Two-Domain DNA Strand Displacement

Luca Cardelli
Microsoft Research

TAB Cambridge 2011-06-14
http://lucacardelli.name
Nanoscale Control Systems

- **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.
**Sensing**

**Aptamers**: natural or artificially evolved DNA molecules that stick to other molecules (highly selectively).

**Adenine riboswitch aptamer**

Constructing

Crosslinking

Chengde Mao, Purdue

Andrew Turberfield, Oxford

Folding DNA into Twisted and Curved Nanoscale Shapes
Hendrik Dietz, Shawn M. Douglas, & William M. Shih
Actuating

DNA tweezers

DNA walkers

Bernard Yurke, Boise State
Computing

- Sensors and Actuators at the 'edge' of the system
  - They can use disparate technologies and phenomena

- Computation in the 'kernel' of the system

- Compositionality in the kernel
  - The components should use uniform inputs and outputs
  - The components should be ‘computationally complete’
“Embedded” Computing

• Using bacterial machinery (e.g.) as the hardware. Using embedded gene networks as the software.

• MIT Registry of Standard Biological Parts

• GenoCAD
  o Meaningful sequences [Cai et al.]

• GEC
  o [Pedersen & Phillips]
“Autonomous” Computing

• Mix & go
  o All (or most) parts are synthesized
  o No manual cycling (cf. early DNA computing)
  o In some cases, all parts are made of DNA (no enzyme/proteins)

• Self-assembled and self–powered
  o Can run on its own (e.g. environmental sensing)
  o Or be embedded into organisms, but running ‘separately’
Curing

A doctor in each cell

Fig. 1 Medicine in 2050: “Doctor in a Cell”
Autonomous DNA Computing
Why Compute with DNA?

• Non–goals
  o Not to solve NP–complete problems.
  o Not to replace electronics.
  o Not necessarily using genes or producing proteins.

• For general ‘molecular programming’
  o To precisely control the organization and dynamics of matter and information at the molecular level.
  o To interact algorithmically with biological entities.
  o The use of DNA is “accidental”: no genes involved.
  o In fact, no material of biological origin.
Domains

• Subsequences on a DNA strand are called **domains.** *PROVIDED* they are “independent” of each other.

• I.e., differently named domains must not hybridize:
  o With each other
  o With each other’s complement
  o With subsequences of each other
  o With concatenations of other domains (or their complements)
  o Etc.

• Choosing domains (subsequences) that are suitably independent is a tricky issue that is still somewhat of an open problem (with a vast literature). But it can work in practice.
Short Domains

Reversible Hybridization
Long Domains

Irreversible Hybridization
Strand Displacement

“Toehold Mediated”
Strand Displacement

Toehold Binding
Strand Displacement

Branch Migration
Strand Displacement
Strand Displacement

Irreversible release
Bad Match

t x z

x

y
Bad Match

Diagram:

- Points labeled: t, x, y, z
- Lines connecting the points
- A mismatch or misalignment highlighted
Bad Match
Bad Match

Cannot proceed
Hence will undo
Four-Domain Architecture

DNA as a universal substrate for chemical kinetics

David Soloveichik\textsuperscript{a,1}, Georg Seelig\textsuperscript{a,b,1}, and Erik Winfree\textsuperscript{e,1}

PNAS | March 23, 2010 | vol. 107 | no. 12 | 5393–5398
Three-Domain Architecture

With garbage collection (separate pass)

Strand Algebras for DNA Computing

Luca Cardelli

DNA Computing and Molecular Programming.
15th International Conference, DNA 15,
“Lulu’s Trouble”

(from D.Soloveichik)
Two-Domain Architecture

- Signals: 1 toehold + 1 recognition region

- Gates: “top-nicked double strands” (or equivalently double strands with open toeholds)

Garbage collection “built into” the gates

Two-Domain DNA Strand Displacement

Luca Cardelli

Transducer $x \rightarrow y$
Transducer $x \rightarrow y$

Input

Built by self-assembly!

**ta** is a *private* signal (a different ‘a’ for each xy pair)
Transducer $x \rightarrow y$
Transducer $x \rightarrow y$
Transducer $x \rightarrow y$
So far, a **tx signal** has produced an **at cosignal**. But we want signals as output, not cosignals.
Transducer $x \rightarrow y$
Transducer $x \rightarrow y$
Here is our output \textit{ty signal}.

But we are not done yet:
1) We need to make the output irreversible.
2) We need to remove the garbage.
We can use (2) to achieve (1).
Transducer $x \rightarrow y$
Transducer $x \rightarrow y$
Transducer $x \rightarrow y$
Transducer $x \rightarrow y$
Transducer $x \rightarrow y$
Transducer $x \rightarrow y$

Done.

N.B. the gate is consumed: it is the energy source.
Reaction Graph for $x \rightarrow y$
General $n \times m$ Join–Fork

- Easily generalized to 2+ inputs (with 1+ collectors).
- Easily generalized to 2+ outputs.

Figure 9: 3-Join $J_{wxyz}$ | $tw$ | $tx$ | $ty \rightarrow tz$: initial state plus inputs $tw, tx, ty$. 
Animations

- Animations
Experiments

Two-domain gate for $A+B \rightarrow B+C$

Experimental Challenges

• Quality of synthetic DNA
  o Chemical synthesis is limited in length and quality.
  o Two-domain scheme enables bacterial synthesis
    • Followed by enzyme digestion to introduce 'nicks'
    • Or nicking by a photosensitive artificial backbone

• Circuit optimization
  o Coming up with simpler systems for simplified experiments (shorter DNA sequences and smaller number of species)
  o Coming up with faster systems (more irreversible operations, and garbage collection).
Ex. Irreversible Output

Standard Transducer

Irreversible-output Transducer
Ex. Oscillator

- Three autocatalytic reactions
  
  \[ X + Y \rightarrow Y + Y \]
  \[ Y + Z \rightarrow Z + Z \]
  \[ Z + X \rightarrow X + X \]

- This is a closed system
  
  - (Or perhaps a performance-critical subsystem)
  - Idea: use an optimized "internal" protocol that preserves the "public" interface of the system
    - Use extra toeholds (instead of private domains) to connect the two halves of each gate (saving a domain).
    - Trick: 1 extra toehold (instead of 3) is sufficient: there is a unique variable \((x,y,z)\) connecting the two halves of gates.
Optimized Oscillator

Original

Optimized

1 day 6 days

1 day 6 days
Ex. Approximate Majority

- Four catalytic/autocatalytic reactions
  - $x + y \rightarrow y + b$
  - $y + x \rightarrow x + b$
  - $b + x \rightarrow x + x$
  - $b + y \rightarrow y + y$

- This is a closed system
  - (Or perhaps a performance-critical subsystem)
  - Same idea.
    - But now 1 extra toehold is not sufficient ($x$ and $y$ catalyze two reactions). However 2 (instead of 4) toeholds are sufficient.
Optimized AM  (Yuan-Jyue Chen)

Original

```
x t y t a t a
  x a
  t a
  y t x t a.1 t a.1
  y a.1
  y x t b t a
  b a
  b t x t a.2 t a.2
  b a.2
  b t x x t a.2
  b a.2
  b t y t y t a.3 t a.3
  b a.3
  b t y y t a.3
  b a.3
  b t y y y t a.3
  b a.3
  b t y y y t
```

Optimized

```
x t y u1 a
  x u1 a
  x a
  y t b t y u1
  y u1 a
  y a
  y t b t x u2 a
  b u2 a
  b t x u2 a
  x u2 a
  b t x x x u2
  x u2 a
  b t x x x u2
  x u2 a
  b t y t y u2 a
  y u2 a
  y a
  y t b t y u2
  y u2 a
  y a
  y t b t y y u2
  b u2 a
  b t y y y u2
  b u2 a
  b t y y y
  b a
```

```
directive sample 30000.0 1000
directive plot <t^ x>; <t^ y>; <t^ b>
directive leak 1.0e-9  (* 1.0 /M/s  * 1.0e-9 nM *)  (* /nM/s *)
directive scale 1.0 (* for stochastic simulation *)
def kt = 1.0e+6 (* /M/s *)
def exp_DeltaG_over_RT = 3.75479e-7 (* T=25C, DeltaG=-8.76 kcal/mol *)
def bind = kt*1.0e-9 (* /nM/s *)
def unbind = kt*exp_DeltaG_over_RT (* /s *)
new t@bind,unbind
new u@bind,unbind (* x + y -> y + z *)
def Cat(N, x, y, z, u) =
```
Verification Issues

• Environment
  - The nano-environment is messy (stochastic noise, failures, etc.)
  - But we should at least ensure our designs are logically correct

• Verifying Components
  - Reversible reactions (infinite traces)
  - Interferences (deadlocks etc.) between copies of the same gate
  - Interferences (deadlocks etc.) between copies of different gates
  - Removal of active byproducts (garbage collection) is tricky

• Verifying Populations
  - Gates come in (large) populations
  - Each population shares private domains (technologically unavoidable)
  - Correctness of populations means proofs with large state spaces
Correctness

• The spec of a transducer:

\[ x \cdot y \mid x \rightarrow y \]

- Is it true at all?
- Is it true \textit{possibly, necessarily, or probabilistically}?
- Is it true in the context of a \textit{population of identical transducers}?
- Is it true \textit{in all possible contexts}?
- If false, does it become true for \textit{infinite populations}?
Interfering Transducers

• Let $a$ be the private transducer domain, but let’s share it between $x.y$ and $y.x$

• Interference: $x.a y \mid y.a x \mid x \not\rightarrow \forall x$

• But still: $x.a y \mid y.a x \mid x \mid y \rightarrow \forall x \mid y$

• A large population of such gates in practice does not deadlock easily.

• The wisdom of crowds: individuals can be wrong, but the population is all right.
Modelchecking DNA Systems

- Using the PRISM stochastic model checker
  - Termination probability of interfering transducers
    \[ x | x.a \ y | y.a \ z \]

Conclusions
Summary

• Molecular Structures
  o Hard to build… but they can build themselves!

• Molecular Languages
  o Natural and unnatural
  o Concurrent, quantitative

• Molecular Compilation
  o Molecular architectures, verification, optimization

• Molecular Programming
  o In silico, in vitro, in vivo…
Acknowledgments

• Microsoft Research
  o Andrew Phillips

• Caltech
  o Winfree Lab

• U.Washington
  o Seelig Lab