Modular verification of DNA strand displacement networks via serializability analysis

Matthew R. Lakin[1], Andrew Phillips[2], and Darko Stefanovic[1,3]

[1] Department of Computer Science, University of New Mexico, NM 87131, USA
[2] Biological Computation Group, Microsoft Research, Cambridge, CB1 2FB, UK
[3] Center for Biomedical Engineering, University of New Mexico, NM 87131, USA

{mlakin,darko}@cs.unm.edu, andrew.phillips@microsoft.com

Abstract. DNA strand displacement gates can be used to emulate arbitrary chemical reactions, and a number of different schemes have been proposed to achieve this. Here we develop modular correctness proofs for strand displacement encodings of chemical reaction networks and show how they may be applied to two-domain strand displacement systems. Our notion of correctness is serializability of interleaved reaction encodings, and we infer this global property from the properties of the gates that encode the individual chemical reactions. This allows correctness to be inferred for arbitrary systems constructed using these components, and we illustrate this by applying our results to a two-domain implementation of a well-known approximate majority voting system.

1 Introduction

The behaviour and kinetics of arbitary chemical reaction networks can be emulated by collections of DNA strand displacement gates [1, 2]. A number of schemes for encoding reactions using strand displacement gates have been proposed, such as four-domain [2], three-domain [3, 4] and two-domain schemes [5]. A common feature of these gates is that they emulate a single-step reaction using a sequence of multiple reactions. These additional steps introduce more opportunities for errors in the design, which may include subtle concurrency bugs that only manifest themselves when a gate is used in a particular context. Therefore it is desirable to develop formal proofs that a given strand displacement gate design is a correct implementation of the desired chemical reactions.

Two-domain encoding schemes are attractive candidates for experimental implementation because they use simple strands and gates without overhangs. However, this necessitates additional intermediate steps in the emulation of the reaction, such as garbage collection steps that convert leftover species into unreactive waste to prevent them slowing down certain reactions. Since a larger number of steps are often needed to encode a given reaction, it is typically 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 [6]. However, in that work we could only verify particular

populations of species, and this was severely limited by the explosion in the size of the state space as the sizes of the species populations increased.

Here we introduce a framework for verification of DNA strand displacement emulations of chemical reaction networks. We adopt a modular approach to proving correctness, by showing that if all components of the system satisfy certain properties then the whole system may be deduced to be correct in a well-defined sense. Our approach is inspired by the concept of *serializability* from database theory [7], 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 interleavings of their reactions are causally equivalent to some serial schedule, in which there is no interleaving between the various reaction encodings. We use simple rewriting rules on reaction traces to serialize them, and apply our technique to the verification of two-domain DNA strand displacement reaction gates [5, 6]. We propose serializability as a reasonable notion of correctness for DNA strand displacement reaction gates because serialized executions of encodings can be directly related to executions of the underlying reactions. Gate designs that are not serializable may display erroneous behaviours that do not correspond to possible behaviours of the underlying reactions, because of unwanted crosstalk between gates. Our correctness criteria will allow us to prove that gate designs do not have such problems.

2 Preliminaries

We now make some preliminary mathematical definitions that will be used throughout the paper. Let $\mathbb{N}$ denote the set of natural numbers, including zero. Following the notation of [8], given a set X, we write $\mathbb{N}^X$ for the set of multisets over X, defined as the set of all functions $f : X \to \mathbb{N}$. 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 is the count associated with the corresponding x_i. For multisets $\mathbf{A}, \mathbf{B} \in \mathbb{N}^X$ we write $\mathbf{A} \circledast \mathbf{B}$ to mean that $(\mathbf{A}(x)) \circledast (\mathbf{B}(x))$ for all $x \in X$, where $\circledast$ is any binary relational operator, for example $\leq$. Similarly, we define arithmetic operations on multisets so that $(\mathbf{A} \pm \mathbf{B})(x) = (\mathbf{A}(x)) \pm (\mathbf{B}(x))$ for all $x \in X$. For subtraction we require that $\mathbf{B} \leq \mathbf{A}$ to avoid negative multiplicities. If $x \in X$ and $n \in \mathbb{N}$, we write $n \cdot x$ for the multiset $\mathbf{A} \in \mathbb{N}^X$ such that $\mathbf{A}(x) = n$ and such that $\mathbf{A}(x') = 0$ for $x' \in X$ where $x' \neq x$. 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 set of chemical species and R is a set of chemical reactions over X. A chemical reaction has the form $\mathbf{R} \to \mathbf{P}$, where $\mathbf{R}, \mathbf{P} \in \mathbb{N}^X$ represent the reactants and products of the reaction, respectively. If $r = (\mathbf{R} \to \mathbf{P})$ then we let $r^{-1} = (\mathbf{P} \to \mathbf{R})$ and observe that $(r^{-1})^{-1} = r$.* Note that we do not consider reaction rates at all in this paper. *For well-formedness of reactions we will stipulate that $\mathbf{R} \neq \mathbf{P}$, and for well-formedness of CRNs we require that all constituent reactions are well-formed. Henceforth we assume that all CRNs are well-formed.*

Definition 2 (CRN states and reduction). *A state* $\mathbf{S}$ *of a CRN* $\mathcal{C} = (X, R)$ *is just a multiset drawn from* $\mathbb{N}^X$. *A reaction* $r = (\mathbf{R} \rightarrow \mathbf{P}) \in R$ *is* enabled *in state* $\mathbf{S}$ *if* $\mathbf{R} \leq \mathbf{S}$, *written* $\mathbf{S} \vdash_{\mathcal{C}} r$. *Furthermore, if* $\mathbf{S}' = \mathbf{S} - \mathbf{R} + \mathbf{P}$ *then we write* $\mathbf{S} \xrightarrow{r} \mathbf{S}'$ *to indicate that applying the reaction* r *to* $\mathbf{S}$ *results in* $\mathbf{S}'$.

Definition 3 (CRN traces). *Given a CRN* $\mathcal{C} = (X, R)$, *a trace* τ *is an ordered list* $[r_1, \ldots, r_i, \ldots]$ *of elements of* R. *Traces may be finite or infinite, and we write* $length(\tau)$ *for the length of the finite trace* τ. *We write* $\text{Traces}(\mathcal{C})$ *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.,* $length(\epsilon) = 0$*).*

Definition 4 (Valid reductions). *A pair* $(\mathbf{S}, \tau)$ *is a* valid reduction *of a CRN* $\mathcal{C}$, *written* $\mathbf{S} \vdash_{\mathcal{C}} \tau$, *if* $\tau \in \text{Traces}(\mathcal{C})$ *and either (i)* $\tau = \epsilon$ *or (ii)* $\tau = [r]{:}\tau'$ *and* $\mathbf{S} \xrightarrow{r} \mathbf{S}'$ *such that* $\mathbf{S}' \vdash_{\mathcal{C}} \tau'$ *also holds. If* τ *is finite, we write* $final_{\mathcal{C}}(\mathbf{S}, \tau)$ *for the final state of the trace, and say that* $\mathbf{S} \xrightarrow{\tau} \mathbf{S}'$ *if* $\mathbf{S}' = final_{\mathcal{C}}(\mathbf{S}, \tau)$.

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* $\tau \in \text{Traces}(\mathcal{C})$. *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* $\mathbf{S}$.

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* $\tau \in \text{Traces}(\mathcal{C})$ *such that* $final_{\mathcal{C}}(\mathbf{S}, \tau)$ *is a terminal state under* $\mathcal{C}$.

Definition 7 (Reversible and irreversible 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 it is* irreversible *if* $\mathbf{S} \xrightarrow{r} \mathbf{S}'$ *implies that* $\mathbf{S}$ *is not reachable from* $\mathbf{S}'$ *under* $\mathcal{C}$. *Note that it is* not *the case that every reaction is necessarily either reversible or irreversible in the above sense: for example, consider the CRN with reactions* $a \rightarrow b$, $b \rightarrow c$ *and* $c \rightarrow a$. *None of these reactions are reversible, but none of the reactions are irreversible either, because there is always a route back to the previous state via the other two reactions.*

3 Two-domain DNA strand displacement gates

We cannot develop a modular verification framework without a common language in which to formalize the system components: this allows us to check for unwanted crosstalk between modules in a uniform way. We will use the DSD language for formalizing DNA strand displacement systems [4, 9]. Here we present a subset of the syntax and reactions needed for the two-domain gates in this paper: see [4] for full definitions.

The syntax and graphical notation for two-domain DSD systems is presented in Table 1. We write `t^`, `u^`, etc., for *toehold domains*, which are short enough to hybridize reversibly (shown in colour), and `a`, `b`, `x`, `y`, etc., for *recognition domains*, which are long enough to hybridize irreversibly (shown in grey).

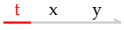
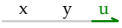
Species category	DSD syntax	Graphical notation
Strands, S		
Signal strand	`<t^ x>`	
Cosignal strand	`<x u^>`	
Extended signal strand	`<t^ x y>`	
Extended cosignal strand	`<x y u^>`	
Inert waste strand	`<x>`	
Gates, G		
Exposed toehold gate segment	`{t^*}`	
Double-stranded signal gate segment	`[t^ x]`	
Double-stranded cosignal gate segment	`[x u^]`	
Collector gate segment	`[x]`	
Segment concatenation	`G1:G2`	Lower strands joined

Table 1. DSD syntax and graphical representation of two-domain DNA species. Here, `G1` and `G2` stand for arbitrary, non-empty gate structures.

We use the asterisk to denote the Watson-Crick complement of a particular domain, and assume that the domains are non-interfering, i.e., `x` will only hybridize with `x*`. Single strands S may be *signals* `<t^ x>` or *cosignals* `<x u^>`. Following [6] and [5], we 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 u^>`. Finally, certain reactions can produce waste strands of the form `<x>`, which are unreactive as they have no toehold to interact with other species.

Figure 1 presents the set of possible reactions between two-domain DNA strands and gates presented in Table 1, according to the *Infinite* DSD semantics [4]. In reaction *(i)*, a signal strand is consumed by a gate via an exposed complementary toehold, producing a new gate with a different exposed toehold and a free cosignal strand. Note that this reaction is reversible, since the cosignal can react with the product gate to release the original signal into solution. Reactions *(ii)* and *(iii)* show irreversible consumption of a cosignal and a signal respectively, by gates containing the appropriate combination of a collector segment and an exposed complementary toehold, sealing off the toehold and releasing an inert waste strand into solution. Finally, reactions *(iv)* and *(v)* show irreversible consumption of an extended signal and an extended cosignal respectively. These reactions seal off a toehold and release an inert waste strand *and* an output strand, which is either a signal or a cosignal. (It is also possible to irreversibly consume an extended strand using two neighbouring collector segments without releasing a signal or cosignal, but we do not consider this because gate designs typically do not require it.) The reactions from Figure 1 are all either reversible or irreversible in the sense of Definition 7, and they are

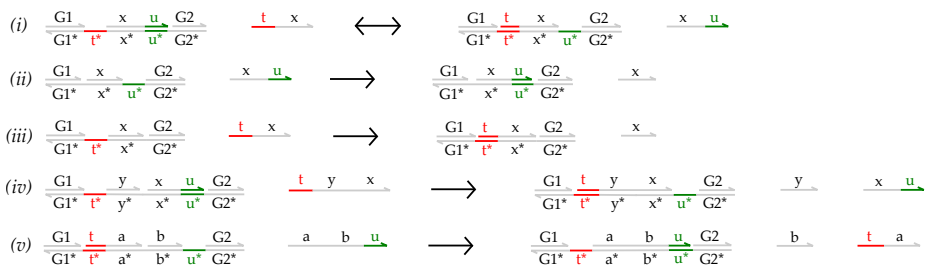


Fig. 1. Two-domain DNA strand displacement reactions employed in this paper. Here, `G1` and `G2` stand for arbitrary, possibly-empty gate structures.

all bimolecular reactions that involve one gate and one strand as reactants. We ignore unproductive reactions [4] in which a toehold binds but cannot initiate a subsequent branch migration reaction.

4 Modular chemical reaction encodings

The goal of this paper is to verify that a CRN involving DNA strand displacement reactions correctly encodes a particular CRN of interest. We specify CRN encodings in a modular way, by defining subsystems which each encode a particular chemical reaction, and composing these to form a single CRN. We refer to the species and reactions of the CRN being encoded as *formal* species and reactions, and let α range over formal reactions and $\mathcal{F}$ over formal CRNs. We will encode formal species using the family of DSD species defined in Table 1, which interact via reactions derived from the rules in Figure 1.

Definition 8 (Species encodings). *For a set X of formal species, we define a bijective* species map $\mathcal{M}$ *which maps every $x \in X$ to a DSD species from Table 1. (As is standard, we assume that encoded formal species never interact directly.) In the case of two-domain systems, we fix a global toehold domain* `t^` *and choose our species map such that $\mathcal{M}(x) =$* `<t^ x>` *for all $x \in X$, i.e., each formal species is encoded by a different domain. Writing x' and $\mathbf{S}'$ for encoded species and states respectively, we lift $\mathcal{M}$ (and the inverse mapping $\mathcal{M}^{-1}$) to operate on states by simply ignoring the species that are not present in their domain of definition, as follows.*

$$(\mathcal{M}(\mathbf{S}))(x') = \begin{cases} \mathbf{S}(\mathcal{M}^{-1}(x')) & \text{if } x' \in \text{image}(\mathcal{M}) \\ 0 & \text{otherwise.} \end{cases}$$

$$(\mathcal{M}^{-1}(\mathbf{S}'))(x) = \begin{cases} \mathbf{S}'(\mathcal{M}(x)) & \text{if } x \in dom(\mathcal{M}) \\ 0 & \text{otherwise.} \end{cases}$$

We will develop encodings of a formal CRN by constructing a *reaction encoding* $[\![\alpha]\!]$ for each constituent formal reaction α, as follows.

Definition 9 (Reaction encodings). *An* encoding $[\![\alpha]\!]$ *of a formal reaction* $\alpha = (\mathbf{R}_\alpha \to \mathbf{P}_\alpha)$ *is a multiset* $\mathbf{F}_\alpha$ *of fuel species from Table 1 such that, if we let* $\mathbf{S}_\alpha^{init} = \mathcal{M}(\mathbf{R}_\alpha) + \mathbf{F}_\alpha$*, then there exists a terminal state* $\mathbf{S}_\alpha^{final}$ *that is universally reachable from* $\mathbf{S}_\alpha^{init}$ *using the DSD reaction rules, and where* $\mathcal{M}^{-1}(\mathbf{S}_\alpha^{init}) = \mathbf{R}_\alpha$ *and* $\mathcal{M}^{-1}(\mathbf{S}_\alpha^{final}) = \mathbf{P}_\alpha$*. 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 to completely execute a single copy of the encoding. Finally, if* $\mathbf{S}_\alpha^{init} \xrightarrow{\tau} \mathbf{S}_\alpha^{final}$ *we say that the trace* τ *is an* execution *of* $[\![\alpha]\!]$*.*

Intuitively, a reaction encoding $[\![\alpha]\!]$ comprises the minimal amount of fuel $\mathbf{F}_\alpha$ which, when placed with the encoded reactants $\mathcal{M}(\mathbf{R}_\alpha)$, can execute the encoding of a *single instance* of the formal reaction α. The definition also requires 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 can 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 redundant reaction pathways in the encoding, each requiring different fuel species. Note that we refer to the species from $\mathbf{F}_\alpha$ from Definition 9 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. This encompasses not only the auxiliary single strands typically referred to as "fuels" in the literature, but also the gate complexes that must be present initially. 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.

Definition 10 (CRN encodings). *Suppose that* $\mathcal{F}$ *is a formal CRN that contains formal reactions* $\alpha_1, \ldots, \alpha_n$ *with corresponding encodings* $[\![\alpha_1]\!], \ldots, [\![\alpha_n]\!]$*. We use the individual reaction encodings to derive a CRN* $\mathcal{E} = (X_\mathcal{E}, R_\mathcal{E})$ *that encodes* $\mathcal{F}$*, by*

- *forming the set of initial species, which comprises all fuel species from the multisets* $\mathbf{F}_{\alpha_1}, \ldots, \mathbf{F}_{\alpha_n}$ *and all encoded formal species* $\mathcal{M}(x)$*, where* x *is mentioned in one of the formal reactions* $\alpha_1, \ldots, \alpha_n$*, and then*
- *recursively computing the set* $X_\mathcal{E}$ *of all reachable species the set* $R_\mathcal{E}$ *of all possible reactions, using the reaction rules from Figure 1.*

We refer the reader to previous work which formally defined the reaction enumeration algorithm from the DSD compiler [4, 10], which can be used to automate the process described in Definition 10.

The most basic correctness property of reaction encodings is that they are capable of emulating any valid trace of formal reactions. For a formal trace $\tau \in \text{Traces}(\mathcal{F})$, we say that a trace $\tau' \in \text{Traces}(\mathcal{E})$ in the encoded CRN is a *serial execution* of $\tau = [r_1, \ldots, r_n]$ if τ' can be decomposed into subtraces $\tau_1 :: \cdots :: \tau_n$ such that the i^{th} subtrace τ_i is an execution of r_i. Since reaction encodings require fuel species to be present, any such statement 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 1 (Fuel required for emulation). *Let $\tau = [\alpha_1, \ldots, \alpha_n] \in \text{Traces}(\mathcal{F})$ 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. Then, $(\mathcal{M}(\mathbf{S}) + \mathbf{F}) \vdash_{\mathcal{E}} \tau_{ser}$ iff $\mathbf{F} \geq \mathit{reqfuel}(\tau)$, where the required fuel, $\mathit{reqfuel}(\tau)$, is defined as the sum of the fuel required by the encoding of each reaction in the formal trace, i.e., $\mathit{reqfuel}(\tau) \triangleq \mathbf{F}_{\alpha_1} + \cdots + \mathbf{F}_{\alpha_n}$.* □

It follows from Lemma 1 that 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, and there is an obvious connection between the serial execution and the formal trace.

Theorem 1 (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 $\mathbf{F} \geq \mathit{reqfuel}(\tau)$ then $(\mathcal{M}(\mathbf{S}) + \mathbf{F}) \vdash_{\mathcal{E}} \tau_{ser}$ for some $\tau_{ser} \in \text{Traces}(\mathcal{E})$ which is a serial execution of τ.* □

5 Soundness of CRN encodings

The completeness result above only concerns one possible reduction trace of the encoding. In this section we prove a more involved *soundness* result, which shows that *all* possible reduction 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 encodings 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 11 (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 $\mathbf{S} \vdash_{\mathcal{C}} \tau$ and that a derivation exists using the following inference rules.*

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

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

The (CANCEL) rule allows the subtrace τ_2 to be removed if its net effect is no change, and the (SWAP) rule allows two neighbouring subtraces τ_2 and τ_3 to be swapped if they may occur in either order. In the latter case, 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. It is not hard to show that trace rewriting preserves validity and the final states of reductions, as stated below.*

Lemma 2. *If* $\mathbf{S} \vdash_{\mathcal{C}} \tau \rightsquigarrow \tau'$ *then* $\mathbf{S} \vdash_{\mathcal{C}} \tau'$ *and* $\mathit{final}_{\mathcal{C}}(\mathbf{S}, \tau) = \mathit{final}_{\mathcal{C}}(\mathbf{S}, \tau')$.

Note that it is *not* the case that any two valid traces from a given starting state must be trace-equivalent—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 can be rewritten to produce a serial execution of a valid formal trace. To obtain this result, we must place additional constraints on the reaction encodings which we will consider.

Definition 12 (Stratified chemical reaction networks). *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* $\tau \in \mathrm{Traces}(\mathcal{C})$ *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* $\Lambda(\mathbf{S}_0, \mathbf{S}') = \Lambda(\mathbf{S}_0, \mathbf{S}) \pm 1$.

We observe that reactions derived from Figure 1 always give rise to stratified CRNs, since the reversible reactions can only be reversed by executing the corresponding inverse reaction and the irreversible reactions all produce an inert species which prevents the system from returning to the previous state. Intuitively, this allows us to subdivide the transitions in our reaction encodings into *forward steps* that move away from the initial state $\mathbf{S}_0$ (i.e., transitions $\mathbf{S} \xrightarrow{r} \mathbf{S}'$ where $\Lambda(\mathbf{S}_0, \mathbf{S}') = \Lambda(\mathbf{S}_0, \mathbf{S}) + 1$) and *backward steps* that move back towards the initial state $\mathbf{S}_0$ (i.e., transitions $\mathbf{S} \xrightarrow{r} \mathbf{S}'$ where $\Lambda(\mathbf{S}_0, \mathbf{S}') = \Lambda(\mathbf{S}_0, \mathbf{S}) - 1$). We must also categorize the species involved in each reaction encoding according to their role: we require that the set of *all* species involved in the encoding $[\![\alpha]\!]$, denoted $\mathit{species}([\![\alpha]\!])$, can be partitioned into:

- $\mathit{formals}([\![\alpha]\!])$ (those species in the image of $\mathcal{M}$);
- $\mathit{waste}([\![\alpha]\!])$ (those species which are unreactive);
- $\mathit{fuels}([\![\alpha]\!])$ (those species which appear in $\mathbf{F}_\alpha$); and
- $\mathit{intermediates}([\![\alpha]\!])$ (the remaining species).

We now abuse the terminology of [11] to define a notion of *copy tolerance*, and use this to state our restrictions on individual reaction encodings.

Definition 13 (Copy tolerance). *A reaction encoding* $[\![\alpha]\!]$ *is* copy tolerant *of a species* x *if* $\{\tau \mid \mathbf{S}_\alpha^{init} \vdash_{\mathcal{E}} \tau\} = \{\tau \mid (\mathbf{S}_\alpha^{init} + n \cdot x) \vdash_{\mathcal{E}} \tau\}$ *for all* $n \in \mathbb{N}$*, where* $\mathcal{E}$ *is the CRN derived from* $[\![\alpha]\!]$ *(extended to include* x *if necessary) and* $\mathbf{S}_\alpha^{init}$ *is the initial state of* $[\![\alpha]\!]$.

* A proof sketch for Lemma 2 is included in the appendices, which can be downloaded from the first author's webpage.

Definition 14 (Transactional reaction encodings). *Consider a formal reaction* $\alpha = (\mathbf{R}_\alpha \to \mathbf{P}_\alpha)$*, encoded as* $[\![\alpha]\!]$ *via the fuel multiset* $\mathbf{F}_\alpha$*. Let* $\mathbf{S}_0 = \mathbf{S}_\alpha^{init} = \mathcal{M}(\mathbf{R}_\alpha) + \mathbf{F}_\alpha$ *denote the initial state consisting of just the required reactants and fuels, and let* $\tau = [r_1, \ldots, r_n] \in \text{Traces}(\mathcal{E})$ *be a terminal trace of the encoding starting from* $\mathbf{S}_0$*, where* $r_i = (\mathbf{R}_i \to \mathbf{P}_i)$ *for* $i \in \{1, \ldots, n\}$*. Labelling 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 irreversible 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. $\mathcal{M}^{-1}(\mathbf{S}_{j-1} - \mathbf{R}_j) = \varnothing$*.*

We say that $[\![\alpha]\!]$ *is* transactional *if 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 copy tolerant of 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 copy tolerant of every species in the multiset* $\mathbf{L}_\alpha$ *of* leftover species*.*

In Definition 14, criterion 1 requires that the trace can be partitioned into two disjoint subtraces by the first irreversible reaction. Given the reaction rules from Figure 1, this implies that all reactions before that point must be reversible. 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). The restrictions on copy 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 copy tolerant of waste species, but that the encoding may not be copy tolerant to certain intermediate species. 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.

Example 1 (Effect of a non-transactional reaction encoding). As a concrete example [12] of what might go wrong 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 irreversible step 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 irreversible steps. 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 irreversible steps 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 state $\{x = 1, a = 1\}$ using the above set of formal reactions: our encoding of this CRN is unsound. The specific problem here is that the y produced by the $x \to y$ reaction is accessible before the encoding has executed an irreversible step to commit to its production. Until there is no way for this product to be reclaimed by the gate that produced it, it is unsound for other reactions to consume it.

We now define *compatible* reaction encodings, in which direct sharing of species between the encodings is only permitted in certain situations.

Definition 15 (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 copy tolerant of 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$([\![\alpha]\!]) \cap$ 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 do 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 can use the DSD semantics and compiler to check for unwanted interference between reaction encodings. We now state some preliminary lemmas needed to prove our main result.**

Lemma 3 (Trace rewriting and reversible reactions). *If $\tau \in \text{Traces}(\mathcal{C})$ consists entirely of reversible reactions and $\mathbf{S} \vdash_{\mathcal{C}} \tau$ then there exists $\tau' \in \text{Traces}(\mathcal{C})$ such that $\mathbf{S} \vdash_{\mathcal{C}} \tau{:}\tau' \rightsquigarrow \epsilon$.*

Lemma 4 (Serializability). *Assume that all reaction encodings are transactional and pairwise compatible, and suppose that $\mathbf{S} \vdash_{\mathcal{E}} \tau$, 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 $\mathbf{S} \vdash_{\mathcal{C}} r_1^\alpha$. Then, there exists τ'_{rest} such that $\mathbf{S} \vdash_{\mathcal{E}} \tau \rightsquigarrow \tau_\alpha{:}\tau'_{rest}$. □

** Proof sketches for Lemma 4 and Theorem 2 are included in the appendices, which can be downloaded from the first author's webpage.

We can now state and prove our main soundness theorem, which is valid for systems composed of reaction encodings that satisfy the criteria in Definition 14 and Definition 15. 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. In doing so we write $pt(\mathbf{X}, \mathbf{F})$ for the set of non-empty formal traces that are valid from the formal state $\mathbf{X}$ and that can be emulated using the fuel $\mathbf{F}$, i.e., $pt(\mathbf{X}, \mathbf{F}) = \{\tau \in \text{Traces}(\mathcal{F}) \mid \mathbf{X} \vdash_{\mathcal{F}} \tau \wedge \mathit{reqfuel}(\tau) \leq \mathbf{F} \wedge \tau \neq \epsilon\}$.

Theorem 2 (Soundness). *For a formal CRN $\mathcal{F}$ with reactions $\alpha_1, \ldots, \alpha_n$, assume that the corresponding reaction encodings are all transactional and pairwise compatible. Let $\mathbf{X}$ range over multisets of formal species, and let $\mathbf{F}$ range over multisets of fuels such that $\mathbf{F} = \mathbf{F}_{\alpha_{k_1}} + \cdots + \mathbf{F}_{\alpha_{k_j}}$, for $k_1, \ldots, k_j \in \{1, \ldots, n\}$ (i.e., there are no incomplete reaction encodings). Then, for all $\tau \in \text{Traces}(\mathcal{E})$ and all $\mathbf{X}$ and $\mathbf{F}$ such that $(\mathcal{M}(\mathbf{X}) + \mathbf{F}) \vdash_{\mathcal{E}} \tau$, either:*

- *$pt(\mathbf{X}, \mathbf{F}) = \varnothing$ and there exists τ' such that $(\mathcal{M}(\mathbf{X}) + \mathbf{F}) \vdash_{\mathcal{E}} \tau{:}\tau' \rightsquigarrow \epsilon$; or*
- *$pt(\mathbf{X}, \mathbf{F}) \neq \varnothing$ 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 $\tau_{formal} \in pt(\mathbf{X}, \mathbf{F})$.* □

6 Verification example

In this section we present an example application of our modular verification strategy to a two-domain strand displacement network [5]. We focus on the two-domain catalyst gate introduced in [6], which implements a reaction of the form $x + y \to x + z$. The DSD code for this gate design is as follows.

```
(* DSD code for two-domain catalyst gate. *)
(* Use with Infinite DSD semantics. *)

(* Define a global toehold *)
new t

(* Catalyst gate module, x + y -> x + 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^> )

(* Example initial state *)
( C(1,x,y,z) | <t^ x> | <t^ y> )
```

The corresponding initial and terminal states of one such reaction gate implementing the reaction $x + y \to x + z$ are shown in Figure 2. In previous work [6], we used these gates to implement the approximate majority voting circuit

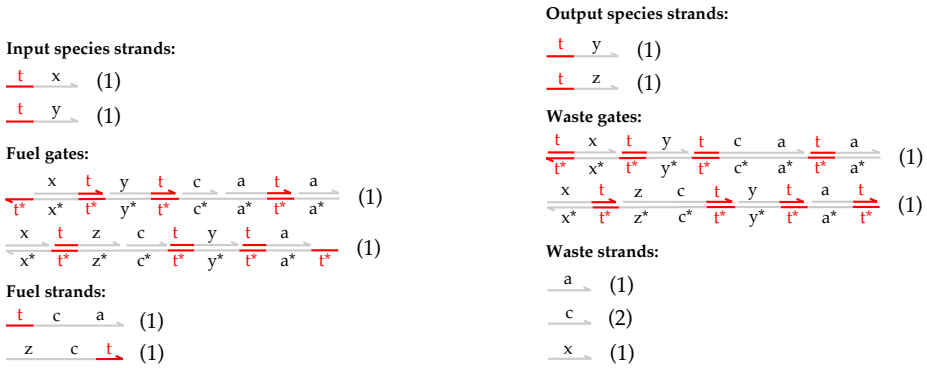


Fig. 2. Initial (left) and terminal (right) states for the catalyst gate which encodes the reaction $x + y \rightarrow y + z$.

of [13], by instantiating the module to implement the four chemical reactions from the approximate majority circuit: *(i)* $x + y \rightarrow y + b$, *(ii)* $x + y \rightarrow x + b$, *(iii)* $x + b \rightarrow x + x$, *(iv)* $y + b \rightarrow y + y$. Thus the catalyst gate module must function correctly both when the two products are different species (for reactions *(i)* and *(ii)*) and when they are the same (for reactions *(iii)* and *(iv)*). It is not difficult to show that each of the resulting reaction encodings satisfies the correctness criteria from Definition 14, and that they are pairwise compatible (Definition 15). The private domains declared within the scope of the gate definition ensure that each reaction encoding involves unique gate species and fuel strands, and the DSD reaction rules from Figure 1 can be used to verify that there are no interactions between the species of the different reaction encodings.*** Therefore, by Theorem 1 and Theorem 2 every trace produced by these reactions can be rewritten to produce a serial execution of the four formal reactions above, so we view these gates as a correct encoding of the approximate majority system.

7 Discussion

We have shown that any strand displacement system composed of reaction encodings that meet the criteria from Definition 14 and Definition 15 is correct in the sense that all traces can be rewritten using the rules from Definition 11 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 gates 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. Although we used

*** Further details are included in the appendices, which can be downloaded from the first author's webpage.

two-domain strand displacement reactions to define our encodings, in principle similar results could be derived for other implementations of chemical reaction networks. It is interesting that the correctness criteria from Definition 14 share much in common with other notions from existing concurrency theory, such as *two-phase locking* [7], 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.

To our knowledge, this paper is the first modular analysis of chemical reaction network encodings. In Definition 14 and Definition 15 we aimed to allow the maximum possible sharing of species between different reaction encodings without invalidating the soundness result in Theorem 2. However, it is worth noting that certain previously published designs for two-domain strand displacement gates fall foul of our restrictions on the structure of reaction encodings and on the sharing of species between encodings. The gate designs from [5] without irreversible product release do not involve a commit reaction as defined in Definition 14, and therefore the soundness theorem does not hold for these gates. Furthermore, certain combinations of these gates may violate our requirement that shared species must fall into the same category in all reaction encodings. In some gates from [5] 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 should be addressed in future two-domain gate designs.

Our definition of species encodings works for any scheme in which there is a bijective mapping between the formal species and the DSD species which represent them, such as the two-domain scheme. A straightforward generalization of our representation language to handle *wildcard domains* should allow us to verify gate designs such as those which use three- and four-domain species encodings using history domains [2, 3]. In these schemes a single formal species is encoded by a family of related DNA species with similar structure but a different history domain. To extend our results to history domains, it would suffice to prove that each reaction encoding can accept any variant of the input strand, regardless of the history domain. Our other key results, such as serializability and soundness, do not rely on a bijective species encoding and should remain valid. Finally, 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.

Prior work on CRN verification [12, 14] has focused on analyzing full systems, and we believe that there is a strong connection between our work and the weak bisimulation-based approach of [14]. In other related work, Cardelli and Laneve [15, 16] developed a theory of reversible computational structures with a strong relationship to DNA strand displacement reaction gates. How-

ever, 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 [11], and in particular the notion of *copy tolerance*, is directly related to our work, as is previous work on CRN programmability [17, 18].

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 $[\![w + x \to y + z]\!]$ 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. Note that in Section 6 we did not verify the module definition directly, but rather a number of specific instantiations of it. Expressing our correctness criteria in a temporal logic would allow module definitions to be checked automatically using a model checker, in order to cover all possible input patterns.

Finally, we note that our formalism and proofs do not take account of reaction rates. Expressing correctness in terms of reachability is both important and natural from a computer science perspective. However, unfavourable kinetics might cause gates that satisfy our reachability criteria to function poorly in practice, as discussed above. Furthermore, certain gate designs that fail to satisfy the criteria might function acceptably in practice due to favourable kinetics, as exemplified by the "wisdom of crowds" example [6, 5]. Proving soundness of gate designs 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 [2] presented similar proofs for a particular encoding of chemical reaction networks using 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 [19]. 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 and 1028238. M.R.L. gratefully acknowledges support from the New Mexico Cancer Nanoscience and Microsystems Training Center.

References

1. D.Y. Zhang and G. Seelig. Dynamic DNA nanotechnology using strand-displacement reactions. *Nat. Chem.*, 3(2):103–113, Feb 2011.

2. D. Soloveichik, G. Seelig, and E. Winfree. DNA as a universal substrate for chemical kinetics. *PNAS*, 107(12):5393–5398, 2010.
3. L. Cardelli. Strand algebras for DNA computing. *Nat. Comput.*, 10(1):407–428, 2010.
4. M.R. Lakin, S. Youssef, L. Cardelli, and A. Phillips. Abstractions for DNA circuit design. *JRS Interface*, 9(68):470–486, 2012.
5. L. Cardelli. Two-domain DNA strand displacement. *Math. Structures Comput. Sci.*, 23:247–271, 2013.
6. M.R. Lakin, D. Parker, L. Cardelli, M. Kwiatkowska, and A. Phillips. Design and analysis of DNA strand displacement devices using probabilistic model checking. *JRS Interface*, 9(72):1470–1485, 2012.
7. C.H. Papadimitriou. The serializability of concurrent database updates. *Journal of the ACM*, 26(4):631–653, 1979.
8. H.-L. Chen, D. Doty, and D. Soloveichik. Deterministic function computation with chemical reaction networks. In D. Stefanovic and A. Turberfield, editors, *DNA18 Proceedings*, volume 7433 of *LNCS*, pages 25–42. Springer, 2012.
9. A. Phillips and L. Cardelli. A programming language for composable DNA circuits. *JRS Interface*, 6(Suppl. 4):S419–S436, 2009.
10. M.R. Lakin, S. Youssef, F. Polo, S. Emmott, and A. Phillips. Visual DSD: a design and analysis tool for DNA strand displacement systems. *Bioinformatics*, 27(22):3211–3213, 2011.
11. A. Condon, B. Kirkpatrick, and J. Maňuch. Reachability bounds for chemical reaction networks and strand displacement systems. In D. Stefanovic and A. Turberfield, editors, *DNA18 Proceedings*, volume 7433 of *LNCS*, pages 43–57. Springer, 2012.
12. S.W. Shin. Compiling and verifying DNA-based chemical reaction network implementations. Master's thesis, California Institute of Technology, 2012.
13. D. Angluin, J. Aspnes, and D. Eisenstat. A simple population protocol for fast robust approximate majority. *Dist. Comput.*, 21(2):87–102, 2008.
14. Q. Dong. A bisimulation approach to verification of molecular implementations of formal chemical reaction networks. Master's thesis, Stony Brook University, 2012.
15. L. Cardelli and C. Laneve. Reversible structures. In F. Fages, editor, *CMSB 2011 Proceedings*, pages 131–140. ACM, 2011.
16. L. Cardelli and C. Laneve. Reversibility in massive concurrent systems. *Sci. Ann. Comp. Sci.*, 21(2):175–198, 2011.
17. D. Soloveichik, M. Cook, E. Winfree, and J. Bruck. Computation with finite stochastic chemical reaction networks. *Nat. Comput.*, 7:615–633, 2008.
18. M. Cook, D. Soloveichik, E. Winfree, and J. Bruck. Programmability of chemical reaction networks. In A. Condon, D. Harel, J. N. Kok, A. Salomaa, and E. Winfree, editors, *Algorithmic Bioprocesses*, pages 543–584. Springer, 2009.
19. M. Kwiatkowska, G. Norman, and D. Parker. PRISM 4.0: verification of probabilistic real-time systems. In *CAV 2011 Proceedings*, volume 6806 of *LNCS*, pages 585–591. Springer, 2011.