DNA strand displacement

DNA reconfiguring itself without enzymes

slides © 2023, David Doty
ECS 232: Theory of Molecular Computation, UC Davis
DNA strands with “long” and “short” (toehold) binding domains
DNA strand displacement example
DNA strand displacement example
DNA strand displacement example

“breathing”/“fraying”
DNA strand displacement example

“breathing”/“fraying”
DNA strand displacement example

“breathing”/“fraying”
DNA strand displacement example

“breathing”/“fraying”
DNA strand displacement example

“breathing”/“fraying”

branch migration
DNA strand displacement example

“breathing”/“fraying”

branch migration
DNA strand displacement example

“breathing”/“fraying”

branch migration

irreversible
DNA strand displacement

DNA strand displacement

DNA strand displacement model

3 rules:
1. bind
2. release
3. displace
Bind rule

single-stranded complementary domains can bind
Release rule

double-stranded complementary domains can unbind
IF they are toehold-length (short, < 8 nt)
Displace rule

A domain (invader) can displace an identical domain (incumbent) of another strand, 
**IF** neighboring domains are already bound.
Displace rule

A domain (invader) can displace an identical domain (incumbent) of another strand, IF neighboring domains are already bound.

Displace

reversible if incumbent strand remains bound by another domain.
Readout

How do we read a “signal” in a DNA strand displacement system?
Fluorophores, when “excited” by light at one wavelength, emit light at a longer wavelength.
Fluorophores, when “excited” by light at one wavelength, emit light at a longer wavelength.
How do we read a “signal”? “signal” = single strand is freed from a double-stranded complex.
Reporter complex depiction
Boolean logic with DNA strand displacement
AND gate

release Z if and only if X and Y are present

[voltages]

[slides credit: Chris Thachuk]
Strand displacement cascade example: AND gate

release Z if and only if X and Y are present

[slides credit: Chris Thachuk]
Strand displacement cascade example: AND gate

release Z if and only if X and Y are present

[slides credit: Chris Thachuk]
Strand displacement cascade example: AND gate

release Z if and only if X and Y are present
Strand displacement cascade example: AND gate

release Z if and only if X and Y are present

[slides credit: Chris Thachuk]
Strand displacement cascade example: AND gate

release Z if and only if X and Y are present

[slides credit: Chris Thachuk]
Strand displacement cascade example: AND gate

release Z if and only if X and Y are present

[slides credit: Chris Thachuk]
Strand displacement cascade example: AND gate

release Z if and only if X and Y are present
Strand displacement cascade example: AND gate

release Z if and only if X and Y are present

[slides credit: Chris Thachuk]
Strand displacement cascade example: AND gate

release Z if and only if X and Y are present.
Strand displacement cascade example: AND gate

release Z if and only if X and Y are present
Strand displacement cascade example: AND gate

release Z if and only if X and Y are present

[slides credit: Chris Thachuk]
Strand displacement cascade example: AND gate

release Z if and only if X and Y are present

output Z
Strand displacement cascade example: AND gate

release Z if and only if X and Y are present

[slides credit: Chris Thachuk]
Composing AND gates

We need a “wire”.

A
B
X
Y
W
Z
We need a “wire” to translate the signal: $X \rightarrow Y$
(with no shared DNA sequences between $X$ and $Y$)
Translator gate (a “wire”)

input X

bind

F₁

F₂

X → Y
Translator gate (a “wire”)
Translator gate (a “wire”)

output Y
Strand displacement cascade example: OR gate

An OR gate can be implemented by multiple translators:

\[ Z \leftarrow W \text{ OR } X \text{ OR } Y \]
Strand displacement cascade example: Avoiding the need for NOT gates using dual-rail logic

NOT gates are tricky with molecular circuits:
  How to make a molecule Y present
  if and only if X is not present??
Strand displacement cascade example: Avoiding the need for NOT gates using dual-rail logic

NOT gates are tricky with molecular circuits: How to make a molecule Y present if and only if X is not present??

Instead we use dual-rail logic, using de Morgan’s Laws to push all the NOT gates to the input.

(Then we can “manually” specify FALSE input values by the presence of a “negated” strand.)
Strand displacement cascade example: Avoiding the need for NOT gates using dual-rail logic

NOT gates are tricky with molecular circuits:
How to make a molecule $Y$ present if and only if $X$ is not present??

Instead we use dual-rail logic, using de Morgan’s Laws to push all the NOT gates to the input.

(Then we can “manually” specify FALSE input values by the presence of a “negated” strand.)
Strand displacement cascade example: Avoiding the need for NOT gates using dual-rail logic

NOT gates are tricky with molecular circuits:
How to make a molecule Y present if and only if X is not present??

Instead we use dual-rail logic, using de Morgan’s Laws to push all the NOT gates to the input.

(Then we can “manually” specify FALSE input values by the presence of a “negated” strand.)

\[
\text{NOT}(A \lor B) = \text{NOT } A \land \text{NOT } B
\]
Strand displacement cascade example: Avoiding the need for NOT gates using dual-rail logic

NOT gates are tricky with molecular circuits:
   How to make a molecule Y present if and only if X is not present??

Instead we use “dual-rail” logic, using de Morgan’s Laws to push all the NOT gates to the input, so we can “manually” specify FALSE input values.

\[
\text{NOT}(A \lor B) = (\text{NOT } A) \land (\text{NOT } B)
\]
Strand displacement cascade example: Avoiding the need for NOT gates using dual-rail logic

NOT gates are tricky with molecular circuits: How to make a molecule $Y$ present if and only if $X$ is not present??

Instead we use “dual-rail” logic, using de Morgan’s Laws to push all the NOT gates to the input, so we can “manually” specify FALSE input values.
Strand displacement cascade example: Avoiding the need for NOT gates using dual-rail logic

NOT gates are tricky with molecular circuits:
  How to make a molecule Y present if and only if X is not present??

Instead we use “dual-rail” logic, using de Morgan’s Laws to push all the NOT gates to the input, so we can “manually” specify FALSE input values.
Strand displacement cascade example: Avoiding the need for NOT gates using dual-rail logic

NOT gates are tricky with molecular circuits:
How to make a molecule Y present if and only if X is not present??

Instead we use “dual-rail” logic, using de Morgan’s Laws to push all the NOT gates to the input, so we can “manually” specify FALSE input values.
Strand displacement cascade example: Avoiding the need for NOT gates using dual-rail logic

NOT gates are tricky with molecular circuits: How to make a molecule Y present if and only if X is not present??

Instead we use “dual-rail” logic, using de Morgan’s Laws to push all the NOT gates to the input, so we can “manually” specify FALSE input values.
Strand displacement cascade example: Avoiding the need for NOT gates using dual-rail logic

NOT gates are tricky with molecular circuits:
How to make a molecule Y present if and only if X is not present??

Instead we use “dual-rail” logic, using de Morgan’s Laws to push all the NOT gates to the input, so we can “manually” specify FALSE input values.
Strand displacement cascade example: Avoiding the need for NOT gates using dual-rail logic

NOT gates are tricky with molecular circuits:
How to make a molecule Y present if and only if X is not present??

Instead we use “dual-rail” logic, using de Morgan’s Laws to push all the NOT gates to the input, so we can “manually” specify FALSE input values.
Strand displacement cascade example: Avoiding the need for NOT gates using dual-rail logic

NOT gates are tricky with molecular circuits: How to make a molecule Y present if and only if X is not present??

Instead we use “dual-rail” logic, using de Morgan’s Laws to push all the NOT gates to the input, so we can “manually” specify FALSE input values.
Strand displacement cascade example: Avoiding the need for NOT gates using dual-rail logic

For each input $X_i$, there are two species $X_i^T$ and $X_i^F$:
Give species $X_i^F$ to specify that Boolean input $X_i = False$
Give species $X_i^T$ to specify that Boolean input $X_i = True$. 
Strand displacement cascade example: Avoiding the need for NOT gates using dual-rail logic

For each input $X_i$, there are two species $X_i^T$ and $X_i^F$:
Give species $X_i^F$ to specify that Boolean input $X_i = False$
Give species $X_i^T$ to specify that Boolean input $X_i = True$. 
Strand displacement cascade example: Avoiding the need for NOT gates using dual-rail logic

For each input $X_i$, there are two species $X_i^T$ and $X_i^F$:
- Give species $X_i^F$ to specify that Boolean input $X_i = False$
- Give species $X_i^T$ to specify that Boolean input $X_i = True$.

For composability, can make output dual-rail as well.
Dual-rail logic computing square root of 4-bit number

\[ y_2 y_1 = \sqrt{x_4 x_3 x_2 x_1} \]

[Scaling up digital circuit computation with DNA strand displacement cascades. Lulu Qian and Erik Winfree, *Science* 2011]
Implementing CRNs with DNA

“Compiling” arbitrary chemical reaction networks into DNA strands that implement the reactions using DNA strand displacement
DNA strand displacement can implement any CRN

unimolecular reaction $X_1 \rightarrow X_2 + X_3$

reaction $i: X_1 \xrightarrow{k_i} X_2 + X_3$ \[1\]
\[↓ \] implement
$X_1 + G_i \xrightarrow{q_i} O_i$ \[2\]
$O_i + T_i \xrightarrow{q_{max}} X_2 + X_3$ \[3\]
\[↓ \] simplify
$X_1 \xrightarrow{q_{i}C_{max}} O_i$ \[4\]
$O_i \xrightarrow{q_{max}C_{max}} X_2 + X_3$ \[5\]
DNA strand displacement can implement any CRN

bimolecular reaction $X_1 + X_2 \rightarrow X_3$

---

**A**

$X_1 
\begin{array}{c}
\text{species identifier} \\
? \\
1 \\
2 \\
3 \\
4 \\
5 \\
6 \\
7 \\
\end{array} 
+

$\begin{array}{c}
\text{species identifier} \\
? \\
1 \\
2 \\
3 \\
4 \\
5 \\
6 \\
7 \\
\end{array} 
$

$q_i = \frac{q_{i}}{q_{\text{max}}}$

$X_2 
\begin{array}{c}
\text{species identifier} \\
? \\
1 \\
2 \\
3 \\
4 \\
5 \\
6 \\
7 \\
\end{array} 
+

$\begin{array}{c}
\text{species identifier} \\
? \\
1 \\
2 \\
3 \\
4 \\
5 \\
6 \\
7 \\
\end{array} 
$

$q_{i} = \frac{q_{i}}{q_{\text{max}}}$

**B**

$X_1 
\begin{array}{c}
\text{species identifier} \\
? \\
1 \\
2 \\
3 \\
4 \\
5 \\
6 \\
7 \\
\end{array} 
+

$\begin{array}{c}
\text{species identifier} \\
? \\
1 \\
2 \\
3 \\
4 \\
5 \\
6 \\
7 \\
\end{array} 
$

$q_{i} = \frac{q_{i}}{q_{\text{max}}}$

$X_2 
\begin{array}{c}
\text{species identifier} \\
? \\
1 \\
2 \\
3 \\
4 \\
5 \\
6 \\
7 \\
\end{array} 
+

$\begin{array}{c}
\text{species identifier} \\
? \\
1 \\
2 \\
3 \\
4 \\
5 \\
6 \\
7 \\
\end{array} 
$

$q_{i} = \frac{q_{i}}{q_{\text{max}}}$

**C**

$X_1 
\begin{array}{c}
\text{species identifier} \\
? \\
1 \\
2 \\
3 \\
4 \\
5 \\
6 \\
7 \\
\end{array} 
+

$\begin{array}{c}
\text{species identifier} \\
? \\
1 \\
2 \\
3 \\
4 \\
5 \\
6 \\
7 \\
\end{array} 
$

$q_{i} = \frac{q_{i}}{q_{\text{max}}}$

$X_2 
\begin{array}{c}
\text{species identifier} \\
? \\
1 \\
2 \\
3 \\
4 \\
5 \\
6 \\
7 \\
\end{array} 
+

$\begin{array}{c}
\text{species identifier} \\
? \\
1 \\
2 \\
3 \\
4 \\
5 \\
6 \\
7 \\
\end{array} 
$

$q_{i} = \frac{q_{i}}{q_{\text{max}}}$

---

[DNA as a universal substrate for chemical kinetics, David Soloveichik, Georg Seelig, and Erik Winfree, *PNAS* 2010]
“Two-domain” scheme for compiling CRN to DSD

reaction $A + B \rightarrow C$

Experimental implementations of CRN-to-DSD schemes
DSD computing approximate majority

Goal:

DSD computing approximate majority

Goal:

\[
\begin{align*}
X + Y & \to 2B \\
X + B & \to 2X \\
Y + B & \to 2Y
\end{align*}
\]

CRN:

DSD computing approximate majority

Goal:

CRN:

\[ X + Y \rightarrow 2B \]
\[ X + B \rightarrow 2X \]
\[ Y + B \rightarrow 2Y \]

DSD implementing chemical “rock-paper-scissors” oscillator

**A**  
Desired dynamics

![Graph showing concentration over time with oscillatory behavior](image)

Molecular program

\[ A, \ B, \ C \]
\[ B + A \xrightarrow{k} 2B \]
\[ C + B \xrightarrow{k} 2C \]
\[ A + C \xrightarrow{k} 2A \]

DNA architecture

![Diagram of DNA architecture](image)

DNA dynamics

![Graph showing DNA dynamics](image)

Concentrations of free A, B, C

**B**

<table>
<thead>
<tr>
<th>Added (A, B, C)</th>
<th>Time (hours)</th>
<th>Helper deriv. (nM/hr)</th>
</tr>
</thead>
<tbody>
<tr>
<td>(0, 10, 13) nM</td>
<td>0-50</td>
<td>CCK, AAq, BBs</td>
</tr>
<tr>
<td>(11, 0, 13) nM</td>
<td>0-50</td>
<td>AAq, BBs, CCK</td>
</tr>
<tr>
<td>(11, 10, 3) nM</td>
<td>0-50</td>
<td>BBs, CCK, AAq</td>
</tr>
</tbody>
</table>