# Domain-based reaction enumeration

Karthik Sarma, Brian Wolfe, and Erik Winfree

June 10, 2010

## 1 Abstract

The programmability of nucleic acid sequences and the predictability of their interactions make them a promising substrate for molecular computing. However, the rational design of nucleic acid systems requires attention to detail at many levels of abstraction. As a result, manual design is time-consuming and error-prone. This problem is compounded by the difficulty and cost of locating problems with such systems by experiment. These complications motivate the creation of automated systems for the design and verification of nucleic acid systems at every level. The pipeline from design to strand synthesis contains several opportunities for verification. At the domain level of abstraction, software can be used to design and simulate systems in order to verify that the reaction logic functions as intended. Sequence design programs can then be used to avoid the introduction of unintended reaction pathways in order to preserve the intended logic. Domain-level verifiers to date can, however, only be used with a limited subset of non-pseudoknotted molecules. Sequence-level verifiers, on the other hand, can work with the full class of non-pseudoknotted complexes. Here, we present a domain-level enumerator that can be used to verify systems that make use of any non-psuedoknotted complex.

## 2 Introduction

Over the past decade, the idea of rationally designed nucleic acid systems has been transformed from an untested theory of the future [1] to a tried and true paradigm for molecular-scale engineering. From the use of DNA as a building block for nanoscale electronics [5] to its use in molecular robotics [4], to potential uses in pharmaceuticals, nucleic acid systems are reliable and effective. However, much as there was a long path from the development of the Stibitz' "Model K" digital computer to the widespread adoption of such devices and the development of systems that could be programmed by individuals not expert in the hardware behind the scenes, so is there a long path from the current state of DNA system development and a future in which bioengineers will be able to "program" molecular computers at a level of abstraction that allows for design unfettered by low-level concerns.

The current nucleic acid systems development paradigm occurs in a series of stages. First, the researcher conceives of a new idea, such as the use of strand displacement to catalyze complex transitions [11] or the use of hairpins to reversibly sequester chain hybridization domains [2]. Next, the researcher writes out a domain-level specification of this system, which details all of the components of the system by abstracting sequences with functional significance as single components called "domains", constructing strands out of sequences of domains, complexes out of these strands, and then detailing the intended interactions between the domains in these complexes. Finally, the researcher assigns nucleotide sequences to each domain in a manner that minimizes the potential for unwanted (by, for example, ensuring that the only long sequences that are complementary are those indicated in the specification).

This paradigm requires a great deal of manual input to go from idea to implementation. As a result, designing highly complex systems quickly becomes impossible, as the number of intended reactions and sequence constraints grows large. In addition, as complexity increases, the probability of manual error in a domain-level specification, such as a failure to recognize a potential reaction pathway or the inclusion of

an erroneous reaction pathway, increases significantly. The potential for error in manual sequence design such as the inclusion of unintended sequence complementarity in such systems similarly increases for more complex systems.

Over the past decade, the size of most nucleic acid systems has been small, so it has generally been possible to manually verify systems through exhaustive analysis. However, recent authors have begun to significantly expand the size of their systems in order to achieve new levels of capability. In order to support this trend, it is necessary to develop mechanisms for the automated verification and testing of large-scale nucleic acid systems. Many authors have previously discussed the problem of automated sequence design [9, 8]. However, to date only very limited systems for analyzing systems at the domain level have been demonstrated [6]. Here, we present a system for the automated domain-based generation of reaction pathways and reachable complexes from an initial set of reagents.

# 3 Methods

## 3.1 The domain-based notation

In order to motivate the following descriptions, we first present the domain-based notation and an example system that we will use to explain our methods.

The domain-based notation (fig. 1) simplifies the display of nucleic acid systems by abstracting sequence information into functionally-relevant components. Because rationally designed systems are typically built in a manner that relies on the interaction of decoupled pathways, this abstraction is very effective for quickly conveying the mechanism of a given system to a reader. In addition, the domain-based notation allows for the design of systems in a sequence-agnostic manner. This is useful for polymorphic systems that are intended to be adapted to different specific sequences depending on particular applications (for example, one may wish to design a polymorphic system that can detect a sequence of DNA, for any given sequence of DNA).

In the domain-based notation, domains are assumed to be completely noncomplementary unless specifically indicated otherwise. This simplification allows a reader to predict what Watson-Crick interactions are possible given only a domain-based description of a system. This capability is the basis for the enumerator discussed below. Once a domain-based system has been developed, it is then necessary to assign sequences to domains in order to make the system work. Since for reasonably-sized systems, it is generally impossible to ensure that all domains are noncomplementary, programs like NUPACK have been developed to optimize sequence assignments in order to minimize the negative effect of any unintended interactions that would be introduced [8].

We will illustrate the domain-specific notation through the following case studies that will also motivate the reaction schema abstraction discussed below.

### 3.1.1 Case Study - Entropy-driven catalytic gate

This case study is a catalytic transduction and amplification system that is polymorphic in the input strand to be amplified [11]. By using the domain-specific notation, this polymorphism is neatly abstracted away from the design.

A system diagram is shown in fig 2. In this system, strand $C$ represents the input strand to be amplified, with a sequence represented by domains 4 and 5. Strand $C$ is also the catalytic strand. The system begins with complex $S$ and strand $F$ in solution. Inspection of these molecules reveals that the most energetically favorable configuration of this system is for strand $F$ to replace the top two strands of complex $S$, which would then be free in solution (complex $W$). We know this because from inspection we can tell that $F$ contains domains that are exactly complementary to the bottom strand of $S$, and that the same number of domains (and thus the same number of bases) would be paired in complex $W$ as in the original state. However, we can also see that there are no unpaired complementary domains in complex $S$ and complex $F$. As a result, we can infer that a reaction between the two molecules is unlikely (because under the

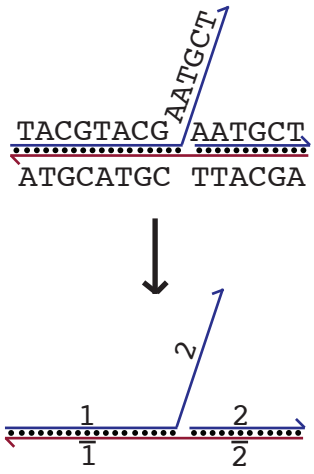



Figure 1: An illustration of the domain-specific notation. In this depiction, the top long strand and the bottom long strand are fully complementary, and the top short strand is also complementary to a portion of the bottom long strand. This complex can be broken up into four functional regions. The first region, denoted on the bottom by domain 1, consists of the left half of the complex. The region denoted by $\bar{1}$ is the complement to regions 1. The second region is denoted by domain 2, and is located both on the top short strand as well as the top long strand. Its complements, region $\bar{2}$ is located only on the bottom strand. We can tell from both the top and bottom diagrams that the top long strand can displace the top short strand to form one complex consisting of two fully hybridized strands, and one free short strand.

domain-specific notation, we assume that there is no hybridization reaction possible between domain $\bar{5}$ and any domain other than domain 5).

The addition of strand $C$, however, opens up one reaction pathway. Through toehold-mediated strand displacement [10], strand $C$ can cause the displacement of strand $SB$ through intermediate states $I1$ and $I2$, resulting in the production of intermediate $I3$. This pathway can be predicted from the domain-specific notation by noting that the free domain 5 on $C$ can bind with the free complementary domain $\bar{5}$ on $S$. This forms $I1$. In complex $I1$, a free domain 4 is directly adjacent to a bound domain 4 and is secured to the complex by domain 5. In these cases, we can predict that the free domain 4 will be able to displace the bound domain 4 to produce intermediate $I2$. In $I2$, the short domain 3 is the only domain holding strand $SB$ to the complex. As such, we can predict that it may spontaneously dissociate to form intermediate $I3$.

Through a similar mechanism, we can see that strand $F$ can bind to $I3$ and displace both $OP$ and $C$ to produce complex $W$ and regenerate the catalyst $C$. Of interest is that the pathway from $I4$ to $W$ is only one of several pathways that produce the same result. For example, strand $C$ could be displaced before strand $OP$, or both could be displaced simultaneously. This is one limitation of the domain-specific notation, as it is not possible to represent intermediate states in which branch migration is occurring in two directions at once. However, for most reactions, the particular order in which strands detach is not important, which allows this formalism to be a good approximation.

## 3.2 Reaction schema

The case study in the previous section exemplified a complete nucleic acid system specified only using the domain-specific notation. Further, it showed how one could use the domain-specific notation to predict what reactions could occur between a set of molecules. In this section, we formalize this notion using set of reaction schema that dictate what reactions can occur among a set of molecules specified in the domain-specific notation. These reaction schema are motivated by the principles of Watson-Crick hybridization as well as empirical observation of binding, release, and branch migration.

### 3.2.1 1-1 Binding

The 1-1 binding reaction is the hybridization of two complementary unpaired domains within a single complex to produce another unpseudoknotted product complex (fig. 3).

The 1-1 binding reaction is unimolecular, so it is considered "fast" in the enumeration.

### 3.2.2 2-1 Binding

The 2-1 binding reaction is the hybridization of two complementary unpaired domains, each in a different complex, to produce a single, unpseudoknotted product complex containing all of the strands contained in both of the original complexes (fig. 4).

The 2-1 binding reaction is bimolecular, so it is considered "slow" in the enumeration.

### 3.2.3 1-1 Release

The 1-1 release reaction is the complete dissociation of a complete helix whose total length is less than a threshold, producing a single connected complex (fig. 5). This threshold models the empirical idea that short helices will dissociate quickly and spontaneously, whereas longer helices will not dissociate on meaningful timescales.

The 1-1 release reaction is unimolecular, so it is considered "fast" in the enumeration.

### 3.2.4 1-2 Release

The 2-1 release reaction is the complete dissociation of a complete helix whose total length is less than a threshold, producing two connected complexes (fig. 6). As in the 1-1 release reaction, the threshold models the differences between the rates of dissociation of short and long helices.

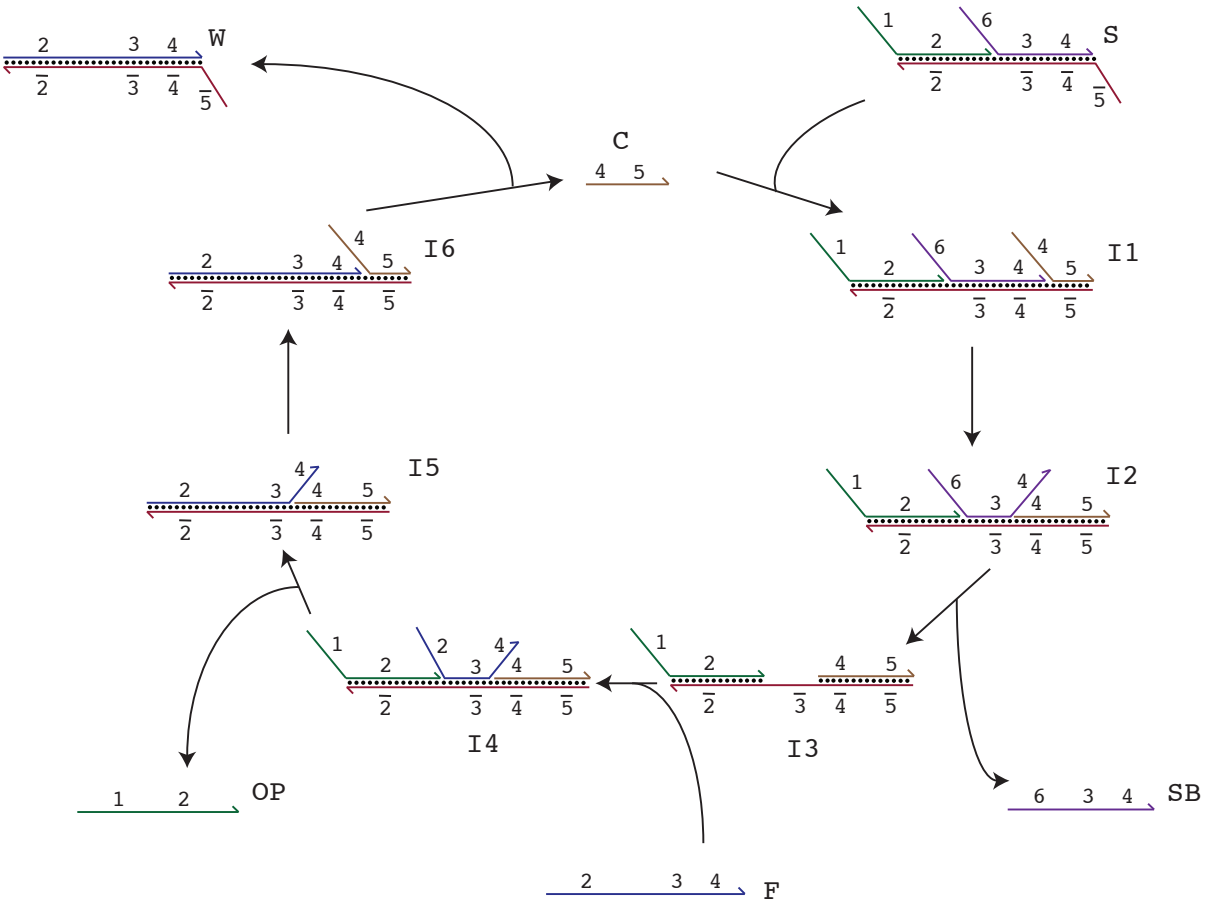



Figure 2: The enzyme-free catalytic system of Zhang et. al. This system transduces signal strand $C$ to outputs $SB$ and $OP$ and amplifies catalytically. The system is polymorphic in the sequence of strand $C$, which is represented by domains 4 and 5. In this system, domains $3, \bar{3}, 5$, and $\bar{5}$ are short (and so may dissociate spontaneously on fast timescales) and all other domains are long.

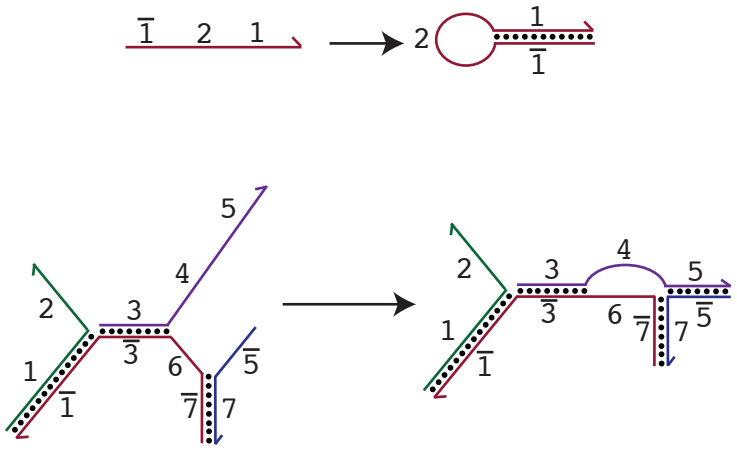



Figure 3: 1-1 binding reactions. Top: A single strand binds to itself to form a hairpin loop. Bottom: Two strands within a complex hybridize to form a multiloop.

The 1-2 release reaction is unimolecular, so it is considered "fast" in the enumeration.

#### 3.2.5 3-way Branch Migration

A three-way branch migration is a unimolecular displacement reaction where an unpaired segment of a strand displaces one side of a double-stranded region in the same external loop as the unpaired segment (the region must be fully complementary to the unpaired segment). This may cause the displaced strand to detach from the original complex if it becomes disconnected from the rest of the complex. Thus, the reaction can produce either one or two unpseudoknotted complexes. The migration reaction proceeds with one domain in each reaction step. This reaction is shown in figure 7.

The 3-way branch migration reaction is unimolecular, so it is considered "fast" in the enumeration.

#### 3.2.6 4-way Branch Migration

A four-way branch migration is a unimolecular rearrangement reaction in which a 4-arm junction rearranges around the center, shifting the configuration of hybridized pairs of domains. This reaction can produce one or two product complexes depending on the original configuration. This reaction is shown in figure 8.

The 4-way branch migration reaction is unimolecular, so it is considered "fast" in the enumeration.

The system detailed in section 3.1.1 is labelled with appropriate reactions in figure 9.

### 3.3 Identification of transient complexes

Given the reaction schema above, one possible method for generating a reaction graph from initial reagents could simply be to take every complex and every pair of complexes currently known to be reachable from the initial set and generate all of the complexes that can be produced though the use of any of the possible reaction classes described above. This process could then proceed iteratively until no new complexes are generated. Though this would work for some systems, the following case study illustrates a major failure mode of this method. We will also present a method for eliminating this failure mode that we use in our algorithm.

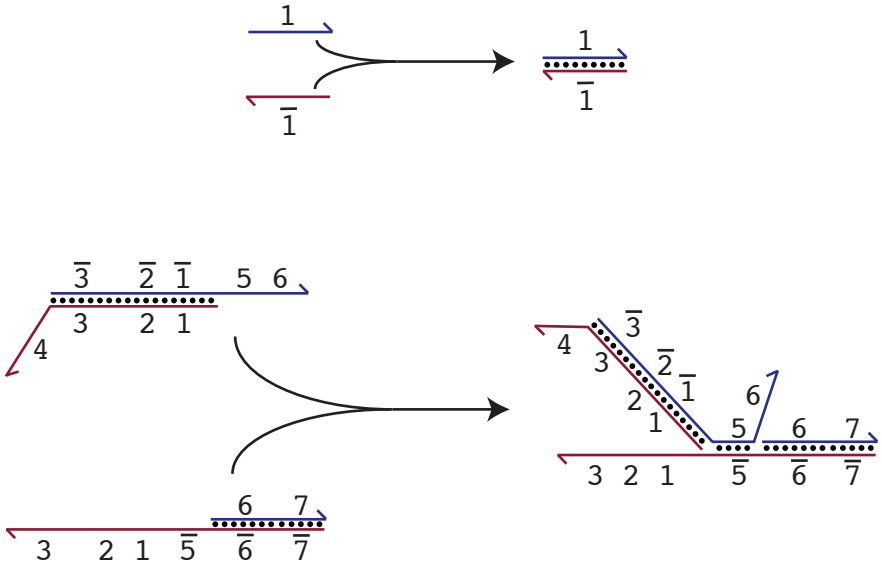



Figure 4: 2-1 binding reactions. Top: Two strands bind to form a single complex. Bottom: Two complexes bind to form a single complex.

### 3.3.1 Case Study - Gate cascade

Figure 10 depicts a simple system in which an input strand ($In$) causes the release of an output strand ($Out$) through a series of intermediate steps. First the input strand displaces strand $T$ from gate $G1$ through toehold-mediated strand displacement (in which a 2-1 bind reaction, then a 3-way branch migration reaction, and finally a 1-2 release reaction occur), passing through intermediate $I1$ (this intermediate is part of the way through the 3-way branch migration reaction and results from our earlier definition in which 3-way branch migration only occurs one domain at a time). Strand $T$ then interacts with gate $G2$ and again performs toehold-mediate strand displacement to produce strand $Out$, passing through intermediate $I2$.

However, under the reaction schema presented above, there are other possible pathways, assuming that the strands and complexes exist with copy number greater than one. Intermediates $I1$ and $I2$ could bind together to form complex $P1$. In addition, because $P1$ has both an unbound domain 1 and an unbound domain $\bar{1}$, $P1$ can then polymerize to arbitrary length, creating an infinite number of reachable complexes (in addition, complex $I1$ can also polymerize, since it has the same free domains).

This kind of polymerization, however, is not something that we would expect to happen empirically in most cases. To illustrate why, we can estimate the various reaction rates involved. Consider the reaction consisting of the transition from the intermediate in which strand $In$ has just bound to complex $G1$ (with the domain 3 holding $In$ to $G1$) to the state in which strand $T$ has been displaced. We can estimate the rate of this reaction as $[I1] * 20$ s$^{-1}$ [10]. Now considered the reaction between $I1$ and $I2$ to form $P1$. As an order of magnitude estimate, we can take the rate of hybridization of this reaction as $[I1][I2] * 3 * 10^6$ mol$^{-1}$ s$^{-1}$ [10]. Comparing these rates, we can see that for reactions conducted at concentrations below the micromolar range, the branch migration reaction is likely to complete much faster than hybridization can occur. Since many experiments are performed in the nanomolar range, this explains why we do not see polymerization of branch migration intermediates empirically.

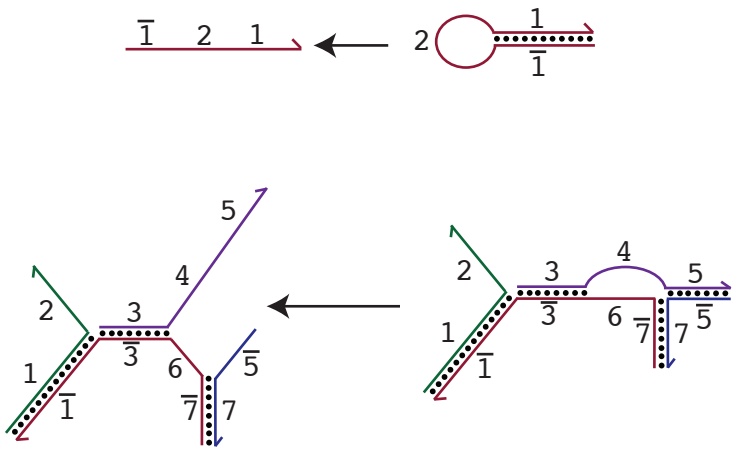



Figure 5: 1-1 release reactions. These reactions are the reverse of the previously shown 1-1 bind reactions.

#### 3.3.2 Identification

In order to model this distinction, we assume for our enumerator that the input systems are to be run at low concentration. Later, we will discuss some of the limitations that this assumption causes. Under this assumption, we classify some reactions as "slow" and others as "fast," where the bimolecular reactions are "slow," and all other reactions are "fast." We then only allow a restricted class of complexes to undergo "slow" reactions. This restricted class, which we denote as "resting states," does not include complexes that are not likely to be present on timescales long enough for biomolecular reactions to occur. Those complexes that are not resting states are referred to as "transient" complexes.

To identify complexes that are transient, we use the intuition that any complex that has a "fast" reaction open to it is likely to undergo that reaction before an interaction with another complex can occur. This intuition fails, however, in situations where there is a cycle of "fast" reactions available in which a complex can transition several times before transitioning to itself. In these situations, we consider two cases. In one case, there is a fast reaction from one of the complexes in the cycle to a complex that is not connected by a series of fast reactions back to any segment of the cycle (which could also be referred to as a set of complexes that are strongly connected by "fast" reactions, or in other words a set in which any two complexes are connected by a series of "fast" reactions). In this case, reactants are likely to go around the cycle quickly until they finally transition through this pathway out of the cycle, after which they cannot re-enter it. Thus, the complexes in the cycle are all transient.

In the other case, there is no such "exit" path from the cycle. Hence, the reactants will simply continually transition between the states in the cycle until a bimolecular reaction happens that removes a reactant. Thus, in this case, every complex in the cycle is considered a "resting state."

### 3.4 Graph enumeration

We will now present a method for enumerating all of the domain-based reactions possible given a starting set of reagents under the constrains discussed above. The algorithm centers around the creation and maintenance of five sets of complexes.

The first set is the set of resting state complexes which have already had all possible unimolecular and bimolecular reactions possible within the set enumerated. This set is denoted as set $E$. The second set is the set of transient complexes which have had all unimolecular reactions enumerated (since they are transient complexes, they are not eligible to undergo bimolecular reactions). This set is denoted as set $T$. The third

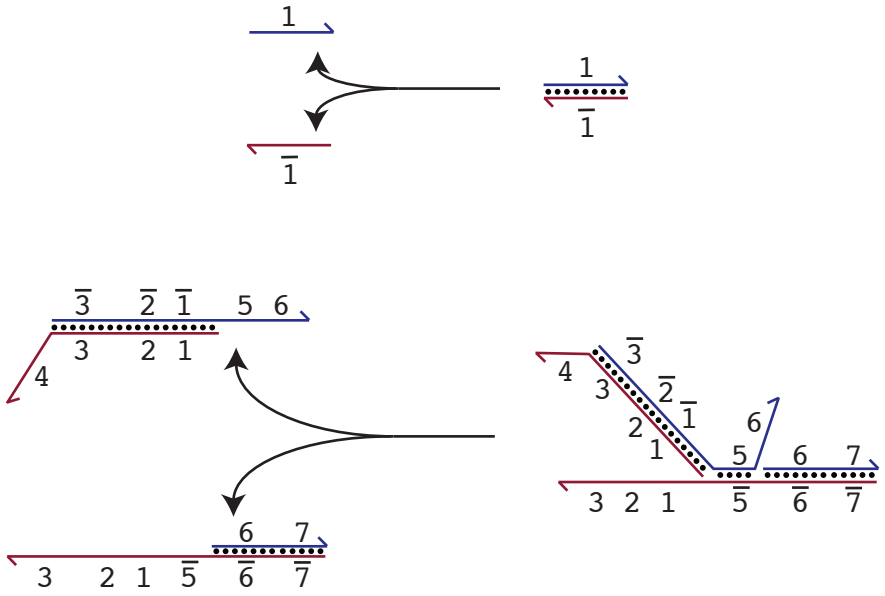



Figure 6: 1-2 release reactions. These reactions are the reverse of the previously shown 2-1 bind reactions.

set is the set of resting state complexes which have not yet had any bimolecular reactions that they might be a reagent in enumerated. This set is denoted as set $S$. The fourth set is the "current neighborhood." It consists of the set of products of unimolecular reactions which can be produced from a pathway starting from the most recently considered complex to be enumerated. This set is denoted as set $N$. The fifth set is the set of starting reactants. This set is denoted as set $B$.

The algorithms works as follows. First, we process all of the starting reagents. For every complex $b \in B$, we add $b$ to the empty set $F$. Then, while $F$ is not empty, we remove an element $f$ from $F$ and add it to $N$. We then compute all of complexes that can be reached from $f$ using one unimolecular reaction step, and add them to $F$ unless they already exist in a different set.

This process terminates eventually because at every step, we remove something from $F$ and add it to $N$. Since a finite number of unique complexes can be generated through unimolecular reactions from any given starting complex, we will eventually have moved all of these from $F$ to $N$ (and once a complex is in $N$, it cannot be added to $F$).

We then use Tarjan's algorithm [7] to find the simply connected neighborhoods (SCCs) of the graph defined by the elements of $N$ and the fast reactions between them (we know they are all connected by fast reactions to complex $b$). For each SCC, we then determine if that SCC has any outgoing fast edges. If so, we add all of the elements of that SCC to the set of transient complexes, $T$. If not, then that SCC (which could be only one complex) represents a resting state, so we add all of the elements of the SCC to list $S$ for further processing.

Once the set $B$ has been exhausted, we continue on to generating the parts of the graph accessible only through unimolecular reactions.

For every complex $s$ in $S$, we add $s$ to set $E$. Then, for every complex $e$ in $E$, we add any complexes that can be generated through a bimolecular reaction between $e$ and $s$ to empty set $B$. We then repeat the process above using the new list $B$, possibly adding additional complexes to lists $S$ and $T$ while avoiding adding duplicates. In order to protect against the possibility of polymerization, any complexes with more than a threshold number of strands that would be generated through a reaction between $e$ and $s$ are considered not

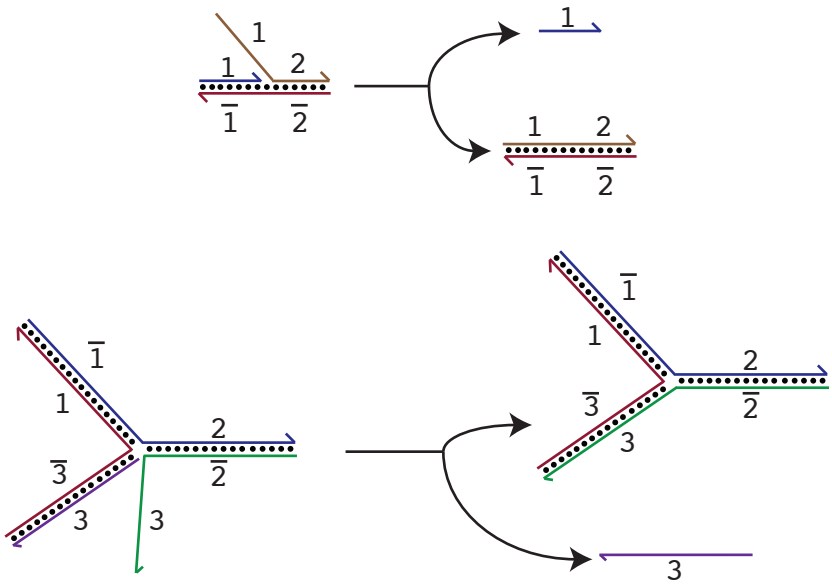



Figure 7: 3-way branch migration reactions. Top: One strand displaces another, causing it to detach from a complex. Bottom: One strand displaces another using a remote toehold; despite the fact that the displaced strand is bound to a different strand than the displacing strand, the junction allows toehold-mediated strand displacement to occur.

to be reachable, and an error is printed to the log to alert the user of the possible polymerization.

## 3.5 Output

### 3.5.1 Graphical Output

The enumerator supports graphical output using the open source tool graphviz [3]. In the default mode, a graph with a node for every unique complex that can be reached and arrows denoting every reaction between them is generated. To distinguish bimolecular reactions from multiple unimolecular reactions, separate 'reaction nodes' are used to denote bimolecular reaction pathways (fig. 11(b)).

For complex systems, however, the number of complexes and reactions could be very high, making the output unreadable. In order to alleviate this problem, we also support a reduced form of graphical output. In this form, only nodes for resting states (with resting states that consist of multiple complexes represented by only one node) are generated, and all reactions are depicted as being between resting states. For example, if resting state complex $A$ could transition to transient complex $B$ and then transition to resting state complex $C$, the reduced output would only display nodes for complexes $A$ and $C$ and a reaction from $A$ to $C$. In this way, the outline of the functioning of a complex system can be quickly ascertained, with further detail being provided by the textual output.

### 3.5.2 Textual Output

The enumerator also supports textual output, in which a list of all complexes and the reactions between them is printed. In order to support interoperability with downstream clients, output via JavaScript Object Notation (JSON), Systems Biology Markup Language (SBML), or plain text is possible.

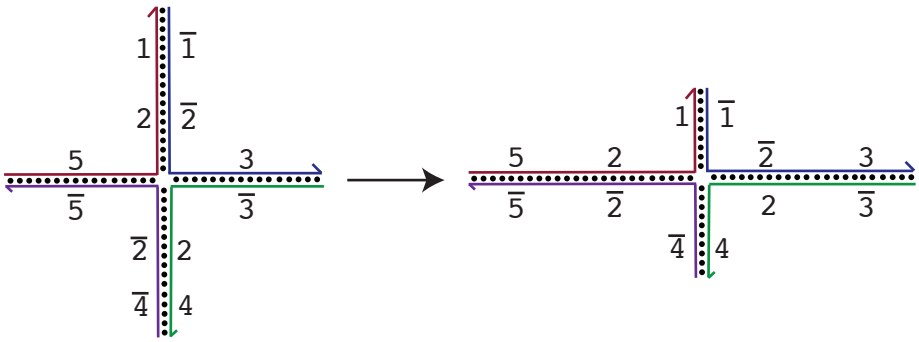



Figure 8: A 4-way branch migration reaction. The reverse reaction is also a 4-way branch migration.

### 3.6 Examples

We have tested this system on a number of previously demonstrated systems, and provide an example in fig 11.

## 4 Discussion

We have demonstrated here an automated system for determining the reaction graph generated by a set of starting DNA reagents. This system can handle arbitrary unpseudoknotted Watson-Crick interactions, and allows for the user to extend the underlying reaction rule system when needed. Using this system, it will now be possible to detect certain types of system pathway errors and search for reaction pathways between end-state complexes that were unintended. This will simplify the design and testing of DNA systems and allow for the debugging of such systems *in silica*. When combined with an appropriate sequence design program, such as NUPACK, both sequence-specific and design-specific errors can be caught before experimental validation.

There are many areas for future work in this program. Allowing for the automatic validation of systems against a desired set of reaction pathways would further simplify the system design problem. In addition, allowing for automated transfer of systems to sequence design software packages and then the automated production of mass-action simulations using rate estimates for reactions in the graph would allow for partial validation at the kinetic level.

Further refinements of the graphical output of the enumerator could also enhance usability. Filtering out unimportant 'dead-end' pathways and transient complexes could allow the user to more rapidly detect meaningful errors. Allowing users to 'zoom in' to the neighborhoods of particular portions of the graph could also allow for low-level pathway verification.

Lifting the restriction against psuedoknotted structures could also help with system validation. Though algorithmic constraints may prevent enumerators like this from working with full pseudoknotted reaction graphs, future enumerators could at least warn users that pseudoknotted pathways might exist would allow users to manually investigate these potential errors.

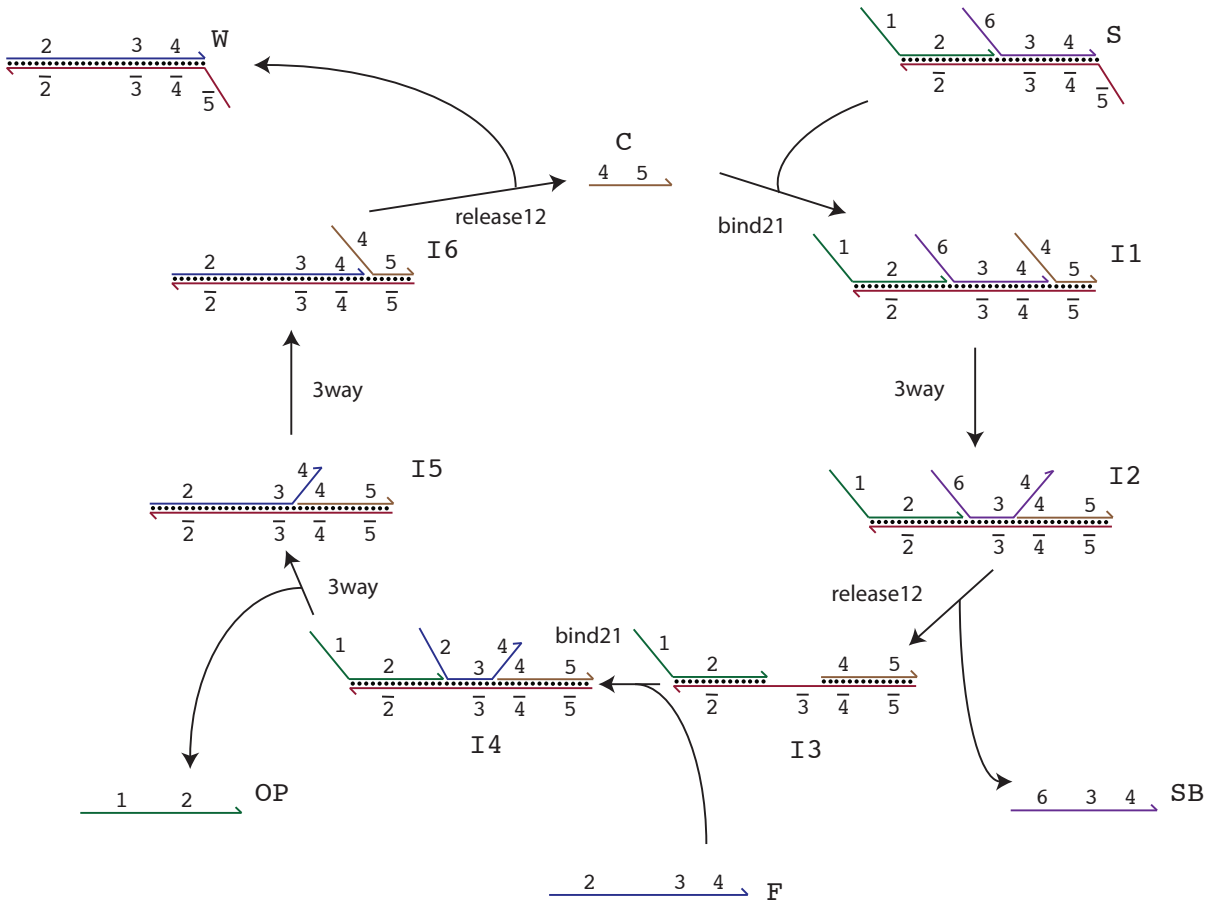



Figure 9: The reactions in this pathway consist of 2-1 binding reactions (bind21), 1-2 release reactions (release12), and 3-way branch migration (3way).

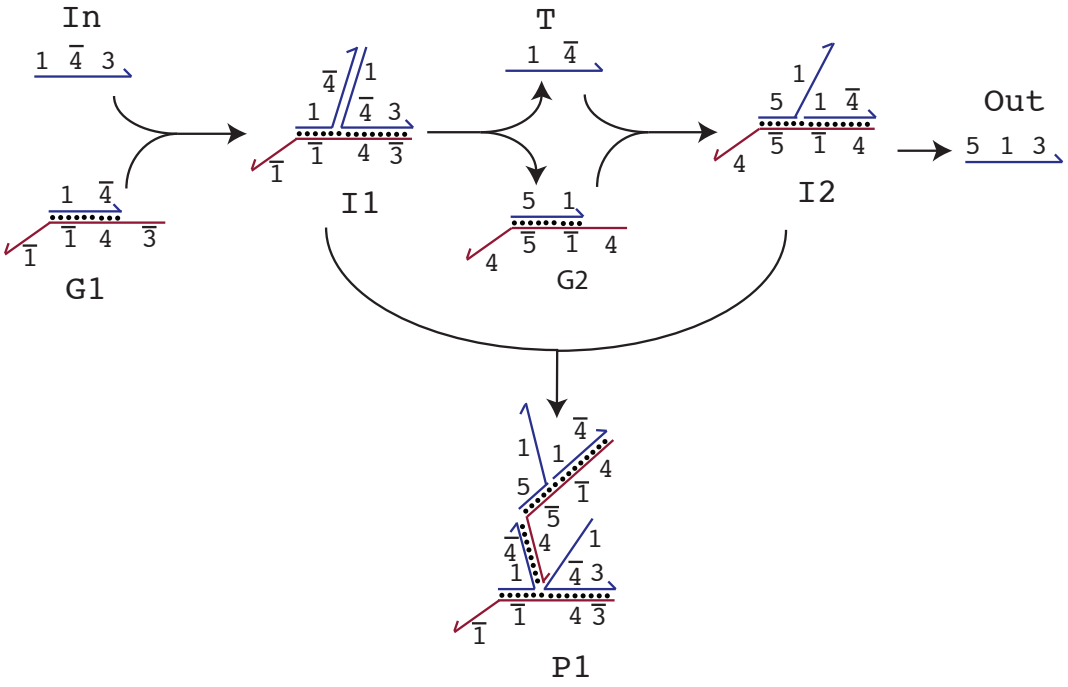



Figure 10: Simple gate-mediated system in which an input strand causes the release of an output strand. In this system, domains 1 and 4 are long, and all other domains are short.

## References


[1] L. M. Adleman. Molecular computation of solutions to combinatorial problems. *Science*, 266(5187): 1021–1024, 1994.

[2] R. M. Dirks and N. A. Pierce. Triggered amplification by hybridization chain reaction. *Proceedings of the National Academy of Sciences of the United States of America*, 101(43):15275–15278, 2004.

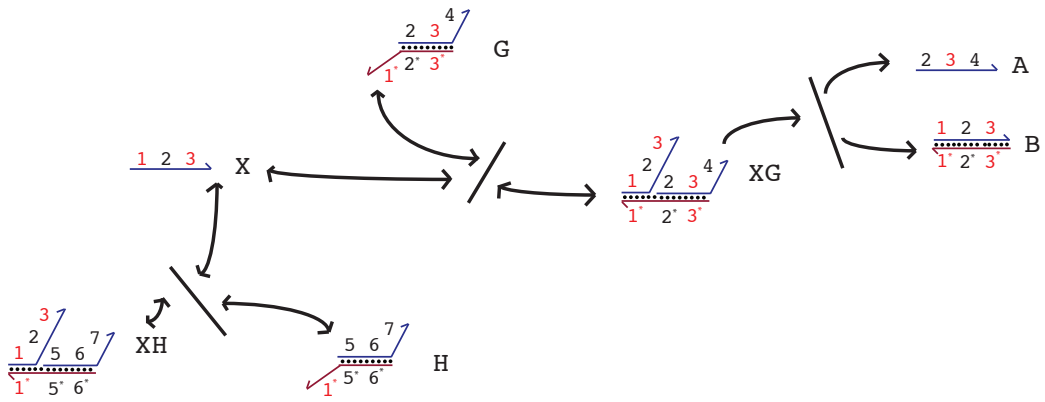
[3] E. R. Gansner and S. C. North. An open graph visualization system and its applications to software engineering. *Software-Practice and Experience*, 30, 2000.

[4] K. Lund, A. J. Manzo, N. Dabby, N. Michelotti, A. Johnson-Buck, J. Nangreave, S. Taylor, R. J. Pei, M. N. Stojanovic, N. G. Walter, E. Winfree, and H. Yan. Molecular robots guided by prescriptive landscapes. *Nature*, 465(7295):206–210, 2010.

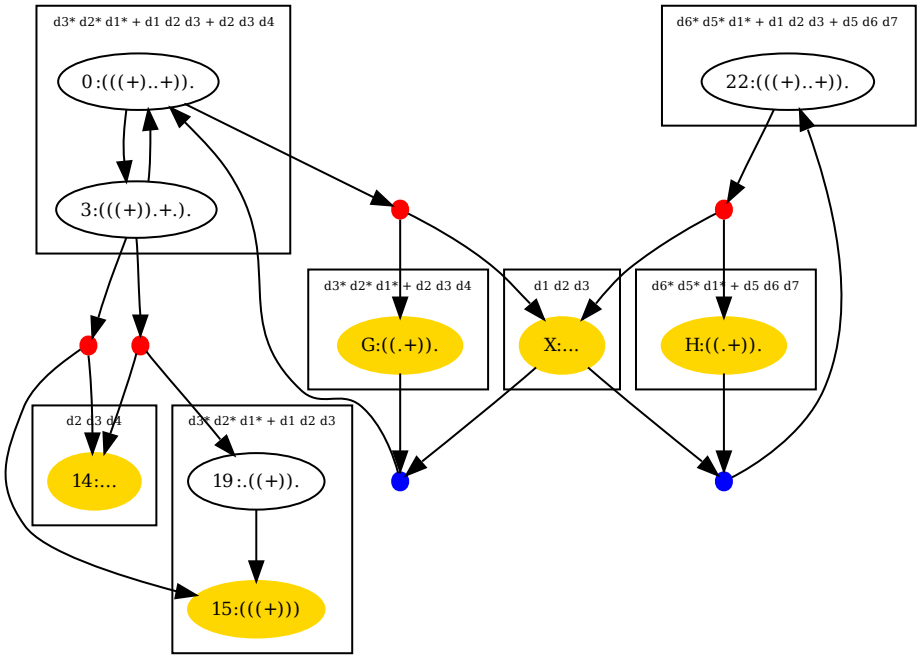
[5] H. T. Maune, S. P. Han, R. D. Barish, M. Bockrath, W. A. Goddard, P. W. K. Rothemund, and E. Winfree. Self-assembly of carbon nanotubes into two-dimensional geometries using DNA origami templates. *Nature Nanotechnology*, 5(1):61–66, 2010.

[6] A Phillips and L Cardelli. A programming language for composable DNA circuits. *Journal of The Royal Society Interface*, 6:S419–S436, 2009.

[7] Robert Tarjan. Depth-first search and linear graph algorithms. *SIAM J. Computing*, 1972.




(a) Manual system diagram.




(b) Generated reaction graph.

Figure 11: A simple strand displacement system. The complex labeled 14 is in fact strand A, and the complex labelled 15 is in fact complex B. The system has correctly identified the transient and end-state complexes (end-states are in yellow).

[8] J. N. Zadeh, B.R. Wolfe, and N.A. Pierce. Nucleic acid sequence design via efficient ensemble optimization. *Submitted*, 2010.

[9] D. Y. Zhang. Towards domain-based sequence design for DNA strand displacement reactions, 2010.

[10] D. Y. Zhang and E. Winfree. Control of DNA strand displacement kinetics using toehold exchange. *Journal of the American Chemical Society*, 131(47):17303–17314, 2009.

[11] D. Y. Zhang, A. J. Turberfield, B. Yurke, and E. Winfree. Engineering entropy-driven reactions and networks catalyzed by DNA. *Science*, 318(5853):1121–1125, 2007.