Verifying Chemical Reaction Network Implementations: A Bisimulation Approach

Robert F. Johnson[1(✉)], Qing Dong[2], and Erik Winfree[1,3]

[1] Bioengineering, California Institute of Technology, Pasadena, USA
rfjohnso@caltech.edu
[2] Computer Science, SUNY Stony Brook, Stony Brook, NY, USA
[3] Computer Science, California Institute of Technology, Pasadena, USA

Abstract. Efforts in programming DNA and other biological molecules have recently focused on general schemes to physically implement arbitrary Chemical Reaction Networks. Errors in some of the proposed schemes have driven a desire for formal verification methods. We show that by interpreting each implementation species as a set of formal species, the concept of weak bisimulation can be adapted to CRNs in a way that agrees with an intuitive notion of a correct implementation. We give examples of how to use bisimulation to prove the correctness of an implementation or detect subtle problems. We examine the complexity of finding a valid interpretation between two CRNs if one exists, and that of checking whether an interpretation is valid. We show that both are PSPACE-complete in the general case, but are NP-complete and polynomial-time respectively under an assumption that holds in many practical cases. We give algorithms for both of those problems.

1 Introduction

In molecular programming, many real and abstract systems can be expressed in the language of Chemical Reaction Networks (CRNs). A CRN specifies a set of chemical species and the set of reactions those species can do, and the CRN model allows us to deduce the global behavior of the system from that local specification. CRNs are a useful way to separately analyze the computational and the physical aspects of a system. We can use the CRN model to help analyze real systems [3,4] or design engineered systems [5,17].

Despite this ideal, there remains a gap between abstract and real CRNs. To illustrate this gap, consider the approximate majority CRN [1,5]:

$$X + Y \xrightarrow{k} 2B$$

$$X + B \xrightarrow{k} 2X$$

$$Y + B \xrightarrow{k} 2Y$$

This abstract CRN quickly and with high probability converts all of the initial X and Y molecules into the same amount of whichever one was initially greater [1].

Y. Rondelez and D. Woods (Eds.): DNA 2016, LNCS 9818, pp. 114–134, 2016.
DOI: 10.1007/978-3-319-43994-5_8

However, no three molecules with exactly this behavior are known to exist. (In a strict sense, no three molecules with *exactly* this behavior can exist, because for all three reactions to be driven forward would require $X + Y$ to be both lower-energy and higher-energy than $2B$.) For contrast, consider the DNA strand displacement system built by Chen et al. [5] meant to implement this abstract CRN. The DNA system uses additional molecules which are consumed as "fuel" to drive these three reactions, ending up with over 25 each of species and reactions. Without knowing that it is meant to be an implementation of the approximate majority CRN, it might be difficult to tell what the DNA system was meant to do. Even knowing the correspondence, it is not obvious that there is no mistake in that complex implementation.

The issue of verifying correctness is exacerbated by the recent profusion of experimental and theoretical implementations in synthetic biology and molecular programming. Of particular interest to us, Soloveichik et al. [16] designed a systematic way to construct a DNA system to simulate an arbitrary CRN. Since then there have been a number of methods to translate an arbitrary CRN into a DNA strand displacement circuit [2,12,16]. While each one gave arguments for why it was a correct implementation, they did not come with a general theory of what it means to correctly implement a CRN. In some cases this led to subtle problems, of which we will give examples later. To be certain that such implementations are correct, CRN verification methods were invented. Such methods include Shin's pathway decomposition [15], Lakin et al.'s serializability analysis [9], and Cardelli's morphisms between CRNs [3].

We present a method for comparing an implementation CRN with an abstract CRN based on the concept of bisimulation from concurrency theory [11]. Our method associates each implementation species with a multiset of formal species, then asserts correctness if the reactions reachable from any implementation state are the same as the corresponding state in the abstract CRN. Like pathway decomposition [15] and serializability [9] but unlike Cardelli's morphisms [3], our bisimulation method works with the stochastic model for low-copy-number CRNs and doesn't take into account rates or kinetics. The use of interpretations instead of pathways means that some implementations considered correct by pathway decomposition are considered incorrect by bisimulation and vice versa. Interpretations also make bisimulation more local than pathway decomposition or serializability, which we hope makes it more understandable and tractable. We show how bisimulation can be used to prove a CRN implementation correct or identify subtle problems. We present an algorithm to check whether a particular interpretation between two CRNs is a bisimulation relation, and an algorithm to find such an interpretation if one exists. We analyze the computational complexity of both problems. We prove that both are PSPACE-complete in the general case but become polynomial time and NP-complete, respectively, when formal reactions are limited to a constant number of reactants. We hope this method can be used in both verifying that engineered systems match their specification and in comparing natural systems to a system simple enough to analyze.

2 The Chemical Reaction Network Model

We work within the *Chemical Reaction Network* (CRN) model. A CRN is a tuple $(\mathcal{S}, \mathcal{R})$, where $\mathcal{S}$ is a finite set of species and $\mathcal{R}$ a finite set of reactions. A *reaction* is itself a tuple (R, P), where the *reactants* R and *products* P are both multisets of species. We require that in any reaction $R \neq P$. We work with the stochastic model of CRNs, where the state of the system is represented by nonnegative integer counts of each species, and states transition discretely to other states when reactions occur. In particular, a state $S \in \mathbb{N}^{\mathcal{S}}$ can transition to a state T if there is some reaction $(R, P) \in \mathcal{R}$ such that $R \leq S$ and $S - R + P = T$. Often a probabilistic semantics is attached to this model, but for our purposes we only need to know whether something is possible.

We use the notation $\{|\ldots|\}$ for multisets interchangeably with the chemical notation, e.g. $2A + B$, $\{|A, A, B|\}$, and $\{|2A, B|\}$ all refer to the same state. Similarly, we sometimes use the chemical notation for reactions, e.g. $A + B \rightarrow 2C$ is the same as $(\{|A, B|\}, \{|2C|\})$. The "reversible reaction" notation $A + B \rightleftharpoons 2C$ is a shorthand for the two reactions $(\{|A, B|\}, \{|2C|\})$ and $(\{|2C|\}, \{|A, B|\})$. Multisets can be added and multiplied by scalars componentwise, and can be compared componentwise: $S \leq T \iff \forall_X S(X) \leq T(X)$, and $S < T$ if $S \leq T$ and $S \neq T$. If $S \leq T$ then subtraction $T - S$ is defined componentwise. Set operations involving multisets implicitly treat each multiset as the set of all objects which appear at least once; e.g. $\{|1, 1, 2|\} \subset \{1, 2, 3\}$ but $\{|1, 1, 2|\} \not\subset \{1\}$.

In this model, each possible behavior of a CRN is specified by a trajectory: an initial state $S_0 \in \mathbb{N}^{\mathcal{S}}$ together with a (finite or infinite) sequence of reactions $r_k = (R_k, P_k) \in \mathcal{R}$. A trajectory implicitly specifies a sequence of states $S_k = S_0 + \sum_{i \leq k}(P_k - R_k)$, but a sequence of states is not enough to specify a trajectory. For example, if $A \rightarrow B$ and $X + A \rightarrow X + B$ are both reactions, then the sequence of two states $(S_0, S_1) = (\{|X, A|\}, \{|X, B|\})$ does not specify which of those two reactions happened, which is sometimes important. A trajectory is *valid* if each reaction (R_k, P_k) can occur in the state resulting from the previous reactions; that is, $R_k \leq S_k$. In general when we speak of "the trajectories of a CRN" we mean the valid trajectories.

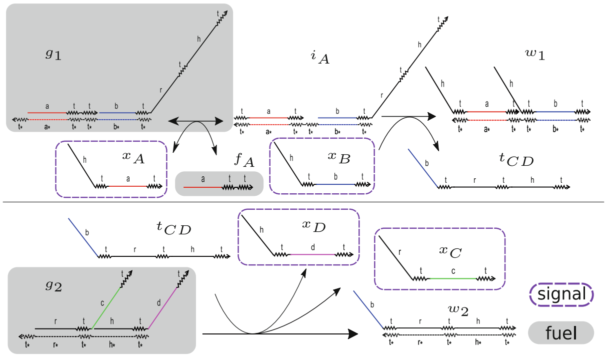
A state T is *reachable* from a state S if T is the result of a valid finite trajectory that starts in S. We say a state T is *coverable* from a state S if there is some $T' \geq T$ such that T' is reachable from S. While the set of reachable states (from any given initial state) is an important aspect of the behavior of a CRN, it does not contain all the information about that CRN. For example, the two CRNs $(\{A, B, C\}, \{A \rightarrow B, B \rightarrow C, C \rightarrow A\})$ and $(\{A, B, C\}, \{A \rightarrow C, C \rightarrow B, B \rightarrow A\})$ have exactly the same set of reachable states T from any given initial state S, but are clearly different in a meaningful way. If however the set of (valid) trajectories of two CRNs are the same, then the two CRNs must be identical: since in particular the length-zero trajectories (i.e. states) are the same, so the sets of species are the same, and the length-one trajectories (single reactions) are the same. We say that two CRNs are *isomorphic* if there is a bijection between the sets of species such that the set of reactions of one, after applying this bijection, equals the set of reactions of the other.

3 The Meaning of Correctness

3.1 Interpretations

Schemes for translating an arbitrary abstract CRN into a DNA Strand Displacement (DSD) implementation [2,12,16] provide designs for the necessary DNA molecules, but how these molecules interact is best described by a model of the relevant biophysics. Reaction enumerators such as Visual DSD [10] and Peppercorn [8] produce, given a set of DNA molecules, a description of their predicted interactions as a CRN, allowing us to compare it to the original CRN using the same language. We refer to the original abstract CRN as the *formal CRN* $(\mathcal{S}, \mathcal{R})$ and the model's enumerated CRN as the *implementation CRN* $(\mathcal{S}', \mathcal{R}')$, which is usually larger than the formal CRN. As a convention, we assume that the formal CRN and the implementation CRN make use of disjoint sets of species. (When using verification to compare a detailed model of a natural system with unknown function to a simpler abstract CRN with known function, the natural system is the implementation and the abstract system is the formal CRN.) There are two other important features typical of engineered implementation CRNs. First, there is typically for each formal species A an implementation species x_A intended to correspond specifically to it, sometimes called a "signal species". Second, certain implementation species must always be present for the system to work, and are designated "fuel species". Fuel species are typically assumed to be held at a constant concentration, for example by setting their initial concentration high enough that it does not vary significantly over the running time of the CRN. In this situation, we can approximate the implementation CRN by a simplified CRN with all fuel species removed; e.g. if g_1 is a fuel, the reaction $x_A + g_1 \rightarrow i_A$ can be replaced by $x_A \rightarrow i_A$ with no loss of meaning.

Figure 1 gives an example of this process for the formal reaction $A + B \rightarrow C + D$, yielding an implementation CRN with four reactions. (Names such as x_A



$$x_A + g_1 \rightleftharpoons i_A + f_A$$
$$i_A + x_B \rightarrow t_{CD} + w_1$$
$$t_{CD} + g_2 \rightarrow x_C + x_D + w_2$$

$$x_A \rightleftharpoons i_A$$
$$i_A + x_B \rightarrow t_{CD} + w_1$$
$$t_{CD} \rightarrow x_C + x_D + w_2$$

Fig. 1. Implementation of $A + B \rightarrow C + D$ using the scheme described in [16]. Left: DNA complexes and reactions. Top right: Direct translation of reactions in the implementation CRN. Bottom right: Implementation CRN after removing fuels.

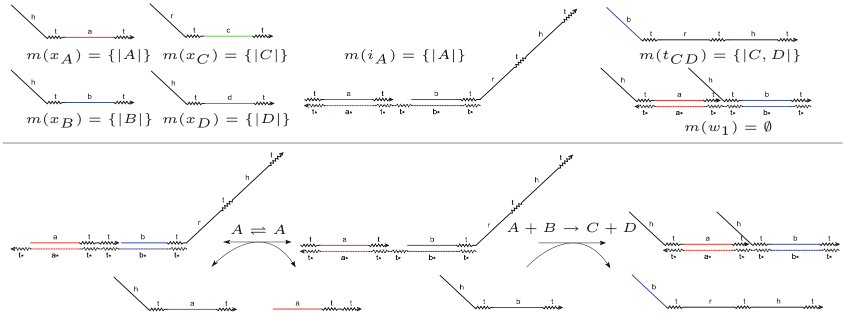


Fig. 2. Interpretation of the implementation CRN in Fig. 1. $m(t_{CD}) = A + B$ would also be a valid interpretation for this CRN.

and t_{CD} are based on the intent of the designers of the CRN, but the subscripts have no theoretical meaning.) The signal species x_A can freely convert to and from i_A, and the strand t_{CD} can produce the signals x_C and x_D (and waste w_2). Intuitively, i_A *is* an A and t_{CD} *is* a C and a D; in this sense the first and third reactions are silent, and the second *is* $A + B \to C + D$. We formalize this by defining an *interpretation* of the implementation species (Fig. 2):

Definition 1. *An* interpretation *is a function* $m : \mathcal{S}' \to \mathbb{N}^{\mathcal{S}}$ *from implementation species to multisets of formal species. We extend this linearly from species to states:* $m(\sum_{i=1}^{n} a_i X_i) = \sum_{i=1}^{n} a_i m(X_i)$*. We also define* $m(R' \to P') = m(R') \to m(P')$ *unless* $m(R') = m(P')$*, in which case* $m(R' \to P') = \tau$ *and we say the reaction is* trivial. *For example, if* $m(i_{AB}) = A+B$, $m(x_A) = A$*, and* $m(t_{BC}) = B+C$ *then* $m(i_{AB}+x_A) = 2A+B$*, and* $m(i_{AB} \to x_A+t_{BC}) = A+B \to A+B+C$.

The interpretation of an implementation reaction is always a pair (R, P) of multisets of formal species, or τ, but (R, P) may not be in $\mathcal{R}$. Any such pair is a *reaction in the language of the formal CRN*, but is a *formal reaction* only if $(R, P) \in \mathcal{R}$. Similarly, (R', P') is an *implementation reaction* only if it is in $\mathcal{R}'$.

In the following notation, S', T', S'', and T'' refer to implementation states; S and T to formal states; r' to an implementation reaction; and r to a reaction in the language of the formal CRN or τ. When a formal reaction r takes state S to state T, we write $S \xrightarrow{r} T$; $S' \xrightarrow{r'} T'$ is similar. Note that if $S \xrightarrow{r} T$, then $r = (R, P) \in \mathcal{R}$ as well as $S-R+P = T$, and analogously for the implementation. Further, we write $S' \xrightarrow{r} T'$ when $S' \xrightarrow{r'} T'$ for some r' with $m(r') = r$, which does not require $r \in \mathcal{R}$ (but does require $r' \in \mathcal{R}'$). Note that if $S' \xrightarrow{\tau} T'$ then $m(S') = m(T')$. To abstract away from trivial reactions, we write $S' \overset{\tau}{\Rightarrow} T'$ to mean S' can reach T' via 0 or more trivial reactions, and $S' \overset{r'}{\Rightarrow} T'$ when $S' \overset{\tau}{\Rightarrow} S'' \xrightarrow{r'} T'' \overset{\tau}{\Rightarrow} T'$. Note that $S' \overset{\tau}{\Rightarrow} S'$ and $S \overset{\tau}{\Rightarrow} S$ are always true. $S' \overset{r}{\Rightarrow} T'$ for $r \neq \tau$ is again defined as $S' \overset{r'}{\Rightarrow} T'$ for some r' with $m(r') = r$. $S \overset{r}{\Rightarrow} T$ for $r \neq \tau$ is defined but trivial: $S \overset{r}{\Rightarrow} T \iff S \xrightarrow{r} T$. When the final state is irrelevant, we sometimes write $S' \overset{r'}{\Rightarrow}$, etc.

3.2 Three Notions of Correctness

Our notion of correctness is motivated by the earlier observation that the set of valid trajectories defines equivalence between formal CRNs, and allowing renaming of species defines isomorphism. Applying this notion to an implementation CRN with an interpretation introduces two difficulties. First, due to trivial reactions, the implementation trajectory may involve more steps. This is easily solved by defining the interpretation of an implementation trajectory to remove trivial reactions. Second, and more seriously, the full set of interpreted implementation trajectories may cover the formal trajectories, yet particular implementation trajectories may experience restricted options for alternative paths. An extreme example of this is an implementation CRN that is subject to deadlock, $(\{x_A, x_B, y_B, x_C\}, \{x_A \to x_B, x_A \to y_B, x_B \to x_C, x_C \to x_A\})$ with the interpretation $m = \{(x_A, A), (x_B, B), (y_B, B), (x_C, C)\}$, which has the same interpreted trajectories as the formal CRN $(\{A, B, C\}, \{A \to B, B \to C, C \to A\})$, which cannot deadlock. To resolve this issue, we need a finer-grained notion of trajectory equivalence that requires equivalence given any initial state. As defined formally below, this is a satisfactory definition of correctness.

However, since the sets of trajectories are generally infinite, we would like a more local definition that facilitates efficient computational analysis. We define three local conditions on the interpretation which we show are equivalent to trajectory equivalence. As further evidence that our notion of correctness is sound, we show that these three conditions are equivalent to a special case of weak bisimulation from concurrency theory [11]. This gives us three notions of correctness, given a formal CRN, an implementation CRN, and an interpretation:

I **Equivalence of trajectories**
- (i) The set of formal trajectories and interpretations of implementation trajectories are equal.
- (ii) For every implementation state S', the set of formal trajectories starting from $m(S')$ and interpretations of implementation trajectories starting from S' are equal.

II **Three conditions on the interpretation**
- (i) *Atomic condition:* For every formal species A, there exists an implementation species x_A such that $m(x_A) = \{|A|\}$.
- (ii) *Delimiting condition:* The interpretation of any implementation reaction is either trivial or a valid formal reaction.
- (iii) *Permissive condition:* If $S \xrightarrow{r}$ and $m(S') = S$, there exists an implementation reaction r' such that $m(r') = r$ and $S' \overset{r'}{\Rightarrow}$.

III **Weak bisimulation**
- (i) For all implementation states S', if $S' \xrightarrow{r} T'$, then $S \overset{r}{\Rightarrow} T$ where $S = m(S')$ and $T = m(T')$.
- (ii) For all formal states S, there exists S' with $m(S') = S$, and for all such S', if $S \xrightarrow{r} T$, then for some T', $S' \overset{r}{\Rightarrow} T'$ and $m(T') = T$.

A few comments are in order. It may seem that the second condition for trajectory equivalence supercedes the first, but it does not: for example, the second condition may be satisfied even if there is no implementation state S' that is interpreted as formal state S, whereas the first condition will not be satisfied in that case.

Our definition of bisimulation in CRNs is in fact a special case of Milner's definition [11] for transition systems. Milner allows an arbitrary relation between states, while we rely on an interpretation m to establish a relation between formal states and implementation states. Our definition of the interpretation enforces several restrictions that, to us, are natural and consistent with the structure of CRNs: they provide a unique interpretation for each implementation state (i.e. the interpretation is a function), subsets of an implementation state can be interpreted separately and additively combined (i.e. the function is linear), and every formal state has at least one corresponding implementation state (i.e. the interpretation is surjective). In fact, *any* relation between formal states and implementation states that is a surjective linear function is induced by some interpretation, as shown in Lemma 1. Thus, we can take advantage of the finite specification of interpretations, while not losing any generality beyond the natural restrictions that we desire. These observations justify describing our notion of bisimulation in CRNs as "surjective linear weak bisimulation".

Lemma 1. *Let $\leftrightarrow \subset \mathbb{N}^{\mathcal{S}} \times \mathbb{N}^{\mathcal{S}'}$ be a relation between formal states and implementation states. If for every implementation state S' there is exactly one formal state S such that $S \leftrightarrow S'$ (function) and for every pair of pairs $S_1 \leftrightarrow S_1'$ and $S_2 \leftrightarrow S_2'$ we have $S_1 + S_2 \leftrightarrow S_1' + S_2'$ (linearity), then there is some interpretation $m : \mathcal{S}' \to \mathbb{N}^{\mathcal{S}}$ which, when extended to implementation states $m : \mathbb{N}^{\mathcal{S}'} \to \mathbb{N}^{\mathcal{S}}$, induces that relation: $S \leftrightarrow S' \iff S = m(S')$. Furthermore, for every S there is some S' such that $S \leftrightarrow S'$ (surjectivity) iff m satisfies the atomic condition.*

Proof. Given that the relation $\leftrightarrow$ is a linear function from $\mathbb{N}^{\mathcal{S}'}$ to $\mathbb{N}^{\mathcal{S}}$, we define the interpretation to be $m(x) = S_x$ where S_x is the unique formal state such that $S_x \leftrightarrow \{|x|\}$. Now, any implementation state S' is some sum of implementation species, $S' = \sum_{x \in \mathcal{S}'} \alpha_x x$, and because we define the interpretation of a state as the sum of interpretations of species, $m(S') = \sum_{x \in \mathcal{S}'} \alpha_x m(x)$. Then by the linearity assumption on $\leftrightarrow$, $m(S') \leftrightarrow S'$. Thus, if $S = m(S')$, then $S \leftrightarrow S'$. Conversely, if $S \leftrightarrow S'$, then $S = m(S')$ because $\leftrightarrow$ is a function.

If we further assume that $\leftrightarrow$ is surjective, then in particular for each formal species A, there must be some S' such that $\{|A|\} \leftrightarrow S'$, i.e. $m(S') = \{|A|\}$. Since $m(S')$ is the sum of interpretations of species in S' and an implementation species cannot interpret to fractional or negative formal species, there must be some species $x_A \in S'$ with $m(x_A) = \{|A|\}$ (and any other species in S' interpret to $\emptyset$). Thus the atomic condition is satisfied. Conversely, if the atomic condition is satisfied, then consider an arbitrary formal state $S = \sum_{A \in \mathcal{S}} \alpha_A A$. Using linearity, let $S' = \sum_{A \in \mathcal{S}} \alpha_A x_A$, so $m(S') = S$, and thus $\leftrightarrow$ must be surjective. □

Theorem 1. *The three definitions of correctness, namely trajectory equivalence, the three conditions on the interpretation, and weak bisimulation, are equivalent.*

Proof. We show that trajectory equivalence implies the three conditions formulation; the three conditions imply weak bisimulation; and weak bisimulation implies trajectory equivalence.

Given trajectory equivalence, we prove the three conditions on m. First, for the atomic condition, consider applying condition I.(i) of trajectory equivalence to formal trajectories of length 0, which are just formal states, and in particular formal states $S_A = \{|A|\}$ for each formal species A. That the set of trajectories are equal implies that there is an implementation trajectory whose interpretation is the (zero-length trajectory) state S_A, i.e. an implementation state S'_A with $m(S'_A) = \{|A|\}$. Then as in Lemma 1, there is some species $x_A \in S'_A$ with $m(x_A) = \{|A|\}$, satisfying the atomic condition. For the delimiting condition, consider implementation trajectories of length 1, specifically for each implementation reaction $r' = (R', P')$ the trajectory $R' \xrightarrow{r'} P'$. If r' is trivial, that is $m(r') = \tau$, its interpreted trajectory is a zero-length trajectory; if not, its interpreted trajectory is $m(R') \xrightarrow{m(r')} m(P')$, which by trajectory equivalence must be a formal trajectory. For that to be so, $m(r')$ must be a reaction in $\mathcal{R}$, thus satisfying the delimiting condition. For the permissive condition, for every formal reaction $r = (R, P)$ and implementation state S' with $m(S') \geq R$, the trajectory $m(S') \xrightarrow{r} T$, where $T = m(S') - R + P$, is a formal trajectory. By condition I.(ii) of trajectory equivalence, there is an implementation trajectory starting in S' whose interpreted trajectory is $m(S') \xrightarrow{r} T$. (Note that condition I.(i) implies this for *some* S' with $m(S') = S$, but not necessarily for every S'.) To have that interpretation, that implementation trajectory must have some reaction r' with $m(r') = r$ and all other reactions trivial; this is the definition of $S' \overset{r}{\Rightarrow}$, satisfying the permissive condition.

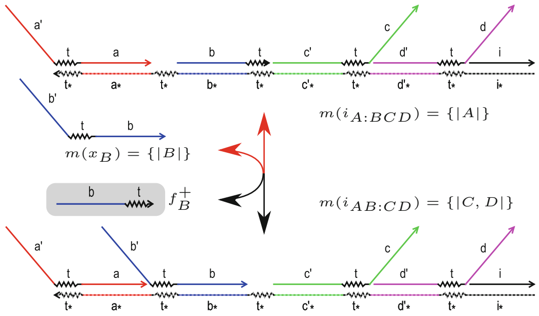
Given the three conditions, we prove weak bisimulation. Given any S' with $m(S') = S$ and $S' \xrightarrow{r'} T'$ where $r' = (R', P')$, by the delimiting condition either $m(r') = \tau$ is trivial or $m(r') = r = (R, P) \in \mathcal{R}$. If trivial, then $m(T') = m(S') = S$ and $S \overset{\tau}{\Rightarrow} S$ is true by convention. If nontrivial, then $r \in \mathcal{R}$; since $S' \xrightarrow{r'}$ we must have $S' \geq R'$, thus $m(S') \geq m(R') = R$, and $S \xrightarrow{r} T$ (therefore $S \overset{r}{\Rightarrow} T$) where $T = S - R + P$. Since $T' = S' - R' + P'$, $m(T') = m(S') - m(R') + m(P') = T$, satisfying condition III.(i) of weak bisimulation. Given any S, by Lemma 1 the atomic condition implies there exists an S' with $m(S') = S$. Given any such S' with $S \xrightarrow{r} T$ where $r = (R, P)$, by the permissive condition there is some r' with $m(r') = r$ and $S' \overset{r'}{\Rightarrow}$, which is an abbreviation for $\exists_{T'} S' \overset{r'}{\Rightarrow} T'$, which is further an abbreviation for $\exists_{S''} S' \overset{\tau}{\Rightarrow} S'' \xrightarrow{r'} T'$. (Strictly speaking $S' \overset{r'}{\Rightarrow} T'$ means there is some $S' \overset{\tau}{\Rightarrow} S'' \xrightarrow{r'} T'' \overset{\tau}{\Rightarrow} T'$, but since we are choosing an arbitrary T' we can take $T' = T''$.) Then $m(S') = m(S'') = S$ since they are connected by trivial reactions, and where $r' = (R', P')$ with $m(R') = R$ and $m(P') = P$ we have $T' = S'' - R' + P'$ so $m(T') = S - R + P = T$, satisfying condition III.(ii) of weak bisimulation.

Given weak bisimulation, we prove trajectory equivalence. We first prove condition I.(ii). Given any S'_0 with $S_0 = m(S'_0)$ and any implementation trajectory $(S'_0, r'_1, \dots, r'_k, \dots)$ with $r'_k = (R'_k, P'_k)$, let $S'_k = S'_{k-1} - R'_k + P'_k = S'_0 - \sum_{i \le k} R'_i + \sum_{i \le k} P'_k$. Letting $S_k = m(S'_k)$ and $r_k = m(r'_k)$, it follows that either $r_k = \tau$ or else $r_k = (R_k, P_k)$ and $S_k = S_{k-1} - R_k + P_k = S_0 - \sum_{i \le k} R_k + \sum_{i \le k} P_k$ by linearity of m. From bisimulation, since each $S'_{k-1} \xrightarrow{r'_k} S'_k$ we have either $r_k = \tau$ and $S_{k-1} = S_k$, or $r \neq \tau$ and $S_{k-1} \xrightarrow{r_k} S_k$, since for $r \neq \tau$ in the formal CRN $S \overset{r}{\Rightarrow} T \iff S \xrightarrow{r} T$. The interpretation of that implementation trajectory is exactly S_0 followed by those reactions $S_{k-1} \xrightarrow{r_k} S_k$ for which $r_k \neq \tau$, and thus the interpretation is a formal trajectory. Conversely, given S'_0 with $S_0 = m(S'_0)$ and any formal trajectory $(S_0, r_1, \dots, r_k, \dots)$ with $r_k = (R_k, P_k)$, letting $S_k = S_{k-1} - R_k + P_k = S_0 - \sum_{i \le k} R_k + \sum_{i \le k} P_k$, we construct an implementation trajectory whose interpretation is that formal trajectory. Given S'_0, define inductively S'_k and r'_k to be an implementation state and reaction such that $S'_{k-1} \overset{r'_k}{\Longrightarrow} S'_k$ with $m(r'_k) = r_k$ and $m(S'_k) = S_k$, which exists by condition III.(ii) of weak bisimulation. Expanding each $\overset{r'_k}{\Longrightarrow}$ implicitly defines an implementation trajectory $(S'_0, r''_{1,1}, \dots, r''_{1,l_1}, r'_1, r''_{2,1}, \dots)$ where each $m(r''_{k,j}) = \tau$ and each $m(r'_k) = r_k$; the interpretation of this trajectory is the formal trajectory $(S_0, r_1, \dots, r_k, \dots)$ as desired, proving condition I.(ii). Condition I.(i) follows from condition I.(ii) of trajectory equivalence and condition III.(ii) of weak bisimulation: every implementation trajectory starts from some S' and by condition I.(ii) its interpretation must be a formal trajectory starting from $m(S')$. Conversely, every formal trajectory starts from some S, by condition III.(ii) of weak bisimulation there is some S' with $m(S') = S$, and by condition I.(ii) of trajectory equivalence there is an implementation trajectory starting from S' whose interpretation is that formal trajectory. □

3.3 Applying Bisimulation

We now consider how to use bisimulation to analyze our example implementation of $A + B \rightarrow C + D$. We use the three conditions formulation. The atomic condition is satisfied by the "signal species" x_A, x_B, x_C, and x_D. For the delimiting condition, we check each implementation reaction individually: $i_A + x_B \rightarrow t_{CD} + w_1$ is interpreted as $A + B \rightarrow C + D$, which is formal, while $x_A \rightleftharpoons i_A$ and $t_{CD} \rightarrow x_C + x_D + w_2$ are trivial. The permissive condition says that for every formal reaction and for every implementation state in which that reaction should be able to happen, it can. There is one formal reaction, $A + B \rightarrow C + D$, and any state in which it should be able to happen must contain an x_B and either an x_A or i_A, since those are the only species whose interpretations contain either B and/or A. If the state contains x_B and i_A, then the reaction $i_A + x_B \rightarrow t_{CD} + w_1$ can happen and satisfies the permissive condition. If the state contains x_B and x_A, then the trivial reaction $x_A \rightarrow i_A$ followed by $i_A + x_B \rightarrow t_{CD} + w_1$ satisfies the permissive condition.

Now consider a different case. Figure 3 shows an implementation of $A + B \rightarrow C + D$ as described by Qian et al. [12] as a means to implement stack machines, along with a natural interpretation. The species $i_{AB:CD}$ is interpreted as $C+D$, while $i_{A:BCD}$ is interpreted as A and x_B as B. This makes the reaction $i_{AB:CD} \rightarrow i_{A:BCD}+x_B$ interpreted as $C+D \rightarrow A+B$, which is not a valid formal reaction. Thus the delimiting condition is unsatisfied, and the implementation is not correct according to bisimulation; any other interpretation will have the same problem. This is not a problem for deterministic stack machines, but it does identify an error with this as a translation scheme for arbitrary CRNs: if the reaction $A+B \rightarrow C+D$ were put together with a reaction $C \rightarrow C+E$, then it would be possible to go from $\{|A, B|\}$ to $\{|A, B, E|\}$ in the implementation CRN when it is not possible in the formal CRN.



$$x_A + f_{ABCD} \rightleftharpoons i_{A:BCD} + f_A^+$$
$$x_B + i_{A:BCD} \rightleftharpoons i_{AB:CD} + f_B^+$$
$$i_{AB:CD} + f_C^- \rightleftharpoons i_{ABC:D} + x_C$$
$$i_{ABC:D} + f_D^- \rightleftharpoons i_{ABCD:} + x_D$$
$$i_{ABCD:} + f_i \rightarrow w_{ABCD}$$

Fig. 3. The translation scheme from [12], when used as a general CRN implementation, violates the delimiting condition. Species named f are fuels.

3.4 Properties of CRN Bisimulation

We describe two properties of CRN bisimulation that are likely to be useful in analyzing larger systems. While bisimulation in the classic sense is an equivalence relation between systems [11], our definition of interpretation-dependent CRN bisimulation is a partial order on the set of CRNs. In particular, CRN bisimulation is transitive, which allows us to do complex proofs of correctness in stages. We also show a modularity condition, where the combination of interpretations can be verified using only properties of each individual interpretation. This is particularly useful for general translation schemes where the translation of a whole CRN is the combination of one "module" for each reaction. As an example, we use modularity to prove that the translation scheme in [16] is correct for any CRN.

We first show that CRN bisimulation is transitive. Consider three CRNs: an abstract CRN $(\mathcal{S}, \mathcal{R})$, an implementation CRN $(\mathcal{S}'', \mathcal{R}'')$, and an intermediate CRN $(\mathcal{S}', \mathcal{R}')$. For example, $(\mathcal{S}, \mathcal{R})$ is an abstract CRN, $(\mathcal{S}'', \mathcal{R}'')$ is a low-level reaction enumeration of a prospective DNA implementation of $(\mathcal{S}, \mathcal{R})$, and $(\mathcal{S}', \mathcal{R}')$ is a more high-level reaction enumeration of the same DNA implementation which abstracts away from certain details. Say we have proven that $(\mathcal{S}', \mathcal{R}')$ is a valid implementation of $(\mathcal{S}, \mathcal{R})$ by finding an interpretation $m_1 : \mathcal{S}' \to \mathbb{N}^{\mathcal{S}}$ which is a bisimulation, and similarly have found an interpretation $m_2 : \mathcal{S}'' \to \mathbb{N}^{\mathcal{S}'}$ which is a bisimulation from $(\mathcal{S}'', \mathcal{R}'')$ to $(\mathcal{S}', \mathcal{R}')$. We want to prove that $(\mathcal{S}'', \mathcal{R}'')$, the system we actually have, is a valid implementation of $(\mathcal{S}, \mathcal{R})$, the system we want. The natural interpretation $m : \mathcal{S}'' \to \mathbb{N}^{\mathcal{S}}$ is $m(x) = (m_1 \circ m_2)(x) = m_1(m_2(x))$, treating m_2 as a function of species and m_1 as extended to a function of states. It turns out that this interpretation is in fact a bisimulation.

Lemma 2 *(Transitivity). If m_2 is a bisimulation from $(\mathcal{S}'', \mathcal{R}'')$ to $(\mathcal{S}', \mathcal{R}')$ and m_1 is a bisimulation from $(\mathcal{S}', \mathcal{R}')$ to $(\mathcal{S}, \mathcal{R})$, then $m = m_1 \circ m_2$ is a bisimulation from $(\mathcal{S}'', \mathcal{R}'')$ to $(\mathcal{S}, \mathcal{R})$.*

Proof. We use the three conditions formulation of correctness. We refer to $(\mathcal{S}, \mathcal{R})$ as the "formal" CRN, $(\mathcal{S}'', \mathcal{R}'')$ as the "implementation" CRN, and $(\mathcal{S}', \mathcal{R}')$ as the "intermediate" CRN. We show that each condition for m follows from the corresponding conditions for m_1 and m_2.

For any formal species A, by the atomic conditions for m_1 and m_2 there is an intermediate species x_A with $m_1(x_A) = \{|A|\}$ and implementation species y_A with $m_2(y_A) = x_A$. Then $m(y_A) = m_1(m_2(y_A)) = m_1(\{|x_A|\}) = \{|A|\}$, thus m satisfies the atomic condition.

For any implementation reaction $r'' = R'' \to P''$, by the delimiting condition for m_2 its interpretation $m_2(r'')$ is either an intermediate reaction $R' \to P' \in \mathcal{R}'$ or is τ. If $m_2(r'') = \tau$, that means $m_2(R'') = m_2(P'')$ and $m(R'') = m_1(m_2(R'')) = m_1(m_2(P'')) = m(P'')$, so $m(R'' \to P'') = m(r'') = \tau$. If $m_2(r'') = R' \to P'$ is a valid intermediate reaction, then $m(r'') = m_1(R' \to P')$, which by the delimiting condition for m_1 is either a valid formal reaction or trivial.

For any formal state S and reaction r with $S \xrightarrow{r}$ and any implementation state S'' with $m(S'') = S$, that means $S' = m_2(S'')$ is an intermediate state with $m_1(S') = S$. By the permissive condition on m_1, there is some r' with $m_1(r') = r$ and $S' \overset{r'}{\Rightarrow}$. Using the permissive condition on m_2 and the argument used in Theorem 1 to show that the permissive condition implies trajectory equivalence, there is a sequence of implementation reactions starting from S'' which implements the intermediate trajectory by which $S' \overset{r'}{\Rightarrow}$. This means that one of those reactions r'' has $m_2(r'') = r'$, some of them interpret via m_2 to various intermediate reactions in that pathway which are trivial under m_1, and the rest of which are trivial under m_2. An implementation reaction trivial under m_2 is trivial under m, as is a reaction which interprets under m_2 to an intermediate reaction trivial under m_1, thus all reactions in this pathway except r'' are trivial under m, so when viewed under m, $S'' \overset{r}{\Rightarrow}$. □

Bisimulation in the classic sense is an equivalence relation on states, which can be extended to an equivalence relation on systems [11]. Our definition of weak bisimulation introduces an asymmetry – one implementation state can only correspond to one formal state, but multiple implementation states can correspond to the same formal state. Bisimulation assumes a set of states and transitions between states where each transition is labelled from a common set of labels, while CRNs do not come with an obvious concept of labels. Our definition implicitly uses the set of all possible reactions using species in the formal CRN (plus the silent τ) as labels, labeling each formal reaction with itself and each implementation reaction with its interpretation. In that context, Lemma 1 and Theorem 1 say that there is an interpretation m which is a CRN bisimulation (satisfies the three conditions, has trajectory equivalence) if and only if there is a relation between states of that system which is a surjective and linear function from implementation states to formal states and is a bisimulation in the sense of [11]. Since we require one CRN to be designated the "formal" CRN in order to define a set of labels, it is difficult to make the concept of a symmetric relation between CRNs meaningful. Instead, CRN bisimulation is an order relation (up to isomorphism):

Lemma 3 *(Partial order). The following relation is a partial order:* $(\mathcal{S}', \mathcal{R}') \gtrsim (\mathcal{S}, \mathcal{R})$ *if there exists an* $m : \mathcal{S}' \to \mathbb{N}^{\mathcal{S}}$ *which satisfies the atomic, delimiting, and permissive conditions (equivalently, its extension* $m : \mathbb{N}^{\mathcal{S}'} \to \mathbb{N}^{\mathcal{S}}$ *is a surjective linear weak bisimulation) with equality defined as* $(\mathcal{S}', \mathcal{R}') \equiv (\mathcal{S}, \mathcal{R})$ *if there exists a bijection* $n : \mathcal{S}' \to \mathcal{S}$ *such that* $(n(\mathcal{S}'), n(\mathcal{R}')) = (\mathcal{S}, \mathcal{R})$ *where* n *is extended naturally to sets and reactions.*

Proof. A partial order must be transitive, reflexive, anti-symmetric. Transitivity (if $a \leq b$ and $b \leq c$ then $a \leq c$) follows immediately from Lemma 2. Relexivity ($a \leq a$) is obvious by letting m be the identity function. It remains to show anti-symmetry (if $a \leq b$ and $b \leq a$, then $a = b$), i.e. that given $(\mathcal{S}_1, \mathcal{R}_1)$ and $(\mathcal{S}_2, \mathcal{R}_2)$ with $m_1 : \mathcal{S}_1 \to \mathbb{N}^{\mathcal{S}_2}$ and $m_2 : \mathcal{S}_2 \to \mathbb{N}^{\mathcal{S}_1}$ that each satisfy the atomic, delimiting, and permissive conditions, $(\mathcal{S}_1, \mathcal{R}_1)$ and $(\mathcal{S}_2, \mathcal{R}_2)$ are identical up to a change of species names. The atomic condition implies that $|\mathcal{S}_1| \leq |\mathcal{S}_2|$ and $|\mathcal{S}_2| \leq |\mathcal{S}_1|$, thus the numbers of species are equal and in particular m_1 is a bijection from species in $\mathcal{S}_1$ to sets of exactly one species in $\mathcal{S}_2$ (and the same is true for m_2). To simplify notation, we let $n(x) = y$ if $m_1(x) = \{|y|\}$; n must be a bijection from $\mathcal{S}_1$ to $\mathcal{S}_2$. (If the CRN has sufficient symmetry, it is not necessarily true that $m_2(n(x)) = \{|x|\}$, for example if both CRNs are $\{A \to C, B \to C\}$ we could have $m_2(n(A)) = \{|B|\}$.) Since n is a bijection, any reaction that would be trivial after interpretation (by either m_1 or m_2) must be trivial before interpretation, and thus cannot exist. By the delimiting condition for m_1, every reaction in $\mathcal{R}_1$ must have its image under n in $\mathcal{R}_2$; by the permissive condition for m_1, every reaction in $\mathcal{R}_2$ must have its preimage under n in $\mathcal{R}_1$; thus the two CRNs are equal up to the isomorphism n. □

In Sect. 3.3 we showed that the translation scheme from [16] is a correct implementation of the single reaction $A + B \to C + D$ according to CRN bisimulation. Intuitively, given a CRN of multiple reactions we should be able to combine the implementations of each such reaction to form a correct implementation of the CRN. In particular, we would like to show that the combined implementation CRN is correct using a condition which we can check on each individual reaction's implementation without having to check any property of the combined CRN. Since, as we will see in Sect. 4, the time required to check an interpretation scales much worse than linearly in the size of the implementation CRN, such a modularity condition would be a significant saving in the time required. While it is not in general true that combining two correct implementation CRNs gives a correct implementation of the combined formal CRN, there is a modularity condition which guarantees that the combined CRN is correct.

We consider an implementation CRN $(\mathcal{S}'_1, \mathcal{R}'_1)$ and formal CRN $(\mathcal{S}_1, \mathcal{R}_1)$ with interpretation $m_1 : \mathcal{S}'_1 \to \mathbb{N}^{\mathcal{S}_1}$, and another implementation CRN $(\mathcal{S}'_2, \mathcal{R}'_2)$ and formal CRN $(\mathcal{S}_2, \mathcal{R}_2)$ with interpretation $m_2 : \mathcal{S}'_2 \to \mathbb{N}^{\mathcal{S}_2}$, where both m_1 and m_2 are bisimulations. We assume the interpretations are compatible: for each $x \in \mathcal{S}'_1 \cap \mathcal{S}'_2$, $m_1(x) = m_2(x)$, which implies $m_1(x) \in \mathbb{N}^{\mathcal{S}_1 \cap \mathcal{S}_2}$. We also assume that the reactions in $\mathcal{R}'_1$ and $\mathcal{R}'_2$ are the only reactions that occur when you combine the implementation species in $\mathcal{S}'_1$ and $\mathcal{S}'_2$; that is, we assume no crosstalk reactions. Whether there is crosstalk can be checked by a reaction enumerator [8,10]. Aside from crosstalk, the main reason for the combined implementation to be incorrect according to bisimulation is some implementation species y in e.g. $\mathcal{S}'_1$ but not in $\mathcal{S}'_2$ whose interpretation contains a formal species $A \in \mathcal{S}_1 \cap \mathcal{S}_2$, where some formal reaction in $\mathcal{R}_2$ with A as a reactant cannot be implemented from an implementation state where y is the representation of A. If any such species y can, via trivial reactions, "release" any formal species in $\mathcal{S}_1 \cap \mathcal{S}_2$ in its interpretation to implementation species in $\mathcal{S}'_1 \cap \mathcal{S}'_2$, then we would think this problem cannot arise. This condition can be checked individually on each module without checking the combined CRN, and we show that this condition guarantees that the combined implementation is correct according to bisimulation.

Theorem 2 *(Modularity). Let m_1 be a bisimulation from $(\mathcal{S}'_1, \mathcal{R}'_1)$ to $(\mathcal{S}_1, \mathcal{R}_1)$ and m_2 be a bisimulation from $(\mathcal{S}'_2, \mathcal{R}'_2)$ to $(\mathcal{S}_2, \mathcal{R}_2)$ where m_1 and m_2 agree on $\mathcal{S}'_1 \cap \mathcal{S}'_2$. Let $\mathcal{S}' = \mathcal{S}'_1 \cup \mathcal{S}'_2$, $\mathcal{R}' = \mathcal{R}'_1 \cup \mathcal{R}'_2$, $\mathcal{S} = \mathcal{S}_1 \cup \mathcal{S}_2$, and $\mathcal{R} = \mathcal{R}_1 \cup \mathcal{R}_2$, and $m : \mathcal{S}' \to \mathbb{N}^{\mathcal{S}}$ equal m_1 on $\mathcal{S}'_1$ and m_2 on $\mathcal{S}'_2$. If for any $x \in \mathcal{S}'$ there is a sequence of trivial reactions $x \overset{\mathcal{T}}{\Rightarrow} Y + Z$ for $Y \in \mathbb{N}^{\mathcal{S}'_1 \cap \mathcal{S}'_2}$ and $m(Z) \cap (\mathcal{S}_1 \cap \mathcal{S}_2) = \emptyset$, then m is a bisimulation from $(\mathcal{S}', \mathcal{R}')$ to $(\mathcal{S}, \mathcal{R})$.*

Proof. We use the three conditions formulation. The atomic condition for m for each formal species A is satisfied by the species x_A that satisfy it for m_1 or m_2, as appropriate, or possibly both; e.g. if $A \in \mathcal{S}_1$ then there is some species $x_A \in \mathcal{S}'_1$ such that $m_1(x_A) = \{|A|\}$, which implies that $x_A \in \mathcal{S}'$ and $m(x_A) = m_1(x_A) = \{|A|\}$. Similarly the delimiting condition for m follows from that for m_1 and m_2: for any implementation reaction $R' \to P'$ in $\mathcal{S}'$, that reaction is in either $\mathcal{R}'_1$ or $\mathcal{R}'_2$, and its interpretation in m agrees with its interpretation in either m_1 or m_2 as appropriate, which is either a trivial reaction or a formal reaction in $\mathcal{R}_1$ or $\mathcal{R}_2$, which is thus in $\mathcal{R}$. (The delimiting condition assumes, as we mentioned above, that no crosstalk reactions exist, which when applying this theory to DNA implementations would be checked by a reaction enumerator.)

For the permissive condition, consider a formal reaction $r = R \rightarrow P$ and implementation state S' where $R \leq m(S')$. Either $r \in \mathcal{R}_1$ or $r \in \mathcal{R}_2$; without loss of generality say $r \in \mathcal{R}_1$. Divide S' into species in the first CRN and species not: let $S' = S'_1 + S'_2$, where $S'_1 \subset \mathcal{S}'_1$ and $S'_2 \cap \mathcal{S}'_1 = \emptyset$. If $m(S'_1) \geq R$, then the permissive condition for m_1 applied to reaction r and state S'_1 mean $S'_1 \overset{r}{\Rightarrow}$, thus $S' \overset{r}{\Rightarrow}$ by the same sequence of reactions ignoring species in S'_2. In the general case, this means the proof is nontrivial only for formal species in R whose implementations in S' are in S'_2, and we need to show that those formal species can be "extracted" into an implementation species in $\mathcal{S}'_1$. This is exactly the modularity condition: for each species $x_i \in S'_2$ there is a sequence of trivial reactions by which $x_i \overset{\mathcal{T}}{\Rightarrow} Y_i + Z_i$, where $Y_i \subset \mathcal{S}'_1$ and $m(Z_i) \cap \mathcal{S}_1 = \emptyset$. In particular, since $R \rightarrow P$ is a reaction in CRN 1, $R \subset \mathcal{S}_1$ and $R \cap m(Z_i) = \emptyset$. We then have $S' \overset{\mathcal{T}}{\Rightarrow} S'_1 + Y + Z$, where $Y = \sum_i Y_i \subset \mathcal{S}'_1$ and $Z = \sum_i Z_i$. Since $R \cap Z = \emptyset$, $R \leq m(S')$, and $m(S') = m(S'_1 + Y) + m(Z)$, we have $R \leq m(S'_1 + Y)$. Since $S'_1 + Y \subset \mathcal{S}'_1$, the permissive condition for m_1 implies $S'_1 + Y \overset{r}{\Rightarrow}$, thus $S' \overset{r}{\Rightarrow}$. □

DNA implementation schemes for arbitrary CRNs such as [2,12,16] typically have a set of common species and for each formal reaction a "module" with additional species and implementation reactions that implement the formal reaction. If the modules have no crosstalk and each one correctly implements its reaction and satisfies the modularity condition, then repeated applications of Theorem 2 prove that the entire CRN is a correct implementation.

Corollary 1. *Consider a formal CRN $(\mathcal{S}, \mathcal{R})$ with n reactions $\mathcal{R} = \{r_i\}_{i=1}^n$, and n implementation "module" CRNs $(\mathcal{S}'_0 \cup \mathcal{S}'_i, \mathcal{R}'_i)$ with species $\mathcal{S}'_0$ in common, where any $\mathcal{S}'_i$ is disjoint from any $\mathcal{S}'_j$ for $j \neq i$. If there are interpretations $m_i : \mathcal{S}'_i \rightarrow \mathcal{S}$ for $0 \leq i \leq n$ such that the interpretation $(m_0 \cup m_i)$ is a bisimulation from $(\mathcal{S}'_0 \cup \mathcal{S}'_i, \mathcal{R}'_i)$ to $(\mathcal{S}, \{r_i\})$, and any $x \in \mathcal{S}'_i$ can be converted $x \overset{\mathcal{T}}{\Rightarrow} Y + Z$ by trival reactions in $\mathcal{R}'_i$ where $Y \in \mathbb{N}^{\mathcal{S}'_0}$ and $m(Z) = \emptyset$, where $m = \bigcup_{i=1}^n m_i$ is the combination of the interpretations, then m is a bisimulation from $(\mathcal{S}'_0 \cup \bigcup_{i=1}^n \mathcal{S}'_i, \bigcup_{i=1}^n \mathcal{R}'_i)$ to $(\mathcal{S}, \mathcal{R})$.*

In particular, the translation scheme from [16] discussed earlier satisfies the condition in Corollary 1 for $\mathcal{S}'_0 = \{x_A \mid A \in \mathcal{S}\}$, i.e. the signal species. Thus Corollary 1 proves that for any number of formal reactions, the scheme in [16] produces a correct implementation CRN, as long as the DSD reaction enumerator produces exactly the described reactions and no additional crosstalk reactions.

4 Checking Bisimulation

We now have a definition of "correct implementation", and can sometimes prove that a particular implementation is or is not correct. We would like to find a general way to check whether any implementation is correct.

We divide "checking bisimulation" into three questions. First, given a formal and implementation CRN and an interpretation, is the interpretation a bisimulation? Second, if (as in most engineered CRN implementations) we have a formal CRN, implementation CRN, and for each formal species A a designated signal species x_A, is there an interpretation which is a bisimulation and has $m(x_A) = \{|A|\}$? Finally, given a formal CRN, implementation CRN, and no additional information, is there an interpretation which is a bisimulation?

We give the complexity in terms of two parameters: the *size* n, the total number of species and reactions in the two CRNs, and the *arity* k, the maximum number of reactants in any formal reaction. We find the problem is easier when k is bounded by a constant, such as $k = 2$ limiting the formal CRN to bimolecular reactions.

4.1 Checking an Interpretation

First we consider the problem of, given an interpretation, checking whether it is a bisimulation. We use the three conditions on an interpretation, having proved in Theorem 1 that they are equivalent to bisimulation and trajectory equivalence. Given two CRNs and an interpretation between them, the atomic and delimiting conditions are trivial to check. This leaves only the permissive condition.

Checking the permissive condition means, for each formal reaction $r = (R, P)$ and implementation state S' with $m(S') \geq R$, S' can reach via trivial reactions some state from which a reaction that is interpreted as r can happen. Although there are infinitely many such S', we only need to consider the minimal such states. Consider two such states S' and S'' where $S'' > S'$ and $m(S'') \geq m(S') \geq R$. If there is some sequence of reactions by which $S' \overset{r}{\Rightarrow}$, then it can happen in S'' also and thus $S'' \overset{r}{\Rightarrow}$. If not, then the permissive condition is false for S', and we do not need to check S''. Thus we need only to check states S' such that $m(S') \geq R$ and there is no $S'_0 < S'$ for which $m(S'_0) \geq R$, which we call *minimal* implementation states (with respect to the given formal reaction r). All such minimal states can be enumerated by, for each reactant X_i in R, choosing some implementation species x_i such that $X_i \in m(x_i)$, removing $m(x_i)$ from R (ignoring any species in $m(x_i)$ not present in R), then applying this process recursively.

Now we have reduced the permissive condition to a finite problem: for each minimal S', check whether it can reach via trivial reactions some state T' from which a reaction r' with $m(r') = r$ can happen. By taking the reactants of each such r' in the implementation CRN, we have a list of multisets R' such that if for each minimal S' there is some such R' such that S' can reach a state greater than or equal to R', then the permissive condition is satisfied. This sounds similar to the covering (or superset reachability) problem: given states S' and T', can S' reach any state $T'' \geq T'$? Unfortunately, the covering problem was proven by Rackoff to be EXPSPACE-complete [13]. In particular, the covering problem is hard because to reach a given state T' from S' may require production and consumption of a large number of some species.

To solve the permissive condition in less than exponential space, we use the fact that for the permissive condition to be satisfied, we need a path from *every* minimal S' to some r' which implements r. Thus, if a minimal state $S'_0 \xrightarrow{\tau} S''$

for $S'' > S'_1$ which is minimal with $m(S'_1) \geq R$, then either $S'_1 \overset{r}{\not\Rightarrow}$ and the permissive condition is false anyway, or $S'_1 \overset{r}{\Rightarrow}$ and we can treat $S'_0 \xrightarrow{\mathcal{T}} S''$ as $S'_0 \xrightarrow{\mathcal{T}} S'_1$ ignoring the extra species. Following this logic out, we visualize the state space as a graph of minimal states with an arrow from S'_i to S'_j if there is a trivial reaction $S'_i \xrightarrow{\mathcal{T}} S'_j + \ldots$. We find that we can check the permissive condition using only paths through this graph with some loops of the form $S' \overset{\mathcal{T}}{\Rightarrow} S' + Z$ for some minimal state S' and multiset Z, which since trivial reactions do not change the interpretation implies that $m(Z) = \emptyset$, thus $m(y) = \emptyset$ for each $y \in Z$. If such a loop exists, then we know that arbitrarily many copies of each such y can be produced in state S', and we can ignore y whenever it appears as a reactant later on the path.

Lemma 4. *Let $(\mathcal{S}, \mathcal{R})$ and $(\mathcal{S}', \mathcal{R}')$ be a formal CRN and an implementation CRN, with interpretation m. Let $r = (R, P) \in \mathcal{R}$ be a formal reaction and S'_0 an implementation state minimal for $m(S'_0) \geq R$. Let z be the number of null species. If the permissive condition is satisfied, then there exists a sequence of $l \leq z$ multisets S'_i that are minimal for $m(S'_i) \geq R$, l disjoint sets Z_i of null species, and nonnegative integers $(\alpha_i)_{1 \leq i \leq l+1}$, $(\beta_i)_{1 \leq i \leq l}$ such that, where $Y_i = \bigcup_{j \leq i} Z_j$, for each $1 \leq i \leq l$ there is a sequence of trivial reactions by which $S'_{i-1} + \alpha_i Y_{i-1} \overset{\mathcal{T}}{\Rightarrow} S'_i$ and $S'_i + \beta_i Y_{i-1} \overset{\mathcal{T}}{\Rightarrow} S'_i + Z_i$, and a sequence of trivial reactions by which $S'_l + \alpha_{l+1} Y_l \overset{r}{\Rightarrow}$, where the same minimal implementation state is never covered twice within the same sequence. Conversely, if such paths exist for* every *formal reaction and minimal implementation state, then the permissive condition is satisfied.*

Proof (Sketch). If the permissive condition is satisfied, then for each minimal S', consider the first reaction on the shortest path by which $S' \overset{r}{\Rightarrow}$. Starting from any given S'_0, the pathway which at each S' takes that first reaction is a valid pathway, and either eventually implements r or eventually repeats the same minimal state S'_1 twice. If it eventually implements r, then the pathway matches the desired pathway with all sequences of trivial reactions except the last one empty. If it eventually repeats, then for each reaction to be the first reaction on the shortest path, the sequence of reactions which loops must be $S'_1 \overset{\mathcal{T}}{\Rightarrow} S'_1 + Z_1$ for Z_1 a nonempty multiset of null species. With the sequence by which $S'_0 \overset{\mathcal{T}}{\Rightarrow} S'_1$ and $S'_1 \overset{\mathcal{T}}{\Rightarrow} S'_1 + Z_1$ as the first two sequences of trivial reactions, consider a modified implementation CRN with all species in Z_1 removed; if the original implementation CRN satisfies the permissive condition, then making reactions easier cannot make the permissive condition false. Applying this by induction on the number of null species gives the remaining segments.

If such paths exist for a given formal reaction r and minimal implementation state S'_0, then $S'_0 \overset{r}{\Rightarrow}$: from S'_0 reach S'_1; produce "as many copies as needed" of Z_1 in the loop $S'_1 \overset{\mathcal{T}}{\Rightarrow} S'_1 + Z_1$, reach S'_2, produce "as many copies as needed" of Z_2, etc. If such paths exist for every formal reaction r and minimal state S'_0 for r, then every minimal state $S'_0 \overset{r}{\Rightarrow}$, thus as discussed above every state with $m(S') \xrightarrow{r}$ has $S' \overset{r}{\Rightarrow}$, thus satisfying the permissive condition. □

We describe two algorithms to check the permissive condition. One runs in space poly(nk) and time poly(n^{kn}); the other runs in space and time poly(n^k).

The space-efficient, loopsearch algorithm goes through each formal reaction (R, P) and each minimal implementation state S' in which $m(S') \geq R$ and searches for the path described in Lemma 4. It iterates through each partition of null species into sets Z_i and choice of states S'_i for which $S'_i \overset{\mathcal{T}}{\Rightarrow} S'_i + Z_i$. A nonrepeating path from S'_i to S'_j or $S'_i + Z_i$ will have length at most N, where N is the number of minimal implementation states. Where $l = \lceil \log_2 N \rceil$, Savitch's theorem [14] says that a path from S'_a to S'_b of length at most 2^l can be found by checking all possible states S'_c for a path from S'_a to S'_c and S'_c to S'_b each of length at most 2^{l-1}, which can be done recursively. This algorithm stores at most $l + z$ minimal states plus a partition of z species at any one time.

The more time-efficient, graph-updating algorithm, for each formal reaction $r = (R, P)$ iteratively builds a table of minimal implementation states S'_i with $m(S'_i) \geq R$ and, for each minimal S'_i, which other minimal S'_j can be reached from S'_i via trivial reactions and which null species can be produced in a loop from S'_i to itself. In each iteration, for each S'_i that is not yet known to be able to implement r, for each trivial reaction of the form $S'_i + Z_1 \xrightarrow{\mathcal{T}} S'_j + Y + Z_2$, where Z_1 and Z_2 contain only null species and all species in Z_1 are known to be producible in a loop from S'_i to itself, it updates the table as follows:

(i) If S'_j is known to be able to implement r, then S'_i can implement r. Otherwise:
(ii) For each $k \neq i$, if S'_j can reach S'_k, then S'_i can reach S'_k.
(iii) If S'_j can reach S'_i, then S'_i can produce in a loop any null species in Z_2, as well as any null species producible in a loop at S'_j.

The algorithm terminates when an iteration passes with no change to the table. If all states are known to be able to implement r, then the permissive condition is satisfied for r; otherwise the permissive condition is false. Using similar but slightly different reasoning as Lemma 4, we can prove that if the permissive condition is true, the algorithm will prove it in at most $(2zn^k + 1)n^k$ iterations.

Theorem 3. *Whether an interpretation is a bisimulation can be checked in polynomial space.*

Proof. The loopsearch algorithm takes polynomial space. □

Theorem 4. *When the number of reactants in a formal reaction k is constant, whether an interpretation is a bisimulation can be checked in polynomial time.*

Proof (Sketch). We show that if the permissive condition is true, the graph-updating algorithm will prove it in at most $(2zn^k + 1)n^k$ iterations. Given a formal reaction $r = (R, P)$ and all states S' which are minimal for $m(S') \geq R$, at any given iteration for some S' and y with $m(y) = \emptyset$ it will be known that $S' \overset{\mathcal{T}}{\Rightarrow} S' + y$. If the permissive condition is true, then as in the proof of Lemma 4, for each S' consider the first reaction on the shortest path by which $S' \overset{r}{\Rightarrow}$

assuming that each S'_i has infinite copies of any species y for which it is known that $S'_i \overset{\mathcal{T}}{\Rightarrow} S'_i + y$. As in that proof, these reactions either give every S' a direct path by which $S' \overset{r}{\Rightarrow}$, or have at least one loop by which some $S'_i \xrightarrow{\mathcal{T}} S'_j + y$ and $S'_j \overset{\mathcal{T}}{\Rightarrow} S'_i$ which is not already known. (If not, then at least one of the reactions involved does not lead to the shortest path.) Let $l \leq n^k$ be the number of states in that loop. If every minimal state can implement r using only known null species, then the shortest paths each have length at most n^k, and algorithm will prove this and terminate in at most n^k iterations. If there is such a loop, it will take at most l iterations to prove that $S'_i \overset{\mathcal{T}}{\Rightarrow} S'_i + y$, and an additional l iterations to prove for each other S'_j in that loop that $S'_j \overset{\mathcal{T}}{\Rightarrow} S'_j + y$. Thus in at most $2n^k$ iterations one more fact will be known. The number of such facts is at most zn^k, all possible pairs of minimal state S' and null species y. If the permissive condition is true, it will be proven in at most $(2zn^k + 1)n^k$ iterations. □

Although polynomial space in the general case is inefficient, we cannot do better. If we have (order of) n^k minimal states, it is possible to embed a PSPACE-complete computation in those n^k states. In particular, a Linear Bounded Automaton computation – a model of a Turing machine with space bounded by the size of its input, for which acceptance is a PSPACE-complete problem [7] – can be embedded in a polynomial-size implementation CRN, such that a given formal reaction is reachable in the implementation if and only if the LBA accepts.

Theorem 5. *CRN bisimulation in the general case is PSPACE-complete.*

Proof (Sketch). Consider a formal CRN with one reaction, $Q + A_1 + \ldots + A_n \rightarrow H$. An implementation CRN can simulate an arbitrary LBA with species representing the states of the LBA interpreted as Q and species representing the ith tape symbol interpreted as A_i. (For example, the reaction $q_i^0 + 0_i \rightarrow q_{i+1}^3 + 1_i$ for each $1 \leq i \leq n$ represents the Turing machine instruction, "in state 0, read a 0, write a 1, move right, go to state 3".) If the interpretation CRN can reset at any time to the starting state with the tape reading a given string x, and can implement the formal reaction only from an accepting state, then the permissive condition is true if and only if the LBA accepts the string x. □

4.2 Finding an Interpretation

We now consider the problem of, given a formal and implementation CRN, can we find an interpretation that is a bisimulation or correctly assert that none exists? An algorithm to enumerate interpretations that satisfy the delimiting condition was given in [6]. This algorithm guarantees that, if an interpretation that is a bisimulation exists, then it will enumerate at least one of them. The algorithm iterates through each possible assignment of each implementation *reaction* to be interpreted as a given formal reaction or trivial; for each assignment, iterates through each partial specification of an interpretation that satisfies the reactions assigned to be formal; then sets up the remaining trivial reactions as a system of equations and finds a minimal solution. By testing each enumerated interpretation with either of the permissive condition tests described above, this algorithm will find an interpretation or assert that none exists.

Theorem 6. *Whether a bisimulation interpretation exists from a given implementation CRN to a given formal CRN is PSPACE-complete.*

Proof (Sketch). The above algorithm runs in polynomial space, thus proving membership. The reactions that simulate a Turing machine in Theorem 5 restrict the interpretation enough that any interpretation other than the one given (up to a permutation of the formal species) will be invalid. □

Theorem 7. *When the number of reactants in a formal reaction k is bounded by a constant, whether a bisimulation interpretation exists is NP-complete.*

Proof (Sketch). If a valid interpretation exists, the above algorithm guarantees that it will produce a valid polynomial-size interpretation which can be checked in polynomial time by Theorem 4. A 3-SAT formula with clauses e.g. $(x_1 \vee \neg x_2 \vee x_3)$ can be encoded in implementation reactions e.g. $s_C \to x_1^t + x_2^f + x_3^t$. Additional reactions $s_T \rightleftharpoons x_i^t + x_i^f$ restrict interpretations that satisfy the delimiting condition to correspond to satisfying assignments of the 3-SAT formula. □

5 Discussion

Comparing Chemical Reaction Networks on different levels of abstraction is an important tool for systematic programming with CRNs. We showed how to adapt the concept of bisimulation to check whether one CRN is a correct implementation of another. We showed that bisimulation can be used to prove the correctness of some existing CRN implementations, and to identify subtle but real problems with others. We discussed transitivity and modularity, which can be used to simplify a bisimulation proof. We presented different algorithms to check bisimulation which are adapted to different cases. We showed that the condition can be checked in polynomial time with favorable assumptions, is NP-complete with less favorable assumptions, and is PSPACE-complete in the general case.

Algorithms such as the graph-updating algorithm and loopsearch algorithm scale better with the number of meaningful species than the number of null species, while engineered CRN implementations generally do not use loops that produce null species. Thus those algorithms will be faster than their worst-case limits in practical cases. For example, the graph-updating algorithm takes at most $(2zn^k + 1)n^k = O(n^{2k+1})$ cycles in theory, where n is the number of implementation species, k the largest number of reactants in a formal reaction, and z the number of implementation species with empty interpretation. When there are no null species (or when none can be produced in a loop, as in schemes such as [16]), this becomes at most n^k cycles.

In CRN bisimulation, we require that *every* implementation species has an interpretation as a (possibly empty) multiset of formal species. In contrast, verification methods such as pathway decomposition [15] or serializability [9] both assume that each formal species is represented by one implementation species, while other implementation species are classified into fuels, wastes, and intermediates. Because of this, pathway decomposition and serializability compare formal reactions to implementation pathways which begin and end with (representations of) formal species, while in bisimulation an individual implementation reaction can be interpreted and compared to the formal CRN. An additional consequence, for pathway decomposition, is that correctness guarantees do not apply to implementation states that cannot be reached from initial states representing formal species, whereas bisimulation is more robust in that correctness is asserted in those cases as well. Furthermore, even in the permissive condition, bisimulation requires that there *exist* an implementation pathway which implements a given formal reaction, while pathway decomposition and serializability both require that *all* implementation pathways have properties which may be nontrivial to check. This locality is what allows us to prove the complexity results given, which we suspect are significantly lower complexity than methods that depend on implementation pathways.

However, the use of interpretations instead of pathways means that in some cases CRN bisimulation and pathway decomposition differ on which implementations they consider correct. Bisimulation can easily be adapted to situations where there is no clear single "canonical representation" of a given formal species, while pathway decomposition has difficulty. For example, the implementation in [12] of the *reversible* formal reaction $A+B \rightleftharpoons C+D$ by reversible implementation reactions $\{x_A \rightleftharpoons i_A, i_A + x_B \rightleftharpoons i_{CD}, i_{CD} \rightleftharpoons x_C + i_D, i_D \rightleftharpoons x_D\}$. Bisimulation considers this correct with the obvious interpretation, while pathway decomposition considers the ability to release x_C then reverse without releasing x_D to be an error. On the other hand, bisimulation has trouble handling implementation species with no well-defined interpretation. Shin describes a "delayed choice" phenomenon where an implementation CRN commits to implementing one of two formal reactions before deciding which one, producing an intermediate that cannot be correctly interpreted as either of the reaction's products or their reactants; such implementations are generally considered incorrect according to bisimulation but pathway decomposition often considers them correct [15]. Shin proposes a hybrid notion of correctness where an implementation CRN is considered correct if it is a correct implementation according to pathway decomposition of some intermediate CRN, and the intermediate CRN is a correct implementation of the formal CRN according to bisimulation [15]. This notion considers correct any implementation that is correct according to either pathway decomposition or bisimulation, plus some others.

One area this theory overlooks is the rates of reactions and the probabilities of reaching certain states. For example, in [16] Soloveichik et al. argue that the concentration of each intermediate is proportional to the product of that of the formal species which we would call its interpretation, and thus the reaction rates are approximately correct. Whether this can be generalized, and whether bisimulation can help this generalization, is an important open question.

Acknowledgements. The authors would like to thank Chris Thachuk, Damien Woods, Dave Doty, and Seung Woo Shin for helpful discussions. RFJ and EW were supported by NSF grants 1317694, 1213127, and 0832824. RFJ was supported by Caltech's Summer Undergraduate Research Fellowship program and an NSF graduate fellowship. QD's current affiliation is Epic Systems, Madison, Wisconsin.

References

1. Angluin, D., Aspnes, J., Eisenstat, D.: A simple population protocol for fast robust approximate majority. Distrib. Comput. **21**, 87–102 (2008)
2. Cardelli, L.: Two-domain DNA strand displacement. Math. Struct. Comput. Sci. **23**, 247–271 (2013)
3. Cardelli, L.: Morphisms of reaction networks that couple structure to function. BMC Syst. Biol. **8**, 1–18 (2014)
4. Cardelli, L., Csikász-Nagy, A.: The cell cycle switch computes approximate majority. Sci. Rep. **2** (2012)
5. Chen, Y.J., Dalchau, N., Srinivas, N., Phillips, A., Cardelli, L., Soloveichik, D., Seelig, G.: Programmable chemical controllers made from DNA. Nat. Nanotechnol. **8**, 755–762 (2013)
6. Dong, Q.: A bisimulation approach to verification of molecular implementations of formal chemical reaction networks. Master's thesis, Stony Brook University (2012)
7. Garey, M.R., Johnson, D.S.: Computers and Intractability: A Guide to the Theory of NP-Completeness. W. H. Freeman & Co., New York (1979)
8. Grun, C., Sarma, K., Wolfe, B., Shin, S.W., Winfree, E.: A domain-level DNA strand displacement reaction enumerator allowing arbitrary non-pseudoknotted secondary structures. CoRR (2015). http://arxiv.org/abs/1505.03738
9. Lakin, M.R., Stefanovic, D., Phillips, A.: Modular verification of chemical reaction network encodings via serializability analysis. Theor. Comput. Sci. **632**, 21–42 (2016)
10. Lakin, M.R., Youssef, S., Polo, F., Emmott, S., Phillips, A.: Visual DSD: a design and analysis tool for DNA strand displacement systems. Bioinformatics **27**, 3211–3213 (2011)
11. Milner, R.: Communication and Concurrency. Prentice-Hall Inc., Upper Saddle River (1989)
12. Qian, L., Soloveichik, D., Winfree, E.: Efficient turing-universal computation with DNA polymers. In: Sakakibara, Y., Mi, Y. (eds.) DNA 16 2010. LNCS, vol. 6518, pp. 123–140. Springer, Heidelberg (2011)
13. Rackoff, C.: The covering and boundedness problems for vector addition systems. Theor. Comput. Sci. **6**(2), 223–231 (1978)
14. Savitch, W.J.: Relationships between nondeterministic and deterministic tape complexities. J. Comput. Syst. Sci. **4**, 177–192 (1970)
15. Shin, S.W., Thachuk, C., Winfree, E.: Verifying chemical reaction network implementations: a pathway decomposition approach. CoRR (2014). http://arxiv.org/abs/1411.0782
16. Soloveichik, D., Seelig, G., Winfree, E.: DNA as a universal substrate for chemical kinetics. Proc. Nat. Acad. Sci. **107**, 5393–5398 (2010)
17. Srinivas, N.: Programming chemical kinetics: engineering dynamic reaction networks with DNA strand displacement. Ph.D. thesis, California Institute of Technology (2015)