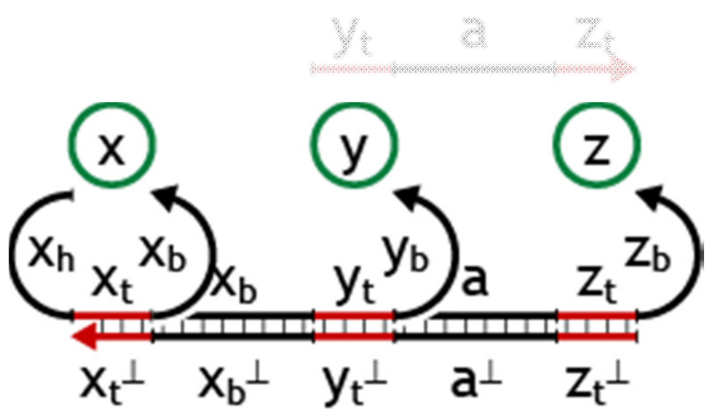

This is the Fork Gate structure

# Fork Gate

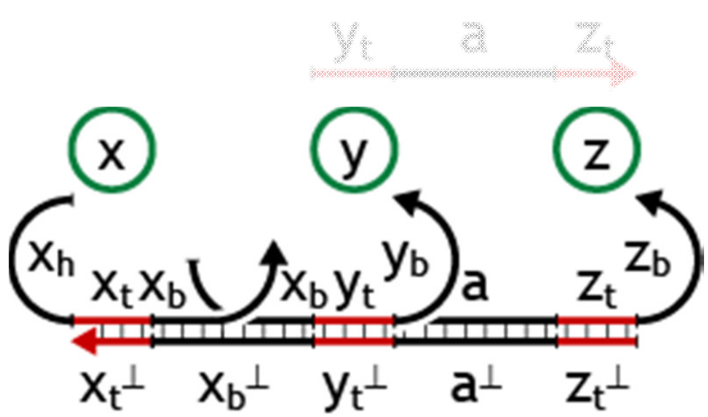


x
$x_h$
$x_t$
$x_b$
$y_t$
a
$z_t$
y
z
$y_b$
$z_b$
$x_b$
$y_t$
a
$z_t$
$x_t^\perp$
$x_b^\perp$
$y_t^\perp$
$a^\perp$
$z_t^\perp$

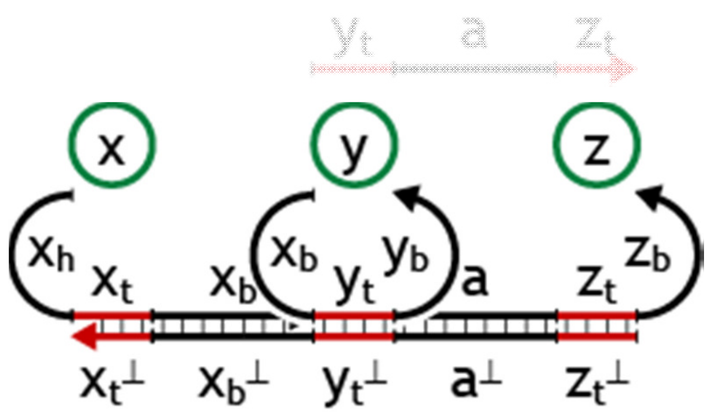
# Fork Gate



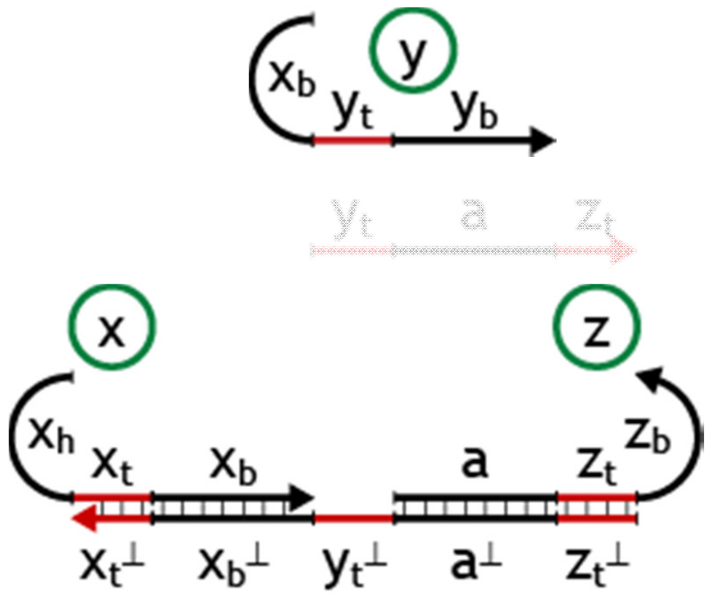
# Fork Gate



# Fork Gate

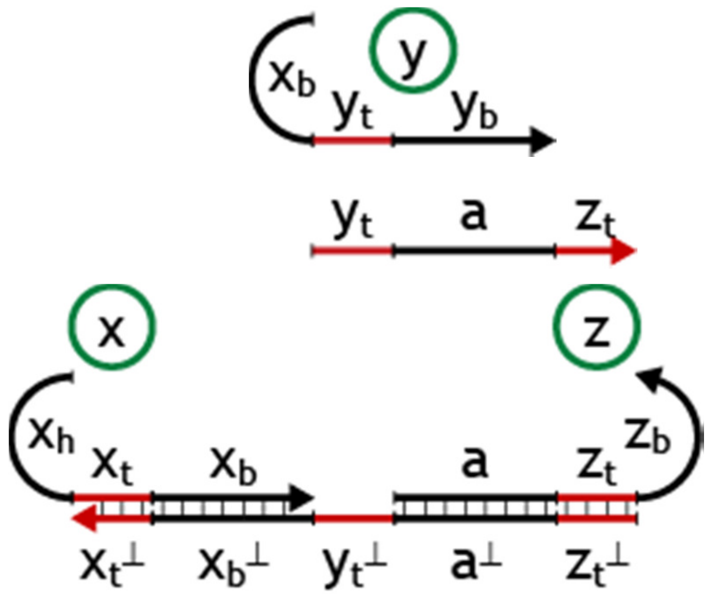


# Fork Gate

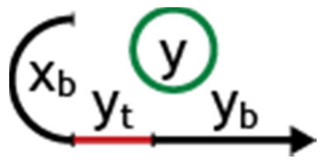


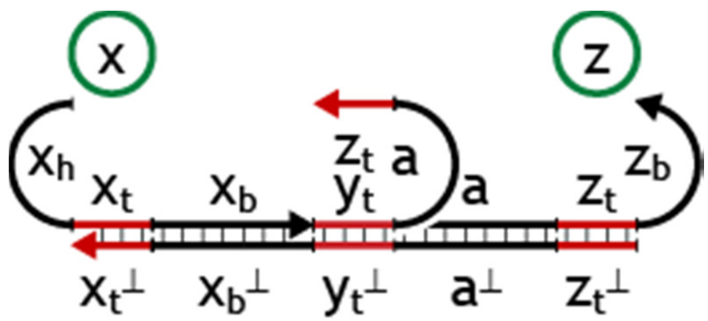
y
$x_b$
$y_t$
$y_b$
$y_t$
a
$z_t$
x
z
$x_h$
$x_t$
$x_b$
a
$z_t$
$z_b$
$x_t^\perp$
$x_b^\perp$
$y_t^\perp$
$a^\perp$
$z_t^\perp$

# Fork Gate

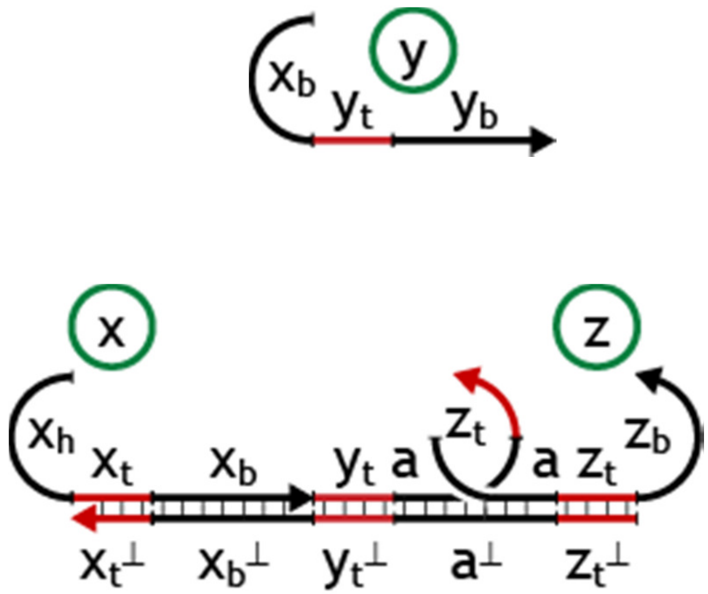


# Fork Gate



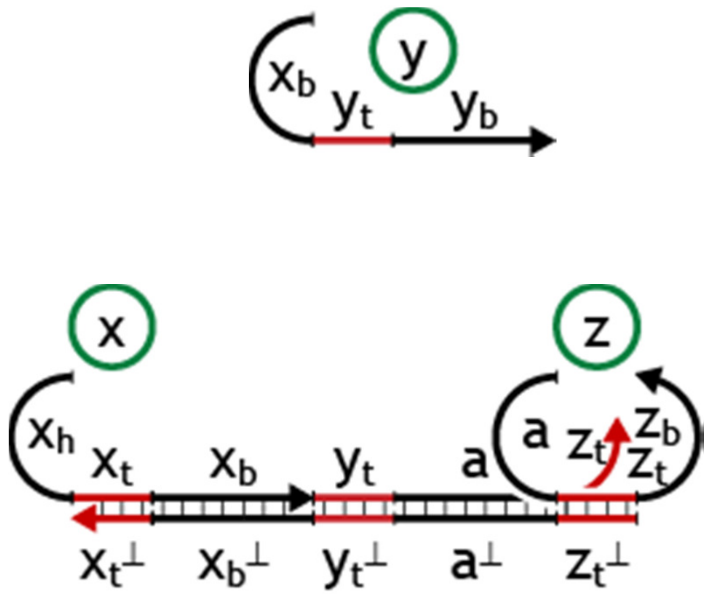



# Fork Gate

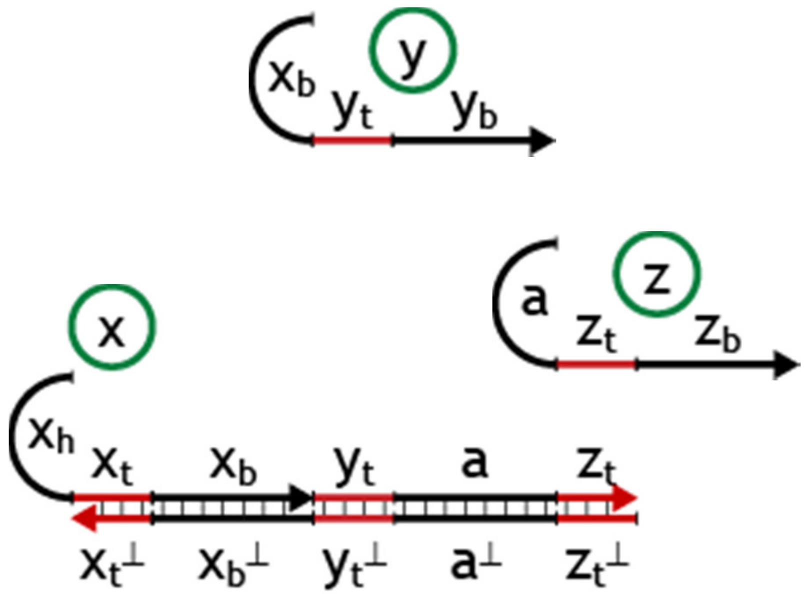


y
$x_b$
$y_t$
$y_b$
x
z
$z_t$
$x_h$
$x_t$
$x_b$
$y_t$
a
a
$z_t$
$z_b$
$x_t^\perp$
$x_b^\perp$
$y_t^\perp$
$a^\perp$
$z_t^\perp$

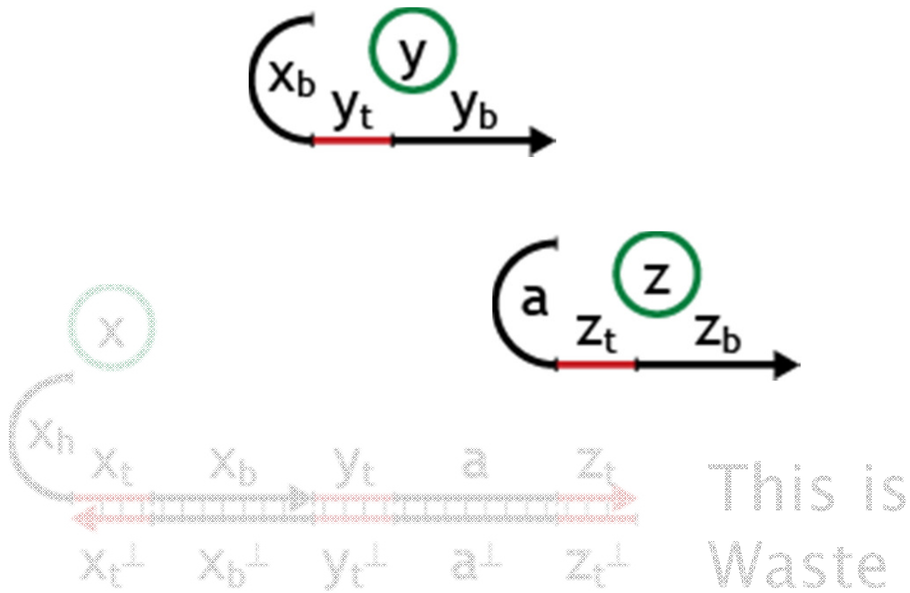
# Fork Gate



# Fork Gate



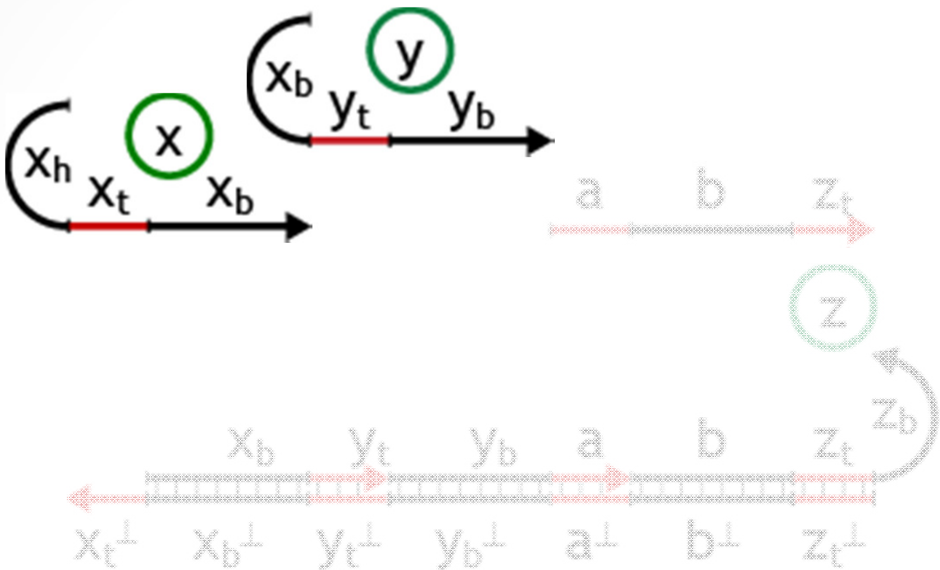
# Fork Gate



# Join Gate

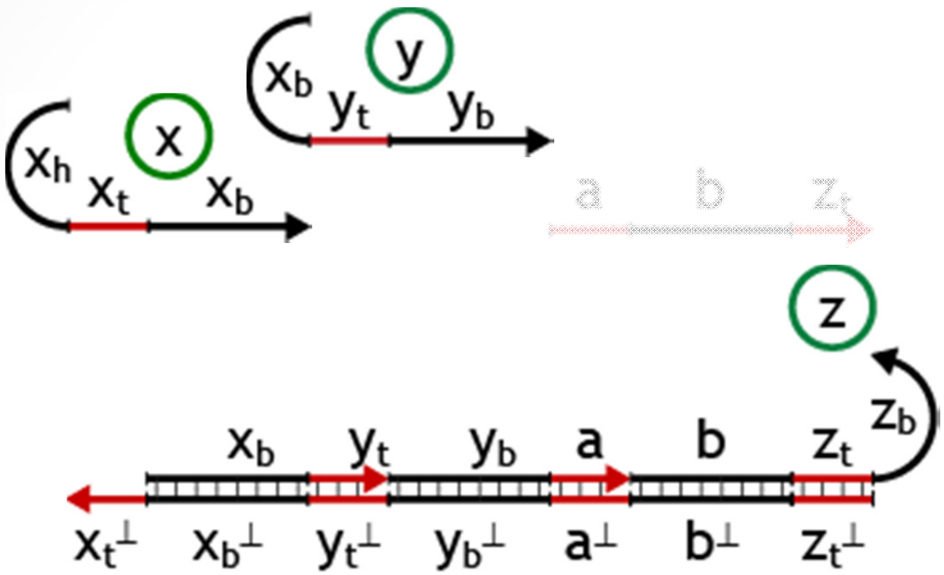
- $x + y \rightarrow z$

# Join Gate

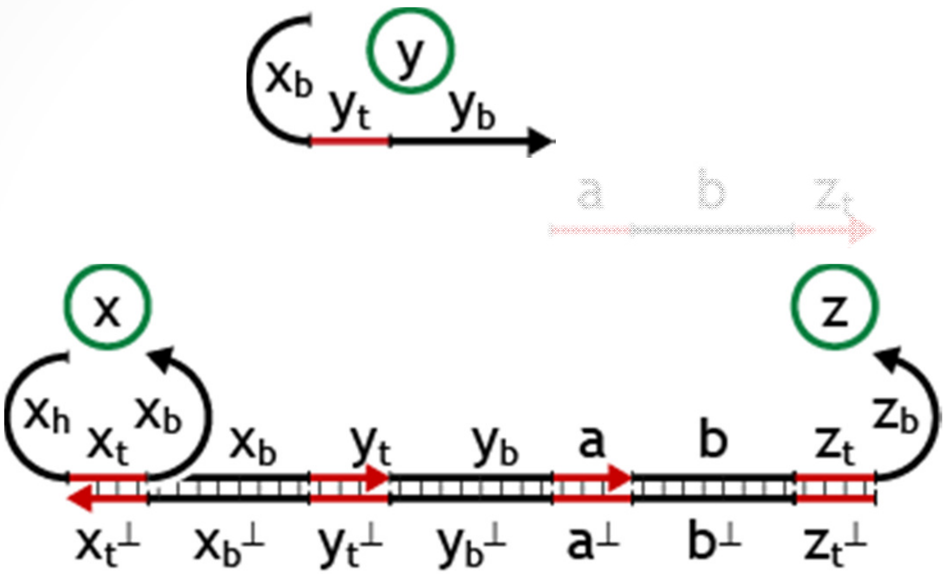



This is the Join Gate structure

# Join Gate

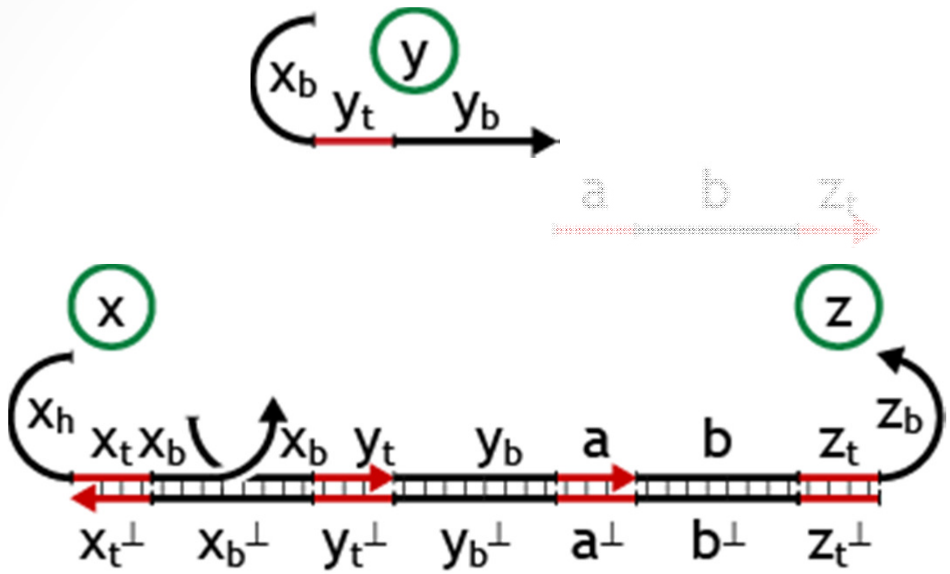


# Join Gate



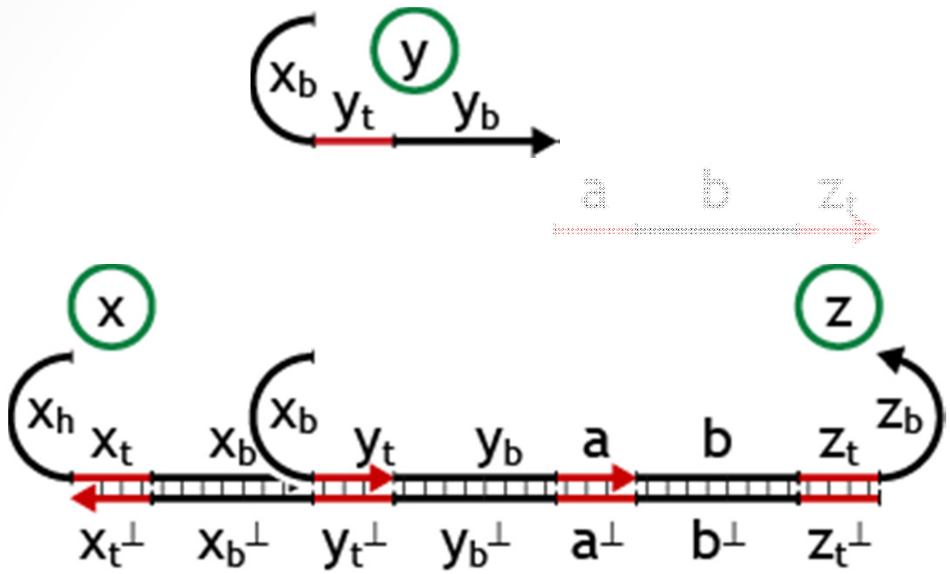
y
$x_b$
$y_t$
$y_b$
a
b
$z_t$
x
z
$x_h$
$x_t$
$x_b$
$x_b$
$y_t$
$y_b$
a
b
$z_t$
$z_b$
$x_t^\perp$
$x_b^\perp$
$y_t^\perp$
$y_b^\perp$
$a^\perp$
$b^\perp$
$z_t^\perp$

# Join Gate

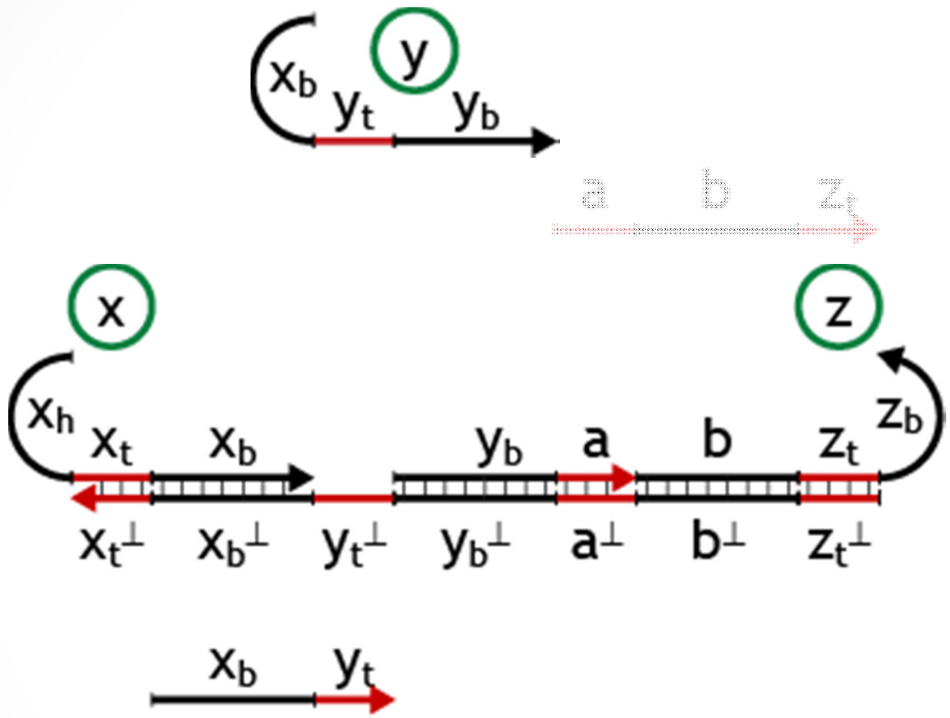


y
$x_b$
$y_t$
$y_b$
a
b
$z_t$
x
z
$x_h$
$x_t$
$x_b$
$x_b$
$y_t$
$y_b$
a
b
$z_t$
$z_b$
$x_t^\perp$
$x_b^\perp$
$y_t^\perp$
$y_b^\perp$
$a^\perp$
$b^\perp$
$z_t^\perp$

# Join Gate

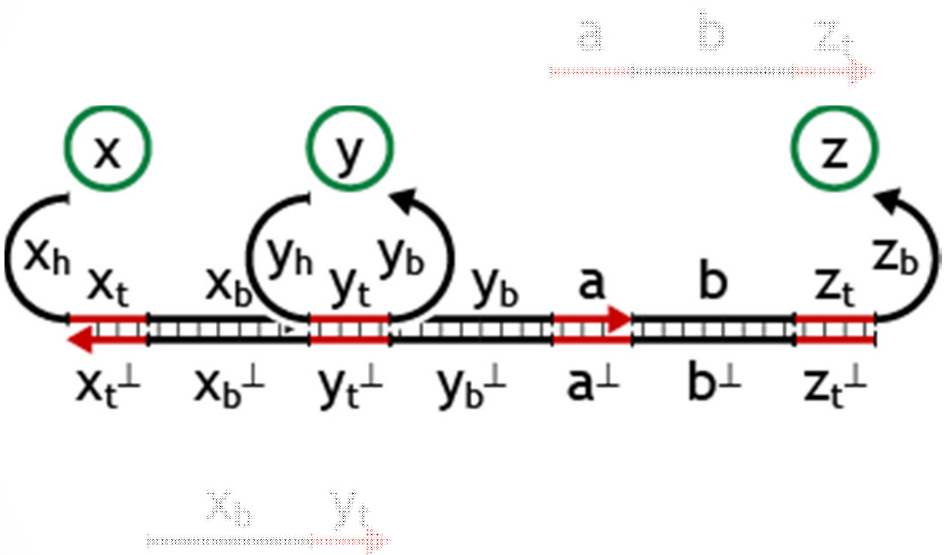


# Join Gate

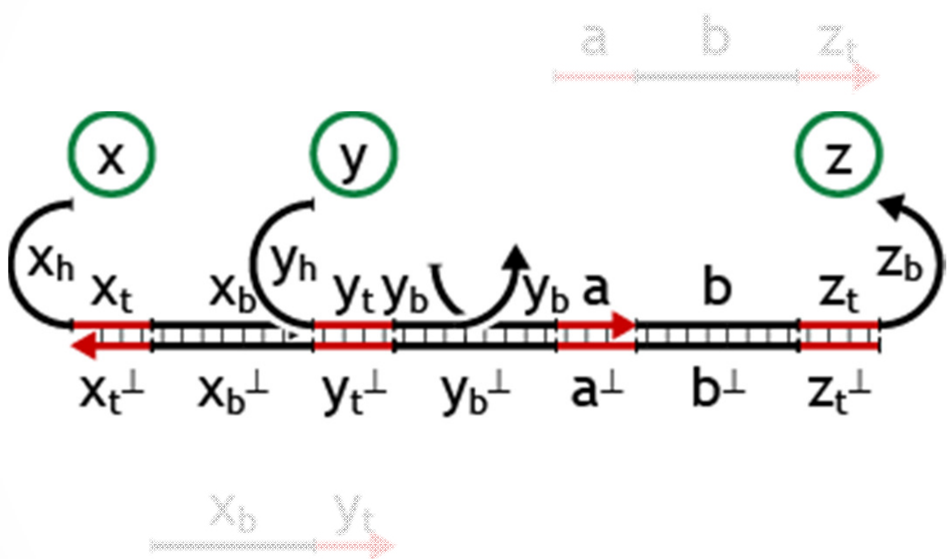


y
$x_b$ $y_t$ $y_b$
a b $z_t$
x
z
$x_h$ $x_t$ $x_b$
$y_b$ a b $z_t$ $z_b$
$x_t^\perp$ $x_b^\perp$ $y_t^\perp$ $y_b^\perp$ $a^\perp$ $b^\perp$ $z_t^\perp$
$x_b$ $y_t$

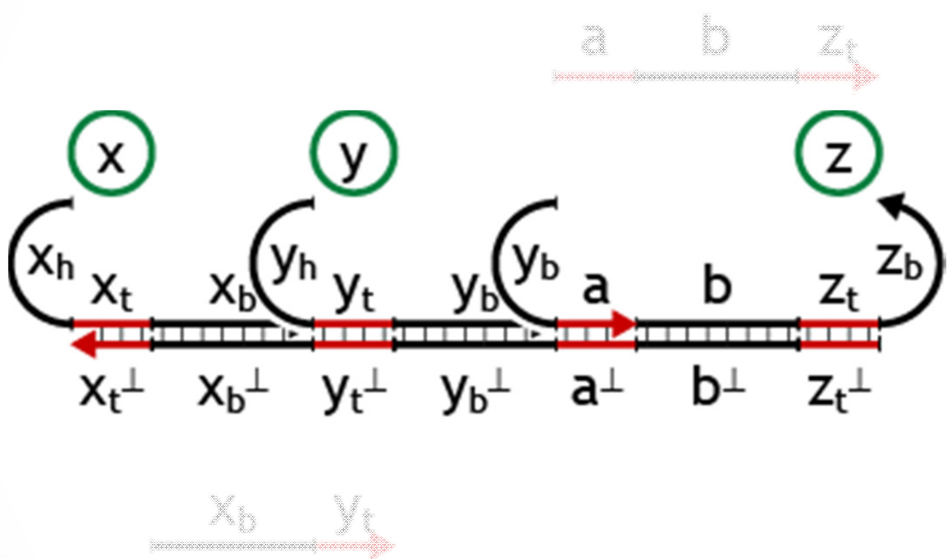
# Join Gate



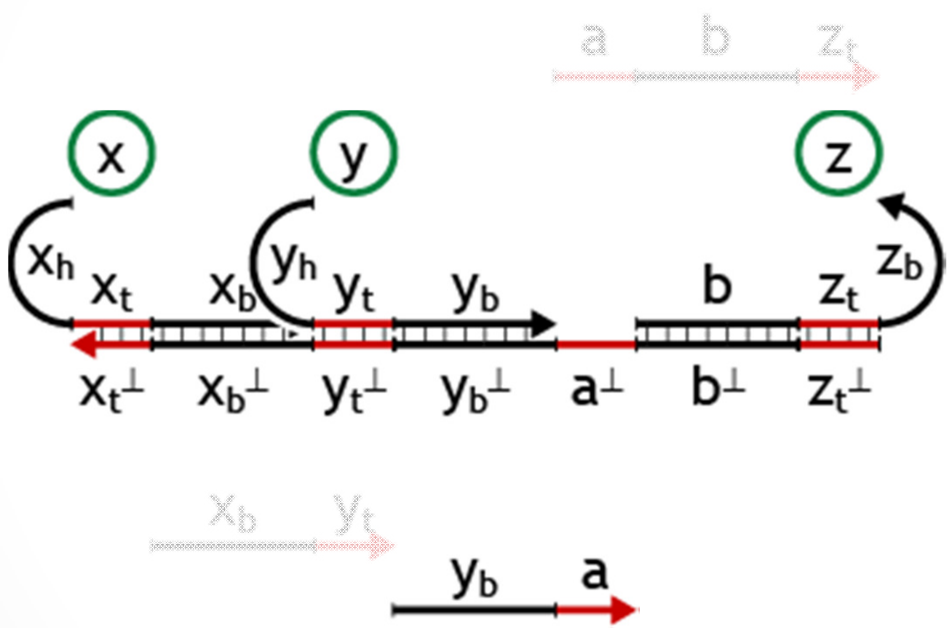
# Join Gate



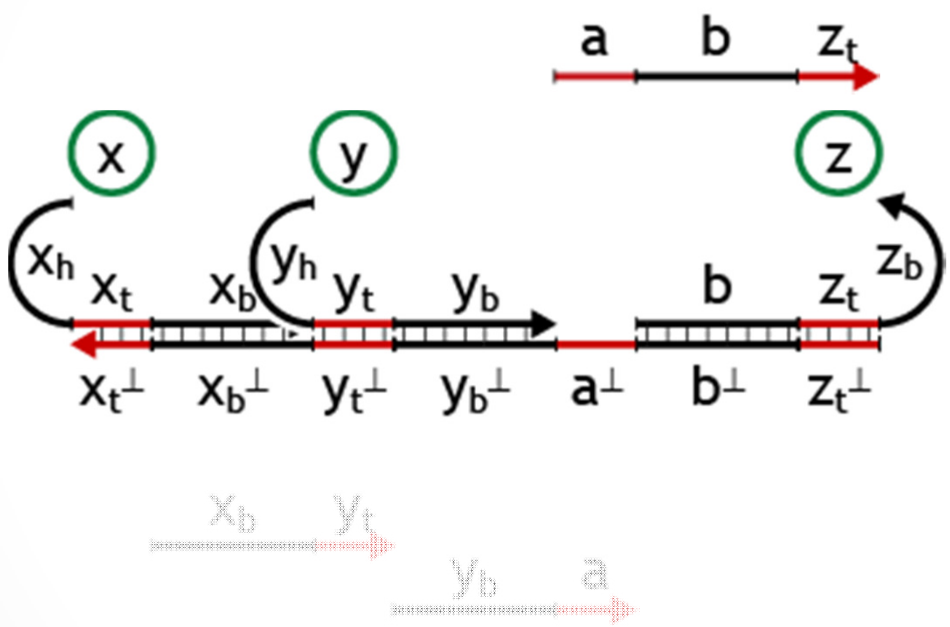
# Join Gate



# Join Gate

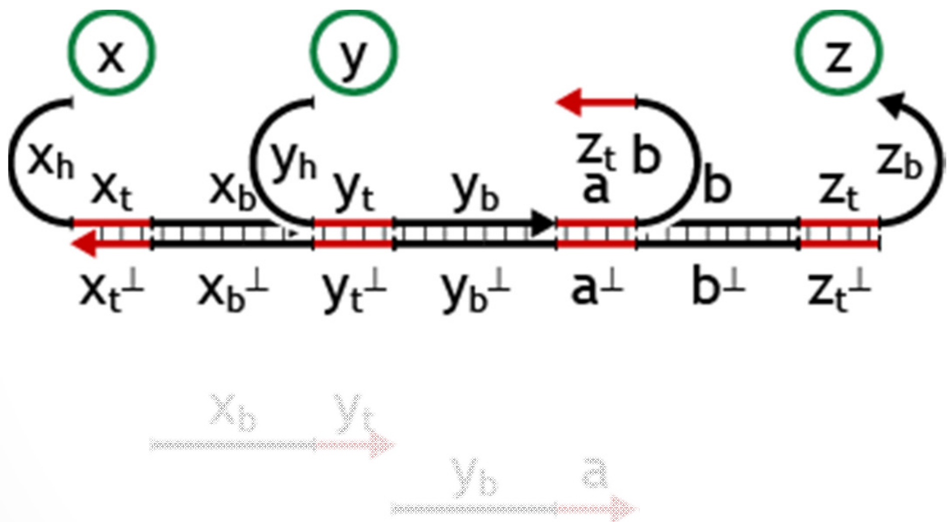


# Join Gate

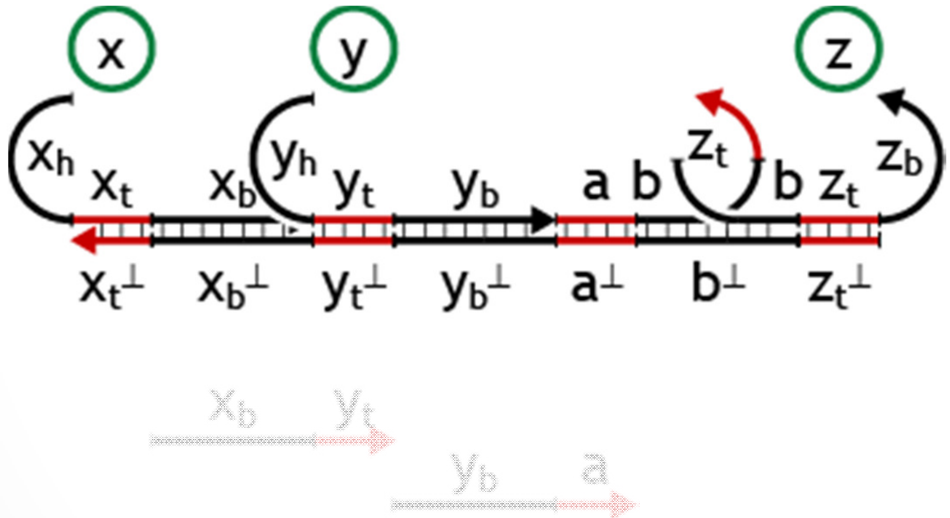


a
b
$z_t$
x
y
z
$x_h$
$x_t$
$x_b$
$y_h$
$y_t$
$y_b$
b
$z_t$
$z_b$
$x_t^{\perp}$
$x_b^{\perp}$
$y_t^{\perp}$
$y_b^{\perp}$
$a^{\perp}$
$b^{\perp}$
$z_t^{\perp}$
$x_b$
$y_t$
$y_b$
a

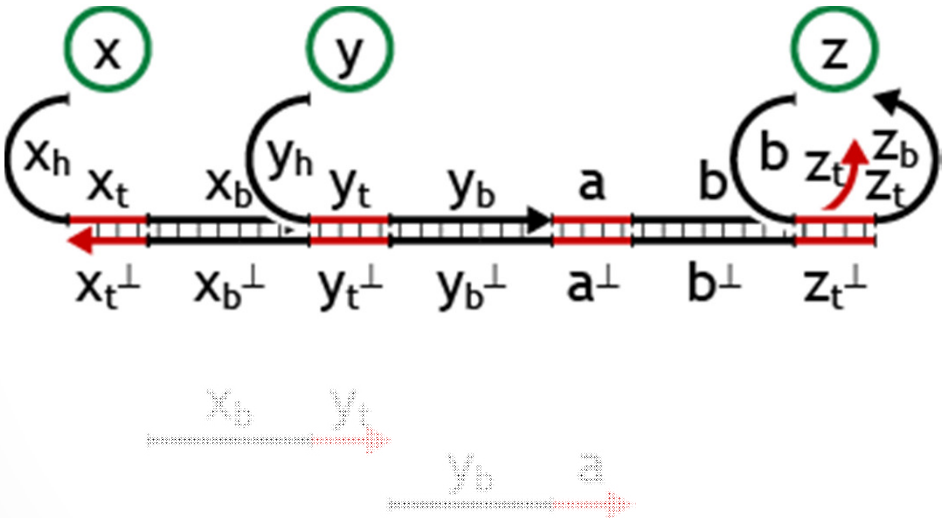
# Join Gate



# Join Gate

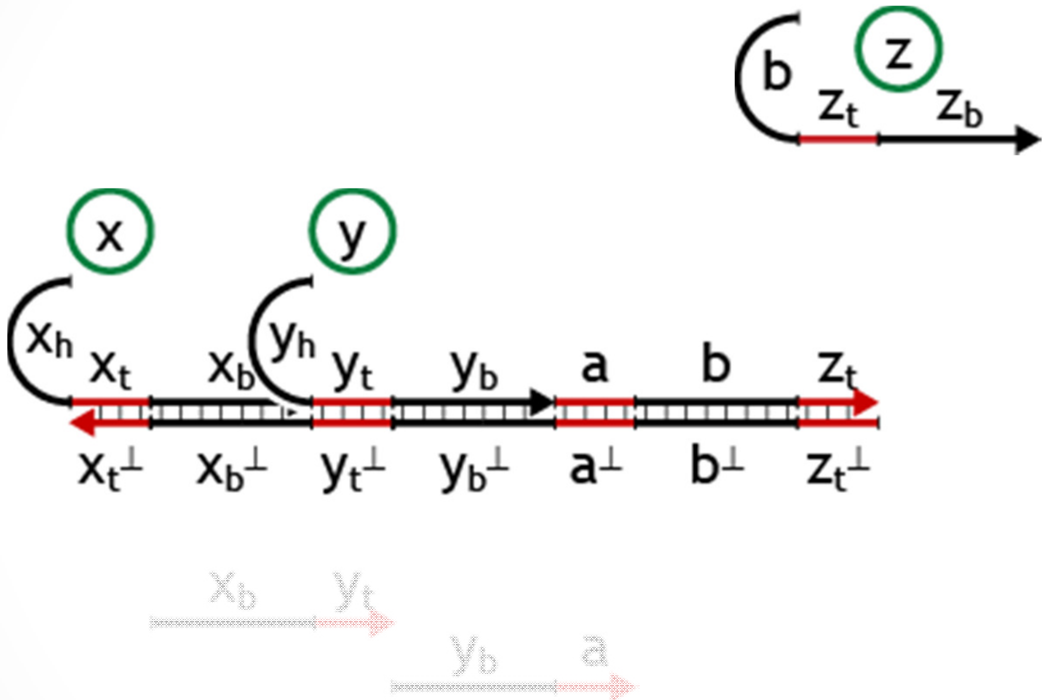

x
y
z
$x_h$
$x_t$
$x_b$
$y_h$
$y_t$
$y_b$
a b
$z_t$
b $z_t$
$z_b$
$x_t^\perp$ $x_b^\perp$ $y_t^\perp$ $y_b^\perp$ $a^\perp$ $b^\perp$ $z_t^\perp$
$x_b$ $y_t$
$y_b$ a

# Join Gate

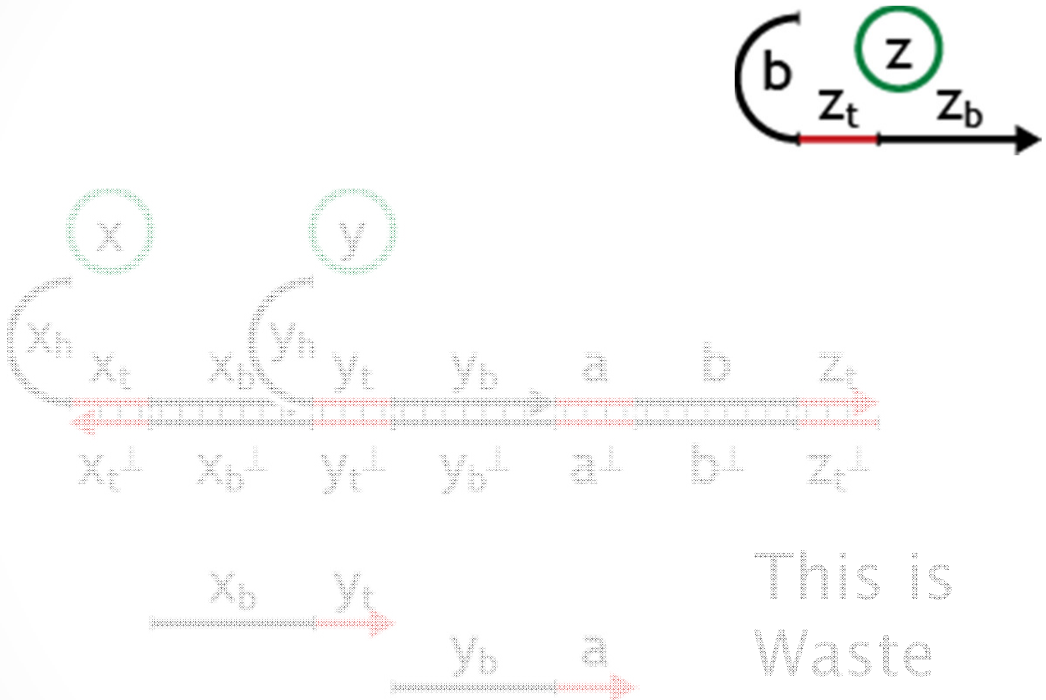


x
y
z
$x_h$ $x_t$ $x_b$ $y_h$ $y_t$ $y_b$ a b b $z_t$ $z_b$ $z_t$
$x_t^\perp$ $x_b^\perp$ $y_t^\perp$ $y_b^\perp$ $a^\perp$ $b^\perp$ $z_t^\perp$
$x_b$ $y_t$
$y_b$ a

# Join Gate

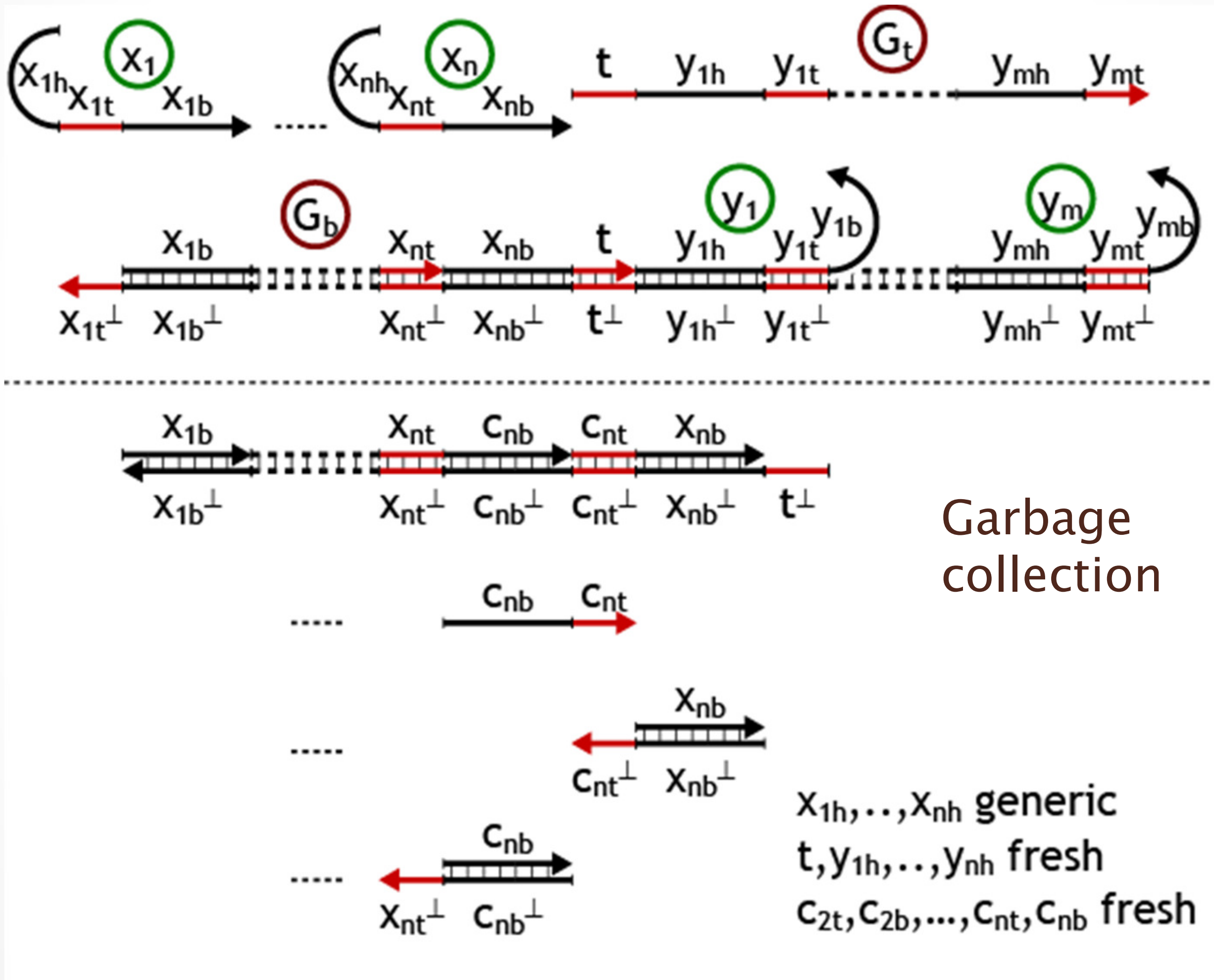

b
z
$z_t$
$z_b$
x
y
$x_h$
$x_t$
$x_b$
$y_h$
$y_t$
$y_b$
a
b
$z_t$
$x_t^{\perp}$
$x_b^{\perp}$
$y_t^{\perp}$
$y_b^{\perp}$
$a^{\perp}$
$b^{\perp}$
$z_t^{\perp}$
$x_b$
$y_t$
$y_b$
a

# Join Gate



z
b
zt
zb
x
y
xh
yh
xt
xb
yt
yb
a
b
zt
xt⊥
xb⊥
yt⊥
yb⊥
a⊥
b⊥
zt⊥
xb
yt
yb
a
This is Waste

# General n-Join/m-Fork Gate



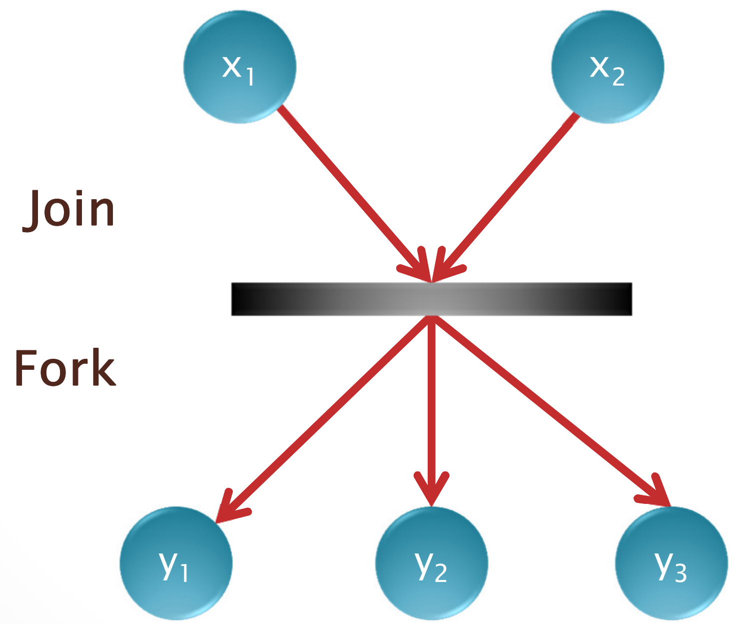
# Gate Design Verification

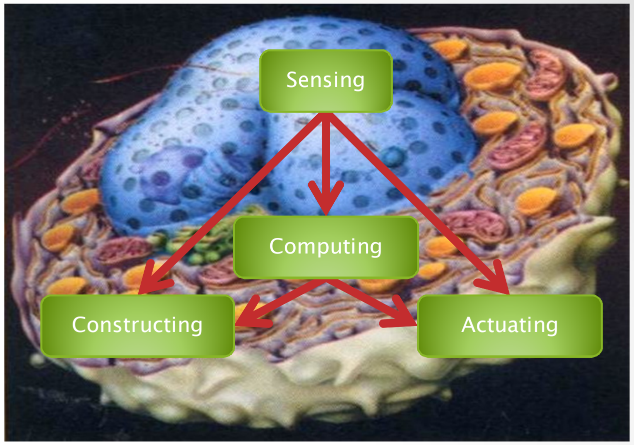
- Active garbage
  - The active join residuals slow down the performance of following joins.
  - → Add a garbage collector to remove the active residuals.
- Interference between gates
  - The join garbage collector interferes with the fork gate.
  - → Modify the fork gate to remove the interference.
- What else could go wrong?
  - Endless possibilities.
  - → Prove that the fork/join gate structures correctly implement fork/join in all larger circuits.

# Strand Algebra

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

- Join + Fork + Populations = (Stochastic) Petri Nets






# Curing

• • •

# A Doctor in Each Cell

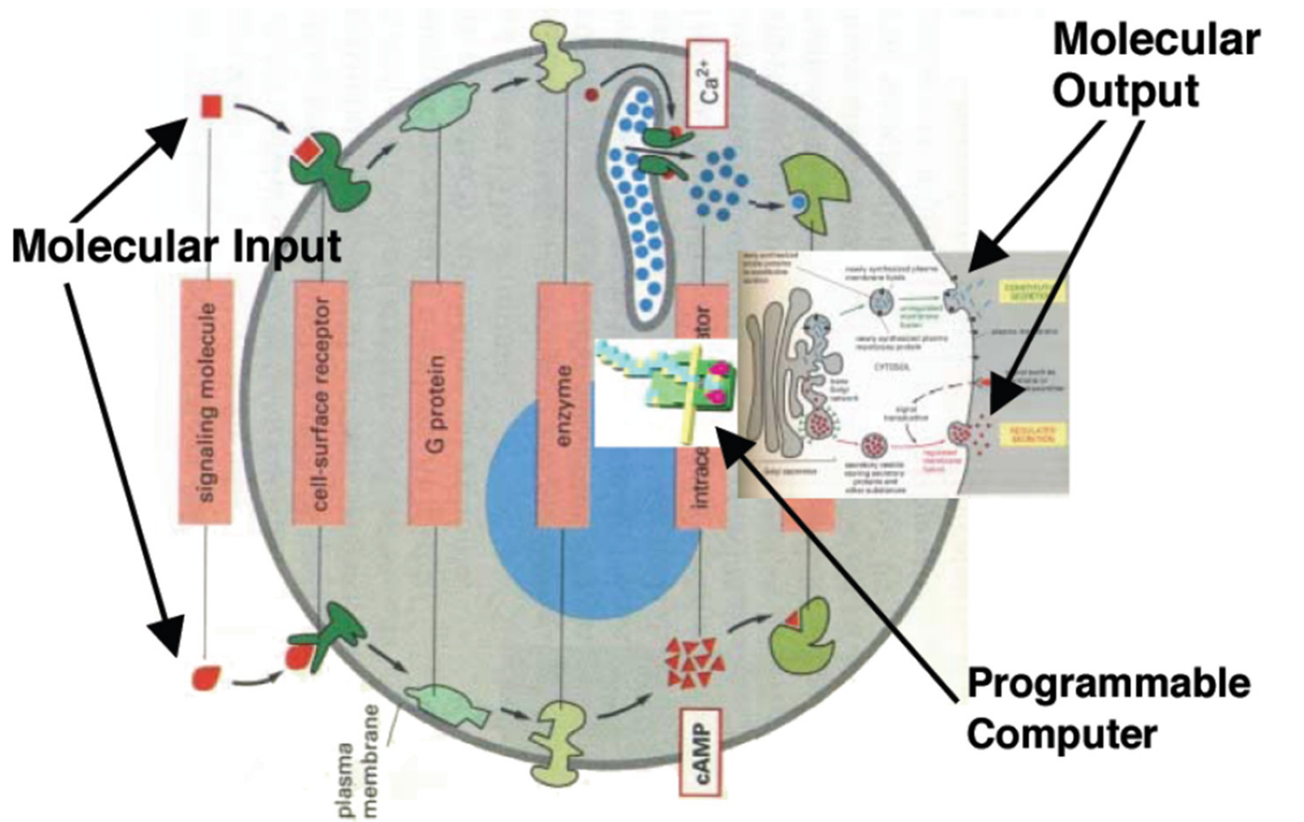


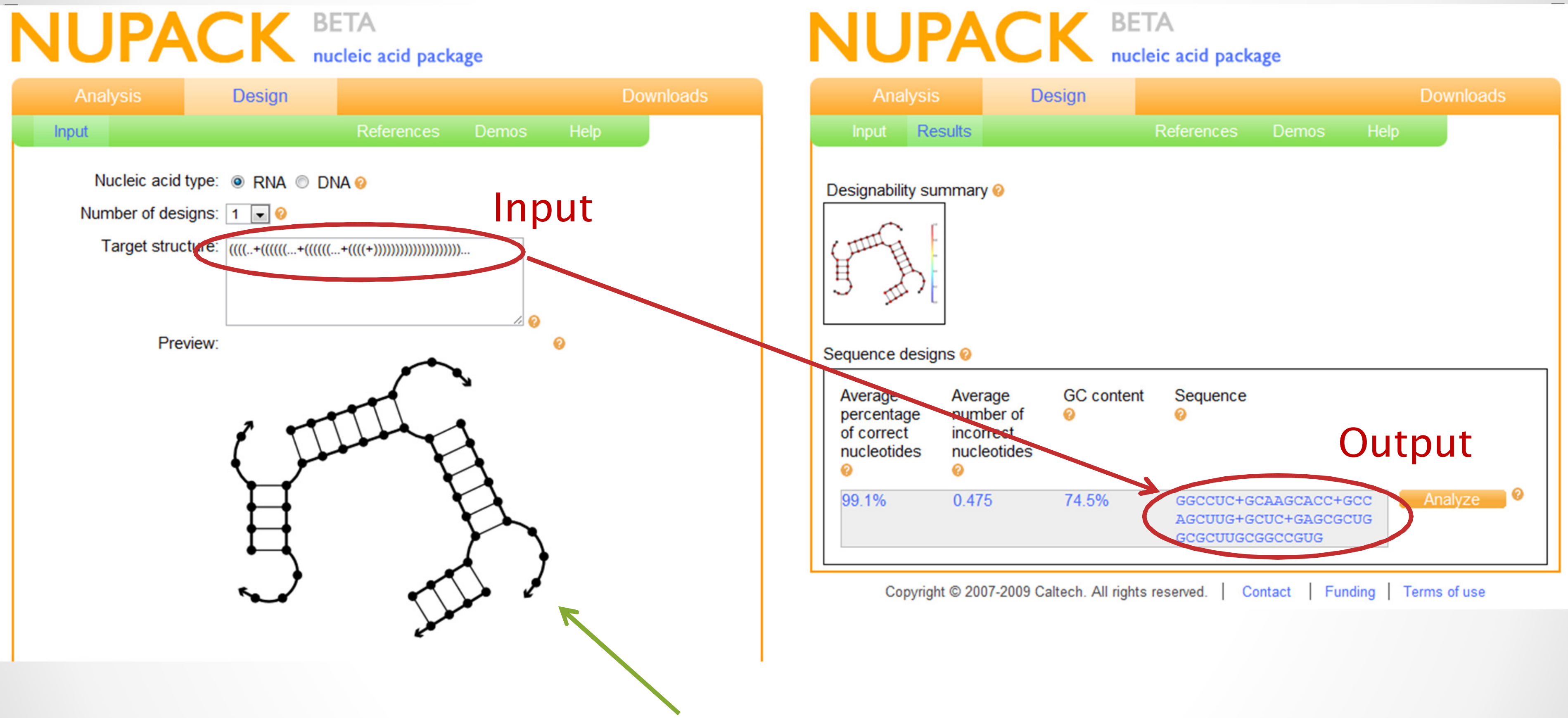

*Fig. 1* *Medicine in 2050: "Doctor in a Cell"*


Ehud Shapiro
Rivka Adar
Kobi Benenson
Gregory Linshitz
Aviv Regev
William Silverman


**Molecules and computation**

# Tools

# Sequence Design




So we can in principle work at this level.

# Visual DSD

## A Strand Displacement Simulator

Matthew Lakin, Simon Youssef, Andrew Phillips

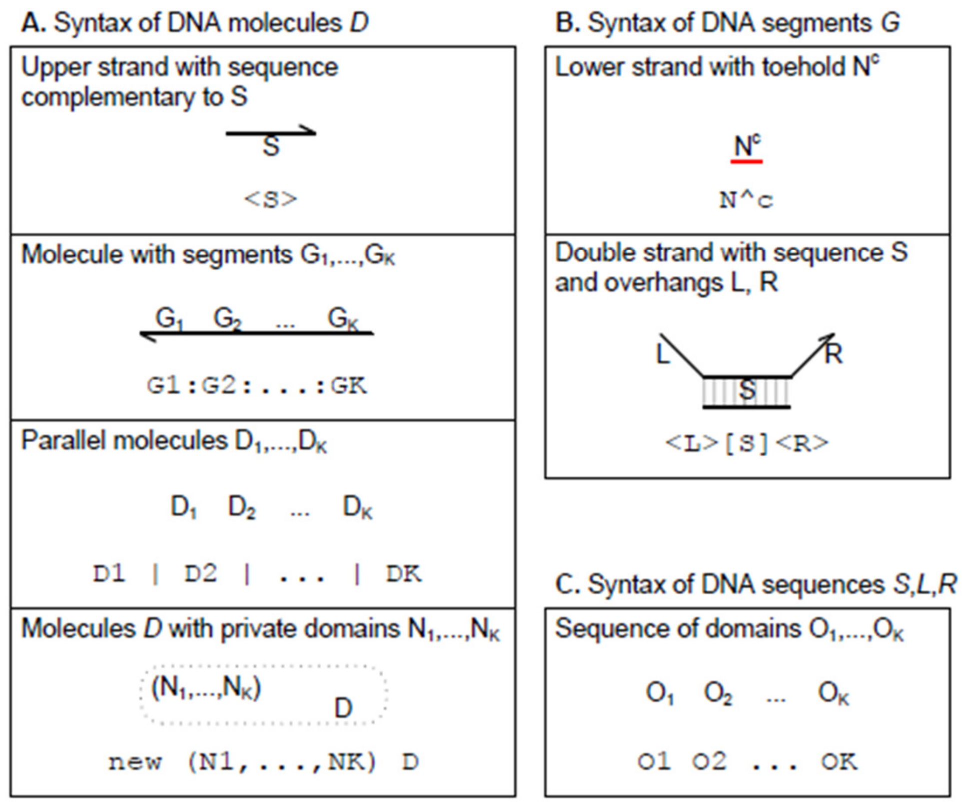
http://lepton.research.microsoft.com/webdna/

# Syntax

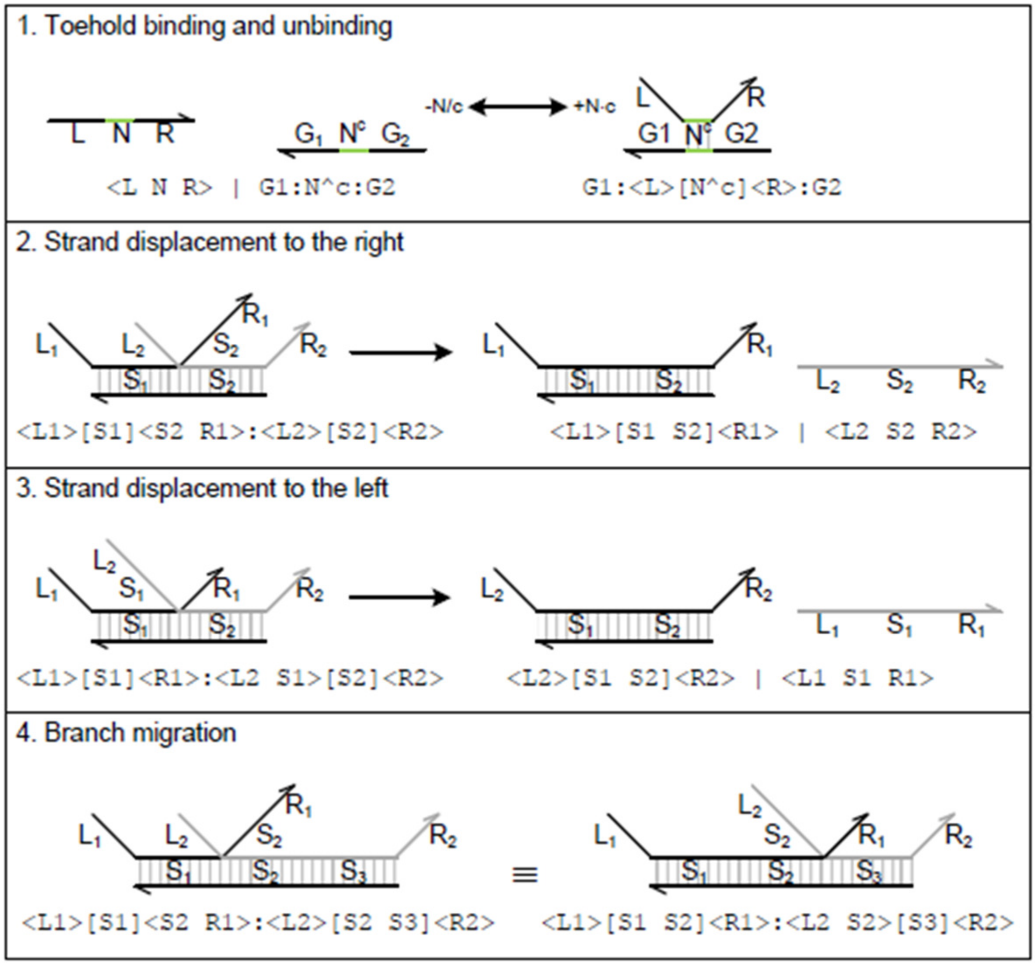
Interface focus | IET | FirstCite e-publishing
J. R. Soc. Interface
doi:10.1098/rsif.2009.0072.focus
Published online

## A programming language for composable DNA circuits

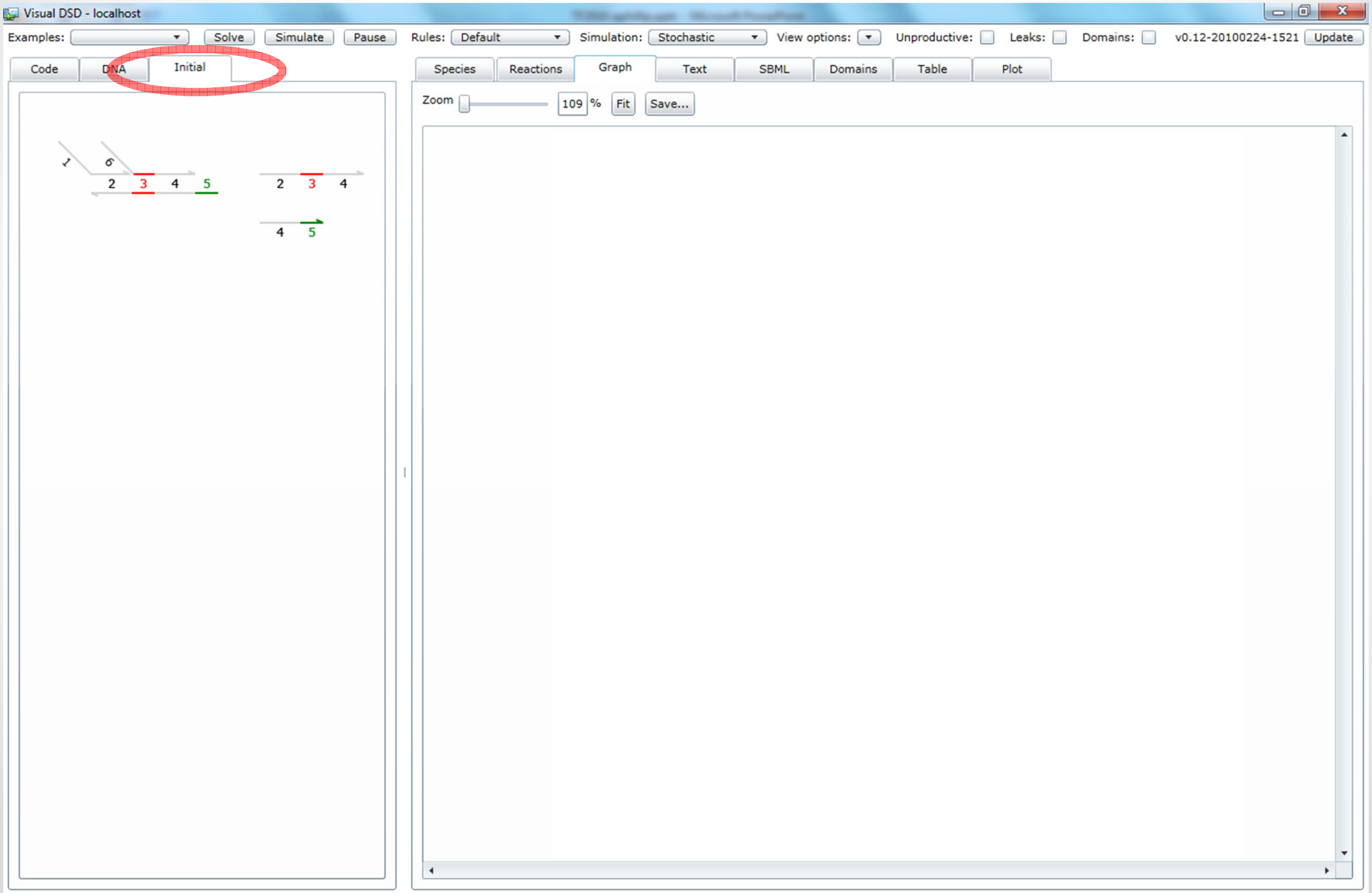
Andrew Phillips* and Luca Cardelli



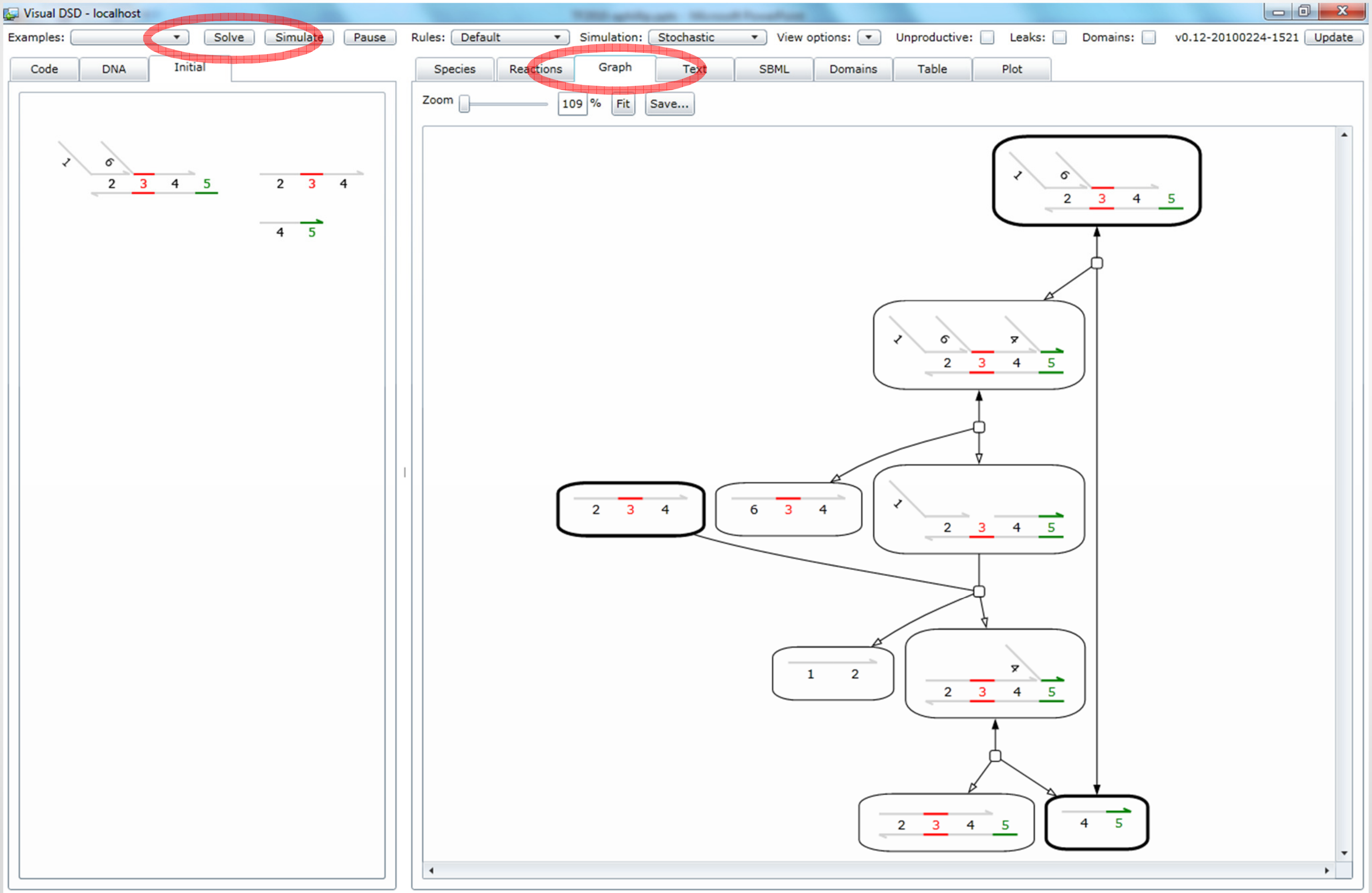
# Dynamics


1. Toehold binding and unbinding
L N R
G1 N^c G2
-N/c
+N·c
L R
G1 N^c G2
<L N R> | G1:N^c:G2
G1:<L>[N^c]<R>:G2
2. Strand displacement to the right
L1 L2 S2 R1 R2 S1 S2
L1 R1 S1 S2
L2 S2 R2
<L1>[S1]<S2 R1>:<L2>[S2]<R2>
<L1>[S1 S2]<R1> | <L2 S2 R2>
3. Strand displacement to the left
L2 S1 L1 R1 R2 S1 S2
L2 R2 S1 S2
L1 S1 R1
<L1>[S1]<R1>:<L2 S1>[S2]<R2>
<L2>[S1 S2]<R2> | <L1 S1 R1>
4. Branch migration
L1 L2 S2 R1 R2 S1 S2 S3
≡
L1 L2 S2 R1 R2 S1 S2 S3
<L1>[S1]<S2 R1>:<L2>[S2 S3]<R2>
<L1>[S1 S2]<R1>:<L2 S2>[S3]<R2>

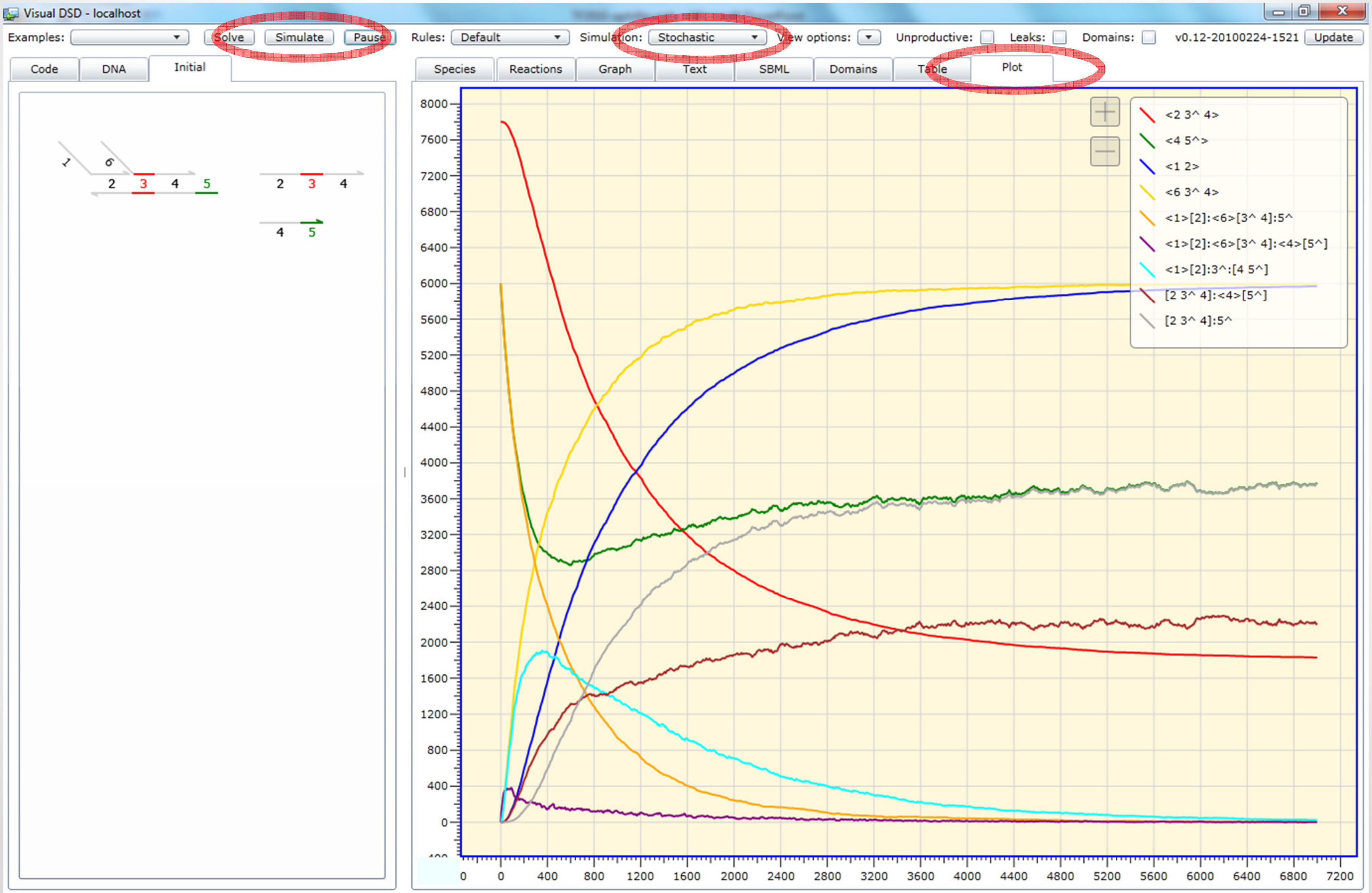
# Initial Species



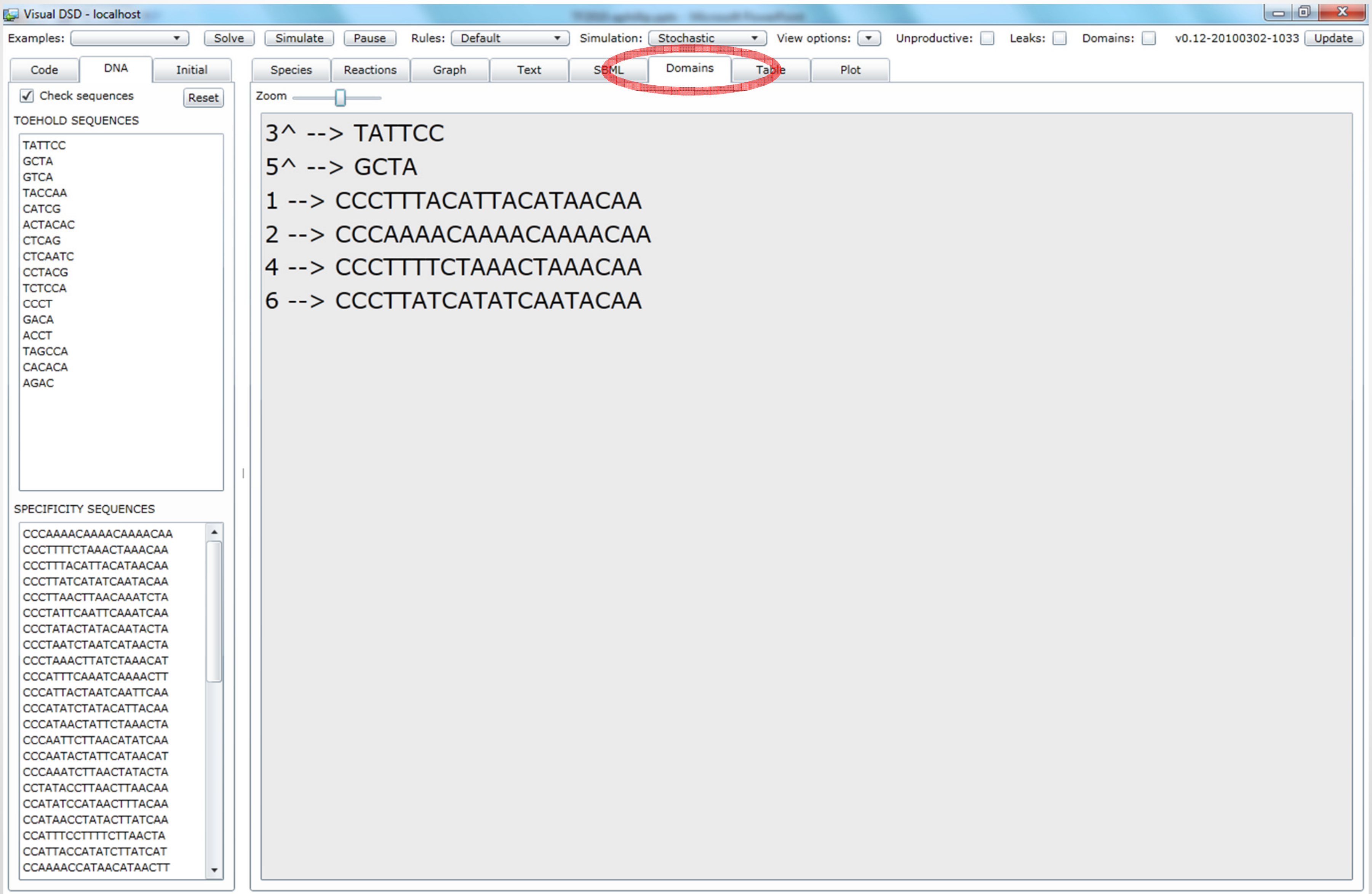
# Reaction Graph



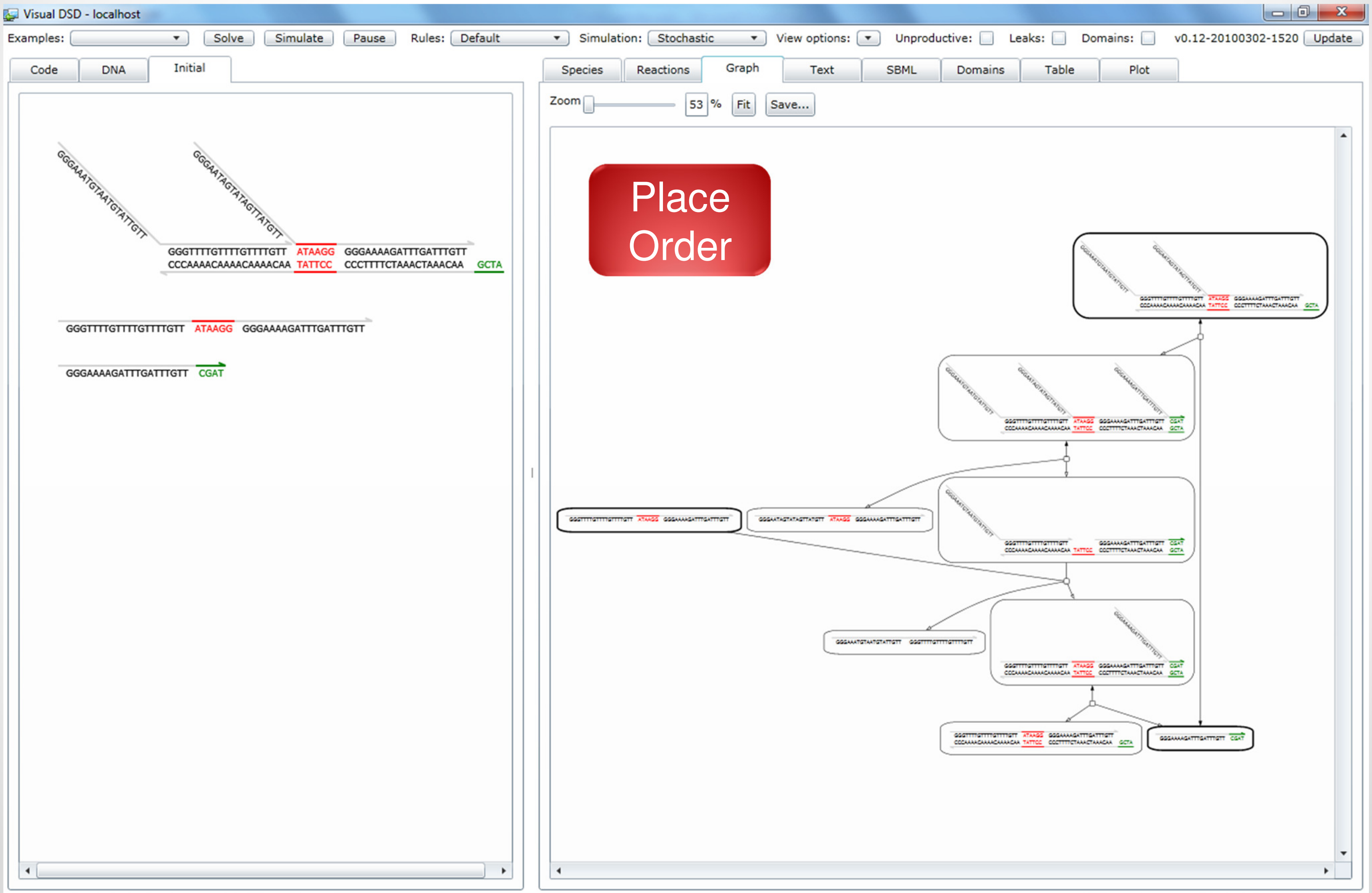
# Simulation


Visual DSD - localhost
Examples:
Solve
Simulate
Pause
Rules:
Default
Simulation:
Stochastic
View options:
Unproductive:
Leaks:
Domains:
v0.12-20100224-1521
Update
Code
DNA
Initial
Species
Reactions
Graph
Text
SBML
Domains
Table
Plot
<2 3^ 4>
<4 5^>
<1 2>
<6 3^ 4>
<1>[2]:<6>[3^ 4]:5^
<1>[2]:<6>[3^ 4]:<4>[5^]
<1>[2]:3^:[4 5^]
[2 3^ 4]:<4>[5^]
[2 3^ 4]:5^

# DNA Sequences



# Final DNA Circuit



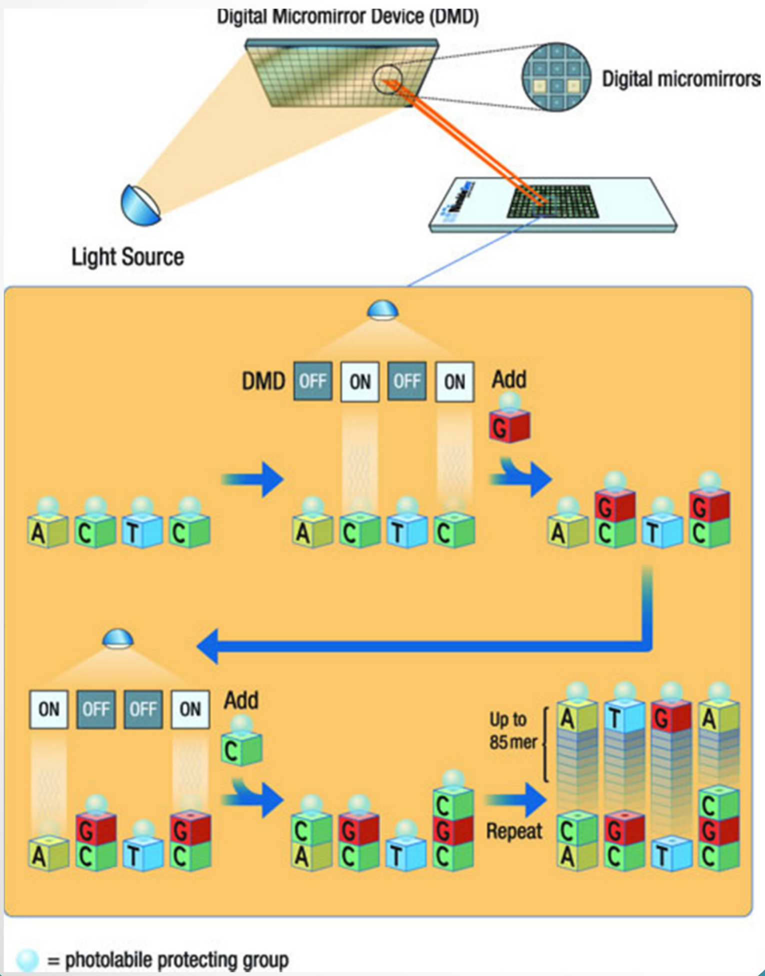
# Experiments

• • •

# How are they Actually Done?

# Sequences to DNA

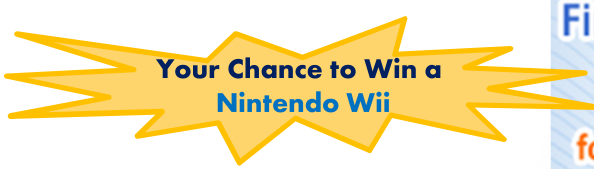
## Oligonucleotide Synthesis






Fastest tunaround time for less money!

**Standard gene synthesis from 0.36 €/bp in just 8 days**





**SameDay® Oligo Service**

**Only £0.57 GBP / Base!**

**Base Pricing**

| Synthesis Scale | Price | |
|---|---|---|
| 25 nmole DNA Oligo | £0.25 GBP / Base | Order |
| 100 nmole DNA oligo | £0.45 GBP / Base | Order |
| 250 nmole DNA oligo | £0.80 GBP / Base | Order |
| 1 µmole DNA oligo | £1.60 GBP / Base | Order |
| 5 µmole DNA oligo | £7.50 GBP / Base | Order |
| 10 µmole DNA oligo | £14.50 GBP / Base | Order |

**Custom DNA/RNA**
Pricing (USD)

| DNA(mg) | Desalted | Purified |
|---|---|---|
| 15 | $700 | $1,050 |
| 50 | $1,200 | $1,450 |
| 100 | $1,500 | $1,800 |
| 250 | $2,000 | $2,400 |
| 500 | $2,900 | $3,400 |
| 1000 | $4,550 | $5,400 |
| 5000 | $9,000 | $10,700 |

| RNA(mg) | Desalted | Purified |
|---|---|---|
| 5 | $1,500 | $1,925 |
| 15 | $1,950 | $2,490 |
| 50 | $2.050 | $2,625 |
| 100 | $2,575 | $3,575 |
| 250 | $4,575 | $5,725 |
| 500 | $7,900 | $9,190 |
| 1000 | $13,900 | $15,900 |
| 5000 | | $37,125 |

Please inquire for larger quantities

# Next-Day Oligos!


| | | | | |
|---|---|---|---|---|
| #1 | Gb_bot | | | $37.95 |
| Product: | 250 nmole DNA oligo | Usually Ships In: | 1 business day **Length:** 69 | |
| Purification: | Standard Desalting | Guaranteed Yield: | 15 ODs = 20.9 nmoles = 456.8 µgrams | |
| Sequence: | 5'- GAG GAG GTT GTG AAG TAA TGC GTG AGA TGT GAT TGT GTT ATG GTG AGG GTA AA A GGT GAA TTGGAG GAG -3' | | | |
| #2 | B tx | | | $11.55 |
| Product: | 250 nmole DNA oligo | Usually Ships In: | 1 business day **Length:** 21 | |
| Purification: | Standard Desalting | Guaranteed Yield: | 15 ODs = 81.1 nmoles = 505.9 µgrams | |
| Sequence: | 5'- CAA TTC ACC TTT TAC CCT CAC -3' | | | |
| #3 | x tb | | | $11.55 |
| Product: | 250 nmole DNA oligo | Usually Ships In: | 1 business day **Length:** 21 | |
| Purification: | Standard Desalting | Guaranteed Yield: | 15 ODs = 73.3 nmoles = 460.9 µgrams | |
| Sequence: | 5'- CAT AAC ACA ATC ACA TCT CAC -3' | | | |
| #4 | b tb B | | | $19.80 |
| Product: | 250 nmole DNA oligo | Usually Ships In: | 1 business day **Length:** 36 | |
| Purification: | Standard Desalting | Guaranteed Yield: | 15 ODs = 46.5 nmoles = 501.8 µgrams | |
| Sequence: | 5'- GCA TTA CTT CAC AAC CTC CTC CAA TTC ACC TTT TAC-3' | | | |
| #5 | B | | | $8.25 |
| Product: | 250 nmole DNA oligo | Usually Ships In: | 1 business day **Length:** 15 | |
| Purification: | Standard Desalting | Guaranteed Yield: | 10 ODs = 73.7 nmoles = 329.1 µgrams | |
| Sequence: | 5'- CAA TTC ACC TTT TAC-3' | | | |
| #6 | GB_bot | | | $11.55 |
| Product: | 250 nmole DNA oligo | Usually Ships In: | 1 business day **Length:** 21 | |
| Purification: | Standard Desalting | Guaranteed Yield: | 15 ODs = 67.3 nmoles = 444.9 µgrams | |
| Sequence: | 5'- GTG AGG GTA AAA GGT GAA TTG-3' | | | |
| #7 | Gt | | | $14.85 |
| Product: | 250 nmole DNA oligo | Usually Ships In: | 1 business day **Length:** 27 | |
| Purification: | Standard Desalting | Guaranteed Yield: | 15 ODs = 63.2 nmoles = 509.1 µgrams | |
| Sequence: | 5'- TCT CAC GCA TTA CTT CAC AAC CTC CTC -3' | | | |
| #8 | tx x | | | $11.55 |
| Product: | 250 nmole DNA oligo | Usually Ships In: | 1 business day **Length:** 21 | |
| Purification: | Standard Desalting | Guaranteed Yield: | 15 ODs = 73.5 nmoles = 461 µgrams | |
| Sequence: | 5'- CCT CAC CAT AAC ACA ATC ACA -3' | | | |
| | | | SubTotal | $127.05 USD |
| | | | Shipping and Handling | $16.00 USD |
| | | | Tax | $12.39 USD |
| | | | Total | $155.44 USD |

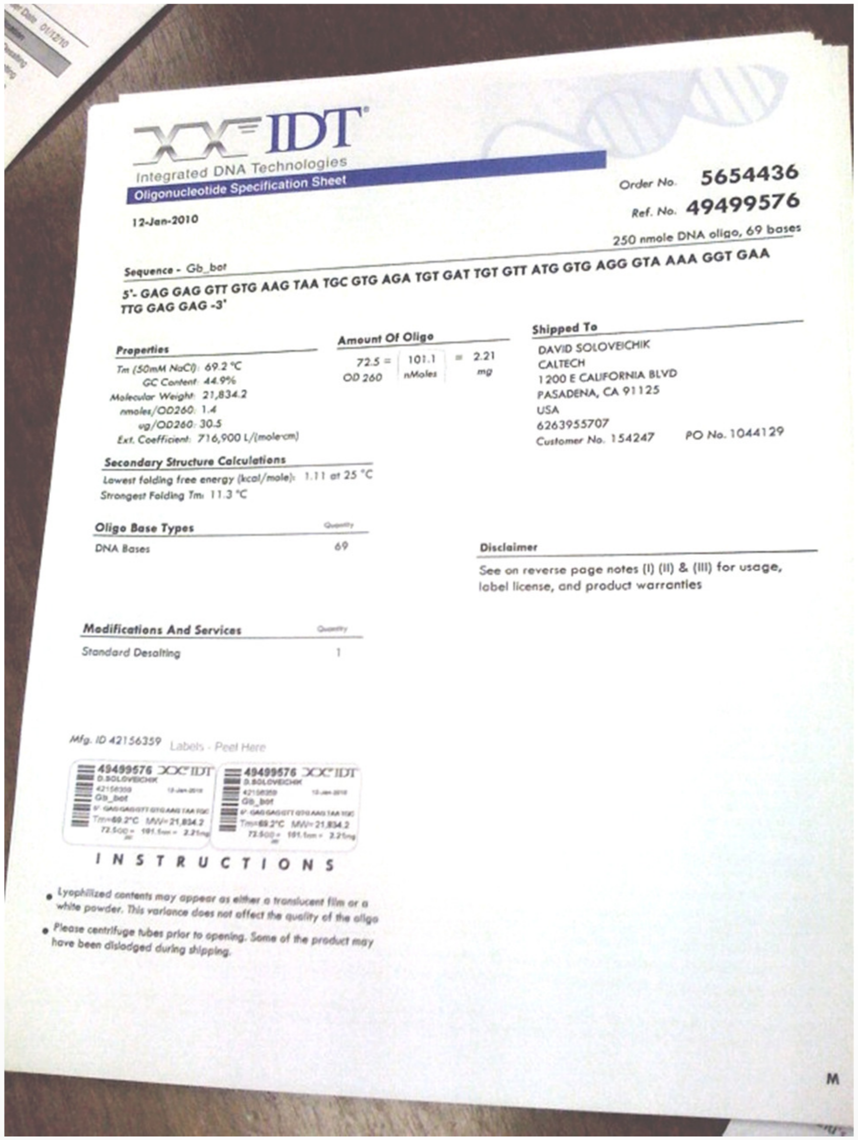
# Wait 24 Hours

• • •

# DNA by Mail


INTEGRATED DNA TECHNOLOGIES, INC.
Custom
Oligonucleotide
Synthesis
Innovation and
Precision in
Nucleic Acid
Synthesis

# Spec Sheet



IDT
Integrated DNA Technologies
Oligonucleotide Specification Sheet

12-Jan-2010

Order No. 5654436
Ref. No. 49499576
250 nmole DNA oligo, 69 bases

Sequence - Gb_bot
5'- GAG GAG GTT GTG AAG TAA TGC GTG AGA TGT GAT TGT GTT ATG GTG AGG GTA AAA GGT GAA TTG GAG GAG -3'

Properties
Tm (50mM NaCl): 69.2 °C
GC Content: 44.9%
Molecular Weight: 21,834.2
nmoles/OD260: 1.4
ug/OD260: 30.5
Ext. Coefficient: 716,900 L/(mole·cm)

Amount Of Oligo
72.5 OD 260 = 101.1 nMoles = 2.21 mg

Shipped To
DAVID SOLOVEICHIK
CALTECH
1200 E CALIFORNIA BLVD
PASADENA, CA 91125
USA
6263955707
Customer No. 154247 PO No. 1044129

Secondary Structure Calculations
Lowest folding free energy (kcal/mole): 1.11 at 25 °C
Strongest Folding Tm: 11.3 °C

| Oligo Base Types | Quantity |
|---|---|
| DNA Bases | 69 |

Disclaimer
See on reverse page notes (I) (II) & (III) for usage, label license, and product warranties

| Modifications And Services | Quantity |
|---|---|
| Standard Desalting | 1 |

Mfg. ID 42156359 Labels - Peel Here

49499576 IDT
D.SOLOVEICHIK
Gb_bot
Tm=69.2°C MW=21,834.2

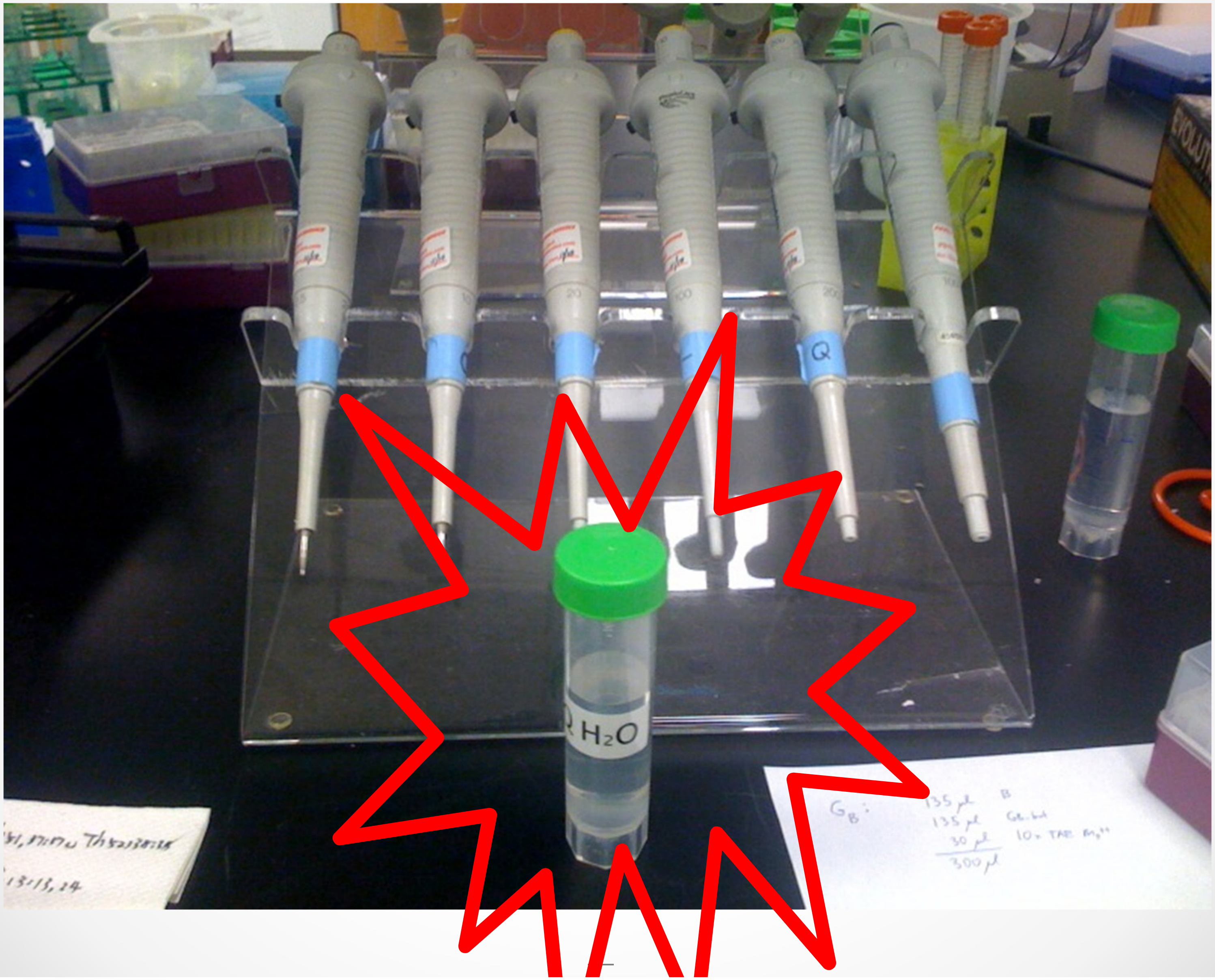
49499576 IDT
D.SOLOVEICHIK
Gb_bot
Tm=69.2°C MW=21,834.2

INSTRUCTIONS

- Lyophilized contents may appear as either a translucent film or a white powder. This variance does not affect the quality of the oligo
- Please centrifuge tubes prior to opening. Some of the product may have been dislodged during shipping.

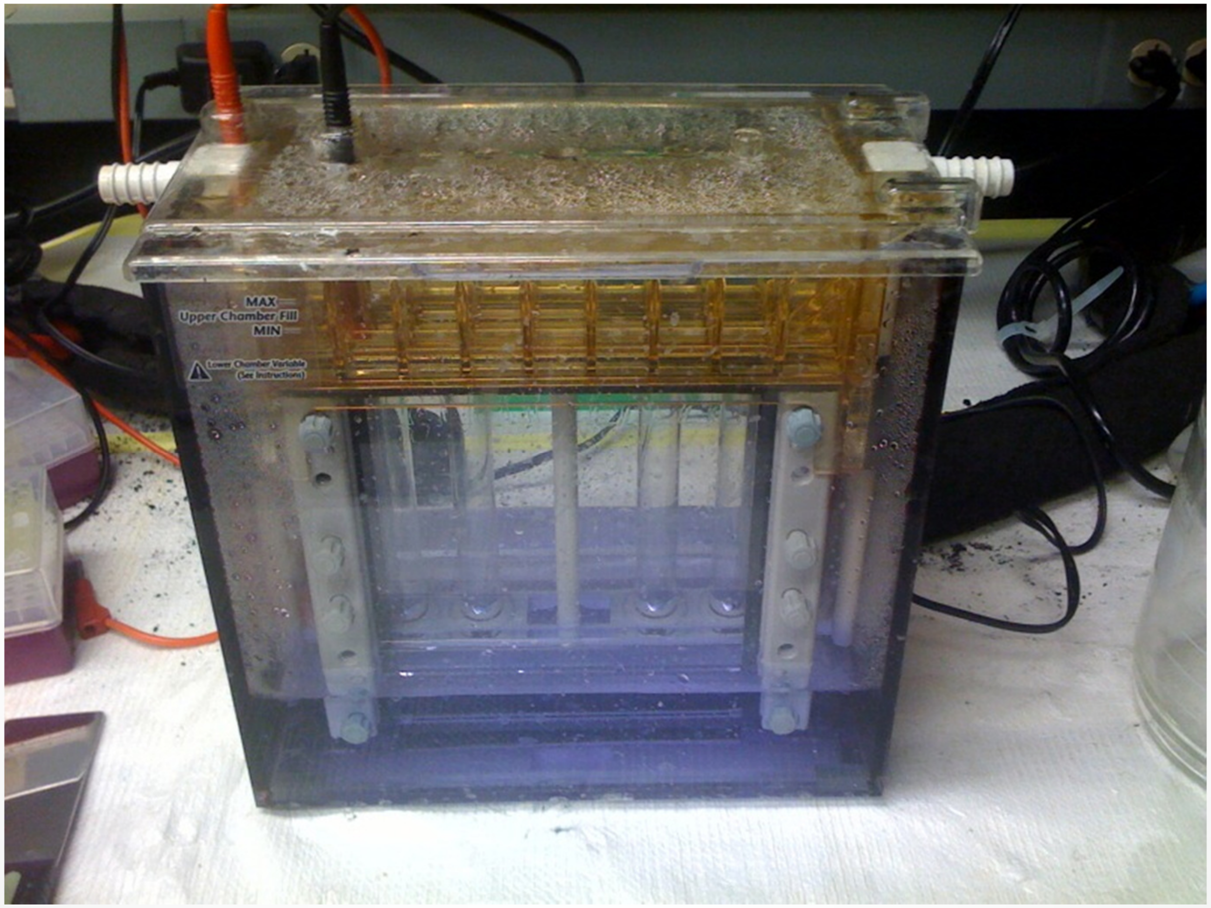
M

# Add Water



# Put DNA into Gel

- Polyacrylamide gel electrophoresis (PAGE)
- Sorts DNA strands by length

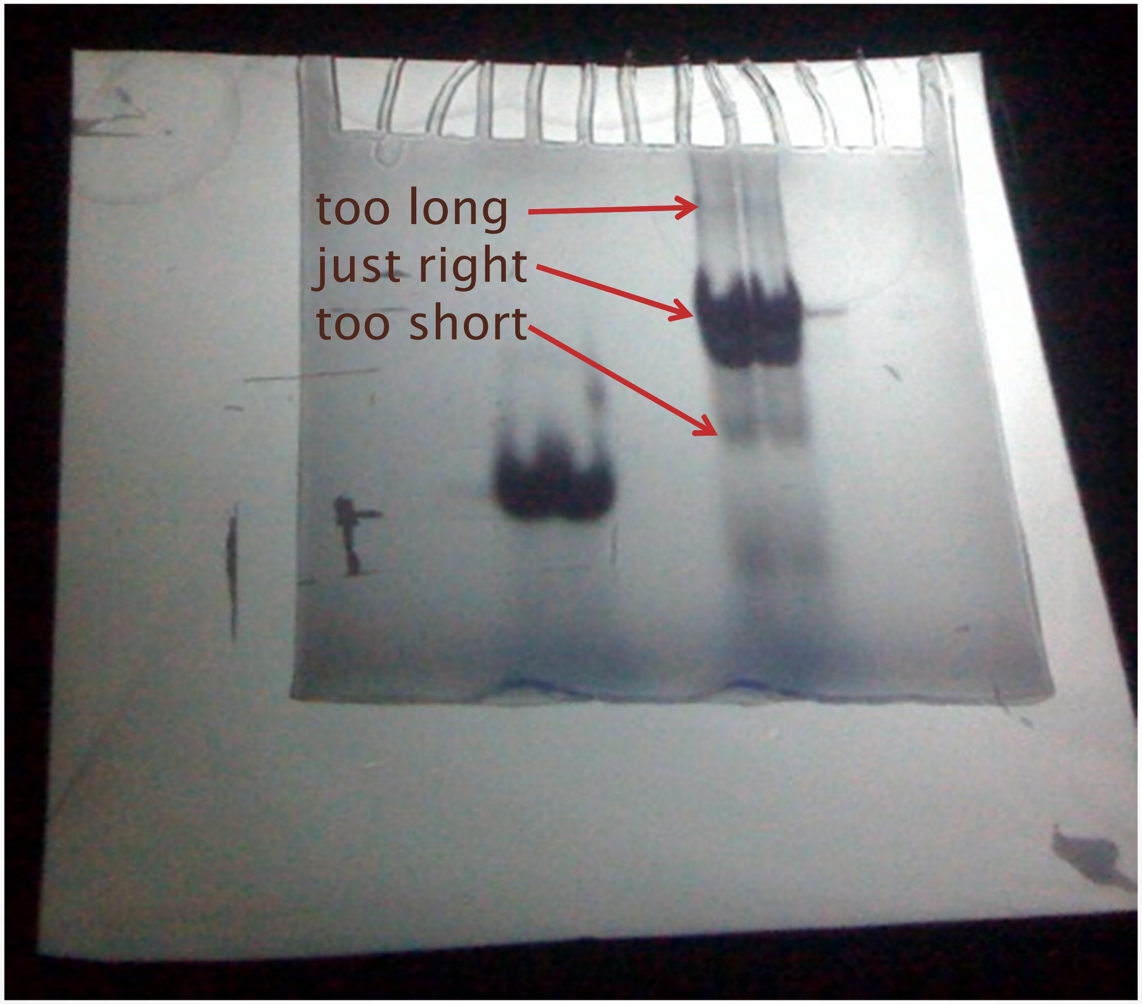


# Wait 6 Hours

• • •

# Get DNA out of Gel

- Find DNA with ultraviolet light. Cut it out.



# Wait 12 Hours

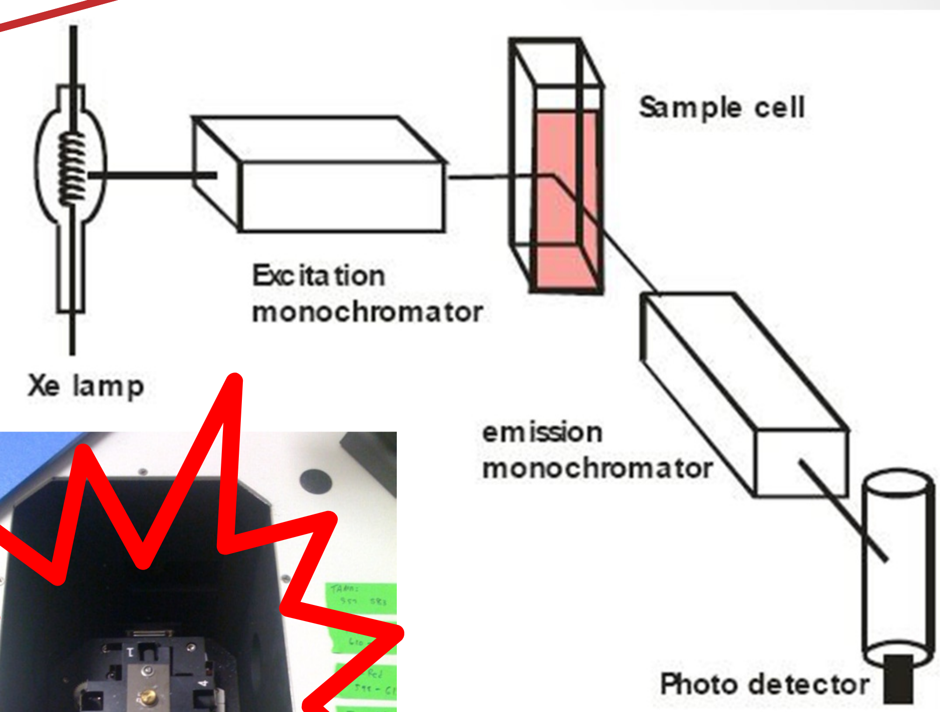
• • •

# Mix DNA Up

- Screaming for robotic automation

# Spectrofluorometer

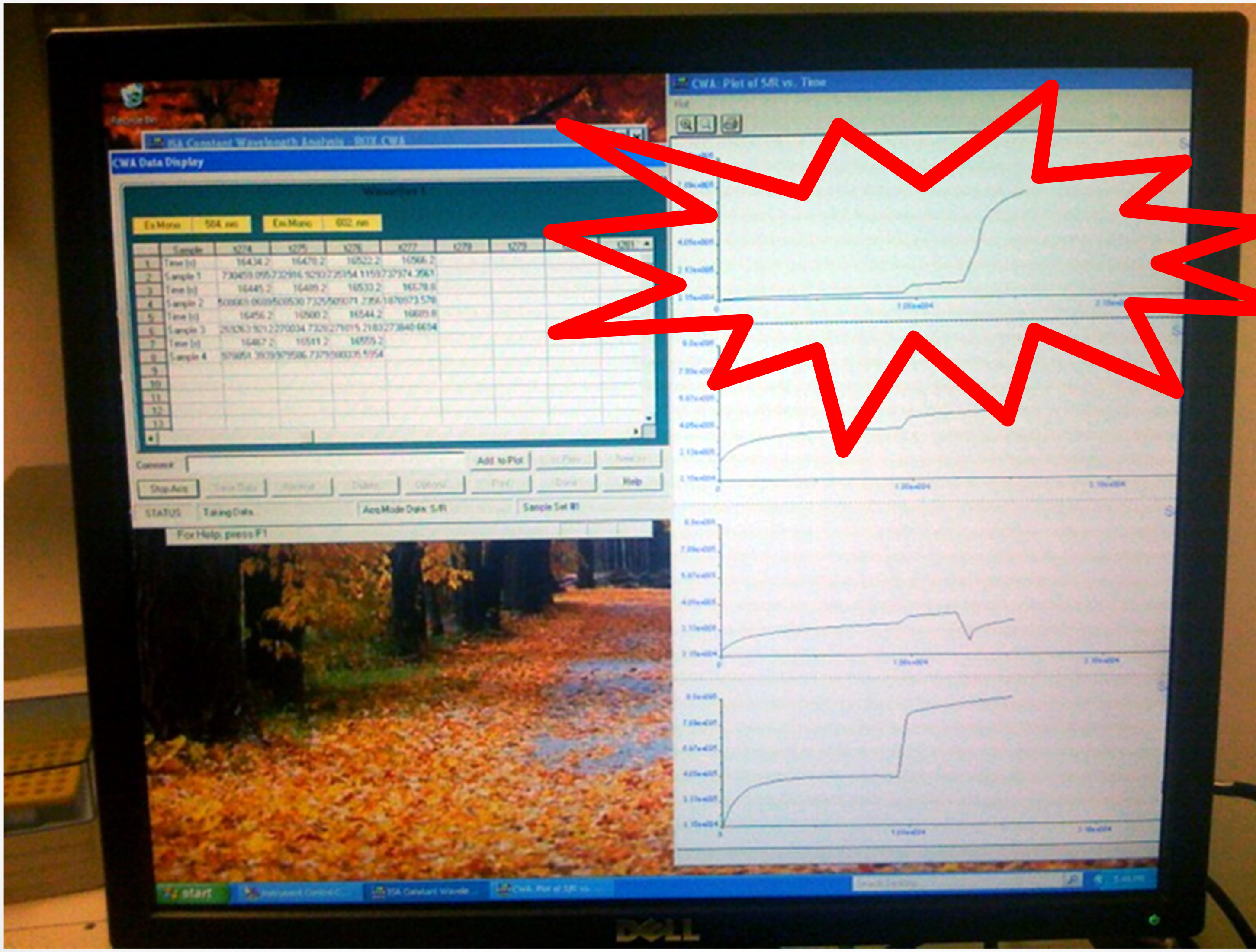
- Fluorescence is your 'print' statement

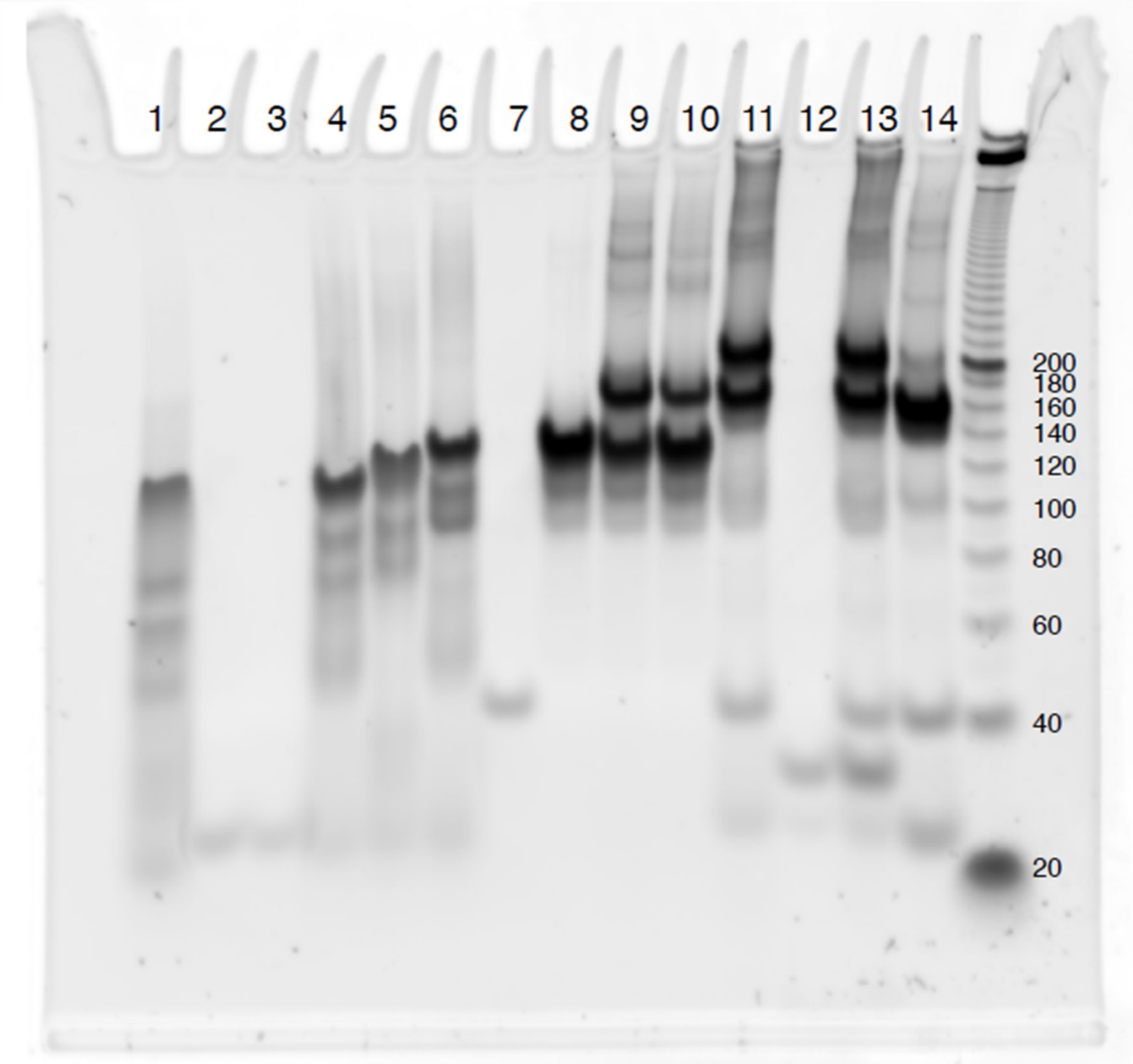

# Go To Lunch

• • •

# Execution Trace



# Core Dump

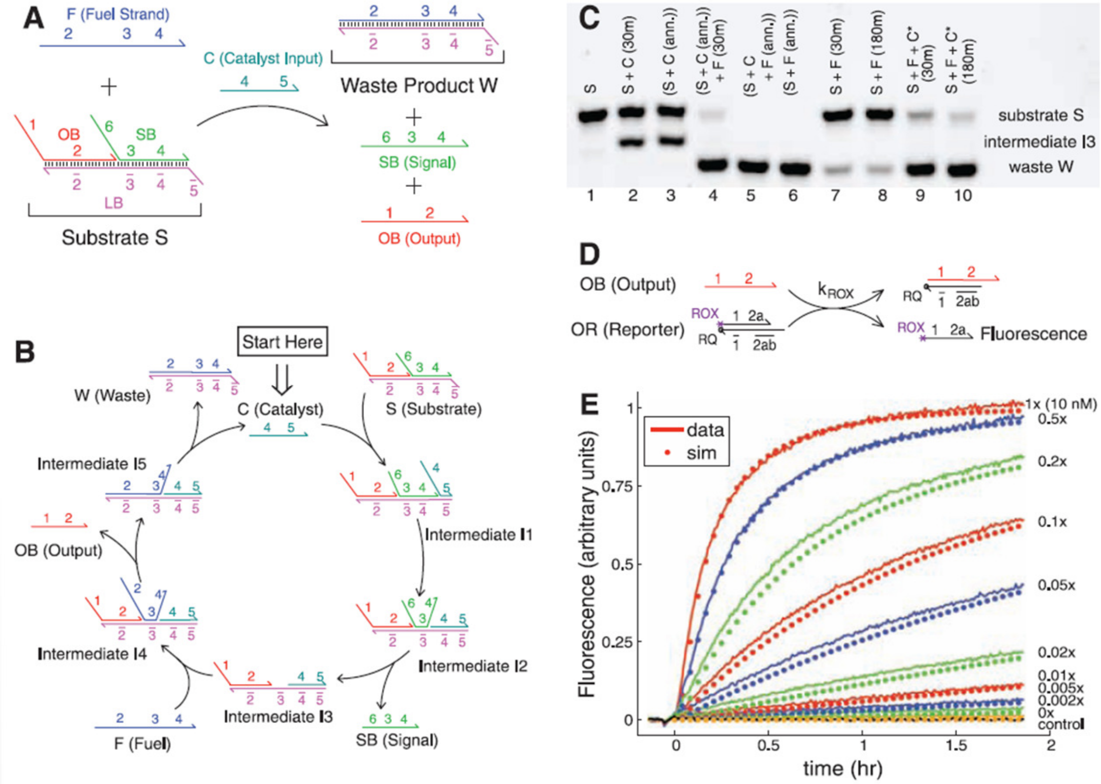


# RINSE
# &
# Repeat

• • •

# Publish!

**Engineering Entropy-Driven Reactions and Networks Catalyzed by DNA**
David Yu Zhang, *et al.*
*Science* **318**, 1121 (2007);
DOI: 10.1126/science.1148532

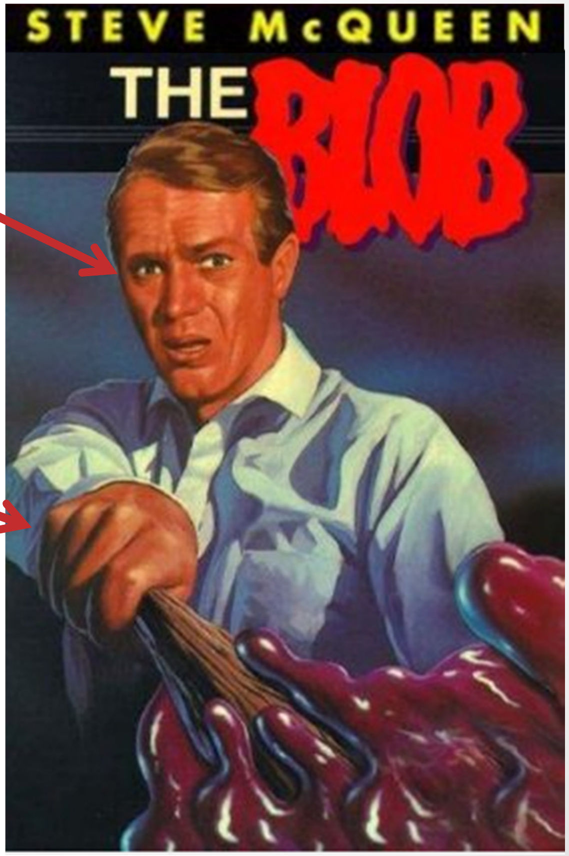


# Health and Safety

- Don't try this at home
  - (Although you could)
- Latex gloves, UV glasses
  - Fear the Gel (acrylamide)
  - Fear the Light (UV)
- Otherwise safe
  - No smells
  - No fires
  - No biohazards
  - No life forms
- Most complex machines:
  - Gel machine
  - Fluorometer
  - Atomic force microscope
- Most dangerous activity:
  - Replacing the light bulb in the fluorometer (hot; may explode)

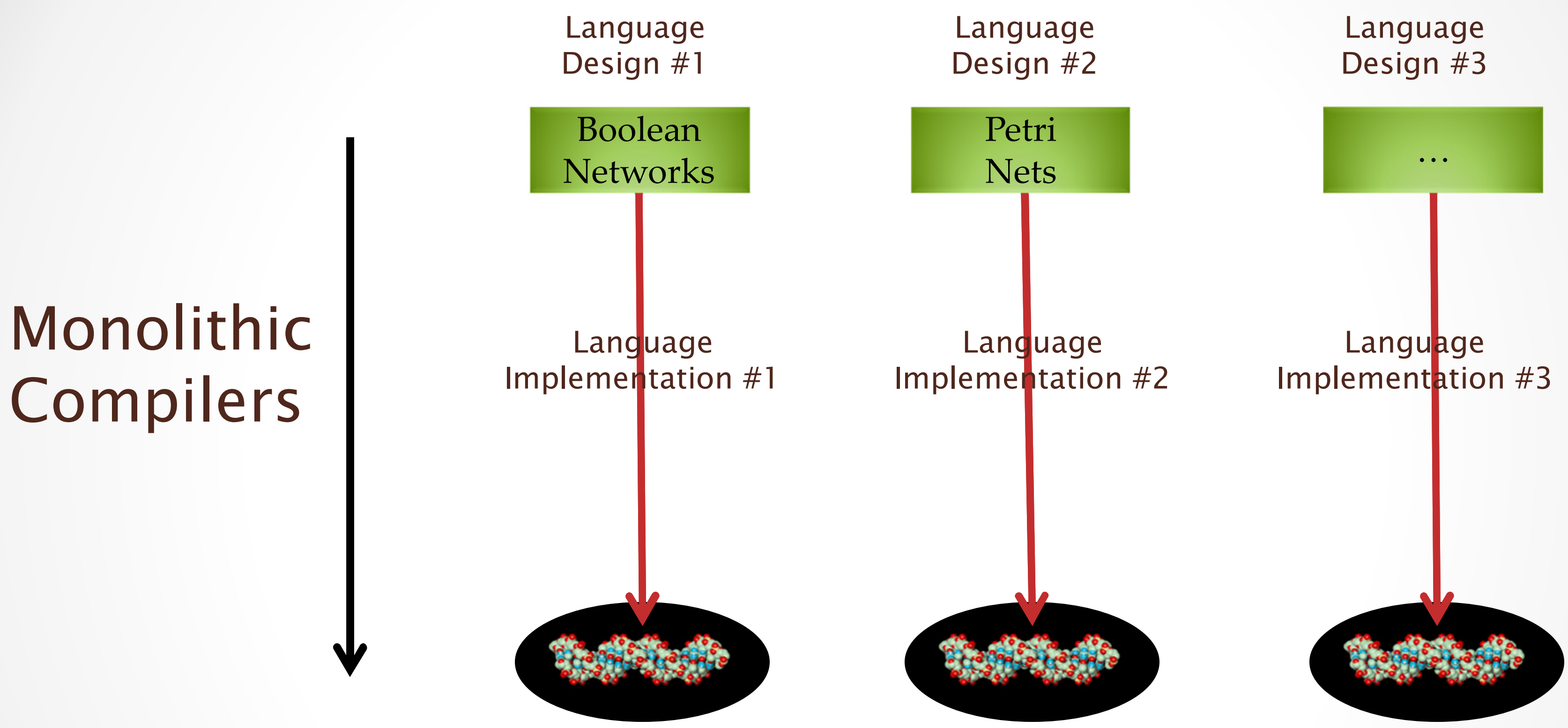
Not wearing glasses

Not wearing gloves

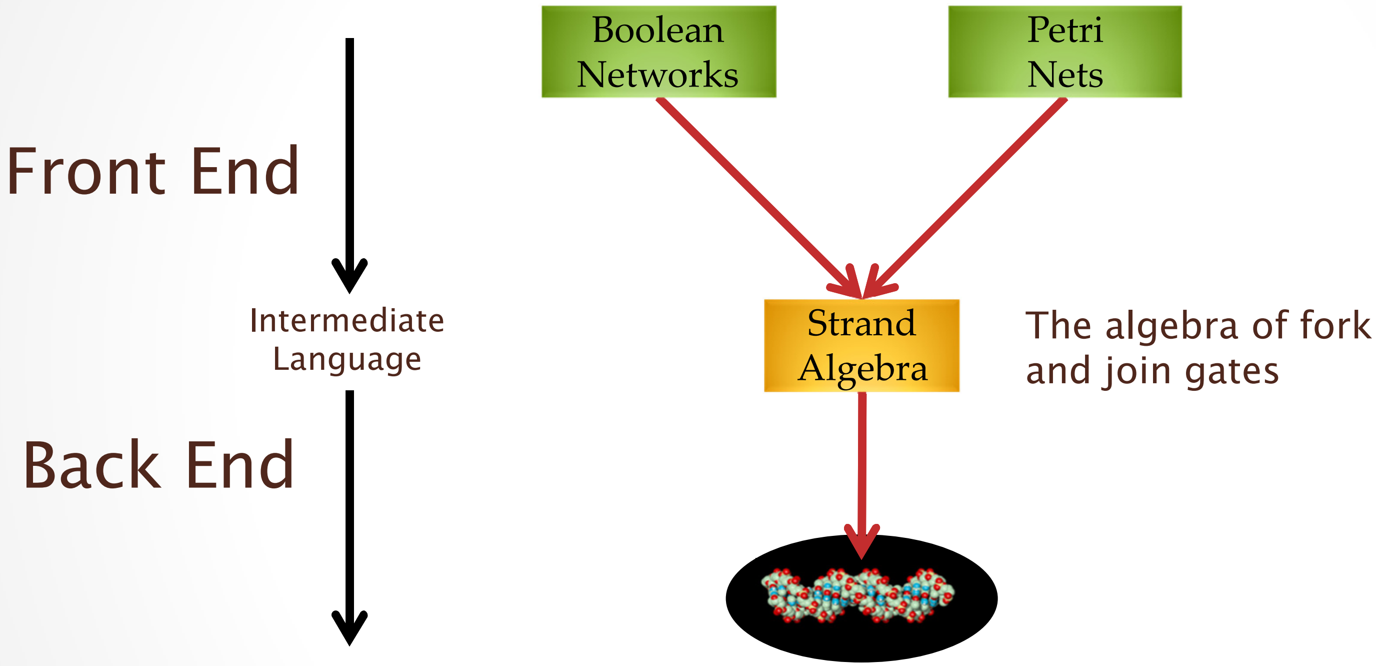


# DNA Compilation

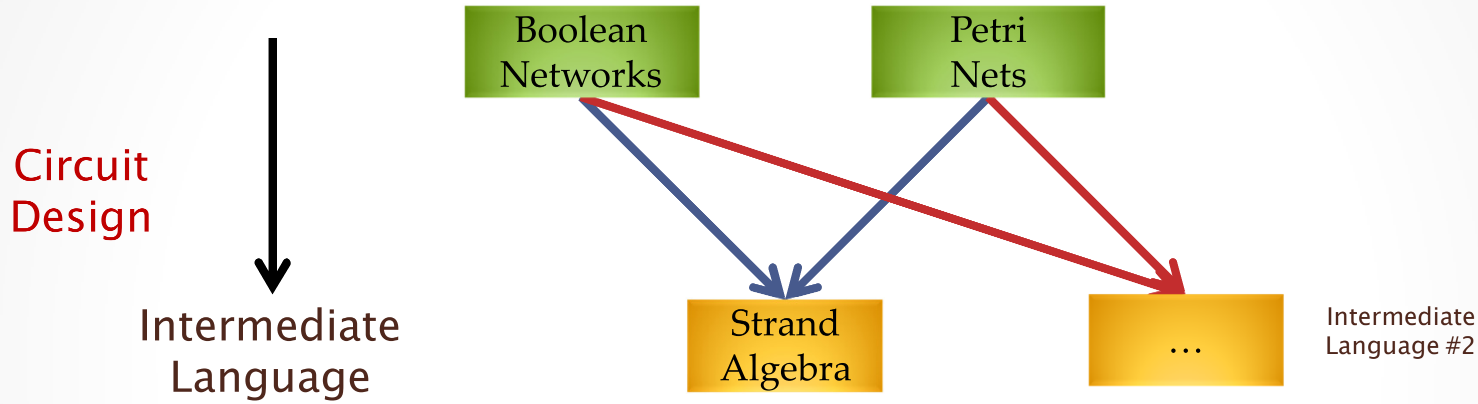
# Compilers


Language
Design #1
Language
Design #2
Language
Design #3
Boolean
Networks
Petri
Nets
...
Monolithic
Compilers
Language
Implementation #1
Language
Implementation #2
Language
Implementation #3

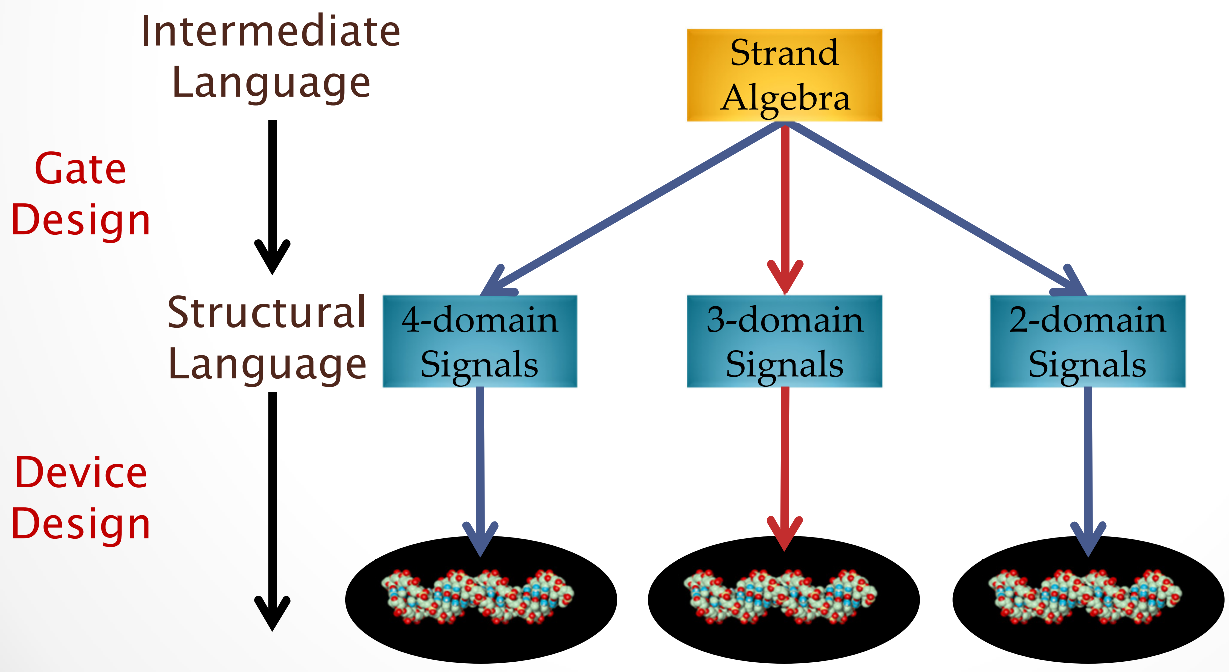
# Intermediate Languages



# Front Ends


Circuit Design
Intermediate Language
Boolean Networks
Petri Nets
Strand Algebra
…
Intermediate Language #2

# Back Ends

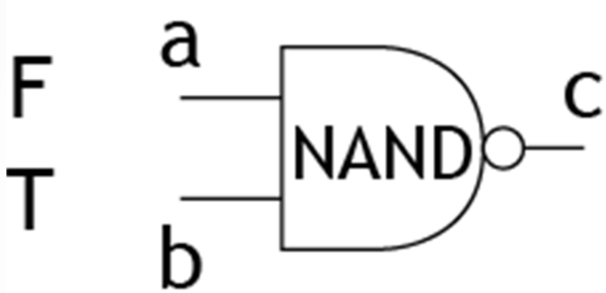

Intermediate Language
Gate Design
Structural Language
Device Design
Strand Algebra
4-domain Signals
3-domain Signals
2-domain Signals

# Compiling Abstract Machines

• • •

# Boolean Networks

Boolean Networks to Strand Algebra



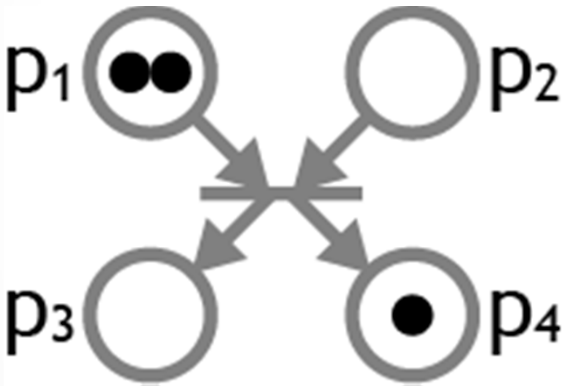

$$
\begin{aligned}
&([a_F, b_F].c_T)^* \mid \\
&([a_F, b_T].c_T)^* \mid \\
&([a_T, b_F].c_T)^* \mid \\
&([a_T, b_T].c_F)^* \mid \\
&a_F \mid b_T
\end{aligned}
$$

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




$$([p_1,p_2].[p_3,p_4])^* \mid p_1 \mid p_1 \mid p_4$$

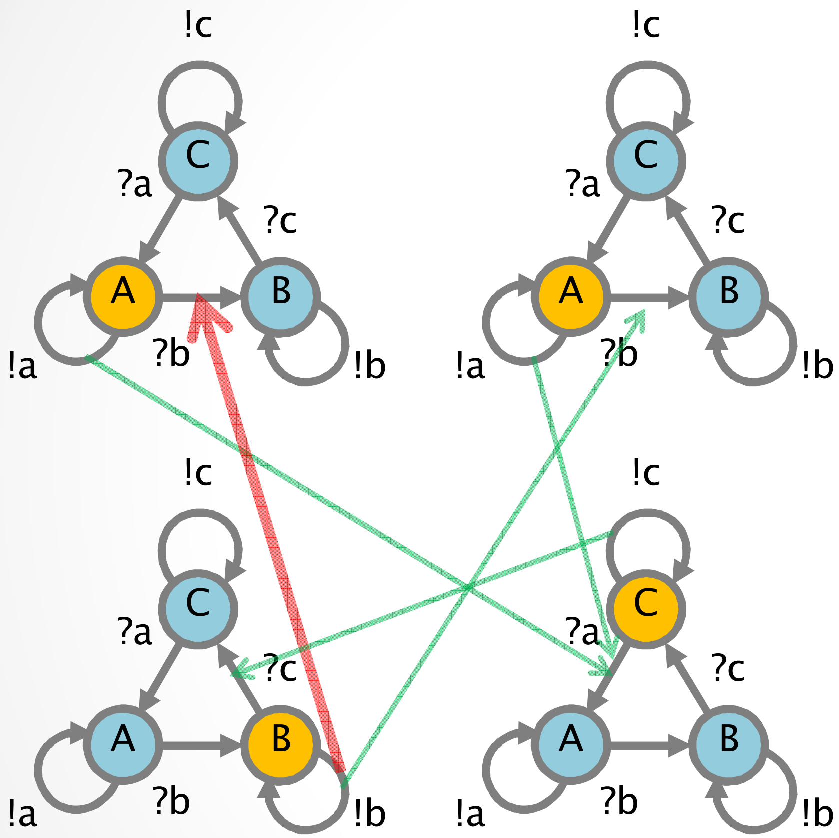
# Chemical Reaction Networks

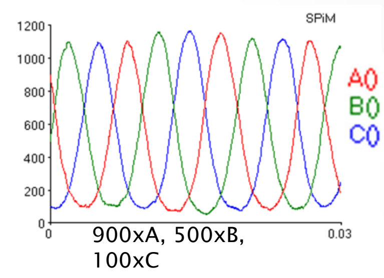
Implementing an arbitrary finite chemical system in DNA with asymptotically correct kinetics
Soloveichick & al. DNA 15

Species become signals
Reactions become gates

$$A + B \to C + D \qquad \Rightarrow \qquad [A,B].[C,D]$$

# Interacting Automata

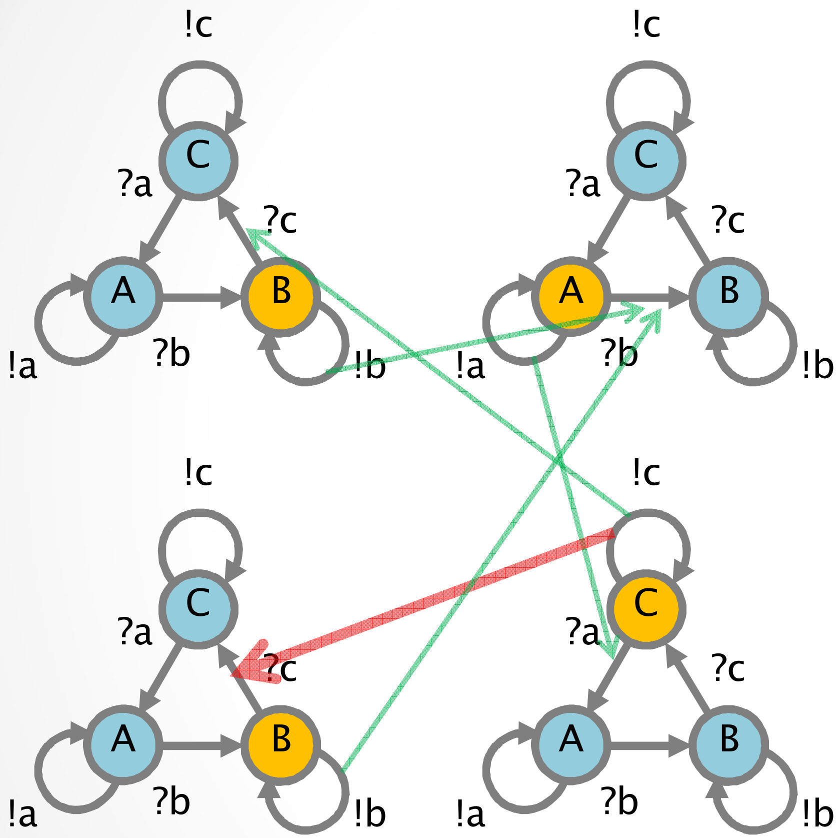


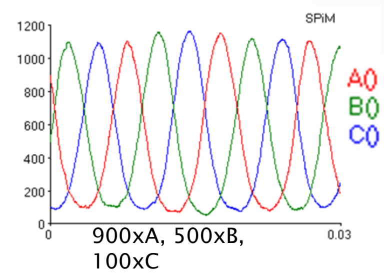




([A,B].[B,B])* |
([B,C].[C,C])* |
([C,A].[A,A])* |
A | A | B | C

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

# Interacting Automata

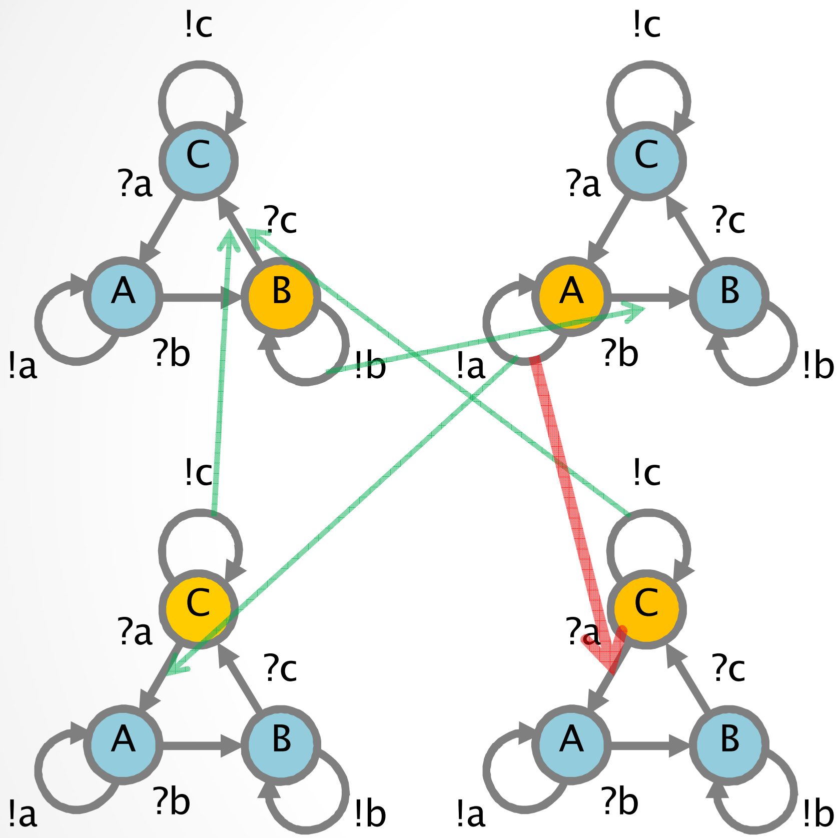


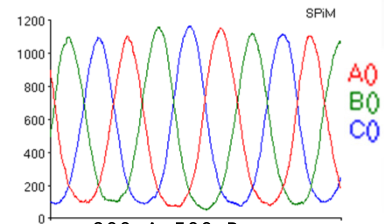




([A,B].[B,B])* |
([B,C].[C,C])* |
([C,A].[A,A])* |
A | B | B | C

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

# Interacting Automata

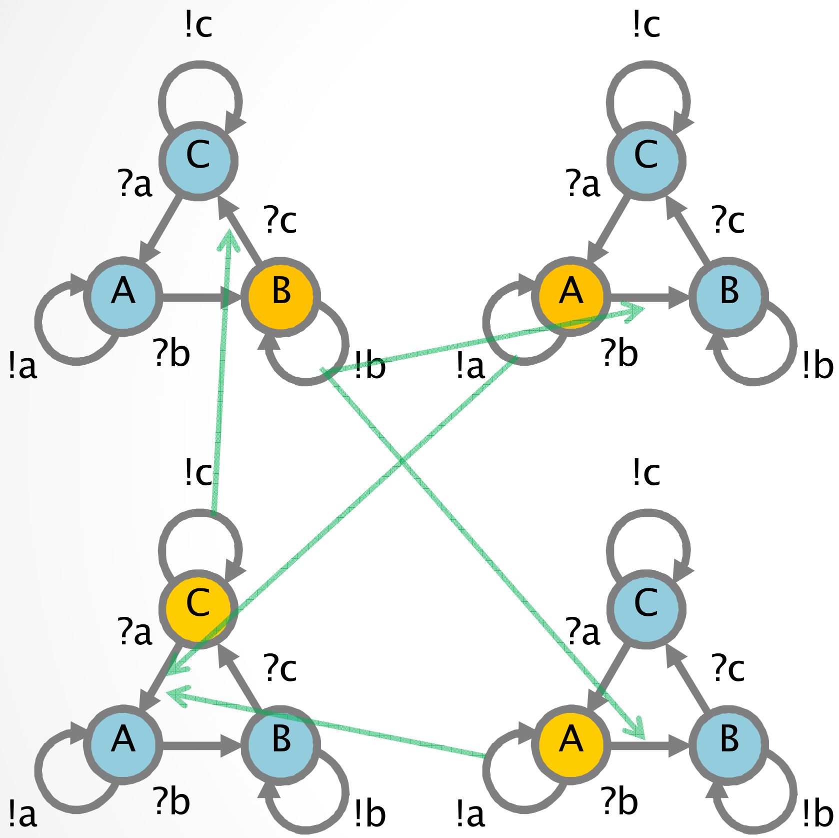


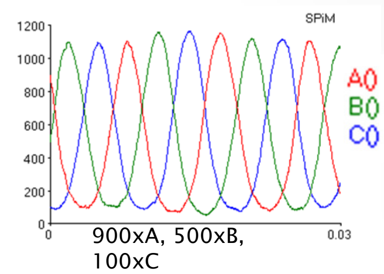




([A,B].[B,B])* |
([B,C].[C,C])* |
([C,A].[A,A])* |
A | B | C | C

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

# Interacting Automata




([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.

# And finally…

• • •

# Workflow

- Abstract Machines to Strand Algebra
  - Or other intermediate language
- Strand Algebra to DSD
  - Or other structural language
- Simulation, analysis, etc.
  - Design iteration
- DSD to Sequences
  - E.g. NuPack, or pre-build strand libraries
- Sequences to DNA
  - Web order
- DNA experiments
  - Fairly basic wet lab
- Deployable Nanotech

# Conclusions

- Programmable Matter
  - Nucleic acids
- Molecular Computation
  - DNA strand displacement
- Molecular Compilation
  - From programming abstractions (Petri Nets, Process Algebra, etc.), through intermediate language (Strand Algebra) to molecule synthesis (DNA).
- Correctness
  - Ensuring molecular programs work as intended
  - Through thermodynamic analysis, simulation, formal verification.

# Acknowledgments


- Illustrations
  - John Reif, Duke
  - Ned Seeman, NYU
  - Erik Winfree, Caltech
  - Bernard Yurke, Boise State
  - Molecular movies by Drew Berry
  - Wikipedia, YouTube
- David Soloveichik