Abstract modelling of tethered DNA circuits (APPENDICES)

Matthew R. Lakin[1], Rasmus Petersen[2], Kathryn E. Gray[2,3], and Andrew Phillips[2]

[1]Department of Computer Science, University of New Mexico, Albuquerque, NM, USA
[2]Biological Computation Group, Microsoft Research, Cambridge, UK
[3]Computer Laboratory, University of Cambridge, Cambridge, UK

mlakin@cs.unm.edu aphillip@microsoft.com

A Syntax and semantics for bulges and internal loops

To extend the DSD syntax to include bulges and internal loops, we extend the grammar in Figure 1 from the main text to introduce a new join operator (:::) which connects two segments by both the upper *and* lower strands:

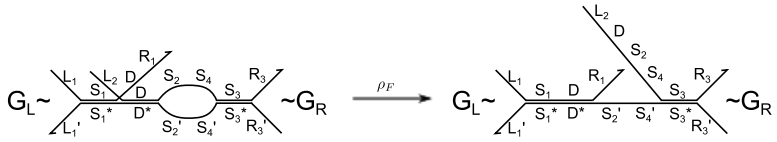
Segment join operators $\sim$ ::= : | :: | :::

This construct enables internal loops (and bulges) to be represented in the DSD language. We assume a normalization rule for gates so that $\{L_1'\}\langle L_1\rangle[S_1]:::[S_2]\langle R_2\rangle\{R_2'\}$ is normalized to $\{L_1'\}\langle L_1\rangle[S_1\ S_2]\langle R_2\rangle\{R_2'\}$, i.e., the ::: connective can only form a *non-empty* internal loop. Furthermore, extending the grammar with the ::: connective introduces the possibility multi-segment gates such as $\langle S'\}[S_1]:::[S_2]\{S''\rangle$, which corresponds to a circularized DNA strand with a double-stranded region in the middle. We assume that well-formedness checks rule out such structures.

Finally, the following are additional unimolecular reaction rules for the formation and elimination of internal loops. To simplify the semantics, we assume that internal loops are sufficiently short that nothing will bind to a domain in one of these structures. Note that the formation rule for internal loops is an instance of the *remote toehold* design concept [1].

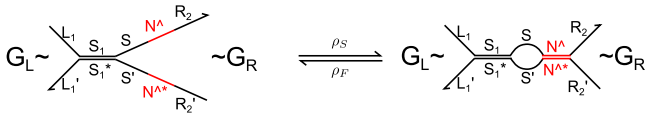
Internal loop / bulge displacement:

```
GL~{L1'}<L1>[S1]<D R1>:<L2>[D]<S2>{S2'}:::{S4'}<S4>[S3]<R3>{R3'}~GR
    -ρF-> GL~{L1'}<L1>[S1 D]<R1>{S2'}:{S4'}<L2 D S2 S4>[S3]<R3>{R3'}~GR
```



Internal toehold binding / unbinding:

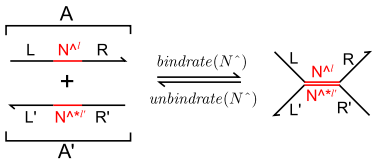
```
GL~{L1'}<L1>[S1]<S N^ R2>{S' N^* R2'}~GR
    <-ρF- ρS-> GL~{L1'}<L1>[S1]<S>{S'}:::[N^]<R2>{R2'}~GR
```



B DSD reaction rules from the main text, with ASCII representations

Two strands binding / unbinding:

```
(<L N^_l R> | {L' N^*_l' R'}) <-unbindrate(N^)- bindrate(N^)-> {L'}<L>[N^]<R>{R'}
```

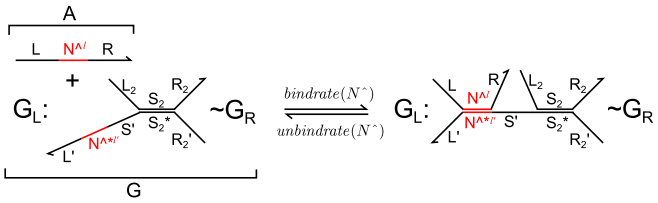


. . . where forward reaction is only derivable if $interact(\ell, \mathtt{A}, \ell', \mathtt{A}')$.

A strand binding to / unbinding from a gate:

`(<L N^_1 R> | GL:{L' N^*_1' S'}<L2>[S2]<R2>{R2'}~GR)`

$\xrightleftharpoons[unbindrate(N^)]{bindrate(N^)}$ `GL:{L'}<L>[N^]<R>{S'}:<L2>[S2]<R2>{R2'}~GR`

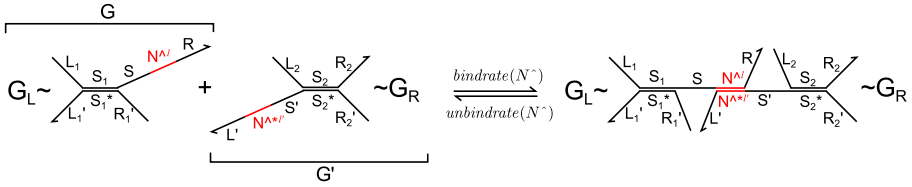


. . . where forward reaction is only derivable if $interact(\ell, \mathtt{A}, \ell', \mathtt{G})$.

Two gates binding / unbinding:

`(GL~{L1'}<L1>[S1]<S N^ R>{R1'} | {L' N^* S'}<L2>[S2]<R2>{R2'}~GR)`

$\xrightleftharpoons[unbindrate(N^)]{bindrate(N^)}$ `GL~{L1'}<L1>[S1]{R1'}::{L'}<S>[N^]<R>{S'}:<L2>[S2]<R2>{R2'}~GR`

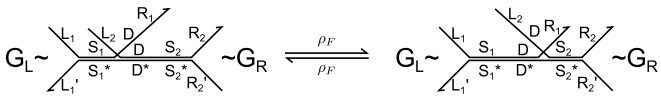


. . . where forward reaction is only derivable if $interact(\ell, \mathtt{G}, \ell', \mathtt{G}')$.

Branch migration:

`GL~{L1'}<L1>[S1]<D R1>:<L2>[D S2]<R2>{R2'}~GR`

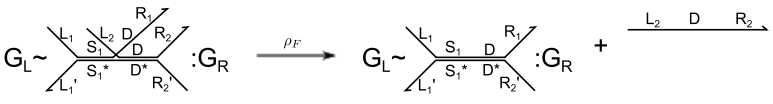
$\xrightleftharpoons[\rho_F]{\rho_F}$ `GL~{L1'}<L1>[S1 D]<R1>:<L2 D>[S2]<R2>{R2'}~GR`



Strand displacement:

`GL~{L1'}<L1>[S1]<D R1>:<L2>[D]<R2>{R2'}:GR`

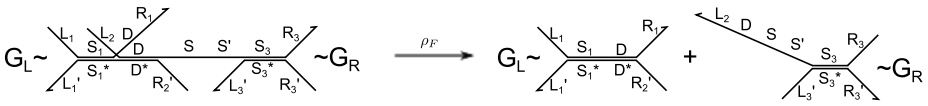
$\xrightarrow{\rho_F}$ `(GL~{L1'}<L1>[S1 D]<R1>{R2'}:GR | <L2 D R2>)`



Gate displacement:

`GL~{L1'}<L1>[S1]<D R1>:<L2>[D]<S>{R2'}::{L3'}<S'>[S3]<R3>{R3'}~GR`

$\xrightarrow{\rho_F}$ `(GL~{L1'}<L1>[S1 D]<R1>{R2'} | {L3'}<L2 D S S'>[S3]<R3>{R3'}~GR)`

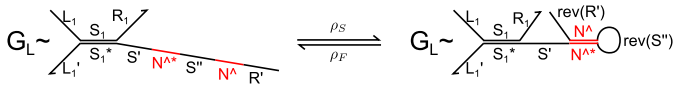


Hairpin displacement:

```
GL~{L1'}<L1>[S1]<D R1>:<L2>[D]{S2>
    --ρF--> GL~{L1'}<L1>[S1 D]<R1>{rev(S2) D rev(L2)}
```

Hairpin binding / unbinding:

```
GL~{L1'}<L1>[S1]<R1>{S' N^* S'' N^ R'}
    <--ρS--
    --ρF--> GL~{L1'}<L1>[S1]<R1>{S'}:<rev(R')>[N^]{rev(S'')>
```



C Contextual rules for DSD reactions

We must also include some rules to express that the set of reactions is closed under some basic structural operations. We write $\mathsf{mirrV}(\mathtt{X})$ for the operation of mirroring a species in the mirror plane "perpendicular to the long axis of the molecule", and $\mathsf{mirrH}(\mathtt{X})$ for the operation of mirroring a species in the mirror plane "parallel to the long axis of the molecule". For example, we have

$$\mathsf{mirrV}(\{\mathtt{d}\}\langle\mathtt{a}\rangle[\mathtt{x\ y}]\langle\mathtt{c}\rangle\{\mathtt{e}\}) = \{\mathtt{e}\}\langle\mathtt{c}\rangle[\mathtt{y\ x}]\langle\mathtt{a}\rangle\{\mathtt{d}\} \qquad \mathsf{mirrH}(\{\mathtt{d}\}\langle\mathtt{a}\rangle[\mathtt{x\ y}]\langle\mathtt{c}\rangle\{\mathtt{e}\}) = \{\mathtt{a}\}\langle\mathtt{d}\rangle[\mathtt{x}^*\ \mathtt{y}^*]\langle\mathtt{e}\rangle\{\mathtt{c}\}$$

Note that we do not need to define a 180° rotation operation, as this can be obtained by composing mirrV and mirrH (in either order, since they are commutative). We also write $\mathsf{cmp}(\mathtt{X})$ for the species obtained by complementing every domain in $\mathtt{X}$ (note that tethers are not subject to the complementation operation), noting that $D^{**} = D$. We lift these operations to parallel compositions of multiple species by applying the operation to each species pointwise. We also use a standard equivalence relation $\equiv$ on systems that encodes well-mixing as standard [2]:

$$(\textsc{refl})\ \frac{}{\mathtt{U} \equiv \mathtt{U}} \qquad (\textsc{sym})\ \frac{\mathtt{U}' \equiv \mathtt{U}}{\mathtt{U} \equiv \mathtt{U}'} \qquad (\textsc{trans})\ \frac{\mathtt{U} \equiv \mathtt{U}' \qquad \mathtt{U}' \equiv \mathtt{U}''}{\mathtt{U} \equiv \mathtt{U}''}$$

$$(\textsc{dSym})\ \frac{}{\mathtt{U} \parallel \mathtt{U}' \equiv \mathtt{U}' \parallel \mathtt{U}} \qquad (\textsc{dComm})\ \frac{}{\mathtt{U} \parallel (\mathtt{U}' \parallel \mathtt{U}'') \equiv (\mathtt{U} \parallel \mathtt{U}') \parallel \mathtt{U}''}$$

$$(\textsc{ctx})\ \frac{\mathtt{U} \equiv \mathtt{U}'}{\mathtt{U} \parallel \mathtt{U}'' \equiv \mathtt{U}' \parallel \mathtt{U}''} \qquad (\textsc{tile})\ \frac{\mathtt{T} \equiv \mathtt{T}'}{[[\mathtt{T}]] \equiv [[\mathtt{T}']]}$$

We now use the above definitions to complete the definition of the DSD reaction relation by closing under well-mixing, domain complementation and structural mirroring.

$$(\textsc{mix})\ \frac{\mathtt{U} \equiv \mathtt{U}' \qquad \mathtt{U}' \to \mathtt{U}'' \qquad \mathtt{U}'' \equiv \mathtt{U}'''}{\mathtt{U} \to \mathtt{U}'''} \qquad (\textsc{mirrV})\ \frac{\mathtt{U} \to \mathtt{U}'}{\mathsf{mirrV}(\mathtt{U}) \to \mathsf{mirrV}(\mathtt{U}')}$$

$$(\textsc{cmp})\ \frac{\mathtt{U} \to \mathtt{U}'}{\mathsf{cmp}(\mathtt{U}) \to \mathsf{cmp}(\mathtt{U}')} \qquad (\textsc{mirrH})\ \frac{\mathtt{U} \to \mathtt{U}'}{\mathsf{mirrH}(\mathtt{U}) \to \mathsf{mirrH}(\mathtt{U}')}$$

D Compilation, simulation and analysis results using DSD implementation of tethered species

In this appendix we present additional results from the compilation, simulation and analysis of the example tethered DNA strand displacement systems in Section 7 of the main text, using the latest version of the Visual DSD implementation.

D.1 Three-stator transmission line

The following DSD code implements the three-stator transmission line example in Figure 5 from the main text.

```
(* 3-stator transmission line *)
directive sample 40000.0 1000
directive simulation deterministic
directive polymers
directive localconcentrations [ (a, 100000); (b, 100000) ]
directive plot input(); fuel(); probe(); reporter()

dom a0 = { colour = "red" }
dom x  = { colour = "green" }
dom y  = { colour = "blue" }
dom r  = { colour = "purple" }
dom Q  = { colour = "black" }
dom F  = { colour = "black" }

def input()    = <a0^ s>
def fuel()     = <y^*>[s*]{x^>
def probe()    = <r^*>[s*]<Q^>{F^}
def origami()  = [[ {tether(a) a0^*}[s]{y^>
                  | {tether(a,b) x^*}[s]{y^>
                  | {tether(b) x^*}[s]{r^> ]]
def reporter() = {s F^}

( input()
| 2 * fuel()
| probe()
| origami()
| 0 * reporter()
)
```

The results from a deterministic simulation of the system are presented in Figure A1. We see that the concentration of the reporter strand approaches 1 nM by the end of the simulation, indicating that all of the transmission lines have succesfully completed signal propagation, allowing the probe to bind to the final stator and release the reporter strand. Expanding out the state space for a single copy of the system allowed us to confirm that the system has a single terminal state. The initial state is shown in Figure A2b and the terminal state is shown in Figure A2c. In particular, the terminal state corresponds to the expected result of executing the transmission line system, as shown in Figure 5 from the main text.

D.2 Communicating transmission lines

The following DSD code encodes an extension of the previous example, in which there are two initial tile species, each containing a three-stator transmission line.

```
(* Two 3-stator transmission lines on different origamis *)
directive sample 40000.0 1000
directive simulation deterministic
directive polymers
directive localconcentrations [ (a, 100000); (b, 100000) ]
directive plot input(); fuel(); probe(); reporter()

dom a0 = { colour = "red" }
dom x  = { colour = "green" }
dom y  = { colour = "blue" }
dom r  = { colour = "purple" }
dom Q  = { colour = "black" }
dom F  = { colour = "black" }
```

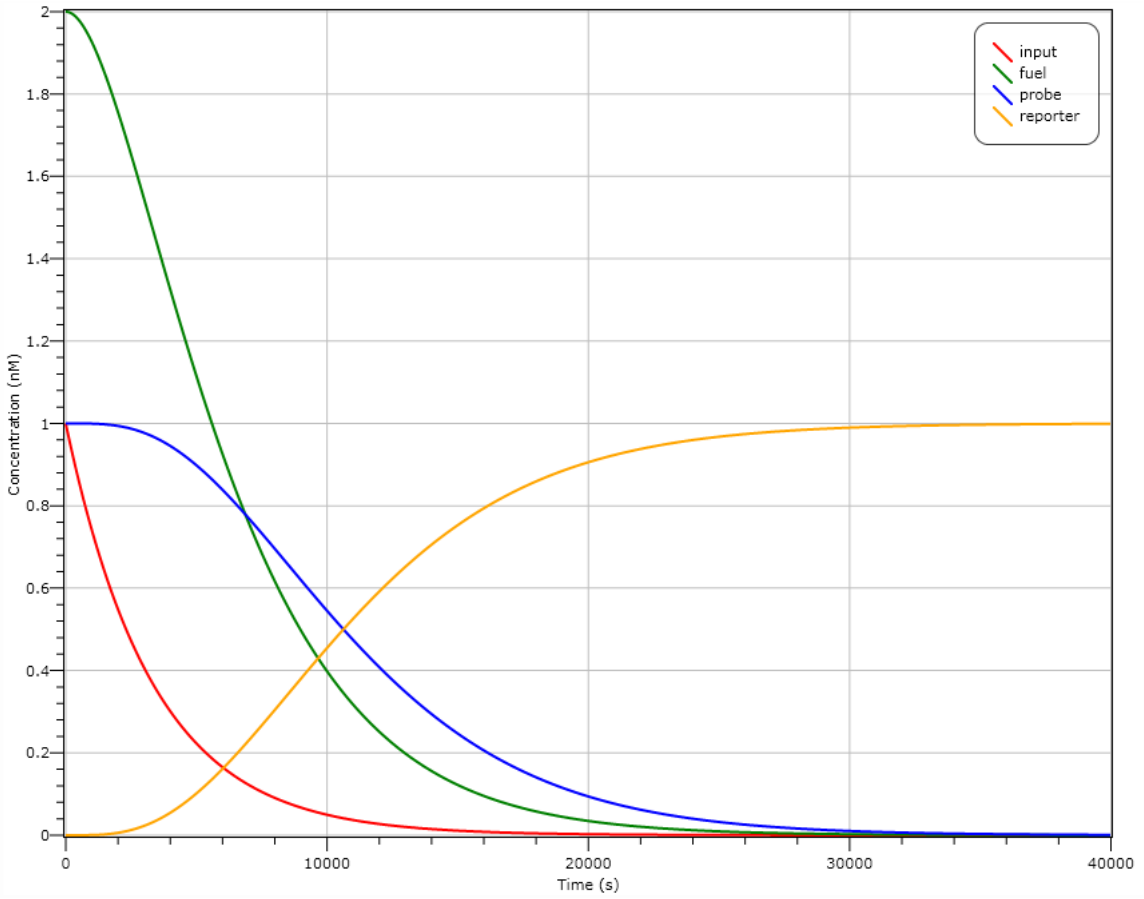
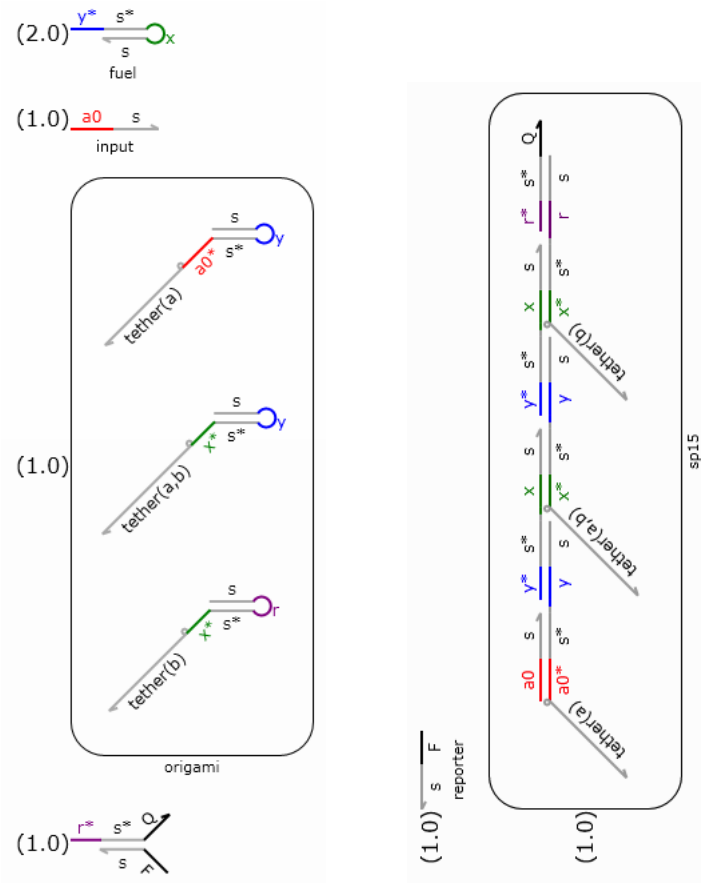


Figure A1: Simulation results for the three-stator transmission line example in Section 7 of the main text.

```
def input()    = <a0^ s>
def fuel()     = <y^*>[s*]{x^>
def probe()    = <r^*>[s*]<Q^>{F^}
def origami1()  = [[ {tether(a) a0^*}[s]{y^>
                   | {tether(a,b) x^*}[s]{y^>
                   | {tether(b) x^*}<b0^>[s] ]]
def origami2()  = [[ {tether(a) b0^*}[s]{y^>
                   | {tether(a,b) x^*}[s]{y^>
                   | {tether(b) x^*}[s]{r^> ]]

def reporter() = {s F^}

( input()
| 4 * fuel()
| probe()
| origami1()
| origami2()
| 0 * reporter()
)
```

When the signal has been passed all the way along the first transmission line, a second signal species is released that diffuses to the second tile to begin signal propagation along the second transmission line. The freely-diffusing reporter probe can only bind to the final stator of the second transmission line. This example demonstrates that species tethered to a tile can communicate with species tethered to another tile via freely-diffusing communicator species. Note that, in this example, the toehold on the communicator strand is initially exposed, as is the complementary toehold on the corresponding stator of the second tile. However, the semantics of tethered species does not allow direct interactions



(a) Initial state.

(b) Terminal state (rotated to save space).

Figure A2: State space analysis results for a single copy of the three-stator transmission line example in Section 7 of the main text.

between species tethered to different tiles, which means that these species cannot interact until the signal has been passed all the way along the first tile, which causes the communicator strand to be displaced from the tile. At this point the communicator strand is free to diffuse, so it can interact with the second tile to initiate signal transmission along the second transmission line.

The results from a deterministic simulation of the system are presented in Figure A3. We see that the concentration of the reporter strand approaches 1 nM by the end of the simulation, indicating that all of the transmission lines (on both tiles) have succesfully completed signal propagation. Expanding out the state space for a single copy of the system allowed us to confirm that the system has a single terminal state. The initial state is shown in Figure A2b and the terminal state is shown in Figure A2c. In particular, the terminal state corresponds to the expected result of executing the transmission line system, as outlined above.

D.3 Threshold-based spatial AND gate

The following DSD code implements the threshold-based spatial AND gate example in Figure 6 from the main text.

```
(* Threshold-based spatial AND gate *)
directive sample 10000.0 1000
directive polymers
directive localconcentrations [ (a, 1000000); (b, 100000);
                                (c, 1000000); (d, 100000) ]

dom a0 = { colour = "red" }
dom x  = { colour = "green" }
dom y  = { colour = "blue" }
```

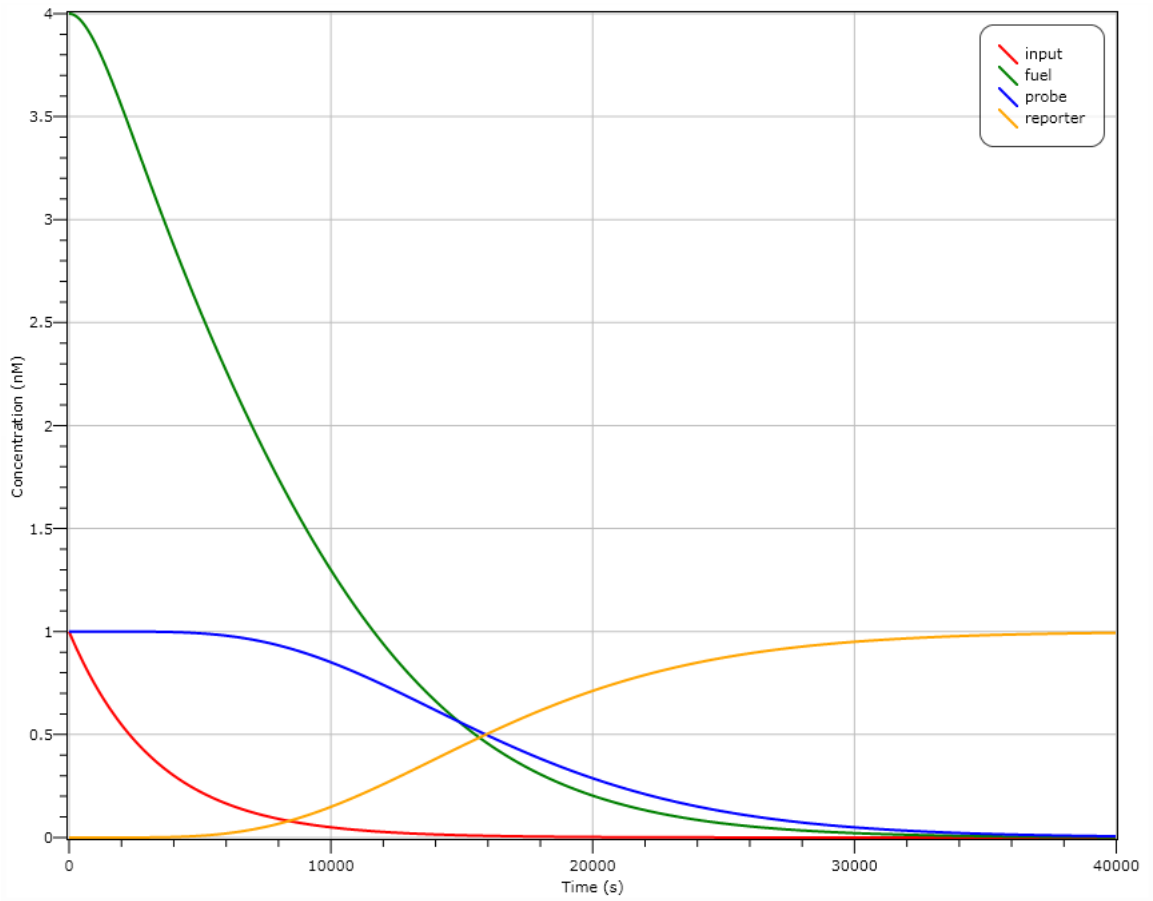
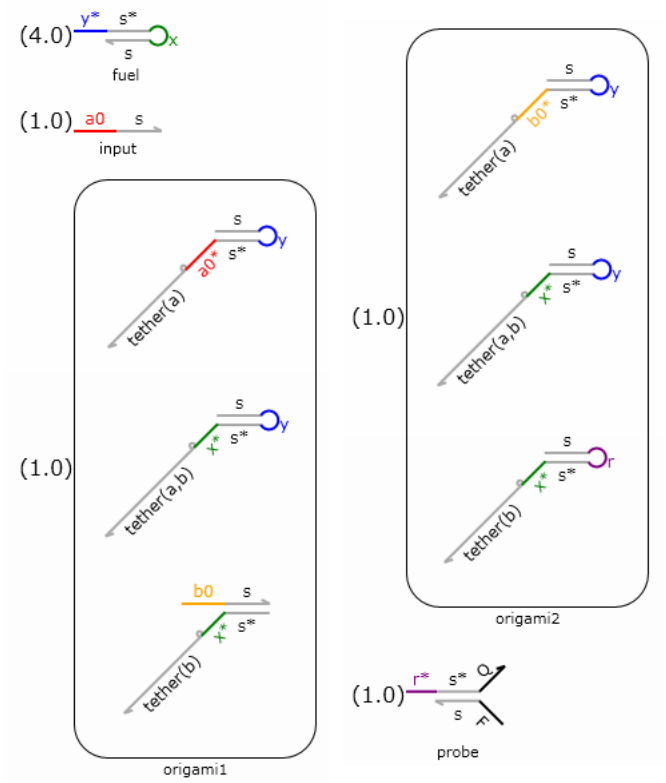


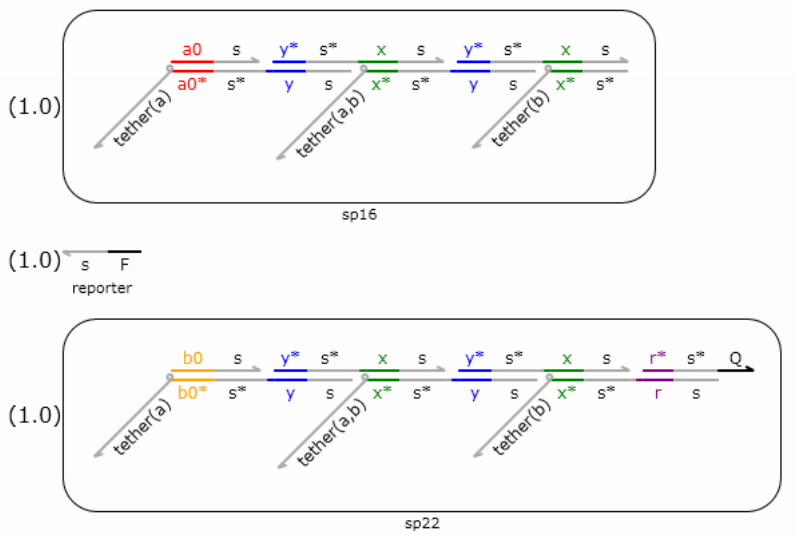
Figure A3: Simulation results for a two-tile version of the three-stator transmission line example in Section 7 of the main text.

```
dom b0 = { colour = "purple" }
dom blank = { colour = "black" }

def input1()   = <a0^ s>
def input2()   = <b0^ s>
def fuel()     = <y^*>[s*]{x^>
def origami()  = [[ {tether(a,b) a0^*}[s]{y^>
                  | {tether(c,d) b0^*}[s]{y^>
                  | {tether(a,c) x^*}[s]{blank^>
                  | {tether(b,d) x^*}[s]{y^> ]]

( input1()
(*| input2()*)
| 3 * fuel()
| origami()
)
```

Figure A5 presents the full state space for a single copy of the threshold-based spatial AND gate system, in the presence of a single input (input 1). The intuition behind this design is that the first input that binds (here, input 1) is more likely to bind to the threshold stator, meaning that both inputs are required to generate output via the alternative (slower) binding target on the output track. In our example, at the point where the state space branches, the computed rate of the reaction leading to the upper (correct) terminal state is $300\,\mathrm{s}^{-1}$, whereas the computed rate of the reaction leading to the lower (incorrect) terminal state is $30\,\mathrm{s}^{-1}$. These values agree with those in Figure 6 from the main text. The initial state is shown in Figure A6a and the two terminal states are shown in Figure A6b,c.



(a) Initial state.



(b) Terminal state.

Figure A4: State space analysis results for a single copy of the two-tile version of the three-stator transmission line example in Section 7 of the main text.

References

[1] A. J. Genot, D. Y. Zhang, J. Bath, and A. J. Turberfield. Remote toehold: A mechanism for flexible control of DNA hybridization kinetics. *J Am Chem Soc*, 133(7):2177–2182, 2011.

[2] L. Cardelli. Strand algebras for DNA computing. In R. Deaton and A. Suyama, editors, *Proceedings of DNA15*, LNCS, pages 12–24. Springer-Verlag, 2009.

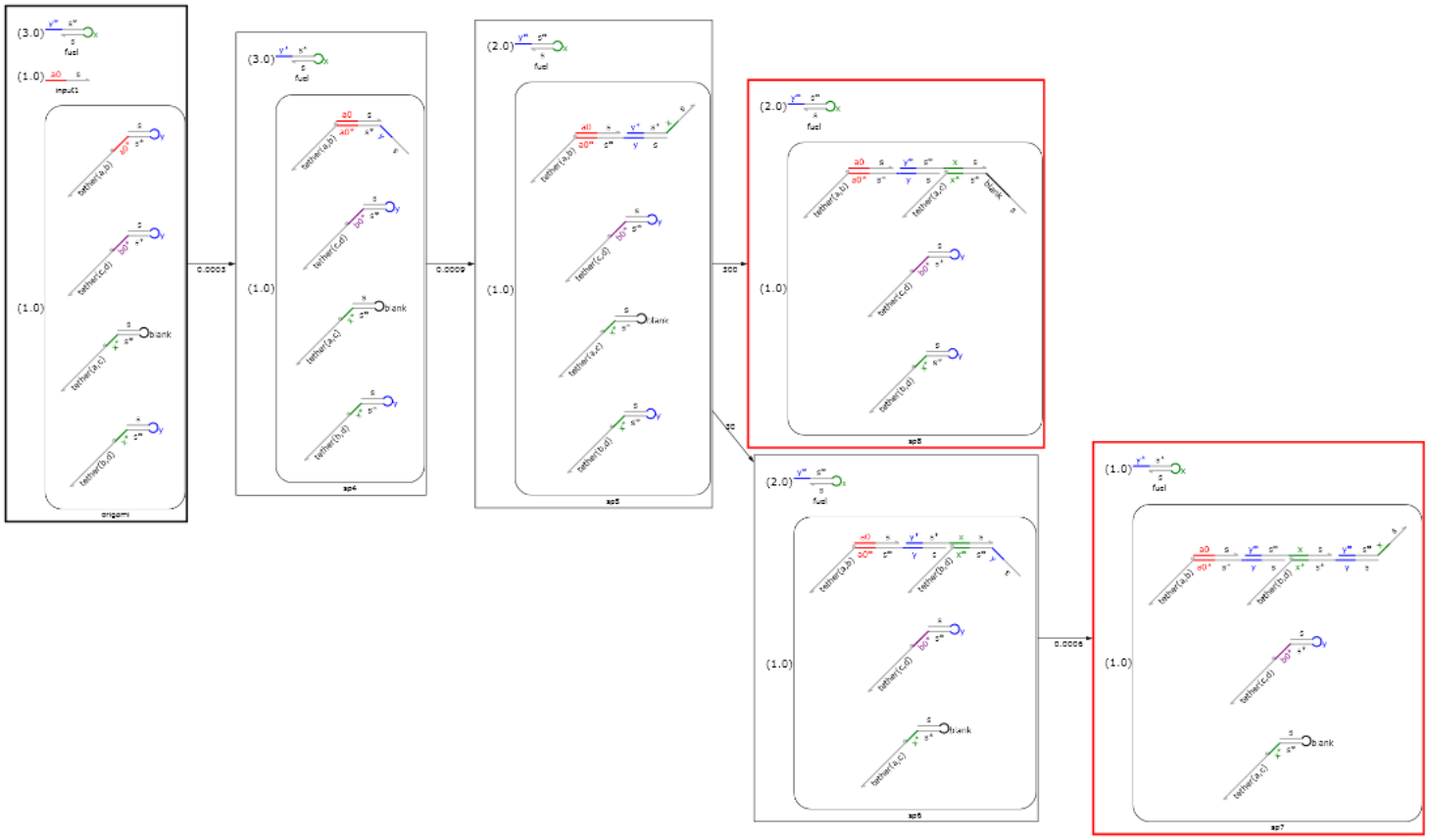
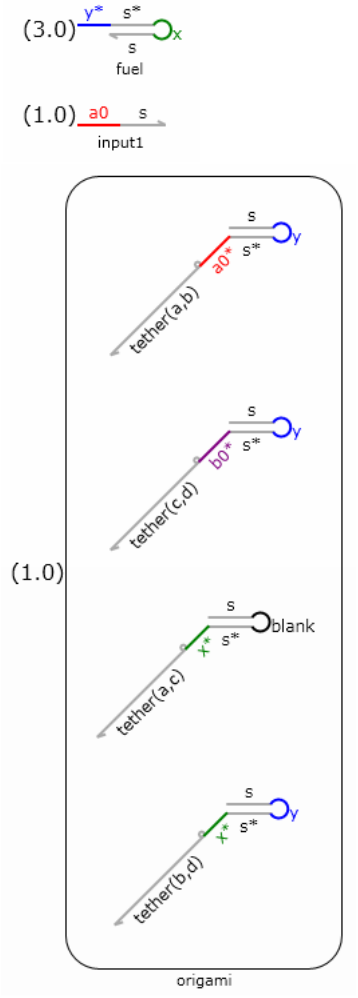
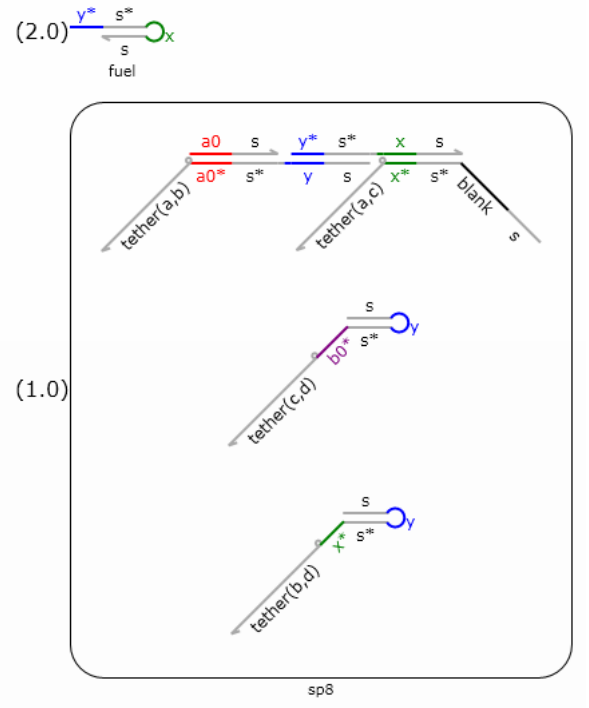


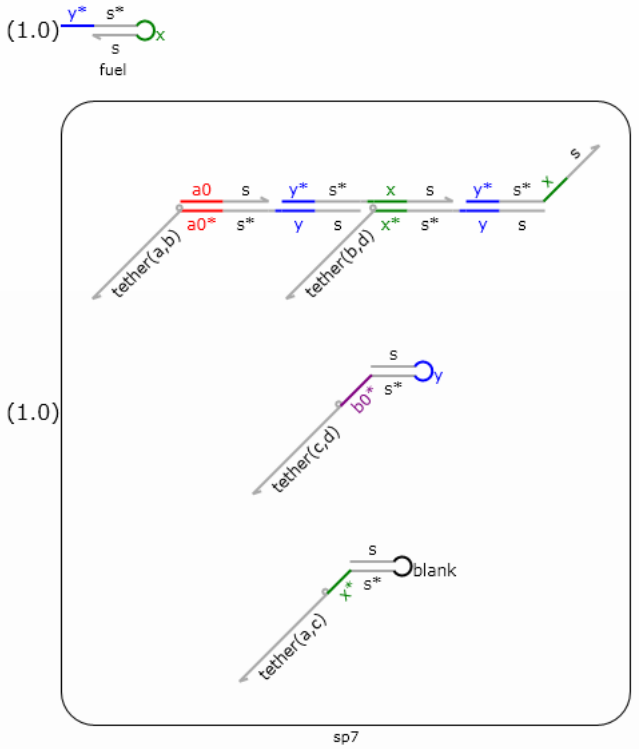
Figure A5: Full state space for a single copy of the threshold-based spatial AND gate example in Figure 6 from the main text, in the presence of a single input (input 1). The initial state is highlighted with a thick black line, and the two terminal states are highlighted with thick red lines.



(a) Initial state.



(b) Correct terminal state.



(c) Incorrect terminal state.

Figure A6: Initial and terminal states for a single copy of the threshold-based spatial AND gate in Figure 6 from the main text.