HHS Public Access

Author manuscript

Theor Comput Sci. Author manuscript; available in PMC 2017 June 13.

Published in final edited form as:

Theor Comput Sci. 2016 June 13; 632: 21–42. doi:10.1016/j.tcs.2015.06.033.

Modular verification of chemical reaction network encodings via serializability analysis

Matthew R. Lakin[a,*], **Darko Stefanovic**[a,b], and **Andrew Phillips**[c,*]

[a]Department of Computer Science, University of New Mexico, Albuquerque, NM, USA

[b]Center for Biomedical Engineering, University of New Mexico, Albuquerque, NM, USA

[c]Microsoft Research, Cambridge, UK

Abstract

Chemical reaction networks are a powerful means of specifying the intended behaviour of synthetic biochemical systems. A high-level formal specification, expressed as a chemical reaction network, may be compiled into a lower-level encoding, which can be directly implemented in wet chemistry and may itself be expressed as a chemical reaction network. Here we present conditions under which a lower-level encoding correctly emulates the sequential dynamics of a high-level chemical reaction network. We require that encodings are transactional, such that their execution is divided by a "commit reaction" that irreversibly separates the reactant-consuming phase of the encoding from the product-generating phase. We also impose restrictions on the sharing of species between reaction encodings, based on a notion of "extra tolerance", which defines species that may be shared between encodings without enabling unwanted reactions. Our notion of correctness is serializability of interleaved reaction encodings, and if all reaction encodings satisfy our correctness properties then we can infer that the global dynamics of the system are correct. This allows us to infer correctness of *any* system constructed using verified encodings. As an example, we show how this approach may be used to verify two- and four-domain DNA strand displacement encodings of chemical reaction networks, and we generalize our result to the limit where the populations of helper species are unlimited.

Keywords

Chemical reaction networks; Modular verification; Serializability; DNA strand displacement

1. Introduction

Recent successes in DNA nanotechnology have demonstrated the ability of scientists and engineers to exert unprecedented control over matter at the nanoscale. This control has been used in dynamic DNA logic circuits using a range of molecular architectures [1, 2, 3, 4] as well as in the assembly of static structures based on tile assembly [5] or on folding of scaffold strands [6]. Recent work has combined the dynamic and static domains by using a dynamic DNA circuit to trigger self-assembly reactions [7] or reconfigure existing structures

*Corresponding authors (mlakin@cs.unm.edu, aphillip@microsoft.com).

[8, 9]. Theoretical work has explored the computational power of DNA tile assembly processes [10] and of dynamic DNA reactions [11].

In the theory underlying these efforts, chemical reaction networks (CRNs) play a critical role. Previous work has shown that stochastic CRNs are Turing-universal, provided that an arbitrarily small probability of error is allowed [12], and the fundamental limits of deterministic CRN computation have been explored [13]. Therefore, CRNs are a convenient and powerful programming language for synthetic biochemical systems, which enable researchers to define the desired interactions as a CRN and explore them through simulation using established tools and methods. When such a high-level formal CRN has been designed, in order to test it in the laboratory the abstract CRN species must be mapped to actual chemical species whose interactions correspond to those of the formal CRN. From a computer science perspective, this can be thought of as *compiling* the CRN into another CRN that encodes a lower-level behavioral description of the system. Thus, a common feature of these encodings is that a single-step reaction in a formal CRN is implemented as a multi-step process in its encoding, which comprises a number of individual chemical reactions. When an encoding of a multi-reaction system is executed, the executions of the individual reaction encodings can, and almost certainly will, be interleaved with each other. Thus, there are many possibilities for bugs in the design of CRN encodings due to unwanted interleavings of reactions that may, for example, prematurely generate or consume certain species. It is therefore desirable to develop proofs of correctness for CRN encodings. However, verifying such massive, concurrent systems is non-trivial because a large number of interactions may be possible from any given state.

Since the inputs and outputs from this process are both CRNs, this enables the definition of a hierarchy of CRNs, each of which encodes the behavior of the CRN above it in the hierarchy. In this paper we will focus on the verification of a single encoding step in which a formal (higher-level) CRN is translated into an encoding (lower-level) CRN. However, the same techniques could be used to verify each step of a multi-step encoding process separately.

Here we present a powerful, modular framework for proving correctness of CRN encodings. Since CRN encodings are typically defined pointwise by encoding each reaction separately, it is natural to exploit this modularity in our proof technique. We will show that if all individual reaction encodings in the system satisfy certain properties then the whole system may be deduced to be correct in a well-defined sense. Our modular approach to proving correctness will allow us to verify the components of large-scale systems individually, without being limited by the sizes of the corresponding state spaces.

Our approach is inspired by the concept of *serializability* from database theory [14], which requires that interleaved concurrent updates to a database must be equivalent to some serial schedule of those updates. We consider a composition of reaction encodings to be correct if all possible interleavings of their reactions can be rewritten using a small number of simple rules to produce a *serial schedule*. A serial schedule is one in which there is no interleaving between the various reaction encodings, that is, the first reaction encoding runs to completion before the second begins, and so on. We propose serializability as a reasonable

notion of correctness for CRN encodings because serialized executions of encodings can be directly related to executions of the underlying reactions. We will use simple rules to rewrite reaction traces in order to serialize them. CRN encodings that are not serializable may display erroneous behaviours that do not correspond to possible behaviours of the formal CRN, because of unwanted crosstalk between individual reaction encodings. Our correctness criteria will allow us to prove that our encodings do not have such problems. We also generalize our results to the case where fuel species populations may be assumed to be unlimited, either because they are relatively very large or are continually fed from an external source. This is of interest in the case of long-running reaction networks such as oscillators [15].

As an example, we will apply our technique to the verification of several encodings of the approximate majority algorithm of Angluin et al [16] using DNA strand displacement reactions. DNA strand displacement is a simple yet powerful framework for molecular computation, in which an invading strand displaces an incumbent strand that is bound to a template [17, 18]. The applications of DNA strand displacement reactions are numerous and varied, including the construction of logic circuits [19, 20] and neural networks [21], control of self-assembled nanoscale systems [22, 7] and molecular motors [17, 23, 24, 25]. Here we are particularly interested in applications of DNA strand displacement to implement dynamic behaviour expressed as chemical reaction networks: Soloveichik et al. showed that DNA strand displacement *reaction gates* [26] provide a general framework for the implementation of arbitrary CRNs in wet chemistry. A number of encoding schemes for implementing chemical reaction networks using DNA strand displacement have been proposed, such as the four-domain [26] and two-domain schemes [27]—these names refer to the domain-level structure of the strands which denote encoded species in each scheme. In previous work we have explored the use of probabilistic model checking for the verification of two-domain DNA strand displacement systems [28]. However, such approaches are limited by the explosion in the size of the state space as the various species populations increase.

The remainder of this paper is structured as follows. In Section 2 we introduce mathematical notation and preliminary definitions, and we formalize modular CRN encodings in Section 3. In Section 4 we present a (straightforward) completeness proof for reaction encodings, and in Section 5 we present a (more involved) soundness proof. We discuss extensions of these results to handle unlimited fuel populations in Section 6. We present examples of encoding verification in Section 7, and conclude with a discussion and a survey of related work in Section 8. This paper is a revised and extended version of a conference paper (M. R. Lakin, A. Phillips, D. Stefanovic, Modular verification of DNA strand displacement networks via serializability analysis, in: D. Soloveichik, B. Yurke (Eds.), Proceedings of DNA19, Vol. 8141 of Lecture Notes in Computer Science, Springer-Verlag, 2013, pp. 133–146).

2. Preliminaries

We now introduce some preliminary mathematical definitions that will be used throughout the paper. Let $\mathbb{N}$ denote the set of natural numbers, including zero. Given a set X, we write

$\mathbb{N}^X$ for the set of multisets over X, defined as the set of all functions $f : X \to \mathbb{N}$, as is standard. By convention we use upper-case boldface symbols for multisets and upper-case italics for sets. We may write multisets explicitly using the notation $\{x_1 = n_1, \ldots, x_k = n_k\}$, where $n_i > 0$ is the count associated with the corresponding x_i. We write *set*(**A**) for the domain of the function underlying the multiset **A**. For multisets **A; B** $\in \mathbb{N}^X$ we write **A** $\circledast$ **B** to mean that (**A**(x)) $\circledast$ (**B**(x)) for all $x \in X$, where $\circledast$ is any binary relational operator, for example $\leq$. Similarly, we define arithmetic operations on multisets so that (**A** $\pm$ **B**)(x) = (**A**(x)) $\pm$ (**B**(x)) for all $x \in X$. For subtraction we require that **B** $\leq$ **A** to avoid negative multiplicities. If $x \in X$ and $n \in \mathbb{N}$, we write $n \cdot x$ for the multiset **A** $\in \mathbb{N}^X$ such that **A**(x) = n and such that **A**(x') = 0 for $x' \in X$ where $x' \neq x$. Similarly, if **X** $\in \mathbb{N}^X$ and $n \in \mathbb{N}$, we write $n \cdot$ **X** for the multiset **A** $\in \mathbb{N}^X$ such that **A**(x) = $n \cdot$ **X**(x) for all $x \in X$. Finally, we will use the $\uplus$ symbol for the union of *disjoint* sets. We now define some key concepts.

Definition 1 (Chemical reaction networks)—*A chemical reaction network (CRN) is a pair* (X, R), *where X is a finite set whose elements are referred to as chemical species and R is a finite set of chemical reactions over X, which are rewrite rules of the form* **R** $\to$ **P**, *where* **R**, **P** $\in \mathbb{N}^X$ *are referred to as the "reactants" and "products" of the reaction, respectively. We will typically write chemical reactions using the standard "plus" notation, e.g.,* $x_1 + \cdots + x_j \to x_1' + \cdots + x_k'$. *If* r = (**R** $\to$ **P**) *then we let* r^{-1} = (**P** $\to$ **R**) *and observe that* $(r^{-1})^{-1} = r$. Note that we do not consider reaction rates at all in this paper. *We say that a chemical reaction* **R** $\to$ **P** *is well-formed if* **R** $\neq$ **P**, *and that a CRN is well-formed if all constituent reactions are well-formed.*

Henceforth, we assume that all CRNs are well-formed.

Definition 2 (CRN states and reactions)—*A state* **S** *of a CRN* $\mathcal{C}$ = (X, R) *is just a multiset drawn from* $\mathbb{N}^X$. *A reaction* r = (**R** $\to$ **P**) $\in R$ *is* enabled *in state* **S** *if* **R** $\leq$ **S**, *written* **S** $\vdash_{\mathcal{C}} r$. (*The CRN* $\mathcal{C}$ *is included here so we have access to the set of possible reactions.*) *Furthermore, if* **S**′ = **S** − **R** + **P** *then we write* $\mathbf{S} \xrightarrow{r} \mathbf{S}'$ *to indicate that applying the reaction r to* **S** *results in* **S**′.

Definition 3 (CRN traces)—*Given a CRN* $\mathcal{C}$ = (X, R), *a trace* τ *is an ordered list* [r_1, …, r_i, …] *of reactions from R. Traces may be finite or infinite, and we write length*(τ) *for the length of the finite trace τ. We write* Traces($\mathcal{C}$) *for the set of all traces that may be generated using reactions from R. We write* $\tau_1 : \tau_2$ *for the trace obtained by concatenating* τ_1 *and* τ_2, *and* ε *to denote the empty trace* (*i.e., length*(ε) = 0).

Definition 4 (Valid traces)—*A trace* τ *is a* valid *trace, starting from* **S** *and under a CRN* $\mathcal{C}$, *written* **S** $\vdash_{\mathcal{C}} \tau$, *if* $\tau \in$ Traces($\mathcal{C}$) *and either* (*i*) $\tau = \varepsilon$ *or* (*ii*) $\tau = [r]{:}\ \tau'$ *and* $\mathbf{S} \xrightarrow{r} \mathbf{S}'$ *such that* **S**′ $\vdash_{\mathcal{C}} \tau'$ *also holds. If τ is finite, we write* $final_{\mathcal{C}}(\mathbf{S}, \tau)$ *for the state that is produced when all reactions in the trace have been executed, and say that* $\mathbf{S} \xrightarrow{\tau} \mathbf{S}'$ *if* **S**′ = $final_{\mathcal{C}}(\mathbf{S}, \tau)$.

Example 2.1 (Examples of valid and invalid traces)—*For example, given a CRN* $\mathcal{C}$ *with species* {a, b, c} *and reactions* {$a \to b$, $a + b + b \to c$} *the trace* τ = [$a \to b$, $a \to b$, $a + b + b \to c$] *is valid from the initial state* **S** = {a = 3}, *i.e.,* **S** $\vdash_{\mathcal{C}} \tau$ *holds, whereas the trace* τ' =

[$a \to b$, $a + b + b \to c$, $a \to b$] *is* not *valid from the same initial state because the second reaction cannot be executed, i.e.,* $\mathbf{S} \vdash_{\mathcal{C}} \tau'$ does not *hold*.

Definition 5 (Reachable states)—*A state* $\mathbf{S}'$ *is* reachable *from a state* $\mathbf{S}$ *under a CRN* $\mathcal{C} = (X, R)$ *if* $\mathbf{S} \xrightarrow{\tau} \mathbf{S}'$ *holds for some* $\tau \in$ Traces($\mathcal{C}$). *Furthermore, we say that a state* $\mathbf{S}'$ *is* universally reachable *from* $\mathbf{S}$ *under* $\mathcal{C}$ *if* $\mathbf{S}'$ *is reachable from every state that is reachable from* $\mathbf{S}$.

We note that the definition of universal reachability is related to that of confluence from term rewriting [30], with the difference that (universal) reachability is defined in terms of the state being reached, whereas confluence is defined in terms of the state from which the reduction starts.

Definition 6 (Terminal states and traces)—*A state* $\mathbf{S}$ *is* terminal *under a CRN* $\mathcal{C} = (X, R)$ *if no reaction* $r \in R$ *is enabled in* $\mathbf{S}$. *A* terminal trace *from a state* $\mathbf{S}$ *is any finite trace* $\tau \in$ Traces($\mathcal{C}$) *such that* $final_{\mathcal{C}}(\mathbf{S}, \tau)$ *is a terminal state under* $\mathcal{C}$.

Definition 7 (Reversible and committed reactions)—*Given a CRN* $\mathcal{C} = (X, R)$, *a reaction* $r \in R$ *is* reversible *if the inverse reaction* r^{-1} *also appears in R, and r is* committed *if, for all* $\mathbf{S}$ *and* $\mathbf{S}'$, $\mathbf{S} \xrightarrow{r} \mathbf{S}'$ *implies that* $\mathbf{S}$ *is not reachable from* $\mathbf{S}'$ *under* $\mathcal{C}$.

Note that, in general, it is *not* the case that every reaction is necessarily either reversible or committed in the above sense: for example, consider the CRN with reactions $a \to b$, $b \to c$ and $c \to a$. None of these reactions are reversible, but none of the reactions are committed either, because there is always a route back to the previous state via the other two irreversible reactions. In practice, such reactions would violate conservation laws: similar reaction cycles in natural biochemical systems typically require the consumption of some fuel species that provides the chemical potential energy to drive the system round the cycle, e.g., the hydrolysis of ATP powers the chemomechanical cycles of motor proteins such as myosin and kinesin. Furthermore, wet chemistry implementations of reaction encodings, e.g., using DNA strand displacement reactions, will also have this property, as the resulting reactions are either directly reversible “toehold exchange” reactions [31], or committed reactions that produce one or more inert “waste” species, which prevent that strand displacement reaction from ever being undone. For this reason, we will only consider encodings in which all reactions are either reversible or committed.

3. Chemical reaction network encodings

In this paper, we consider CRN encodings that are constructed in a pointwise, modular fashion by generating a separate encoding of each formal reaction and combining these to produce an encoding of the entire formal CRN. We will assume the existence of a formal CRN $\mathcal{F} = (X_{\mathcal{F}}, R_{\mathcal{F}})$, along with an encoding CRN $\mathcal{E} = (X_{\mathcal{E}}, R_{\mathcal{E}})$ that is intended to provide an implementation of $\mathcal{F}$. The first step in this process is to define the encoding of species from a formal CRN into the encoding CRN.

Definition 8 (Species encodings)—*A species encoding is a function* $\mathcal{M}$: $X_{\mathcal{F}} \rightarrow 2^{X_{\mathcal{E}}}$ *such that* $\mathcal{M}(x_1) \cap \mathcal{M}(x_2) = \emptyset$ *for* $x_1 \neq x_2 \in X_{\mathcal{F}}$. *We write* $\mathcal{M}^{-1}$ *for the "inverse" mapping, which is defined as follows:*

$$\mathscr{M}^{-1}(x') = \begin{cases} x & if\ x' \in \mathscr{M}(x) \\ undefined & otherwise. \end{cases}$$

To motivate the above definition, the ability to use a family of species to encode a single formal species will enable us to model a range of practical DNA implementations of CRN encodings, as will be seen below. The intuition here is that a formal state **S**, in which **S**(x) = n_x for each $x \in X_{\mathcal{F}}$, can be represented by *any* encoding state **S**′ in which $\sum_{x' \in \mathcal{M}(x)} \mathbf{S}'(x') = n_x$ for each $x \in X_{\mathcal{F}}$, i.e., the *sum* of the populations of all encoding species that represent a given formal species x yields the encoded population of that formal species. Since the sets returned by $\mathcal{M}$ are disjoint for distinct input values, one can tell exactly which species from $X_{\mathcal{F}}$ (if any) is encoded by a given species from $X_{\mathcal{E}}$ (this criterion is related to injectivity). This property also ensures that $\mathcal{M}^{-1}$ is a function.

Furthermore, this definition requires that $X_{\mathcal{E}} \supseteq (\cup_{x \in X_{\mathcal{F}}} \mathcal{M}(x))$, i.e., there are sufficient species in $X_{\mathcal{E}}$ to handle *all possible encodings* of the species from $X_{\mathcal{F}}$ under the species encoding $\mathcal{M}$. Any species present in $X_{\mathcal{E}}$ that do *not* correspond directly to encoded species from $X_{\mathcal{F}}$ will mediate the transitions between encoded species, as specified by the formal CRN. Note, also, that $\mathcal{M}^{-1}$ is undefined on any encoding species that does not encode a formal species. Examples of species encodings used in practical systems are the four-domain DNA strand displacement encoding introduced in [26] (see Section 7.3) and the two-domain DNA strand displacement encoding introduced in [27] (see Section 7.4).

Now, to simplify the presentation of the theorems and their proofs, we will overload the notation for species encodings to apply directly to sets and multisets of species. We will use $\mathcal{M}$ to also represent its pointwise application to states, which are just multisets of formal species, as follows.

Definition 9 (Species encodings applied to states)—*For a state* $\mathbf{S} = \{x_1 = n_1, \ldots, x_k = n_k\}$ *of formal species, we write* $\mathcal{M}(\mathbf{S})$ *to stand for* some *multiset*

$\{ x_1^1 = n_1^1, \ldots, x_1^{j_1} = n_1^{j_1}, \ldots, x_k^1 = n_k^1, \ldots, x_k^{j_k} = n_k^{j_k}\}$ *where, for all* $i \in \{1, \ldots, k\}$, $\{x_i^1, \ldots, x_i^{j_i}\} \subseteq \mathscr{M}(x_i)$ *and* $n_i^1 + \cdots + n_i^{j_i} = n_i$.

For a state $\mathbf{S}' = \{x_1' = n_1, \ldots, x_k' = n_k\}$ *of encoding species, we let* $\mathcal{M}^{-1}(\mathbf{S}') = \{x_1 = n_1, \ldots, x_j = n_j\}$ (*for some* $j \leq k$) *where, for all* $i \in \{1, \ldots, j\}$, $x_i \in \mathcal{M}^{-1}(\mathbf{S}')$ *iff* $x_i = \mathcal{M}^{-1}(x')$ *for some* $x' \in \mathbf{S}'$, *and* $n_i = \sum_{x'' \in \mathcal{M}(x_i)} \mathbf{S}'(x'')$.

The above definition is not deterministic because a species encoding $\mathcal{M}$ may offer a range of possibilities for encoding a given formal species. Having defined species encodings, we now present a formal definition of individual reaction encodings, which will be subsequently composed to produce encodings of entire formal CRNs, in a modular fashion.

Definition 10 (Reaction encodings)—*The* reaction encoding *of a formal reaction* $\alpha = (\mathbf{R}_\alpha \rightarrow \mathbf{P}_\alpha)$ *using a species encoding* $\mathcal{M}$ *is* $[\![\alpha]\!] = (\mathcal{E}_\alpha, \mathbf{F}_\alpha)$. *The encoding CRN* $\mathcal{E}_\alpha = (X_{\mathcal{E}_\alpha}, R_{\mathcal{E}_\alpha})$ *is such that*:

1. $X_{\mathcal{E}_\alpha} = (\biguplus_{x \in (set(\mathbf{R}_\alpha \cup \mathbf{P}_\alpha))} \mathcal{M}(x)) \uplus O_\alpha$, *where* O_α *contains* all *species involved in the encoding that are not encodings of formal species, and*
2. $\mathbf{F}_\alpha$ *is a multiset of species from* O_α *such that, for any possible initial state* $\mathbf{S}_\alpha^{init} = \mathscr{M}(\mathbf{R}_\alpha) + \mathbf{F}_\alpha$ (*recall that there are multiple possible results of* $\mathcal{M}(\mathbf{R}_\alpha)$, *as defined in Definition 8), there exists a terminal state* $\mathbf{S}_\alpha^{final}$ *that is universally reachable from* $\mathbf{S}_\alpha^{init}$ *using the chemical reactions from* $R_{\mathcal{E}_\alpha}$, *and where* $\mathscr{M}^{-1}(\mathbf{S}_\alpha^{init}) = \mathbf{R}_\alpha$ *and* $\mathscr{M}^{-1}(\mathbf{S}_\alpha^{final}) = \mathbf{P}_\alpha$.

We also require that no $\mathbf{F} < \mathbf{F}_\alpha$ *has the above properties, meaning that* $\mathbf{F}_\alpha$ *is the minimal amount of fuel needed for a trace that reaches the terminal state, as described above. If* $\mathbf{S}_\alpha^{init} \xrightarrow{\tau} \mathbf{S}_\alpha^{final}$ *we say that the trace* τ *is an* execution *of* $[\![\alpha]\!]$.

In Section 7 we present several examples of different reaction encodings, all applied to the same abstract CRN, the approximate majority voting system originally introduced in [16].

Intuitively, a reaction encoding $[\![\alpha]\!]$ comprises a CRN $\mathcal{E}_\alpha$ that describes the possible interactions within the reaction encoding and a multiset $\mathbf{F}_\alpha$ containing the minimal amount of fuel which, when placed with the encoded reactants $\mathcal{M}(\mathbf{R}_\alpha)$, allows a single execution of $[\![\alpha]\!]$ to run to completion from any initial state $\mathbf{S}_\alpha^{init} = \mathscr{M}(\mathbf{R}_\alpha) + \mathbf{F}_\alpha$. We also require that the encoding always finishes in the same terminal state $\mathbf{S}_\alpha^{final}$, in which the only encoded formal species are the products of α. Furthermore, the encoding should never get stuck in a state from which $\mathbf{S}_\alpha^{final}$ is not reachable.

We assume that the minimal fuel multiset is unique, as is the case in all existing published reaction encodings. The fuel minimality condition could only be violated if there were two independent reaction pathways in the encoding, each requiring different fuel species. Note that we refer to the species from $\mathbf{F}_\alpha$ from Definition 10 as *fuels*, a term we use to mean *any* species that must be present initially in order for the chemical reactions in the encoding to run to completion. In addition to these fuel species, and the species that directly encode formal species, the encoding CRNs will, in general, contain other species that are intermediaries produced during the execution of the encoding, and waste species. At this stage, these other categories are not critical to the exposition, as they are most relevant to restricting crosstalk to ensure soundness of encodings, so we will defer further discussion of them until Section 5. Furthermore, note that the requirement of a terminal state that is universally reachable implies that there can only be a single terminal state: if there were a second terminal state, then by definition the first would not be reachable from the second and hence would not be universally reachable.

Note that the definition of reaction encodings from Definition 10 specifies the behaviour of the system from *any* initial state $\mathbf{S}_\alpha^{init}$. This is because if the species encoding $\mathcal{M}$ maps any of the reactants of the formal reaction to more than one different encoding species then there

will be more than one possible corresponding initial state. We require that every reaction encoding must respect the species encoding in the sense that all possible patterns of encoded reactants should produce the desired result from the reaction encoding. In making this definition, we rely on the fact that $\mathcal{M}(x)$ returns the set of all possible encodings of the formal species x. Thus, we can check all possible patterns of encoded reactants for correct behaviour under the reactant encoding. This issue disappears in the special case where $\mathcal{M}$ maps each formal species into precisely one encoding species, as is the case in encodings based on two-domain DNA strand displacement [27].

Now, individually defined reaction encodings can be combined to produce an encoding of a full CRN, which we define as follows.

Definition 11 (CRN encodings)—*Consider a formal CRN* $\mathcal{F} = (X_{\mathcal{F}}, R_{\mathcal{F}})$ *and a species encoding* $\mathcal{M}$, *where* $R_{\mathcal{F}} = \{a_1, \ldots, a_n\}$ *and* $[\![a_i]\!] = (\mathcal{E}_{a_i}, \mathbf{F}_{a_i})$, *for* $i \in \{1, \ldots, n\}$, *where* $\mathcal{E}_{a_i} = (X_{\mathcal{E}_{a_i}}, R_{\mathcal{E}_{a_i}})$. *Then, the CRN encoding of* $\mathcal{F}$ *comprises*:

1. *an encoding CRN* $\mathcal{E} = (X_{\mathcal{E}}, R_{\mathcal{E}})$, *where* $X_{\mathcal{E}} = (X_{\mathcal{E}_{a_1}} \cup \cdots \cup X_{\mathcal{E}_{a_n}})$ *and* $R_{\mathcal{E}} = (R_{\mathcal{E}_{a_1}} \cup \cdots \cup R_{\mathcal{E}_{a_n}})$, *and*
2. *the list* $\mathbf{F}_{a_1}, \ldots, \mathbf{F}_{a_n}$ *of fuel multisets from the individual reaction encodings.*

Note that the set of possible reactions in $\mathcal{E}$ is just the union of the possible sets of reactions from the CRNs that encode the individual formal reactions from $\mathcal{F}$. If a particular species is shared between two reaction encodings, then it may react with other species from either encoding according to the rules of the corresponding CRNs from the individual encodings.

4. Completeness of CRN encodings

We now consider correctness properties of an encoding CRN $\mathcal{E}$ with respect to the formal CRN $\mathcal{F}$. The most basic correctness property of CRN encodings is that they are capable of emulating any valid trace of formal reactions, which we define as follows.

Definition 12 (Serial executions)—*For a formal trace* $\tau \in \text{Traces}(\mathcal{F})$, *we say that a trace* $\tau' \in \text{Traces}(\mathcal{E})$ *in the encoding CRN* $\mathcal{E}$ *is a* serial execution *of* $\tau = [r_1, \ldots, r_n]$ *if* τ' *can be decomposed into a concatenation* $\tau_1 : \ldots : \tau_n$ *of subtraces such that the* i^{th} *subtrace* τ_i *is an execution of* r_i.

Since reaction encodings require fuel species to be present, any statement about the correctness of reaction encodings must be predicated on the amount of fuel available in the system. Thus it is important to identify the minimal amount of fuel needed to emulate a given trace of formal reactions.

Lemma 4.1 (Fuel required for emulation)—*Let* $\tau = [a_1, \ldots, a_n] \in \text{Traces}(\mathcal{F})$ *be a finite trace of formal reactions, let* $\mathbf{S}$ *be a formal state such that* $\mathbf{S} \vdash_{\mathcal{F}} \tau$, *and let* $\tau_{ser} \in \text{Traces}(\mathcal{E})$ *be a serial execution of* τ *using the encoding CRN* $\mathcal{E}$. *Then, for any* $\mathcal{M}(\mathbf{S})$, *it is the case that* $(\mathcal{M}(\mathbf{S}) + \mathbf{F}) \vdash_{\mathcal{E}} \tau_{ser}$ *iff* $\mathbf{F} \geq \text{reqfuel}(\tau)$, *where the required fuel,* $\text{reqfuel}(\tau)$, *is defined as the sum of the fuel required by the encoding of each reaction in the formal trace, i.e.,* $\text{reqfuel}(\tau) \triangleq \mathbf{F}_{a_1} + \cdots + \mathbf{F}_{a_n}$.

Proof: This result follows from the definitions of reaction encodings and CRN encodings from Definition 10 and Definition 11, respectively. (Given that traces are sequences of reactions, they are inherently sequential). The "only if" direction of this proof uses the assumption that all reactions are either reversible or committed, in the sense of Definition 7.

We can now state and prove a straightforward completeness theorem for the emulation of formal traces by serial executions.

Theorem 4.2 (Completeness)—*Let* $\tau \in \text{Traces}(\mathcal{F})$ *be a finite formal trace and let* $\mathbf{S}$ *be a formal state. If* $\mathbf{S} \vdash_{\mathcal{F}} \tau$ *and* $\mathbf{F} \geq \text{reqfuel}(\tau)$ *both hold then, for any* $\mathcal{M}(\mathbf{S})$, *it is the case that* $(\mathcal{M}(\mathbf{S}) + \mathbf{F}) \vdash_{\mathcal{E}} \tau_{ser}$ *holds for some* $\tau_{ser} \in \text{Traces}(\mathcal{E})$ *that is a serial execution of* τ *using the encoding CRN* $\mathcal{E}$.

Proof: This result follows from Lemma 4.1, because we can emulate any finite formal trace by creating an initial state with sufficient encoded species $\mathcal{M}(\mathbf{S})$ and fuels $\mathbf{F}$ so that the corresponding serial execution can be run.

5. Soundness of CRN encodings

The completeness result above only concerns one possible trace of the encoding. In this section we prove a more involved *soundness* result, which shows that *all* possible traces of the encoding are equivalent to a serial execution of some valid formal trace. This is a reasonable notion of correctness because this implies that every possible trace of the encoding can be causally related to some valid formal trace. To make this connection, we define a notion of *rewriting* on valid reaction traces, which allows reactions to be moved around and deleted from the trace if doing so preserves the causal relationships between reactions in the trace.

Definition 13 (Trace rewriting)—*The trace rewriting relation is indexed by a CRN* $\mathcal{C}$ *and the starting state* $\mathbf{S}$. *We write* $\mathbf{S} \vdash_{\mathcal{C}} \tau \rightsquigarrow \tau'$ *to mean that, for a CRN* $\mathcal{C}$ *and starting state* $\mathbf{S}$, *the trace* τ *can be rewritten to produce the trace* τ'. *These judgments are derived by constructing a proof tree using the following inference rules.*

$$(\text{REFL})\frac{\mathbf{S}\vdash_{\mathscr{C}}\tau}{\mathbf{S}\vdash_{\mathscr{C}}\tau\rightsquigarrow\tau} \qquad (\text{TRANS})\frac{\mathbf{S}\vdash_{\mathscr{C}}\tau\rightsquigarrow\tau' \qquad \mathbf{S}\vdash_{\mathscr{C}}\tau'\rightsquigarrow\tau''}{\mathbf{S}\vdash_{\mathscr{C}}\tau\rightsquigarrow\tau''}$$

$$(\text{CANCEL})\frac{\mathbf{S}\vdash_{\mathscr{C}}\tau_1:\tau_2:\tau_3 \qquad \mathbf{S}\xrightarrow{\tau_1}\mathbf{S}'\xrightarrow{\tau_2}\mathbf{S}'}{\mathbf{S}\vdash_{\mathscr{C}}\tau_1:\tau_2:\tau_3\rightsquigarrow\tau_1:\tau_3} \qquad (\text{SWAP})\frac{\mathbf{S}\vdash_{\mathscr{C}}\tau_1:\tau_2:\tau_3:\tau_4 \qquad \mathbf{S}\vdash_{\mathscr{C}}\tau_1:\tau_3:\tau_2:\tau_4}{\mathbf{S}\vdash_{\mathscr{C}}\tau_1:\tau_2:\tau_3:\tau_4\rightsquigarrow\tau_1:\tau_3:\tau_2:\tau_4}$$

The inference rules from Definition 13 follow a standard format from structural operational semantics and can be interpreted either top-down (if the premises from the top line hold, then the conclusion from the bottom line holds also) or bottom-up (to prove that the conclusion holds, we must prove that the premises hold). We note that, by the definition of the inference rules from Definition 13, the trace rewriting rules can only be applied to *valid* traces. The (REFL) rule ensures that the trace rewriting relation is reflexive, i.e., any valid trace can be trivially rewritten to itself, and the (TRANS) rule means that the rewriting

relation is transitive, i.e., multiple rewrite steps can be combined into a single rewriting judgment using this rule. The (CANCEL) rule allows the subtrace τ_2 to be removed if its net effect is no change, which allows a cycle of a reaction and its inverse to be canceled if they appear next to each other in the trace. The (SWAP) rule allows two neighbouring subtraces τ_2 and τ_3 to be swapped if they may occur in either order, when preceded by τ_1 and followed by τ_4, i.e., subtraces that are causally independent can occur in either order. In the (SWAP) rule, since executing a reaction trace is essentially a series of addition and subtraction operations on the species populations, which are commutative, it follows that $\tau_1 : \tau_2 : \tau_3$ and $\tau_1 : \tau_3 : \tau_2$ both produce the same final state. Finally, we note that the prefix and suffix traces τ_1 and τ_3 from (CANCEL), and τ_1 and τ_4 from (SWAP) may be empty, so these rules can also be applied at the very beginning or the very end of a trace without the need for additional contextual rules.

Example 5.1 (Trace rewriting)—*Consider a CRN* $\mathcal{C} = (\{x_1, x_2, x_3\}, \{r_1, r_2, r_3\})$*, where*

$$r_1 = (x_1 \to x_2) \quad r_2 = (x_1 + x_2 \to x_3) \quad r_3 = (x_1 \to x_3),$$

and an initial state $\mathbf{S} = \{x_1 = 3\}$*. Note that* $\mathbf{S} \vdash_{\mathcal{C}} [r_1, r_2, r_3]$ *holds, by the following sequence of reactions:*

$$\{x_1 = 3\} \xrightarrow{r_1} \{x_1 = 2, x_2 = 1\} \xrightarrow{r_2} \{x_1 = 1, x_3 = 1\} \xrightarrow{r_3} \{x_3 = 2\}.$$

Since there are sufficient copies of x_1 *in the initial state,* r_3 *can be executed independently of the other two reactions. Therefore, we could execute* r_3 *first, i.e., the trace rewriting judgment* $\mathbf{S} \vdash_{\mathcal{C}} [r_1, r_2, r_3] \rightsquigarrow [r_3, r_1, r_2]$ *also holds. This can be derived by constructing a proof tree as follows.*

$$(\text{TRANS})\frac{(\text{SWAP})\dfrac{\overline{\mathbf{S}\vdash_{\mathscr{C}}[r_1,r_2,r_3]}}{\mathbf{S}\vdash_{\mathscr{C}}[r_1,r_2,r_3]\rightsquigarrow[r_1,r_3,r_2]}\quad\overline{\mathbf{S}\vdash_{\mathscr{C}}[r_1,r_3,r_2]}}{\mathbf{S}\vdash_{\mathscr{C}}[r_1,r_2,r_3]\rightsquigarrow[r_1,r_3,r_2]}\quad(\text{SWAP})\dfrac{\overline{\mathbf{S}\vdash_{\mathscr{C}}[r_1,r_3,r_2]}}{\mathbf{S}\vdash_{\mathscr{C}}[r_1,r_3,r_2]\rightsquigarrow[r_3,r_1,r_2]}\quad\overline{\mathbf{S}\vdash_{\mathscr{C}}[r_3,r_1,r_2]}}{\mathbf{S}\vdash_{\mathscr{C}}[r_1,r_2,r_3]\rightsquigarrow[r_3,r_1,r_2]}$$

Note, however, that $\mathbf{S} \vdash_{\mathcal{C}} [r_1, r_2, r_3] \rightsquigarrow [r_2, r_1, r_3]$ *does* not *hold. This is because the* (SWAP) *rule cannot be applied to the first two reactions in this judgment, due to the absence of* x_2 *in the initial state.*

It is important to show that trace rewriting preserves validity and the final states produced by executing the traces. This can be done as follows.

Lemma 5.2—*If* $\mathbf{S} \vdash_{\mathcal{C}} \tau \rightsquigarrow \tau'$ *then* $\mathbf{S} \vdash_{\mathcal{C}} \tau'$ *and* $final_{\mathcal{C}}(\mathbf{S}, \tau) = final_{\mathcal{C}}(\mathbf{S}, \tau')$.

Proof: Assume that $\mathbf{S} \vdash_{\mathcal{C}} \tau \rightsquigarrow \tau'$. By definition of the inference rules from Definition 13 it follows that $\mathbf{S} \vdash_{\mathcal{C}} \tau$. If (CANCEL) was used to derive $\mathbf{S} \vdash_{\mathcal{C}} \tau \rightsquigarrow \tau'$ then τ has the form $\tau_1 : \tau_2 : \tau_3$ and τ' has the form $\tau_1 : \tau_3$, and furthermore we know that $\mathbf{S} \xrightarrow{\tau_1} \mathbf{S}' \xrightarrow{\tau_2} \mathbf{S}'$. It follows immediately that $\mathbf{S} \vdash_{\mathcal{C}} \tau_1 : \tau_3$, and hence that $\mathbf{S} \vdash_{\mathcal{C}} \tau'$ and $final_{\mathcal{C}}(\mathbf{S}, \tau) = final_{\mathcal{C}}(\mathbf{S}, \tau')$, as required.

On the other hand, if (SWAP) was used to derive $\mathbf{S} \vdash_{\mathcal{C}} \tau \rightsquigarrow \tau'$ then τ has the form $\tau_1 : \tau_2 : \tau_3 : \tau_4$ and τ' has the form $\tau_1 : \tau_3 : \tau_2 : \tau_4$. Furthermore, we know that $\mathbf{S} \vdash_{\mathcal{C}} \tau_1 : \tau_3 : \tau_2$, and $final_{\mathcal{C}}(\mathbf{S}, \tau_1 : \tau_3 : \tau_2) = final_{\mathcal{C}}(\mathbf{S}, \tau_1 : \tau_2 : \tau_3)$. Thus it follows that $\mathbf{S} \vdash_{\mathcal{C}} \tau_1 : \tau_3 : \tau_2 : \tau_4$, and hence that $\mathbf{S} \vdash_{\mathcal{C}} \tau'$ and $final_{\mathcal{C}}(\mathbf{S}, \tau) = final_{\mathcal{C}}(\mathbf{S}, \tau')$, as required. The cases for the remaining rules, (REFL) and (TRANS), are straightforward.

Note that it is *not* the case that any two valid traces from a given starting state can be interconverted using these trace rewriting rules—indeed, this is the crux of our analysis. Proving soundness is challenging because we must show that *all* possible interleavings of the various reaction encodings in a given system can be rewritten to produce a serial execution of a valid formal trace. In particular, sharing of species between reaction encodings may lead to non-serializable behaviour, e.g., if a species generated during the execution of one reaction encoding can erroneously trigger part of a second reaction encoding. Therefore, to obtain a soundness result, we must place several additional constraints on the reaction encodings which we will consider.

Definition 14 (Stratified CRNs)—*If a state* $\mathbf{S}'$ *is reachable from* $\mathbf{S}$ *under* $\mathcal{C}$*, we write* $\Lambda(\mathbf{S}, \mathbf{S}')$ *for the length of the shortest trace* $\tau \in$ Traces($\mathcal{C}$) *such that* $\mathbf{S} \xrightarrow{\tau} \mathbf{S}'$*. Then, we say that the CRN* $\mathcal{C}$ *is* stratified *if, for any starting state* $\mathbf{S}_0$ *and any states* $\mathbf{S}$ *and* $\mathbf{S}'$ *which are reachable from* $\mathbf{S}_0$ *under* $\mathcal{C}$*, it is the case that* $\mathbf{S} \xrightarrow{r} \mathbf{S}'$ *implies* $\Lambda(\mathbf{S}_0, \mathbf{S}') = \Lambda(\mathbf{S}_0, \mathbf{S}) \pm 1$.

Example 5.3 (Example of stratified and non-stratified CRNs)—*As a simple example, a CRN containing species* $\{x_1, x_2, x_3\}$ *and reactions* $\{x_1 \to x_2, x_2 \to x_3\}$ is *stratified, whereas a CRN containing the same species and reactions* $\{x_1 \to x_2, x_2 \to x_3, x_1 \to x_3\}$ is not *stratified, because* x_1 *can be converted to* x_3 *either via the intermediate species* x_2 *(with a trace of length 2) or directly (with a trace of length 1).*

We consider it reasonable to restrict our attention to stratified CRNs. In particular, CRNs comprising the form of reactions typical to chemical implementation frameworks such as DNA strand displacement [17, 18] give rise to stratified CRNs (when considered at a fixed level of abstraction [15]), as the type of “shortcircuit” reaction mentioned above is not possible. This can be seen in the state spaces from our examples encoded using four-domain and two-domain strand displacement schemes, which are illustrated in Figures 6 and 9.

In the context of reaction encodings, the knowledge that the encoding CRN is stratified enables us to impose similar directionality on the individual reactions that execute the encoding, as follows.

Definition 15 (Forward and backward reactions in reaction encodings)—*Consider a reaction encoding* $[\![\alpha]\!]$ *that uses a stratified encoding CRN* $\mathcal{E}$*. Recall the initial state* $\mathbf{S}_{\alpha}^{init}$ *of the encoding, as defined in Definition 10, and construct the state space of the encoding, that is, the graph of reachable states starting from* $\mathbf{S}_{\alpha}^{init}$ *and using the reaction rules from* $R_{\mathcal{E}}$ *to make state transitions. Then, we say that a transition* $\mathbf{S} \xrightarrow{r} \mathbf{S}'$ *in the state space is a* forward step *if* $\Lambda(\mathbf{S}_{\alpha}^{init}, \mathbf{S}') = \Lambda(\mathbf{S}_{\alpha}^{init}, \mathbf{S}) + 1$*, i.e., the reaction takes us farther from the initial*

state. Similarly, we say that the transition is a reverse step *if* $\Lambda(\mathbf{S}_\alpha^{init}, \mathbf{S}') = \Lambda(\mathbf{S}_\alpha^{init}, \mathbf{S}) - 1$, *i.e., the reaction moves us back towards the initial state.*

In the state spaces from our example reaction encodings, shown in Figures 2, 6, and 9, the distances from the initial state to any given state in the state space can be easily calculated: thus, it is clear which reactions are forward steps and which are backward steps. All of these examples yield stratified CRNs. Indeed, it would be meaningless to apply Definition 15 to a reaction encoding that did not yield a stratified CRN, as the distance from the initial state would be ill-defined.

Henceforth, we will assume that all CRNs are stratified. Furthermore, to simplify the presentation, we will also categorize the species involved in each reaction encoding according to their role. Hence we require that the set $X_{\mathcal{E}_\alpha}$ of *all* species involved in the reaction encoding $[\![\alpha]\!]$, which we will also write as *species* ($[\![\alpha]\!]$), can be partitioned into:

1. *formals*($[\![\alpha]\!]$)—those species in the image of $\mathcal{M}$;
2. *waste*($[\![\alpha]\!]$)—those species which are unreactive, i.e., never appear as a reactant in any reaction in $R_{\mathcal{E}_\alpha}$;
3. *fuels*($[\![\alpha]\!]$)—those species which appear in $\mathbf{F}_\alpha$; and
4. *intermediates*($[\![\alpha]\!]$)—the remaining species.

Now, to determine if two reaction encodings can be safely used with each other, we must test whether the presence of any extra copies of a particular species will cause an encoding to function incorrectly. To this end, we define a notion of *extra tolerance*, as follows.

Definition 16 (Extra tolerance)—*A reaction encoding* $[\![\alpha]\!]$ *is extra tolerant with respect to a species x if*

$$\{\tau | \mathbf{S}_\alpha^{init} \vdash_{\mathcal{E}_\alpha} \tau\} = \{\tau | (\mathbf{S}_\alpha^{init} + n \cdot x) \vdash_{\mathcal{E}_\alpha} \tau\}$$

for any $n \in \mathbb{N}$, *where* $\mathcal{E}_\alpha$ *is the CRN from the encoding* $[\![\alpha]\!]$ *(extended to include x if necessary) and* $\mathbf{S}_\alpha^{init}$ *is any initial state of* $[\![\alpha]\!]$.

Thus, $\mathbf{S}_\alpha^{init}$ contains an encoding of the reactant species and the necessary fuel species. We can decide extra tolerance for a reaction encoding by checking whether the set of traces produced from any initial state of the encoding is identical in the presence or absence of the species *x*. If the sets of traces are the same, it follows that the state spaces have the same structure, but with *n* additional copies of *x* in each state.

For example, in a four-domain DNA strand displacement reaction encoding, of the kind described in Section 7.3, including additional copies of the gate complexes does not enable any additional reactions, so the encoding is extra tolerant with respect to these species. However, the encoding is *not* extra tolerant with respect to the intermediate strand that is released from the reactant-binding gate to activate the product-generating gate. If this

intermediate strand is present in the initial state then the products can be released straight away, which produces a different state space than for the correct execution of the encoding.

We now use the above notion of extra tolerance to state the restrictions on individual reaction encodings that we will impose so that a soundness result can be proved.

Definition 17 (Transactional reaction encodings)—*Consider a formal reaction* $\alpha = (\mathbf{R}_\alpha \rightarrow \mathbf{P}_\alpha)$*, encoded as* $[\![\alpha]\!] = (\mathcal{E}_\alpha, \mathbf{F}_\alpha)$*. Let* $\mathbf{S}_0 = \mathbf{S}_\alpha^{init} = \mathcal{M}(\mathbf{R}_\alpha) + \mathbf{F}_\alpha$ *denote an initial state consisting of just the required reactants and fuels, and let* $\tau = [r_1, \ldots, r_n] \in \text{Traces}(\mathcal{E}_\alpha)$ *be a terminal trace of the encoding starting from* $\mathbf{S}_0$*, where* $r_i = (\mathbf{R}_i \rightarrow \mathbf{P}_i)$ *for* $i \in \{1, \ldots, n\}$*. Labeling the corresponding sequence of states as* $\mathbf{S}_0 \xrightarrow{r_1} \mathbf{S}_1 \xrightarrow{r_2} \cdots \xrightarrow{r_{n-1}} \mathbf{S}_{n-1} \xrightarrow{r_n} \mathbf{S}_n$*, we say that* r_j *is a commit reaction if the following criteria are all satisfied:*

1. r_j is the first committed reaction in τ;
2. *if* $x \in$ *formals*$([\![\alpha]\!])$ *occurs in* $r_1, \ldots, r_{j-1}$ *or* $\mathbf{R}_j$ *then* $x \in \mathcal{M}(\mathbf{R}_\alpha)$*, and these occurrences are all either reactants of forward steps or products of backward steps;*
3. *if* $x \in$ *formals*$([\![\alpha]\!])$ *occurs in* $\mathbf{P}_j$ *or* $r_{j+1}, \ldots, r_n$ *then* $x \in \mathcal{M}(\mathbf{P}_\alpha)$*, and these occurrences are all either products of forward steps or reactants of backward steps;*
4. $\mathcal{M}^{-1}(\mathbf{S}_{j-1} - \mathbf{R}_j) = \emptyset$.

We say that $[\![\alpha]\!]$ *is transactional if, for every possible initial state* $\mathbf{S}_0$*, every terminal trace from* $\mathbf{S}_0$ *has a commit reaction satisfying the above criteria and if every terminal trace visits the same set of states prior to the commit reaction. We also require that* $[\![\alpha]\!]$ *is extra tolerant with respect to all formal and fuel species involved in the encoding, and that the terminal state has the form* $\mathcal{M}(\mathbf{P}_\alpha) + \mathbf{L}_\alpha$*, where* $[\![\alpha]\!]$ *is extra tolerant with respect to every species in the multiset* $\mathbf{L}_\alpha$ *of* leftover species.

In Definition 17, criterion 1 requires that the trace can be partitioned into two disjoint subtraces by the first committed reaction. Criteria 2 and 3 require that only input formal species can engage in reactions before the commit reaction, and only output formal species can engage in reactions after, and furthermore that the consumption of input formal species before the commit reaction and the production of output formal species after the commit reaction always drive the system forwards. Criterion 4 ensures that all necessary reactants are in fact consumed by the time the commit reaction is reached: this is needed in case the reactants and products have some species in common. These criteria correctly handle those cases where the commit reaction itself may consume encoded formal species as reactants and/or release encoded formal species as products. The restrictions on extra tolerance ensure that the behaviour of the encoding is identical in the presence of additional copies of fuels or formal species: note that any encoding is extra tolerant with respect to waste species, but that the encoding may not be extra tolerant with respect to certain intermediate species. We require that every terminal trace visits the same set of states prior to the commit reaction so that there is a single binding pathway for the reactants: this is necessary to show that backward reactions in pre-commit traces can always be eliminated using the trace rewriting

rules. Finally, the restrictions on leftover species in the terminal state delimit those species which may be safely left behind by a reaction encoding that does not fully garbage-collect its intermediate species to produce unreactive waste. The correctness arguments for our three example encodings from Section 7 (Propositions 1, 2, and 3) illustrate the reasoning required to show that encodings are transactional, and the following example illustrates the problems that may arise due to the use of a non-transactional encoding.

Example 5.4 (Problems caused by a non-transactional reaction encoding)— *Following* [32], *we now present a concrete example of the kind of bug that may occur in a CRN encoding when using a non-transactional reaction encoding. Consider the following set of formal reactions:* $\{x \to y, y + a \to y + b\}$, *and suppose that our encoding of* $x \to y$ *is not transactional because the output y can be released before the first committed reaction in the execution of the encoding. Then, from an initial state corresponding to the formal state* $\{x = 1, a = 1\}$ *the following sequence of operations is possible:*

1. *Run the encoding of* $x \to y$ *until y is produced, but* without *executing any committed reactions. This produces a new state corresponding to* $\{y = 1, a = 1\}$.
2. *Completely execute the encoding of* $y + a \to y + b$, *which results in a state corresponding to* $\{y = 1, b = 1\}$.
3. *Unwind the partial execution of* $x \to y$, *which is possible because no committed reactions have been executed in this encoding. The final state corresponds to the formal state* $\{x = 1, b = 1\}$.

Note that the formal state $\{x = 1, b = 1\}$ *is not reachable from the initial formal state* $\{x = 1, a = 1\}$ *using the above set of formal reactions. The only formal reaction that can be executed from the initial state* ($\{x = 1, a = 1\}$) *is* $x \to y$, *therefore, any serial execution must begin with an execution of* $[\![x \to y]\!]$. *Similarly, the next execution must be* $[\![y + a \to y + b]\!]$. *However, this results in a state corresponding to* $\{y = 1, b = 1\}$, *not* $\{x = 1, b = 1\}$ *as obtained above. The specific problem with this example is that the set of reactions outlined above relies on the fact that the y produced by the partial execution of* $[\![x \to y]\!]$ *reaction is available before the encoding has executed a committed reaction to commit to its production. Thus the y species can be used to catalyse the conversion of a to b before being reclaimed and converted back to x. However, in transactional encodings this cannot occur, because the commit reaction must be executed before any products can be released.*

However, requiring that the reaction encodings are individually transactional is not sufficient for correctness because of the possibility of unwanted behaviour caused by crosstalk between reaction encodings. In general, this could occur either via unwanted direct sharing of species, or via crosstalk reactions between species from different reaction encodings. Therefore, we now define *compatible* reaction encodings, in which direct sharing of species between the encodings is only permitted in certain circumstances.

Definition 18 (Compatible reaction encodings)— *We say that two reaction encodings,* $[\![\alpha]\!]$ *and* $[\![\beta]\!]$, *are compatible if every shared species in species*$([\![\alpha]\!]) \cap$ *species*$([\![\beta]\!])$ *appears in the same category (formal, waste, fuel, or intermediate) in both encodings, and if*

both encodings are extra tolerant with respect to every shared species. Furthermore, we require that a species from $[\![\alpha]\!]$ *can only interact with a species from* $[\![\beta]\!]$ *if at least one of those species occurs in species*($[\![\alpha]\!]$) ∩ *species*($[\![\beta]\!]$).

Hence, different reaction encodings may share formal species, waste species, and fuel strands. They may also share intermediate strands provided that the presence of additional copies of those species does not enable additional reaction pathways in either reaction encoding. In all cases, shared species must appear in the same category in both reaction encodings, and no species may interact with any species from a reaction encoding in which it is not present as a species. We note that, when CRN encodings are specified in a modular fashion via CRNs as in Definition 10 and Definition 11, it is in fact not possible for species to interact unless they occur in the same reaction encoding, because we rely on the CRNs from the reaction encodings to specify which species may react. However, we include this possibility in Definition 18 so that our definitions can also be applied when we have a general mechanism to compute whether an arbitrary pair of species may interact. Thus, the following proof techniques can be used to verify both high-level encodings using CRNs and lower-level encodings that account for species structure to check for unwanted interference between reaction encodings, e.g., by using the DSD programming language and compiler [15].

We now present some preliminary lemmas that will be needed to prove our main soundness result.

Lemma 5.5 (Trace rewriting and reversible reactions)—*If* $\tau \in$ Traces($\mathcal{C}$) *consists entirely of reversible reactions and* $\mathbf{S} \vdash_{\mathcal{C}} \tau$ *then there exists* $\tau' \in$ Traces($\mathcal{C}$) *such that* $\mathbf{S} \vdash_{\mathcal{C}} \tau{:}\tau' \rightsquigarrow \varepsilon$.

Proof: Straightforward: if $\tau = [r_1, \ldots, r_n]$ then let $\tau' = [r_n^{-1}, \ldots, r_1^{-1}]$. Then, the last reaction from τ and the first reaction from τ' can be cancelled out via rule (cancel), and so on, eventually leaving the empty trace ε, as required.

Lemma 5.6 (Serializability)—*Assume that all reaction encodings are transactional and pairwise compatible, and suppose that* $\mathbf{S} \vdash_{\varepsilon} \tau$, *where*

$$\tau = \tau_1{:}[r_1^{\alpha}]{:}\cdots{:}\tau_{k-1}{:}[r_{k-1}^{\alpha}]{:}\tau_k{:}[r_{com}^{\alpha}]{:}\tau_{k+1}{:}[r_{k+1}^{\alpha}]{:}\cdots{:}\tau_n{:}[r_n^{\alpha}]{:}\tau_{rest},$$

where r_{com}^{α} *is the first commit reaction in* τ, *where* $\tau_\alpha = [r_1^{\alpha}, \ldots, r_{k-1}^{\alpha}, r_{com}^{\alpha}, r_{k+1}^{\alpha}, \ldots, r_n^{\alpha}]$ *is an execution of* $[\![\alpha]\!]$ *and where* $\mathbf{S} \vdash_{\varepsilon} r_1^{\alpha}$. *Then, there exists* τ'_{rest} *such that* $\mathbf{S} \vdash_{\varepsilon} \tau \rightsquigarrow \tau_\alpha{:}\tau'_{rest}$.

Proof: We assume that $\mathbf{S} \vdash_{\varepsilon} \tau$, i.e., that

$$\mathbf{S} \vdash_{\varepsilon} \tau_1{:}[r_1^{\alpha}]{:}\cdots{:}\tau_{k-1}{:}[r_{k-1}^{\alpha}]{:}\tau_k{:}[r_{com}^{\alpha}]{:}\tau_{k+1}{:}[r_{k+1}^{\alpha}]{:}\cdots{:}\tau_n{:}[r_n^{\alpha}]{:}\tau_{rest}.$$

Since r^{α}_{com} is the first commit reaction in τ, all reactions prior to it must be reversible. Also, we note that the only species which can be shared between encodings are those with respect to which the encodings are extra tolerant, and, because of the stratified CRN property, a single reaction encoding cannot consume more fuel or formal species than is necessary to complete an execution.

By assumption, we know that $\mathbf{S} \vdash_{\mathscr{E}} \tau_1 : [r^{\alpha}_1]$ and $\mathbf{S} \vdash_{\mathscr{E}} [r^{\alpha}_1]$. We must show that there exists $\tau_1^{'}$ such that $\mathbf{S} \vdash_{\mathscr{E}} \tau_1 : [r^{\alpha}_1] \rightsquigarrow [r^{\alpha}_1] : \tau_1^{'}$ holds, for any $\mathbf{S}$. Suppose that x is a reactant of r^{α}_1, and that there are n copies of x in the initial state. Then, we observe that τ_1 and $[r^{\alpha}_1]$ may only conflict if τ_1 consumes all n copies of x and subsequently releases just enough copies of x so that r^{α}_1 can be subsequently executed: if r^{α}_1 were to be executed first then there would be fewer than n copies of x remaining, so τ_1 could not be executed to completion. Since τ_1 is a trace of pre-commit reactions, in which formal species are consumed by forward steps and re-emitted by backward steps, it follows from Definition 17 that the encoding must revisit one of its previous states in order to release a species. Therefore we can use (cancel) to derive $\mathbf{S} \vdash_{\mathscr{E}} \tau_1 \rightsquigarrow \tau_1^{'}$, for some $\tau_1^{'}$ which contains no backward steps. It follows that $\mathbf{S}$ contains sufficient species to execute both $\tau_1^{'}$ and r^{α}_1, and hence it follows that

$$\mathbf{S} \vdash_{\mathscr{E}} \tau \rightsquigarrow [r^{\alpha}_1] : \tau_1^{'} : \tau_2 : [r^{\alpha}_2] : \cdots : \tau_{k-1} : [r^{\alpha}_{k-1}] : \tau_k : [r^{\alpha}_{com}] : \tau_{k+1} : [r^{\alpha}_{k+1}] : \cdots : \tau_n : [r^{\alpha}_n] : \tau_{rest}.$$

By repeated application of this argument we get that

$$\mathbf{S} \vdash_{\mathscr{E}} \tau \rightsquigarrow [r^{\alpha}_1, \ldots, r^{\alpha}_{k-1}, r^{\alpha}_{com}] : \tau^{'} : \tau_{k+1} : [r^{\alpha}_{k+1}] : \cdots : \tau_n : [r^{\alpha}_n] : \tau_{rest},$$

where $\tau^{'} = \tau_1^{'} : \cdots : \tau_k^{'}$.

Now, r^{α}_n is the last reaction in a complete execution of $[\![a]\!]$, which means that it must be a forward step. By Definition 17, forward steps after the commit reaction only have fuels and intermediates as reactants, and by Definition 18 we know that fuel and intermediate strands could potentially be shared with other reaction encodings. However, by Definition 16 we know that if any reaction in τ_n consumes a reactant x of r^{α}_n, then that reaction must be from a copy of the reaction encoding which is in a state where it could consume x anyway, and hence that copy must have its own copy of x present. Hence, since we know that r^{α}_n can be executed directly after r^{α}_{n-1}, we can move r^{α}_n forward using the (swap) rule, to get

$$\mathbf{S} \vdash_{\mathscr{E}} \tau \rightsquigarrow [r^{\alpha}_1, \ldots, r^{\alpha}_{k-1}, r^{\alpha}_{com}] : \tau^{'} : \tau_{k+1} : [r^{\alpha}_{k+1}] : \cdots : \tau_{n-1} : [r^{\alpha}_{n-1}, r^{\alpha}_n] : \tau_n : \tau_{rest}.$$

Now consider the subtrace $[r^{\alpha}_{n-1}, r^{\alpha}_n]$. In general, a trace of post-commit reactions of the form $[r^{\alpha}_{n-m}, \ldots, r^{\alpha}_n]$ could contain both forward and backward steps, which means that these reactions could consume reactants that correspond to formal species. This has the

potential to cause a conflict if some reaction in τ_{n-m} also consumes that species. However, by Definition 17 we know that a formal species can only be consumed by a post-commit reaction if that reaction is a backward step, and this means that by the end of the subtrace $[r^{\alpha}_{n-m}, \ldots, r^{\alpha}_{n}]$, any formal species consumed by a backward step within $[r^{\alpha}_{n-m}, \ldots, r^{\alpha}_{n}]$ will eventually be re-generated by a subsequent reaction within $[r^{\alpha}_{n-m}, \ldots, r^{\alpha}_{n}]$. Thus, τ_{n-1} and $[r^{\alpha}_{n-1}, r^{\alpha}_{n}]$ can be swapped using the (swap) from Definition 13, to give

$$\mathbf{S} \vdash_{\mathcal{E}} \tau \rightsquigarrow [r^{\alpha}_{1}, \ldots, r^{\alpha}_{k-1}, r^{\alpha}_{com}] : \tau' : \tau_{k+1} : [r^{\alpha}_{k+1}] : \cdots : \tau_{n-2} : [r^{\alpha}_{n-2}, r^{\alpha}_{n-1}, r^{\alpha}_{n}] : \tau_{n-1} : \tau_{n} : \tau_{rest}.$$

Furthermore, by repeated application of this argument we get that

$$\mathbf{S} \vdash_{\mathcal{E}} \tau \rightsquigarrow [r^{\alpha}_{1}, \ldots, r^{\alpha}_{k-1}, r^{\alpha}_{com}, r^{\alpha}_{k+1}, \ldots, r^{\alpha}_{n}] : \tau'_{rest},$$

as required, where $\tau'_{rest} = \tau' : \tau_{k+1} : \cdots : \tau_{n} : \tau_{rest}$, i.e., $\tau'_{rest} = \tau'_{1} : \cdots : \tau'_{k} : \tau_{k+1} : \cdots : \tau_{n} : \tau_{rest}$.

We can now state and prove our main soundness theorem, which is valid for CRN encodings composed of reaction encodings that satisfy the criteria in Definition 17 and Definition 18. Since the set of all traces includes incomplete executions of reaction encodings, we require that any trace can be *extended* to produce a serializable execution. To express this we will write $^{pt}_{\mathcal{F}}(\mathbf{X}, \mathbf{F})$ for the set of "possible traces", that is, the set of non-empty formal traces that are valid from the formal state **X** and that can be emulated using the fuel **F**, i.e., $^{pt}_{\mathcal{F}}(\mathbf{X}, \mathbf{F}) \triangleq \{\tau \in \text{Traces}(\mathcal{F}) \mid \mathbf{X} \vdash_{\mathcal{F}} \tau \wedge \mathit{reqfuel}(\tau) \leq \mathbf{F} \wedge \tau \neq \varepsilon\}$. This is required so that we can assess whether any reaction encodings can be executed using the available species, and therefore we must know the reaction encodings at this stage. Then, the soundness theorem and its proof are as follows.

Theorem 5.7 (Soundness)—*Fix a formal CRN* $\mathcal{F}$ *with reactions* $a_1, \ldots, a_n$, *with corresponding reaction encodings into an encoding CRN* $\mathcal{E}$ *that are all transactional and pairwise compatible. Let* **X** *range over multisets of formal species, and let* **F** *range over multisets of fuels such that* $\mathbf{F} = k_1 \cdot \mathbf{F}_{a1} + \cdots + k_n \cdot \mathbf{F}_{an}$, *where* $k_1, \ldots, k_n \in \mathbb{N}$ *and* $k_1, \ldots, k_n \geq 0$. *Then, for all* $\tau \in \text{Traces}(\mathcal{E})$ *and all* **X** *and* **F** *such that* $(\mathcal{M}(\mathbf{X}) + \mathbf{F}) \vdash_{\mathcal{E}} \tau$, *either:*

- $^{pt}_{\mathcal{F}}(\mathbf{X}, \mathbf{F}) = \emptyset$ *and there exists* τ' *such that* $(\mathcal{M}(\mathbf{X}) + \mathbf{F}) \vdash_{\mathcal{E}} \tau : \tau' \rightsquigarrow \varepsilon$; *or*
- $^{pt}_{\mathcal{F}}(\mathbf{X}, \mathbf{F}) \neq \emptyset$ *and there exists* τ' *such that* $(\mathcal{M}(\mathbf{X}) + \mathbf{F}) \vdash_{\mathcal{E}} \tau : \tau' \rightsquigarrow \tau_{ser}$, *where* τ_{ser} *is a serial execution of some* $\tau_{formal} \in {}^{pt}_{\mathcal{F}}(\mathbf{X}, \mathbf{F})$.

Proof: We begin by noting that **F** can be expressed as integer multiples of the multisets of fuel species required to execute the various reaction encodings. Then, by Definition 18 there can be no direct interaction between species that do not appear in the same reaction encoding. Furthermore, the only species which are shared are those with respect to which the encodings are extra tolerant, and a single copy of a given reaction encoding must be extra tolerant with respect to any formal or fuel species that it uses. Thus, the trace τ must consist of an interleaving of reactions which all correspond to a valid state transition from

precisely one of the constituent reaction encodings. Furthermore, since there is only a finite supply of fuel, there can only be finitely many commit reactions in τ. However, τ could include infinite cycles of reversible reactions—we assume that τ has already been rewritten to eliminate these, using the rewrite rules from Definition 13. We proceed by mathematical induction on $\chi(\tau)$, the number of commit reactions in τ.

Base case: $\chi(\tau) = 0$.. Since there are no commit reactions, all reactions in τ must be reversible. Therefore, by Lemma 5.5 there exists a trace τ'' such that $(\mathcal{M}(\mathbf{X}) + \mathbf{F}) \vdash_{\varepsilon} \tau{:}\tau'' \rightsquigarrow \varepsilon$. Then, if $pt_{\mathcal{F}}(\mathbf{X}, \mathbf{F}) = \emptyset$ we can set $\tau' = \tau''$ to get that $(\mathcal{M}(\mathbf{X}) + \mathbf{F}) \vdash_{\varepsilon} \tau{:}\tau' \rightsquigarrow \varepsilon$, as required. On the other hand, if $pt_{\mathcal{F}}(\mathbf{X}, \mathbf{F}) \neq \emptyset$ then there must be a formal reaction a such that $\mathbf{X} \vdash_{\varepsilon} a$ and for which $reqfuel([a]) \leq \mathbf{F}$. Then, for a complete execution τ_a of $[\![a]\!]$, we can set $\tau' = \tau''{:}\tau_a$ to get $(\mathcal{M}(\mathbf{X}) + \mathbf{F}) \vdash_{\varepsilon} \tau{:}\tau' \rightsquigarrow \tau_a$, which is a serial execution of $[a] \in pt_{\mathcal{F}}(\mathbf{X}, \mathbf{F})$, as required.

Inductive case: $\chi(\tau) = n+1$.. Suppose that r^{α}_{com} is the first commit reaction in τ, belonging to the encoding of the formal reaction $a = (\mathbf{R}_a \rightarrow \mathbf{P}_a)$. By Definition 17, in order to pass the commit reaction we must consume all of the species from $\mathcal{M}(\mathbf{R}_a)$, and hence it follows that $\mathbf{R}_a \leq \mathbf{X}$. Furthermore, since we are only considering initial states with fuel to run complete encodings, we get that $[a] \in pt_{\mathcal{F}}(\mathbf{X}, \mathbf{F})$ and hence $pt_{\mathcal{F}}(\mathbf{X}, \mathbf{F}) \neq \emptyset$.

We proceed by identifying those reactions from τ which precede and follow r^{α}_{com} to make up a trace from $[\![a]\!]$. If τ does not contain a full execution of $[\![a]\!]$, let τ'_a denote a trace which completes an execution of $[\![a]\!]$. Since $[a] \in pt_{\mathcal{F}}(\mathbf{X}, \mathbf{F})$ we know that $(\mathscr{M}(\mathbf{X})+\mathbf{F})\vdash_{\mathscr{E}} r^{\alpha}_1$ holds, where r^{α}_1 is the first reaction of the execution of $[\![a]\!]$. Then, by Lemma 5.6 we get that $(\mathscr{M}(\mathbf{X})+\mathbf{F})\vdash_{\mathscr{E}} \tau{:}\tau'_a \rightsquigarrow \tau_\alpha{:}\tau_{rest}$, where τ_a is an execution of $[\![a]\!]$.

Let $\mathbf{X}'$ be a formal state such that $\mathbf{X} \xrightarrow{\alpha} \mathbf{X}'$. Then, we get $(\mathscr{M}(\mathbf{X})+\mathbf{F}) \xrightarrow{\tau_\alpha} (\mathscr{M}(\mathbf{X}')+(\mathbf{F}-\mathbf{F}_\alpha)+\mathbf{L}_\alpha)$, where $\mathbf{L}_a$ is the non-formal species which remain after the execution of $[\![a]\!]$ has completed. By Lemma 5.2 we can deduce that $(\mathcal{M}(\mathbf{X}') + (\mathbf{F} - \mathbf{F}_a) + \mathbf{L}_a) \vdash_{\varepsilon} \tau_{rest}$. All reaction encodings must be extra tolerant with respect to the species from $\mathbf{L}_a$, which means that the reactions in τ_{rest} may occur with or without the additional leftover species from $\mathbf{L}_a$. It follows that $(\mathcal{M}(\mathbf{X}') + (\mathbf{F} - \mathbf{F}_a)) \vdash_{\varepsilon} \tau_{rest}$, and we write $\mathbf{S}_a = \mathcal{M}(\mathbf{X}') + (\mathbf{F} - \mathbf{F}_a)$.

Since $\chi(\tau_{rest}) = n = \chi(\tau) - 1$, we can invoke our induction hypothesis on τ_{rest}. Now, we perform a case split on whether $pt_{\mathcal{F}}(\mathbf{X}', \mathbf{F} - \mathbf{F}_a)$ is empty. If $pt_{\mathcal{F}}(\mathbf{X}', \mathbf{F} - \mathbf{F}_a) = \emptyset$, by induction there exists τ'_b such that $\mathbf{S}_\alpha\vdash_{\mathscr{E}} \tau_{rest}{:}\tau'_b \rightsquigarrow \varepsilon$, and hence that $\mathbf{S}_\alpha+\mathbf{L}_\alpha\vdash_{\mathscr{E}} \tau_{rest}{:}\tau'_b \rightsquigarrow \varepsilon$. It follows that $(\mathscr{M}(\mathbf{X})+\mathbf{F})\vdash_{\mathscr{E}} \tau{:}(\tau'_a{:}\tau'_b) \rightsquigarrow \tau_\alpha$, and since τ_a is a serial execution of $[a] \in pt_{\mathcal{F}}(\mathbf{X}, \mathbf{F})$, we get the result.

If $pt_{\mathcal{F}}(\mathbf{X}', \mathbf{F} - \mathbf{F}_a) \neq \emptyset$, by induction there exists τ'_b such that $\mathbf{S}_\alpha\vdash_{\mathscr{E}} \tau_{rest}{:}\tau'_b \rightsquigarrow \tau_{ser}$, where τ_{ser} is a serial execution of some formal trace $\tau_{formal} \in pt_{\mathcal{F}}(\mathbf{X}', \mathbf{F} - \mathbf{F}_a)$. From this we get $\mathbf{S}_\alpha+\mathbf{L}_\alpha\vdash_{\mathscr{E}} \tau_{rest}{:}\tau'_b \rightsquigarrow \tau_{ser}$, and hence that $(\mathscr{M}(\mathbf{X})+\mathbf{F})\vdash_{\mathscr{E}} \tau{:}(\tau'_a{:}\tau'_b) \rightsquigarrow \tau_\alpha{:}\tau_{ser}$. Since τ_{formal}

$\in {}^{pt_{\mathcal{F}}}(\mathbf{X}', \mathbf{F} - \mathbf{F}_a)$ it follows that $[a]{:}\ \tau_{formal} \in {}^{pt_{\mathcal{F}}}(\mathbf{X}, \mathbf{F})$, and since $\tau_a{:}\tau_{ser}$ is a serial execution of $[a]{:}\tau_{formal}$, we get the result.

6. Unlimited fuel populations

The results proved in Section 4 and Section 5 were predicated on the size of a finite multiset of fuel species from the individual reaction encodings. These fuel species are consumed as the reaction encodings are executed, and the finite amount of fuel present initially imposed an upper bound on the number of encoded reactions that could be executed. However, in certain situations we may want to make the assumption that the populations of fuel species are unlimited. In particular, this simplification is valid if the actual fuel populations are much larger than the populations of the species that encode formal species, or if the fuel populations are being replenished from an external source. Furthermore, making this assumption allows us to study encodings of formal CRNs that admit infinite traces, such as the formal CRN $\mathcal{F}$ that contains the formal species *a*, *b*, and *c*, and the following three formal reactions.

$$\alpha_1{=}(a{+}b \rightarrow b{+}b) \quad \alpha_2{=}(b{+}c \rightarrow c{+}c) \quad \alpha_3{=}(c{+}a \rightarrow a{+}a)$$

This CRN implements a three-phase oscillator that we have studied as a DNA strand displacement system in previous work [15]. It is straightforward to show that $\{a = 2, b = 1, c = 1\} \vdash_{\mathcal{F}} [\alpha_1, \alpha_2, \alpha_3, \alpha_1, \alpha_2, \alpha_3, \ldots]$ holds, because we can derive the following reaction sequence, which returns to its starting state.

$$\{a{=}2, b{=}1, c{=}1\} \xrightarrow{\alpha_1} \{a{=}1, b{=}2, c{=}1\} \xrightarrow{\alpha_2} \{a{=}1, b{=}1, c{=}2\} \xrightarrow{\alpha_3} \{a{=}2, b{=}1, c{=}1\}$$

Thus, it is of interest to generalize our correctness results to the case with unlimited fuel populations.

In fact, it is straightforward to extend Theorem 4.2 and Theorem 5.7 to handle unlimited fuel populations. In the case of Theorem 4.2, we simply observe that any finite or infinite formal trace can be emulated using an unlimited amount of fuel, so Lemma 4.1 can can be generalized to handle infinite formal traces if the multiset **F** contains unlimited populations for the necessary fuel species. The case of Theorem 5.7 is slightly more involved because the proof presented above was obtained by mathematical induction on the number of commit reactions present in the encoding trace, which does not allow us to deduce anything about the case for infinite traces. However, we observe that, even for an infinite trace, we can still ascertain the order in which the commit reactions occur. Therefore, if we have unlimited fuel populations then we can repeatedly apply Theorem 5.7 to serialize finite prefixes of the infinite trace, since after the execution of each finite prefix we will still have unlimited populations of fuel left over. Thus, we straightforwardly obtain extensions of Theorem 4.2 and Theorem 5.7 that allow us to reason about infinite formal traces, provided that we also have unlimited fuel populations.

7. Examples

In this section we present examples of CRN encodings using abstract CRNs as well as several commonly-used frameworks based on DNA strand displacement. We will apply these to a well-known chemical algorithm for distributed consensus voting, known as the "approximate majority" algorithm.

7.1. The approximate majority system

In previous work [28], we studied encodings of the approximate majority (AM) distributed voting algorithm of [16]. We verified the implementation (for specific initial states) using probabilistic model checking [33, 34]. An experimental implementation of the approximate majority circuit has been demonstrated using DNA strand displacement reactions [35], and similar network structures and dynamics have been identified in cellular regulatory networks governing the cell cycle [36]. Therefore this particular CRN is of both theoretical and practical interest.

In this section we will present example applications of our modular verification strategy to various encodings of the approximate majority CRN. Our modular approach will allow us to infer correctness for arbitrarily large populations of species in the initial state, providing us with stronger guarantees that the circuit will function as intended than in our previous work on verification of DNA strand displacement circuits using probabilistic model checking [28]. We write $\mathcal{A} = (X_{\mathcal{A}}, R_{\mathcal{A}})$ for the approximate majority CRN, where:

$$X_{\mathcal{A}} = \{x, y, b\} \quad R_{\mathcal{A}} = \{(x+y \to y+b), (y+x \to x+b), (b+x \to x+x), (b+y \to y+y)\}.$$

The intuition behind the AM system is as follows. The initial state of the system consists entirely of species x and y, and the system is guaranteed to converge to a heterogeneous population of whichever species was initially in the majority, provided that the initial majority is large enough relative to the absolute population sizes [16]. To see why this happens, observe that the first two reactions (($x + y \to y + b$), ($y + x \to x + b$)) mean that, when x and y meet, one of them is converted to the intermediate species b, with equal probability of x and y being converted. Then, the final two reactions (($b + x \to x + x$), ($b + y \to y + y$)) mean that b is converted to x if it encounters x first, or to y if it encounters y first. In a well-mixed solution, which species b encounters is dependent on the relative population sizes of x and y in the solution. Hence, it is more likely that b will be recruited to whichever species is currently in the majority, which enables a small initial majority to be amplified to produce a heterogeneous state. Furthermore, once a heterogeneous state consisting entirely of x or y is achieved, no further reactions can take place.

7.2. An abstract CRN encoding of the approximate majority system

We now present an encoding of the approximate majority CRN $\mathcal{A}$, using abstract CRNs to encode the individual approximate majority reactions. We begin by fixing a species encoding $\mathcal{M}$ such that $\mathcal{M}(z) = \{\hat{z}\}$ for every formal species z, i.e., each formal species is encoded by precisely one species in the encoding CRN. Now, the abstract CRN encodings of the four approximate majority reactions are presented in Figure 1. We form the overall

encoding CRN $\mathcal{E}$ by combining the four individual encoding CRNs $\mathcal{E}_{a_1}$, $\mathcal{E}_{a_2}$, $\mathcal{E}_{a_3}$, $\mathcal{E}_{a_4}$, as described in Definition 11. To begin checking the correctness properties of this CRN encoding, we must first construct the state spaces for the individual reaction encodings: Figure 2 presents the state space for the encoding of the formal reaction a_1 from Figure 1.

Proposition 1 (Correctness of AM encoding using abstract CRN)—*The encoding of the AM system using an abstract CRN, as presented in* Figure 1 and Figure 2, is correct in the sense of Theorem 4.2 and Theorem 5.7.

Proof: We will only discuss the encoding of a_1 because all four encodings follow the same pattern. We begin by noting that the state space from Figure 2 satisfies the definition of a reaction encoding from Definition 10 because the initial and terminal states encode the formal reactant and product species respectively. Furthermore, the terminal state is unique and universally reachable, and the fuel multiset $\mathbf{F}_{a1}$ is unique and minimal, and the encoding CRN is stratified, as illustrated by the state space diagram from Figure 2.

To see that the encoding is transactional, we note that every terminal trace of the encoding passes through the same commit reaction (labeled Commit in Figure 2). Before the commit reaction, encoded formal species are only reactants of forward reactions and products of backward reactions, as required. Similarly, after the commit reaction, encoded formal species are only products of forward reactions and reactants of backward reactions, and when the commit reaction is executed, all formal reactant species have been consumed. It is straightforward to show that the encoding is extra tolerant with respect to the formal and fuel species, because all fuel species are used precisely once in each terminal trace. Furthermore, it is easy to see that the encoding is extra tolerant with respect to the “leftover” species in the terminal state, because they are either waste species or an intermediate species that can only be a reactant of a backwards reaction (therefore, the system has to undergo the corresponding forward reaction to reach the state from which the extra intermediate species can be consumed), and all pre-commit traces follow the same pathway.

Finally, we must check that the individual reaction encodings are pairwise compatible, in the sense of Definition 18. This is straightforward because the only shared species are the encoded formal species, and from the above arguments we know that the encodings are extra tolerant with respect to these. Furthermore, the fact that each reaction encoding is presented as a separate abstract CRN means that there is no way to derive cross-talk reactions between species that only occur in different reaction encodings. This is a limitation of encodings specified as abstract CRNs, which we will discuss further below.

Thus, we conclude that the abstract CRN encoding from Figure 1 is a correct encoding of the AM system, in the sense of Theorem 4.2 and Theorem 5.7.

7.3. A four-domain DNA strand displacement encoding of the approximate majority system

We now develop the abstract CRN encoding of the AM system from Section 7.2 into a form more suitable for a direct implementation in wet chemistry. Here, our chemistry of choice is DNA strand displacement [17, 18]. The basic mechanism of strand displacement reactions is illustrated in Figure 3. In particular, in this section we focus on the four-domain encoding

framework, which was previously introduced as a means of implementing arbitrary CRNs using just DNA reactions [26]. The four-domain encoding is so called because each formal species is represented by single strands of DNA whose sequence is broken down into four domains: an arbitrary "history" domain and three additional domains (two toeholds and a long domain) that encode the identity of the formal species. The species encoding a given formal species may have different history domains: the history domain simply records which reaction encoding generated the strand, and does not affect the subsequent reactions in which the strand may take part. Thus we will define the species encoding function to have the form

$$\mathscr{M}(x)=\{<\mathrm{h}\ \mathrm{tx1}\hat{}\ \mathrm{x}\ \mathrm{tx2}\hat{}>|\mathrm{h}\in H\}$$

where H is the finite set of history domains that may be associated with the encoded species x within our entire encoding, and tx1^ and tx2^ are toehold domains which, together with the long domain x, define which species is encoded by the strand. Hence, this encoding will fully exploit the fact that our species encodings represent each formal species as a *set* of encoding species (Definition 8).

We will represent the species from DNA strand displacement-based CRN encodings using the syntax of the DSD programming language [15, 37]. This provides a clear and concise representation for the structures of species involved in strand displacement reactions and a convenient operational semantics for automatically deriving the possible interactions between species. We will use the "Infinite" DSD semantics [15] throughout, in which we ignore all "unproductive" reactions where a strand binds to a complex via a complementary toehold but cannot complete a subsequent strand displacement reaction. We refer the reader to the above-referenced papers for the details of the syntax and semantics of the DSD language. The Visual DSD compiler can be used to generate the corresponding CRN for four-domain DNA strand displacement reaction encodings. Figure 4 presents DSD code that implements four-domain species encodings and a parameterized module that encodes of formal reactions of the form $x + y \rightarrow y + z$, with a specific instantiation to the reaction $x + y \rightarrow y + b$ from the AM system.

The goal of this four-domain encoding example is to provide a concrete biochemical implementation of the abstract CRN encoding presented in Section 7.2. The set of derived chemical reactions is presented in Figure 5, and the corresponding state space is presented in Figure 6. Here, the fuel species that correspond to the encoding of the $x + y \rightarrow y + b$ reaction are the two multi-strand complexes present in the initial state. We observe a direct mapping between the species and states from Figure 6 and those from Figure 2. Furthermore, the "private domains", declared using the new keyword in the DSD module definitions, will be instantiated differently for each use of the module, meaning that fuel, intermediate and waste species will not be shared between modules. Furthermore, this fact will prevent the long intermediate strand from interacting with any species from another module.

Proposition 2 (Correctness of AM encoding using four-domain strand displacement)—*The encoding of the AM system using four-domain strand displacement, as presented in* Figure 5 and Figure 6, is correct in the sense of Theorem 4.2 and Theorem 5.7.

Proof: We can deploy arguments similar to those from Proposition 1 to argue that this 4-domain strand displacement encoding of $x + y \rightarrow y + b$ is correct. Briefly, by inspection of Figure 6 we see that this encoding of $x + y \rightarrow y + b$ satisfies the definition of a reaction encoding because the initial and terminal states encode the correct reactants and products, the terminal state is unique and universally reachable, the fuel multiset is unique and minimal, and the CRN is stratified.

Furthermore, the reaction labeled Commit in Figure 6 is a commit reaction in the sense of Definition 17: all traces pass through it, and prior to the commit reaction the forward reactions consume formal species and the backward reactions release formal species, and the opposite is true after the commit reaction. (Note that, in this case, the commit reaction actually consumes the second formal reactant species.) Furthermore, all pre-commit traces follow the same pathway as there is just a single reversible reaction in the pre-commit phase of the state space, and it is straightforward to show that the encoding is extra tolerant with respect to all formal species, fuel species, and leftover species in the terminal state. Given that the state spaces for all four reactions from the formal AM CRN $\mathcal{A}$ follow a similar pattern, we can make similar arguments for all four formal reactions from $\mathcal{A}$. Even in the cases where the two products are the same species, the state space follows the same pattern because there will still be a single strand displacement reaction that simultaneously releases both product strands into solution by displacing across their history domains. Furthermore, the absence of species sharing due to the use of private history domains means that it is straightforward to show that the four encodings are pairwise compatible.

A closer inspection of the corresponding set of generated reactions, as presented in Figure 5, reveals that there is not a one-to-one correspondence with the reactions from Section 7.2, because an additional reaction appears in Figure 5. This additional reaction appears because the formal species y appears as both a reactant and a product of the underlying formal reaction $x + y \rightarrow y + b$, and in the four-domain encoding the reactant and product versions of the y species have different history domains. While the $\mathcal{M}$ encoding defined above considers these to represent the same formal species, in the DSD programming language they are distinct species. Therefore, when using this encoding our definition of the encoding CRNs for the four AM reactions must take account of all possible variants on the species encoding strands, and we must ensure that the reaction encoding behaves correctly for all possible formal reactant species.

To see that the compiled CRN from Figure 5 is agnostic to the different variants of the strands that encode the reactants, it suffices to observe that the history domains of the incoming strands are never involved in the strand displacement reactions that drive the execution of the encoding forward. It is true that the output product strands must be displaced via their history domains, but the invader strand for this reaction is a fuel strand that is part of the reaction encoding and can therefore be matched to the history domains of

the output strands, which we are at liberty to fix on a per-encoding basis. Therefore, reactions similar to those from Figure 5 could be derived for incoming reactant strands with *any* history domains. Thus, we can extend the CRN from Figure 5 by including additional reactions and intermediate species for all of the species variants required for a full encoding of the four-reaction AM system.

Therefore, we can deduce that a system containing four encodings of the individual AM reactions based on the above template encodes the AM system correctly, in the sense of Theorem 4.2 and Theorem 5.7.

7.4. A two-domain DNA strand displacement encoding of the approximate majority system

We now present an alternative strand displacement encoding of the approximate majority system, this time using the two-domain approach developed by Cardelli [27]. The two-domain encoding has a number of favorable characteristics for experimental implementation: *(i)* the single-stranded species are short, typically consisting of just a toehold domain and a long domain, which minimizes the possibility of secondary structure formation, and *(ii)* the complexes that make up the reaction encodings have simple structures with no overhanging strands. The latter property means that complexes can be manufactured in bulk in bacteria, which produces higher-quality strands than is currently available using solid-state synthesis technology [35]. This approach was used in an experimental implementation of the approximate majority system [35]. While the simplicity of two-domain structures is enticing from a practical perspective, the design necessitates additional intermediate steps in the reaction encoding. Hence, it may be less obvious that the design is correct. In previous work we have explored the use of probabilistic model checking for the verification of two-domain strand displacement systems [28].

In the two-domain encoding, all strands that encode a given formal species are identical. Hence, the species encoding function $\mathcal{M}$ is such that $\mathcal{M}(x) = \{< t^ x >\}$ for every formal species x. Following previous work [28, 27], we also extend the basic two-domain syntax with extended strands to enable irreversible product release, by including extended strands of the form <t^ x y> and <x y t^>. The DSD code in Figure 7 presents two-domain species encodings, and a parameterized encoding of formal reactions of the form $x + y \rightarrow y + z$.

In previous work [28], we used the above catalyst encoding design to implement the approximate majority system, and we verified the implementation (for specific initial states) using probabilistic model checking [33, 34]. In the two-domain case, the species encoding is simpler but the execution of the reaction encoding involves more individual steps. Figure 8 presents all reactions and species derived from the two-domain encoding of $x + y \rightarrow y + b$, and the corresponding state space is presented in Figure 9. This visualization of the state space shows that the encoding has appropriate initial and terminal states, that the terminal state is universally reachable, and that every terminal trace from the initial state has a commit reaction (in fact, this is the same reaction in all cases) which satisfies the criteria from Definition 17.

Proposition 3 (Correctness of AM encoding using two-domain strand displacement)—*The encoding of the AM system using two-domain strand displacement,*

as presented in Figure 8 and Figure 9, is correct in the sense of Theorem 4.2 and Theorem 5.7.

Proof: The reaction module from Figure 7 can be instantiated to produce encodings of all four reactions from the AM system, and analysed similarly. For reasons of space, we do not present the full details for all four encodings, though it is important to note that the reaction encodings must work correctly in both the catalytic case when the two products are different species (i.e., reactions $x + y \rightarrow y + b$ and $y + x \rightarrow x + b$) and the autocatalytic case when the two products are the same species (i.e., reactions $b + x \rightarrow x + x$ and $b + y \rightarrow y + y$). By similar arguments to those presented above we conclude that all four encodings satisfy Definition 17.

It remains to show that the four reaction encodings are pairwise compatible in the sense of Definition 18. It is not hard to check that the only shared species between these encodings are waste strands, strands which correspond to formal species and certain intermediate strands that are the cosignals of formal species strands. We know that the encodings are extra tolerant with respect to the waste strands, hence we must show that the encodings are extra tolerant with respect to the remaining shared strands. We can use the DSD compiler to achieve this by (separately) adding an extra copy of each of these species to the starting state of the relevant reaction encodings and verifying that the state space is identical in each case (modulo the additional copies of those species in every state). Adding a single copy of each suffices because none of the reactions from Figure 8 (or any other two-domain strand displacement reactions) involve more than one copy of any reactant, so just one additional copy will be enough to reveal any additional reactions enabled by extra copies of this species. Although Definition 11 specified that the CRNs from individual reaction encodings are simply composed to produce the overall CRN encoding, without the possibility of any additional crosstalk reactions, we can actually use the DSD compiler to verify that no crosstalk is possible by checking all pairs of non-shared species from different encodings.

Hence, we obtain that the two-domain encoding in this section correctly encodes the AM system in the sense of Theorem 4.2 and Theorem 5.7, i.e., any trace generated by these four reaction encodings can be rewritten to produce a serial trace which corresponds to a valid execution of the formal reactions from the AM system.

7.5. Comparison of DNA strand displacement encodings of the approximate majority system

In this section, we have presented three different encodings of the approximate majority system: one based on abstract CRN species, one using four-domain DNA strand displacement reactions, and one using two-domain DNA strand displacement. In each case, we have demonstrated that the encoding satisfies the criteria imposed by our correctness theorems, and therefore we conclude that all three faithfully encode the formal approximate majority CRN $\mathcal{A}$. We conclude this section by noting that these results also allow us to draw conclusions about the relationships of the various encodings to each other. If two encodings are sound and complete with regard to the formal CRN, then any and all traces from one encoding can be serialized such that a similar serial trace can be derived from the other

encoding, and vice versa. Thus the two encodings may be considered as equivalent to one another.

8. Discussion

Chemical reaction networks have been shown to be a convenient means of specifying the desired behavior of synthetic biochemical systems based on DNA strand displacement [38, 12, 26, 35]. Therefore, verification of chemical reaction network encodings is an important area of future research. In previous work, we pioneered the application of probabilistic model checking [33, 34] and its application to probabilistic verification of two-domain DNA strand displacement circuits [28] and to DNA stack machine designs [11, 39].

Here, we have shown that any CRN encoding composed of individual reaction encodings that meet the criteria from Definition 17 and Definition 18 is correct in the sense that all traces can be rewritten using the rules from Definition 13 into a serialized trace in which each execution of a reaction encoding runs to completion before the next one starts. This notion of correctness is reasonable because reaction encodings are intended to encode a single rewriting step in the formal reactions, and if a trace cannot be rewritten in this way there must be a concurrency bug in the reaction encodings that allows them to produce a trace unrelated to any trace of the underlying formal reactions.

This work is a generalization of our previous conference paper [29]. The current paper provides full details of proofs and explicitly defines reaction encodings in terms of arbitrary CRNs as opposed to specifically strand displacement reaction systems. Furthermore, we have generalized our previous work to enable the definition of species encodings in which each formal species is represented by a structurally similar family of encoding species, such as the four-domain species encoding using history domains [26]. To our knowledge, this work is the first modular analysis of chemical reaction network encodings.

It is interesting that the correctness criteria from Definition 17 share much in common with other notions from existing concurrency theory, such as *two-phase locking* [14], in which each transaction has an initial phase of lock acquisition where exclusive access is obtained to the necessary resources, followed by a phase of lock release where those access rights are gradually relinquished. In our case, the first phase consumes the inputs and the second phase produces the outputs.

In Definition 17 and Definition 18 we aimed to allow the maximum possible sharing of species between different reaction encodings without invalidating the soundness result in Theorem 5.7. However, it is worth noting that certain published designs for strand displacement systems that implement chemical reaction networks fall foul of our restrictions on the structure of reaction encodings and on the sharing of species between encodings. For example, the two-domain reaction encoding designs from [27] without irreversible product release do not involve a commit reaction as defined in Definition 17, and therefore the soundness theorem does not apply to these encodings. Designs such as this that do not meet our criteria may be investigated further, to see if they are indeed flawed or if there may be a weaker version of our conditions that they do satisfy. Analyses such as ours are particularly

important for low copy-number systems, such as surface-tethered strand displacement networks that have recently been proposed [40, 41] and implemented in the laboratory [42].

Furthermore, certain combinations of two-domain strand displacement reaction encodings may violate our requirement that shared species must fall into the same category in all reaction encodings. In some designs from [27], it is possible for certain global cosignals to serve as an intermediate in one reaction encoding and as a fuel in another, which could adversely affect the kinetics of the reactions producing that strand as an intermediate if an excess quantity of that strand is supplied as fuel. This subtle point could be addressed in future two-domain reaction encoding designs.

In this work we used arbitrary CRNs to specify individual reaction encodings, which were combined to produce a CRN that encodes the formal CRN. For maximum generality we assumed that the CRNs involved abstract species without any internal structure. Therefore, the model considered in this paper is more abstract than in our previous work [29], where we restricted ourselves to verifying concrete encodings using DNA strand displacement reactions. It is worth noting, however, that our Definition 18 still includes the requirement that species can only interact if they appear in the same reaction encoding, so that the same formalism can be applied to both the case of an abstract CRN and the case where a more detailed semantics of individual species interactions is available. Thus, if we do have additional information on potential interactions between species from different modules, we can use this information to check for cross-talk reactions that may invalidate our correctness guarantees. An example of this approach is to use automated enumeration of reactions in a given domain, as exemplified by the DSD programming language and compiler [15] in the case of strand displacement systems.

8.1. Related Work

Previous work on the verification of chemical reaction network implementations based on pathway decomposition [32] or bisimulation-based approaches [43] is clearly related to this work.

Our definition of commit reactions in the context of transactional reaction encodings (Definition 17) is closely related to the definition of “regular” pathways from [32]. It is not *quite* the same notion, as the definition of “regular” pathways from [32] requires just that the sequence of states in the pathway can be partitioned such that the formal species in the first subsequence are all the formal reactants and the formal species in the second subsequence are all formal products. In Definition 17, we also require that the formal reactants (before the commit reaction) are consumed by “forward” reactions and produced by “backward” reactions and, similarly, that the formal products (after the commit reaction) are produced by “forward” reactions and consumed by “backward” reactions. (This fact is used at various points in the proof of Lemma 5.6 to justify applications of the trace rewriting rules.) Our more restrictive definition excludes reaction encodings that, for example, generate additional copies of their reactants during the initial phase that are later consumed, which would be a somewhat unnatural way to encode a formal reaction and, to our knowledge, is not seen in any proposed encoding framework. As an example, consider an encoding of the formal reaction $x + y \rightarrow z$ in which the formal species x, y, and z are represented by encoding

species x, $\hat{y}$, and $\hat{z}$ respectively, and where the following reactions encode the formal reaction using a fuel species f_1 and intermediate species i_1 and i_2:

$$\{\hat{x}+\hat{y}+f_1 \rightarrow i_1, i_1 \rightarrow \hat{x}+\hat{y}+f_1, i_1 \rightarrow i_2+\hat{x}, i_2+\hat{x} \rightarrow i_1, i_2+\hat{x} \rightarrow \hat{z}\}$$

Here, the first and second reactions are the two directions of a reversible reaction, as are the third and fourth reactions. The final reaction is the commit reaction, which produces the encoded product species $\hat{z}$. The key point here is that, while the first reaction consumes both reactant species x and $\hat{y}$, another copy of x is subsequently re-emitted and then consumed by the commit reaction. This is not permitted by our definitions, as a pre-commit "forward" reaction (the third reaction above) is releasing a formal species. Thus, our correctness results do not apply to this candidate encoding. However, this would be considered a valid encoding by [32], because all states before the commit reaction are a subset of the products, and the pathway cannot be decomposed at the point where x is re-emitted because an intermediate species is also present. The definition of encodings from [32] furthermore deliberately excludes "fuel" and "waste" species from encodings: the former because they are assumed to be present in excess. In our approach, finite quantities of "fuel" species are explicitly accounted for in the formalism and, as an extension, we can also handle the case where fuel populations are present in such excess as to be considered unlimited (Section 6).

Furthermore, there is clearly a strong connection between our approach and the approach of [43]. In Section 2.2 of [43], proving equivalence in terms of trajectories is akin to proving our completeness result. Similarly, in the definition of the "three conditions of interpretation" from Section 2.2 of [43], we require the "atomic condition" to hold by our definition of species mappings, and proving the "permissive condition" and the "delimiting condition" is akin to proving our completeness result. The notion of equivalence based on weak bisimulation from Section 2.2 of [43] is closely related to our notion of transactional reaction encodings, where the silent ("tau") reactions correspond to those on either side of the commit reaction in our definition, and the non-silent reaction corresponds to the commit reaction. The approach to equivalence taken in [43], in terms of interpreting potentially any encoding species as representing some formal species, is somewhat different from our approach where we only consider a well-defined class of encoding species as potentially representing formal species. Our approach is perhaps more practically oriented, as the encoding species that we consider to represent formal species are those that would be monitored directly in an experimental system, e.g., by adding reporter gates for those strands. It would be a challenging proposition to monitor the populations of all the various intermediate species that could be interpreted as a given formal species in the approach from [43]. The above example that is disallowed by our definitions would also be considered a valid encoding in the context of [43], via the interpretation that maps x to $\{x\}$, $\hat{y}$ to $\{y\}$, $\hat{z}$ to $\{z\}$, f_1 to the empty set, i_1 to $\{x, y\}$, and i_2 to $\{y\}$.

The key distinction of our work relative to [32] and [43] is that we can verify individual reaction encodings in isolation and infer correctness for any system constructed solely of verified encodings, subject to restrictions on sharing of species between the encodings. In the case of [32], verifying a particular CRN requires running an exponential-time algorithm

to enumerate the formal basis. In [43], the problem of finding a valid interpretation of species in the encoding CRN that serves as a witness to the correctness of the encoding "has no known polynomial time solution". In our approach, once each reaction encoding has been individually verified, the additional verification required to prove correctness of an arbitrary CRN implemented using these encodings scales quadratically with the number of different encodings in the system (because pairwise sharing between all encodings must be checked), and is independent of the initial species populations or the size of the full state space of the system. This is a significant improvement on standard model checking approaches in which the cost of verification scales with the size of the state space, which typically grows exponentially as the initial species populations increase.

We conjecture that our notion of equivalence between a formal CRN and its encoding CRN (which has been constructed in a modular manner, as described above) is a strict subset of the notions of equivalence presented in [32] and [43]. That is, if, under our definitions, an encoding CRN correctly implements a formal CRN, then the two CRNs are equivalent under the "pathway decomposition" approach of [32] and also under the weak bisimulation-based approach of [43]. Formally proving this conjecture is a matter for further study, but we do not believe that the reverse implications would hold, for reasons outlined above: our definition of reaction encodings places restrictions on the behavior of encoded formal species that are not present in either [32] or [43]. Nevertheless, while there is clearly a close relationship between the various notions of CRN equivalence, we believe that they are all worthy of further study as the various presentations shed light on different aspects of the encoding problem.

To address the issues with model checking mentioned above, recent work has employed SMT solvers that provide more compact representations of the state space, so that larger-scale DNA strand displacement systems with more copies of the initial species can be verified [44]. Probabilistic model checking has also been used to study molecular walkers [45], which is a promising application domain because molecular walkers are either single-molecule or low copy number systems, which limits the size of the state space and makes checking of the entire state space more feasible. In other related work, Cardelli and Laneve [46, 47] developed a theory of reversible computational structures with a strong relationship to DNA strand displacement reaction encodings. However, that work did not take the initial state of a computation into account and was therefore not capable of distinguishing between traces where reactions involving the same species could be safely permuted. Existing work on reachability in CRNs [48] was an inspiration to our work, in particular the notion of copy tolerance from that paper.

8.2. Future Work

We have demonstrated that the restrictions we imposed on reaction encodings are *sufficient* to obtain a serializability result. Another important future research direction will be to determine which restrictions are *necessary* to derive such a result. We conjecture that the restrictions presented in this paper are both necessary and sufficient, though proving this would be a question for future work. Given the delicacy of some of these proofs, we believe that it will become important to mechanize the metatheory of chemical reaction networks, as

has been done for much metatheory in the area of programming language semantics [49], so that results such as this can be formally verified using computer-aided theorem proving techniques.

Our modular approach provides a path to verification of *module definitions*, for example, checking that a definition which maps arbitrary species w, x, y, and z to the corresponding reaction encoding $[\![w + x \rightarrow y + z]\!]$ produces a correct reaction encoding for *any* values of w, x, y, and z, some of which might in fact represent the same formal species. This notion of modular molecular programming is already embodied in tools such as the DSD domain-specific programming language [37, 15]. Note that to verify the approximate majority circuit encodings in Section 7 we had to verify four specific instantiations of the catalyst encoding module. In future work, we plan to express our correctness criteria in a temporal logic, so that module definitions can be checked automatically using a model checker, allowing us to cover all possible input patterns. This would potentially enable us to verify a set of strand displacement primitives sufficient to encode arbitrary CRNs consisting of unimolecular and bimolecular reactions, from which it would follow that any CRN encoded using these encodings would be correct in the sense of the definitions from this paper. This would provide a formally verified basis for implementing arbitrary CRNs using strand displacement networks that are correct by construction.

Finally, we note that our formalism and proofs do not take account of reaction rates. Expressing correctness in terms of reachability of states is both important and natural from a computer science perspective. However, unfavourable kinetics might cause CRN encodings that satisfy our reachability criteria to function poorly in practice, as discussed above in the case of certain two-domain strand displacement systems. Furthermore, certain designs that fail to satisfy the criteria might function acceptably in practice due to favourable kinetics, as exemplified by the “wisdom of crowds” example [28, 27]. Proving soundness of CRN encodings is already challenging without considering reaction rates, and indeed it is not clear how such a correctness result would be formulated in a modular setting when considering reaction rates. Previous work [26] presented proofs for a particular encoding of chemical reaction networks using four-domain DNA strand displacement, and future extensions of our work may enable such results to be proved for arbitrary reaction encodings in a modular way. For instance, it may be possible to relate the expected time to fully execute a reaction encoding to the rate of the corresponding formal reaction, either by solving the corresponding continuous-time Markov chain analytically or by using a probabilistic model checker such as prism [34]. However, such efforts would be complicated by the fact that the output species from a reaction encoding are typically released gradually, some time before the final irreversible reaction that concludes the execution of the encoding. Hence, it is not obvious which point in time should be considered as the end of the execution of the encoding for the purposes of proving results about the kinetics.

Acknowledgments

This material is based upon work supported by the National Science Foundation under grants 1027877, 1028238 and 1318833, and M.R.L. gratefully acknowledges support from the New Mexico Cancer Nanoscience and Microsystems Training Center (NIH/NCI grant 5R25CA153825).

Bibliography

1. Zhang DY, Seelig G. Dynamic DNA nanotechnology using strand-displacement reactions. Nature Chemistry. 2011; 3(2):103–113. DOI: 10.1038/nchem.957
2. Stojanovic MN, Stefanovic D. A deoxyribozyme-based molecular automaton. Nature Biotechnology. 2003; 21(9):1069–1074.
3. Kim J, White KS, Winfree E. Construction of an *in vitro* bistable circuit from synthetic transcriptional switches. Molecular Systems Biology. 2(1)
4. Fujii T, Rondelez Y. Predator-prey molecular ecosystems. ACS Nano. 2012; 7(1):27–34. [PubMed: 23176248]
5. Winfree E, Liu F, Wenzler LA, Seeman NC. Design and self-assembly of two-dimensional DNA crystals. Nature. 1998; 394:539–544. [PubMed: 9707114]
6. Rothemund PWK. Folding DNA to create nanoscale shapes and patterns. Nature. 2006; 440:297–302. [PubMed: 16541064]
7. Zhang DY, Hariadi RF, Choi HM, Winfree E. Integrating DNA strand-displacement circuitry with DNA tile self-assembly. Nature Communications. 2013; 4:1965.
8. Zhang F, Nangreave J, Liu Y, Yan H. Reconfigurable DNA origami to generate quasifractal patterns. Nano Letters. 2012; 12:3290–3295. [PubMed: 22559073]
9. Wei B, Ong LL, Chen J, Jaffe AS, Yin P. Complex reconfiguration of DNA nanostructures. Angewandte Chemie International Edition. 2014; 53:7475–7479.
10. Doty, D.; Lutz, JH.; Patitz, MJ.; Schweller, RT.; Summers, SM.; Woods, D. The tile assembly model is intrinsically universal. Proceedings of the 53rd Annual IEEE Symposium on Foundations of Computer Science; 2012.
11. Qian, L.; Soloveichik, D.; Winfree, E. Efficient Turing-universal computation with DNA polymers. In: Sakakibara, Y.; Mi, Y., editors. Proceedings of DNA16, Vol. 6518 of Lecture Notes in Computer Science. Springer-Verlag; 2011. p. 123-140.
12. Cook, M.; Soloveichik, D.; Winfree, E.; Bruck, J. Programmability of chemical reaction networks. In: Condon, A.; Harel, D.; Kok, JN.; Salomaa, A.; Winfree, E., editors. Algorithmic Bioprocesses. Springer-Verlag; 2009. p. 543-584.
13. Chen, H-L.; Doty, D.; Soloveichik, D. Deterministic function computation with chemical reaction networks. In: Ste-fanovic, D.; Turberfield, A., editors. Proceedings of DNA18, Vol. 7433 of Lecture Notes in Computer Science. Springer-Verlag; 2012. p. 25-42.
14. Papadimitriou CH. The serializability of concurrent database updates. Journal of the ACM. 1979; 26(4):631–653.
15. Lakin MR, Youssef S, Cardelli L, Phillips A. Abstractions for DNA circuit design. Journal of the Royal Society Interface. 2012; 9(68):470–486.
16. Angluin D, Aspnes J, Eisenstat D. A simple population protocol for fast robust approximate majority. Distributed Computing. 2008; 21(2):87–102.
17. Yurke B, Turberfield AJ, APM, Simmel FC, Neumann JL. A DNA-fuelled molecular machine made of DNA. Nature. 2000; 406:605–608. [PubMed: 10949296]
18. Zhang DY, Seelig G. Dynamic DNA nanotechnology using strand-displacement reactions. Nature Chemistry. 2011; 3:103–113.
19. Seelig G, Soloveichik D, Zhang DY, Winfree E. Enzyme-free nucleic acid logic circuits. Science. 2006; 314:1585–1588. [PubMed: 17158324]
20. Qian L, Winfree E. Scaling up digital circuit computation with DNA strand displacement cascades. Science. 2011; 332:1196–1201. [PubMed: 21636773]
21. Qian L, Winfree E, Bruck J. Neural network computation with DNA strand displacement cascades. Nature. 2011; 475:368–372. [PubMed: 21776082]
22. Yin P, Choi HMT, Calvert CR, Pierce NA. Programming biomolecular self-assembly pathways. Nature. 2008; 451:318–322. [PubMed: 18202654]
23. Yurke B, APM. Using DNA to power nanostructures. Genetic Programming and Evolvable Machines. 2003; 4:111–122.

24. Muscat RA, Bath J, Turberfield AJ. A programmable molecular robot. Nano Letters. 2011; 11(3): 982–987. [PubMed: 21275404]
25. Muscat, RA.; Strauss, K.; Ceze, L.; Seelig, G. DNA-based molecular architecture with spatially localized components. Proceedings of ISCA '13; 2013.
26. Soloveichik D, Seelig G, Winfree E. DNA as a universal substrate for chemical kinetics. Proceedings of the National Academy of Sciences of the United States of America. 2010; 107(12): 5393–5398. [PubMed: 20203007]
27. Cardelli L. Two-domain DNA strand displacement. Mathematical Structures in Computer Science. 2013; 23:247–271.
28. Lakin MR, Parker D, Cardelli L, Kwiatkowska M, Phillips A. Design and analysis of DNA strand displacement devices using probabilistic model checking. Journal of the Royal Society Interface. 2012; 9(72):1470–1485.
29. Lakin, MR.; Phillips, A.; Stefanovic, D. Modular verification of DNA strand displacement networks via serializability analysis. In: Soloveichik, D.; Yurke, B., editors. Proceedings of DNA19, Vol. 8141 of Lecture Notes in Computer Science. Springer-Verlag; 2013. p. 133-146.
30. Baader, F.; Nipkow, T. Term Rewriting And All That. Cambridge University Press; 1998.
31. Zhang DY, Winfree E. Control of DNA strand displacement kinetics using toehold exchange. Journal of the American Chemical Society. 2009; 131(47):17303–17314. DOI: 10.1021/ja906987s [PubMed: 19894722]
32. Shin, SW. Master's thesis. California Institute of Technology; 2012. Compiling and verifying DNA-based chemical reaction network implementations.
33. Heath J, Kwiatkowska M, Norman G, Parker D, Tymchyshyn O. Probabilistic model checking of complex biological pathways. Theoretical Computer Science. 2008; 391:239–257.
34. Kwiatkowska, M.; Norman, G.; Parker, D. Proceedings of CAV 2011, Vol. 6806 of Lecture Notes in Computer Science. Springer-Verlag; 2011. prism 4.0: verification of probabilistic real-time systems; p. 585-591.
35. Chen Y-J, Dalchau N, Srinivas N, Phillips A, Cardelli L, Soloveichik D, Seelig G. Programmable chemical controllers made from DNA. Nature Nanotechnology. 2013; 8:755–762.
36. Cardelli L, Csikász-Nagy A. The cell cycle switch computes approximate majority. Scientific Reports. 2012; 2:656. [PubMed: 22977731]
37. Lakin MR, Youssef S, Polo F, Emmott S, Phillips A. Visual DSD: a design and analysis tool for DNA strand displacement systems. Bioinformatics. 2011; 27(22):3211–3213. [PubMed: 21984756]
38. Soloveichik D, Cook M, Winfree E, Bruck J. Computation with finite stochastic chemical reaction networks. Natural Computing. 2008; 7:615–633.
39. Lakin, MR.; Phillips, A. Modelling, simulating and verifying Turing-powerful strand displacement systems. In: Cardelli, L.; Shih, W., editors. Proceedings of DNA17, Vol. 6937 of Lecture Notes in Computer Science. Springer-Verlag; 2011. p. 130-144.
40. Chandran, H.; Gopalkrishnan, N.; Phillips, A.; Reif, J. Localized hybridization circuits. In: Cardelli, L.; Shih, W., editors. Proceedings of DNA17, Vol. 6937 of Lecture Notes in Computer Science. Springer-Verlag; 2011. p. 64-83.
41. Qian, L.; Winfree, E. Parallel and scalable computation and spatial dynamics with DNA-based chemical reaction networks on a surface. In: Murata, S.; Kobayashi, S., editors. Proceedings of DNA20, Vol. 8727 of Lecture Notes in Computer Science. Springer-Verlag; 2014. p. 114-131.
42. Teichmann M, Kopperger E, Simmel FC. Robustness of localized DNA strand displacement cascades. ACS Nano. 2014; 8(8):8487–8496. [PubMed: 25089925]
43. Dong, Q. Master's thesis. Stony Brook University; 2012. A bisimulation approach to verification of molecular implementations of formal chemical reaction networks.
44. Yordanov, B.; Wintersteiger, C.; Hamadi, Y.; Phillips, A.; Kugler, H. Functional analysis of large-scale DNA strand displacement circuits. In: Soloveichik, D.; Yurke, B., editors. Proceedings of DNA19, Vol. 8141 of Lecture Notes in Computer Science. Springer-Verlag; 2013. p. 189-203.
45. Dannenberg, F.; Kwiatkowska, M.; Thachuk, C.; Turberfield, AJ. DNA walker circuits: Computational potential, design, and verification. In: Soloveichik, D.; Yurke, B., editors.

Proceedings of DNA19, Vol. 8141 of Lecture Notes in Computer Science. Springer-Verlag; 2013. p. 31-45.

46. Cardelli, L.; Laneve, C. In: Fages, F., editor. Reversible structures; Proceedings of CMSB; 2011; ACM; 2011. p. 131-140.
47. Cardelli L, Laneve C. Reversibility in massive concurrent systems. Scientific Annals of Computer Science. 2011; 21(2):175–198.
48. Condon, A.; Kirkpatrick, B.; Manŭch, J. Reachability bounds for chemical reaction networks and strand displacement systems. In: Stefanovic, D.; Turberfield, A., editors. Proceedings of DNA18, Vol. 7433 of Lecture Notes in Computer Science. Springer-Verlag; 2012. p. 43-57.
49. Aydemir, BE.; Bohannon, A.; Fairbairn, M.; Foster, JN.; Pierce, BC.; Sewell, P.; Vytiniotis, D.; Washburn, G.; Weirich, S.; Zdancewic, S. Mechanized metatheory for the masses: The PoplMark challenge. In: Hurd, J.; Melham, TF., editors. Proceedings of TPHOLs 2005, Vol. 3603 of Lecture Notes in Computer Science. Springer-Verlag; 2005. p. 50-65.

- Encoding of $\alpha_1 = (x + y \rightarrow y + b)$ is $[\![\alpha_1]\!] = (\mathcal{E}_{\alpha_1}, \mathbf{F}_{\alpha_1})$, where $\mathcal{E}_{\alpha_1} = (X_{\mathcal{E}_{\alpha_1}}, R_{\mathcal{E}_{\alpha_1}})$ and where
  - $X_{\mathcal{E}_{\alpha_1}} = \{\widehat{x}, \widehat{y}, \widehat{b}, f_1^1, f_1^2, i_1^1, i_1^2, i_1^3, w_1^1, w_1^2\}$,
  - $R_{\mathcal{E}_{\alpha_1}} = \{(\widehat{x} + f_1^1 \rightarrow i_1^1 + i_1^2), (i_1^1 + i_1^2 \rightarrow \widehat{x} + f_1^1), (\widehat{y} + i_1^1 \rightarrow i_1^3 + w_1^1), (i_1^3 + f_1^2 \rightarrow \widehat{y} + \widehat{b} + w_1^2)\}$,
  - $\mathbf{F}_{\alpha_1} = \{f_1^1 = 1, f_1^2 = 1\}$.
- Encoding of $\alpha_2 = (y + x \rightarrow x + b)$ is $[\![\alpha_2]\!] = (\mathcal{E}_{\alpha_2}, \mathbf{F}_{\alpha_2})$, where $\mathcal{E}_{\alpha_2} = (X_{\mathcal{E}_{\alpha_2}}, R_{\mathcal{E}_{\alpha_2}})$ and where
  - $X_{\mathcal{E}_{\alpha_2}} = \{\widehat{x}, \widehat{y}, \widehat{b}, f_2^1, f_2^2, i_2^1, i_2^2, i_2^3, w_2^1, w_2^2\}$,
  - $R_{\mathcal{E}_{\alpha_2}} = \{(\widehat{y} + f_2^1 \rightarrow i_2^1 + i_2^2), (i_2^1 + i_2^2 \rightarrow \widehat{y} + f_2^1), (\widehat{x} + i_2^1 \rightarrow i_2^3 + w_2^1), (i_2^3 + f_2^2 \rightarrow \widehat{x} + \widehat{b} + w_2^2)\}$,
  - $\mathbf{F}_{\alpha_2} = \{f_2^1 = 1, f_2^2 = 1\}$.
- Encoding of $\alpha_3 = (b + x \rightarrow x + x)$ is $[\![\alpha_3]\!] = (\mathcal{E}_{\alpha_3}, \mathbf{F}_{\alpha_3})$, where $\mathcal{E}_{\alpha_3} = (X_{\mathcal{E}_{\alpha_3}}, R_{\mathcal{E}_{\alpha_3}})$ and where
  - $X_{\mathcal{E}_{\alpha_3}} = \{\widehat{x}, \widehat{b}, f_3^1, f_3^2, i_3^1, i_3^2, i_3^3, w_3^1, w_3^2\}$,
  - $R_{\mathcal{E}_{\alpha_3}} = \{(\widehat{b} + f_3^1 \rightarrow i_3^1 + i_3^2), (i_3^1 + i_3^2 \rightarrow \widehat{b} + f_3^1), (\widehat{x} + i_3^1 \rightarrow i_3^3 + w_3^1), (i_3^3 + f_3^2 \rightarrow \widehat{x} + \widehat{x} + w_3^2)\}$,
  - $\mathbf{F}_{\alpha_3} = \{f_3^1 = 1, f_3^2 = 1\}$.
- Encoding of $\alpha_4 = (b + y \rightarrow y + y)$ is $[\![\alpha_4]\!] = (\mathcal{E}_{\alpha_4}, \mathbf{F}_{\alpha_4})$, where $\mathcal{E}_{\alpha_4} = (X_{\mathcal{E}_{\alpha_4}}, R_{\mathcal{E}_{\alpha_4}})$ and where
  - $X_{\mathcal{E}_{\alpha_4}} = \{\widehat{y}, \widehat{b}, f_4^1, f_4^2, i_4^1, i_4^2, i_4^3, w_4^1, w_4^2\}$,
  - $X_{\mathcal{E}_{\alpha_4}} = \{(\widehat{b} + f_4^1 \rightarrow i_4^1 + i_4^2), (i_4^1 + i_4^2 \rightarrow \widehat{b} + f_4^1), (\widehat{y} + i_4^1 \rightarrow i_4^3 + w_4^1), (i_4^3 + f_4^2 \rightarrow \widehat{y} + \widehat{y} + w_4^2)\}$,
  - $\mathbf{F}_{\alpha_4} = \{f_4^1 = 1, f_4^2 = 1\}$.

Figure 1.

Abstract CRN encoding of the AM reaction system. Note that all four encodings follow a common pattern.

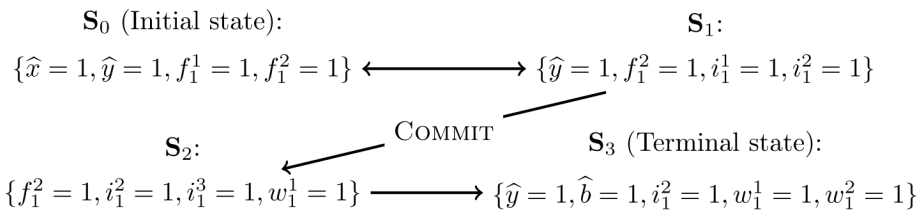

Figure 2.

State space for $[\![a_1]\!]$ from Figure 1, derived from the fuel multiset $\mathbf{F}_{a1}$ and the encoded initial species $\mathcal{M}(x)$ and $\mathcal{M}(y)$ using the reaction encoding CRN $\mathcal{E}_{a_1}$.

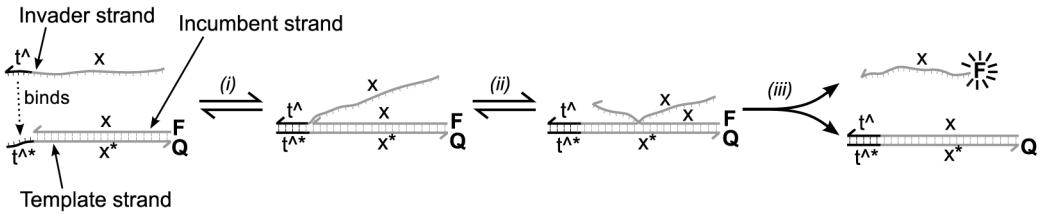

Figure 3.

DNA strand displacement reaction mechanism. *(i)* The toehold $\hat{t}$ on the single-stranded invader is complementary to the toehold on the two-strand complex, which is a complex of the template and incumbent strands. The toehold on the template strand provides a nucleation point for the invader to bind (reversibly) and initiate the strand displacement reaction. *(ii)* Breathing at the end of the duplex enables the invader strand to begin displacing the incumbent strand from the template strand, since the sequences of their x domains match. This initiates a random walk *branch migration* reaction along the x domain. *(iii)* If the branch migrates all the way to the far end of the x domain, the incumbent strand is displaced from the template strand. In this example, the strand displacement reaction is irreversible because there is no toehold for the displaced strand to rebind. As shown here, the complex can be labeled with a fluorophore-quencher pair to provide a readout of successful completion of the strand displacement reaction due to increased fluorescence when the fluorophore and quencher are separated.

```
(* Encoded species *)
def S(N,tx1,x,tx2) = new h
( N * <h tx1^ x tx2^> )

(* Catalyst gate module, implements x + y -> y + z *)
def C(N,tx1,x,tx2,ty1,y,ty2,tz1,z,tz2) = new h1 new h2
( N * {tx1^*}[x tx2^ ty1^]:[y ty2^]<h1 ty1^ h2 tz1^>
| N * {ty2^*}[h1 ty1^]<y ty2^>:[h2 tz1^]<z tz2^> )

(* Producing a single copy of x + y -> y + b *)
( C(1,tx1,x,tx2,ty1,y,ty2,tb1,b,tb2)
| S(1,tx1,x,tx2)
| S(1,ty1,y,ty2) )
```

Figure 4.
DSD code for a four-domain species encoding, and an encoding of the reaction $x + y \rightarrow y + b$ from the AM system.

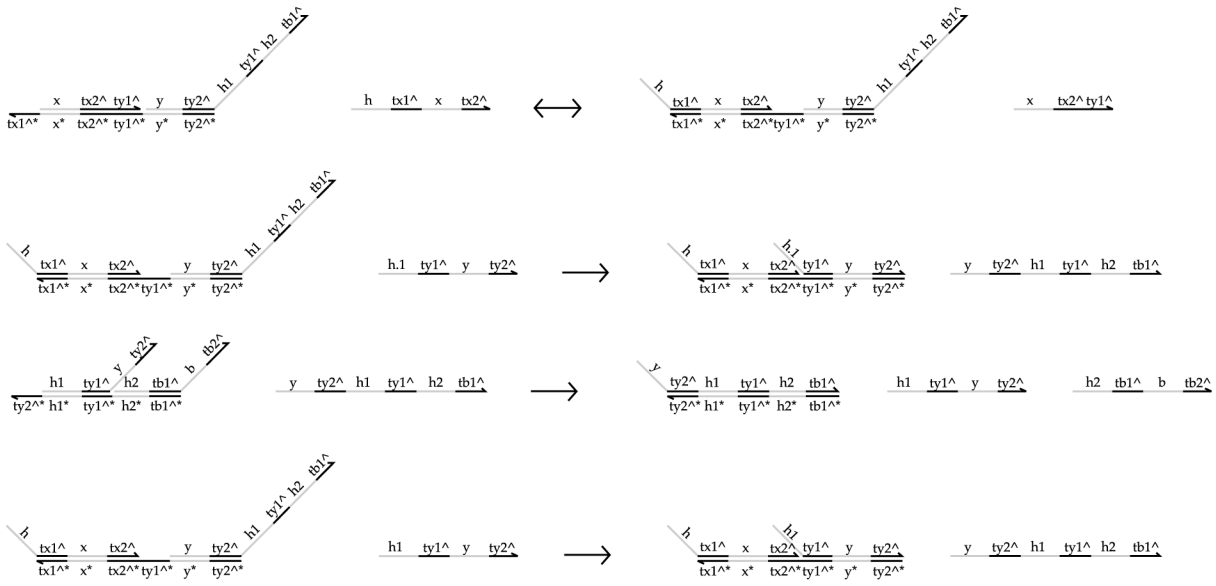

Figure 5.

Compiled CRN of the four-domain strand displacement reaction encoding of the formal reaction $x + y \rightarrow y + b$ from the AM system. We represent toeholds in black and long domains in grey. This figure is the equivalent of Figure 1 for the four-domain strand displacement encoding.

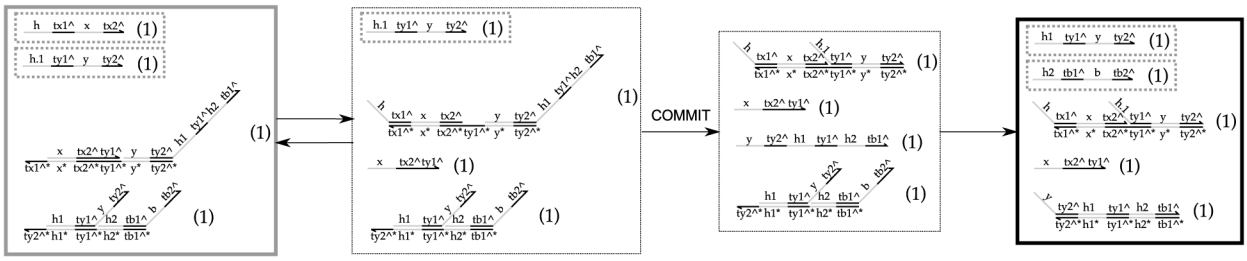

Figure 6.

Compiled state space of the four-domain strand displacement reaction encoding of the formal reaction $x + y \rightarrow y + b$ from the AM system. As in Figure 5, we represent toeholds in black and long domains in grey. The initial state has a thick grey outline, the terminal state has a thick black outline, and the intermediate states are shown with broken outlines. The strands corresponding to formal species have been highlighted with dashed grey outlines. The state transition labeled “commit” indicates the commit reaction in the reaction encoding. This figure is the equivalent of Figure 2 for the four-domain strand displacement encoding.

```
(* Encoded species *)
def S(N,x) =
( N * <t^ x> )

(* Catalyst gate module, implements x + y -> y + z *)
def C(N,x,y,z) = new a new c
( N * {t^*}[x t^]:[y t^]:[c]:[a t^]:[a]
| N * [x]:[t^ z]:[c]:[t^ y]:[t^ a]{t^*}
| N * <t^ c a>
| N * <z c t^> )

(* Producing a single copy of x + y -> y + b *)
( C(1,x,y,b)
| S(1,x)
| S(1,y) )
```

Figure 7.
DSD code for a two-domain species encoding, and an encoding of the reaction $x + y \rightarrow y + b$ from the AM system.

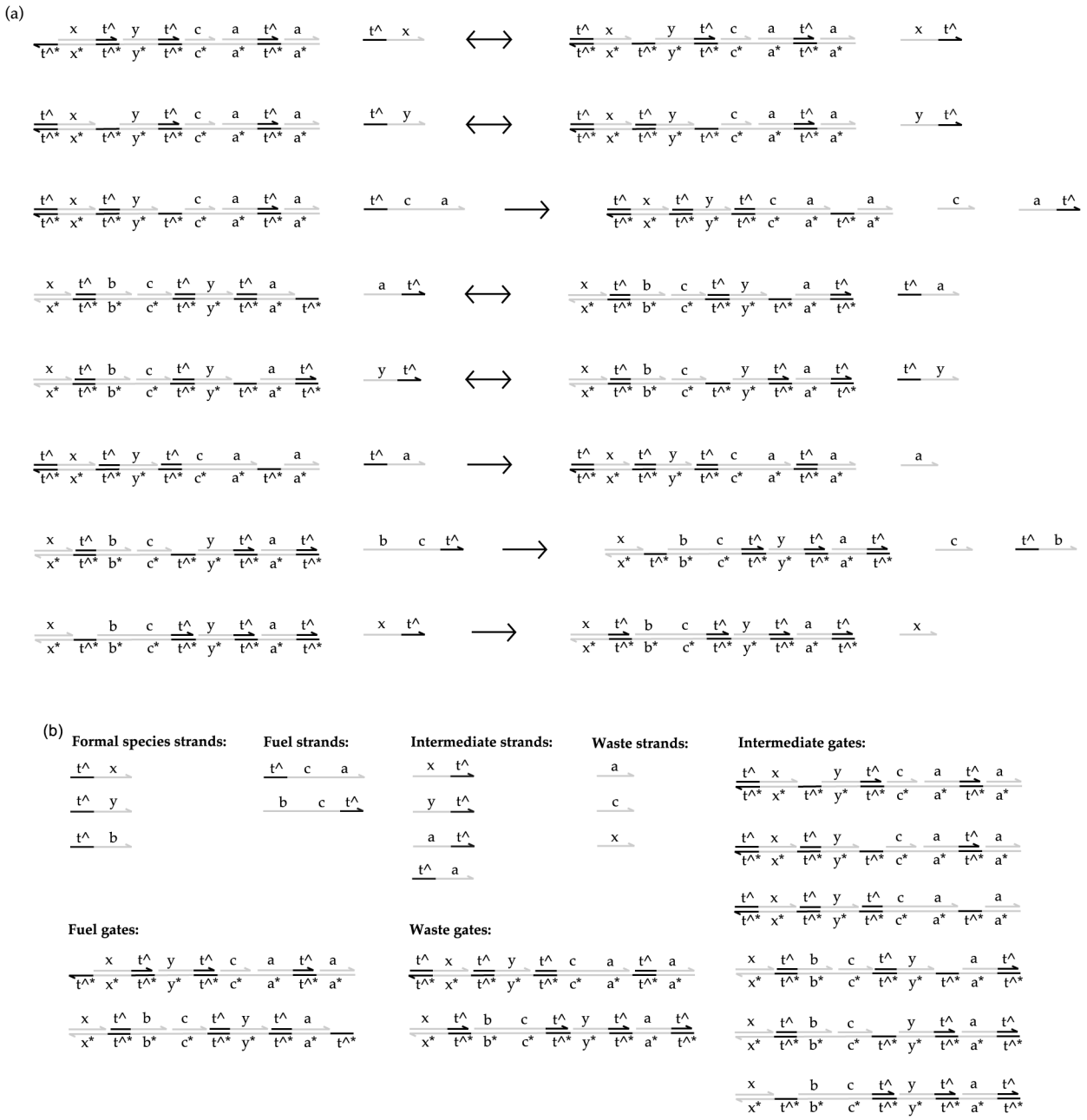

Figure 8.

Summary of reactions and species derived from a two-domain encoding of $x + y \rightarrow y + b$. As in Figure 5, we represent toeholds in black and long domains in grey. (a) All possible reactions using the formal reactant species and the fuel species from the two-domain encoding. (b) All species generated by these reactions, divided into categories (see above). This figure is the equivalent of Figure 1 for the two-domain strand displacement encoding.

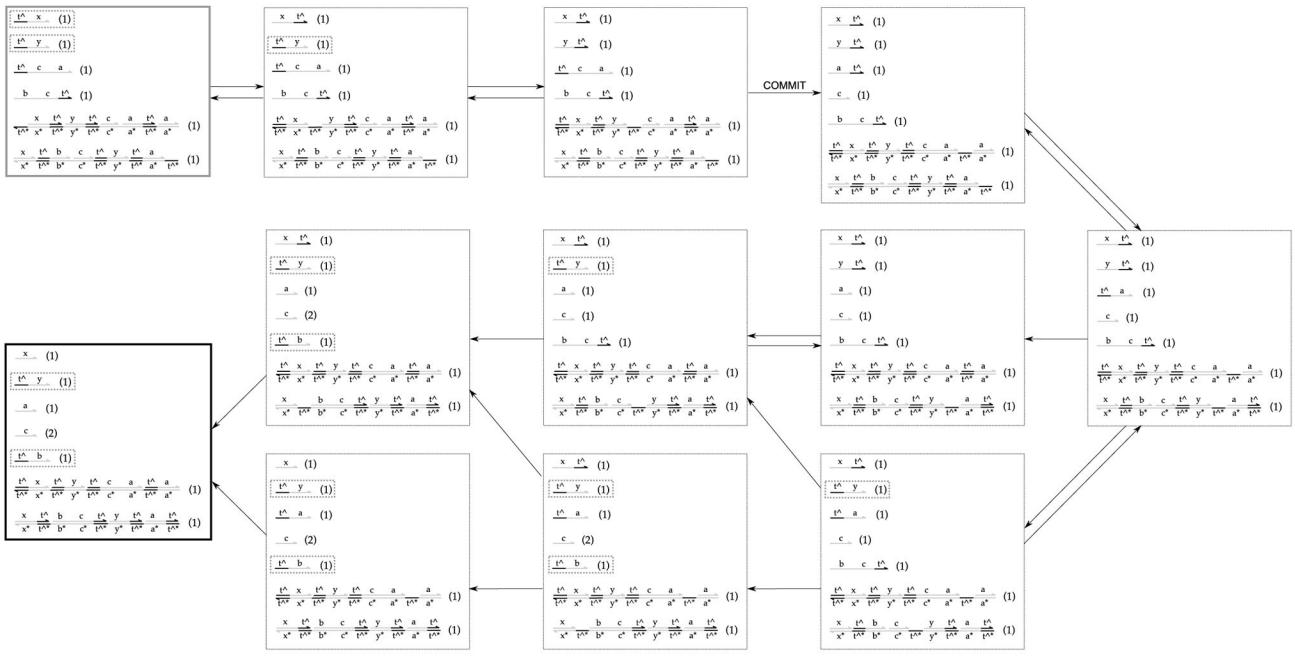

Figure 9.

Summary of catalyst encoding state space derived from an encoding of the reaction $x + y \rightarrow y + b$. As in Figure 5, we represent toeholds in black and long domains in grey. The initial state has a thick grey outline, the terminal state has a thick black outline, and the intermediate states are shown with broken outlines. The strands corresponding to formal species have been highlighted with dashed grey outlines. The state transition labeled "commit" indicates the commit reaction in the reaction encoding. This figure is the equivalent of Figure 2 for the two-domain strand displacement encoding.