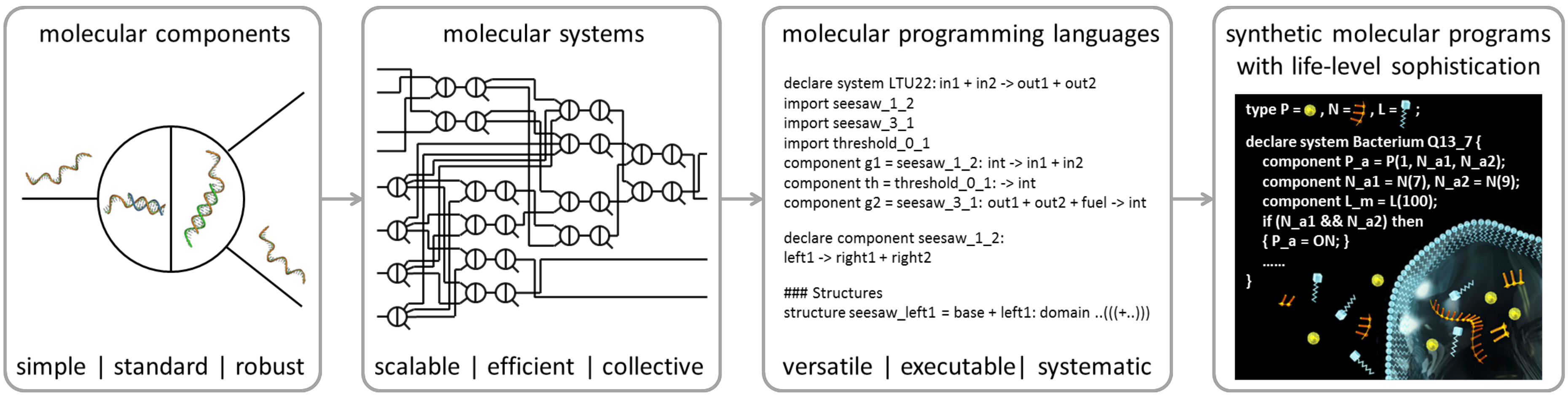
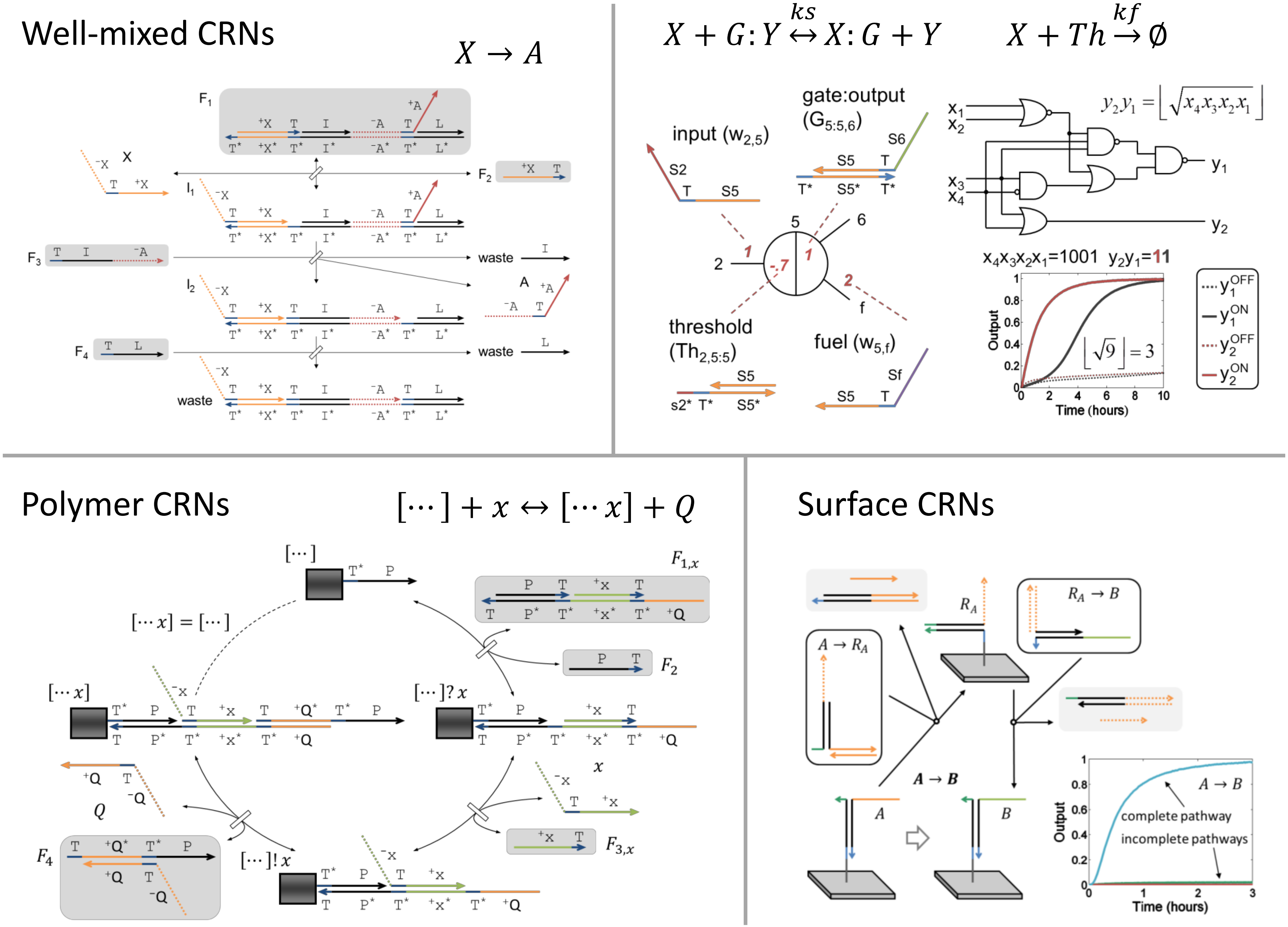
Implementing complex CRNs with modular DNA components



Lulu Qian

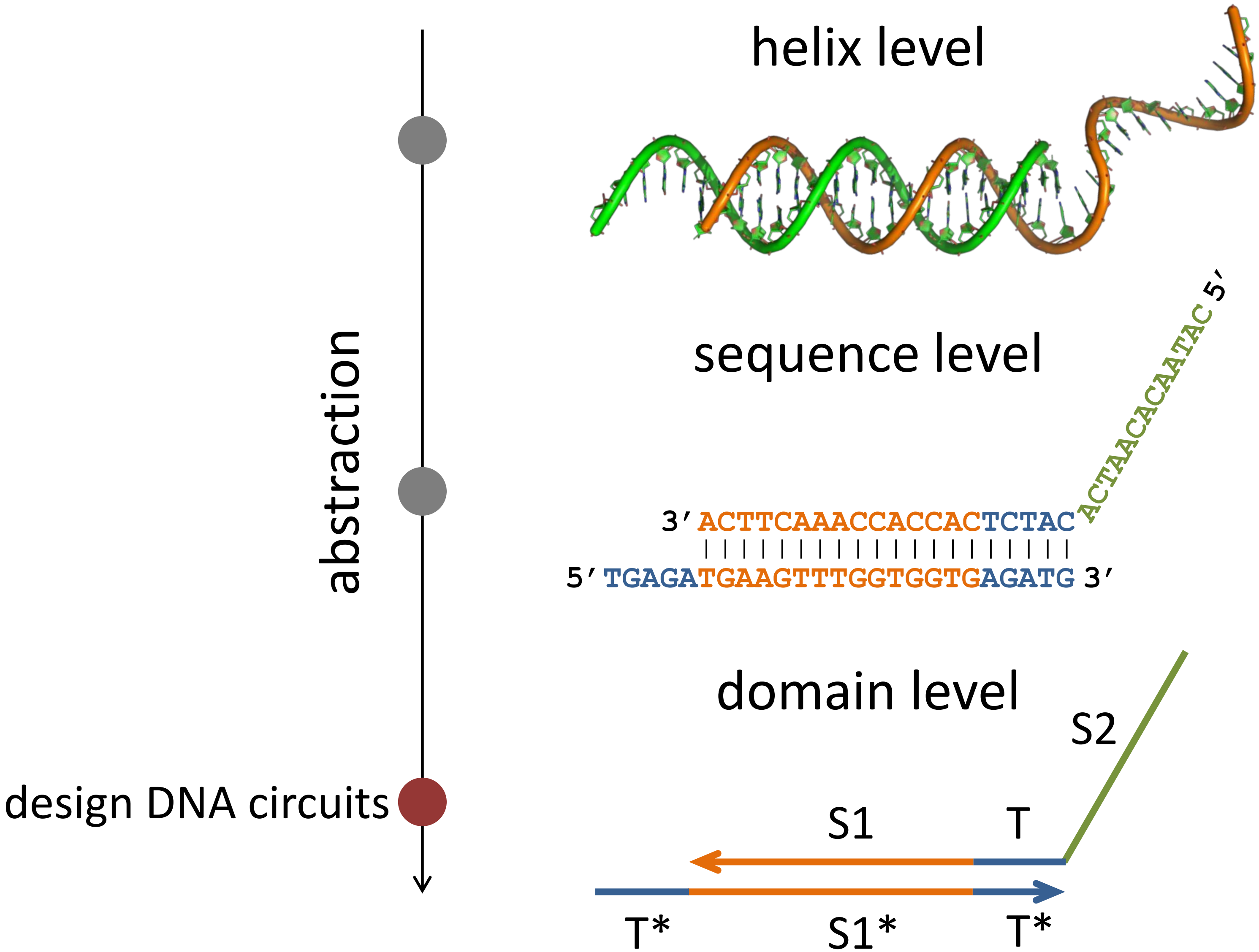
Bioengineering

Caltech



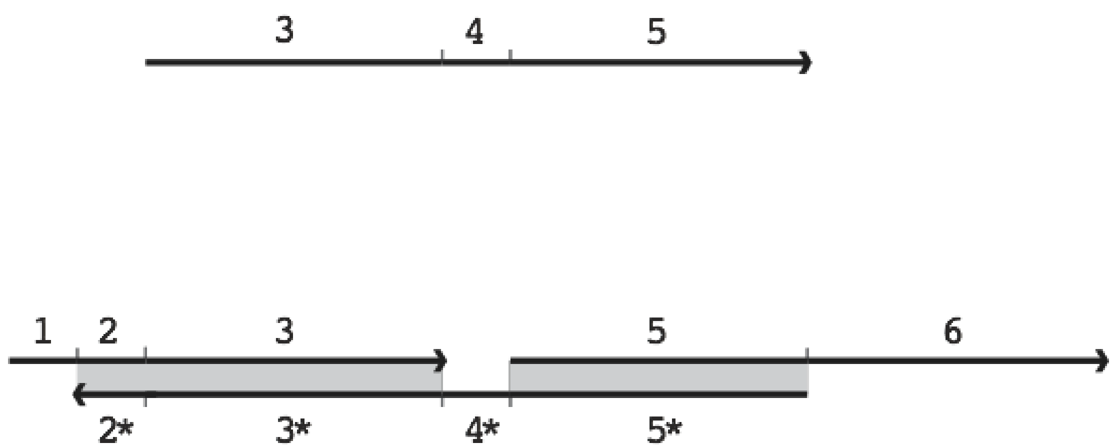
Well-mixed CRNs
X → A
F1
F2
F3
F4
I1
I2
waste
A
X + G:Y ↔ X:G + Y
ks
X + Th → ∅
kf
input (w2,5)
gate:output (G5:5,6)
threshold (Th2,5:5)
fuel (w5,f)
S2
S5
S6
Sf
T
T*
S5*
s2*
x1
x2
x3
x4
y1
y2
x4x3x2x1=1001 y2y1=11
Output
Time (hours)
Polymer CRNs
[⋯] + x ↔ [⋯x] + Q
[⋯]
[⋯x] = [⋯]
[⋯x]
[⋯]?x
[⋯]!x
F1,x
F2
F3,x
F4
Q
x
Surface CRNs
A → RA
RA → B
RA
A → B
A
B
complete pathway
incomplete pathways
Output
Time (hours)

Representations of DNA molecules



Principles of DNA strand displacement circuits

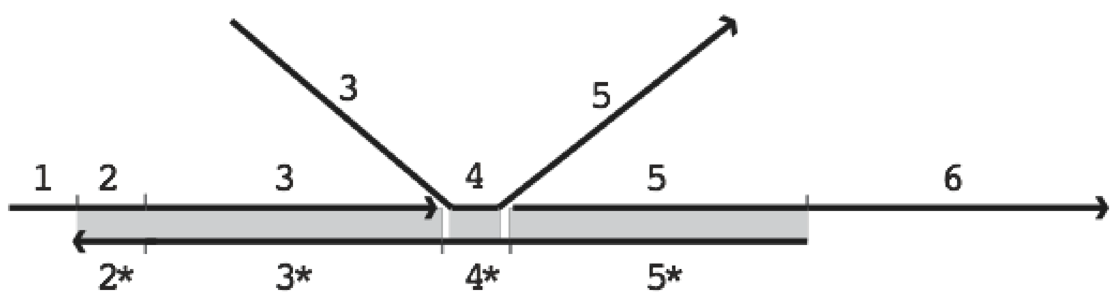
bind: two complementary domains can bind.
unbind: any strands held by only a short domain can unbind.
displace: a domain can displace an identical domain if this extends existing binding.



Credit: David Soloveichik

Principles of DNA strand displacement circuits

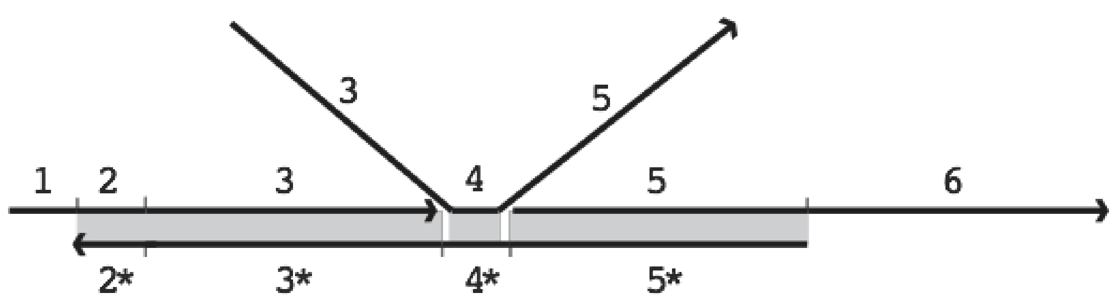
bind: two complementary domains can bind.
unbind: any strands held by only a short domain can unbind.
displace: a domain can displace an identical domain if this extends existing binding.



Credit: David Soloveichik

Principles of DNA strand displacement circuits

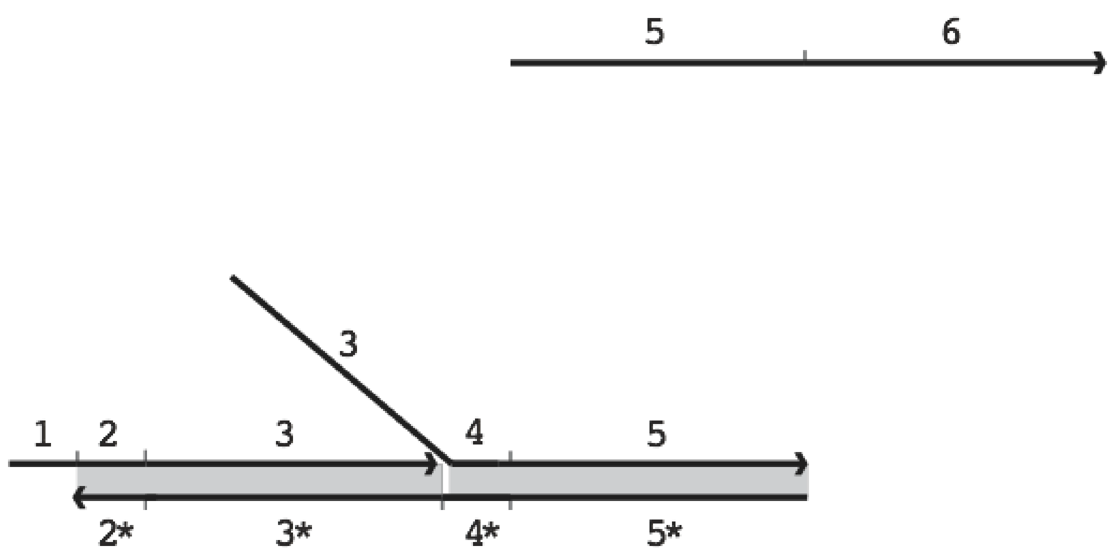
bind: two complementary domains can bind.
unbind: any strands held by only a short domain can unbind.
displace: a domain can displace an identical domain if this extends existing binding.



Credit: David Soloveichik

Principles of DNA strand displacement circuits

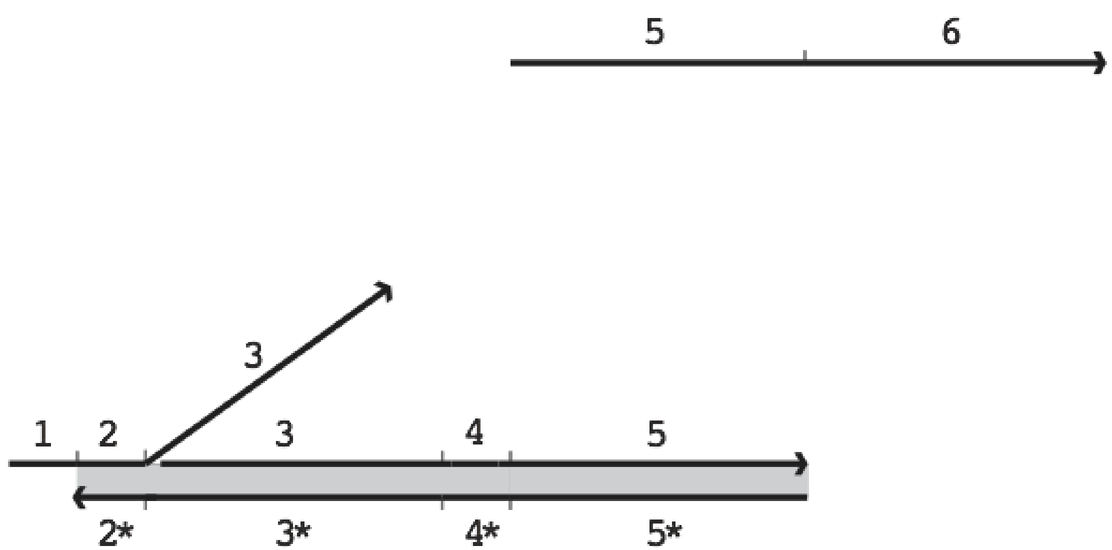
bind: two complementary domains can bind.
unbind: any strands held by only a short domain can unbind.
displace: a domain can displace an identical domain if this extends existing binding.



Credit: David Soloveichik

Principles of DNA strand displacement circuits

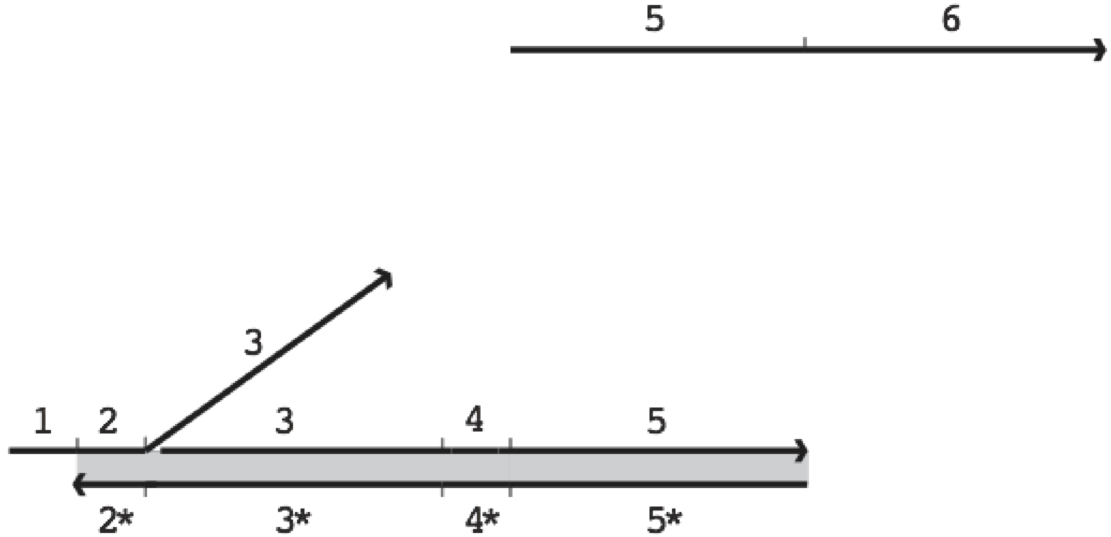
bind: two complementary domains can bind.
unbind: any strands held by only a short domain can unbind.
displace: a domain can displace an identical domain if this extends existing binding.



Credit: David Soloveichik

Principles of DNA strand displacement circuits

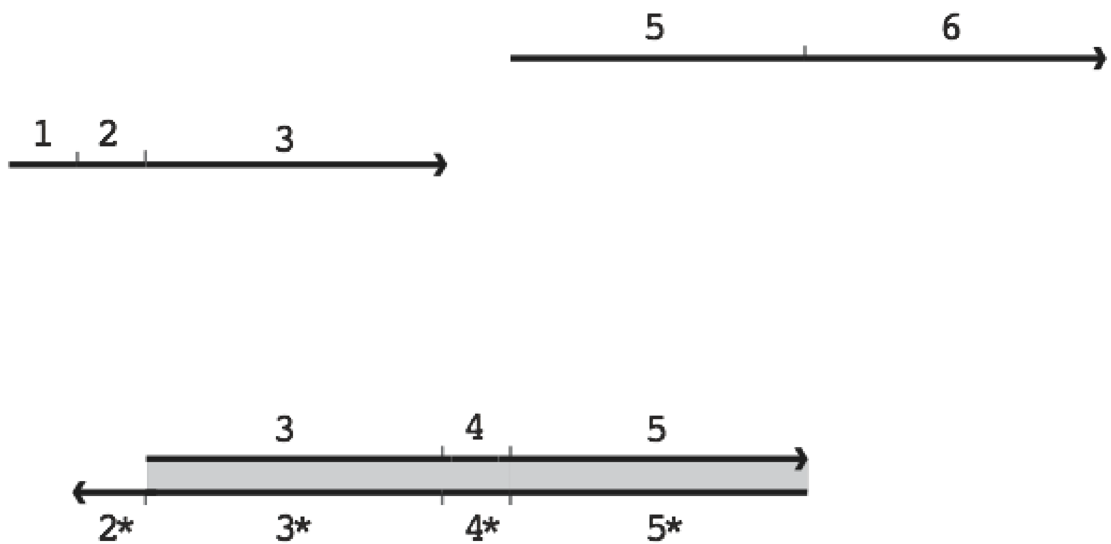
bind: two complementary domains can bind.
unbind: any strands held by only a short domain can unbind.
displace: a domain can displace an identical domain if this extends existing binding.



Credit: David Soloveichik

Principles of DNA strand displacement circuits

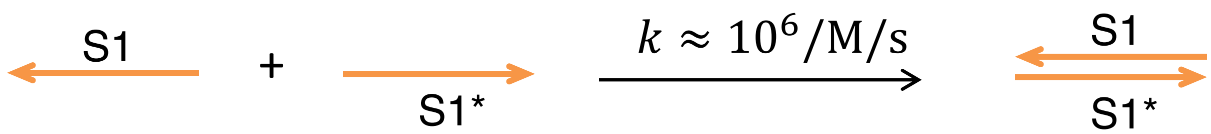
bind: two complementary domains can bind.
unbind: any strands held by only a short domain can unbind.
displace: a domain can displace an identical domain if this extends existing binding.



Credit: David Soloveichik

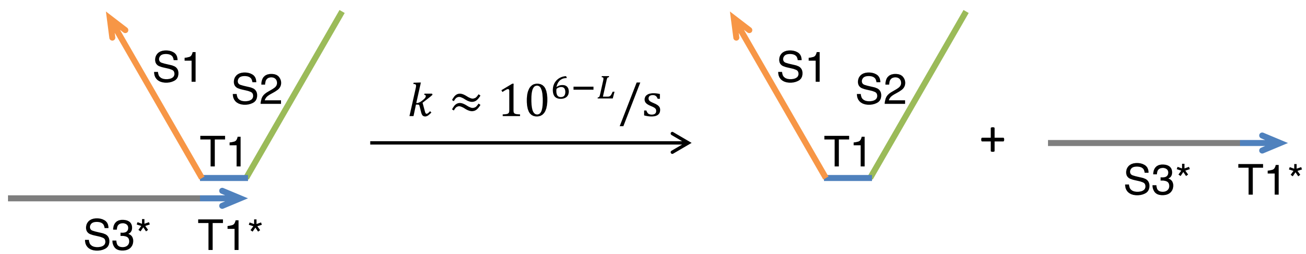
Principles of DNA strand displacement circuits

bind: two complementary domains can bind.



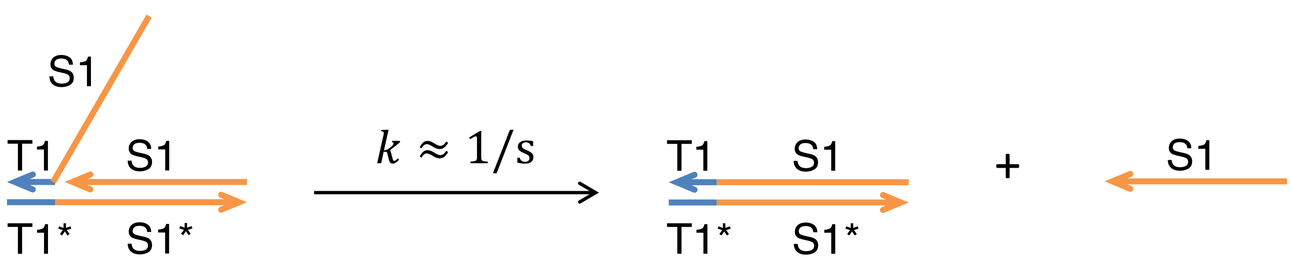
$$\Delta G^\circ \approx -L \cdot \Delta G^\circ_{\mathrm{bp}}$$

unbind: any strands held by only a short domain can unbind.



$$\Delta G^\circ \approx L \cdot \Delta G^\circ_{\mathrm{bp}}$$

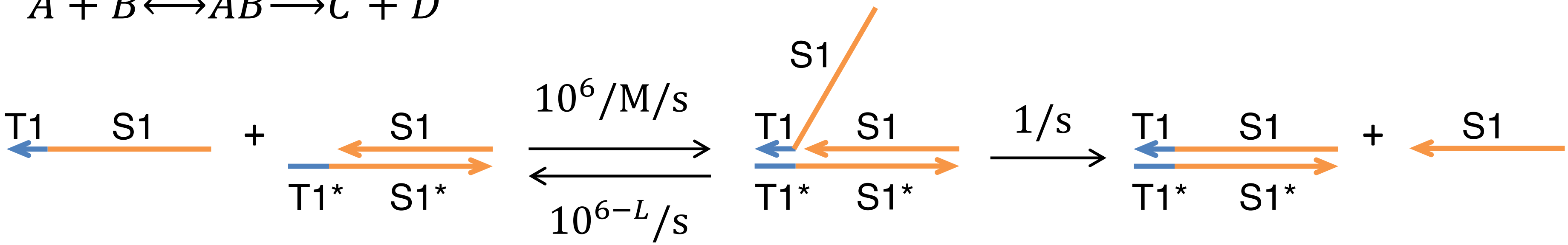
displace: a domain can displace an identical domain if this extends existing binding.



$$\Delta G^\circ = 0$$

Kinetics of toehold-mediated strand displacement

$A + B \longleftrightarrow AB \longrightarrow C + D$



simplify: $A + B \xrightarrow{k_{eff}} C + D$ $\quad k_{eff} = ?$

$k_{eff} \approx 10^L/\mathrm{M/s}$ when $L \leq 6$

otherwise $k_{eff} \approx 10^6/\mathrm{M/s}$

L: toehold length |T1|

collision rate: $10^6[A][B]$

collision success probability: $\dfrac{1/\mathrm{s}}{1/\mathrm{s} + 10^{6-L}/\mathrm{s}}$

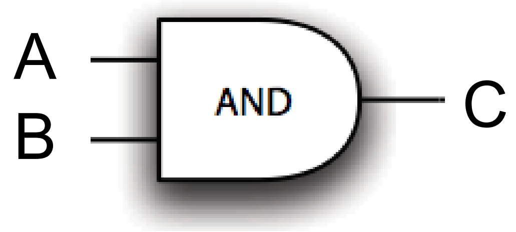
net rate of success: $\underbrace{10^6 \cdot \dfrac{1}{1 + 10^{6-L}}}_{k_{eff}}[A][B]$

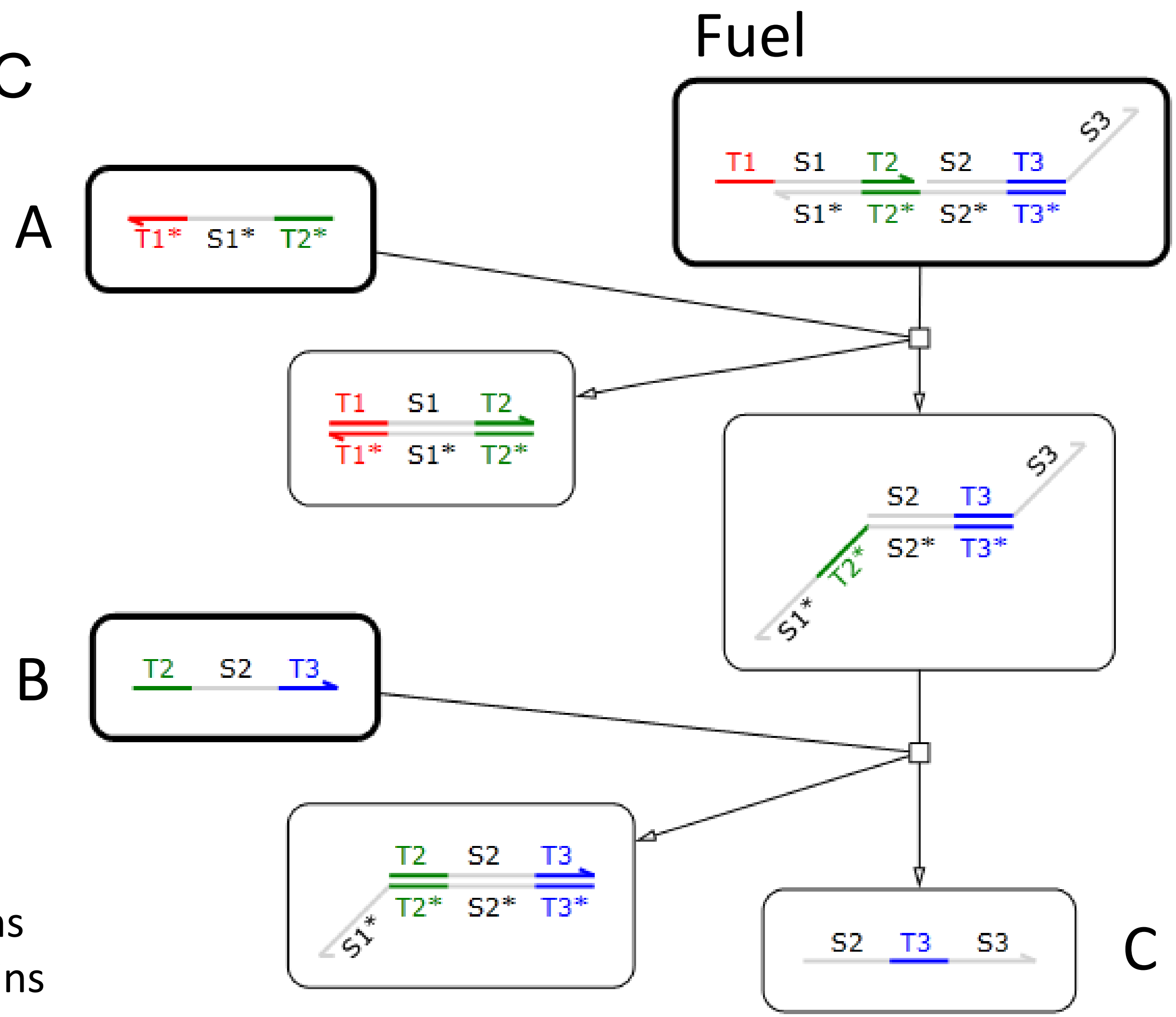
Zhang et al, *JACS* 2009

Srinivas et al, *NAR* 2013

This approximation is valid for low concentrations of A and B (e.g. [A]=[B]=100nM) such that the unimolecular reaction is sufficiently faster than the bimolecular reaction.

Examples of simple strand displacement circuits

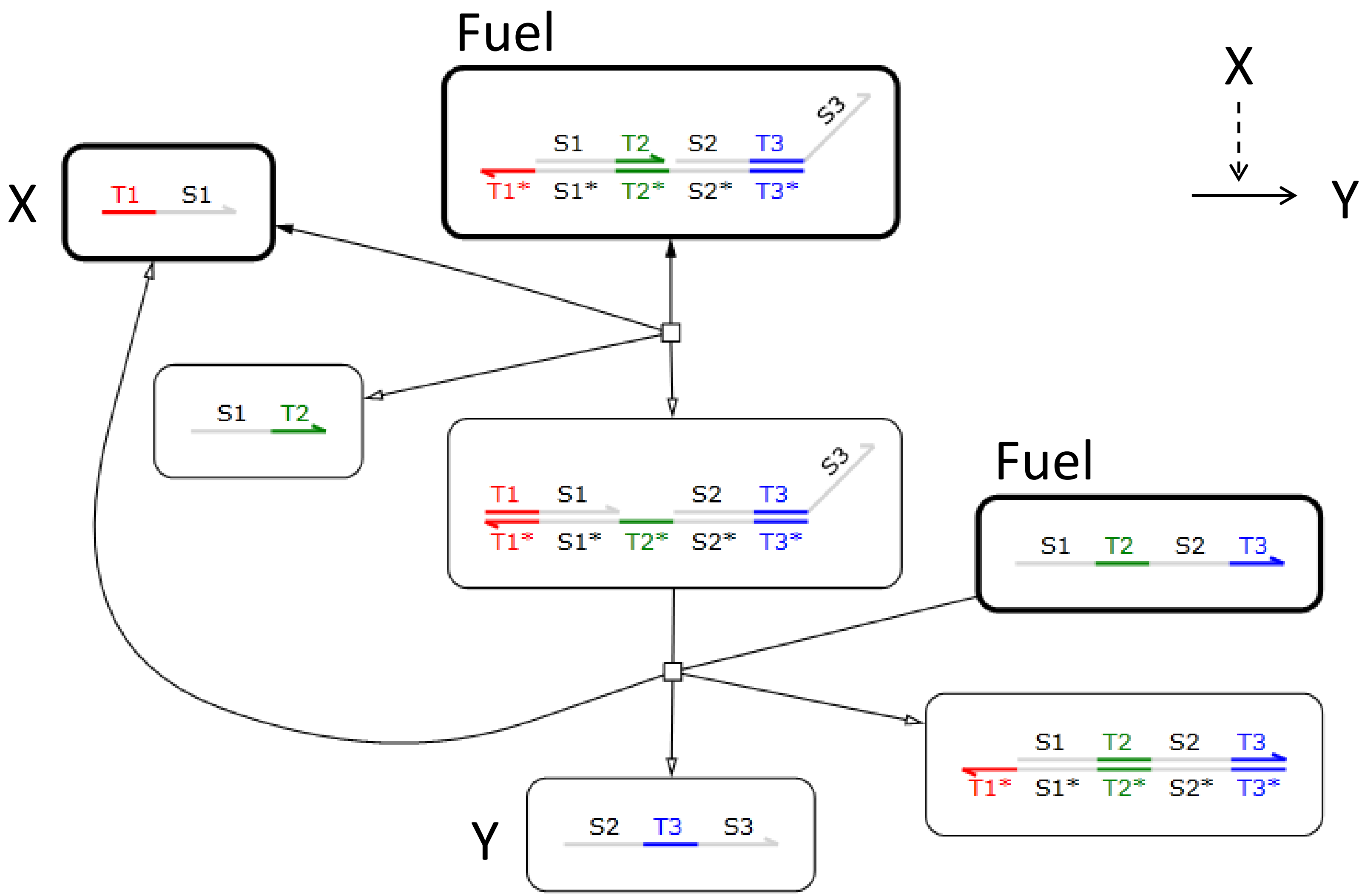




S1, S2, S3 are long domains
T1, T2, T3 are short domains

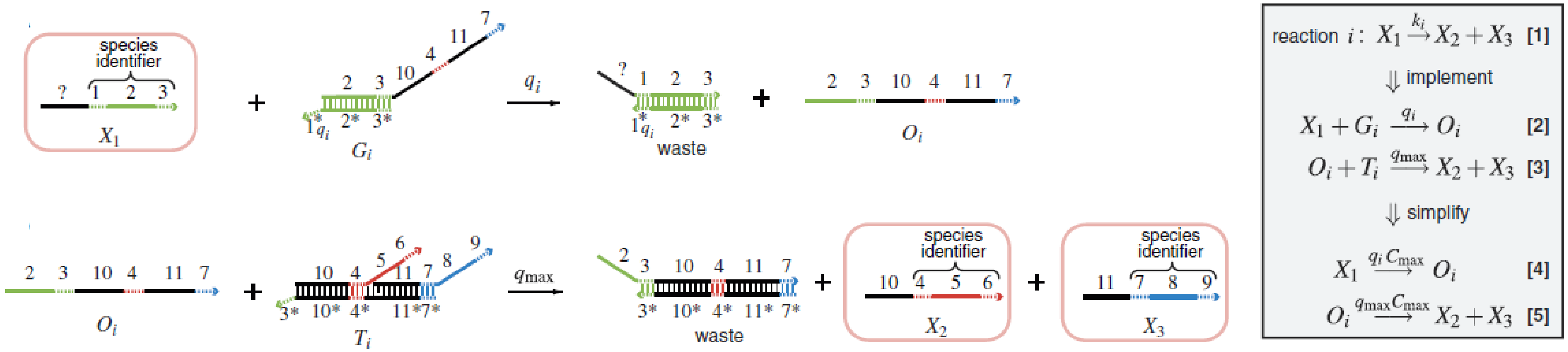
Seelig et al, *Science* 2006

Examples of simple strand displacement circuits

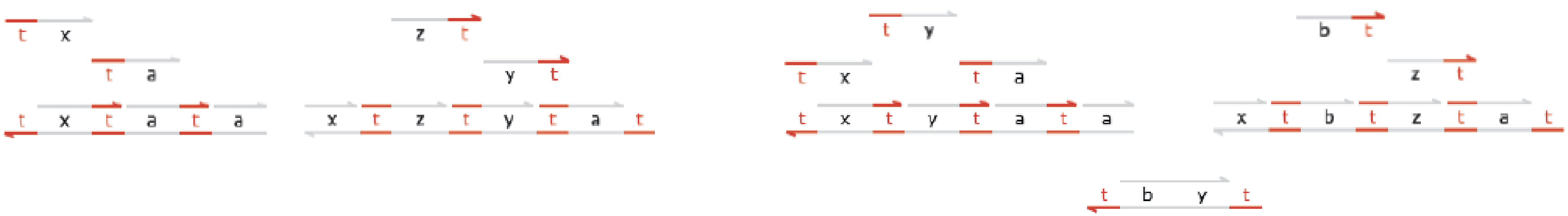


Zhang et al, *Science* 2007

Can one implement arbitrary CRNs with DNA strand displacement circuits?



Soloveichik et al, *PNAS* 2010



$\text{Fork } F_{xyz} \mid tx \to ty \mid tz$

$\text{Join } J_{xyz} \mid tx \mid ty \to tz$

Cardelli, *Math. Struct. Comput. Sci.* 2013

Conditions of a successful CRNs implementation

1. logical conditions

$$R_1 + R_2 + \cdots + R_n \rightarrow P_1 + P_2 + \cdots + P_m$$

a. The reaction pathway much first consume a molecule of each reactant, and then produce a molecule of each product.

b. The reaction pathway much first become irreversible after all reactants have been consumed and before any product has been produced.

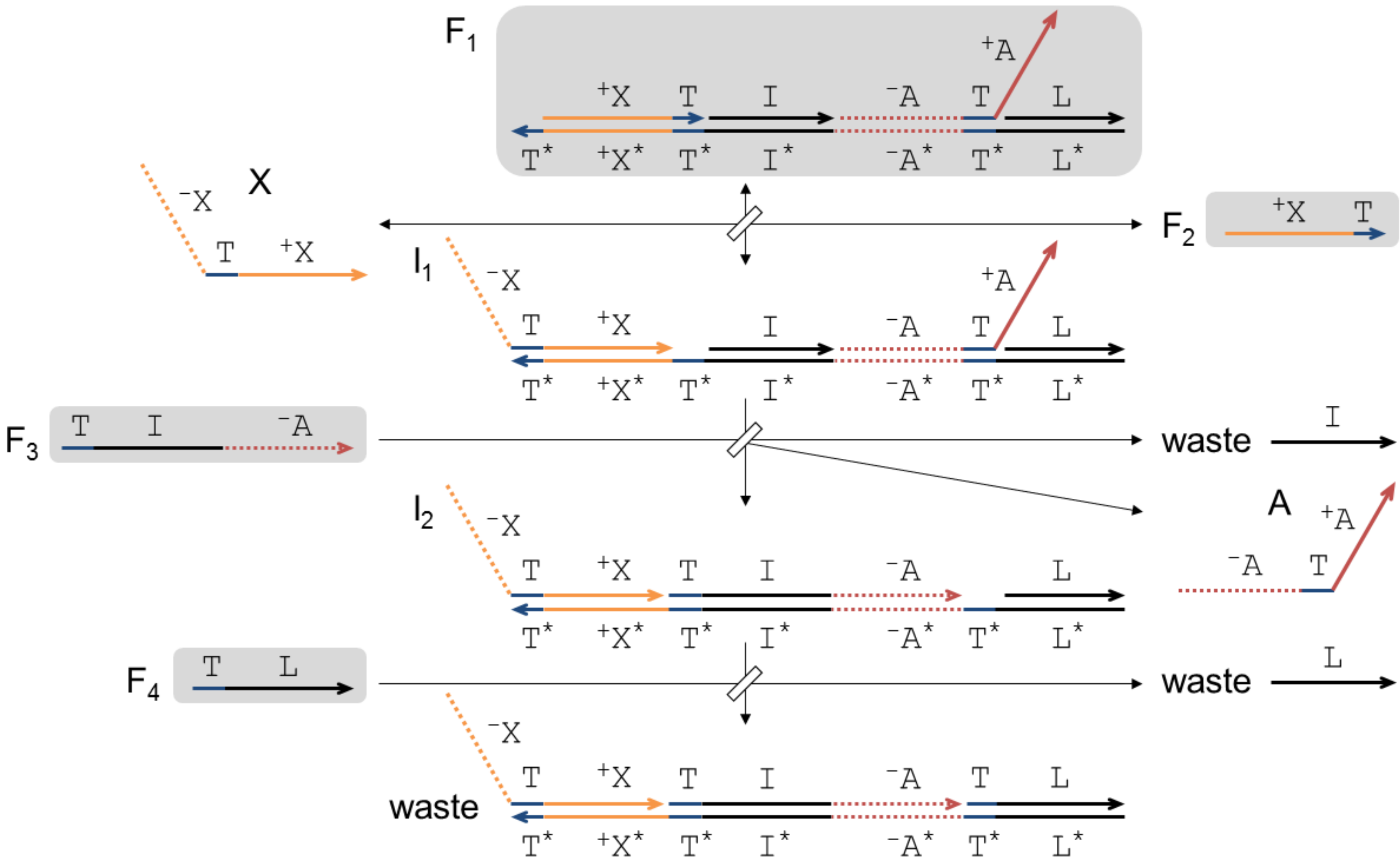
2. kinetics conditions

The rate of a reaction scales with the concentration of all reactants.

3. composable conditions

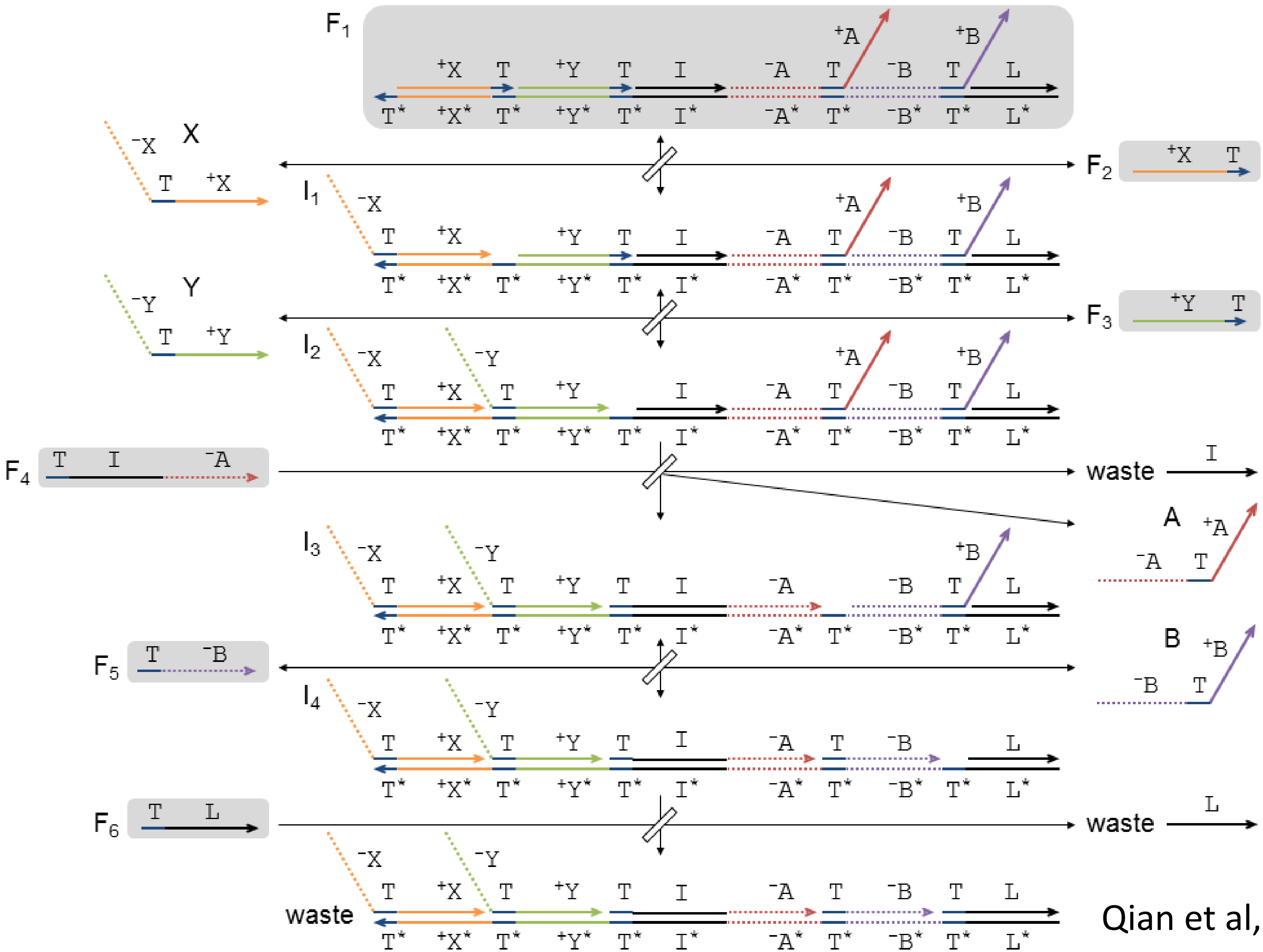
a. All chemical species are implemented with the same form.
b. No fuel or intermediate species crosstalk.

$$X \rightarrow A$$



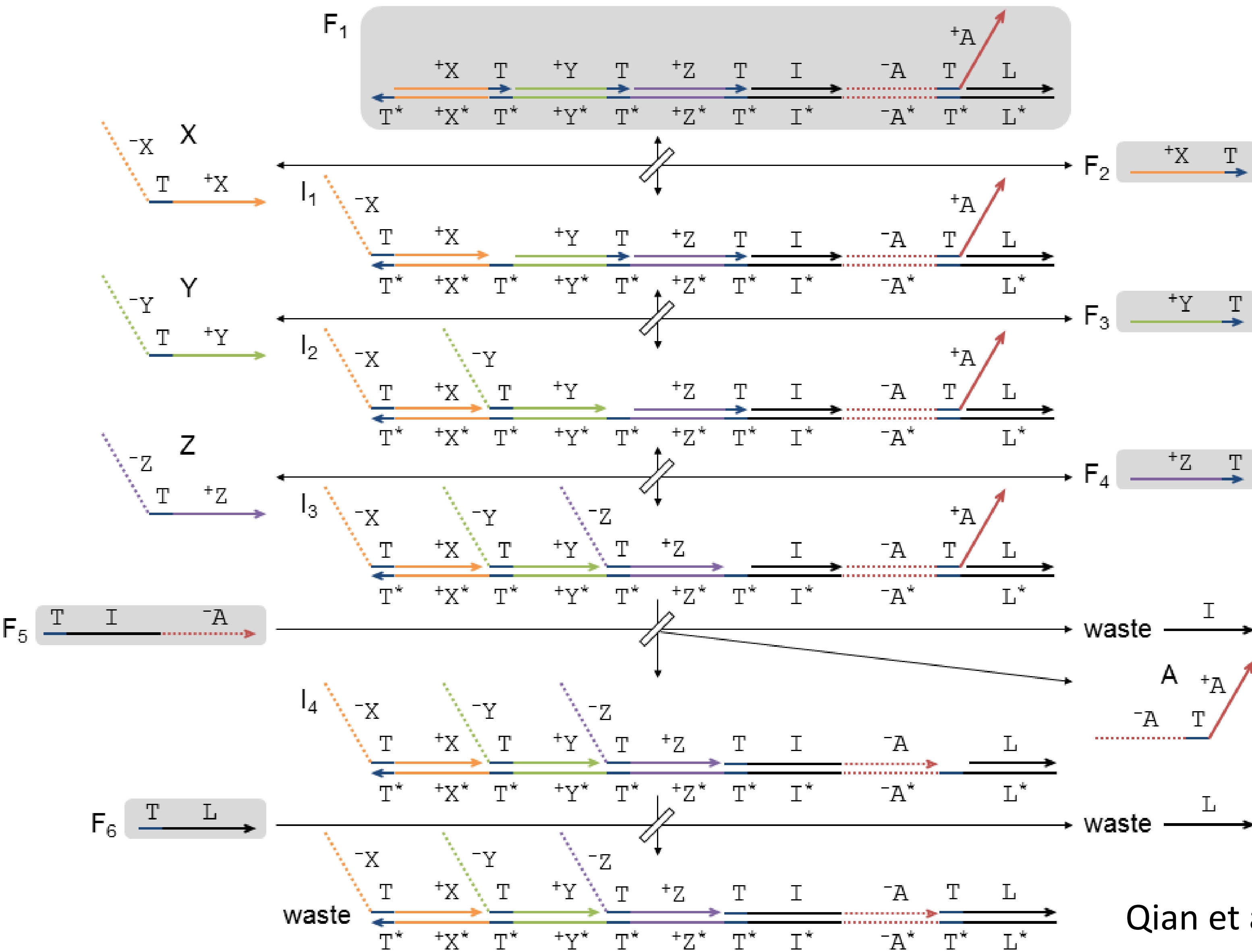
Qian et al, *LNCS* 2011

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



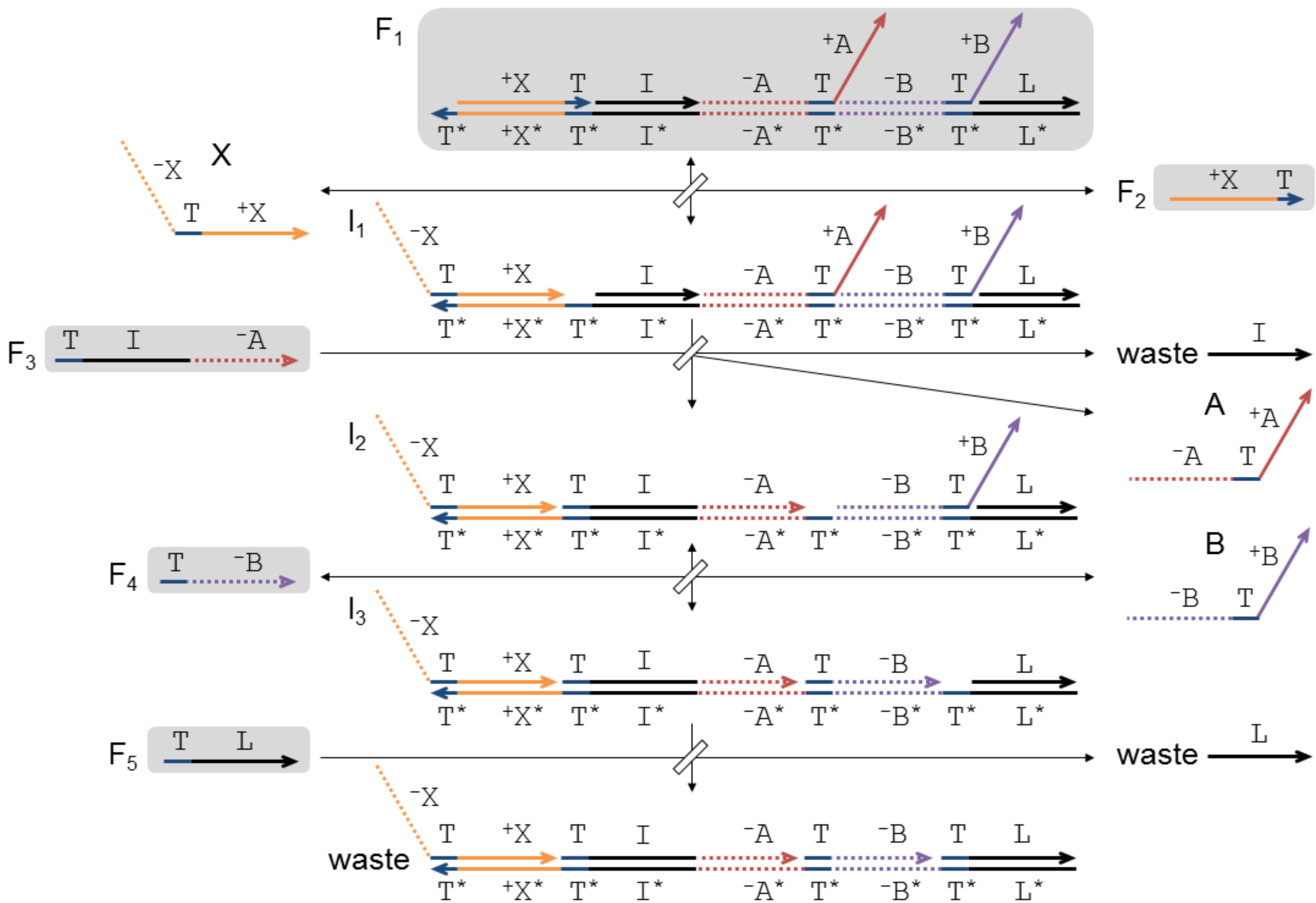
Qian et al, *LNCS* 2011

$$X + Y + Z \rightarrow A$$



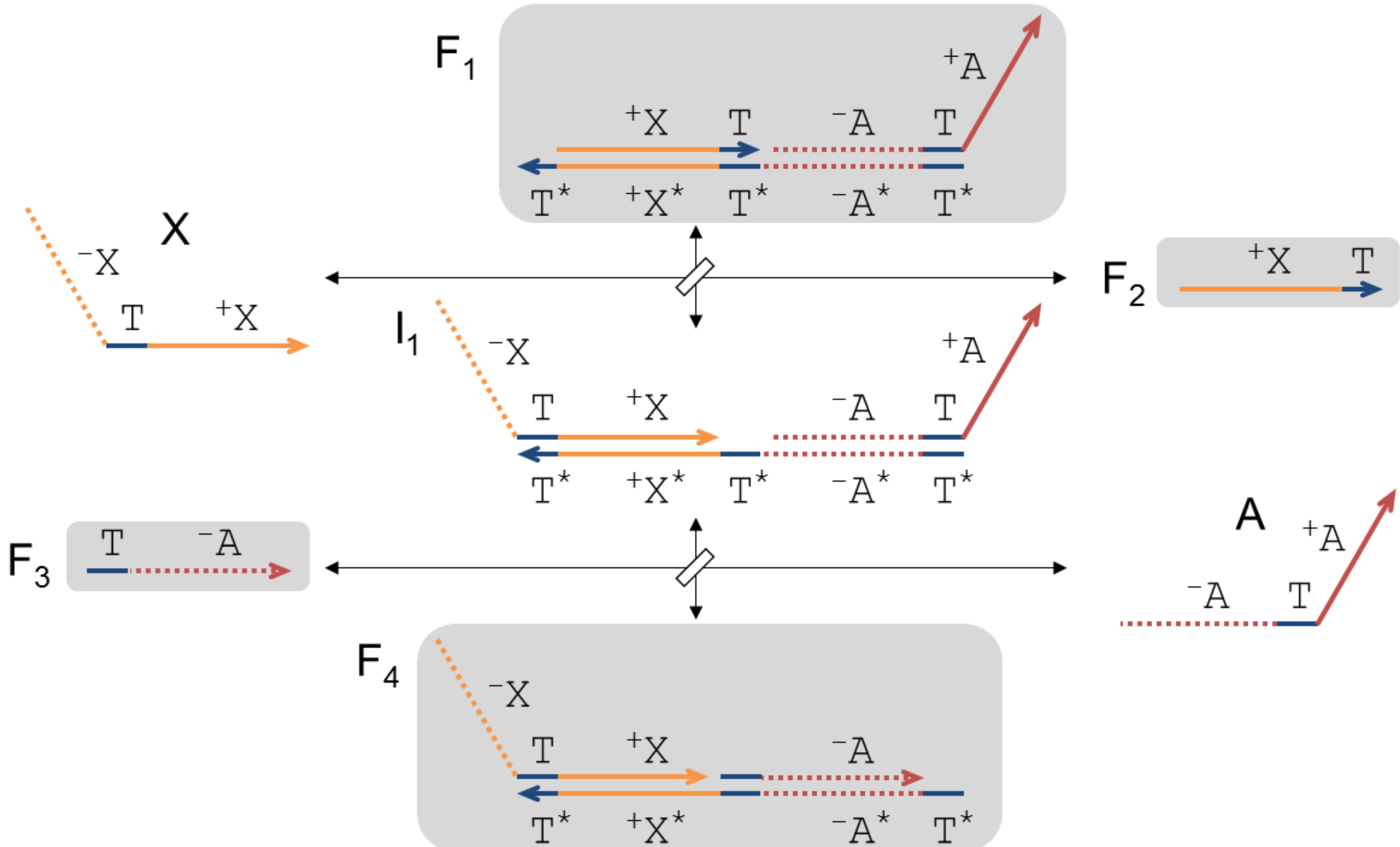
Qian et al, *LNCS* 2011

$$X \rightarrow A + B$$



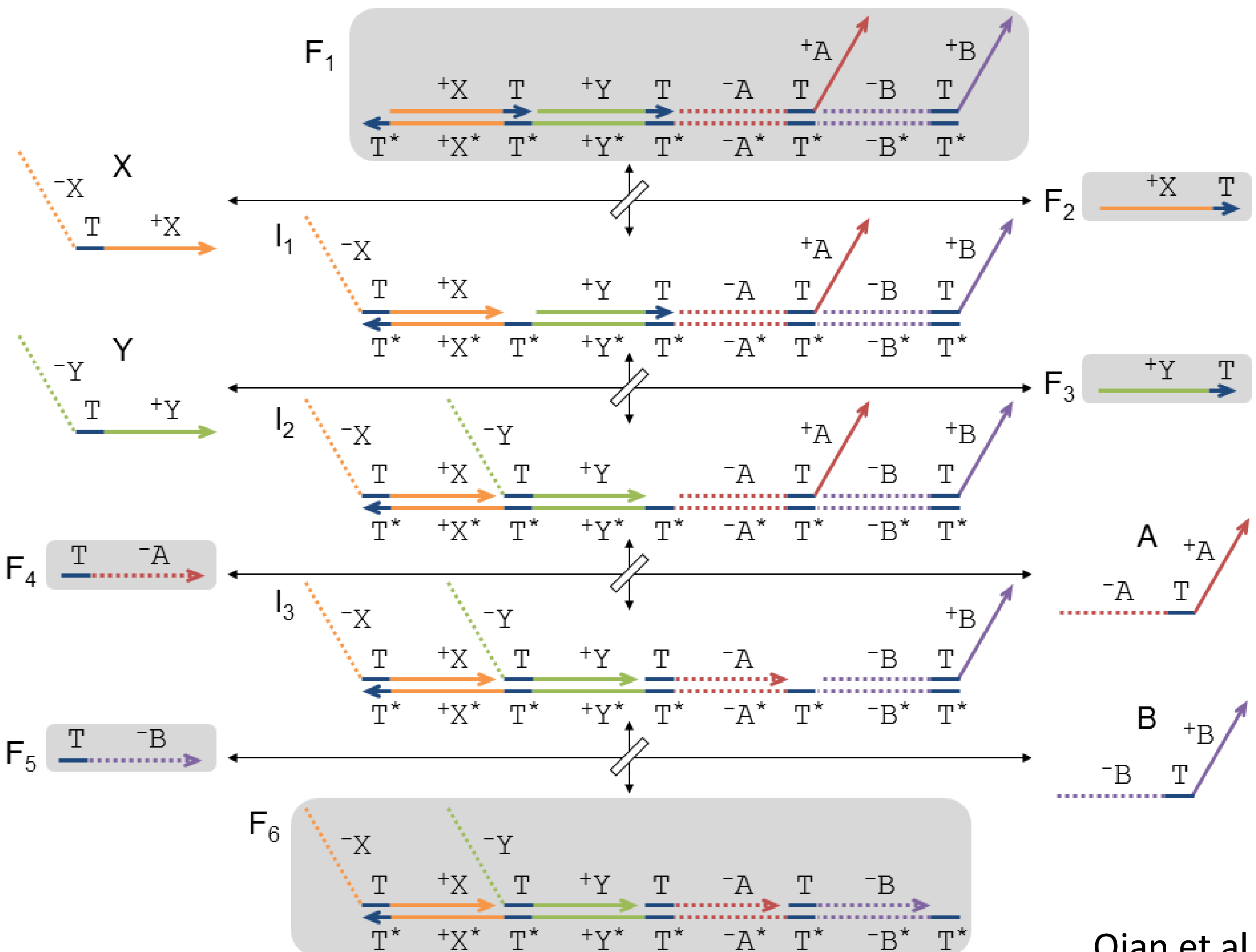
Qian et al, *LNCS* 2011

$X \leftrightarrow A$



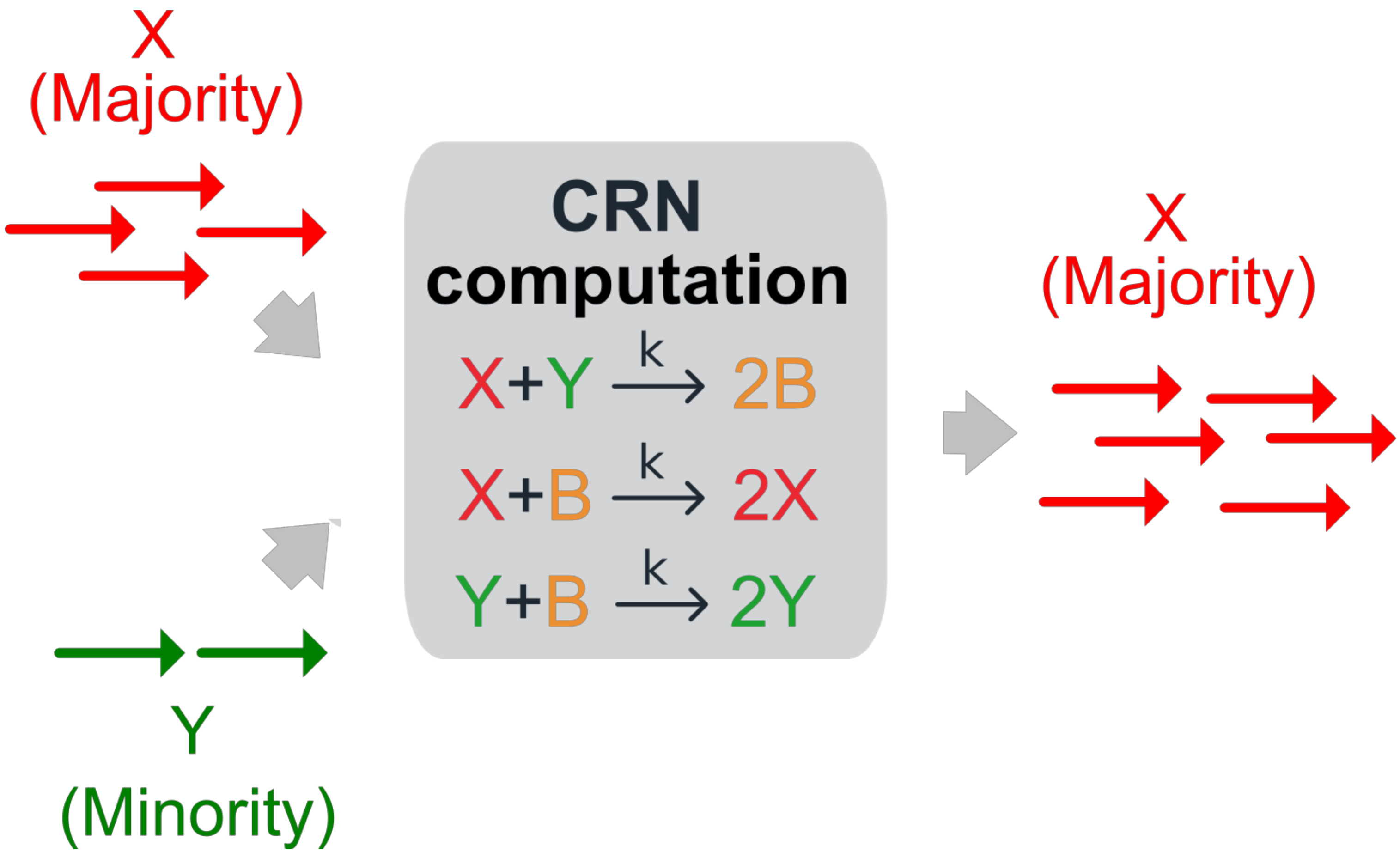
Qian et al, *LNCS* 2011

$$X + Y \leftrightarrow A + B$$



Qian et al, *LNCS* 2011

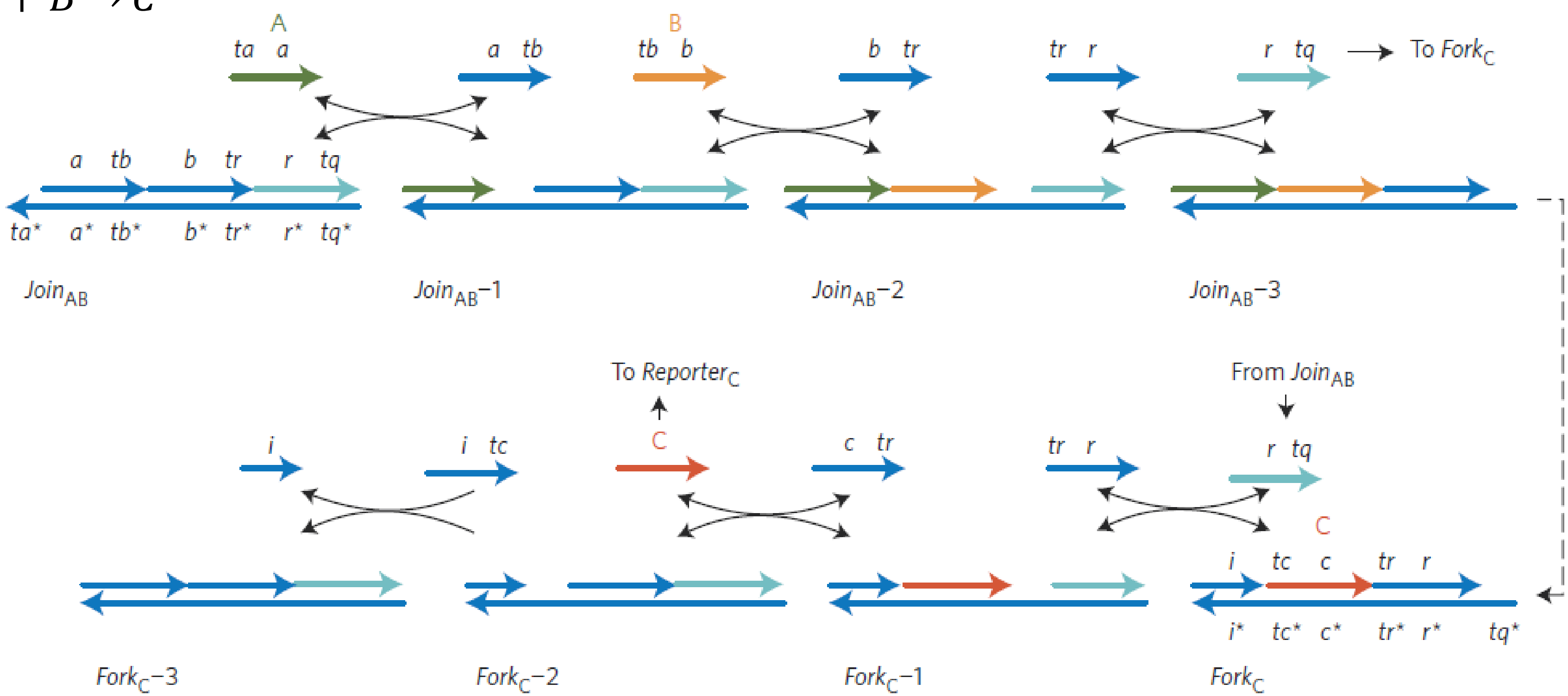
CRNs to DNA implementation: a consensus network



Chen et al, *Nature Nanotechnology* 2013

CRNs to DNA implementation: a consensus network

$$A + B \xrightarrow{k} C$$

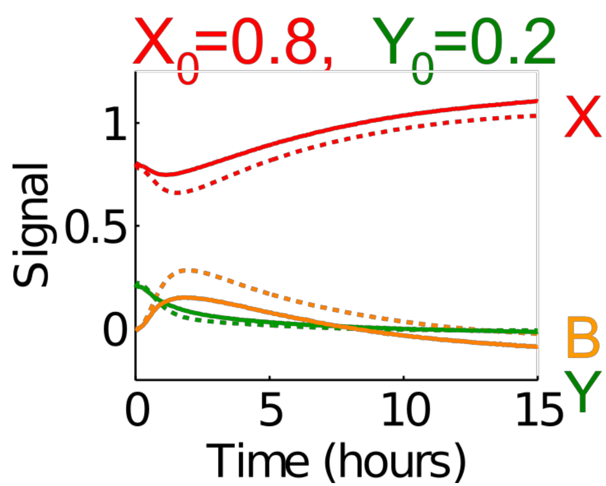


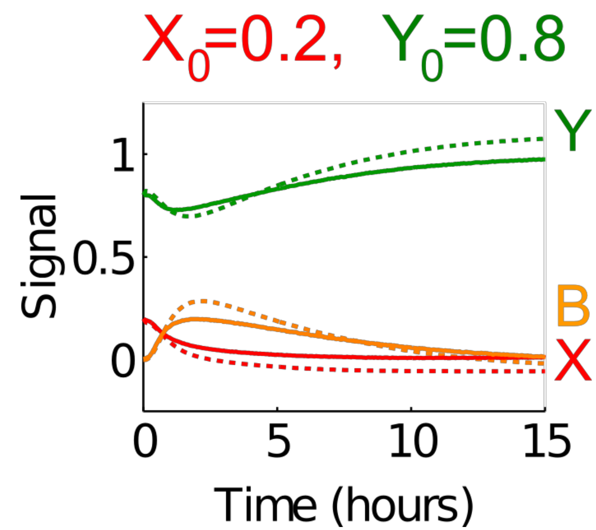
Chen et al, *Nature Nanotechnology* 2013

CRNs to DNA implementation: a consensus network

CRN computation

$$X+Y \xrightarrow{k} 2B$$
$$X+B \xrightarrow{k} 2X$$
$$Y+B \xrightarrow{k} 2Y$$





Dashed lines: simulations
Solid lines: experiments

Chen et al, *Nature Nanotechnology* 2013

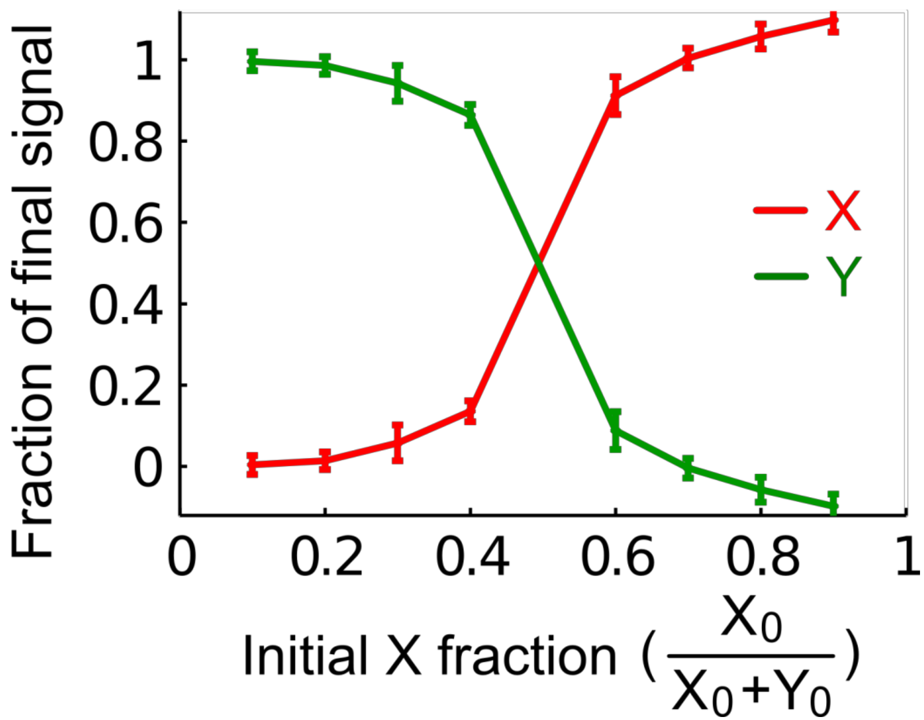
CRNs to DNA implementation: a consensus network

CRN computation

$$X + Y \xrightarrow{k} 2B$$

$$X + B \xrightarrow{k} 2X$$

$$Y + B \xrightarrow{k} 2Y$$

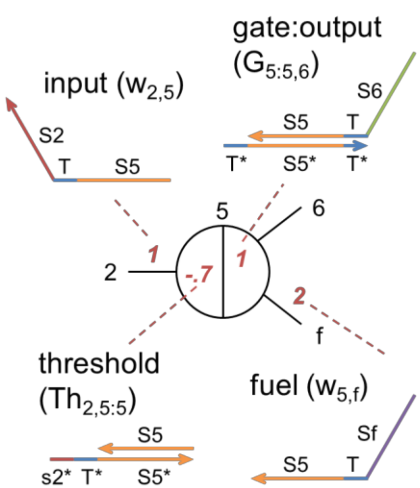


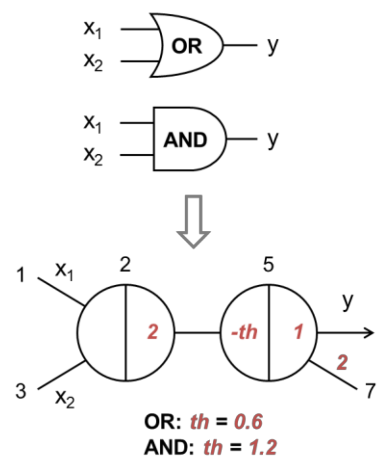
Chen et al, *Nature Nanotechnology* 2013

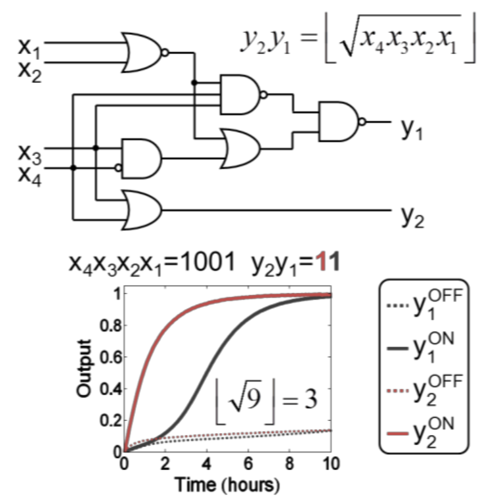
What is the simplest DNA building blocks for creating CRNs with complex behaviors?

How robustly can DNA-based CRNs scale up?

$X + G{:}Y \overset{ks}{\leftrightarrow} X{:}G + Y \qquad X + Th \overset{kf}{\rightarrow} \emptyset$



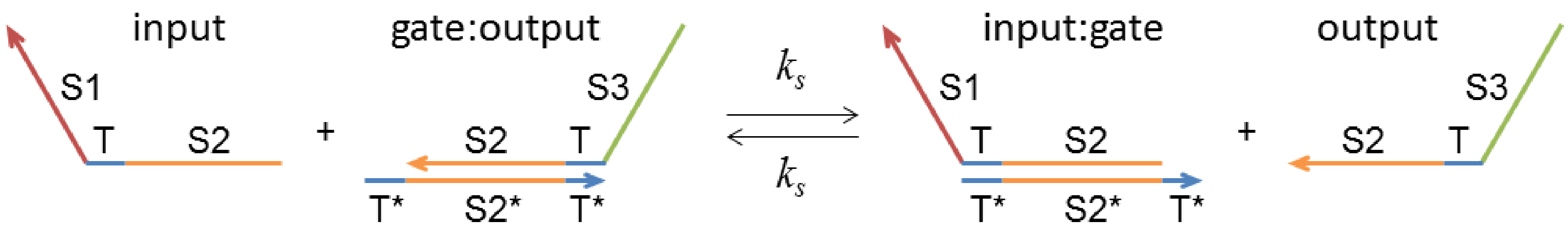




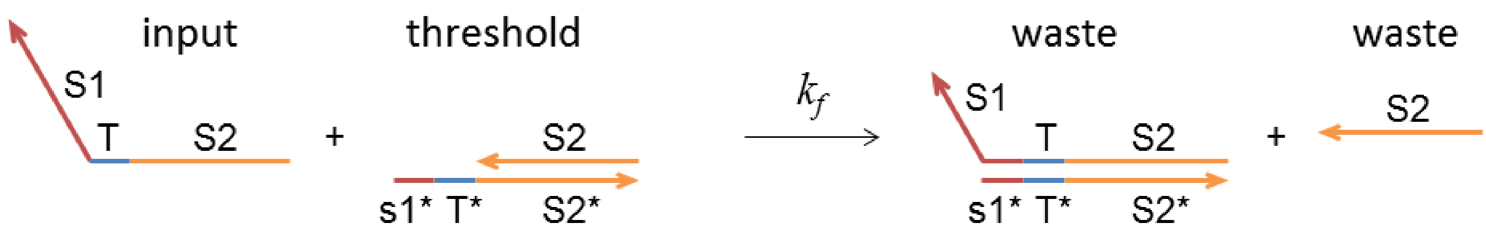
Qian et al, *Science* 2011

Basic reactions in seesaw networks

1. seesawing



2. thresholding



|T*| = 5nt $\quad k_s \approx 10^5/\mathrm{M/s}$

|s1* T*| = 8nt $\quad k_f \approx 10^6/\mathrm{M/s}$

$k_f \gg k_s$

$k_{eff} \approx 10^L/\mathrm{M/s}$ when $L \leq 6$

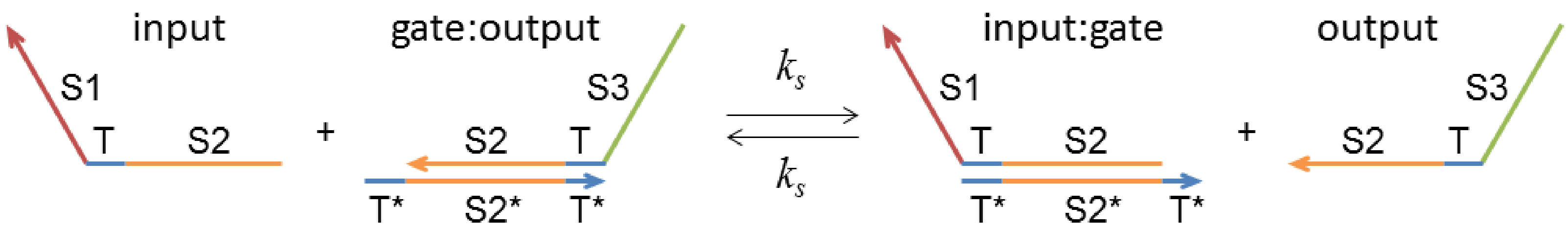
otherwise $k_{eff} \approx 10^6/\mathrm{M/s}$

L: toehold length

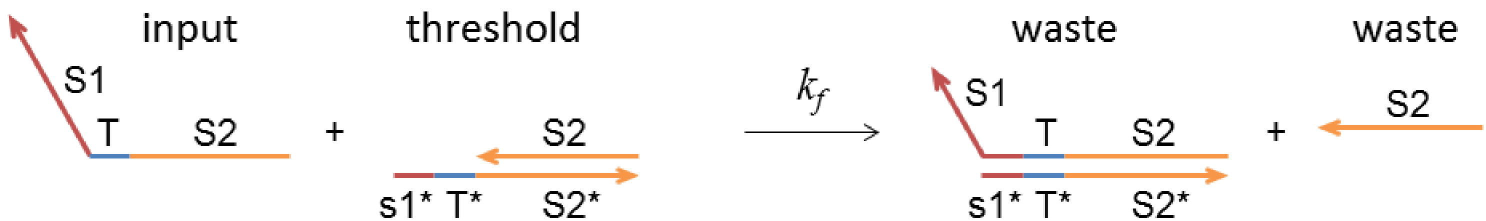
$$\begin{cases} \text{if [input]}\,|_{t=0} < \text{[threshold]}\,|_{t=0} & \text{[output]}|_{t\to\infty} \approx 0 \\ \text{if [input]}\,|_{t=0} > \text{[threshold]}\,|_{t=0} & \text{[output]}|_{t\to\infty} \gg 0 \end{cases}$$

Basic reactions in seesaw networks

1. seesawing

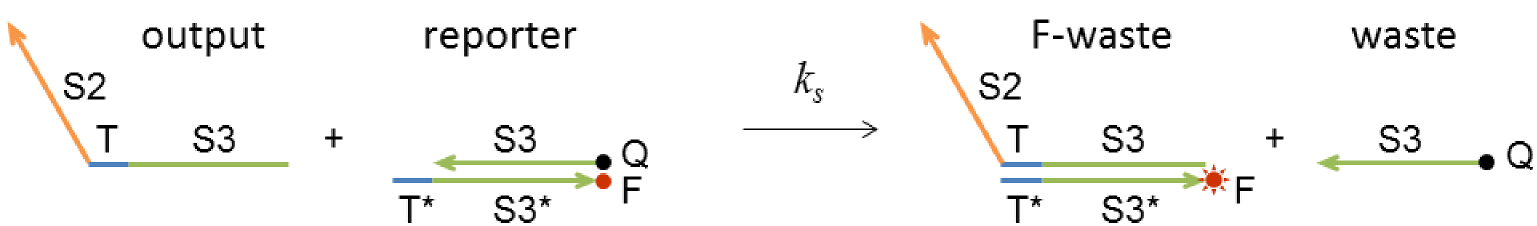


2. thresholding



3. reporting

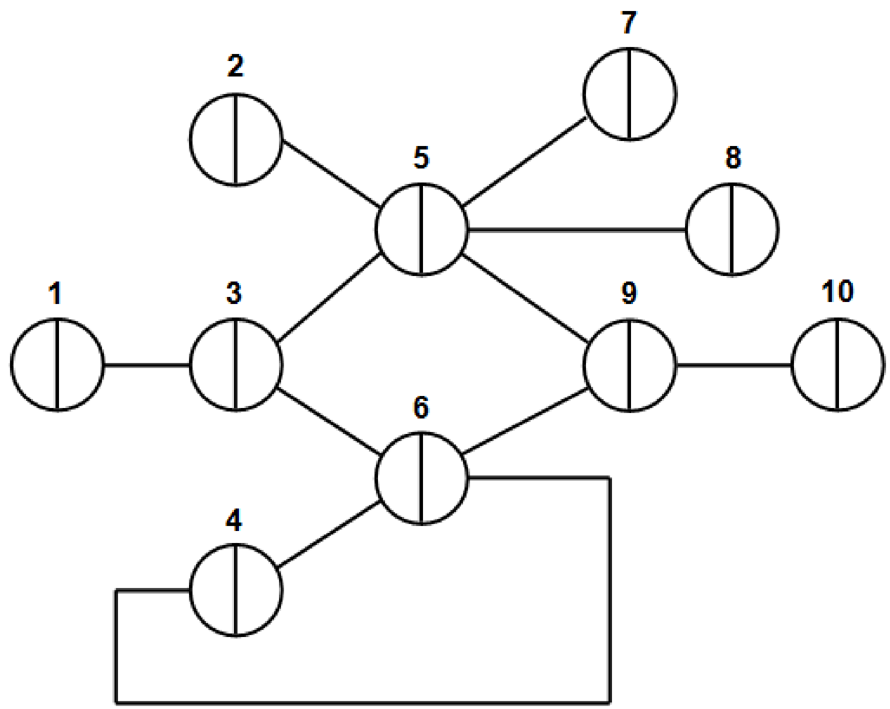
read the output of the computation with fluorescence signal



Seesaw abstraction

A seesaw network has a number of two-sided nodes and a number of wires. Each node can be connected to any number of wires on each side. Each wire connects exactly two nodes. Each node has an identity: $i, i \in \{1, 2, 3, \dots\}$

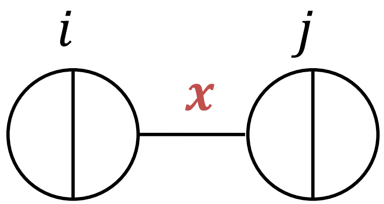
Example:



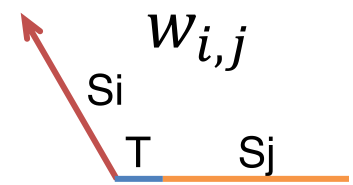
Seesaw abstraction

1. free signal

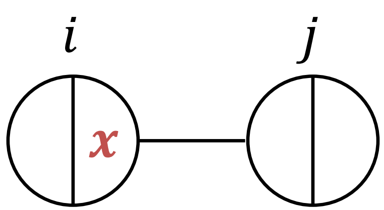
x is relative to a standard concentration (e.g. 100nM)



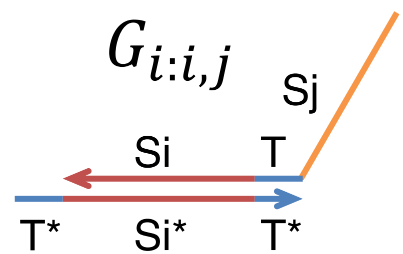
$$[w_{i,j}]\big|_{t=0} = x$$



2. bound signal

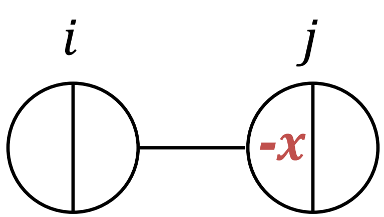


$$[G_{i:i,j}]\big|_{t=0} = x$$

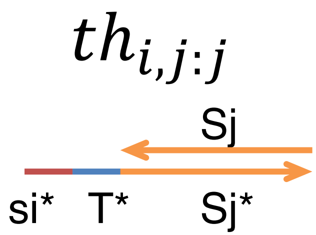


signal $w_{i,j}$ bound to the right side of gate i

3. threshold



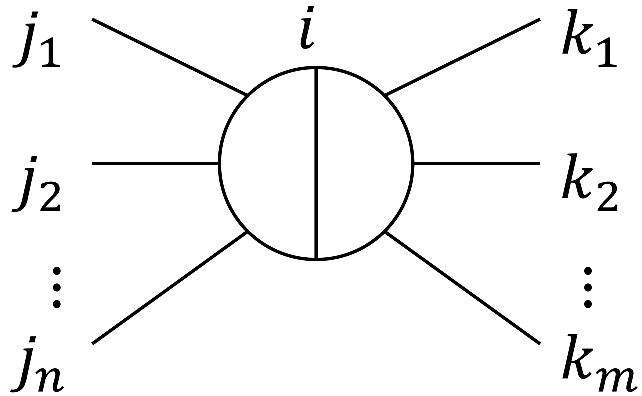
$$[th_{i,j:j}]\big|_{t=0} = x$$



threshold on gate j to absorb signal $w_{i,j}$

Seesaw abstraction

For any node i in a seesaw network:



for all $j \in \{ j_1, j_2, \ldots, j_n \}$ and $k \in \{ k_1, k_2, \ldots, k_m \}$

$$w_{j,i} + G_{i:i,k} \overset{k_s}{\longleftrightarrow} G_{j,i:i} + w_{i,k}$$

$$w_{j,i} + th_{j,i:i} \xrightarrow{k_f} \emptyset$$

Seesaw abstraction

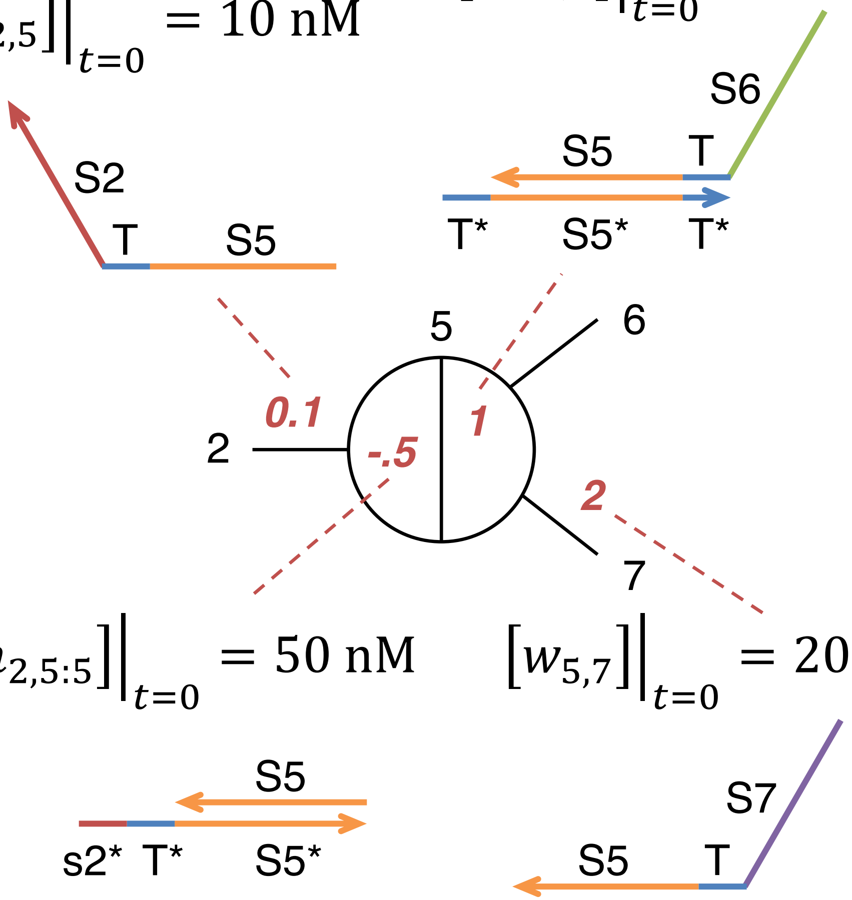
Example: standard concentration is 100 nM

$$[w_{2,5}]\big|_{t=0} = 10\ \text{nM}$$

$$[G_{5:5,6}]\big|_{t=0} = 100\ \text{nM}$$

$$[Th_{2,5:5}]\big|_{t=0} = 50\ \text{nM}$$

$$[w_{5,7}]\big|_{t=0} = 200\ \text{nM}$$



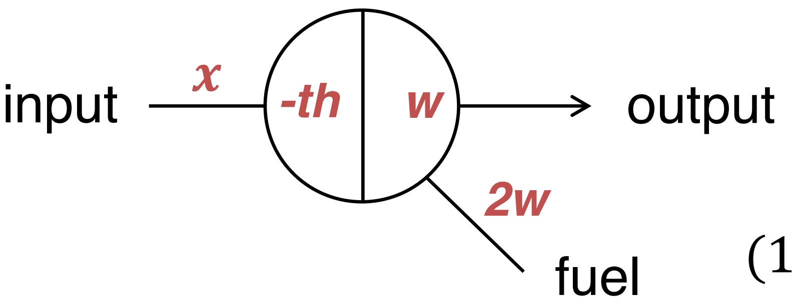
$$w_{2,5} + G_{5:5,6} \overset{k_s}{\longleftrightarrow} G_{2,5:5} + w_{5,6}$$

$$w_{2,5} + G_{5:5,7} \overset{k_s}{\longleftrightarrow} G_{2,5:7} + w_{5,7}$$

$$w_{2,5} + th_{2,5:5} \overset{k_f}{\longrightarrow} \emptyset$$

Two types of seesaw gates

1. amplifying gate



$$[\text{input}]\Big|_{t=0} = x \quad [\text{threshold}]\Big|_{t=0} = th$$

$$[\text{gate:output}]\Big|_{t=0} = w \quad [\text{fuel}]\Big|_{t=0} = 2w$$

$$(1)\ \text{input} + \text{threshold} \xrightarrow{k_f} \emptyset$$

$$(2)\ \text{input} + \text{gate:output} \xrightarrow{k_s} \text{input:gate} + \text{output}$$

$$(3)\ \text{input} + \text{gate:fuel} \xleftrightarrow{k_s} \text{input:gate} + \text{fuel}$$

pathway (net effect) of (2) and (3) with the above initial conditions:

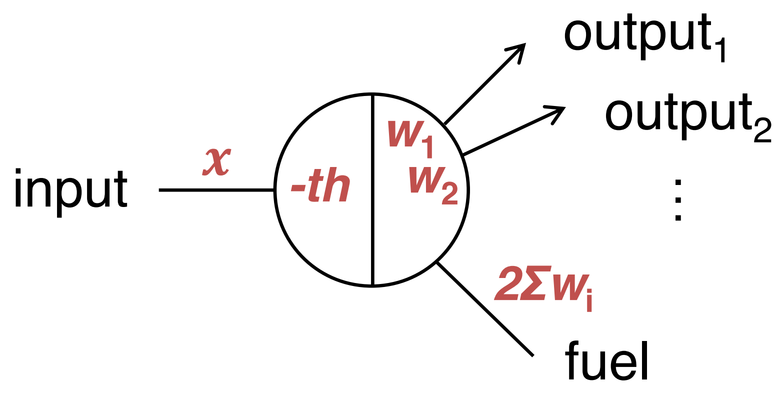
$$\text{input} + \text{gate:output} + \text{fuel} \longrightarrow \text{input} + \text{gate: fuel} + \text{output}$$

simplify: $\text{input} \longrightarrow \text{input} + \text{output}$ when supply of gate:output and fuel last

$$\begin{cases} \text{if } x \leq th, [\text{output}]|_{t\to\infty} = 0 \\ \text{if } x > th, [\text{output}]|_{t\to\infty} = w \end{cases}$$

Two types of seesaw gates

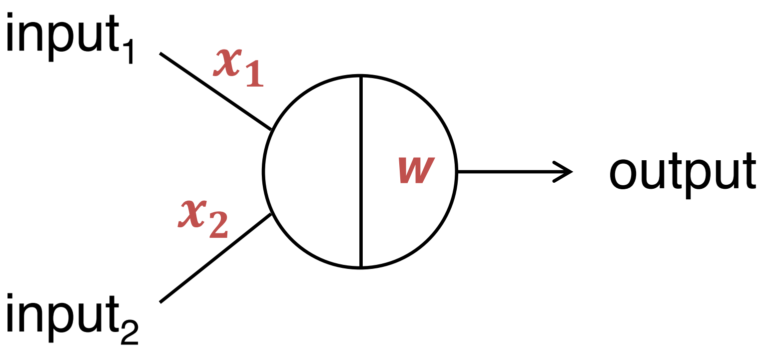
1. amplifying gate



$$\begin{cases} \text{if } x \le th, \ [\text{output}_1]|_{t\to\infty} = 0, \ [\text{output}_2]|_{t\to\infty} = 0, \ldots \\ \text{if } x > th, \ [\text{output}_1]|_{t\to\infty} = w_1, \ [\text{output}_2]|_{t\to\infty} = w_2, \ldots \end{cases}$$

Two types of seesaw gates

2. integrating gate



$w > x_1 + x_2$

$$[\text{input}_1]\big|_{t=0} = x_1$$

$$[\text{input}_2]\big|_{t=0} = x_2$$

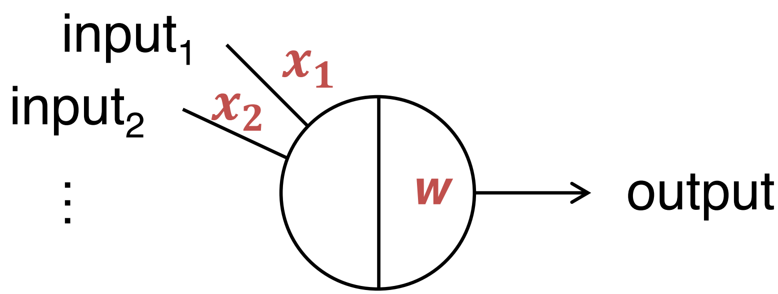
$$[\text{gate:output}]\big|_{t=0} = w$$

$$\begin{cases} \text{input}_1 + \text{gate:output} \xrightarrow{k_s} \text{input}_1\text{:gate} + \text{output} \\ \text{input}_2 + \text{gate:output} \xrightarrow{k_s} \text{input}_2\text{:gate} + \text{output} \end{cases}$$

$$\Rightarrow [\text{output}]\big|_{t\to\infty} = x_1 + x_2$$

Two types of seesaw gates

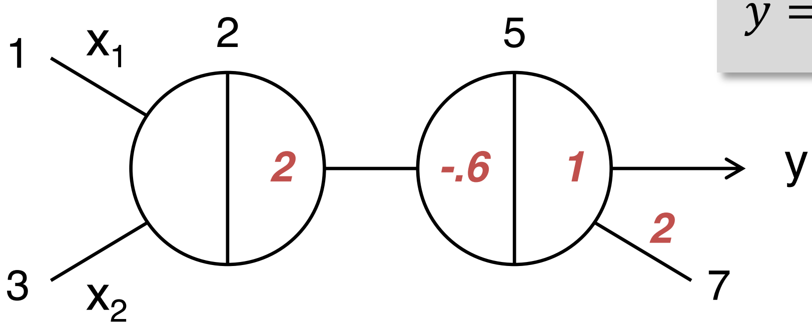
2. integrating gate



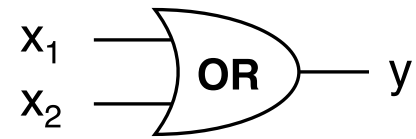
$$w > \sum x_i$$

$$[\text{output}]\Big|_{t\to\infty} = \sum x_i$$

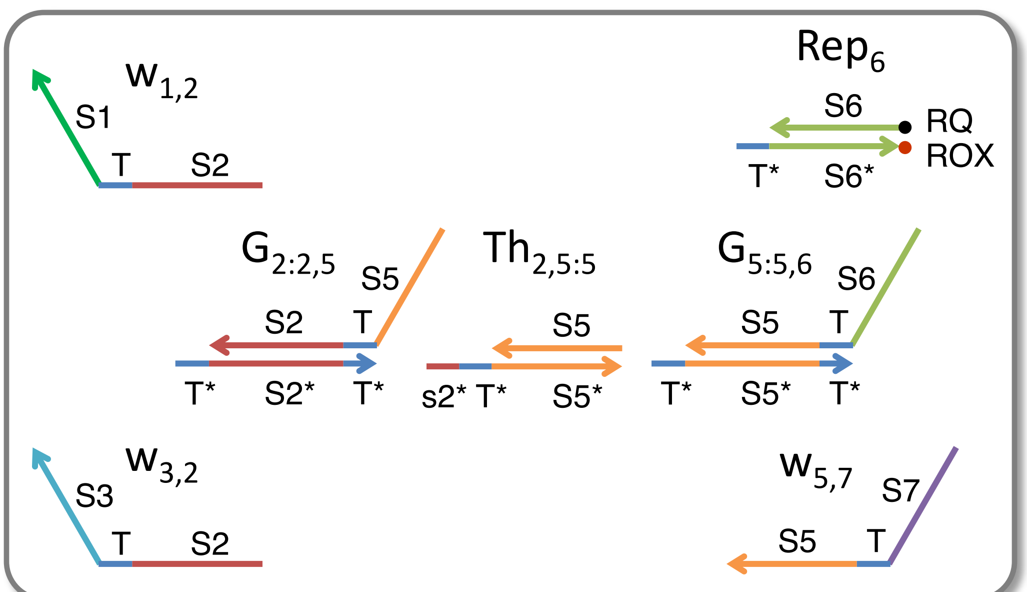
Logic gates

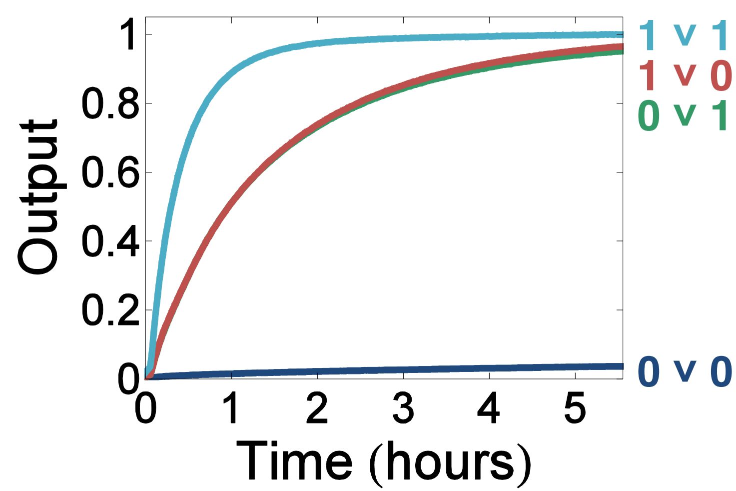


$$y = \begin{cases} 1 & x_1 + x_2 > 0.6 \\ 0 & x_1 + x_2 \leq 0.6 \end{cases}$$



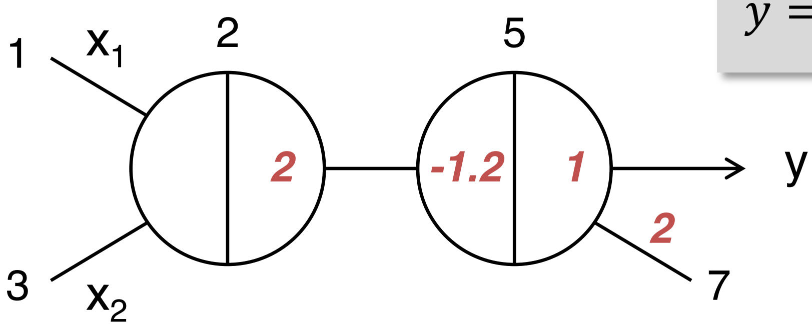
OFF: 0 ~ 0.2
ON: 0.8 ~ 1



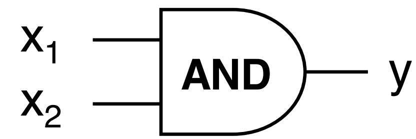


0=0.1x 1=0.9x 1x = 100 nM

Logic gates

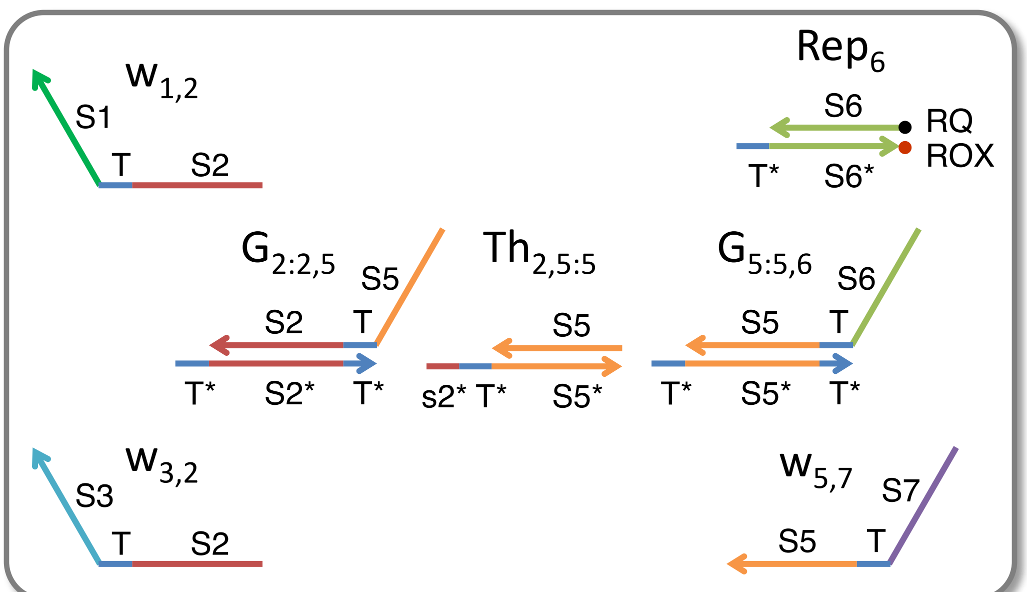


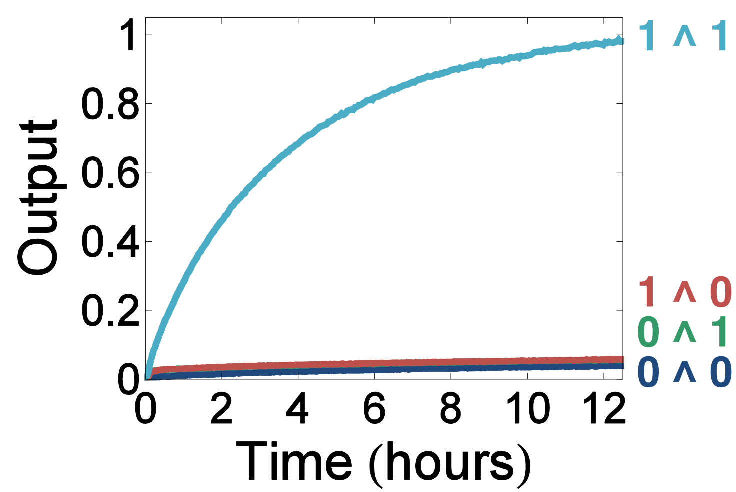
$$y = \begin{cases} 1 & x_1 + x_2 > 1.2 \\ 0 & x_1 + x_2 \leq 1.2 \end{cases}$$



OFF: 0 ~ 0.2

ON: 0.8 ~ 1

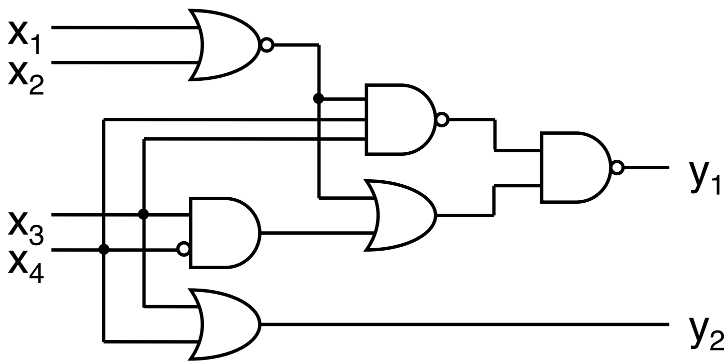




0=0.1x 1=0.9x 1x = 100 nM

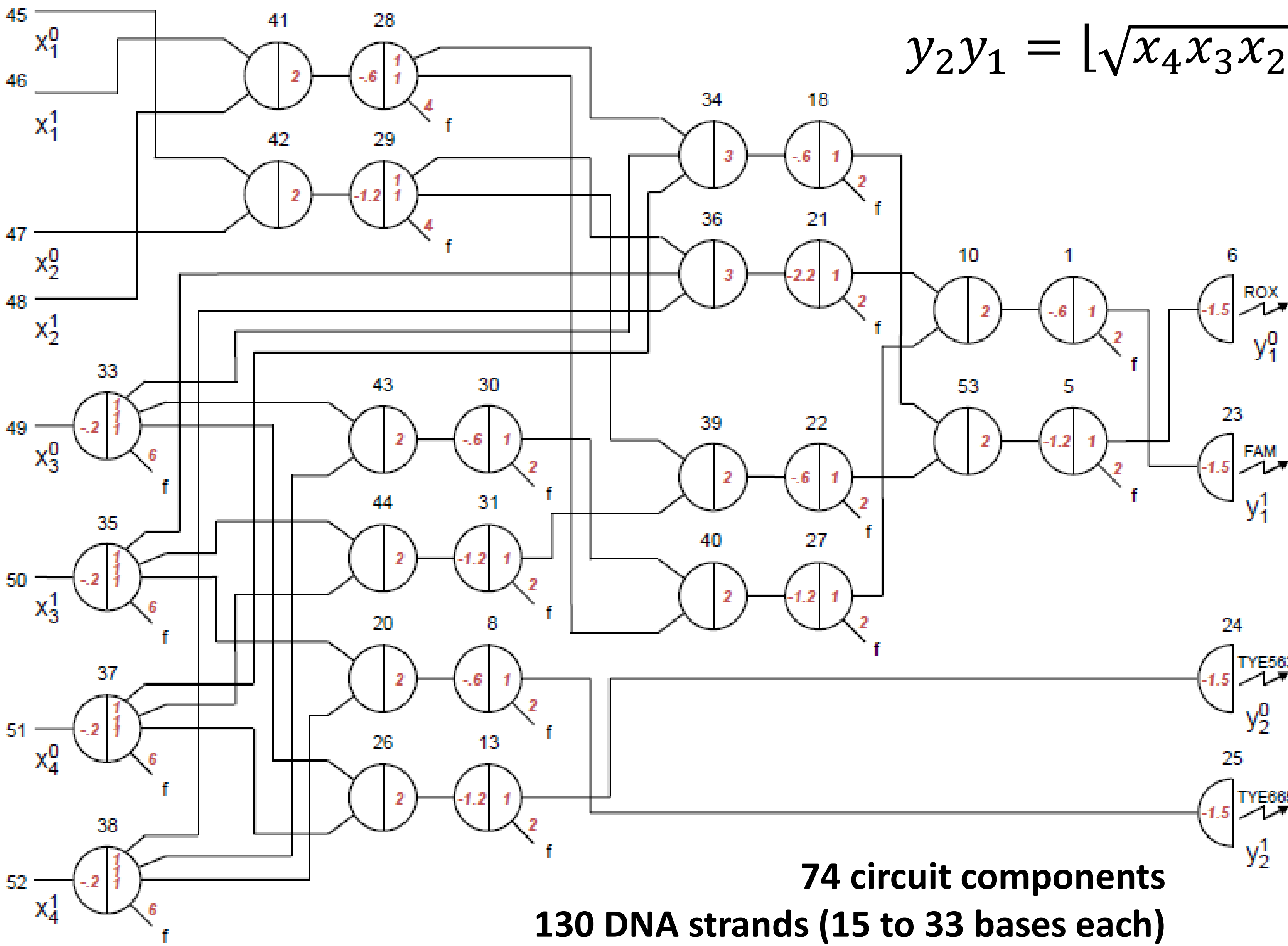
A four-bit square root circuit

$$y_2 y_1 = \lfloor \sqrt{x_4 x_3 x_2 x_1} \rfloor$$



A four-bit square root circuit

$$y_2 y_1 = \lfloor \sqrt{x_4 x_3 x_2 x_1} \rfloor$$

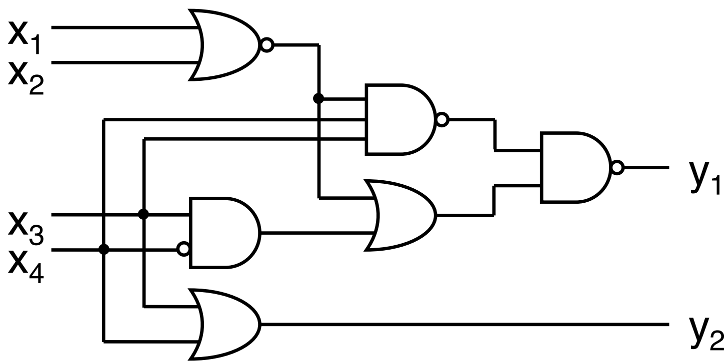


74 circuit components

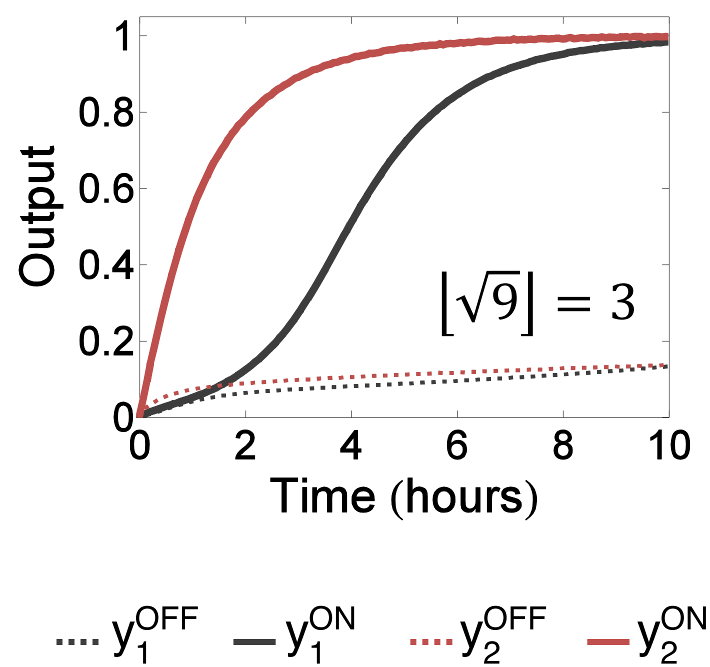
130 DNA strands (15 to 33 bases each)

A four-bit square root circuit

$$y_2 y_1 = \lfloor \sqrt{x_4 x_3 x_2 x_1} \rfloor$$

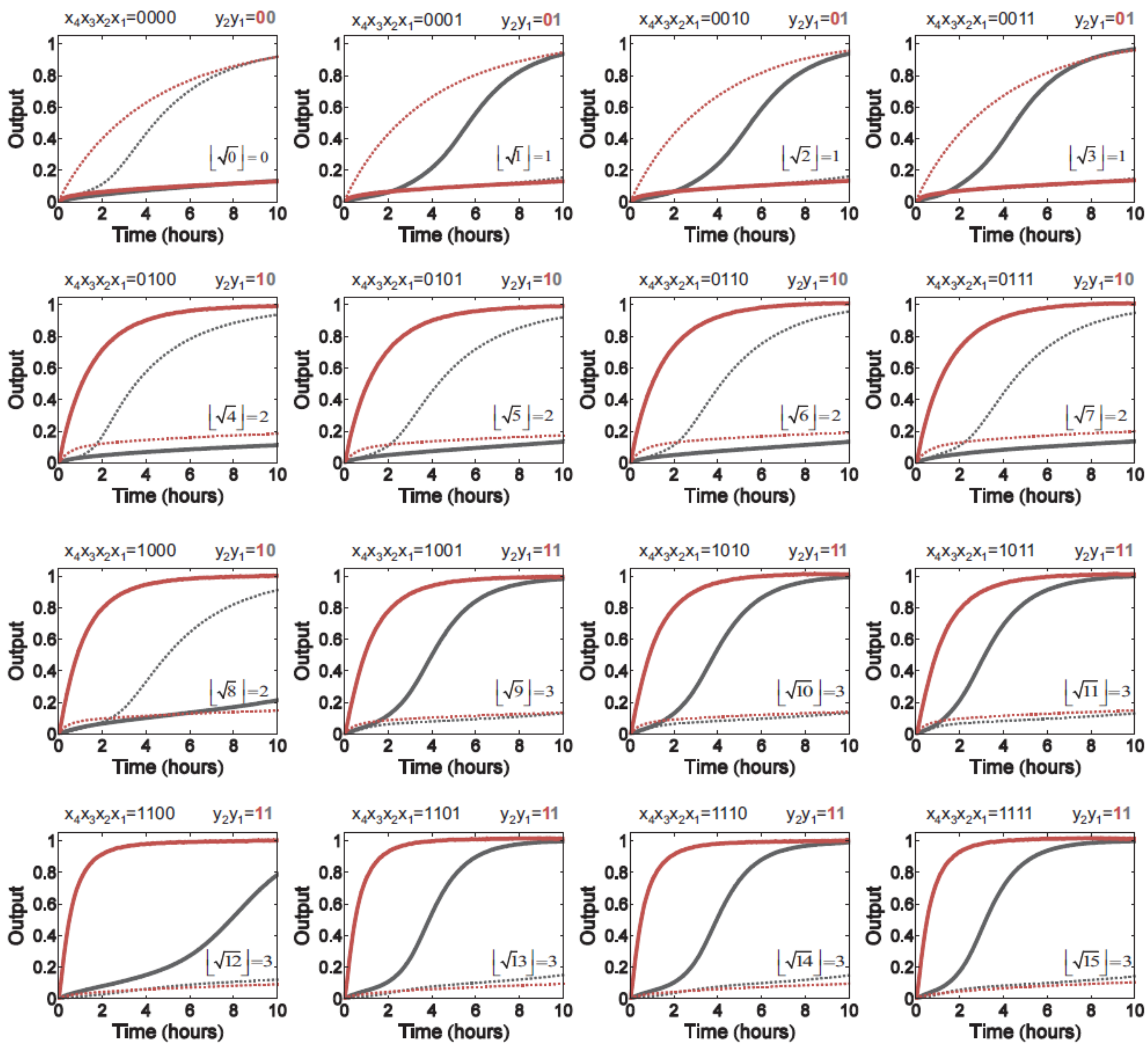


$x_4x_3x_2x_1$=1001 y_2y_1=11



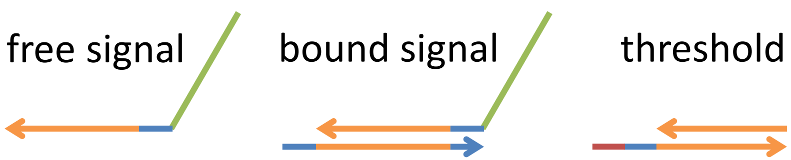
0=0.1x 1=0.9x 1x = 50 nM

A four-bit square root circuit

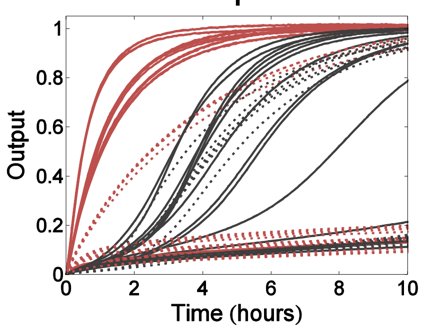


Simplicity and robustness

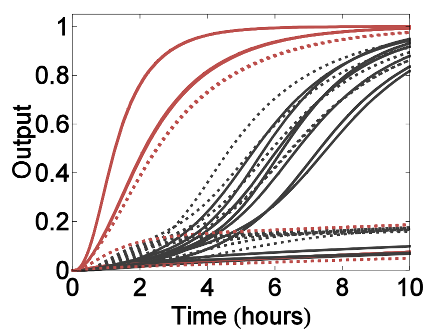
single- or double-stranded circuit components



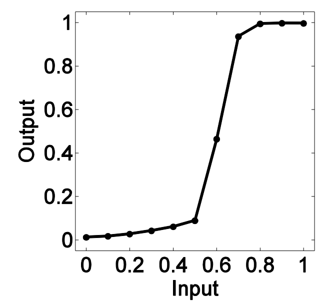
experiments

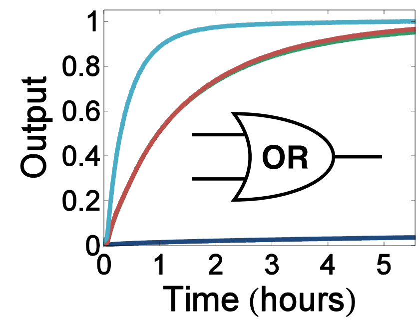


simulations

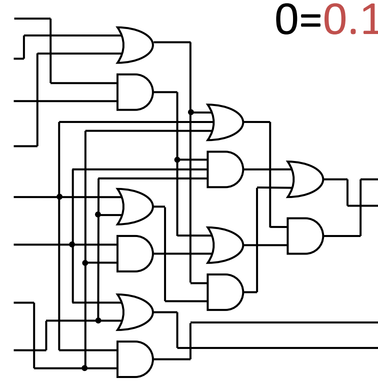


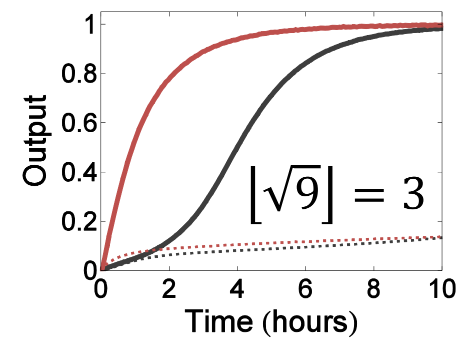
circuit performance stays roughly the same with size

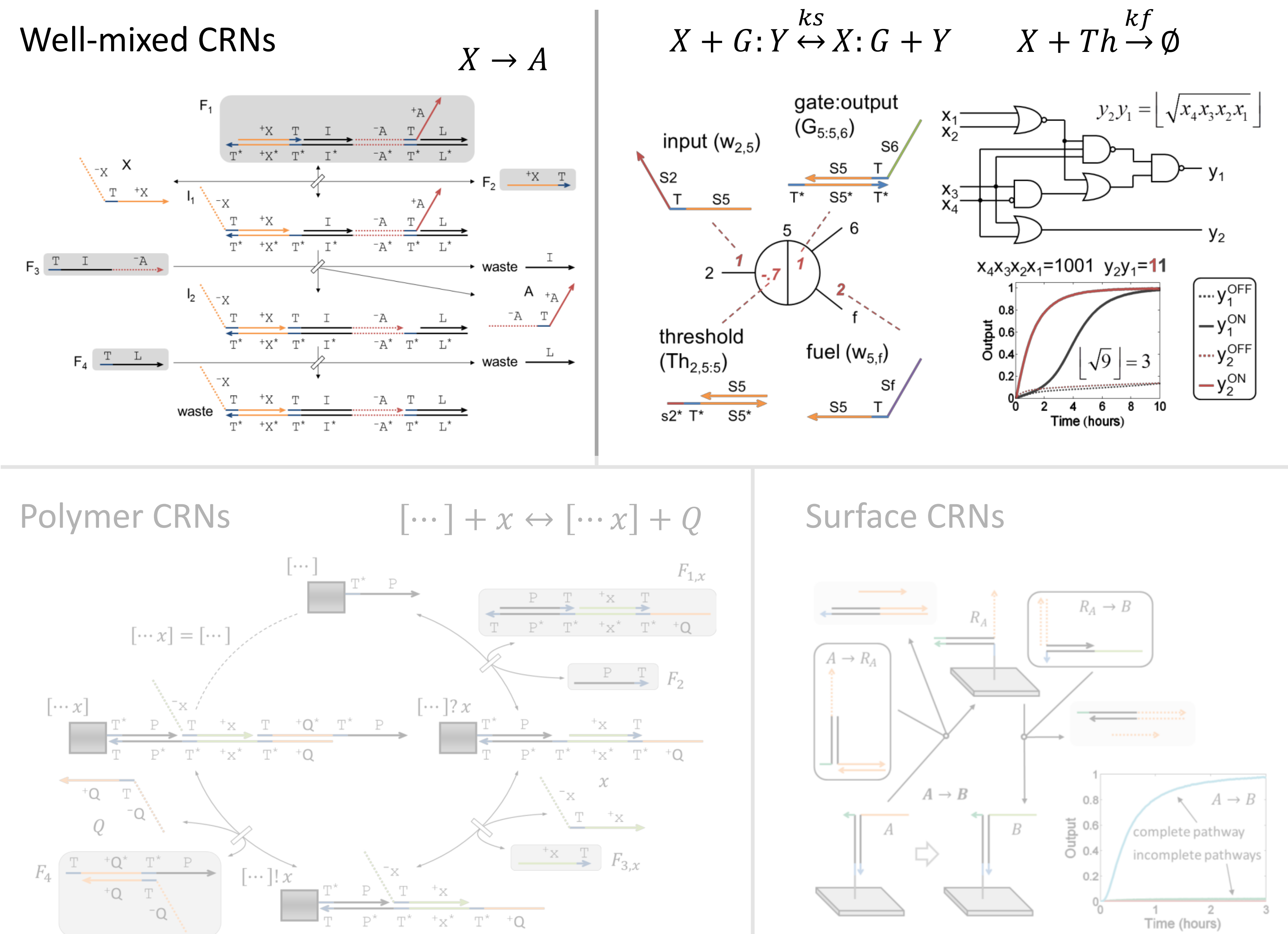


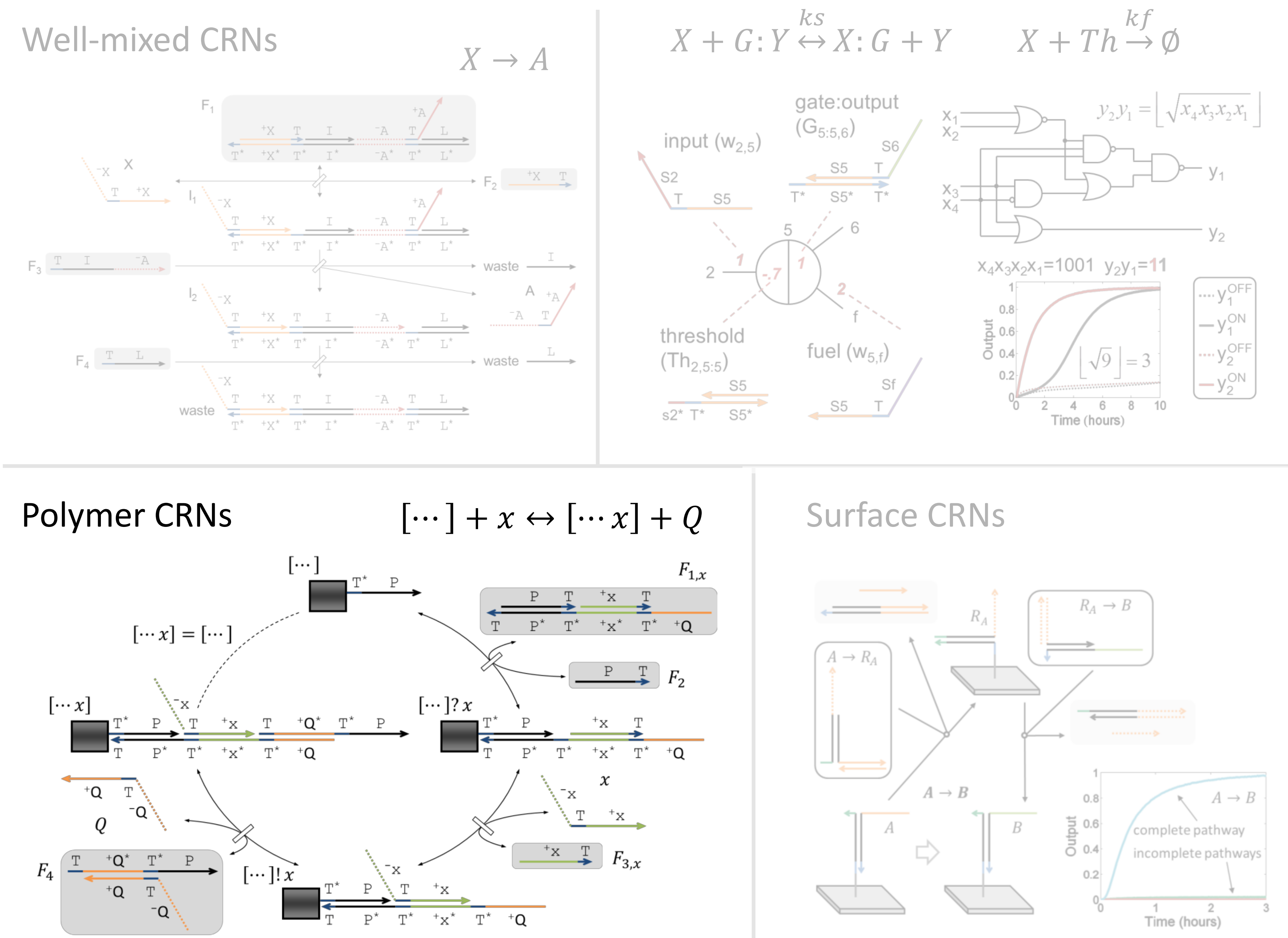


0=0.1x 1=0.9x





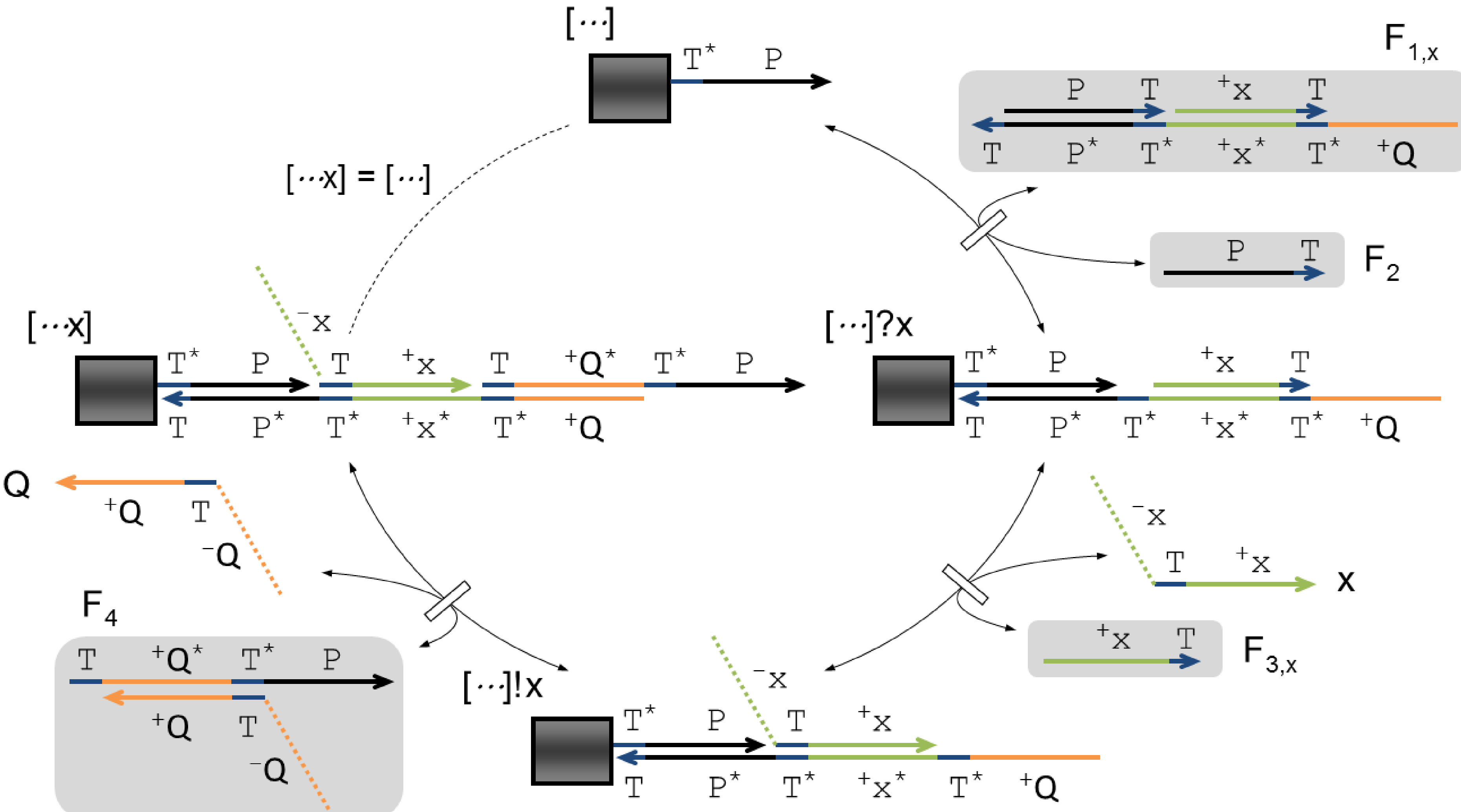

Well-mixed CRNs
X → A
F1
waste
gate:output (G5:5,6)
input (w2,5)
threshold (Th2,5:5)
fuel (w5,f)
X + G:Y ↔ X:G + Y
X + Th → Ø
ks
kf
x4x3x2x1=1001 y2y1=11
Output
Time (hours)
Polymer CRNs
[⋯] + x ↔ [⋯x] + Q
Surface CRNs
A → RA
RA → B
A → B
complete pathway
incomplete pathways



Well-mixed CRNs
X → A
F1
F2
F3
F4
I1
I2
waste
X + G:Y ↔ X:G + Y (ks)
X + Th → ∅ (kf)
input (w2,5)
gate:output (G5:5,6)
threshold (Th2,5:5)
fuel (w5,f)
x4x3x2x1=1001 y2y1=11
Output
Time (hours)
Polymer CRNs
[⋯] + x ↔ [⋯x] + Q
[⋯]
[⋯x] = [⋯]
[⋯x]
[⋯]?x
[⋯]!x
F1,x
F2
F3,x
F4
Q
x
Surface CRNs
A → RA
RA → B
A → B
complete pathway
incomplete pathways
Output
Time (hours)

A DNA polymer reaction $[\cdots] + x \leftrightarrow [\cdots x] + Q$

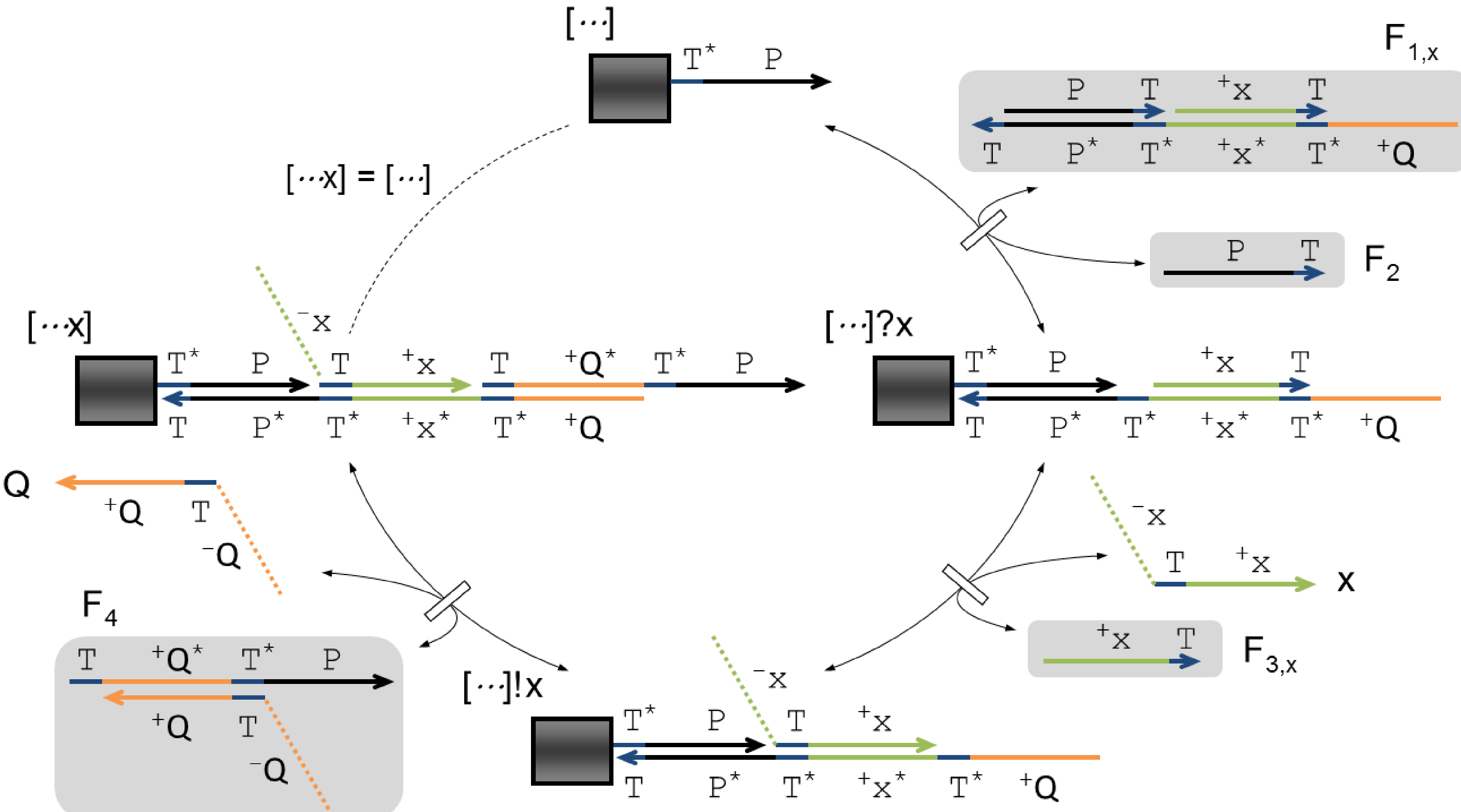
push: $[\cdots] + x \rightarrow [\cdots x] + Q$ $\qquad$ Q is a confirmation signal



A DNA polymer reaction $[\cdots] + x \leftrightarrow [\cdots x] + Q$

pop: $[\cdots x] + Q \to [\cdots] + x$

Q is a query signal



A DNA stack machine implementation

transition rules:

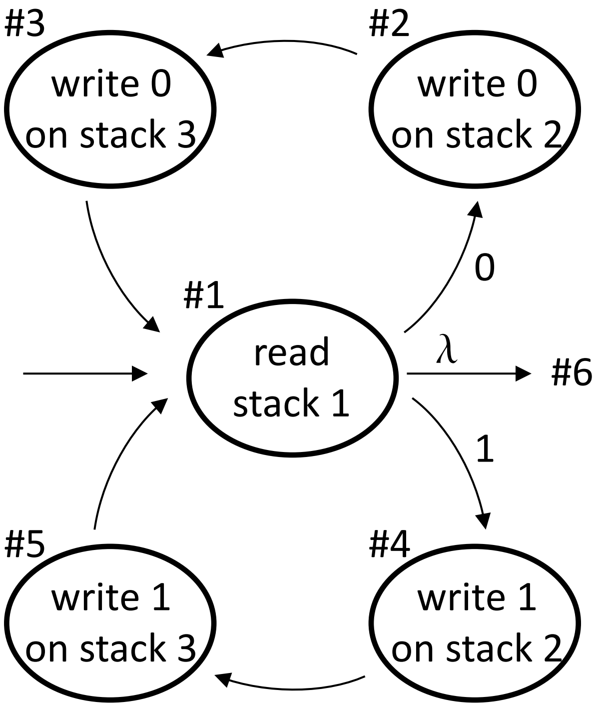
current state, pop symbol, stack number --> next state, push symbol, stack number

$$Q \rightleftharpoons Q_i$$

$$[\cdots]_i + x_i \rightleftharpoons [\cdots x]_i + Q_i$$

$$
\begin{array}{llcl}
1. & \alpha\ x\ i \longrightarrow \beta\ y\ j & \Rightarrow & S_\alpha + x_i \rightarrow S_\beta + y_j \\
2. & \alpha\ x\ i \longrightarrow \beta & \Rightarrow & S_\alpha + x_i \rightarrow S_\beta + Q \\
3. & \alpha \quad\ \longrightarrow \beta\ y\ j & \Rightarrow & S_\alpha + Q \rightarrow S_\beta + y_j \\
4. & \alpha\ \lambda\ i \longrightarrow \beta\ \lambda\ i & \Rightarrow & S_\alpha + \perp_i \rightarrow S_\beta + \perp_i
\end{array}
$$

A DNA stack machine implementation



$\#1\ 0\ 1 \longrightarrow \#2$

$\#1\ 1\ 1 \longrightarrow \#4$

$\#1\ \lambda\ 1 \longrightarrow \#6\ \lambda\ 1$

$\#2 \longrightarrow \#3\ 0\ 2$

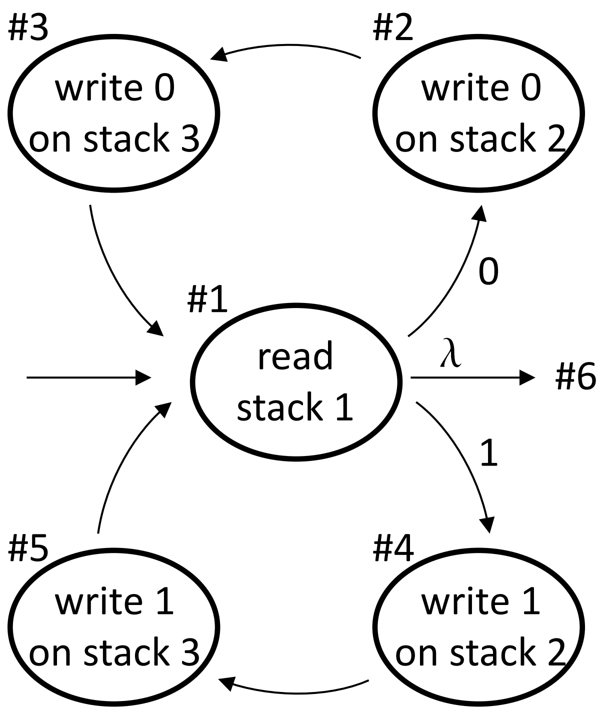
$\#3 \longrightarrow \#1\ 0\ 3$

$\#4 \longrightarrow \#5\ 1\ 2$

$\#5 \longrightarrow \#1\ 1\ 3$

1. (#1, 00111, λ, λ)
2. (#4, 0011, λ, λ)
3. (#5, 0011, 1, λ)
4. (#1, 0011, 1, 1)
5. (#4, 001, 1, 1)
6. (#5, 001, 11, 1)
7. (#1, 001, 11, 11)
8. (#4, 00, 11, 11)
9. (#5, 00, 111, 11)
10. (#1, 00, 111, 111)
11. (#2, 0, 111, 111)
12. (#3, 0, 1110, 111)
13. (#1, 0, 1110, 1110)
14. (#2, λ, 1110, 1110)
15. (#3, λ, 11100, 1110)
16. (#1, λ, 11100, 11100)
17. (#6, λ, 11100,11100)

A DNA stack machine implementation



$\#1\ 0\ 1 \longrightarrow \#2$

$\#1\ 1\ 1 \longrightarrow \#4$

$\#1\ \lambda\ 1 \longrightarrow \#6\ \lambda\ 1$

$\#2 \longrightarrow \#3\ 0\ 2$

$\#3 \longrightarrow \#1\ 0\ 3$

$\#4 \longrightarrow \#5\ 1\ 2$

$\#5 \longrightarrow \#1\ 1\ 3$

$S_{\#1} + 0_1 \longrightarrow S_{\#2} + Q$

$S_{\#1} + 1_1 \longrightarrow S_{\#4} + Q$

$S_{\#1} + \perp_1 \longrightarrow S_{\#6} + \perp_1$

$S_{\#2} + Q \longrightarrow S_{\#3} + 0_2$

$S_{\#3} + Q \longrightarrow S_{\#1} + 0_3$

$S_{\#4} + Q \longrightarrow S_{\#5} + 1_2$

$S_{\#5} + Q \longrightarrow S_{\#1} + 1_3$

$Q_1 \longleftrightarrow Q$

$Q_2 \longleftrightarrow Q$

$Q_3 \longleftrightarrow Q$

$[\ldots]_1 + 0_1 \longleftrightarrow [\ldots 0]_1 + Q_1$

$[\ldots]_1 + 1_1 \longleftrightarrow [\ldots 1]_1 + Q_1$

$[\ldots]_2 + 0_2 \longleftrightarrow [\ldots 0]_2 + Q_2$

$[\ldots]_2 + 1_2 \longleftrightarrow [\ldots 1]_2 + Q_2$

$[\ldots]_3 + 0_3 \longleftrightarrow [\ldots 0]_3 + Q_3$

$[\ldots]_3 + 1_3 \longleftrightarrow [\ldots 1]_3 + Q_3$

A DNA stack machine implementation

$S_{\#1} + 0_1 \longrightarrow S_{\#2} + Q$

$S_{\#1} + 1_1 \longrightarrow S_{\#4} + Q$

$S_{\#1} + \bot_1 \longrightarrow S_{\#6} + \bot_1$

$S_{\#2} + Q \longrightarrow S_{\#3} + 0_2$

$S_{\#3} + Q \longrightarrow S_{\#1} + 0_3$

$S_{\#4} + Q \longrightarrow S_{\#5} + 1_2$

$S_{\#5} + Q \longrightarrow S_{\#1} + 1_3$

$Q_1 \longleftrightarrow Q$

$Q_2 \longleftrightarrow Q$

$Q_3 \longleftrightarrow Q$

$[\ldots]_1 + 0_1 \longleftrightarrow [\ldots 0]_1 + Q_1$

$[\ldots]_1 + 1_1 \longleftrightarrow [\ldots 1]_1 + Q_1$

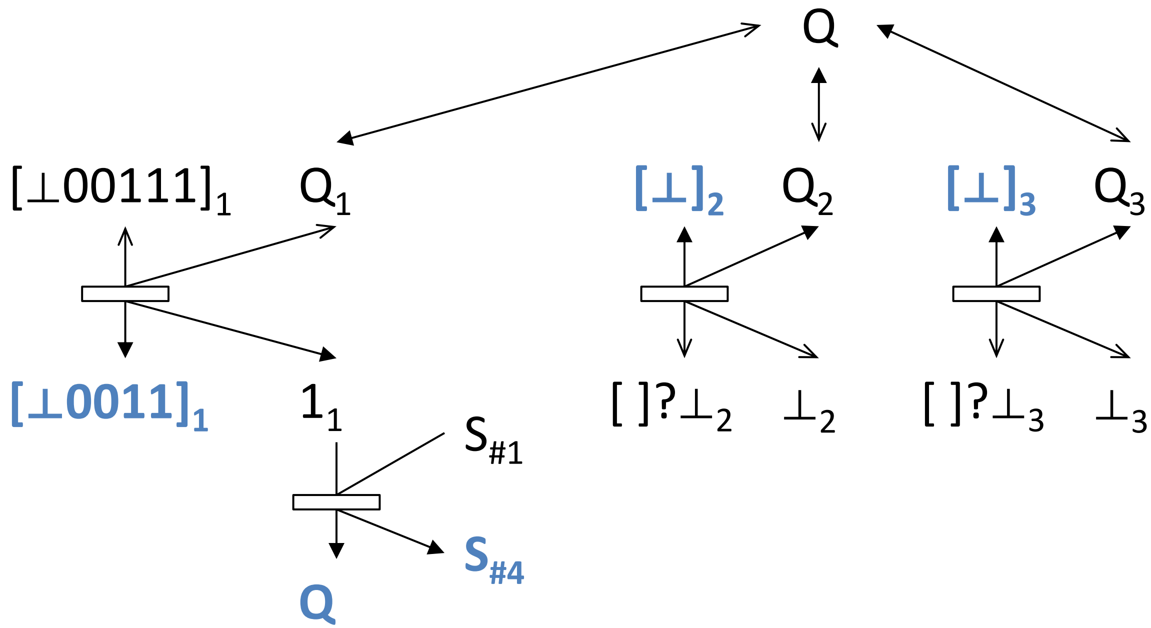
$[\ldots]_2 + 0_2 \longleftrightarrow [\ldots 0]_2 + Q_2$

$[\ldots]_2 + 1_2 \longleftrightarrow [\ldots 1]_2 + Q_2$

$[\ldots]_3 + 0_3 \longleftrightarrow [\ldots 0]_3 + Q_3$

$[\ldots]_3 + 1_3 \longleftrightarrow [\ldots 1]_3 + Q_3$

1. $S_{\#1}$, Q, $[\bot 00111]_1$, $[\bot]_2$, $[\bot]_3$



2. $S_{\#4}$, Q, $[\bot 0011]_1$, $[\bot]_2$, $[\bot]_3$

A DNA stack machine implementation

$S_{\#1} + 0_1 \longrightarrow S_{\#2} + Q$

$S_{\#1} + 1_1 \longrightarrow S_{\#4} + Q$

$S_{\#1} + \perp_1 \longrightarrow S_{\#6} + \perp_1$

$S_{\#2} + Q \longrightarrow S_{\#3} + 0_2$

$S_{\#3} + Q \longrightarrow S_{\#1} + 0_3$

$S_{\#4} + Q \longrightarrow S_{\#5} + 1_2$

$S_{\#5} + Q \longrightarrow S_{\#1} + 1_3$

$Q_1 \longleftrightarrow Q$

$Q_2 \longleftrightarrow Q$

$Q_3 \longleftrightarrow Q$

$[\ldots]_1 + 0_1 \longleftrightarrow [\ldots 0]_1 + Q_1$

$[\ldots]_1 + 1_1 \longleftrightarrow [\ldots 1]_1 + Q_1$

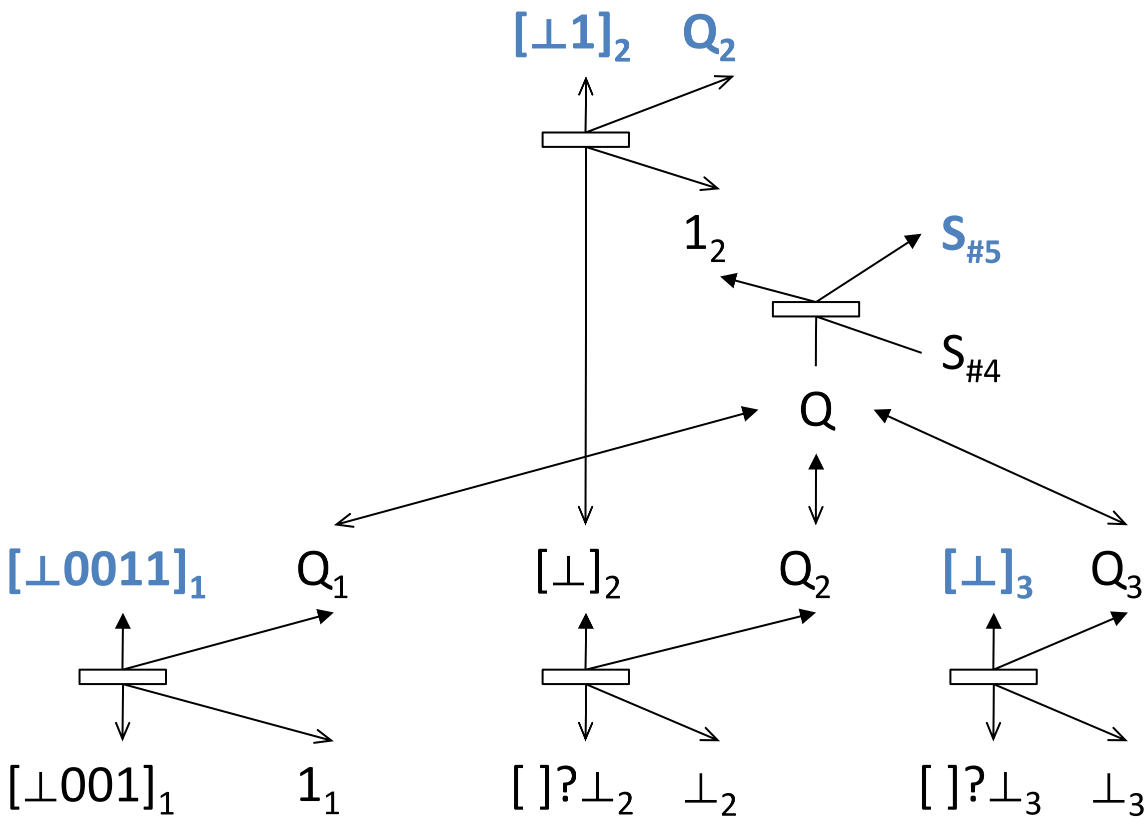
$[\ldots]_2 + 0_2 \longleftrightarrow [\ldots 0]_2 + Q_2$

$[\ldots]_2 + 1_2 \longleftrightarrow [\ldots 1]_2 + Q_2$

$[\ldots]_3 + 0_3 \longleftrightarrow [\ldots 0]_3 + Q_3$

$[\ldots]_3 + 1_3 \longleftrightarrow [\ldots 1]_3 + Q_3$

1. $S_{\#1}$, Q, $[\perp 00111]_1$, $[\perp]_2$, $[\perp]_3$
2. $S_{\#4}$, Q, $[\perp 0011]_1$, $[\perp]_2$, $[\perp]_3$

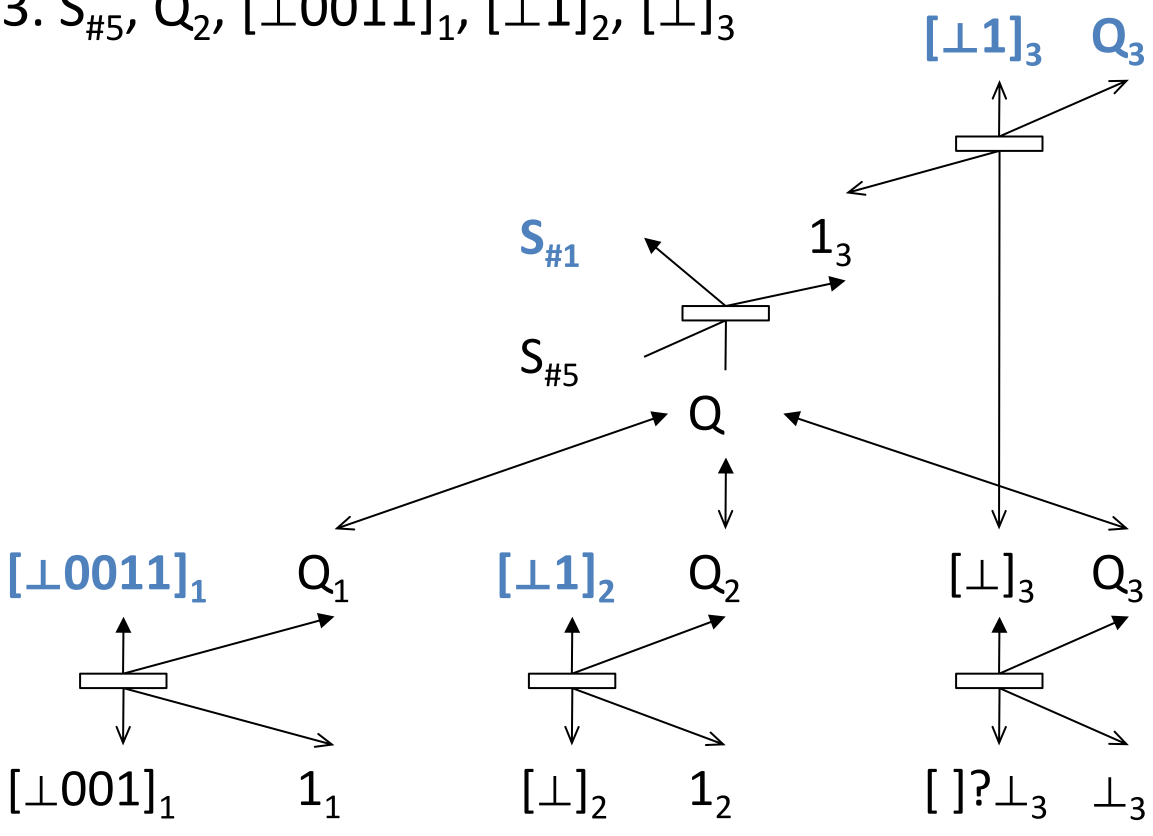


3. $S_{\#5}$, Q_2, $[\perp 0011]_1$, $[\perp 1]_2$, $[\perp]_3$

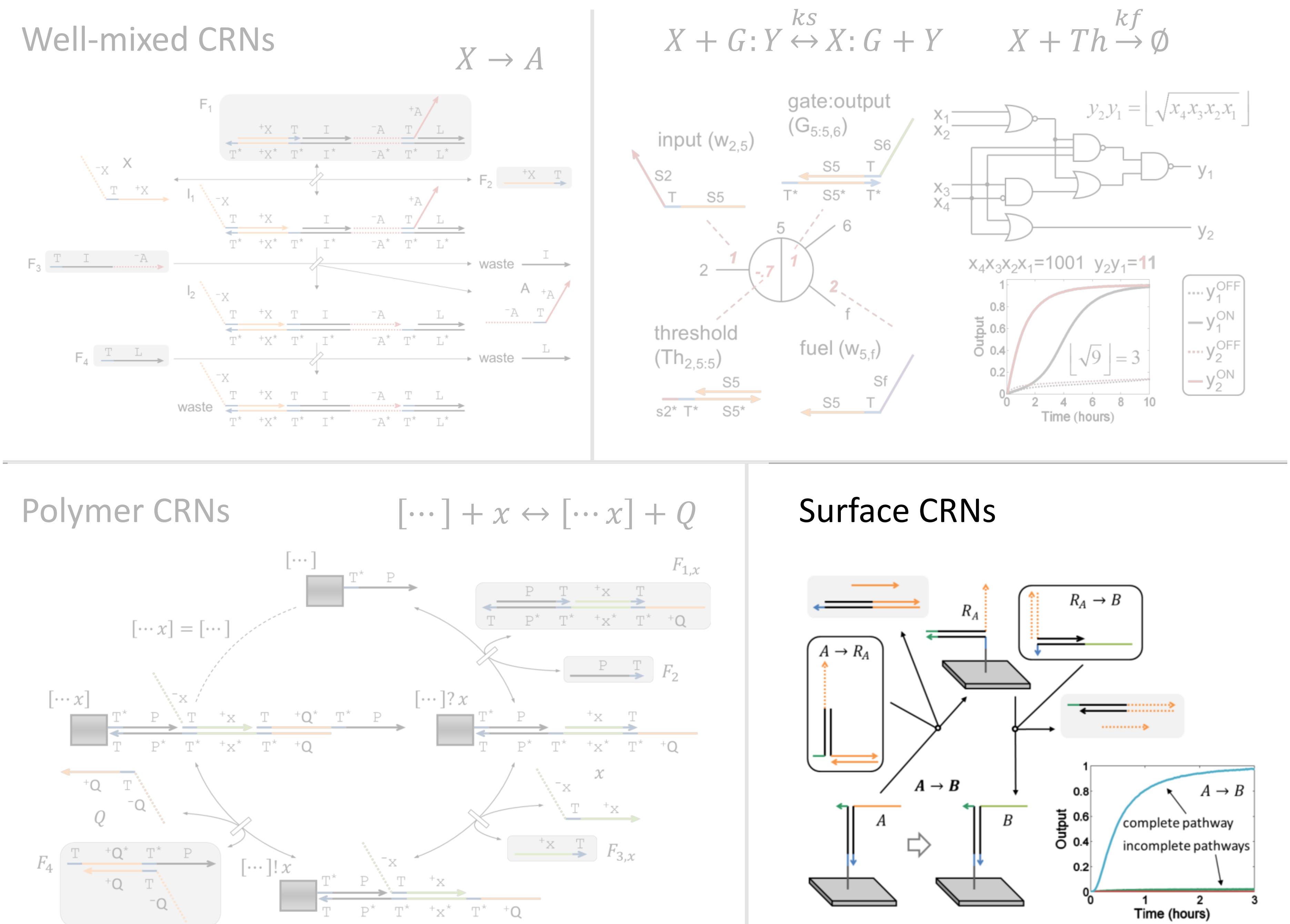
A DNA stack machine implementation

$S_{\#1} + 0_1 \longrightarrow S_{\#2} + Q$
$S_{\#1} + 1_1 \longrightarrow S_{\#4} + Q$
$S_{\#1} + \perp_1 \longrightarrow S_{\#6} + \perp_1$
$S_{\#2} + Q \longrightarrow S_{\#3} + 0_2$
$S_{\#3} + Q \longrightarrow S_{\#1} + 0_3$
$S_{\#4} + Q \longrightarrow S_{\#5} + 1_2$
$S_{\#5} + Q \longrightarrow S_{\#1} + 1_3$
$Q_1 \longleftrightarrow Q$
$Q_2 \longleftrightarrow Q$
$Q_3 \longleftrightarrow Q$
$[\ldots]_1 + 0_1 \longleftrightarrow [\ldots 0]_1 + Q_1$
$[\ldots]_1 + 1_1 \longleftrightarrow [\ldots 1]_1 + Q_1$
$[\ldots]_2 + 0_2 \longleftrightarrow [\ldots 0]_2 + Q_2$
$[\ldots]_2 + 1_2 \longleftrightarrow [\ldots 1]_2 + Q_2$
$[\ldots]_3 + 0_3 \longleftrightarrow [\ldots 0]_3 + Q_3$
$[\ldots]_3 + 1_3 \longleftrightarrow [\ldots 1]_3 + Q_3$

1. $S_{\#1}$, Q, $[\perp 00111]_1$, $[\perp]_2$, $[\perp]_3$
2. $S_{\#4}$, Q, $[\perp 0011]_1$, $[\perp]_2$, $[\perp]_3$
3. $S_{\#5}$, Q_2, $[\perp 0011]_1$, $[\perp 1]_2$, $[\perp]_3$

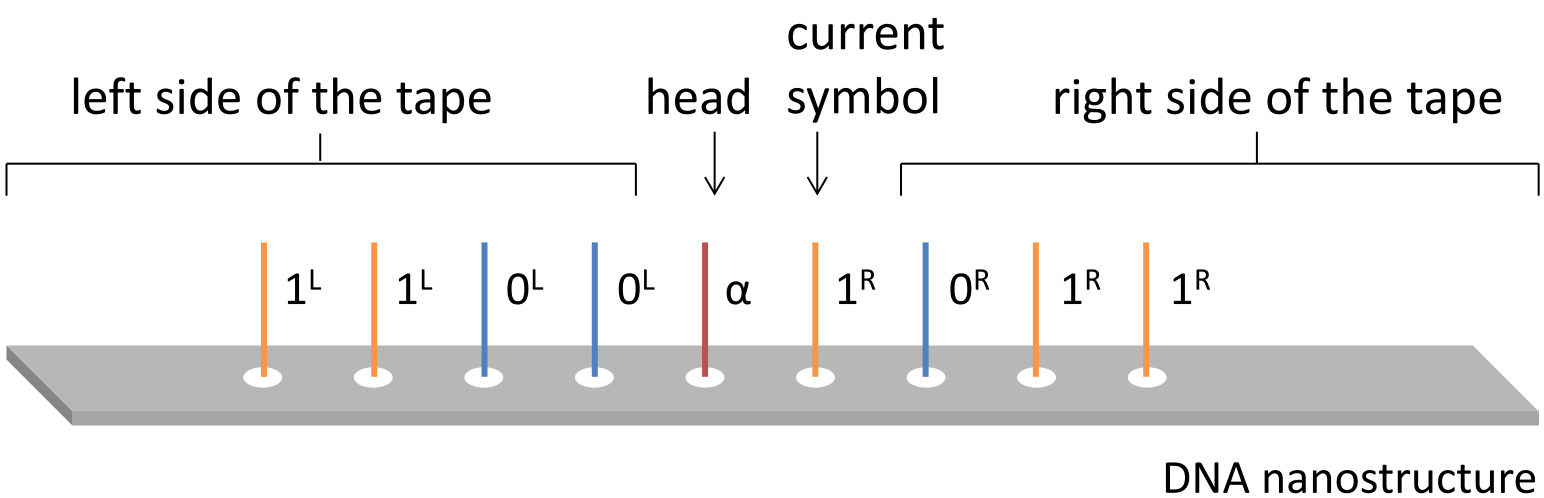


4. $S_{\#1}$, Q_3, $[\perp 0011]_1$, $[\perp 1]_2$, $[\perp 1]_3$



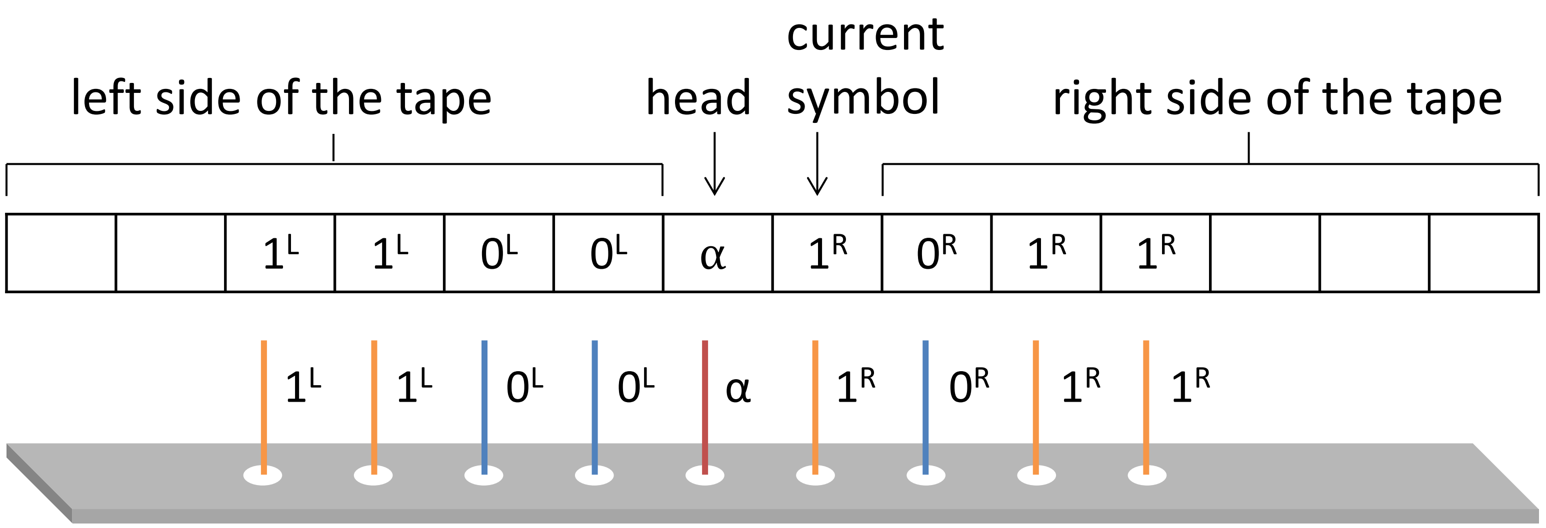
Well-mixed CRNs
X → A
F1
waste
A
F2
F3
F4
I1
I2
X + G:Y ↔ X:G + Y
ks
X + Th → Ø
kf
gate:output
(G5:5,6)
input (w2,5)
threshold
(Th2,5:5)
fuel (w5,f)
S2
S5
S6
T
Sf
x4x3x2x1=1001 y2y1=11
Output
Time (hours)
Polymer CRNs
[⋯] + x ↔ [⋯x] + Q
[⋯x] = [⋯]
[⋯]?x
[⋯]!x
F1,x
F2
F3,x
F4
Q
x
Surface CRNs
A → RA
RA → B
RA
A → B
A
B
complete pathway
incomplete pathways
Output
Time (hours)

Can we use a DNA nanostructure to organize symbols and states represented as single-stranded DNA signals on a surface and serve as a tape for a molecular Turing machine?

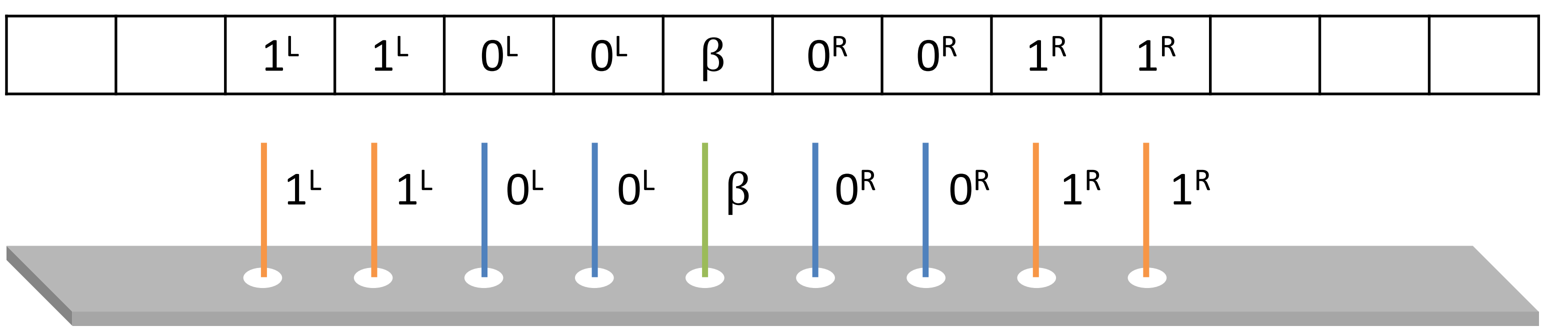


Qian et al, *LNCS* 2014

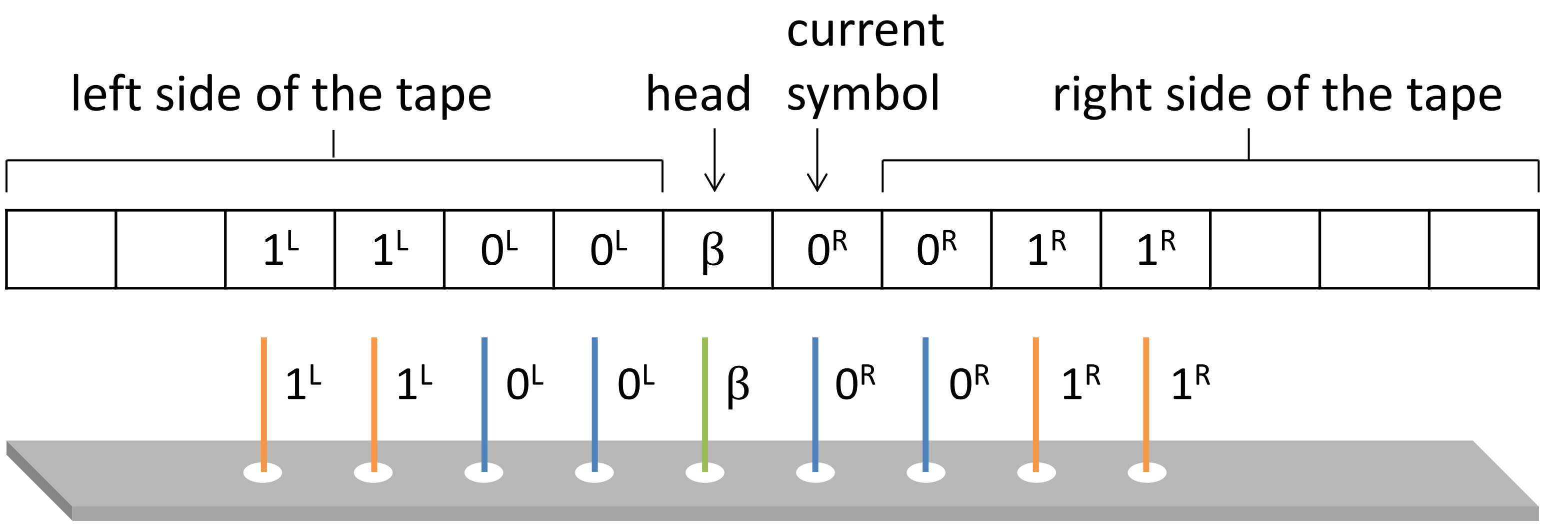
Efficient molecular Turing machine



transition rule: $\{\alpha, 1\} \rightarrow \{\beta, 0\}$ $\qquad \alpha + 1^R \rightarrow \beta + 0^R$

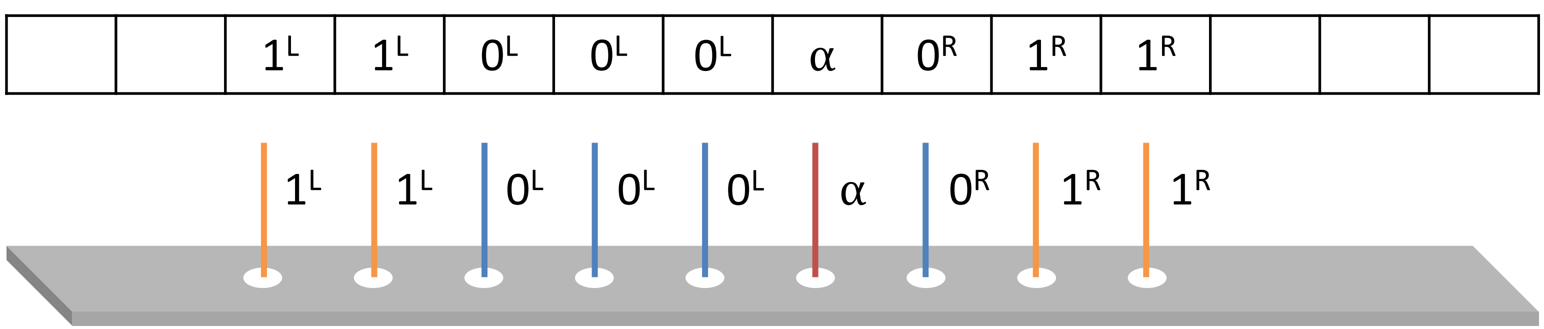


Efficient molecular Turing machine



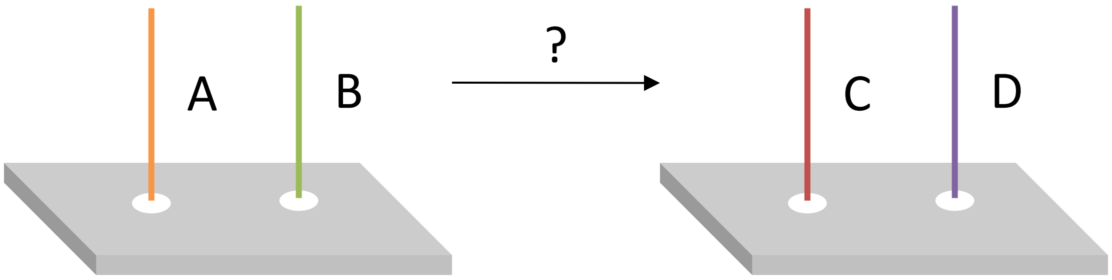
transition rule: $\{\beta\} \rightarrow \{\alpha, +\}$

$$\begin{cases} \beta + 0^R \rightarrow 0^L + \alpha \\ \beta + 1^R \rightarrow 1^L + \alpha \end{cases}$$



How do we cooperatively change two neighboring signals on a surface?

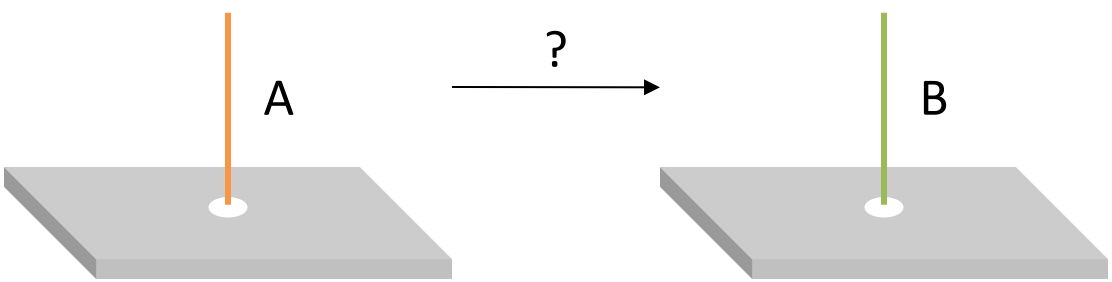
$$A + B \rightarrow C + D$$



Surface-based formal bimolecular reaction

How do we change a signal on a surface from an original state to a new state?

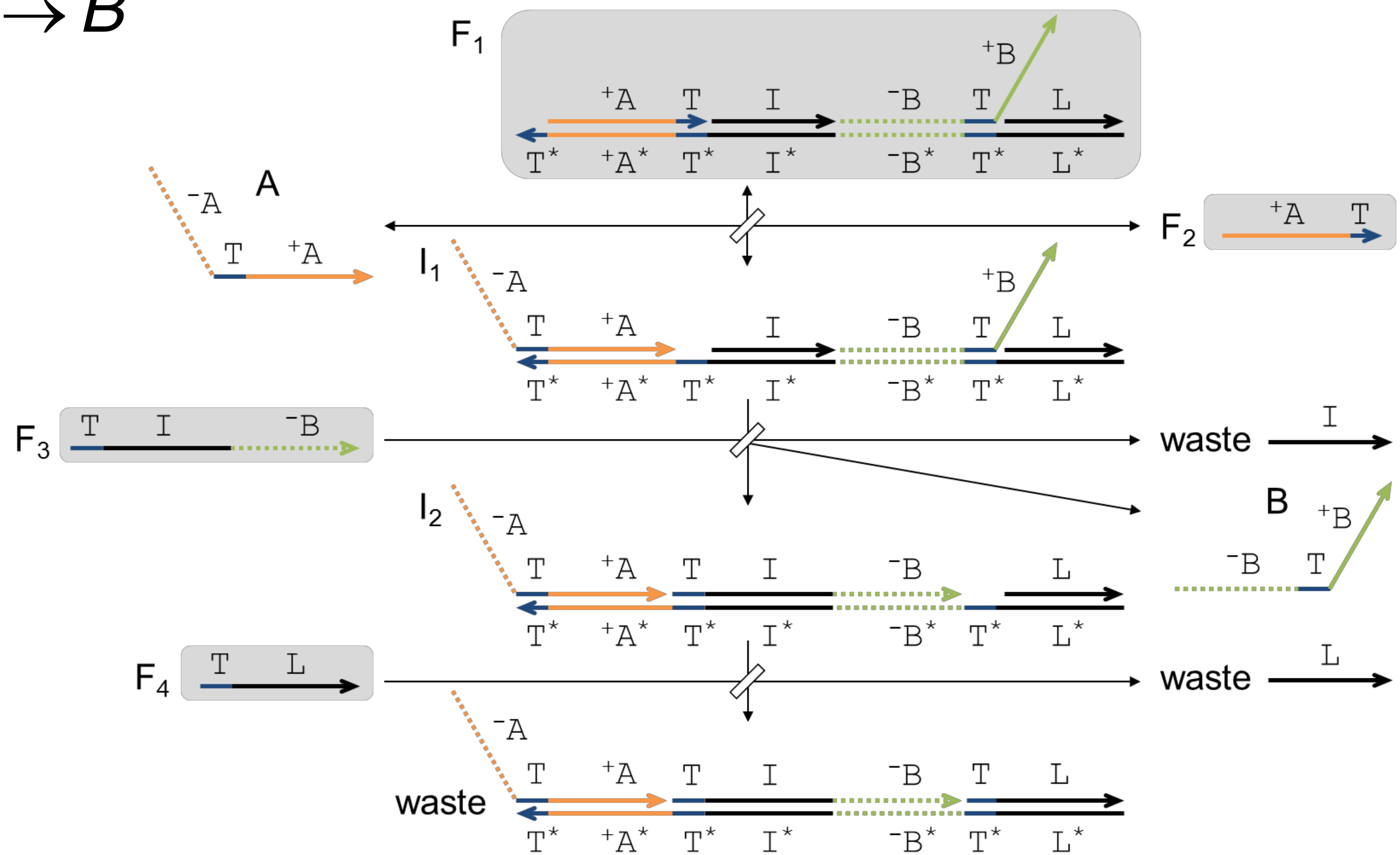
$$A \rightarrow B$$



Surface-based formal unimolecular reaction

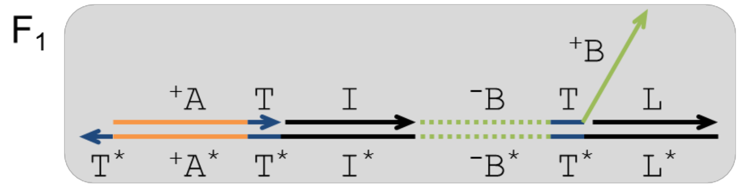
Implementation of well-mixed formal unimolecular reaction

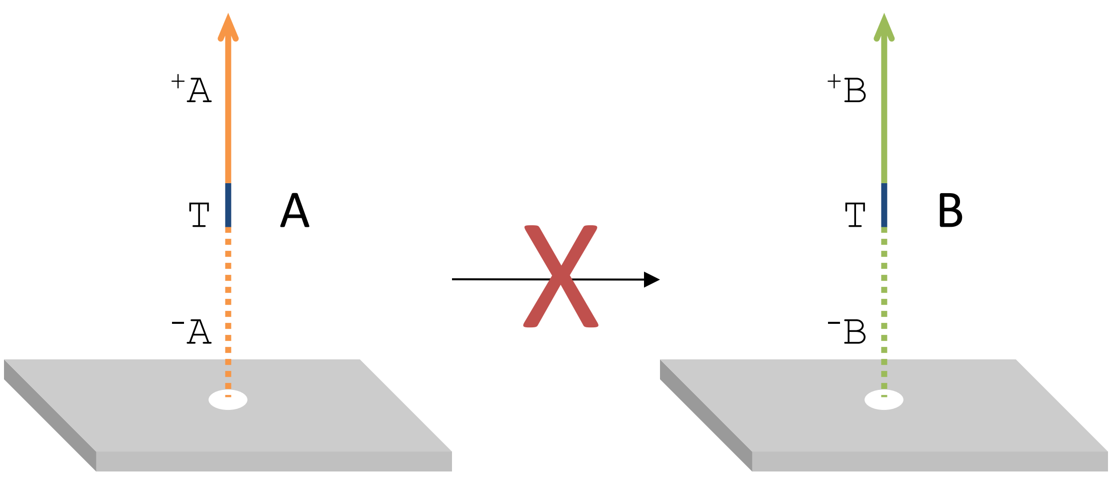
$$A \rightarrow B$$



Implementation of well-mixed formal unimolecular reaction

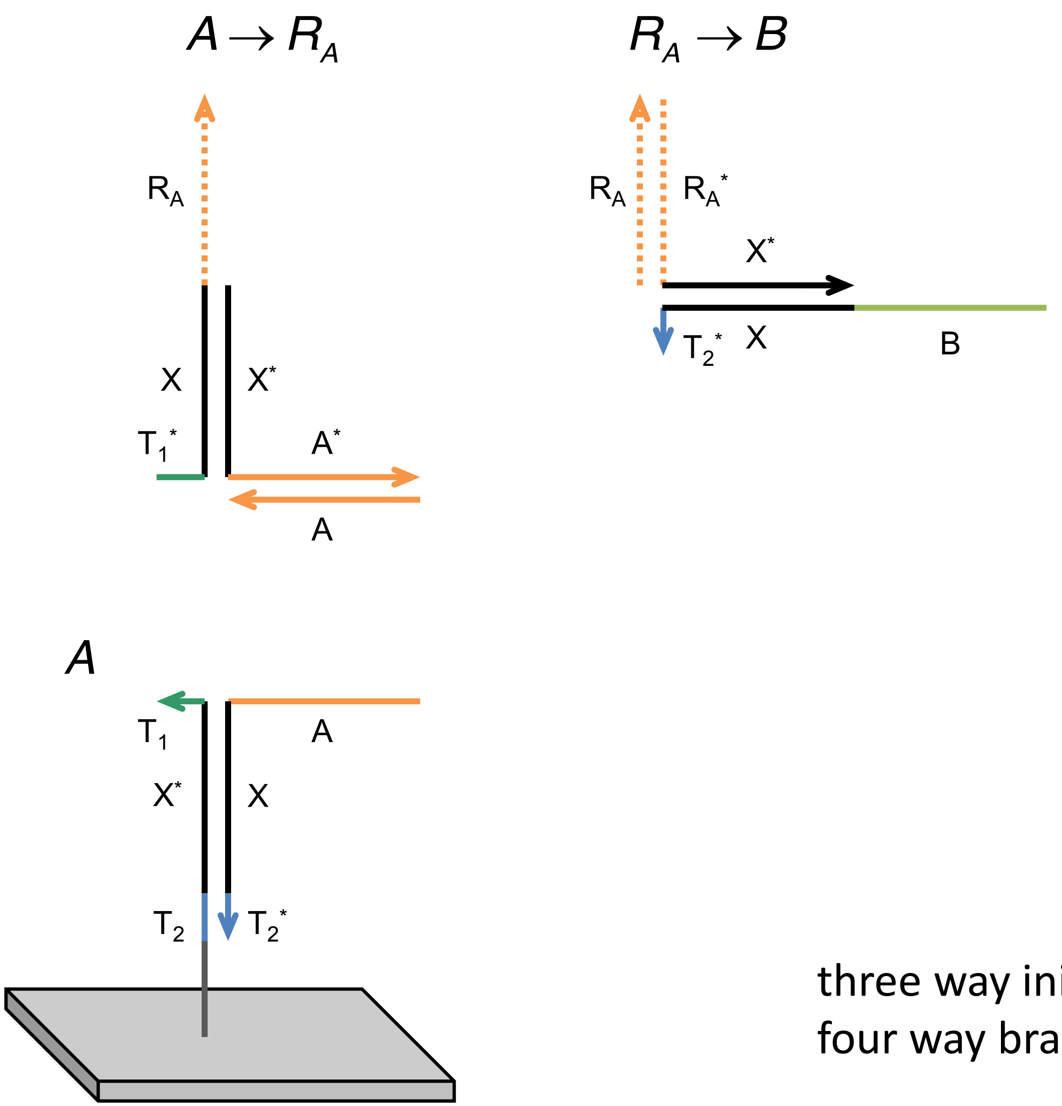
$A \rightarrow B$





Implementation of surface-based formal unimolecular reaction

$A \to B$

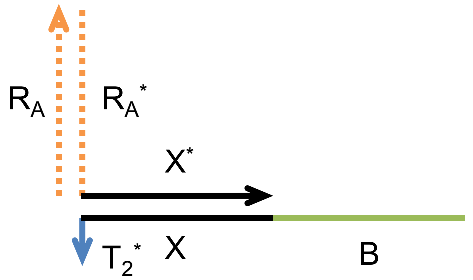


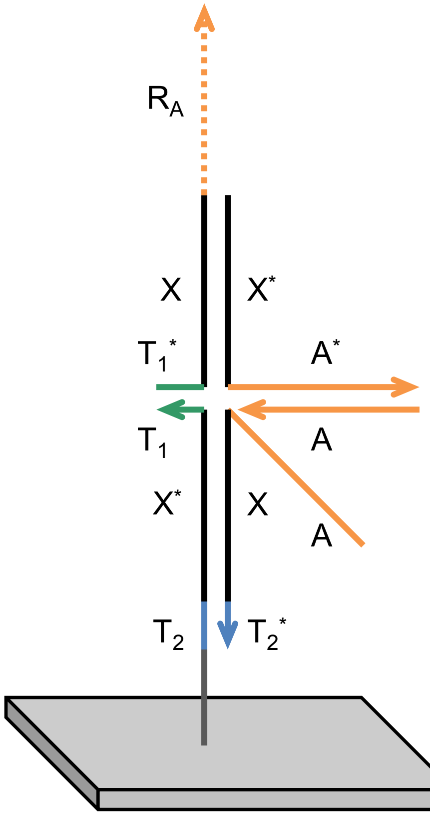
three way initiated
four way branch migration

Implementation of surface-based formal unimolecular reaction

$$A \rightarrow B$$

$$R_A \rightarrow B$$



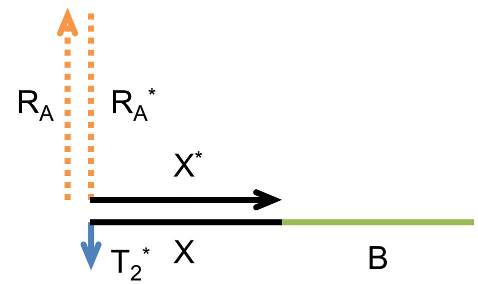


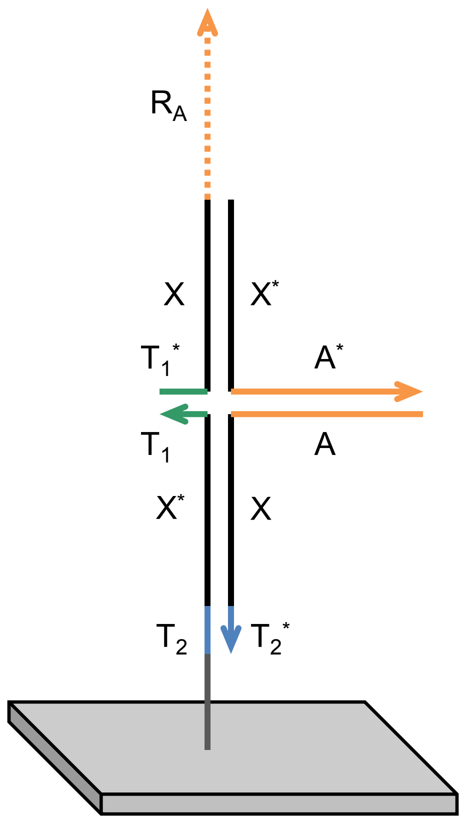
three way initiated
four way branch migration

Implementation of surface-based formal unimolecular reaction

$$A \rightarrow B$$

$$R_A \rightarrow B$$



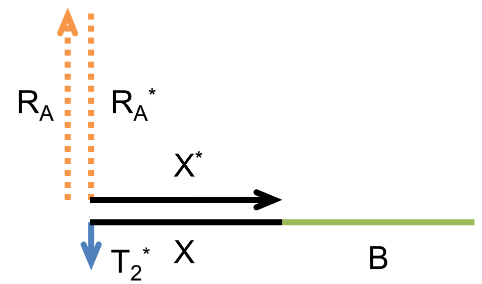


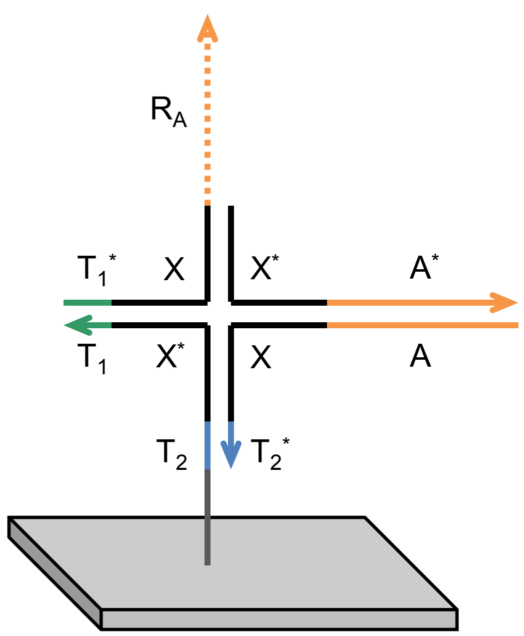
three way initiated
four way branch migration

Implementation of surface-based formal unimolecular reaction

$$A \rightarrow B$$

$$R_A \rightarrow B$$



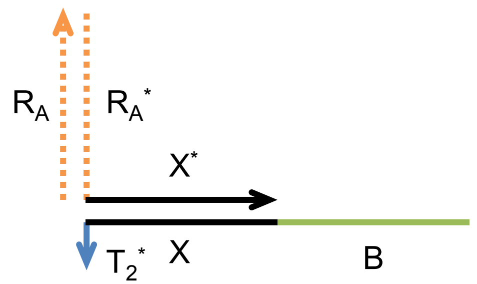


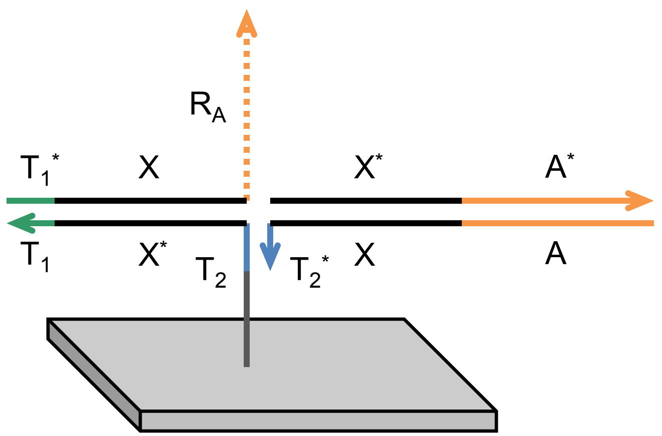
three way initiated
four way branch migration

Implementation of surface-based formal unimolecular reaction

$$A \to B$$

$$R_A \to B$$



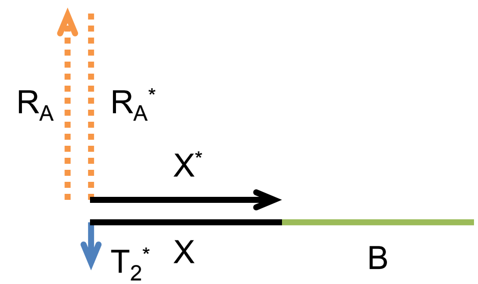


three way initiated
four way branch migration

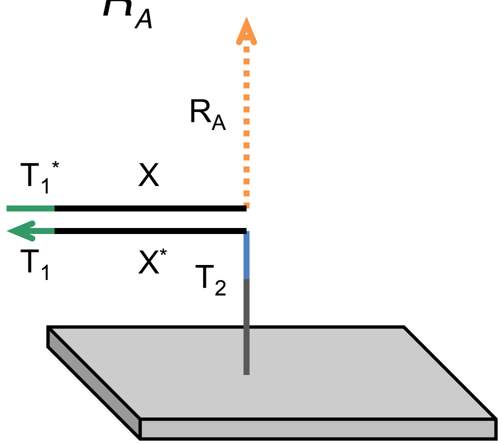
Implementation of surface-based formal unimolecular reaction

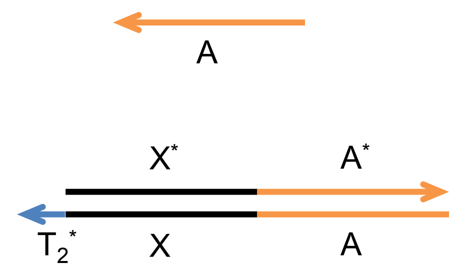
$$A \rightarrow B$$

$$R_A \rightarrow B$$



$$R_A$$

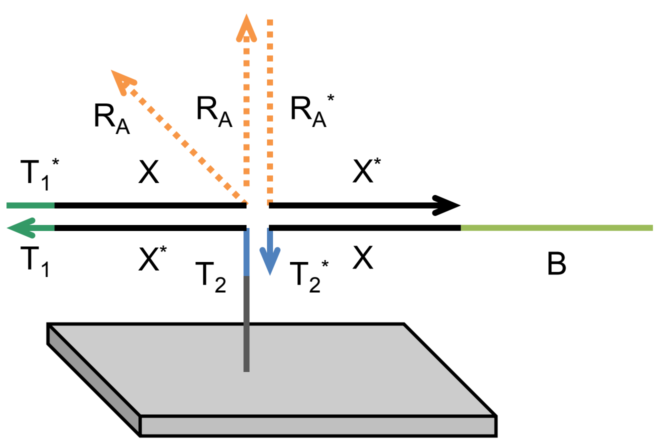


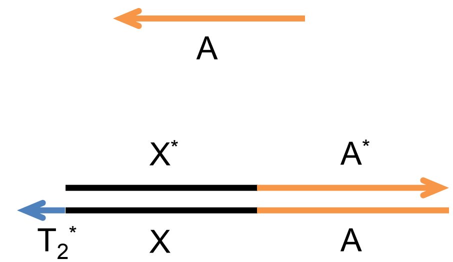


three way initiated
four way branch migration

Implementation of surface-based formal unimolecular reaction

$$A \to B$$

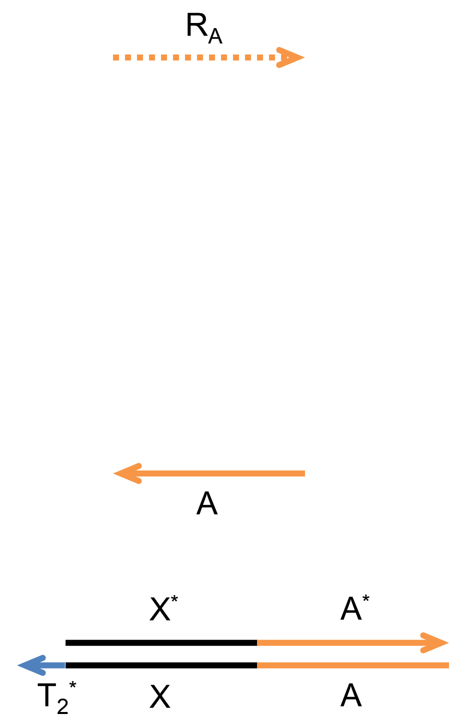


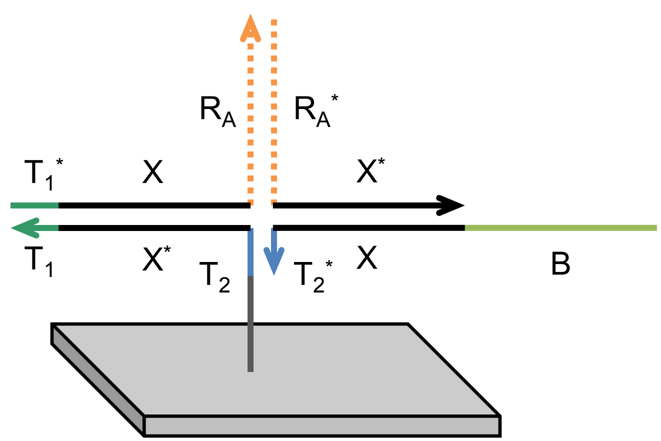


three way initiated
four way branch migration

Implementation of surface-based formal unimolecular reaction

$$A \rightarrow B$$

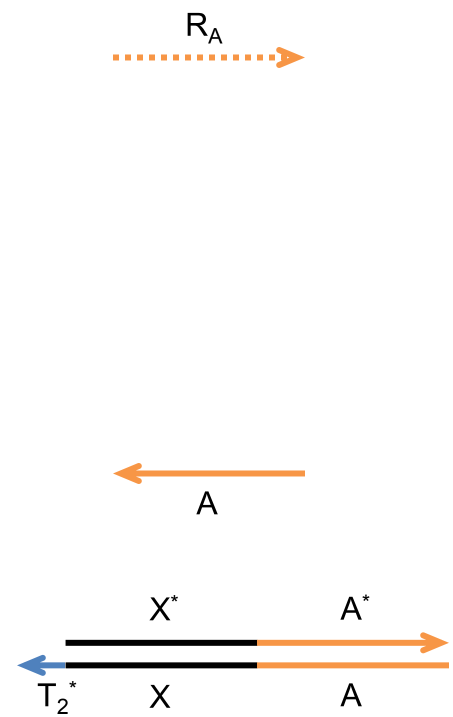


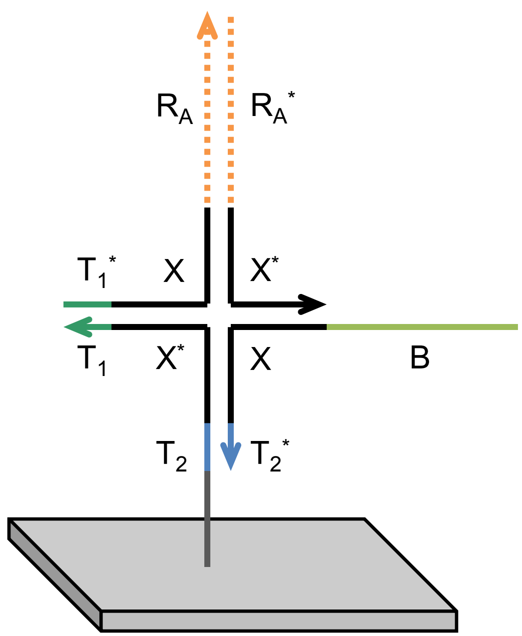


three way initiated
four way branch migration

Implementation of surface-based formal unimolecular reaction

$$A \rightarrow B$$

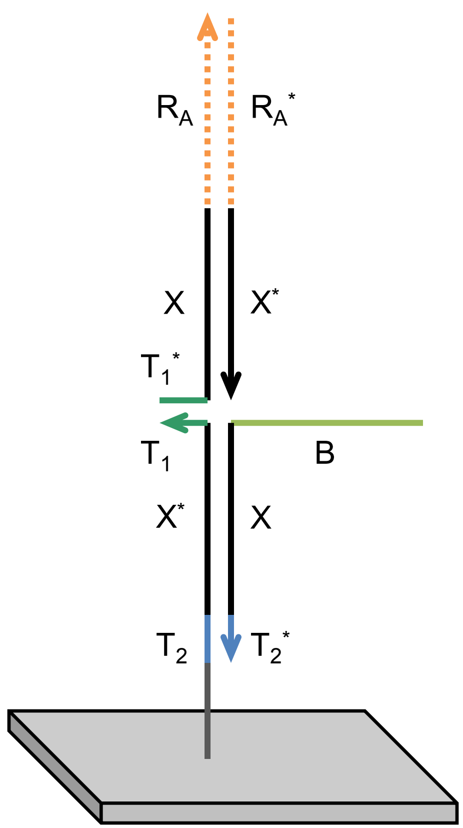


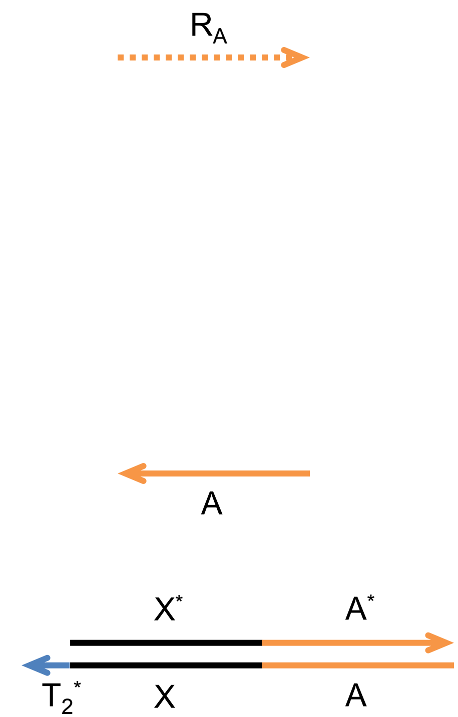


three way initiated
four way branch migration

Implementation of surface-based formal unimolecular reaction

$$A \rightarrow B$$

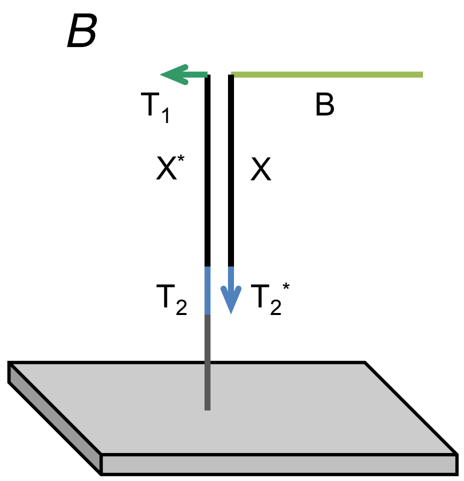


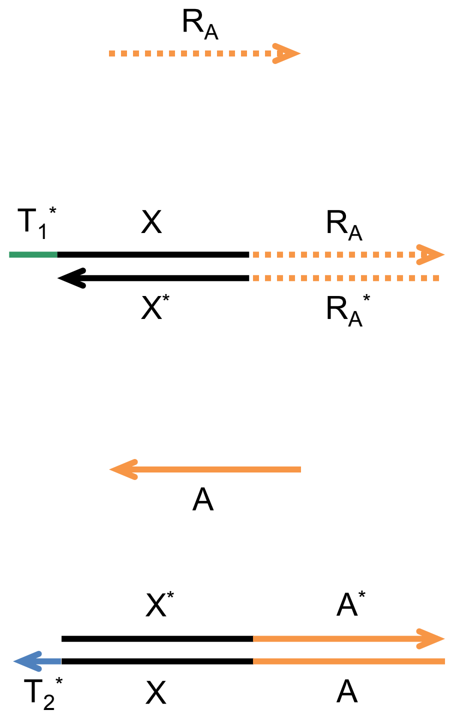


three way initiated
four way branch migration

Implementation of surface-based formal unimolecular reaction

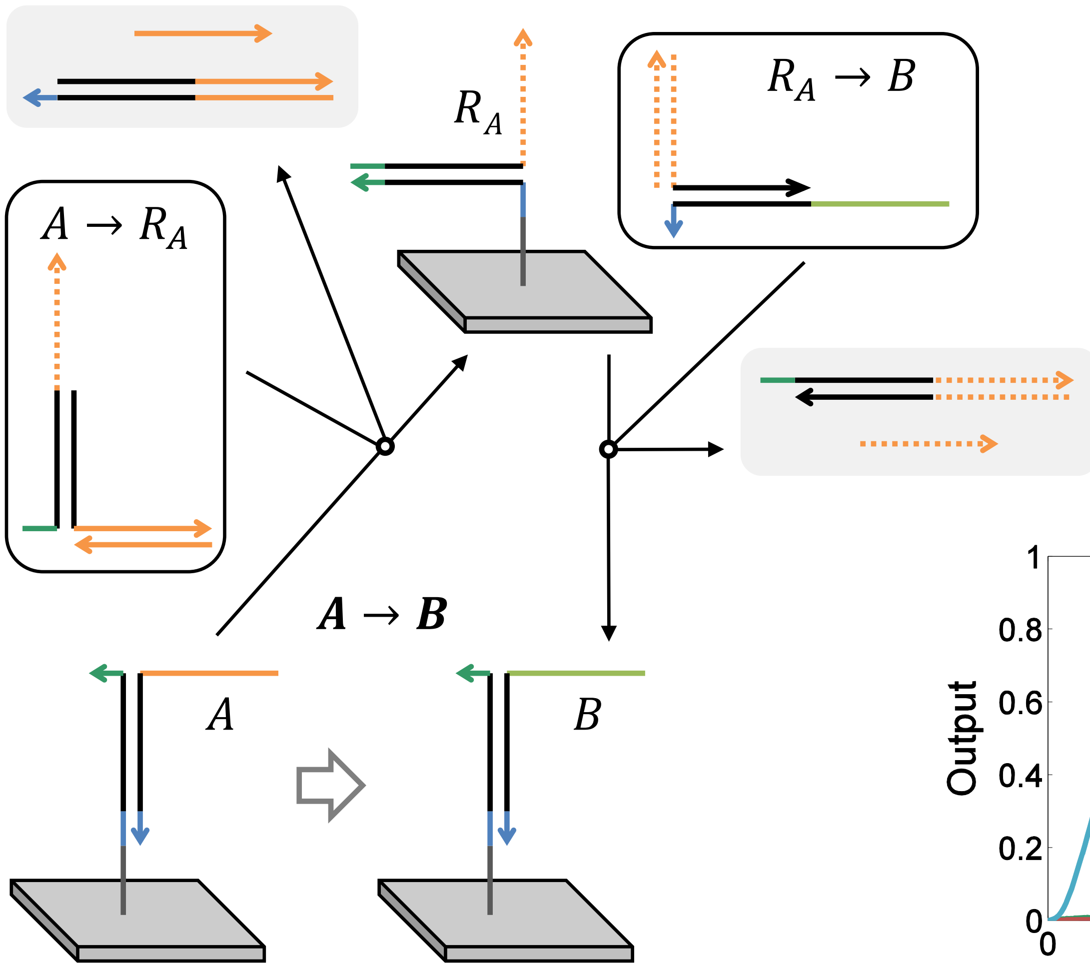
$$A \rightarrow B$$



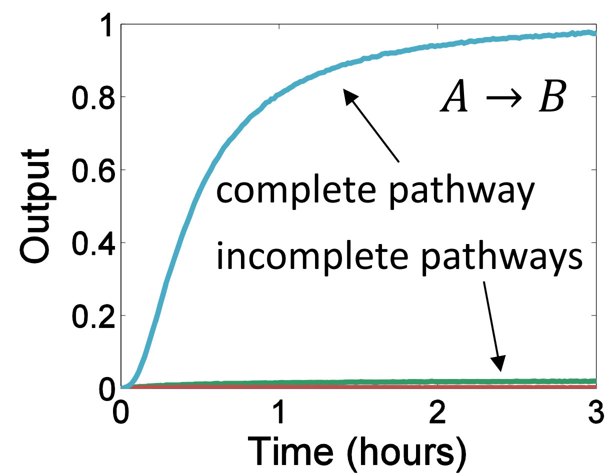


three way initiated
four way branch migration

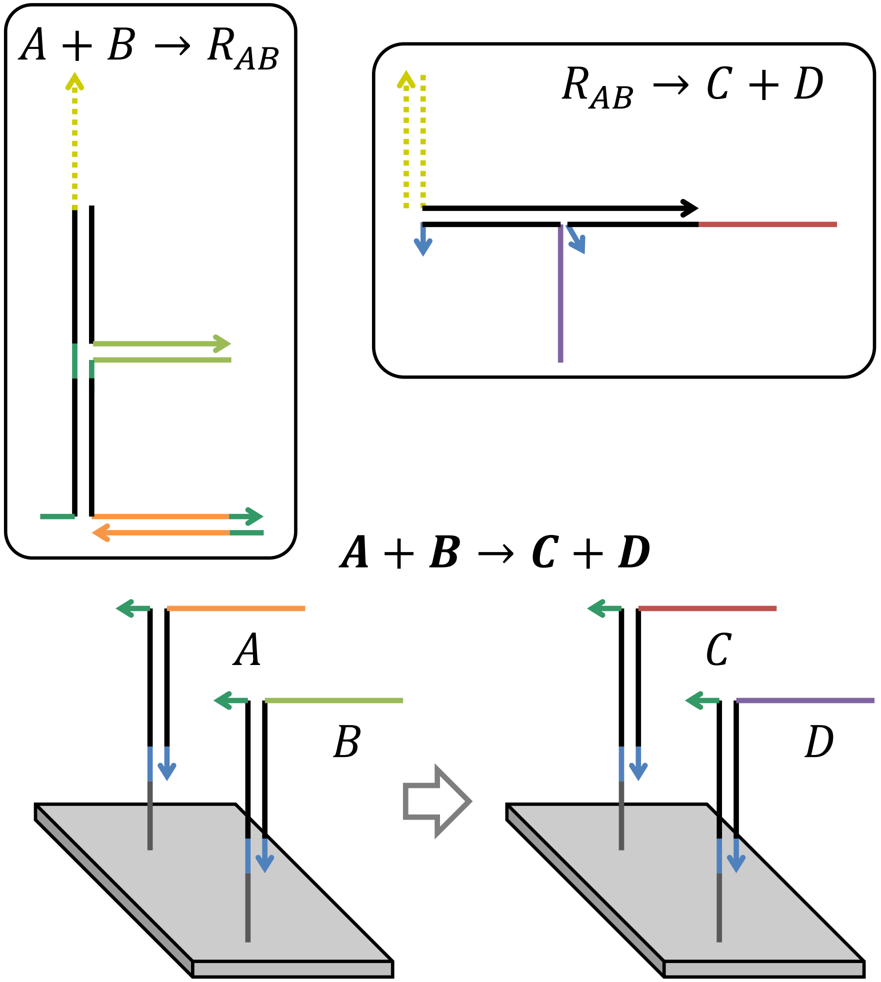
Implementation of surface-based formal unimolecular reaction



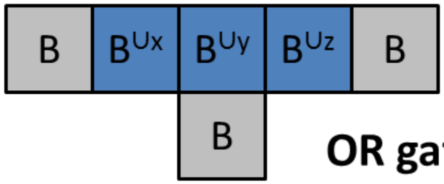
three way initiated
four way branch migration



Implementation of surface-based formal bimolecular reaction

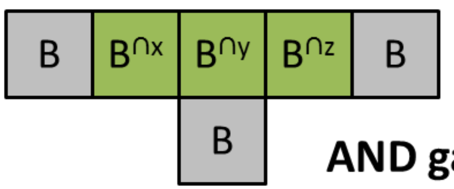


Continuously active logic circuits with surface CRNs



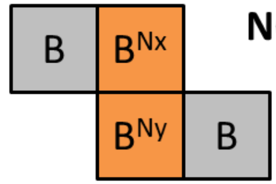
OR gate

$$\begin{cases} 0^{\cup x} + 0^{\cup y} \rightarrow B^{\cup x} + 0^{\cup k} \\ 0^{\cup x} + 1^{\cup y} \rightarrow B^{\cup x} + 1^{\cup k} \\ 1^{\cup x} + 0^{\cup y} \rightarrow B^{\cup x} + 1^{\cup k} \\ 1^{\cup x} + 1^{\cup y} \rightarrow B^{\cup x} + 1^{\cup k} \\ 0^{\cup k} + B^{\cup z} \rightarrow B^{\cup y} + 0^{\cup z} \\ 1^{\cup k} + B^{\cup z} \rightarrow B^{\cup y} + 1^{\cup z} \end{cases}$$



AND gate

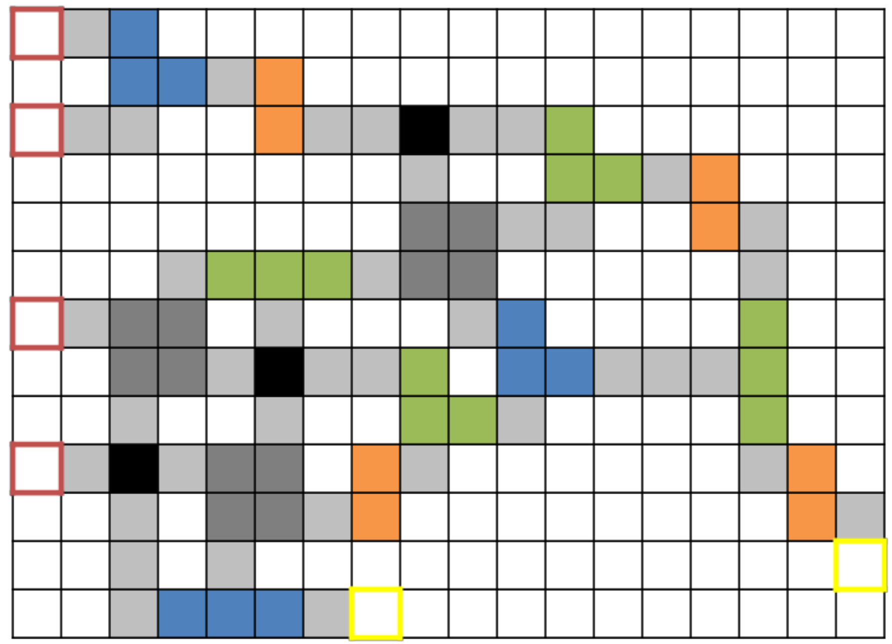
$$\begin{cases} 0^{\cap x} + 0^{\cap y} \rightarrow B^{\cap x} + 0^{\cap k} \\ 0^{\cap x} + 1^{\cap y} \rightarrow B^{\cap x} + 0^{\cap k} \\ 1^{\cap x} + 0^{\cap y} \rightarrow B^{\cap x} + 0^{\cap k} \\ 1^{\cap x} + 1^{\cap y} \rightarrow B^{\cap x} + 1^{\cap k} \\ 0^{\cap k} + B^{\cap z} \rightarrow B^{\cap y} + 0^{\cap z} \\ 1^{\cap k} + B^{\cap z} \rightarrow B^{\cap y} + 1^{\cap z} \end{cases}$$



NOT gate

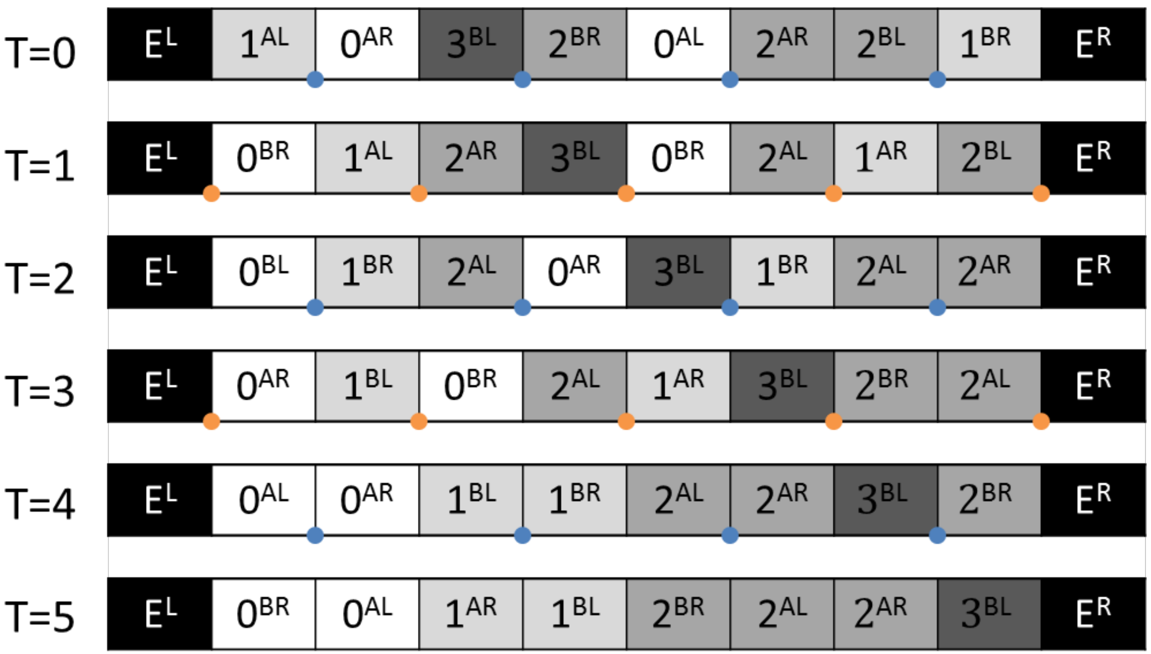
$$\begin{cases} 0^{Nx} + B^{Ny} \rightarrow B^{Nx} + 1^{Ny} \\ 1^{Nx} + B^{Ny} \rightarrow B^{Nx} + 0^{Ny} \end{cases}$$

A square-root circuit on DNA origami:



blue sites: OR, **green** sites: AND, **orange** sites: NOT
grey sites: wires, outlined sites: inputs or outputs

Dynamically-updating cellular automata with Surface CRNs



each transition rule

$$\{x, y\} \rightarrow \{x^*, y^*\}$$

is implemented with:

$$\begin{cases} x^{AL} + y^{AR} \rightarrow x^{*BR} + y^{*AL} \\ x^{BL} + y^{BR} \rightarrow x^{*AR} + y^{*BL} \end{cases}$$

edge conditions for each state x are implemented with:

$$\begin{cases} \mathrm{E}^L + x^{AR} \rightarrow \mathrm{E}^L + x^{AL} \\ \mathrm{E}^L + x^{BR} \rightarrow \mathrm{E}^L + x^{BL} \\ x^{AL} + \mathrm{E}^R \rightarrow x^{BR} + \mathrm{E}^R \\ x^{BL} + \mathrm{E}^R \rightarrow x^{AR} + \mathrm{E}^R \end{cases}$$

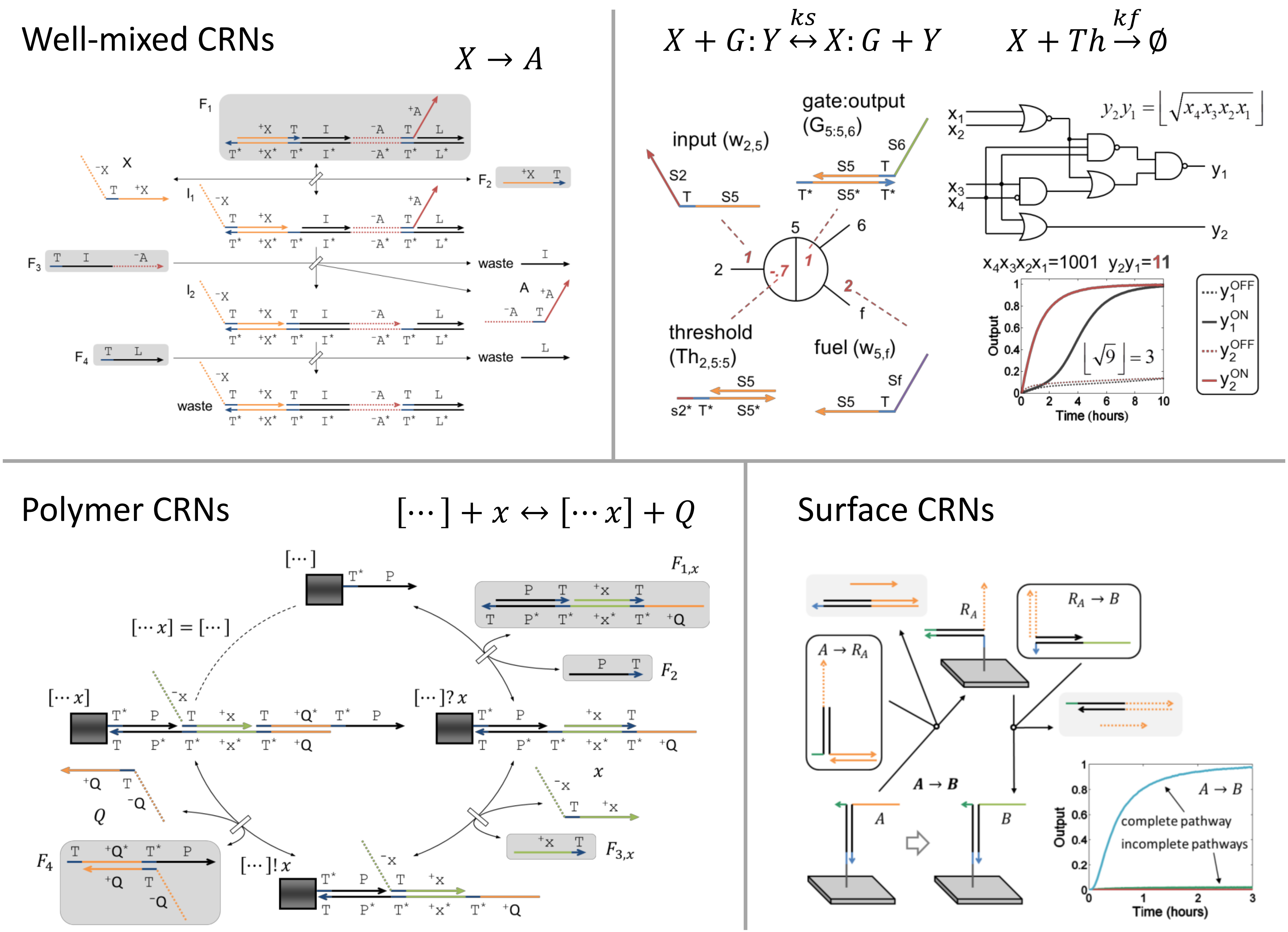
transition rules for sorting numbers between 0 to 3:

$\{0,0\} \rightarrow \{0,0\}$ $\{0,1\} \rightarrow \{0,1\}$ $\{0,2\} \rightarrow \{0,2\}$ $\{0,3\} \rightarrow \{0,3\}$

$\{1,0\} \rightarrow \{0,1\}$ $\{1,1\} \rightarrow \{1,1\}$ $\{1,2\} \rightarrow \{1,2\}$ $\{1,3\} \rightarrow \{1,3\}$

$\{2,0\} \rightarrow \{0,2\}$ $\{2,1\} \rightarrow \{1,2\}$ $\{2,2\} \rightarrow \{2,2\}$ $\{2,3\} \rightarrow \{2,3\}$

$\{3,0\} \rightarrow \{0,3\}$ $\{3,1\} \rightarrow \{1,3\}$ $\{3,2\} \rightarrow \{2,3\}$ $\{3,3\} \rightarrow \{3,3\}$



Well-mixed CRNs
X → A
F1
F2
F3
F4
I1
I2
waste
input (w2,5)
gate:output (G5:5,6)
threshold (Th2,5:5)
fuel (w5,f)
S2
S5
S6
Sf
x4x3x2x1=1001 y2y1=11
Output
Time (hours)
Polymer CRNs
[⋯] + x ↔ [⋯x] + Q
[⋯x] = [⋯]
F1,x
F2
F3,x
F4
Surface CRNs
A → RA
RA → B
A → B
complete pathway
incomplete pathways
Output
Time (hours)

Forward thinking and backward thinking

Given a computational task, how can we implement it with CRNs?

Given a few types of chemical reactions with experimentally successful implementations, what kind of interesting computational tasks can be performed?

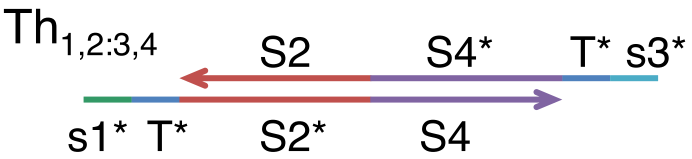
1. What kind of interesting computational tasks can be performed with the following types of well-mixed reactions?

$$X + G{:}Y \leftrightarrow X{:}G + Y$$

$$X + Th \rightarrow \emptyset$$

Feed-forward logic circuits: Qian et al, *Science* 2011
Neural networks (linear threshold circuits): Qian et al, *Nature* 2011

$$X + Y + Th \rightarrow \emptyset$$



Linear I/O systems: Oishi et al, *IET Syst. Biol.* 2011

2. What kind of interesting computational tasks can be performed with the following types of polymer reactions?

$$[\cdots] + x \leftrightarrow [\cdots x] + Q$$

Stack machines: Qian et al, *LNCS* 2011

$$[\cdots x] + y \leftrightarrow [\cdots xy] + Q$$

$$[\cdots x \cdots] + y \leftrightarrow [\cdots xy \cdots] + Q$$

3. What kind of interesting computational tasks can be performed with the following two types of surface reactions?

$$A \rightarrow X$$

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

Sequential logic, Turing machines, cellular automata : Qian et al, *LNCS* 2014

Acknowledgement

Collaborators:

David Soloveichik
Damien Woods
Erik Winfree
Jehoshua Bruck

National Science Foundation
The Molecular Programming Project
Faculty Early Career Development Award

Burroughs Wellcome Fund
Career Award at Scientific Interface

Caltech
Biology and Biological Engineering