A Coupled Reconfiguration Mechanism for Single-Stranded DNA Strand Displacement Systems

Hope Amber Johnson
The University of British Columbia, Vancouver, Canada

Anne Condon
The University of British Columbia, Vancouver, Canada

Abstract

DNA Strand Displacement (DSD) systems model basic reaction rules, such as toehold-mediated strand displacement and 4-way branch migration, that modify complexes of bound DNA strands. DSD systems have been widely used to design and reason about the correctness of molecular programs, including implementations of logic circuits, neural networks, and Chemical Reaction Networks. Such implementations employ a valuable toolkit of mechanisms – sequences of basic reaction rules – that achieve catalysis, reduce errors (e.g., due to leak), or simulate simple computational units such as logic gates, both in solution and on surfaces. Expanding the DSD toolkit of DSD mechanisms can lead to new and better ways of programming with DNA.

Here we introduce a new mechanism, which we call *controlled reconfiguration*. We describe one example where two single-stranded DSD complexes interact, changing the bonds in both complexes in a way that would not be possible for each independently on its own via the basic reaction rules allowed by the model. We use *coupled reconfiguration* to refer to instances of controlled reconfiguration in which two reactants change each other in this way. We note that our DSD model disallows pseudoknots and that properties of our coupled reconfiguration construction rely on this restriction of the model.

A key feature of our coupled reconfiguration example, which distinguishes it from mechanisms (such as 3-way strand displacement or 4-way branch migration) that are typically used to implement molecular programs, is that the reactants are single-stranded. Leveraging this feature, we show how to use coupled reconfiguration to implement Chemical Reaction Networks (CRNs), with a DSD system that has both single-stranded signals (which represent the species of the CRN) and single-stranded fuels (which drive the CRN reactions). Our implementation also has other desirable properties; for example it is capable of implementing reversible CRNs and uses just two distinct toeholds. We discuss drawbacks of our implementation, particularly the reliance on pseudoknot-freeness for correctness, and suggest directions for future research that can provide further insight on the capabilities and limitations of controlled reconfiguration.

2012 ACM Subject Classification Theory of computation; Applied computing → Chemistry

Keywords and phrases Molecular programming, DNA Strand Displacement, Chemical Reaction Networks

Digital Object Identifier 10.4230/LIPIcs.DNA.2022.3

Funding *Hope Amber Johnson*: Supported by an NSERC Discovery Grant.
Anne Condon: Supported by an NSERC Discovery Grant.

1 Introduction

1.1 Summary

An important task in the field of molecular programming is to develop effective molecular programming languages, and to develop an effective abstraction hierarchy. Ideally, human-readable and human-writable programs at the top level of the hierarchy can be transformed

without further human input to a molecular implementation at the bottom level. A strong candidate pair of layers is the Chemical Reaction Network (CRN) model and its implementation with DNA Strand Displacement (DSD) systems. CRNs are a particularly well-studied abstract model of well-mixed chemical systems, in which many interesting programs can be and have been written [2, 3, 28, 36, 41]. Moreover, arbitrary CRNs can be transformed into DSD systems that implement the original CRN [7, 10, 30, 37, 39].

However, while we now have a good understanding of the capabilities and limitations of CRNs, we cannot say the same for DSD systems. Most DSD systems [35] use one of two simple mechanisms, three-way toehold exchange or four-way strand exchange, and the range of alternative mechanisms that could be used when designing DSD systems that simulate CRNs or other molecular programs is not well understood. We believe a more formal understanding of DSD would aid in the design process, exploring more – and possibly more efficient – ways of getting things done with DSD systems. Some work has been done toward this goal [4, 21, 23, 26, 29]. In the process of trying to improve formal understanding, we discovered a new mechanism in an unexplored area of the DSD design space. In this paper we present this new mechanism, show how it can be used to implement arbitrary CRNs, and discuss its further implications for understanding DSD.

We describe further background information, then we discuss our contribution, a new DSD mechanism we call *controlled reconfiguration* and its subclass *coupled reconfiguration*. This new mechanism has a number of properties fundamentally different from typical existing DSD mechanisms, including the use of single-stranded signals and fuels, and a stronger than usual dependence on the assumption of pseudoknot-freeness. We discuss the implications and challenges arising from those, including the ability of coupled reconfiguration to implement arbitrary CRNs.

1.2 Background

Our work revolves around DNA Strand Displacement (DSD): certain behaviors of DNA strands that have been used to build useful molecular devices, including logic circuits and neural networks [11, 31], DNA walkers on surfaces [43, 45], and many other things [35]. In one particularly powerful use, DSD can implement arbitrary Chemical Reaction Networks [7, 10, 30, 37, 39]. Figure 1 gives an example of a DSD implementation of a single reaction; details of the implementation are described further in Section 2. As in this example, DSD systems often rely on *fuels*, complexes of multiple DNA strands in a specific configuration which have to be made by an external process.

The Chemical Reaction Network (CRN) model is an abstract description of chemical dynamics that is widely used as a language for programming molecules. We focus on stochastic CRNs, which describe how counts of molecular species evolve when molecules react in a well-mixed solution. Stochastic CRNs are closely related to population protocols, a programming model in distributed systems [1, 2]. Simple CRNs (or population protocols) can compute approximate majority [3], simulate a 3-peak oscillator [39], or simulate boolean circuits [28]. In fact, CRNs can simulate Turing machines, albeit with a small probability of error [36]. However, the class of functions computable by CRNs without error is quite limited, being just the semilinear functions [1, 2, 9, 17, 27, 33]. Consequently, polymer systems [8, 24, 25, 30, 40] and surface CRNs [12, 32] have been proposed as more powerful molecular programming languages, capable of simulating Turing machines without error.

Importantly, arbitrary CRNs can, in theory, be "compiled" into physically realizable DSD systems [37], and in fact all of the CRN models listed above have DSD implementations [12, 30, 32]. These compilation schemes provide an abstraction hierarchy for molecular

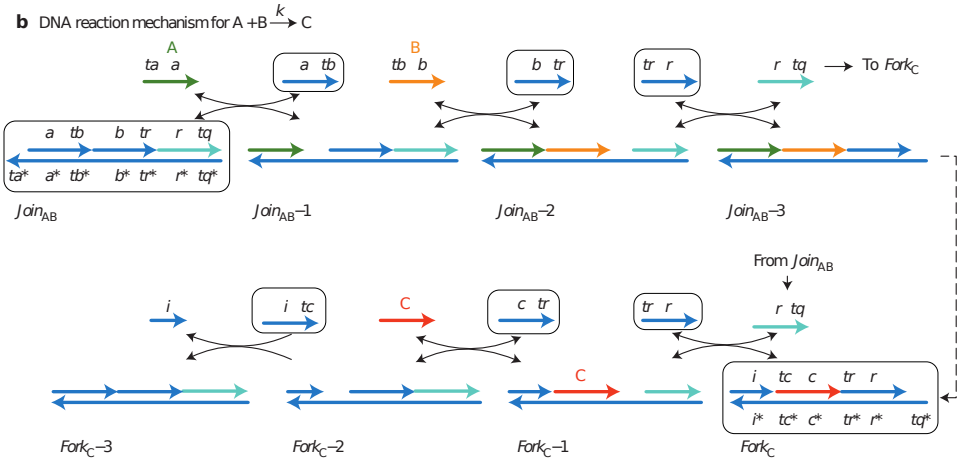

Figure 1 DSD implementation of a CRN reaction $A + B \to C$, adapted from Chen et al. [10]. The implementation consumes various *fuel* strands and complexes (in rounded boxes) to convert *signal* strands (labeled) A and B into the signal strand C. The implementation is a sequence of *primitive reactions*, as listed in Figure 2. For example, at the top left, the fuel $Join_{AB}$ reacts with the signal A, first by binding of toehold domains ta and ta^* (primitive reaction b), followed by 3-way strand displacement (primitive reaction m_3), producing complex $Join_{AB}$-1 and the strand with long domain a followed by toehold tb. In a more general system with multiple reactions, the fuels (and initial signals) must all be produced by an external source.

programming, analogous to compiling high-level languages into assembly and machine languages in traditional programming. CRNs are at the top level of the hierarchy, experimental DNA systems at the bottom, and formal DNA systems in the middle.

While the powerful models above have DSD implementations, we don't yet fully understand the capabilities and limitations of DSD systems. For example, consider the difficulties in building large DSD systems. So far, experimentally demonstrated DSD systems are usually small. CRN implementations, for example, have been demonstrated for CRNs of 3-4 species and reactions [10, 39]. Logic circuits and winner-take-all circuits specifically designed to scale up to larger systems have reached 6-gate circuits [31] and 100-input winner-take-all circuits [11]. Srivinas et al. have examined the effect of "leak" on DSD systems, which would get larger as systems try to scale up [39]. In particular, more complex fuels are less reliable. The previous examples of scaled-up circuits minimize error by using fuels of at most 2 strands [11, 31, 44]. "Leakless" gates have been proposed that would minimize this problem [42, 48], but this has not yet led to reliable large-scale CRN implementations. For this reason we are interested in understanding the capabilities of DSD better, to determine whether even simpler fuels can be used to do more complex tasks such as CRN implementations. We have some previous work exploring ways to formally implement CRNs with 2-stranded fuels [21, 23], but this also has not yet led to experimentally demonstrated large-scale CRN implementations. Existing DSD systems based on single-stranded fuels, such as the hybridization chain reaction [16] or assembly of multi-armed junctions [50], aggregate strands into large, multi-stranded complexes that mostly don't come apart, and so are not suitable for CRN simulation. To explore new DSD mechanisms that could be used for simulation of CRNs and other molecular programming models, we turn to formal DSDs.

Two examples of formal DSD models are Visual DSD [26, 29] and Peppercorn [4, 5]. These models ignore some aspects of experimental DNA systems, such as the actual sequence of bases in a DNA strand, and instead model strands as sequences of *domains*. The models

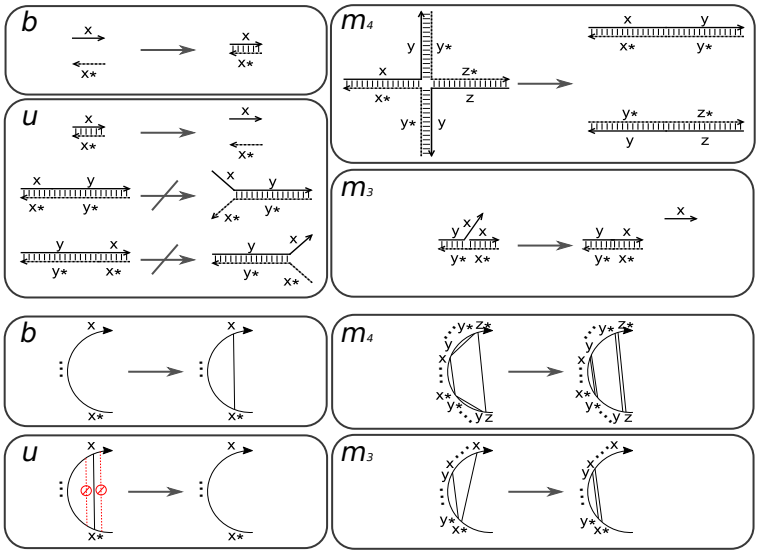

Figure 2 The four primitive reactions, or basic steps, of our DSD model, similar to those used by Petersen et al. [29] and our previous work [21]. Top half: the typical diagram format used in DSD, image adapted from our previous work [21]. The four primitive reactions shown are: domain binding (b), toehold (i.e., short domain) unbinding (u), 4-way branch migration of long domains (m_4) and three-way branch migration of long domains (m_3). Bottom: the same reactions in the circle diagram format we will use throughout this work. Circles indicate strands, with domains labeled; arrows indicate 3' ends; lines or arcs between domains indicate bonds. The crossed-out dashed bonds in the u reaction indicate that the reaction only happens if no such bonds exist. Regions represented by ... may or may not include strand breaks.

also specify how DNA complexes are formed and modified via primitive reactions, or basic steps that involve binding and/or unbinding of domains. The different models broadly agree on the four primitive reactions, shown in Figure 2, but differ on some details. Given an initial set of complexes, the set of DNA complexes that can be enumerated from the initial set using the four primitive reactions, plus the associated reactions, comprise the species and reactions of a CRN which is referred to as the enumerated CRN.

This also allows formal verification that a DSD implementation of an original CRN is correct by comparing the enumerated and original CRNs. Definitions of "correct" implementation include pathway decomposition [34] and CRN bisimulation [22] as well as some others, and the whole process can be automated by the Nuskell compiler [5]. But formalization also allows abstract reasoning about the model, for example, our previous work proved that no DSD system can implement CRNs with a particular, very restrictive, set of properties [21].

Formal models define the four basic steps, but most work with DSD is done on the level of what Peppercorn calls *condensed reactions* [4, 21], short sequences of basic steps that "usually" occur as a unit. In fact, most DSD systems [35] use a variant of the same, simple condensed reaction, toehold-mediated 3-way branch migration, with a few exceptions. But while the set of basic steps is defined by the model, the space of possible condensed reactions is much less limited and not well explored. New condensed reactions might allow DSD systems with simpler fuels that are less prone to leak, or we might find a proof that the existing mechanisms are the simplest possible methods for doing complex tasks.

1.3 Our contributions

In this work, we describe a fundamentally new class of condensed reactions, which we call *controlled reconfiguration*, and a subclass we call *coupled reconfiguration*. This controlled reconfiguration mechanism relies on assumptions made in the formal model that haven't been tested experimentally, so whether it will work is uncertain. However, if the mechanism works as intended, it would enable a fully general CRN implementation using only single-stranded complexes as fuels, as opposed to the multi-stranded complexes of existing methods.

In Section 2, we define the CRN and DSD models that we use throughout the paper. In Section 3 we show the *controlled reconfiguration* and *coupled reconfiguration* mechanisms that are the subject of this paper. In Section 4 we show how coupled reconfiguration can be used to implement arbitrary CRNs, and give the sketch of a proof that the implementation is formally correct according to CRN bisimulation [22]. In Section 5 we discuss the ways in which controlled reconfiguration relies on the no-pseudoknots assumption of the model, and what can be learned from it even if that assumption turns out not to hold for physical DNA. Finally, Section 6 contains further discussion and conclusions.

2 Chemical Reaction Networks and DNA Strand Displacement Systems

We first describe Chemical Reaction Networks (CRNs), and then describe our formalization of DNA Strand Displacement (DSD) systems, which is a particular type of Chemical Reaction Network. We conclude by comparing our DSD system formalization with other variants in the literature.

Chemical Reaction Networks describe how states of a system of molecules change as a result of reactions. Let $\mathcal{S}$ be a set of molecular *species*. A *state* $\mathbf{s}$ is a multiset of species; we'll represent such multisets as functions $\mathbf{s} : \mathcal{S} \to \mathbb{N}$ mapping species to nonnegative integers, so $\mathbf{s}(\sigma)$ is the number of copies of σ in multiset $\mathbf{s}$. We also represent a multiset, such as $\mathbf{s}$, using summation notation: $\sum_{\sigma:\mathbf{s}(\sigma)>0} \mathbf{s}(\sigma)\sigma$. We use $\emptyset$ to denote the empty multiset. A *reaction* is a tuple $(\mathbf{r}, \mathbf{p}, k)$, where the reactants $\mathbf{r}$ and products $\mathbf{p}$ are multisets of species and k is a positive real number, called the reaction rate constant. We write reaction $(\mathbf{r}, \mathbf{p}, k)$ as

$$\sum_{\sigma:\mathbf{r}(\sigma)>0} \mathbf{r}(\sigma)\sigma \xrightarrow{k} \sum_{\sigma:\mathbf{p}(\sigma)>0} \mathbf{p}(\sigma)\sigma.$$

The reaction is *unimolecular* if $\mathbf{r}$ has size 1, and *bimolecular* if $\mathbf{r}$ has size 2 (where n copies of a single species σ counts as size n). Sometimes we omit the rate constant, when it is not pertinent to the constructions presented. For example, if $\mathcal{S} = \{A, B, C\}$, $\mathbf{r}$ is the multiset containing two A's and one C, and $\mathbf{p}$ is the multiset containing one B, then we can write the reaction $(\mathbf{r}, \mathbf{p})$ (ignoring the rate constant) as

$$2A + C \to B.$$

Formally, a *Chemical Reaction Network* (CRN) is a pair $(\mathcal{S}, \mathcal{R})$, where $\mathcal{S}$ is a finite set of species and $\mathcal{R}$ is a finite set of reactions involving the species in $\mathcal{S}$. Reaction $(\mathbf{r}, \mathbf{p})$ is *applicable* to state $\mathbf{s}$ if $\mathbf{s}(\sigma) \geq \mathbf{r}(\sigma)$ for all $\sigma \in \mathcal{S}$, and the result of the reaction is the state $\mathbf{s} - \mathbf{r} + \mathbf{p}$. If both $(\mathbf{r}, \mathbf{p})$ and $(\mathbf{p}, \mathbf{r})$ are in $\mathcal{R}$, we say that the reaction $(\mathbf{r}, \mathbf{p})$ is *reversible* and we write

$$\sum_{\sigma:\mathbf{r}(\sigma)>0} \mathbf{r}(\sigma)\sigma \rightleftharpoons \sum_{\sigma:\mathbf{p}(\sigma)>0} \mathbf{p}(\sigma)\sigma.$$

Our formalization of DNA Strand Displacement (DSD) systems is similar to several variants in the literature [5, 21, 26, 29, 38] and we note below where our model diverges from others. A *DSD system* is specified as a finite multiset of *complexes*, and a set of four *DSD primitive reactions*, or basic steps.

- A complex is a sequence of *strands* that are connected by *bonds*. Each strand is a sequence of *domains*, where each domain has an identity, or label, taken from some finite set. For each domain identity x in the set, there is also a complement, x^* with $(x^*)^* = x$, and bonds of a complex are always between domains with complementary identities. Domains can be *long* or *short*, and we refer to short domains as *toeholds*. Two consecutive domains on the same strand are *adjacent*. See Figure 2 for a visual representation of a complex: strands are arranged along a half-circle, with the sequence of strand domains in clockwise order, and an arc connects each pair of bound (complementary) domains. Complexes must be *pseudoknot free*: there is some ordering of strands along the circumference of a circle in which no bonds cross. (This ordering is in fact unique, see [14]).
- Primitive reactions, or basic steps, transform complexes while maintaining pseudoknot-freeness, and are of four types (also shown in Figure 2):
  - *Binding* (b): A bond is added between a pair of complementary domains. The domains may be in the same complex or in two different complexes; in the latter case the bond forms one new complex from the two participating complexes.
  - *Toehold unbinding* (u): A bond between a toehold t and its complement t^* is removed. This rule can only be applied if there is no bond between a pair of domains d and d^* that are adjacent to t and t^* respectively. Toehold unbinding applied to a complex produces either one or two complexes.
  - *3-way branch migration* (m_3): A bound long domain switches partners. That is, a complex in which long domains x_1 and x_2 are bound, and domain x_3 with same label as x_1 is unbound, is transformed so that x_2 and x_3 are bound and x_1 is unbound. This rule can be applied only when there is a bond between (complementary) domains respectively adjacent to x_2 and x_3.
  - *4-way branch migration* (m_4). Two pairs of bound long domains swap partners. That is, a complex in which domain x_1 is bound to x_2 and domain x_3 is bound to x_4, where x_1 and x_3 have the same domain label (and so x_2 and x_4 have the complementary domain label) is transformed so that x_1 is bound to x_4, and x_2 is bound to x_3. This rule can be applied only when there is a bond between domains adjacent to x_1 and x_4, plus a bond between domains adjacent to x_2 and x_3.
- The following sequences of steps (sometimes called *motifs* [23]) appear often, acting as a unit, in our mechanisms and many others:
  - *Toehold-mediated 3-way strand exchange* (tx_3). One pair of toeholds binds (b). The two toeholds were adjacent to a pair of bound domains x, x^* and an unbound domain x, respectively, which are now able to branch migrate (m_3). The previous bond was adjacent to a pair of toeholds on the other side and preventing them from unbinding, but now they are able to do so (u).
  - *Two-toehold-mediated 4-way strand exchange* (tx_4). Two pairs of toeholds bind in either order, in such a way that a four-way junction is formed (b, b). The four-way junction then branch migrates (m_4). The previous bonds in the four-way junction were each adjacent to a pair of toeholds on the opposite side, which are now capable of unbinding in either order (u, u).

(Both of those motifs are reversible as long as the initial toehold binding(s) is/are reversible, and in fact the reverse of each one is an instance of the same motif. Concrete examples of the motifs occur in Figures 3 and 4.)

The four basic steps are often grouped into *condensed reactions*, sequences of one bimolecular step followed by as many unimolecular steps as necessary until no further meaningful unimolecular steps are possible [4, 21]. The tx_3 and tx_4 motifs, for example, are often used as entire condensed reactions [23, 35], but can also be used as parts of larger condensed reactions. We don't use the details of this definition much, but the mechanisms we present in Section 3 are in fact condensed reactions.

Let C be a set of complexes. The *DSD system enumerated from* C, using the set of reaction rules $\{b, u, m_3, m_4\}$, is the smallest CRN $(\mathcal{S}, \mathcal{R})$ such that any complex in C is a species in $\mathcal{S}$, any application of DSD reaction rules b, u, m_3 or m_4 to reactants in $\mathcal{S}$ is a reaction in $\mathcal{R}$, and the products of each such reaction are species in $\mathcal{S}$.

We end this section by summarizing differences between our DSD system formalization and other formalizations in the literature. A key point is that our definition enforces pseudoknot-freeness: no reactions may introduce pseudoknots. The DSD systems of Badelt et al. [5], Johnson [21], and Petersen et al. [29] also enforce pseudoknot-freeness, but those of Lakin et al. [26] and Spaccasassi et al. [38] allow pseudoknotted complexes. We emphasize that our constructions in the following sections would *not* be correct in a model that includes pseudoknots. We discuss the implications of our pseudoknot-freeness constraint further in Section 5.

Another difference is that in some models [21, 26], toeholds can mediate 3-way strand displacement even when not adjacent to the long domains that bind; rather the toeholds can be "adjacent" to the reaction in a weaker sense than we use it here [29], or in fact be "remote" [20, 21]. We also do not include yet other DNA strand displacement reactions that have been used in experimental work, such as cooperative strand displacement [51]. None of these additions to the model are needed for our purposes. In contrast with pseudoknots, we believe that our correctness arguments can be adapted even if the model is generalized in these ways.

3 Controlled reconfiguration and coupled reconfiguration

We present a class of mechanisms that we call *controlled reconfiguration*, and a subclass we call *coupled reconfiguration*. The mechanisms work in the formal DSD model as defined in Section 2, but have not yet been tested experimentally. In particular, these methods depend on the assumption that domains cannot bind if it would create a pseudoknot, which is one of the more questionable assumptions. Whether this mechanism describes the behavior of real DNA strands, or whether it is better interpreted as an exploration of the formal model, is discussed properly in Section 5. All further discussion until then is assuming everything functions as described in the model.

Figure 3 shows an example of a *controlled reconfiguration* mechanism. The long strand, as a single copy by itself, has two stable configurations, and cannot switch between them; all the free X^* and Y^* domains that could start the reconfiguration are themselves blocked off by existing bonds, and the interaction necessary to do so would be a pseudoknot. The short strand displaces the bond between Z domains, triggering a sequence of steps each involving a domain "freed" by the previous displacement. Eventually, the same Z bond is re-formed and the short strand is released, with the long strand having reconfigured. Intuitively, the bonds function as locked doors, preventing what's behind them from interacting with what's outside, until unlocked with a key hidden behind the previous door.

The controlled reconfiguration mechanism is an example, and proof-of-concept, of how one strand can control the reconfiguration of another. However, in the mechanism in Figure 3, the "control" strand only allows the reconfiguring strand to freely switch between its two

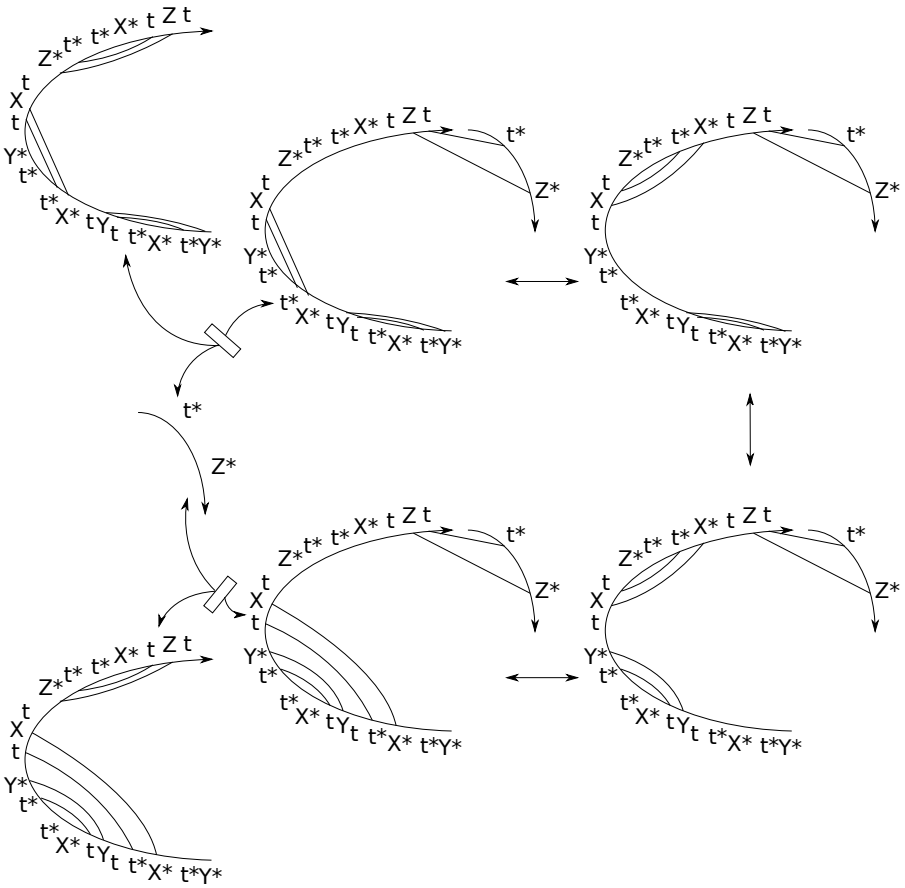

Figure 3 A simple controlled reconfiguration mechanism. t is a short domain, X, Y, and Z are long. Each step in the figure is a tx_3 motif. The net effect is that the long strand changes from the configuration in the top left to the configuration in the bottom left, using the short strand as a catalyst, while the long strand could not change configuration on its own.

configurations. Neither the control strand nor any other part of the system knows which configuration the reconfiguring strand is in, so it can't, for example, force the strand into a desired configuration and then proceed to the next task. This limits its usability as a module in larger systems. For that, we would want a mechanism where two different strands simultaneously serve as the control for each other's reconfiguration, in such a way that either both can change or neither can change, but not just one or the other. Figure 4 shows the *coupled reconfiguration* mechanism, which we will use for our formal CRN implementation.

In the coupled reconfiguration mechanism, each strand has a pair of A_1, A_1^* domains and a pair of A_2, A_2^* domains, but crossed over such that (excluding pseudoknots) only one pair can be bound at once, and it cannot reconfigure on its own. Further, there is a pair of G ("guard") domains that, by their bond, prevent random other strands from interacting with the domains inside the strand. The "left" and "right" strands are asymmetric in the toehold identities, and one left and one right strand can come together and exchange G bonds by a tx_4 motif. This opens each strand up to interacting with the other. (Two left strands or two right strands do not have complementary external toeholds, so such pairs cannot interact.) Then, if one of the strands was in the A_1 configuration and the other in the A_2 configuration, they can go through a process of exchanging bonds that eventually leads to both strands

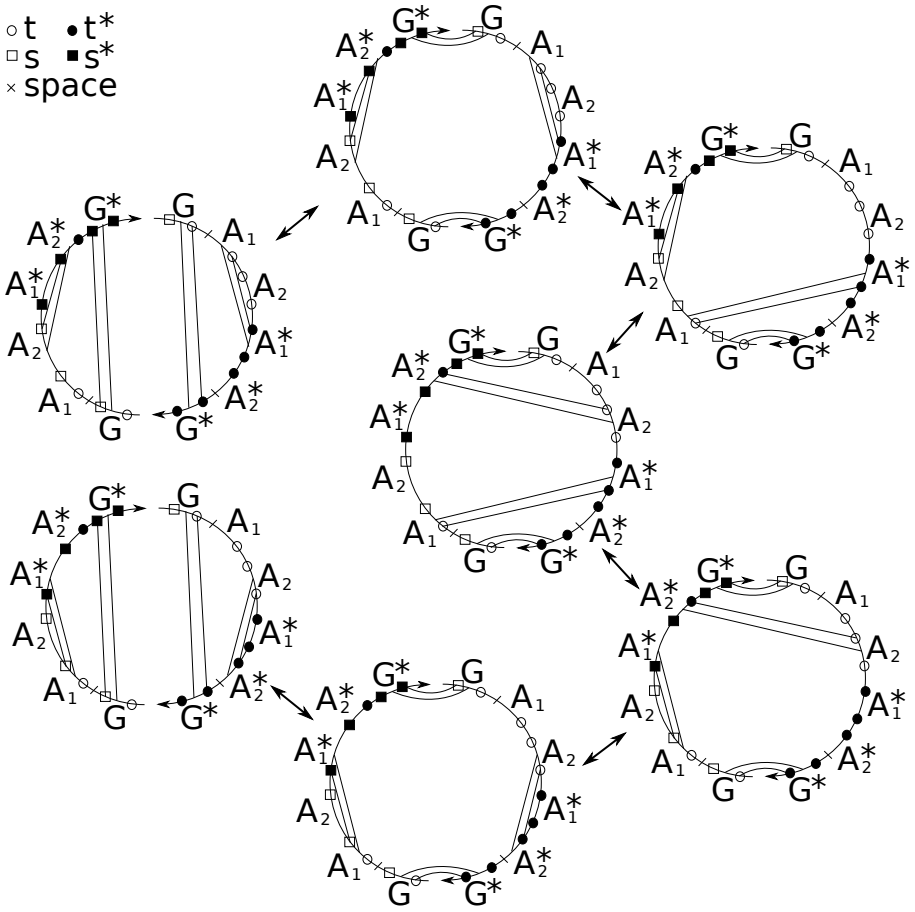

Figure 4 The coupled reconfiguration mechanism. Long domains are given by name while toeholds are represented by symbols. Space domains don't interact with anything, but prevent some spurious interactions. A left and a right strand can join together to reconfigure via a tx_4 motif on G domains, and separate when the reconfiguration is done by reversing the tx_4 motif. Then a sequence of tx_3 motifs lets each strand enable the reconfiguration of the other, one strand moving from the A_1 configuration to the A_2 configuration and the other doing the opposite. The mechanism is fully reversible, but the strands can only separate if either both strands have changed configuration or neither has.

having swapped configurations. The G bonds can then reverse the tx_4 motif to separate again, but this can only happen when either both have reconfigured or neither has. Any left and right strand in the same configuration can join together, but no further reaction can happen except for separating unchanged.

The reaction as shown in Figure 4 is fully reversible, but can be made irreversible by adding any of the following four toeholds: a t^* after the A_1^* on the left strand or after the A_1^* on the right strand makes the reaction only go forward (clockwise around the diagram shown), while a t before the A_2 on the left strand or before the A_1 on the right strand makes the reaction only go backward. In either case, the strand the toehold is added to becomes unreactive after the reaction, while the other strand is still capable of further coupled reconfigurations (with counterpart strands that don't have the added toehold).

The coupled reconfiguration reaction can be abstracted as $A_{2,l} + A_{1,r} \rightleftharpoons A_{1,l} + A_{2,r}$. Since each configuration has a different bond, there are regions of each strand that are covered in one configuration and uncovered in the other, and different (left and/or right) strands can share the A_1, A_2 domain identities while having different content in the conditionally covered regions. This mechanism turns out to be powerful enough to implement arbitrary CRNs.

We think of "controlled reconfiguration" more generally as any DSD reaction where two or more complexes interact and separate, where (at least) one of the complexes at the end is made up of the same strands in a different configuration, such that the reconfiguration could not have happened in that complex on its own. The example in Figure 3 is in particular "catalytic controlled reconfiguration", where the complex that controls the reconfiguration is itself unchanged. Then "coupled reconfiguration" more generally is where two or more complexes interact and separate, where (at least) two of the complexes at the end have reconfigured in a way that each one could not on its own, and further such that the reaction could have reconfigured both complexes or neither, but not only one of them. Then the mechanisms we presented were examples of controlled and coupled reconfiguration, but other examples exist. In particular, Zhang and Winfree have presented a mechanism that meets this definition of controlled reconfiguration, but is neither catalytic nor coupled [52].

4 Implementing CRNs with coupled reconfiguration

The coupled reconfiguration mechanism enables different strands to, in a certain intuitive sense, pass information to each other by reconfiguring. This is sufficient to implement, among other things, arbitrary CRNs. In Section 4.1 we discuss how to combine coupled reconfiguration reactions to implement CRNs. In Section 4.2 we give an argument for formal correctness of this method according to CRN bisimulation [22].

4.1 The mechanism of a reaction

We first consider the reaction $A + B \rightleftharpoons C$, which is sufficient to implement arbitrary reversible CRNs (and arbitrary CRNs, if both directions are available as irreversible reactions). We use what in Section 3 we referred to as "right" strands as *species* strands that represent the abstract species, with species A getting a unique pair of long domains A_1 and A_2, and corresponding to right strands with the (toeholds omitted) sequence $G\ A_1\ A_2\ A_1^*\ A_2^*\ G^*$. Such a strand in the configuration with A_1 domains bound to each other represents the species A, and we call the strand in that configuration the *signal* strand A_{on}. The same strand in the A_2 configuration is called A_{off}, and represents nothing. Species B and C get unique domains B_1, B_2, C_1, and C_2, according to the same pattern.

To implement the reaction we use what in Section 3 we called "left" strands as *gate* strands. Where r refers to the reaction $A + B \rightleftharpoons C$, the gate strand will interleave the domains for A, B, and C in the order shown in Figure 5. To implement the forward reaction, the gate strand starts in configuration r_{for}, with bonds A_2, B_2, and C_1. A coupled reconfiguration reaction with A_{on} produces r_{Afor}, where the B_2 bond is exposed. This then reacts with B_{on} to produce r_{Cback}, covering up the A_1 bond but exposing the C_1 bond. That allows a reaction with C_{off}, producing C_{on} and r_{back}, where the B_2^* and A_2^* domains are covered by the C_2 bond. Thus, the gate consumes a signal A and B and produces a signal C, where each step covering up the previous step and uncovering the next step ensures that they can only be done in that order. The reverse reaction is implemented by the reverse of this process, starting with r_{back} and eventually producing r_{for}.

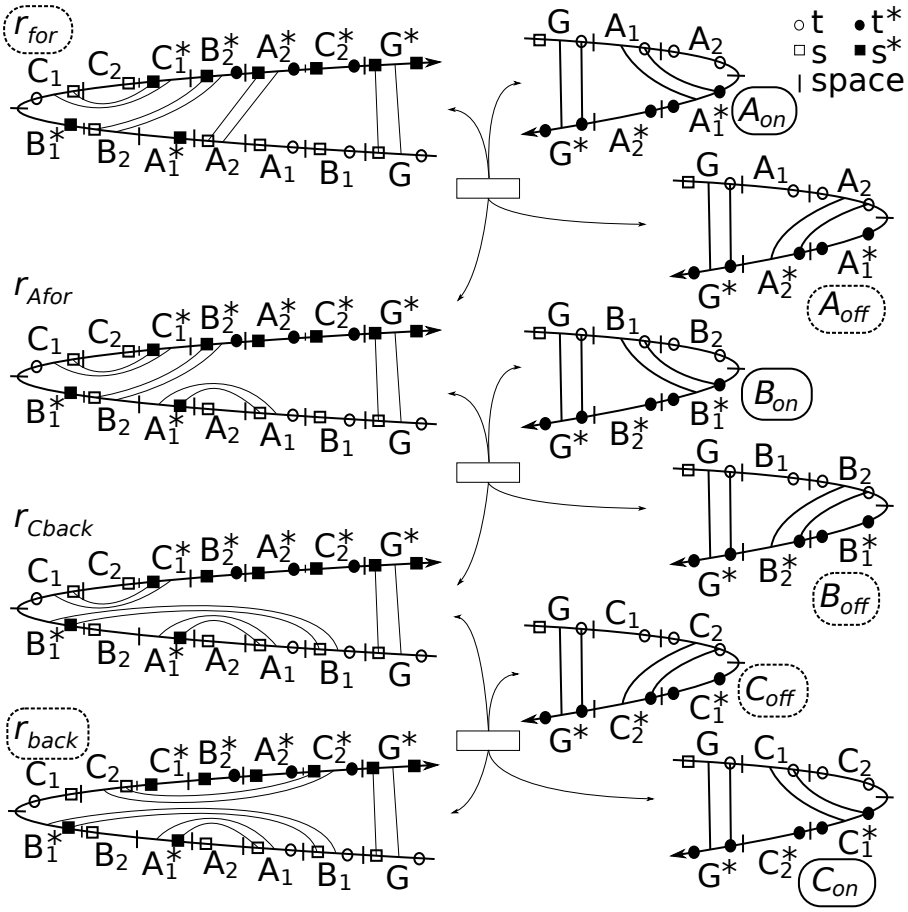

Figure 5 Simulation of the reaction $r = A + B \rightleftharpoons C$. Each reaction represents an instance of the coupled reconfiguration sequence shown in Figure 4. Spacer domains separate each long domain and its flanking toeholds from those of the adjacent domain, to prevent certain unintended interactions. Species are given names with solid boxes for signal strands, dashed boxes for fuel strands, and no boxes for intermediate (other) strands.

4.2 Correctness of the mechanism

To prove our mechanism correct, we use CRN bisimulation [22]. Here, however, we only give a sketch of the process. First, we can describe the DSD system as a CRN, the *implementation CRN*, compared to the *formal CRN* that it allegedly implements. To do that we enumerate the possible interactions between strands in our DSD system, according to the rules in Section 2.

The complexes in the system include some designated as *fuels*, namely r_{for} and r_{back} for each reaction r, and A_{off} for each species A, and some designated as *signals*, namely A_{on} for each species A. Fuels are assumed to be always present in "enough" quantity (and how to achieve that is left as an exercise for the experimenter), while signals represent the corresponding formal species. The interactions between strands fall into the following categories:

- Spurious toehold bindings that don't lead to an m_4 step between G domains.
- "Nuisance" interactions between a gate strand and a species strand that have no intended, and no possible, productive reaction.
- Productive reactions between strands intended to interact.

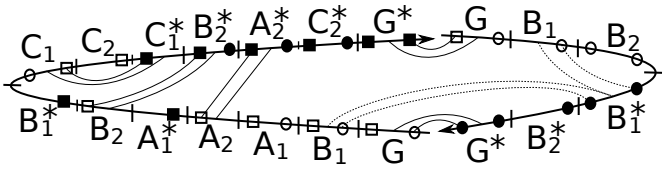

Figure 6 A nuisance interaction between r_{for} and B_{on}. Any gate strand and any signal strand can bind and open up the G domains, but only pairs intended to react can meaningfully reconfigure. In this example, r_{for} and B_{on} can bind, open G domains, and even do toehold exchange (represented by dashed lines) on the B_1 domains, but further interaction with B domains is blocked by the A_2 bond. The only possible continuation is for the reaction to eventually reverse itself and separate back into r_{for} and B_{on} unmodified.

In the first category, arbitrary sequences of b and u steps between strands on external toeholds can happen, but can only make a meaningful change if they eventually form a binding of two toeholds that enables an m_4 step on G domains. Otherwise, the strands involved will all eventually unbind unchanged. This can form arbitrarily large (and increasingly unlikely) chains, which to formally treat requires the techniques of Polymer Reaction Network bisimulation [24]; this can be done, but we don't include it here.

In the second category, any gate strand and any species strand can bind on two toeholds and open the G domains in a tx_4 motif. But they can only do a coupled reconfiguration reaction if the gate strand has the domains corresponding to the species strand in the opposite configuration (e.g. a gate strand with a B_2 bond interacting with a species strand with a B_1 bond) and none of those domains on the gate strand are blocked by any other bond. Any other combination may start, but cannot finish, a coupled reconfiguration reaction, and can only eventually separate into the original unmodified strands. Figure 6 gives an example of such an interaction.

The third category is the reactions shown in Figure 5. Effectively, this means the set of reachable complexes is the set of things shown in that figure; intermediate states between them; and spurious bindings from the first two categories. Modeling the system with the *condensed reaction* model [4, 21] allows us to treat the first and second categories as trivial, abstract away from the intermediates, and describe the implementation CRN for the formal reaction $A + B \rightleftharpoons C$ as follows:

$$\begin{aligned} r_{for} + A_{on} &\rightleftharpoons r_{Afor} + A_{off} \\ r_{Afor} + B_{on} &\rightleftharpoons r_{Cback} + B_{off} \\ r_{Cback} + C_{off} &\rightleftharpoons r_{back} + C_{on} \end{aligned}$$

Since fuels are always present (or always producible), bisimulation and other verification methods often work with an implementation CRN where fuels are removed:

$$\begin{aligned} A_{on} &\rightleftharpoons r_{Afor} \\ r_{Afor} + B_{on} &\rightleftharpoons r_{Cback} \\ r_{Cback} &\rightleftharpoons C_{on} \end{aligned}$$

CRN bisimulation involves an *interpretation* of implementation species as (multisets of) formal species, usually denoted m; here $m(A_{on}) = A$, $m(B_{on}) = B$, $m(C_{on}) = C$, $m(r_{Afor}) = A$, and $m(r_{Cback}) = C$. We see that: (a) any implementation reaction, when interpreted, is either trivial or $A + B \rightleftharpoons C$; (b) any non-signal implementation species (r_{Afor} and r_{Cback})

can turn into the signal species (A_{on}, B_{on}, or C_{on}) for its interpretation; and (c) the signal species can implement any reaction that their corresponding formal species should be able to do. These are the conditions of modular CRN bisimulation [22].

Because of the modularity condition [22], and because our comments on (lack of) crosstalk reactions apply between modules as well as within a module, an arbitrary number of reactions of the form $A + B \rightleftharpoons C$ can be implemented by combining the implementations for each one. Because arbitrary reversible CRNs can be implemented up to CRN bisimulation by reactions of the form $A + B \rightleftharpoons C$ and CRN bisimulation is transitive [22], this mechanism can implement arbitrary reversible reactions. To implement an irreversible CRN, use the method mentioned in Section 3 to make coupled reconfiguration irreversible by adding a toehold to the gate strand: alter $r_{Afor} + B_{on} \rightarrow r_{Cback}$ for $A + B \rightarrow C$ and $C_{on} \rightarrow r'_{Afor} + B_{on}$ for $C \rightarrow A + B$ (where r'_{Afor} is the altered version of r_{Afor} that can interact meaningfully with A_{off} but not B_{on}).

5 Pseudoknots and questioning the model

Whether any mechanism, written on paper, will actually work in the lab, is always a question. But controlled reconfiguration, in particular, depends on an assumption of the DSD model that hasn't been particularly well tested: the assumption that if a step would form a pseudoknot, that step can't happen. So while it's possible that controlled reconfiguration will work experimentally as we presented it, it's reasonably likely that it won't. However, there are reasons why, even if it doesn't work as presented in unmodified DNA, the fact that it exists in the model of DSD is still important for the study and design of DSD systems.

Traditionally the "no-pseudoknots" assumption was not a belief that pseudoknots don't happen, but a belief that we don't want to use them. RNA pseudoknots appear in many biological examples [46], and DNA nanotechnology applications such as tile assembly [18], DNA origami [13], and RNA origami [19] use fundamentally pseudoknotted structures. However, pseudoknotted structures are much less well understood: pseudoknots introduce steric contributions to energy that aren't modeled by the standard, nearest-neighbor models [49], and even with an energy model, predicting pseudoknotted structures is computationally harder [15].

For this reason, most engineered DSD systems [35] are designed such that the intended pathway does not involve pseudoknots, but also there are no non-trivial unintended pathways even when allowing pseudoknots, so that the mechanism works as intended whether or not pseudoknots are possible. In contrast, the coupled reconfiguration pathway has plenty of unintended pathways that would be possible if pseudoknots were possible, and works only under the assumption that they are not.

The first thing this result does is illustrate a difference between the model and how actual DNA behaves, suggesting ways to refine the model. For example, we might want to formally add the condition that the intended pathway has to exclude pseudoknots, but unintended pathways with pseudoknots break the system. CRN bisimulation [22] suggests one way to formalize this: the permissive condition is checked with pseudoknots banned and the delimiting condition is checked with pseudoknots allowed. So this result suggests that it would be worth investigating whether coupled reconfiguration, or single-stranded fuel-based CRNs in general, are possible or provably impossible in such a model.

Another interpretation of our result is that *if* some molecule behaves like it does in this model, *then* controlled reconfiguration can be used with that molecule. Even if unmodified DNA at standard experimental conditions forms these pseudoknots, there may be an artificial

nucleic acid, or unrelated biomolecule with strand displacement-like behavior, that doesn't. If a single-stranded-fuel-based CRN implementation proves to be a particularly useful molecular device, it may be worth trying to find or design such a molecule.

Possibly relevant to this question are the experimental results of Zhang and Winfree's switchable DNA catalyst [52]. There, the catalyst switches between two states based on an input, where a hypothetical pseudoknotted spontaneous switching pathway exists. However, in that example one state is favored over the other with a one-directional non-pseudoknotted spontaneous switching pathway, and can only be held in the less favorable state when an additional strand is bound. So the experimental results may be a useful starting point for investigation, but are not conclusive in any direction on their own.

Finally, treating DSD as a model of computation, we'd like to know both how computationally powerful DSD is, and which features of the model give it which powers. This result shows us that the model with pseudoknots completely disallowed can do coupled reconfiguration, which is powerful enough to implement CRNs with single-stranded fuels. It further suggests that the ability to treat bonds as logical blockers or locked doors, preventing the two sides from interacting, is what gives the model that power.

6 Discussion and conclusions

We have introduced the mechanisms of controlled reconfiguration and coupled reconfiguration, and shown how they can be used to construct single-stranded CRN implementations. We discussed how the questionable assumption of pseudoknot-freeness means that the mechanism might not work as designed, but its existence in the model would likely have interesting implications. If the mechanism does work, either as designed or something similar, it would have a number of challenges as well as a number of benefits for DSD system design in general. Here we discuss some of the larger challenges, some of the larger benefits, and some possible aspects for further investigation.

Internal toeholds

In the model, we've assumed that m_3 and m_4 steps only happen if there are bound domains adjacent to the future reaction, usually toeholds, and that u steps can happen whenever there are no *adjacent* domains holding the toeholds together, even if the unbinding doesn't separate complexes. Both of those are not well explored, and different models have made different assumptions [21, 29]. We note that, first of all, our mechanism doesn't depend on toehold identities to prevent reactions from happening, so it should work in a model where internal m_3 steps don't need immediately adjacent toeholds, such as the one from [21]. Second, the concept of "toehold" in general is a domain of the right length to bind and unbind reversibly, and by doing so, hold things together long enough for m_3 and m_4 steps to happen. When the binding is between otherwise separate complexes, that length is 4-6 bases [53]. Internally, the length may be the same 4-6 bases, or may be more like 2 bases, but whatever length it happens to be, that length is the one we would use for the internal toeholds.

Simulating reaction rate constants

Our constructions in Sections 3 and 4 do not address how to specify rate constants for the reactions of the DSD system that simulates a CRN, so as to ensure consistency with the rate constants of the CRN. Soloveichik et al. [37] explain how this can be done within reasonable

timescales, for their DSD simulation of a CRN, through toehold design and accounting for buffering effects. While the details of their simulation are quite different from ours, the key principles of their approach should also apply to our construction. We note also that many useful CRNs have the nice property of being *rate independent* in the sense that their correctness does not depend on specific reaction rate constants [6, 9, 47].

Fuel synthesis

Our construction uses single-stranded fuels, compared to the double-stranded or larger fuels of typical DSD systems. Synthesis of single strands is typically easier than synthesis of multi-stranded complexes [31]. However, a challenge is to ensure that each fuel strand is in its specified secondary structure, which is one of two or more stable (pseudoknot-free) structures for the strand. It might be possible to take advantage of co-transcriptional folding to ensure that fuel complexes have the correct structures, or it might be possible to filter out strands with incorrect structures by hybridization to a surface via a free long domain.

Single-stranded DSD

Existing single-stranded DSD systems involve single strands interacting with each other and aggregating [16, 50]. In contrast, the typical CRN implementation [37] involves reactions where (usually) one strand interacts with a larger complex, eventually staying bound and releasing (usually) one other strand, where either the new complex or the new strand (or both) is a meaningful signal that goes on to interact with the rest of the system. In particular, in those systems, most complexes are the only "stable" configuration of the strands that make them up. With single-stranded fuels that are meant to stay single-stranded, this clearly isn't possible, otherwise there would be no meaningful reactions. But if the single strands had multiple configurations that they could switch between on their own, that wouldn't implement bimolecular $A + B \rightleftharpoons C$ logic. So the novelty of coupled reconfiguration, and its ability to implement that logic, is why it makes single-stranded non-aggregative DSD systems possible.

Locality and impossibility

In our previous work on impossibility in DSD systems [21], one of the important steps was a locality theorem: that, with the restrictions in that result (notably including forbidding m_3 steps), if a sequence of unimolecular steps breaks a four-way junction, then eventually reforms the same four-way junction with the same domains, then the same result can be gotten by a sequence of steps that never breaks the junction. The coupled reconfiguration mechanism shows by counterexample that without those restrictions, and assuming pseudoknots can't be formed, the same statement would not hold. In our previous work, this theorem showed that with those restrictions, CRNs with 2-stranded fuels were impossible; with the assumptions in this work, coupled reconfiguration makes CRNs with single-stranded fuels possible. Further, to our knowledge controlled reconfiguration is the first DSD mechanism that involves a bond being initially present, broken, then reformed, with an effect that was not possible without breaking that bond. So we suspect that this concept of locality is interesting, and worth investigating whether other such mechanisms exist, especially ones that don't depend on pseudoknots being unable to be formed.

Expanding the DSD toolbox

We've expanded the DSD toolbox by introducing controlled reconfiguration and coupled reconfiguration, and shown its application to reversible DSD simulation of reversible CRNs, in a way that is easily adapted also to non-reversible CRNs. Our implementation scheme also has the following desirable properties defined in our previous work [21]: systematic with $O(1)$ toeholds, physically reversible, and using only bimolecular condensed reactions, and we have already shown that it is systematically correct under modular CRN bisimulation and uses single-stranded fuels. There are a number of CRN implementations that each fail a different one of the desirable conditions [23], and the coupled reconfiguration-based implementation fails a condition which was not listed but is also desirable, the condition of not relying on the assumption that pseudoknots don't happen. So it would be interesting to investigate whether there is a variant of coupled reconfiguration, or another implementation scheme, that fulfills the same conditions without relying on that assumption.

Further, the coupled reconfiguration mechanism reveals a capability of the current model of DSD that physical DNA strands may or may not have, either in the mechanism as we presented it or in a different mechanism with similar properties. If it does, then we would like to see if the advantages of the scheme enable physical DSD implementations with less error than existing schemes. Whether it does or not, it would be interesting to investigate the importance of the assumption that pseudoknots don't happen, and how it affects the power of DSD systems.

References

1 Dana Angluin, James Aspnes, Zoë Diamadi, Michael J. Fischer, and René Peralta. Computation in networks of passively mobile finite-state sensors. *Distributed Computing*, pages 235–253, March 2006. doi:10.1007/s00446-005-0138-3.

2 Dana Angluin, James Aspnes, and David Eisenstat. Stably computable predicates are semilinear. In *Proceedings of the twenty-fifth annual ACM symposium on Principles of distributed computing*, pages 292–299. ACM, 2006. doi:10.1145/1146381.1146425.

3 Dana Angluin, James Aspnes, and David Eisenstat. A simple population protocol for fast robust approximate majority. *Distributed Computing*, 21:87–102, 2008. doi:10.1007/s00446-008-0059-z.

4 Stefan Badelt, Casey Grun, Karthik V Sarma, Brian Wolfe, Seung Woo Shin, and Erik Winfree. A domain-level DNA strand displacement reaction enumerator allowing arbitrary non-pseudoknotted secondary structures. *Journal of the Royal Society Interface*, 17(167):20190866, 2020. doi:10.1098/rsif.2019.0866.

5 Stefan Badelt, Seung Woo Shin, Robert F Johnson, Qing Dong, Chris Thachuk, and Erik Winfree. A general-purpose CRN-to-DSD compiler with formal verification, optimization, and simulation capabilities. In Damien Woods and Yannick Rondelez, editors, *DNA Computing and Molecular Programming*, volume 9818 of Lecture Notes in Computer Science, pages 232–248. Springer, 2017. doi:10.1007/978-3-319-66799-7_15.

6 Daniele Cappelletti, Andrés Ortiz-Muñoz, David F. Anderson, and Erik Winfree. Stochastic chemical reaction networks for robustly approximating arbitrary probability distributions. *Theoretical Computer Science*, 801:64–95, 2020. doi:10.1016/j.tcs.2019.08.013.

7 Luca Cardelli. Two-domain DNA strand displacement. *Mathematical Structures in Computer Science*, 23(02):247–271, 2013. doi:10.1017/S0960129512000102.

8 Luca Cardelli and Gianluigi Zavattaro. On the computational power of biochemistry. In *International Conference on Algebraic Biology*, pages 65–80. Springer, 2008. doi:10.1007/978-3-540-85101-1_6.

9 Ho-Lin Chen, David Doty, and David Soloveichik. Deterministic function computation with chemical reaction networks. *Natural Computing*, 13(4):517–534, 2014. doi:10.1007/s11047-013-9393-6.

10 Yuan-Jyue Chen, Neil Dalchau, Niranjan Srinivas, Andrew Phillips, Luca Cardelli, David Soloveichik, and Georg Seelig. Programmable chemical controllers made from DNA. *Nature Nanotechnology*, 8(10):755–762, 2013. doi:10.1038/nnano.2013.189.

11 Kevin M Cherry and Lulu Qian. Scaling up molecular pattern recognition with DNA-based winner-take-all neural networks. *Nature*, 559(7714):370, 2018. doi:10.1038/s41586-018-0289-6.

12 Samuel Clamons, Lulu Qian, and Erik Winfree. Programming and simulating chemical reaction networks on a surface. *Journal of the Royal Society Interface*, 17(166):20190790, 2020. doi:10.1098/rsif.2019.0790.

13 Swarup Dey, Chunhai Fan, Kurt V Gothelf, Jiang Li, Chenxiang Lin, Longfei Liu, Na Liu, Minke AD Nijenhuis, Barbara Saccà, Friedrich C Simmel, Hao Yan, and Pengfei Zhan. DNA origami. *Nature Reviews Methods Primers*, 1(1):1–24, 2021. doi:10.1038/s43586-020-00009-8.

14 Robert M Dirks, Justin S Bois, Joseph M Schaeffer, Erik Winfree, and Niles A Pierce. Thermodynamic analysis of interacting nucleic acid strands. *SIAM review*, 49(1):65–88, 2007. doi:10.1137/060651100.

15 Robert M Dirks and Niles A Pierce. A partition function algorithm for nucleic acid secondary structure including pseudoknots. *Journal of computational chemistry*, 24(13):1664–1677, 2003. doi:10.1002/jcc.10296.

16 Robert M Dirks and Niles A Pierce. Triggered amplification by hybridization chain reaction. *Proceedings of the National Academy of Sciences*, 101(43):15275–15278, 2004. doi:10.1073/pnas.0407024101.

17 David Doty and Monir Hajiaghayi. Leaderless deterministic chemical reaction networks. In David Soloveichik and Bernard Yurke, editors, *DNA 2013: Proceedings of The 19th International Meeting on DNA Computing and Molecular Programming*, volume 8141 of *Lecture Notes in Computer Science*, pages 46–60. Springer International Publishing, 2013. doi:10.1007/978-3-319-01928-4_4.

18 Constantine G Evans, Rizal F Hariadi, and Erik Winfree. Direct atomic force microscopy observation of DNA tile crystal growth at the single-molecule level. *Journal of the American Chemical Society*, 134(25):10485–10492, 2012. doi:10.1021/ja301026z.

19 Cody Geary, Guido Grossi, Ewan KS McRae, Paul WK Rothemund, and Ebbe S Andersen. RNA origami design tools enable cotranscriptional folding of kilobase-sized nanoscaffolds. *Nature chemistry*, 13(6):549–558, 2021. doi:10.1038/s41557-021-00679-1.

20 Anthony J Genot, David Yu Zhang, Jonathan Bath, and Andrew J Turberfield. Remote toehold: a mechanism for flexible control of DNA hybridization kinetics. *Journal of the American Chemical Society*, 133(7):2177–2182, 2011. doi:10.1021/ja1073239.

21 Robert F. Johnson. Impossibility of sufficiently simple chemical reaction network implementations in DNA strand displacement. In Ian McQuillan and Shinnosuke Seki, editors, *Unconventional Computation and Natural Computation*, pages 136–149. Springer International Publishing, 2019. doi:10.1007/978-3-030-19311-9_12.

22 Robert F Johnson, Qing Dong, and Erik Winfree. Verifying chemical reaction network implementations: A bisimulation approach. *Theoretical Computer Science*, 2018. doi:10.1016/j.tcs.2018.01.002.

23 Robert F. Johnson and Lulu Qian. Simplifying Chemical Reaction Network Implementations with Two-Stranded DNA Building Blocks. In Cody Geary and Matthew J. Patitz, editors, *26th International Conference on DNA Computing and Molecular Programming (DNA 26)*, volume 174 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 2:1–2:14, Dagstuhl, Germany, 2020. Schloss Dagstuhl–Leibniz-Zentrum für Informatik. doi:10.4230/LIPIcs.DNA.2020.2.

24 Robert F Johnson and Erik Winfree. Verifying polymer reaction networks using bisimulation. *Theoretical Computer Science*, 843:84–114, 2020. doi:10.1016/j.tcs.2020.08.007.

25 Matthew R. Lakin and Andrew Phillips. Modelling, simulating and verifying Turing-powerful strand displacement systems. In Luca Cardelli and William Shih, editors, *DNA Computing and Molecular Programming*, pages 130–144, Berlin, Heidelberg, 2011. Springer Berlin Heidelberg. doi:10.1007/978-3-642-23638-9_12.

26 Matthew R. Lakin, Simon Youssef, Filippo Polo, Stephen Emmott, and Andrew Phillips. Visual DSD: a design and analysis tool for DNA strand displacement systems. *Bioinformatics*, 27(22):3211–3213, 2011. doi:10.1093/bioinformatics/btr543.

27 Jérôme Leroux. Vector addition systems reachability problem (a simpler solution). In *EPiC*, volume 10, pages 214–228. Andrei Voronkov, 2012.

28 Marcelo O Magnasco. Chemical kinetics is Turing universal. *Physical Review Letters*, 78(6):1190–1193, 1997. doi:10.1103/PhysRevLett.78.1190.

29 Rasmus L Petersen, Matthew R Lakin, and Andrew Phillips. A strand graph semantics for DNA-based computation. *Theoretical computer science*, 632:43–73, 2016. doi:10.1016/j.tcs.2015.07.041.

30 Lulu Qian, David Soloveichik, and Erik Winfree. Efficient Turing-universal computation with DNA polymers. In Yasubumi Sakakibara and Yongli Mi, editors, *DNA Computing and Molecular Programming*, volume 6518 of Lecture Notes in Computer Science, pages 123–140. Springer, 2011. doi:10.1007/978-3-642-18305-8_12.

31 Lulu Qian and Erik Winfree. Scaling up digital circuit computation with DNA strand displacement cascades. *Science*, 332(6034):1196–1201, 2011. doi:10.1126/science.1200520.

32 Lulu Qian and Erik Winfree. Parallel and scalable computation and spatial dynamics with DNA-based chemical reaction networks on a surface. In Satoshi Murata and Satoshi Kobayashi, editors, *DNA Computing and Molecular Programming*, volume 8727 of Lecture Notes in Computer Science, pages 114–131. Springer, 2014. doi:10.1007/978-3-319-11295-4_8.

33 Charles Rackoff. The covering and boundedness problems for vector addition systems. *Theoretical Computer Science*, 6(2):223–231, 1978. doi:10.1016/0304-3975(78)90036-1.

34 Seung Woo Shin, Chris Thachuk, and Erik Winfree. Verifying chemical reaction network implementations: A pathway decomposition approach. *Theoretical Computer Science*, 2017. doi:doi:10.1016/j.tcs.2017.10.011.

35 Friedrich C. Simmel, Bernard Yurke, and Hari R. Singh. Principles and applications of nucleic acid strand displacement reactions. *Chemical Reviews*, 119(10):6326–6369, 2019. PMID: 30714375. doi:10.1021/acs.chemrev.8b00580.

36 David Soloveichik, Matthew Cook, Erik Winfree, and Jehoshua Bruck. Computation with finite stochastic chemical reaction networks. *Natural Computing*, 7(4):615–633, 2008. doi:10.1007/s11047-008-9067-y.

37 David Soloveichik, Georg Seelig, and Erik Winfree. DNA as a universal substrate for chemical kinetics. *Proceedings of the National Academy of Sciences*, 107(12):5393–5398, 2010. doi:10.1073/pnas.0909380107.

38 Carlo Spaccasassi, Matthew R Lakin, and Andrew Phillips. A logic programming language for computational nucleic acid devices. *ACS Synth. Biol.*, 8(7):1530–1547, July 2019. doi:10.1021/acssynbio.8b00229.

39 Niranjan Srinivas, James Parkin, Georg Seelig, Erik Winfree, and David Soloveichik. Enzyme-free nucleic acid dynamical systems. *Science*, 358, 2017. doi:10.1126/science.aal2052.

40 Allison Tai and Anne Condon. Error-free stable computation with polymer-supplemented chemical reaction networks. In Chris Thachuk and Yan Liu, editors, *DNA Computing and Molecular Programming*, pages 197–218, Cham, 2019. Springer International Publishing. doi:10.1007/978-3-030-26807-7_11.

41 Chris Thachuk and Anne Condon. Space and energy efficient computation with DNA strand displacement systems. In Darko Stefanovic and Andrew Turberfield, editors, *DNA Computing and Molecular Programming*, volume 7433 of Lecture Notes in Computer Science, pages 135–149. Springer, 2012. doi:10.1007/978-3-642-32208-2_11.

42 Chris Thachuk, Erik Winfree, and David Soloveichik. Leakless DNA strand displacement systems. In Andrew Phillips and Peng Yin, editors, *DNA Computing and Molecular Programming*, volume 9211 of Lecture Notes in Computer Science, pages 133–153. Springer, 2015. doi:10.1007/978-3-319-21999-8_9.

43 Anupama J. Thubagere, Wei Li, Robert F. Johnson, Zibo Chen, Shayan Doroudi, Yae Lim Lee, Gregory Izatt, Sarah Wittman, Niranjan Srinivas, Damien Woods, Erik Winfree, and Lulu Qian. A cargo-sorting DNA robot. *Science*, 357(6356), 2017. doi:10.1126/science.aan6558.

44 Anupama J Thubagere, Chris Thachuk, Joseph Berleant, Robert F Johnson, Diana A Ardelean, Kevin M Cherry, and Lulu Qian. Compiler-aided systematic construction of large-scale DNA strand displacement circuits using unpurified components. *Nature Communications*, 8:14373, 2017. doi:10.1038/ncomms14373.

45 Toma E Tomov, Roman Tsukanov, Yair Glick, Yaron Berger, Miran Liber, Dorit Avrahami, Doron Gerber, and Eyal Nir. DNA bipedal motor achieves a large number of steps due to operation using microfluidics-based interface. *Acs Nano*, 11(4):4002–4008, 2017. doi:10.1021/acsnano.7b00547.

46 F. H. van Batenburg, A. P. Gultyaev, C. W. Pleij, J. Ng, and J. Oliehoek. Pseudobase: a database with RNA pseudoknots. *Nucleic acids research*, 28(1):201–204, January 2000. doi:10.1093/nar/28.1.201.

47 Marko Vasic, Cameron Chalk, Austin Luchsinger, Sarfraz Khurshid, and David Soloveichik. Programming and training rate-independent chemical reaction networks, 2021. arXiv:2109.11422.

48 Boya Wang, Chris Thachuk, Andrew D Ellington, Erik Winfree, and David Soloveichik. Effective design principles for leakless strand displacement systems. *Proceedings of the National Academy of Sciences*, 115(52):E12182–E12191, 2018. doi:10.1073/pnas.1806859115.

49 Tianbing Xia, John SantaLucia, Mark E. Burkard, Ryszard Kierzek, Susan J. Schroeder, Xiaoqi Jiao, Christopher Cox, and Douglas H. Turner. Thermodynamic parameters for an expanded nearest-neighbor model for formation of RNA duplexes with Watson-Crick base pairs. *Biochemistry*, 37(42):14719–14735, 1998. PMID: 9778347. doi:10.1021/bi9809425.

50 Peng Yin, Harry MT Choi, Colby R Calvert, and Niles A Pierce. Programming biomolecular self-assembly pathways. *Nature*, 451(7176):318–322, 2008. doi:10.1038/nature06451.

51 David Yu Zhang. Cooperative hybridization of oligonucleotides. *Journal of the American Chemical Society*, 133(4):1077–1086, 2010. doi:10.1021/ja109089q.

52 David Yu Zhang and Erik Winfree. Dynamic allosteric control of noncovalent DNA catalysis reactions. *Journal of the American Chemical Society*, 130(42):13921–13926, 2008. doi:10.1021/ja803318t.

53 David Yu Zhang and Erik Winfree. Control of DNA strand displacement kinetics using toehold exchange. *Journal of the American Chemical Society*, 131(47):17303–17314, 2009. doi:10.1021/ja906987s.