# Comparing Chemical Reaction Networks: A Categorical and Algorithmic Perspective

Luca Cardelli
Microsoft Research & University of Oxford, UK
luca@microsoft.com

Mirco Tribastone  Max Tschaikowski
Andrea Vandin
IMT Institute for Advanced Studies Lucca, Italy
{name.surname}@imtlucca.it

## Abstract

We study chemical reaction networks (CRNs) as a kernel language for concurrency models with semantics based on ordinary differential equations. We investigate the problem of comparing two CRNs, i.e., to decide whether the trajectories of a *source* CRN can be matched by a *target* CRN under an appropriate choice of initial conditions. Using a categorical framework, we extend and relate model-comparison approaches based on structural (syntactic) and on dynamical (semantic) properties of a CRN, proving their equivalence. Then, we provide an algorithm to compare CRNs, running linearly in time with respect to the cardinality of all possible comparisons. Finally, we apply our results to biological models from the literature.

***Categories and Subject Descriptors*** F.1.1 [*Models of Computation*]: Relations between models

***Keywords*** chemical reaction networks, bisimulation, model comparison, ordinary differential equations

## 1. Introduction

Chemical reaction networks (CRNs) are an established model of interaction in many natural sciences such as organic and inorganic chemistry, ecology, epidemiology, and systems biology. In informatics, they have been receiving increasing attention due to the powerful analogy between computational processes and molecular systems [22], leading to a wealth of cross-fertilization that has produced, to cite a few, foundational results on the computational power of CRNs (e.g., [15, 28]), languages for the specification of complex biomolecular systems [10], and model-reduction techniques based on traditional approaches within theoretical computer science such as abstract interpretation [11] and bisimulation [4, 16].

In this paper we study the problem of comparing CRNs with respect to their deterministic trajectories generated by the well-known quantitative semantics based on ordinary differential equations (ODEs). This associates each species of the CRN with an ODE that provides the net change of its concentration as a function of time. We are mainly motivated by applications where the hitherto unavailable possibility of formally (and automatically) comparing CRNs may provide answers of biological relevance. For instance, a subject of investigation in evolutionary biology is to understand whether a system can be postulated to have evolved into another one that still retains some of the original behavior [3, 5]. In DNA computing, one would like to compare a *specification* CRN, which just represents the actual dynamics of interest, with respect to an *implementation* CRN, where the interactions reflect certain physical and technological constraints imposed by the materials and protocols employed [24].

We formulate the comparison problem as the question of deciding whether the ODE solution of a given *source* CRN can be matched by a *target* CRN under an appropriate choice of initial conditions. More precisely, we ask for a mapping between the ODEs of the two CRNs such that the solutions coincide at all time points. This notion, called *emulation*, has been recently introduced in [3], but no procedure to compute it was provided. Later, emulation has been related in [6] to a *backward differential equivalence* (BDE). This is an equivalence relation over the variables of an ODE system such that equivalent variables have the same ODE solutions whenever initialized equally. Thus, emulation can be seen as a particular BDE over the ODE of the "union CRN" containing both the source and target networks; this is somewhat reminiscent of the typical approach for relating two process models by means of some behavioral equivalence over their disjoint union (e.g. [13]).

A partition-refinement algorithm to compute the largest BDE (i.e., the unique BDE relation that contains any other BDE relation) has been provided in [6]. It cannot be used, however, to find emulations. This is because a BDE may not represent a mapping from source species into target species. For instance, an equivalence class may not contain any species of the target CRN; or it may contain more than one (which would also impose the constraint that equivalent target species have the same initial conditions). In fact, finding an emulation means nothing else than finding a BDE where each equivalence class contains exactly one species of the target CRN and at least one species of the source CRN.

Our main goal is to develop a framework for CRN comparison together with an algorithm to compute all possible emulations between networks. Not only do we consider comparisons at the *dynamical/semantic* level through emulation, but we also study comparisons at the *structural/syntactic* level. In particular, we are concerned with establishing mappings of species and reactions from the source CRN to the target CRN. Practically, this is useful in applications because it allows one to understand, for instance, how a certain functionality, i.e., a reaction, can be found across two networks. This is relevant in the aforementioned evolutionary studies [3, 5]. From a theoretical viewpoint, structural relations are interesting because they provide a finitary, discrete view for a behavior evolving over continuous time and state space.

*LICS '16*, July 05-08, 2016, New York, NY, USA
Copyright © 2016 held by owner/author(s). Publication rights licensed to ACM.
ACM 978-1-4503-4391-6/16/07…$15.00
DOI: http://dx.doi.org/10.1145/2933575.2935318

Structure and dynamics have been only partially related in [3]. The notions of *stoichiomorphism* and *reactant morphism* provide syntactic conditions that are sufficient to obtain an emulation at the ODE level; however, the converse does not hold. The first contribution of our paper is to fully reconcile these two levels. In the same spirit of [3], we develop a new structural notion, called *flux morphism*, that characterizes emulation between CRNs. We show this in a categorial setting where we prove that the category of CRNs with flux morphisms as arrows is equivalent to the category of CRNs where arrows are emulations.

A naive algorithm for finding all emulations between two CRNs would employ the brute-force approach of trying out all possible partitions that are refinements of the largest BDE for the ODE system of the union CRN. Obviously, however, this is an intractable task if not for trivial models. We avoid this naive exploration by exploiting a novel geometric interpretation of BDE and, interestingly, using the coarsest BDE refinement algorithm [4, 6, 7] as an inner step. Our key insight is that a BDE induces a linear space that can be spanned by an appropriate subset of the generalized eigenvectors for the Jacobian matrix. Based on this fact, the algorithm starts by building an initial set of so-called *guiding partitions* from these generalized eigenvectors. (Each such generalized eigenvector induces the finest partition of the species such that two species are in the same block iff the corresponding coordinates of the generalized eigenvector are equivalent.)

The main step is to observe that given a BDE, a refinement can be obtained by computing the coarsest BDE of that partition subject to the condition that it is also a refinement of a guiding partition. The algorithm collects these refinements by recursively visiting the BDE partition obtained by refining the current one with each and every guiding partition, starting from the largest BDE of the original ODE system. The properties of the guiding partitions guarantee that the algorithm returns all BDE partitions. The algorithm takes as input an ODE system with a totally differentiable drift; hence, a fortiori it provides all possible emulations between two CRNs. In addition, by our categorical characterization result, any emulation that the algorithm finds can be always related to structural relations between the two CRNs under consideration.

We show the usefulness of our results by studying CRN comparisons of models of biological systems examined in [3]. With a prototype implementation of the algorithm, we were able to confirm all the emulations manually derived in [3], and to establish new emulations for further models that were previously not considered. Additionally, we were able to show that certain models studied in [3] cannot be related by means of an emulation. In this context, we make also a contribution of theoretical nature. The conditions for emulation/BDE typically depend on a given choice of the rate parameters. We introduce a class of CRNs, so-called *unimodal influence networks* (which include all models of [3] and many others in the literature, e.g. [25]), for which the absence of an emulation for *any* choice of rates is implied by the absence of emulation for a *specific* choice of rates (i.e., when they are all set to one).

***Further related work.*** CRN comparisons have been recently proposed in several works [17–19, 23], but none of them takes reaction rates into account. A notion of CRN comparison that considers kinetics is presented in [24], but this is specialized for a specific implementation of a CRN using DNA; furthermore, technically the result of correspondence therein established is based on an asymptotic fast-slow decomposition of the dynamics whereby fast species are assumed to be found in the stationary regime, while emulation requires equivalent traces at all time points.

BDE generalizes *backward bisimulation* developed in [4], which applies only to *elementary* CRNs (i.e., networks with reactions having at most two reagent species). In addition, we use BDE in this paper because the algorithm for computing all BDE partitions, as discussed, works for more general ODEs for which the coarsest-refinement algorithm of [6] can be used.

Boreale relates bisimulation to linear invariant subspaces for weighted automata [1]. Instead, the idea of exploiting geometrical properties of the ODE system can be traced back to a seminal work by Li and Rabitz. In [21] they show that aggregations via a linear transformation of the state space can be related to the Jacobian matrix of the ODE system. There are two crucial differences with respect to our contribution.

i) The aggregations examined in [21] concern the possibility of deriving a new ODE system where each variable represents a linear combination of the original variables. Unlike many model-reduction approaches for CRNs (e.g., [2, 8, 9, 11]) or control systems (see [27] and references therein) BDE cannot be seen as an instance of that framework. Hence, its geometric interpretation is a new result in its own right.

ii) The results of [21] are specific to mass-action ODE systems for elementary CRNs, while our algorithm works for totally differentiable drifts. In addition, a possible implementation of [21] would require symbolic reasoning over infinite sets, which our algorithm can avoid by using the syntax-driven partition-refinement approach of [4].

***Structure of the paper.*** Section 2 sets the scene by providing the previously established notions of morphism, emulation and BDE [3, 6]. Section 3 first introduces the notion of flux morphism and then generalizes all major results of [3, 6]. Using those novel results, we then prove that the category of CRNs with flux morphisms as arrows is equivalent to the category of CRNs where arrows are emulations. We continue in Section 4 by providing an algorithm that computes all BDE partitions of an ODE system underlying a totally differentiable drift. This algorithm is then used in Section 5 to decide whether certain CRNs from evolutionary biology are related by means of emulation.

## 2. Preliminaries

***Notation.*** Throughout the paper, $S$ is a set of indices. We write $A \to B$ and $B^A$ for the functions from $A$ to $B$. Moreover, we set $\mathrm{dom}(f) = \{a \mid \exists b((a,b) \in f)\}$, $f(X) = \{f(a) \mid a \in X\}$ and $f^{-1}(Y) = \{a \in \mathrm{dom}(f) \mid f(a) \in Y\}$ for all $X \subseteq \mathrm{dom}(f)$ and $Y \subseteq f(\mathrm{dom}(f))$. A set $X \subseteq \mathrm{dom}(f)$ is called invariant with respect to a function $f$ if $f(X) \subseteq X$.

***Comparison of ODE systems.*** Let the drift $f : \mathbb{R}^S \to \mathbb{R}^S$ be totally differentiable. Adopting Newton's notation, $\dot{v} = f(v)$ denotes the ODE system given, in components, by $\dot{v}_x = f_x(v)$, where $x \in S$. Given an initial condition $v(0) \in \mathbb{R}^S$, Picard-Lindelöf's theorem ensures that $\dot{v} = f(v)$ has a unique solution $v : \mathrm{dom}(v) \to \mathbb{R}^S, t \mapsto v(t)$. The assumption $\mathrm{dom}(f) = \mathbb{R}^S$ is made to simplify presentation and can be easily removed.

**Definition 1** (Backward differential equivalence)**.** *Fix a drift $f : \mathbb{R}^S \to \mathbb{R}^S$ and a partition $\mathcal{H}$ of $S$. A vector $v \in \mathbb{R}^S$ is* constant on *$\mathcal{H}$ if for all $x, y \in H$ and $H \in \mathcal{H}$ it holds that $v_x = v_y$. We call $\mathcal{H}$ backward differential equivalent (BDE) if, for any $v \in \mathbb{R}^S$ that is constant on $\mathcal{H}$, also $f(v)$ is constant on $\mathcal{H}$.*

In BDE, the trajectories of equivalent variables are identical if initialized with the same values.

**Theorem 1.** *A partition $\mathcal{H}$ is BDE if and only if, for any $v(0) \in \mathbb{R}^S$ that is constant on $\mathcal{H}$, the solution of $\dot{v} = f(v)$ with initial condition $v(0)$, $v(t)$, is constant on $\mathcal{H}$ for all $t \in \mathit{dom}(v)$.*

**Example 1.** *Let $S = \{x, y\}$ and consider the drift*

$$f_x(v) = -2 \cdot v_x^2 + v_y^2 \qquad f_y(v) = -2 \cdot v_y^2 + v_x^2$$

*Then, the partition $\{\{x, y\}\}$ of $S$ is BDE and $v_x(t) = v_y(t)$ for all $t \geq 0$ whenever $v_x(0) = v_y(0)$.*

We take the notion of emulation from [3].

**Definition 2** (Emulation). *Fix a source drift $f : \mathbb{R}^S \to \mathbb{R}^S$ and target drift $\hat{f} : \mathbb{R}^{\hat{S}} \to \mathbb{R}^{\hat{S}}$. The surjective function $\mu : S \to \hat{S}$ is an emulation if $f_x(\hat{v} \circ \mu) = \hat{f}_{\mu(x)}(\hat{v})$ for all $\hat{v} \in \mathbb{R}^{\hat{S}}$ and $\hat{x} \in \hat{S}$.*

Note that $\hat{v} \circ \mu \in \mathbb{R}^S$ and $(\hat{v} \circ \mu)_x = \hat{v}_{\mu(x)}$ for all $x \in S$. Two ODE systems are related by means of an emulation if the trajectories of the source ODE system coincide with those of the target ODE system whenever the initial conditions of both systems are equal with respect to $\mu$, as stated next.

**Theorem 2.** *Let $\mu : S \to \hat{S}$ be an emulation between the source ODE system $f : \mathbb{R}^S \to \mathbb{R}^S$ and the target ODE system $\hat{f} : \mathbb{R}^{\hat{S}} \to \mathbb{R}^{\hat{S}}$. Then, $v(t) = \hat{v}(t) \circ \mu$ for all $t \geq 0$ whenever $v(0) = \hat{v}(0) \circ \mu$.*

**Example 2.** *Consider $\hat{S} = \{\hat{x}\}$ and $\hat{f}_{\hat{x}}(\hat{v}) = -\hat{v}_{\hat{x}}$. Then the function $\mu(x) = \mu(y) = \hat{x}$ is an emulation between $f : \mathbb{R}^{\{x,y\}} \to \mathbb{R}^{\{x,y\}}$ and $\hat{f} : \mathbb{R}^{\{\hat{x}\}} \to \mathbb{R}^{\{\hat{x}\}}$, where $f$ is as in Example 1. In particular, it holds that $\hat{v}_{\hat{x}}(t) = v_x(t) = v_y(t)$ for all $t \geq 0$ whenever $\hat{v}_{\hat{x}}(0) = v_x(0) = v_y(0)$.*

The example above indicates that there is a close relation between emulation and BDE, as observed in [4].

**Proposition 1.** *Fix a source drift $f : \mathbb{R}^S \to \mathbb{R}^S$, a target drift $\hat{f} : \mathbb{R}^{\hat{S}} \to \mathbb{R}^{\hat{S}}$ with $S \cap \hat{S} = \emptyset$ and a surjective function $\mu : S \to \hat{S}$. The partition $\{\mu^{-1}(\hat{x}) \cup \{\hat{x}\} \mid \hat{x} \in \hat{S}\}$ is a BDE if and only if $\mu$ is an emulation.*

***Chemical Reaction Networks.*** Formally, a CRN $(S, R)$ is a pair consisting of a finite set of species $S$ and a finite set of chemical reactions $R$. A reaction is a triple written in the form $\rho \to^\alpha \pi$, where $\rho$ and $\pi$ are the multisets of species *reactants* and *products*, respectively, and $\alpha > 0$ is the reaction rate parameter. We denote by $\rho(x)$ the multiplicity of species $x$ in the multiset $\rho$. The *flux stoichiometry* $\phi(x, r)$ of a species $x$ in a reaction $r = \rho \to^\alpha \pi$ is the difference between product and reactant multiplicity, times the rate coefficient $\alpha$, i.e. $\phi(x, r) = \phi(x, \rho \to^\alpha \pi) = \alpha \cdot (\pi(x) - \rho(x))$. It describes the amount of substance $x$ transformed through reaction $r$ in a time unit. A given $\mu : S \to \hat{S}$ can be trivially lifted to multisets over $S$, e.g., $\mu(x + y) = \mu(x) + \mu(y)$.

The ODE system $\dot{v} = f(v)$ underlying a CRN $(S, R)$ is $f : \mathbb{R}^S \to \mathbb{R}^S$, where each component $f_x$, with $x \in S$, is defined as

$$f_x(v) := \sum_{\rho \to^\alpha \pi \in R} \phi(x, \rho \to^\alpha \pi) \cdot [\![\rho \to^\alpha \pi]\!]_v$$
$$:= \sum_{\rho \to^\alpha \pi \in R} (\pi(x) - \rho(x)) \cdot \alpha \cdot \prod_{y \in S} v_y^{\rho(y)}$$

**Example 3.** *The network $(\{x, y\}, \{x + x \to^1 y, y + y \to^1 x\})$ induces the drift of Example 1.*

This represents the well-known *mass-action* kinetics, where the reaction rate is proportional to the concentrations of the reactants involved. Since the ODE system of a CRN is given by polynomials, the drift $f$ is totally differentiable, meaning that there exists a unique solution of $\dot{v} = f(v)$ for any initial condition $v(0)$.

**Theorem 3** (see [4, 6]). *For any partition $\mathcal{G}$ of $S$, the coarsest BDE partition $\mathcal{H}$ that refines $\mathcal{G}$ exists and can be calculated using a partition refinement algorithm.*

***Structural properties of CRNs.*** Emulation is a dynamical property of an ODE system that is implied by the syntactical properties of CRNs reactant morphism and the stoichiomorphism from [3].

**Definition 3.** *Let $(S, R)$ and $(\hat{S}, \hat{R})$ denote the source and the target CRNs, respectively. A pair of functions $(\mu, \sigma) \in (S \to \hat{S}) \times (R \to \hat{R})$ is*

- *a* reactant morphism *if, for all $\rho \to^\alpha \pi \in R$ there exist $\hat{\alpha}$ and $\hat{\pi}$ with $\sigma(\rho \to^\alpha \pi) = \mu(\rho) \to^{\hat{\alpha}} \hat{\pi} \in \hat{R}$.*
- *a* stoichiomorphism *whenever $\sum_{r \in \sigma^{-1}(\hat{r})} \phi(x, r) = \phi(\mu(x), \hat{r})$ for all $x \in S$ and $\hat{r} \in \hat{R}$.*

**Theorem 4** (see [3]). *Fix a source network $(S, R)$, a target network $(\hat{S}, \hat{R})$ and let $(\mu, \sigma) \in (S \to \hat{S}) \times (R \to \hat{R})$ be a reactant morphism and stoichiomorphism. Then, $\mu : S \to \hat{S}$ is an emulation between the source drift $f$ and the target drift $\hat{f}$.*

***Category theory.*** Following the usual notation, we denote by $|\mathcal{C}|$ the objects of a given category $\mathcal{C}$, while $\mathrm{Hom}_{\mathcal{C}}(a, b)$ refers to the set of arrows from object $a$ to object $b$, where $a, b \in |\mathcal{C}|$. A function between categories $\Psi : \mathcal{C} \to \mathcal{D}$ is called a functor if $\Psi(a) \in |\mathcal{D}|$ for all $a \in |\mathcal{C}|$ and $\Psi(\psi) : \Psi(a) \to \Psi(b) \in \mathrm{Hom}_{\mathcal{D}}(\Psi(a), \Psi(b))$ for all $\psi : a \to b \in \mathrm{Hom}_{\mathcal{C}}(a, b)$. Additionally, $\Psi$ has to preserve identities and compositions.

**Definition 4.** *A functor $\Psi$ establishes the equivalence of categories $\mathcal{C}$ and $\mathcal{D}$ if for any $a, b \in |\mathcal{C}|$ it holds that*

- *the function $\Psi : Hom_{\mathcal{C}}(a, b) \to Hom_{\mathcal{D}}(\Psi(a), \Psi(b))$ is bijective;*
- *and for each $d \in |\mathcal{D}|$ there exists some $c \in |\mathcal{C}|$ such that $d$ is isomorphic to $\Psi(c)$.*

## 3. Equivalence of Structure and Dynamics

***Flux morphisms.*** Ordinary morphisms from [3] are only sufficient conditions for emulation in general.

**Example 4.** *Consider the source network*

$$x + y \to^1 x \qquad x + z \to^1 x$$
$$y + z \to^1 y \qquad y + z \to^1 2z + y$$

*and the target network $\hat{x} + \hat{y} \to^1 \hat{x}$. Then, $\mu(x) := \hat{x}$ and $\mu(y) := \mu(z) := \hat{y}$ defines an emulation $\mu : S \to \hat{S}$. However, there exists no $\sigma : R \to \hat{R}$ such that $(\mu, \sigma)$ is a reactant morphism. This is because the reactions $y + z \to^1 y$ and $y + z \to^1 2z + y$ are "redundant", i.e., they can be dropped without affecting the underlying drift. At the same time, they introduce the reactant $y + z$ that yields $2\hat{y} = \mu(y + z) \neq \hat{x} + \hat{y}$.*

We tackle this problem by introducing the notion of quotient reactant morphism.

**Definition 5.** *Two reactions $r_1 = \rho_1 \to^{\alpha_1} \pi_1, r_2 = \rho_2 \to^{\alpha_2} \pi_2 \in R$ of a CRN $(S, R)$ are* reactant equivalent, *written $r_1 \sim r_2$, if and only if $\rho_1 = \rho_2$. In the following, let $[r]$ denote the equivalence class of $r$ with respect to $\sim$. Moreover, for any given multiset $\rho'$, define $R_{|\rho'} = \{\rho \to^\alpha \pi \in R \mid \rho = \rho'\}$.*

Note that $[\rho \to^\alpha \pi] = R_{|\rho}$ for any $\rho \to^\alpha \pi \in R$.

**Definition 6.** *Fix a source CRN $(S, R)$ and a target CRN $(\hat{S}, \hat{R})$. A total surjective function $\mu : S \to \hat{S}$ and a partial function $\sigma : R/{\sim} \hookrightarrow \hat{R}/{\sim}$ form a* quotient reactant morphism *if*

*i) for any $[\rho \to^\alpha \pi] \in R/{\sim}$, the function $\sigma$ is either undefined if $\hat{R}_{|\mu(\rho)} = \emptyset$ or it satisfies*

$$\sigma([\rho \to^\alpha \pi]) = \hat{R}_{|\mu(\rho)} \quad \textit{when} \quad \hat{R}_{|\mu(\rho)} \neq \emptyset \qquad (1)$$

*ii) the union of classes for which $\sigma$ is not defined is* redundant with respect to $\mu$, *i.e. for all $x \in S$ and $\hat{v} \in \mathbb{R}^{\hat{S}}$ it holds that*

$0 = \sum_{r \in R_0} \phi(x, r) \cdot [\![r]\!]_{\hat{v} \circ \mu}$*, where* $R_0 = \bigcup \{e \in R/{\sim} \mid e \notin$ *dom*$(\sigma)\}$.

**Example 5.** *With $\mu$ as in Example 4 and $\sigma$ given by*

$$\sigma([x + y \to^1 x]) = \sigma([x + z \to^1 x]) = [\hat{x} + \hat{y} \to^1 \hat{x}],$$
$$\sigma([y + z \to^1 y]) = \emptyset,$$

*$(\mu, \sigma)$ is a quotient reactant morphism because $y + z \to^1 y$ and $y + z \to^1 2z + y$ of the source network cancel each other out.*

In order to lift Theorem 4 to the new notion of quotient reactant morphism, the notion of stoichiomorphism has to be relaxed as well.

**Definition 7.** *For a source CRN $(S, R)$ and a target CRN $(\hat{S}, \hat{R})$, the total surjective function $\mu : S \to \hat{S}$ and the* partial *function $\sigma : R/{\sim} \hookrightarrow \hat{R}/{\sim}$ form a quotient stoichiomorphism if for all $x \in S$ and $\hat{e} \in \hat{R}/{\sim}$ it holds that*

$$\sum_{e \in \sigma^{-1}(\hat{e})} \sum_{r \in e} \phi(x, r) = \sum_{\hat{r} \in \hat{e}} \phi(\mu(x), \hat{r})$$

The quotient reactant morphism discussed in Example 5 is easily verified to be a quotient stoichiomorphism.

**Definition 8.** *For a source CRN $(S, R)$ and a target CRN $(\hat{S}, \hat{R})$, a mapping $(\mu, \sigma) \in (S \to \hat{S}) \times (R/{\sim} \hookrightarrow \hat{R}/{\sim})$ is called* flux morphism *if $(\mu, \sigma)$ is a quotient reactant morphism and a quotient stoichiomorphism.*

The following result shows that a reactant morphism and a stoichiomorphism give rise to a flux morphism.

**Theorem 5.** *If $(\mu, \underline{\sigma}) : (S, R) \to (\hat{S}, \hat{R})$ is a reactant morphism and stoichiomorphism, then $(\mu, \sigma) : (S, R/{\sim}) \to (\hat{S}, \hat{R}/{\sim})$, where $\sigma$ is induced by $\underline{\sigma}$ via $\sigma([r]) := [\underline{\sigma}(r)]$, is a flux morphism.*[1]

The result below, instead, shows that Theorem 4 carries over to flux morphisms.

**Theorem 6.** *Fix a source CRN $(S, R)$, a target CRN $(\hat{S}, \hat{R})$ and let $(\mu, \sigma) \in (S \to \hat{S}) \times (R/{\sim} \hookrightarrow \hat{R}/{\sim})$ be a flux morphism. Then, $\mu$ is also an emulation, meaning that $f(\hat{v} \circ \mu) = \hat{f}(\hat{v}) \circ \mu$ for all $\hat{v} \in \mathbb{R}^{\hat{S}}$.*

For instance, since $(\mu, \sigma)$ from Example 5 defines a flux morphism, $\mu : S \to \hat{S}$ is an emulation.

The following is a partial converse of Theorem 6: emulation and quotient reactant morphism yield a quotient stoichiomorphism. It is an important step towards our first main result.

**Proposition 2.** *Fix a source CRN $(S, R)$, a target CRN $(\hat{S}, \hat{R})$ and let $(\mu, \sigma) \in (S \to \hat{S}) \times (R/{\sim} \hookrightarrow \hat{R}/{\sim})$ be a quotient reactant morphism and $\mu$ an emulation. Then, $(\mu, \sigma)$ is also a quotient stoichiomorphism.*

We are in a position to state our first main result. It is a converse of Theorem 6 and, as has been observed in Example 4, cannot be stated on the domain of ordinary notions from [3].

**Theorem 7.** *Fix a source CRN $(S, R)$, a target CRN $(\hat{S}, \hat{R})$ and a function $\mu : S \to \hat{S}$. Then, $\mu$ is an emulation if and only if $(\mu, \sigma)$ is a flux morphism where $\sigma$ is the unique partial function $\sigma : R/{\sim} \hookrightarrow \hat{R}/{\sim}$ that satisfies condition (1).*

*Proof.* Since the only if direction follows from Theorem 6, let us assume that $\mu : S \to \hat{S}$ is an emulation and set $\sigma([\rho \to^\alpha \pi]) := \hat{R}_{|\mu(\rho)}$ if $\hat{R}_{|\mu(\rho)} \neq \emptyset$. Thanks to Proposition 2, it suffices to show that the so defined $(\mu, \sigma)$ is a quotient reactant morphism.

Let $f'$ denote the aggregated drift underlying $\mu$ that arises from the drift $f$ of $(S, R)$, that is

$$f'_{\mu(x)}(\hat{v}) = \sum_{\rho \to^\alpha \pi \in R} \phi(x, r) \cdot [\![\mu(\rho)]\!]_{\hat{v}}$$

for all $x \in S$ and $\hat{v} \in \mathbb{R}^{\hat{S}}$. Note that $f'$ is well-defined because $\mu$ is an emulation, that is it holds that $f'_{\mu(x)}(\hat{v}) = f'_{\mu(x')}(\hat{v})$ if $\mu(x) = \mu(x')$. Instead, let $\hat{f}$ denote the drift of $(\hat{S}, \hat{R})$. Since $\mu$ is an emulation, we have

$$\hat{f}_{\mu(x)}(\hat{v}) = f_x(\hat{v} \circ \mu) = f'_{\mu(x)}(\hat{v})$$

for all $x \in S$ and $\hat{v} \in \mathbb{R}^{\hat{S}}$. Note that the above equalities hold for all $\hat{v} \in \mathbb{R}^{\hat{S}}$ and $x \in S$. Hence, since two polynomials that coincide on all values need to have the same monomials, we thus infer that $\hat{f}_{\hat{x}}$ and $f'_{\hat{x}}$ have the same monomials for all $\hat{x} \in \hat{S}$. By construction, the monomials of $f'$ are contained in $\{c \cdot \mu(\rho) \mid \rho \to^\alpha \pi \in R, c \in \mathbb{R}\}$. Hence, if $\rho \to^\alpha \pi \in R$ and there are no $\hat{\alpha}, \hat{\pi}$ such that $\mu(\rho) \to^{\hat{\alpha}} \hat{\pi} \in \hat{R}$, we infer that reactions $\{\rho' \to^{\alpha'} \pi' \in R \mid \mu(\rho') = \mu(\rho)\}$ are redundant for all $\hat{v} \circ \mu$ with $\hat{v} \in \mathbb{R}^{\hat{S}}$. This shows that $(\mu, \sigma)$ is a quotient reactant morphism. ☐

**Example 6.** *Let $(\mu, \sigma)$ be as in Example 5. Then, Theorem 7 implies that $\mu : S \to \hat{S}$ is an emulation between the source and the target network. Conversely, by Theorem 7, it suffices to show that $\mu : S \to \hat{S}$ is an emulation to infer that the unique $\sigma$ that is induced by (1) and $\mu$ is such that $(\mu, \sigma)$ is a flux morphism.*

Using Theorem 7 we next prove that flux morphism implies the notion of BDE, and vice versa.

**Theorem 8.** *Fix a source CRN $(S, R)$ and a target CRN $(\hat{S}, \hat{R})$ with $S \cap \hat{S} = \emptyset$. Then, there is a flux morphism $(\mu, \sigma) : (S \to \hat{S}) \times (R/{\sim} \hookrightarrow \hat{R}/{\sim})$ if and only if there exists a BDE partition $\mathcal{H}$ of $S \cup \hat{S}$ such that $|H \cap S| \geq 1$ and $|H \cap \hat{S}| = 1$ for all $H \in \mathcal{H}$.*

**Example 7.** *We have seen that $(\mu, \sigma)$ from Example 5 is a flux morphism and used Theorem 7 in Example 6 to conclude that $\mu : S \to \hat{S}$ is an emulation. Proposition 1 ensures that*

$$\{\mu^{-1}(\hat{x}) \cup \{\hat{x}\} \mid \hat{x} \in \hat{S}\} = \{\{x, \hat{x}\}, \{y, z, \hat{y}\}\}$$

*is a BDE partition of the union CRN $(S \cup \hat{S}, R \cup \hat{R})$, thus confirming the if direction of Theorem 8.*

We end the paragraph by partially lifting the "change of rates" theorem of [3] to flux morphisms. It states, essentially, that morphisms respect, to a certain degree, a change of rate coefficients.

The following notions will be needed.

**Definition 9.** *Fix a CRN $(S, R)$. A bijection $\iota : R \to R'$ that satisfies $\iota(\rho \to^\alpha \pi) = \rho \to^{\alpha'} \pi$ for all $\rho \to^\alpha \pi \in R$ is called a change of rates.*

**Definition 10.** *We call a set of reactions $\sim$-uniform if any two reactions $\rho_1 \to^{\alpha_1} \pi_1, \rho_2 \to^{\alpha_2} \pi_2 \in R$ satisfy $\alpha_1 = \alpha_2$ if $\rho_1 = \rho_2$. A change of rates is called $\sim$-uniform if it leads to a $\sim$-uniform set of reactions.*

We are in a position to state the result.

**Theorem 9.** *Fix a source CRN $(S, R)$, a target CRN $(\hat{S}, \hat{R})$ with a $\sim$-uniform set of reactions $\hat{R}$ and let $(\mu, \sigma) \in (S \to \hat{S}) \times (R/{\sim} \hookrightarrow \hat{R}/{\sim})$ be a flux morphism. Then, for any $\sim$-uniform change of rates $\hat{\iota} : \hat{R} \to \hat{R}'$, there exists a change of rates*

[1] *Note to the reviewers.* The proofs are in the appendix that will be published in the case of acceptance.

*$\iota : R \to R'$ such that $(\mu, \sigma') : (S, R') \hookrightarrow (\hat{S}, \hat{R}')$, where $\sigma'$ is induced by $\mu : S \to \hat{S}$ and condition (1), is a flux morphism.*

In [3] no assumptions were made on the nature of rate change. This comes, however, at the price of making the stronger assumption of ordinary morphisms.

The assumption in Theorem 9 cannot be dropped. To see this, consider the source CRN $x \to^{\alpha_1} x, x \to^{\alpha_2} 2x$ and the target CRN $\hat{x} \to^{\beta_1} \emptyset$, $\hat{x} \to^{\beta_2} 3\hat{x}$. If $\beta_1 = \beta_2 = \beta$, setting $\alpha_1 = \alpha_2 = \beta$ induces a quotient stoichiomorphism. Instead, if $\beta_1 = 10$ and $\beta_2 = 1$, there is no pair $\alpha_1, \alpha_2 > 0$ for which there exists a quotient stoichiomorphism.

***Equivalence of structure and dynamics.*** We now build on our findings to establish an equivalence result between the category of structurally related networks $\mathcal{C}^m$ and the category of dynamically related networks $\mathcal{C}^e$.

**Definition 11.** *The category $\mathcal{C}^m$ has CRNs as objects and flux morphisms as arrows. The identity morphism of $(S, R)$ is the pair of identity functions $(id_S, id_R)$, while the composition of two flux morphisms $(\mu, \sigma) : (S, R) \to (\hat{S}, \hat{R})$ and $(\hat{\mu}, \hat{\sigma}) : (\hat{S}, \hat{R}) \to (\hat{S}', \hat{R}')$ is given by*

$$\big(\hat{\mu}, \hat{\sigma}\big) \circ \big(\mu, \sigma\big)(x, [\rho \to^{\alpha} \pi]) := \big(\hat{\mu}(\mu(x)), \hat{R}'_{|\hat{\mu}(\mu(\rho))}\big)$$

The definition of $\mathcal{C}^e$ is straightforward and follows next.

**Definition 12.** *The category $\mathcal{C}^e$ has CRNs as objects and emulations as arrows. The identity emulation is given by the identity function $id_S$ and the composition of emulations is defined in the obvious way.*

The next result ensures that $\mathcal{C}^m$ and $\mathcal{C}^e$ are indeed categories.

**Lemma 1.** *Emulations and flux morphisms are closed under composition.*

Category $\mathcal{C}^m$ relates CRNs that share the same structure, while $\mathcal{C}^e$ relates CRNs that emulate each other. We relate now both categories by means of a functor.

**Definition 13.** *Set $\Psi : \mathcal{C}^m \to \mathcal{C}^e$ by $\Psi((S, R)) = (S, R)$ for all $(S, R) \in |\mathcal{C}^m|$ and by $\Psi((\mu, \sigma)) = \mu$ for all $(\mu, \sigma) \in Hom_{\mathcal{C}^m}((S, R), (\hat{S}, \hat{R}))$ and $(S, R), (\hat{S}, \hat{R}) \in |\mathcal{C}^m|$.*

Note that $\Psi$ is a well-defined functor because Theorem 6 ensures that $\mu$ is an emulation if $(\mu, \sigma)$ is a flux morphism.

The next result is a consequence of Theorem 7. It states that by studying structural properties of CRNs one does not lose symmetries present at the ODE level. It thus formally shows the equivalence of structure and function on the level of CRNs in [3].

**Theorem 10.** *The functor $\Psi : \mathcal{C}^m \to \mathcal{C}^e$ yields the equivalence of $\mathcal{C}^m$ and $\mathcal{C}^e$.*

## 4. Algorithmic Model Comparison

In this section we present an algorithm for the calculation of all BDE partitions underlying a totally differentiable drift $f : \mathbb{R}^S \to \mathbb{R}^S$. In addition to being of importance on its own in the area of model reduction, it can be used to decide whether a source ODE system $f$ and a target ODE system $\hat{f}$ are related by means of an emulation. Thus, in particular, the algorithm provides a technique to decide whether there exists an arrow between two given CRNs in $\mathcal{C}^e$ (and, thanks to Theorem 10, also in $\mathcal{C}^m$). In the case there are arrows, the algorithm calculates all of them.

We reiterate that the partition refinement algorithm of Theorem 3 cannot be used to tackle the problem because it calculates *the coarsest* BDE partition but does not tell one whether this partition can be split further into finer BDE partitions. In particular, the number of possible refinements of the coarsest BDE partition is still too large to be analyzed efficiently. As discussed in Section 1, the main idea behind our algorithm is to find a set of guiding partitions which, if applied to the partition refinement algorithm, guarantees to find any BDE partition. We thus perform a *guided*, instead of a *brute-force*, search.

***Calculating all BDEs.*** We first introduce the set of vectors $\mathcal{U}_\mathcal{H}$ that is constant on a given partition $\mathcal{H}$ of $S$ and observe that $\mathcal{U}_\mathcal{H}$ is a linear subspace of $\mathbb{R}^S$.

**Lemma 2.** *Fix a partition $\mathcal{H}$ of $S$. Then, the set $\mathcal{U}_\mathcal{H} := \{v \in \mathbb{R}^S \mid v$ is constant on $\mathcal{H}\}$ is a linear subspace of $\mathbb{R}^S$.*

For convenience, we write space instead of linear space. The first step towards our algorithm is to observe that, whenever $\mathcal{H}$ is a BDE partition, $\mathcal{U}_\mathcal{H}$ is an invariant space of the Jacobi matrix of $f$ evaluated at point $\mathbb{1} \in \mathbb{R}^S$, written $J(\mathbb{1})$, where $\mathbb{1}_x = 1$ for all $x \in S$. This observation is inspired by [21], where subspaces underlying linear transformations of ODE state spaces are shown to be invariant sets of the transpose of $J(\mathbb{1})$.

**Theorem 11.** *Fix a totally differentiable $f : \mathbb{R}^S \to \mathbb{R}^S$ and assume that $\mathcal{H}$ is a BDE partition of $S$. It holds that $J(\mathbb{1})v \in \mathcal{U}_\mathcal{H}$ for any $v \in \mathcal{U}_\mathcal{H}$, meaning that $\mathcal{U}_\mathcal{H}$ is an invariant space of the Jacobi matrix of $f$ evaluated at $\mathbb{1} \in \mathbb{R}^S$.*

*Proof.* Recall that for a given drift $f : \mathbb{R}^S \to \mathbb{R}^S$, $v \mapsto f(v)$, a partition $\mathcal{H}$ of $S$ is BDE if and only if $f(v)$ is constant on $\mathcal{H}$ whenever $v$ is constant on $\mathcal{H}$. Fix an arbitrary $v' \in \mathcal{U}_\mathcal{H}$. Since $f$ is totally differentiable, there exists a continuous function $r : \mathbb{R}^S \to \mathbb{R}^S, v \mapsto r(v)$ such that

$$f(v' + v) = f(v') + J(v') \cdot v + r(v)$$

for all $v \in \mathbb{R}^S$, $r(v') = 0$ and $\lim_{v \to v'} \frac{r(v)}{\|v - v'\|} = 0$. (Note that $v \mapsto f(v') + J(v') \cdot v$ is the linearization of $f$ at point $v'$.) Fix arbitrary $x, y \in H$ and $H \in \mathcal{H}$. Then, in the case $v \in \mathcal{U}_\mathcal{H}$, the above discussion implies that

$$\begin{aligned} f_x(v' + v) - f_y(v' + v) &= f_x(v') - f_y(v') \\ &\quad + (J(v') \cdot v)_x - (J(v') \cdot v)_y + r_x(v) - r_y(v), \end{aligned}$$

thus yielding

$$\begin{aligned} \frac{r_y(v) - r_x(v)}{\|v\|} &= \tfrac{1}{\|v\|}\Big((J(v') \cdot v)_x - (J(v') \cdot v)_y\Big) \\ &= (J(v') \cdot \tfrac{v}{\|v\|})_x - (J(v') \cdot \tfrac{v}{\|v\|})_y \end{aligned}$$

Since $v \in \mathcal{U}_\mathcal{H}$ can be chosen arbitrarily small, this implies that $(J(v') \cdot v)_x = (J(v') \cdot v)_y$ for all $v \in \mathcal{U}_\mathcal{H}$. Hence, $\mathcal{U}_\mathcal{H}$ is an invariant space of $J(v')$ for any $v' \in \mathcal{U}_\mathcal{H}$. Since $\mathbb{1} \in \mathcal{U}_\mathcal{H}$, this yields the claim. □

**Example.** *Set $S = \{1, 2, 3\}$ and consider the drift $f : \mathbb{R}^S \to \mathbb{R}^S$*

$$\begin{pmatrix} v_1 \\ v_2 \\ v_3 \end{pmatrix} \mapsto \begin{pmatrix} -v_1 v_3 + v_3 v_2 \\ -v_2 v_1 + v_1 v_3 \\ -v_3 v_2 + v_2 v_1 \end{pmatrix}$$

*The Jacobi matrix at point $\mathbb{1}$ is then*

$$J(\mathbb{1}) = \begin{pmatrix} -1 & 1 & 0 \\ 0 & -1 & 1 \\ 1 & 0 & -1 \end{pmatrix} \tag{2}$$

*With this, it holds that $J(\mathbb{1}) \cdot (1, 1, 1)^T = (0, 0, 0)^T$, thus showing that the subspace $\{\eta \cdot (1, 1, 1)^T \mid \eta \in \mathbb{R}\}$ of $\mathbb{R}^3$ is an invariant space of $J(\mathbb{1})$.*

Having established that any $\mathcal{U}_{\mathcal{H}}$ of a BDE partition $\mathcal{H}$ is an invariant set of $J(\mathbb{1})$, we ask ourselves next how to calculate the invariant subspaces of $J(\mathbb{1})$. This is a classic topic of linear algebra, so let us recall some elementary notions.

**Definition 14.** *Fix $A \in \mathbb{R}^{S\times S}$. If $\lambda \in \mathbb{R}$ and $w \in \mathbb{R}^S \setminus \{0\}$ are such that $Aw = \lambda w$, we call $\lambda$ the eigenvalue corresponding to the eigenvector $w$. The set of all eigenvalues of $A$ is called spectrum and is denoted by $\sigma(A)$. The eigenspace of $\lambda \in \sigma(A)$ is given by $E_\lambda(A) = \{w \in \mathbb{R}^S \mid (A - \lambda I)w = 0\}$. The matrix $A$ is diagonalizable if there exists a basis of $\mathbb{R}^S$ consisting of eigenvectors of $A$.*

In order to simplify presentation, we first discuss the situation in the special case where $J(\mathbb{1})$ is diagonalizable.

Let us assume that $\mathcal{W}$ is an invariant set of $J(\mathbb{1})$. Since $J(\mathbb{1})$ is diagonalizable and $E_\lambda(J(\mathbb{1})) \cap E_{\lambda'}(J(\mathbb{1})) = \{0\}$ whenever $\lambda \neq \lambda'$, $\mathbb{R}^S$ can be decomposed into the eigenspaces of $J(\mathbb{1})$, i.e. $\mathbb{R}^S = \bigoplus_\lambda E_\lambda(J(\mathbb{1}))$. Moreover, since this implies that $\mathcal{W} = \bigoplus_\lambda(\mathcal{W} \cap E_\lambda(J(\mathbb{1})))$, we infer that any invariant set of $J(\mathbb{1})$ can be written as a direct sum of subsets of eigenspaces.

At the same time, any subspace of an eigenspace $E_\lambda(J(\mathbb{1}))$ is an invariant space of $J(\mathbb{1})$ because $Aw = \lambda w$ for all $w \in E_\lambda(J(\mathbb{1}))$. Thus, it suffices to determine the set of all possible subspaces of $E_\lambda(J(\mathbb{1}))$. To get an idea how those look like, let us fix some $\lambda \in \sigma(J(\mathbb{1}))$ and assume that $E_\lambda(J(\mathbb{1})) = \langle u_1, u_2, u_3\rangle$, where $\langle w'_1, \ldots, w'_k\rangle$ denotes the set of all linear combinations $\sum_{i=1}^k \eta_i w'_i$. The one dimensional invariant sets contained in $E_\lambda(J(\mathbb{1}))$ are then given by the family $\langle \eta_1 u_1 + \eta_2 u_2 + \eta_3 u_3\rangle$ where $\eta_1, \eta_2, \eta_3 \in \mathbb{R}$ such that $|\eta_1| + |\eta_2| + |\eta_3| \neq 0$. Instead, the two dimensional invariant sets of $E_\lambda(J(\mathbb{1}))$ are given by the families $\langle \eta u_1 + \eta' u_2, u_3\rangle$, $\langle \eta u_1 + \eta' u_3, u_2\rangle$ and $\langle u_1, \eta u_2 + \eta' u_3\rangle$ with $|\eta| + |\eta'| \neq 0$. Finally, there is only one three dimensional invariant set, namely $E_\lambda(J(\mathbb{1})) = \langle u_1, u_2, u_3\rangle$ itself.

The above discussion leads to the following.

**Theorem 12.** *Let $J(\mathbb{1})$ be diagonalizable. Then, for any BDE space $\mathcal{U}_{\mathcal{H}}$, there are linearly independent vectors $w_1^\lambda, \ldots, w_{k_\lambda}^\lambda \in E_\lambda(J(\mathbb{1}))$ with $0 \leq k_\lambda \leq dimE_\lambda(J(\mathbb{1}))$ such that $\mathcal{U}_{\mathcal{H}} = \bigoplus_\lambda \langle w_1^\lambda, \ldots, w_{k_\lambda}^\lambda\rangle$.*

Thus, if we fix for each $\lambda \in \sigma(J(\mathbb{1}))$ a basis $u_1^\lambda, \ldots, u_{\dim E_\lambda}^\lambda \in \mathbb{R}^S$ of $E_\lambda(J(\mathbb{1}))$, then for any $w_i^\lambda$ there exist coefficients $\eta_j^{(\lambda,i)} \in \mathbb{R}$ so that $w_i^\lambda = \sum_{j=1}^{\dim E_\lambda} \eta_j^{(\lambda,i)} u_j^\lambda$.

We now drop the assumption of $J(\mathbb{1})$ being diagonalizable. To this end, from now on until Theorem 14, we work on $\mathbb{C}^S$ as a vector space *over the field* $\mathbb{C}$.

Definition 14 carries over to the complex case by replacing each $\mathbb{R}$ with $\mathbb{C}$. In particular, $\sigma(A) \subseteq \mathbb{C}$.

**Definition 15.** *For $A \in \mathbb{C}^{S\times S}$, the generalized eigenspace of $\lambda \in \sigma(A)$ is $E^*_\lambda(A) = \{z \in \mathbb{C}^n \mid (A - \lambda I)^{\nu_\lambda} z = 0\}$, where $\nu_\lambda$ denotes the algebraic multiplicity of $\lambda$. Call any $z \in E^*_\lambda(A) \setminus \{0\}$ a generalized eigenvector of $A$.*

It can be shown that $E^*_\lambda(A)$ is an invariant set of $A$ for any $\lambda \in \sigma(A)$ and that $\mathbb{C}^n = \bigoplus_\lambda E^*_\lambda(A)$ over field $\mathbb{C}$. Moreover, the following well-known result holds.

**Theorem 13.** *A matrix $A \in \mathbb{C}^{S\times S}$ is in Jordan normal form if all entries not on the diagonal and the superdiagonal are zero and*

$$A = \begin{pmatrix} A_1 & & \\ & \ddots & \\ & & A_p \end{pmatrix}, \quad A_i = \begin{pmatrix} \lambda & 1 & & \\ & \lambda & \ddots & \\ & & \ddots & 1 \\ & & & \lambda \end{pmatrix}$$

*where $\lambda \in \sigma(A)$. It can be shown that for any $A \in \mathbb{C}^{S\times S}$ there exist a basis $B \in \mathbb{C}^{S\times S}$ of $\mathbb{C}^S$ (over field $\mathbb{C}$) consisting of generalized eigenvectors of $A$ such that $B^{-1}AB$ is in Jordan normal form.*

Let $\Re(z)$ and $\Im(z)$ denote the real and imaginary part of any $z \in \mathbb{C}$, respectively, and $\mathbb{H} = \{(x, y) \in \mathbb{C} \mid y \geq 0\}$ be the upper half plane of $\mathbb{C}$. We continue by generalizing Theorem 12.

**Theorem 14.** *Let $A \in \mathbb{R}^{S\times S}$ and let $B^{-1}AB$ be a Jordan normal form of $A$. Let $B_\lambda = \{z_1^\lambda, \ldots, z_{\dim E^*_\lambda}^\lambda\} \subseteq \mathbb{C}^S$ be the generalized eigenvectors from $B$ that span $E^*_\lambda(A)$ over field $\mathbb{C}$. With*

$$B'_\lambda := \{\Re(z_i^\lambda) \mid z_i^\lambda \in B_\lambda\} \cup \{\Im(z_i^\lambda) \mid z_i^\lambda \in B_\lambda\}$$

*for all $\lambda \in \sigma(J(\mathbb{1})) \cap \mathbb{H}$, let $\mathcal{B}_\lambda = \{u_1^\lambda, \ldots, u_{d_\lambda}^\lambda\}$ be a basis of $\langle B'_\lambda\rangle$ over field $\mathbb{R}$. Then, for any BDE space $\mathcal{U}_{\mathcal{H}}$, there are linearly independent vectors $w_1^\lambda, \ldots, w_{k_\lambda}^\lambda \in \langle \mathcal{B}_\lambda\rangle$ over field $\mathbb{R}$ with $0 \leq k_\lambda \leq d_\lambda$ so that $\mathcal{U}_{\mathcal{H}} = \bigoplus_\lambda \langle w_1^\lambda, \ldots, w_{k_\lambda}^\lambda\rangle$ over field $\mathbb{R}$.*

The above result suggests the following. First, calculate a basis $B \in \mathbb{C}^{S\times S}$ underlying a Jordan normal form of $J(\mathbb{1})$. Afterwards, compute $\mathcal{B}_\lambda$ from $B \in \mathbb{C}^{S\times S}$ for all $\lambda \in \sigma(J(\mathbb{1})) \cap \mathbb{H}$. Then, any BDE space $\mathcal{U}_{\mathcal{H}}$ is spanned by vectors $w_i^\lambda$ where $w_i^\lambda = \sum_{j=1}^{d_\lambda} \eta_j^{(\lambda,i)} u_j^\lambda$ with $\eta_j^{(\lambda,i)} \in \mathbb{R}$ and $1 \leq i \leq k_\lambda$.

**Example.** *A possible Jordan decomposition $D = B^{-1}J(\mathbb{1})B$ of $J(\mathbb{1})$ from (2) is*

$$D = \begin{pmatrix} 0 & 0 & 0 \\ 0 & \frac{3}{2} - \frac{\sqrt{3}}{2}i & 0 \\ 0 & 0 & \frac{3}{2} + \frac{\sqrt{3}}{2}i \end{pmatrix}$$
$$B = \begin{pmatrix} 1 & -1 + \sqrt{3}i & -1 - \sqrt{3}i \\ 1 & -1 - \sqrt{3}i & -1 + \sqrt{3}i \\ 1 & 2 & 2 \end{pmatrix}$$

*From this, we infer that $\sigma(J(\mathbb{1})) = \{0, \frac{3}{2} - \frac{\sqrt{3}}{2}i, \frac{3}{2} + \frac{\sqrt{3}}{2}i\}$. Applying the construction of Theorem 14, we obtain*

$$\mathcal{B}_0 = \{(1, 1, 1)^T\}$$
$$\mathcal{B}_{\frac{3}{2}+\frac{\sqrt{3}}{2}i} = \{(-1, -1, 2)^T, (-\sqrt{3}, \sqrt{3}, 0)^T\} \qquad (3)$$

From now on, let us fix some BDE partition $\mathcal{H}$ of $S$ and write $\mathcal{U}_{\mathcal{H}} = \bigoplus_\lambda \langle w_1^\lambda, \ldots, w_{k_\lambda}^\lambda\rangle$. Our goal is to provide an algorithm that finds $\mathcal{H}$ by establishing $k_\lambda$ and $w_1^\lambda, \ldots, w_{k_\lambda}^\lambda$ for all $\lambda \in \sigma(J(\mathbb{1})) \cap \mathbb{H}$. To this end, we will need the following.

**Definition 16.** *Given some $v \in \mathbb{R}^S$, let $\mathcal{H}_v$ denote the finest partition of $S$ on which $v$ is constant, i.e. set $\mathcal{H}_v := S/{\approx_v}$ where $x \approx_v y$ if and only if $v_x = v_y$. Moreover, let $\mathbb{G}(\mathcal{W}) := \{\mathcal{H}_v \mid v \in \mathcal{W}\}$ for any subspace $\mathcal{W} \subseteq \mathbb{R}^S$.*

**Proposition 3.** *It holds that $\mathcal{H}$ refines $\mathcal{H}_v$ for any $v \in \mathcal{U}_{\mathcal{H}}$. More generally, $\mathcal{H}$ refines $S/(\approx_{v_1} \cap \ldots \cap \approx_{v_k})$ for any $v_1, \ldots, v_k \in \mathcal{U}_{\mathcal{H}}$. Crucially, $\mathcal{H}$ is the coarsest BDE partition that refines all $\mathcal{H}_{w_i^\lambda}$, where $\lambda \in \sigma(J(\mathbb{1})) \cap \mathbb{H}$ and $1 \leq i \leq k_\lambda$.*

Recall that Theorem 3 features a partition refinement algorithm that takes an initial partition $\mathcal{G}$ as input and returns the coarsest BDE partition that refines $\mathcal{G}$. Since $\mathcal{H}_{w_i^\lambda} \in \mathbb{G}(\langle u_1^\lambda, \ldots, u_{d_\lambda}^\lambda\rangle)$, the idea is to invoke our partition refinement algorithm with elements from the *finite* set $\bigcup_\lambda \mathbb{G}(\langle u_1^\lambda, \ldots, u_{d_\lambda}^\lambda\rangle)$ to perform the guided search. Our second main result is stated next and follows from Proposition 3.

**Theorem 15.** *Given a drift $f : \mathbb{R}^S \to \mathbb{R}^S$ and the underlying set of guiding partitions $\mathbb{G} := \bigcup_\lambda \mathbb{G}_\lambda := \bigcup_\lambda \mathbb{G}(\langle u_1^\lambda, \ldots, u_{d_\lambda}^\lambda\rangle)$, Algorithm 1 computes the set of all BDE partitions $\mathfrak{B}$ of $S$ by*

```
$\mathcal{V}_{\mathfrak{B}} \leftarrow \emptyset$
$\mathcal{H} \leftarrow$ compute the coarsest BDE partition that refines $\{S\}$
$\mathcal{V}_{todo} \leftarrow \mathcal{H}$
while $\mathcal{V}_{todo} \neq \emptyset$ do
  $\mathcal{H} \leftarrow$ pick some element of $\mathcal{V}_{todo}$
  for all $\mathcal{G} \in \mathbb{G}$ do
    $\mathcal{H}' \leftarrow$ get the coarsest BDE partition refining $\mathcal{H}$ and $\mathcal{G}$
    if $\mathcal{H}' \notin \mathcal{V}_{todo}$ and $\mathcal{H}' \notin \mathcal{V}_{\mathfrak{B}}$ then
      $\mathcal{V}_{todo} \leftarrow \mathcal{V}_{todo} \cup \{\mathcal{H}'\}$
    end if
  end for
  $\mathcal{V}_{todo} \leftarrow \mathcal{V}_{todo} \setminus \{\mathcal{H}\}$
  $\mathcal{V}_{\mathfrak{B}} \leftarrow \mathcal{V}_{\mathfrak{B}} \cup \{\mathcal{H}\}$
end while
return $\mathcal{V}_{\mathfrak{B}}$
```

**Algorithm 1:** Computes all BDE partitions of $S$ using the guiding set $\mathbb{G} = \bigcup_\lambda \mathbb{G}_\lambda$.

*invoking the partition refinement algorithm at most $|\mathfrak{B}| \cdot |\mathbb{G}| + 1$ times.*

The overall complexity of Algorithm 1 also depends on the drift and on how difficult it is to calculate the sets $\mathbb{G}_\lambda$. For instance, if the drift arises from a system of polynomials in $|S|$ variables with degree at most two and $|R|$ denotes the number of monomials present in all polynomials (as is the case in our applications of Section 5), then the partition refinement algorithm needs at most $\mathcal{O}(|R| \cdot |S| \cdot \log(|S|))$ steps [7]. By invoking an SMT solver, the partition refinement algorithm has been extended to a much richer class of drifts [6]. At the same time, however, it has been shown that the class is expressive enough to encode tautology, which is coNP-complete. Hence, efficiency holds only in the case of subclasses.

We wish to point out that $|\mathfrak{B}| \cdot |\mathbb{G}| + 1$ is a worst case bound that may be rarely attained in practice. To see why, assume for simplicity that $\mathcal{U}_\mathcal{H} = \langle w_1^\lambda, \ldots, w_{k_\lambda}^\lambda \rangle$ for some $\lambda \in \sigma(J(\mathbb{1})) \cap \mathbb{H}$. Then, indeed, in all examples provided in Section 5, we could observe that the number of subsets $W \subseteq \{w_1^\lambda, \ldots, w_{k_\lambda}^\lambda\}$ such that $\langle W \rangle$ is a BDE space was small. This implies that the coarsest BDE partition that refines $\mathcal{H}_{w_i^\lambda}$ for some $1 \leq i \leq k_\lambda$ is likely to be $\mathcal{H}$ itself. Put different, feeding the partition refinement algorithm with a partition underlying one single $\mathcal{H}_{w_i^\lambda}$ usually leads to a partition that refines almost all remaining partitions $\mathcal{H}_{w_j^\lambda}$, where $j \neq i$.

We now turn to the calculation of $\mathbb{G}_\lambda$, with $\lambda \in \sigma(J(\mathbb{1})) \cap \mathbb{H}$. For the ease of notation, we assume in the remainder of the paragraph that $S = \{1, \ldots, n\}$ for some $n \geq 1$. The following is easily seen.

**Remark 1.** *Theorem 14 ensures $\mathbb{G}(\mathcal{B}_\lambda) = \{\mathcal{H}_{u_1^\lambda}\}$ if $d_\lambda = 1$.*

In the case where $d_\lambda > 1$, however, there are infinitely many possible directions $w_i^\lambda / \|w_i^\lambda\|$, see also discussion before Theorem 12. Note, however, that if $u \in \langle \mathcal{B}_\lambda \rangle$ has two coordinates, say $i$ and $j$, that coincide, then there exists an $\eta \in \mathbb{R}^{d_\lambda}$ such that $\sum_{l=1}^{d_\lambda} \eta_l \cdot u_l^\lambda \cdot e_i = \sum_{l=1}^{d_\lambda} \eta_l \cdot u_l^\lambda \cdot e_j$, where $e_k$ denotes the vector whose $k$-th coordinate is 1 and all other coordinates are zero. This motivates the following.

**Definition 17.** *For any pair set $\mathcal{P} \subseteq \{1, \ldots, n\}^2$, set $\mathcal{L}_\mathcal{P}^\lambda := \{\eta \in \mathbb{R}^{d_\lambda} \mid \forall (i,j) \in \mathcal{P} \big[ \sum_{l=1}^{d_\lambda} \eta_l \cdot u_l^\lambda \cdot (e_i - e_j) = 0 \big] \}$.*

That is, $\mathcal{L}_\mathcal{P}^\lambda$ denotes the coordinates $\eta$ with respect to basis $u_1^\lambda, \ldots, u_{d_\lambda}^\lambda$ that yield vectors $u \in \langle \mathcal{B}_\lambda \rangle$ such that the $i$-th and the $j$-th coordinates of $u$ are equal for all $(i,j) \in \mathcal{P}$. Note that any set of pairs $\mathcal{P} \subseteq S^2$ induces the partition $\mathcal{H} = S/\mathcal{P}^*$, where $\mathcal{P}^*$ denotes the transitive closure of $\mathcal{P}$. Thus, for $\mathcal{P}$ arbitrary, the set $\mathcal{L}_\mathcal{P}^\lambda$ is the solution space of a linear system of equations $\Theta\eta = 0$ with $\Theta \in \mathbb{R}^{(n-m) \times m}$ and $m = |S/\mathcal{P}^*|$. We get $(n-m)$ rows because each block $\{i_1, \ldots, i_\nu\} \in S/\mathcal{P}^*$ induces $\nu - 1$ linear equations that ensure $u(e_{i_1} - e_{i_k}) = 0$ for all $2 \leq k \leq \nu$.

Our ultimate goal is to find all $\mathcal{H} \in \mathbb{G}_\lambda$ by performing a search on pair sets $\mathcal{P}$. Note that $\mathcal{P}$ can be seen as a list of constraints (namely, the pairs of coordinates that have to coincide) imposed on the linear combinations of $u_1^\lambda, \ldots, u_{d_\lambda}^\lambda$. In particular, by adding additional pairs to $\mathcal{P}$, the dimension of $\mathcal{L}_\mathcal{P}^\lambda$ can either stay the same or become smaller. (The dimension can stay the same, for instance, if $u(e_i - e_j) = 0$ whenever $u(e_j - e_k) = 0$. In such case, one gets $\mathcal{L}_{\{(i,j)\}}^\lambda = \mathcal{L}_{\{(i,j),(j,k)\}}^\lambda = \mathcal{L}_{\{(j,k)\}}^\lambda$, meaning that adding $(j,k)$ to $\{(i,j)\}$ (or $(i,j)$ to $\{(j,k)\}$) does not reduce the dimension.)

With this in mind, we make the following pivotal observation. Define for any set of pairs $\mathcal{P}$ the underlying closure as $\overline{\mathcal{P}} := \{(i,j) \mid \mathcal{L}_{\mathcal{P} \cup \{(i,j)\}} = \mathcal{L}_\mathcal{P}\}$. It is not hard to see that $\overline{\mathcal{P}}$ is the largest equivalence relation such that $\sum_{l=1}^{d_\lambda} \eta_l \cdot u_l^\lambda \cdot (e_i - e_j) = 0$ for all $(i,j) \in \overline{\mathcal{P}}$ and $\eta \in \mathcal{L}_\mathcal{P}$. Consequently, the solution space $\mathcal{L}_\mathcal{P}$ corresponds to the partition $S/\overline{\mathcal{P}}$ in $\mathbb{G}_\lambda$. Note also that $\mathcal{L}_{\overline{\mathrm{id}}} = \mathbb{R}^{d_\lambda}$ and that $\dim \mathcal{L}_{\overline{\mathcal{P}} \cup \{(i,j)\}} < \dim \mathcal{L}_{\overline{\mathcal{P}}} = \dim \mathcal{L}_\mathcal{P}$ for any pair set $\mathcal{P}$ and $(i,j) \notin \overline{\mathcal{P}}$. Consequently, by starting with the closure $\overline{\mathrm{id}}$, the set of all closures can be obtained recursively by visiting, for any computed closure $\mathcal{P}$, its underlying closures $\overline{\mathcal{P} \cup \{(i,j)\}}$, where $(i,j) \notin \mathcal{P}$.

The above observation is formalized in Algorithm 2. There, $\mathfrak{P}$ contains the closures that are processed in the current iteration of the main while loop in line 3. Since $\mathrm{id} = \{(i,i) \mid 1 \leq i \leq n\}$ corresponds to the maximal solution space $\mathbb{R}^{d_\lambda}$ and each iteration of the main while loop seeks to reduce the maximal dimension present in $\mathfrak{P}$, we initialize it with the closure of id. We now discuss the body of the main while loop. Lines 5-14 compute for each closure $\mathcal{P} \in \mathfrak{P}$ the underlying guiding partition $\{1, \ldots, n\}/\mathcal{P}$. In the case the solution space underlying a closure $\mathcal{P}$ has dimension one, adding any further pair $(i,j) \notin \mathcal{P}$ will lead to the trivial solution space $\mathcal{L}_{\mathcal{P} \cup \{(i,j)\}} = \emptyset$. Consequently, we can remove $\mathcal{P}$ from $\mathfrak{P}$. Lines 20-35 compute for each $\mathcal{P}' \in \{\mathcal{P} \cup \{(i,j)\} \mid \mathcal{P} \in \mathfrak{P}, (i,j) \notin \mathcal{P}\}$ the underlying closure $\overline{\mathcal{P}'}$ and add it to $\mathfrak{P}'$. Crucially, Line 31 ensures that no closure is added more than once to $\mathfrak{P}'$. Although this check does not improve the worst case bounds, it allows for a substantial speed-up in practice.

Before giving a formal statement concerning the correctness and the running time of the algorithm, we first discuss it on an example.

**Example.** *Let us apply Algorithm 2 to the bases given in (3). The case $\mathcal{B}_0 = \{(1,1,1)^T\}$ yields $\mathbb{G}_0 = \{\{\{1,2,3\}\}\}$, so let us focus on $\mathcal{B}_\lambda = \{(-1,-1,2)^T, (-\sqrt{3}, \sqrt{3}, 0)^T\}$ with $\lambda = \frac{3}{2} + \frac{\sqrt{3}}{2}i$. We first calculate $\mathcal{L}_{\{(1,2)\}}^\lambda$, $\mathcal{L}_{\{(1,3)\}}^\lambda$ and $\mathcal{L}_{\{(2,3)\}}^\lambda$. By definition, $(\eta_1, \eta_2) \in \mathcal{L}_{\{(1,2)\}}^\lambda$ whenever*

$$0 = \Big(\eta_1(-1,-1,2)^T + \eta_2(-\sqrt{3}, \sqrt{3}, 0)^T\Big)(e_1 - e_2),$$

*which is equivalent to $-\eta_1 - \sqrt{3}\eta_2 = -\eta_1 + \sqrt{3}\eta_2$. That is, $\mathcal{L}_{\{(1,2)\}}^\lambda = \{(\eta_1, \eta_2) \mid \eta_2 = 0\}$. Further, we infer that $(\eta_1, \eta_2) \in \mathcal{L}_{\{(1,3)\}}^\lambda$ if and only if $-\eta_1 - \sqrt{3}\eta_2 = 2\eta_1$, yielding $\mathcal{L}_{\{(1,3)\}}^\lambda = \{(\eta_1, \eta_2) \mid \eta_2 = -\sqrt{3}\eta_1\}$. In a similar fashion one obtains $\mathcal{L}_{\{(2,3)\}}^\lambda = \{(\eta_1, \eta_2) \mid \eta_2 = \sqrt{3}\eta_1\}$. The vectors induced by $\mathcal{L}_{\{(1,2)\}}^\lambda$, $\mathcal{L}_{\{(1,3)\}}^\lambda$ and $\mathcal{L}_{\{(2,3)\}}^\lambda$ are thus given by*

$$(\eta_1, \eta_1, 2\eta_1) = \eta_1(-1,-1,2)^T + 0 \cdot (-\sqrt{3}, \sqrt{3}, 0)^T$$
$$(2\eta_1, -4\eta_1, 2\eta_1) = \eta_1(-1,-1,2)^T - \sqrt{3}\eta_1(-\sqrt{3}, \sqrt{3}, 0)^T$$
$$(-4\eta_1, 2\eta_1, 2\eta_1) = \eta_1(-1,-1,2)^T + \sqrt{3}\eta_1(-\sqrt{3}, \sqrt{3}, 0)^T,$$

```
1:  $\mathfrak{P} \leftarrow \{\overline{\mathit{id}}\}$
2:  $\mathbb{G}_\lambda \leftarrow \emptyset$
3:  **while true do**
4:      // Compute the guiding partition underlying each closure
5:      **for all** $\mathcal{P} \in \mathfrak{P}$ **do**
6:          **if** $\mathcal{P} \notin \mathbb{G}_\lambda$ **then**
7:              $\mathbb{G}_\lambda \leftarrow \mathbb{G}_\lambda \cup \{\mathcal{P}\}$
8:          **end if**
9:          $\mathbb{S}^\lambda_\mathcal{P} \leftarrow$ construct the linear system inducing $\mathcal{L}^\lambda_\mathcal{P}$
10:         $\mathbb{B}^\lambda_\mathcal{P} \leftarrow$ compute the basis of $\mathcal{L}^\lambda_\mathcal{P}$ by solving $\mathbb{S}^\lambda_\mathcal{P}$
11:         **if** dim$\mathcal{L}^\lambda_\mathcal{P} = 1$ **then**
12:             $\mathfrak{P} \leftarrow \mathfrak{P} \setminus \{\mathcal{P}\}$
13:         **end if**
14:     **end for**
15:     **if** $\mathfrak{P} = \emptyset$ **then**
16:         **break**
17:     **end if**
18:     // Generate new closures from the current ones
19:     $\mathfrak{P}' \leftarrow \emptyset$
20:     **while** $\mathfrak{P} \neq \emptyset$ **do**
21:         $\mathcal{P} \leftarrow$ pick some element of $\mathfrak{P}$
22:         $\mathfrak{P} \leftarrow \mathfrak{P} \setminus \{\mathcal{P}\}$
23:         **for all** $(i,j) \notin \mathcal{P}$ **do**
24:             $\mathcal{P}' \leftarrow \mathcal{P} \cup \{(i,j)\}$
25:             $\mathbb{S}^\lambda_{\mathcal{P}'} \leftarrow$ construct the linear system inducing $\mathcal{L}^\lambda_{\mathcal{P}'}$
26:             $\mathbb{B}^\lambda_{\mathcal{P}'} \leftarrow$ compute the basis of $\mathcal{L}^\lambda_{\mathcal{P}'}$ by solving $\mathbb{S}^\lambda_{\mathcal{P}'}$
27:             **if** dim$\mathcal{L}^\lambda_{\mathcal{P}'} = 0$ **then**
28:                 **continue**
29:             **end if**
30:             $\mathcal{P}' \leftarrow \overline{\mathcal{P}'}$
31:             **if** $\mathcal{P}' \notin \mathfrak{P}'$ **then**
32:                 $\mathfrak{P}' \leftarrow \mathfrak{P}' \cup \{\mathcal{P}'\}$
33:             **end if**
34:         **end for**
35:     **end while**
36:     $\mathfrak{P} \leftarrow \mathfrak{P}'$
37: **end while**
38: **return** $\mathbb{G}_\lambda$
```

**Algorithm 2:** Computes the guiding set $\mathbb{G}_\lambda$.

*where $\eta_1 \in \mathbb{R}$, respectively. Armed with this, we are in a position to describe Algorithm 2.*

*Since $\mathcal{L}^\lambda_{\{(1,2)\}}$, $\mathcal{L}^\lambda_{\{(1,3)\}}$ and $\mathcal{L}^\lambda_{\{(2,3)\}}$ are pairwise different subspaces of $\mathbb{R}^2$, it holds that $\overline{id} = id$ and the (trivial) guiding partition $\{\{1\},\{2\},\{3\}\}$ is added to $\mathbb{G}_\lambda$ by lines 5-14. Afterwards, lines 20-35 set $\mathfrak{P}'$ to $\{id \cup \{(1,2)\}, id \cup \{(1,3)\}, id \cup \{(2,3)\}\}$ and the solutions spaces $\mathcal{L}^\lambda_{\{(1,2)\}}$, $\mathcal{L}^\lambda_{\{(1,3)\}}$ and $\mathcal{L}^\lambda_{\{(2,3)\}}$ are computed. The first iteration finishes by giving $\mathfrak{P}$ the value of $\mathfrak{P}'$.*

*During the second iteration, lines 5-14 add $\{\{1,2\},\{3\}\}$, $\{\{1,3\},\{2\}\}$, $\{\{1\},\{2,3\}\}$ to $\mathbb{G}_\lambda$ while emptying $\mathfrak{P}$. This is because $\mathcal{L}^\lambda_{\{(1,2)\}}$, $\mathcal{L}^\lambda_{\{(1,3)\}}$ and $\mathcal{L}^\lambda_{\{(2,3)\}}$ are all of dimension one. The algorithm terminates then in the second iteration with $\mathbb{G}_\lambda = \{\{\{1\},\{2\},\{3\}\}, \{\{1,2\},\{3\}\}, \{\{1,3\},\{2\}\}, \{\{1\},\{2,3\}\}\}$.*

The following crucial result can be shown.

**Theorem 16.** *If $d_\lambda = 1$, then $\mathbb{G}_\lambda = \{u_1^\lambda\}$. Instead, if $d_\lambda > 1$, Algorithm 2 computes $\mathbb{G}_\lambda$ in at most $\mathcal{O}(n^{2d_\lambda+4})$ steps.*

Before giving the proof, we again stress that the above bounds are only attained under the pessimistic assumption that lines 20-35 lead to pairwise different closures. In all models we have considered in Section 5, however, the algorithm showed decent performance because each iteration had a substantial number of closures $\mathcal{P}, \mathcal{P}' \in \mathfrak{P}$ and pairs $(i,j) \notin \mathcal{P}$, $(i',j') \notin \mathcal{P}'$ such that $\overline{\mathcal{P} \cup \{(i,j)\}} = \overline{\mathcal{P}' \cup \{(i',j')\}}$. In particular, in the case of a highly symmetric network with $n = 12$ and $\max_\lambda d_\lambda = 7$, we were able to calculate all BDE partitions in around 40 min on a 2.6 GHz Intel Core i5 machine with 4GB of RAM.

We also argue that, although $d_\lambda$ can be equal to $n$, usually $\max_\lambda d_\lambda \ll n$. In fact, it is well-known that the set of $n \times n$ matrices that have $n$ pairwise different eigenvalues (which suffices $\max_\lambda d_\lambda \leq 2$) is dense in $\mathbb{R}^{n \times n}$.

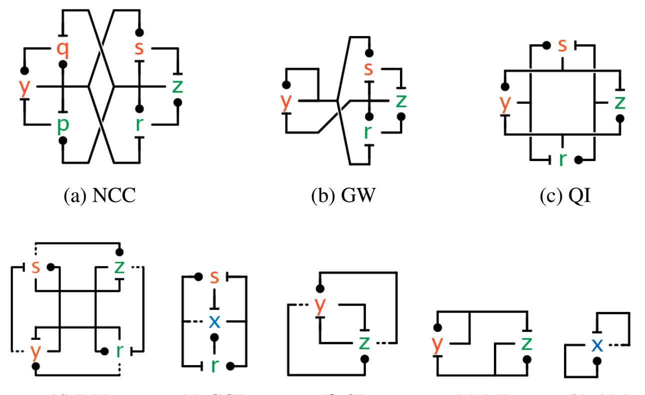
**Remark 2.** *Algorithm 2 allows for a bounded computation if the additional break condition $\binom{n}{2} \cdot |\mathfrak{P}| > \texttt{bound}$ is added in line 15. Although a bounded computation may fail to find all BDE partitions, the number of missed partitions can be expected to be small if $\texttt{bound}$ is of decent size because, as pointed out in the discussion after Theorem 15, Algorithm 1 needs usually only a subset of the guiding set in order to find all BDE partitions. Moreover, note that Algorithm 1 returns always a set of BDE partitions, meaning that the bounded computation is sound.*

*Proof.* The correctness of the algorithm follows from the discussion after Definition 17. In the following, all line numbers refer to Algorithm 2. Let $\mathfrak{P}_\nu$ be the content of $\mathfrak{P}$ at the beginning of iteration $\nu$ of the while loop in line 3, where the first iteration has index $\nu = 0$. By construction, Algorithm 2 ensures that $\dim \mathcal{L}_\mathcal{P} \leq d_\lambda - \nu$ for any $\mathcal{P} \in \mathfrak{P}_\nu$. Since closures whose solution space has dimension one are removed in lines 5-14, this ensures that there are at most $d_\lambda$ iterations of the main while loop. Moreover, we note that lines 20-35 increase the number of pair sets at most by the factor $\binom{n}{2} \leq n^2$, hence $|\mathfrak{P}_\nu| \leq |\mathfrak{P}_{\nu-1}| \cdot n^2$ for all $\nu \geq 1$, thus yielding $|\mathfrak{P}_\nu| \leq (n^2)^\nu$ for all $\nu \geq 0$. By combining both statements, we infer that $\mathfrak{P}$ cannot exceed the size of $|\mathfrak{P}_{d_\lambda - 1}| \leq (n^2)^{d_\lambda - 1} = n^{2(d_\lambda - 1)}$. We now take a closer look at the algorithm. Any closure $\mathcal{P}$ is stored in terms of the corresponding partition $\{1,\ldots,n\}/\mathcal{P}$ and each partition is encoded by a row vector $p \in \mathbb{R}^{1 \times n}$ where $p(i)$ denotes the smallest index of the block to which $i$ belongs (e.g., the partition $\{\{1,3\},\{2,4\},\{5\}\}$ is encoded by the row vector $(1,2,1,2,5)$). With this, lines 9, 10, 24, 25 and 26 can be solved in $\mathcal{O}(n^3)$ steps. The computation in line 30 is accomplished in the following way. With $v_1,\ldots,v_d$ being the vectors of $\mathbb{B}^\lambda_{\mathcal{P}'}$, we first calculate $\mathcal{H}_{v_1},\ldots,\mathcal{H}_{v_d}$. Afterwards, we compute the coarsest partition that refines all $\mathcal{H}_{v_1},\ldots,\mathcal{H}_{v_d}$. Since $d \leq n$, line 30 thus needs at most $\mathcal{O}(n^4)$ steps. We further note that the number of entries in $\mathfrak{P}$, $\mathfrak{P}'$ and $\mathbb{G}_\lambda$ is bounded by $\sum_{i=0}^{\nu}(n^2)^i \leq (n^2)^{\nu+1}$ in the $\nu$-th iteration. Hence, if $\mathfrak{P}$, $\mathfrak{P}'$ and $\mathbb{G}_\lambda$ are implemented as red-black trees, the underlying methods `add`, `remove` and `in` will cost at most $\mathcal{O}(\log((n^2)^{\nu+1}) \cdot n) \leq \mathcal{O}(n^2 \cdot \log(n))$. (The multiplication by $n$ is due to the sorting with respect to partitions.)

This shows that the $\nu$-th iteration of the main while loop costs at most $(n^2)^\nu \cdot \mathcal{O}(n^4)$ steps. Since lines 20-35 are not invoked in the last iteration and there are at most $d_\lambda$ iterations, the total number of operations is at most $\sum_{0 \leq \nu \leq d_\lambda - 1}(n^2)^\nu \cdot \mathcal{O}(n^4) \leq \mathcal{O}(n^{2d_\lambda+4})$. □

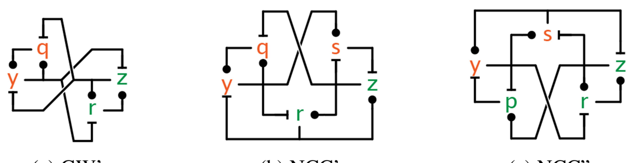
The computation of a Jordan decomposition of a matrix $A \in \mathbb{R}^{n \times n}$ takes $\mathcal{O}(n^3)$ steps. Instead, $J(\mathbb{1})$ can be calculated in polynomial time if the drift $f$ is such that each $f_i : \mathbb{R}^n \to \mathbb{R}$ is a polynomial in several variables. Consequently, Algorithm 1 readily applies to polynomial ODE systems.

***Calculating all emulations.*** With Algorithm 1 it is easy to decide whether two ODE systems are related by means of an emulation. Fix a source drift $f : \mathbb{R}^S \to \mathbb{R}^S$ and a target drift $\hat{f} : \mathbb{R}^{\hat{S}} \to \mathbb{R}^{\hat{S}}$ such that $S \cap \hat{S} = \emptyset$. Since one can always rename variables, we can make this assumption without loss of generality.



(a) NCC (b) GW (c) QI (d) DN (e) CCR (f) SI (g) MI (h) AM

Figure 1: Unimodal influence networks from [3].



(a) GW' (b) NCC' (c) NCC"

Figure 2: Unimodal influence networks not studied in [3].

Recall that Proposition 1 ensures that $\mu : S \to \hat{S}$ is an emulation if and only if $\{\mu^{-1}(\hat{x}) \cup \{\hat{x}\} \mid \hat{x} \in \hat{S}\}$ is a BDE partition of $S \cup \hat{S}$. Thus, if we apply Algorithm 1 to the union drift $g : \mathbb{R}^{S \cup \hat{S}} \to \mathbb{R}^{S \cup \hat{S}}$ where $g_x(w) = f_x(w_{|S})$ and $g_{\hat{x}}(w) = \hat{f}_{\hat{x}}(w_{|\hat{S}})$ for all $w \in \mathbb{R}^{S \cup \hat{S}}$, all we have to do is to check whether there exist BDE partitions $\mathcal{H}$ of $S \cup \hat{S}$ that satisfy $|\mathcal{H}| = |\hat{S}|$ and $|H \cap S| \geq |H \cap \hat{S}| = 1$ for all $H \in \mathcal{H}$.

Another approach is to apply Algorithm 1 to the source drift $f : \mathbb{R}^S \to \mathbb{R}^S$. In the case there are no BDE partitions $\mathcal{H}$ of $S$ that have exactly $|\hat{S}|$ blocks, the source and the target drifts cannot be related to each other. In the case there are, one has to decide for each $\mathcal{H}$ of $S$ that satisfies $|\mathcal{H}| = |\hat{S}|$ whether the target drift is isomorphic to the aggregated drift underlying $f$ and $\mathcal{H}$. That is, one has to decide whether there exist a $\mu : S \to S$ with $\mathcal{H} = \{\mu^{-1}(x) \mid x \in \mu(S)\}$ and a renaming $\eta : \mu(S) \to \hat{S}$ such that $\hat{f}_{\hat{x}}(\hat{v}) = f_{\eta^{-1}(\hat{x})}(\hat{v} \circ \eta \circ \mu)$ for all $\hat{v} \in \mathbb{R}^{\hat{S}}$ and $\hat{x} \in \hat{S}$. This has the advantage that Algorithm 1 considers the target drift $f$ instead of the union drift $g$, at the expense of deciding whether two drifts are isomorphic. We use this approach in Section 5 because all target networks considered there are of moderate size.

## 5. Applications

Figure 1 shows formal pictorial definitions of networks studied in [3] and known as *influence networks*. Each symbol (e.g., $x$ and $y$) is a node that corresponds to three distinct chemical species (e.g., $x_0$, $x_1$, $x_2$, and $y_0$, $y_1$, $y_2$) and at most four chemical reactions. The reactions depend on how nodes are connected in the influence network. Each node can have a connection at each terminal: *high output* (solid line), representing the species with subscript 0, *low output* (dashed line), representing the species with subscript 2, *activation input* (circle) and *inhibition input* (bar). Species with index 1 introduce nonlinearity in transitions [3] and are never otherwise connected to the network. If $i$ and $a$ are the inhibitor and activation input species for node $x$, respectively, then $x$ is associated with the following reactions:

$$x_0 + i \to^{\alpha_{01}} i + x_1, \qquad x_1 + i \to^{\alpha_{12}} i + x_2,$$
$$x_2 + a \to^{\alpha_{21}} a + x_1, \qquad x_1 + a \to^{\alpha_{10}} a + x_0,$$

where $\alpha_{01}, \alpha_{12}, \alpha_{21}, \alpha_{10}$ are given rate coefficients of node $x$. An influence network is called *unimodal* if all species are activated or inhibited by at most one species.

The AM network models a cell cycle switch that is needed to avoid genetic instability during replication [3]. One of the main results of [3] was to show, by hand, that all (unimodal) influence networks in Figure 1 emulate AM, which essentially means that they implement more complex versions of a cell cycle switch. Using larger networks instead of smaller ones of apparently equal function can be beneficial for a number of practical reasons, including enhanced stability with respect to stochastic noise [5].

By invoking Algorithm 1, we were able to automatically rediscover the emulations from [3] (where any rate was set to 1). In particular we confirmed that all networks in Figure 1 emulate AM. Moreover, we applied our algorithm also to networks in Figure 2, which are further possible evolutionary transitions between cell cycle switches that were not covered in [3]. This provided us with a computer aided proof for the fact that also those networks emulate AM. Indeed we found the following union BDE partitions:

$$\begin{aligned}
\mathcal{H}_{\text{GW'}\cup\text{AM}} &= \{\{x_0, q_0, r_2, y_0, z_2\}, \{x_1, q_1, r_1, y_1, z_1\}, \\
&\quad \{x_2, q_2, r_0, y_2, z_0\}\}, \\
\mathcal{H}_{\text{NCC'}\cup\text{AM}} &= \{\{x_0, q_2, r_0, s_2, y_2, z_0\}, \{x_1, q_1, r_1, s_1, y_1, z_1\}, \\
&\quad \{x_2, q_0, r_2, s_0, y_0, z_2\}\}, \\
\mathcal{H}_{\text{NCC"}\cup\text{AM}} &= \{\{x_0, p_2, r_2, s_0, y_0, z_2\}, \{x_1, p_1, r_1, s_1, y_1, z_1\}, \\
&\quad \{x_2, p_0, r_0, s_2, y_2, z_0\}\}.
\end{aligned}$$

The algorithm can also be used to verify that no emulation can relate two unimodal influence networks. In such a case, one can infer that two networks do not share certain biological properties, such as switches. For instance we were able to verify that NCC does not emulate GW in the case where all rate coefficients are set to one, while they both emulate AM in those conditions.

These tests are replicable using our prototype implementation available at `http://sysma.imtlucca.it/tools/cage/`.

As with all quantitative notions of model comparison, emulation is sensitive to rate parameters. We now study how a more parametric analysis is possible in the case of unimodal influence networks.

First we show that the assumptions made in Theorem 9 can be dropped. That is, the change of rates theorem from [3] carries over to flux morphisms if the target is a unimodal influence network.

**Theorem 17.** *Fix a source CRN* $(S, R)$*, a target unimodal influence network* $(\hat{S}, \hat{R})$ *and let* $(\mu, \sigma) \in (S \to \hat{S}) \times (R/\!\sim\, \hookrightarrow \hat{R}/\!\sim)$ *be a flux morphism. Then, for any change of rates* $\hat{\iota} : \hat{R} \to \hat{R}'$*, there exists a change of rates* $\iota : R \to R'$ *such that* $(\mu, \sigma') : (S, R') \hookrightarrow (\hat{S}, \hat{R}')$*, where* $\sigma'$ *is induced by* $\mu : S \to \hat{S}$ *and condition (1), is a flux morphism.*

Dually, it is interesting to ask whether the *absence* of emulations holds true for all possible rates. Unfortunately, it is not clear how to lift Algorithm 2 in the case when $J(\mathbb{1})$ has variables as entries because this leads to parametric bases $\mathcal{B}_\lambda$. Another problem is that, in general, the eigenvalues of a matrix with parameters cannot be expressed in terms of formulae because of Abel's impossibility theorem. Instead, we tackle this problem in a different way.

**Definition 18.** *Let* $(S, R)$ *and* $(\hat{S}, \hat{R})$ *be some unimodal influence networks. An emulation* $\mu : S \to \hat{S}$ *is triplet preserving if, for any* *triplet $x_0, x_1, x_2$ of $(S, R)$ there exists a triplet $\hat{x}_0, \hat{x}_1, \hat{x}_2$ of $(\hat{S}, \hat{R})$ such that $\mu(x_1) = \hat{x}_1$ and $\{\mu(x_0), \mu(x_2)\} = \{\hat{x}_0, \hat{x}_2\}$.*

In the case of unimodal influence networks, only triplet preserving emulations reveal biologically meaningful relations (since subscript-one species are intermediate). The next result is based on Theorem 17 and allows one to argue about *families* of networks by considering a *single* pair of networks with unit rates.

**Theorem 18.** *Let $(S, R)$ and $(\hat{S}, \hat{R})$ be unimodal influence networks and assume that $\mu : S \to \hat{S}$ is a triplet preserving emulation. Then, $\mu$ is also an emulation of the networks $(S, R')$ and $(\hat{S}, \hat{R}')$, where $R'$ and $\hat{R}'$ arise from $R$ and $\hat{R}$, respectively, by changing all rate coefficients to one.*

For instance, since NCC does not emulate GW in the case where all rate coefficients are set to one, Theorem 18 ensures that, *for any choice* of rate coefficients, there exists no triplet preserving emulation that relates NCC to GW. In a similar fashion, we were able to show that there exists no triplet preserving emulation from NCC to DN and from CCR to MI, respectively.

## 6. Conclusion

We have developed a framework for model comparison of chemical reaction networks based on the notion of emulation as a mapping between species from a source CRN into a target CRN. We characterized this semantic property in terms of structural conditions through flux morphisms. In addition to being useful in applications, this approach provides an explanation in terms of *discrete* structures of behavior evolving as ordinary differential equations over *continuous* time and state spaces.

We argued that the problem of finding emulations cannot be simply cast to the more traditional question of computing a largest behavioural equivalence, i.e. backward differential equivalence (BDE), because one would further require the constraint that every equivalence class contain exactly one species of the target CRN. Instead, we developed a new algorithm for computing emulations, that owes much to a novel (in computer science) geometric interpretation of an equivalence relation for differential equations.

For quantitative notions of model comparison, it has long been argued that *exact* variants such as ours are too strong because they heavily depend on the choice of the numerical parameters, suggesting approximate notions instead (e.g., [12, 14, 20, 26]). Here we have started to tackle this issue by finding a *family* of models to which an emulation found with a given choice of parameter carries over. In the special but biologically relevant case of (unimodal) influence networks, instead, we proved that the absence of an emulation for a specific choice of rates implies absence for any choice of rates. In the longer term, we believe that our contribution can be seen as a stepping stone on which to build approximate variants understood as appropriate perturbations on exact comparisons.

***Acknowledgments.*** We thank the anonymous referees for helpful comments. This work was partially supported by the EU project QUANTICOL, 600708. L. Cardelli is partially funded by a Royal Society Research Professorship.

## References

[1] M. Boreale. Weighted bisimulation in linear algebraic form. In *CONCUR*, 2009.

[2] N. M. Borisov, N. I. Markevich, J. B. Hoek, and B. N. Kholodenko. Signaling through receptors and scaffolds: Independent interactions reduce combinatorial complexity. *Biophysical Journal*, 89(2):951–966, 2005.

[3] L. Cardelli. Morphisms of reaction networks that couple structure to function. *BMC Systems Biology*, 8(1), 2014.

[4] L. Cardelli, M. Tribastone, M. Tschaikowski, and A. Vandin. Forward and backward bisimulations for chemical reaction networks. In *CONCUR*, 2015.

[5] L. Cardelli, A. Csikász-Nagy, N. Dalchau, M. Tribastone, and M. Tschaikowski. Noise reduction in complex biological switches. *Scientific Reports*, 6:20214, 2016.

[6] L. Cardelli, M. Tribastone, M. Tschaikowski, and A. Vandin. Symbolic computation of differential equivalences. In *POPL*, 2016.

[7] L. Cardelli, M. Tribastone, M. Tschaikowski, and A. Vandin. Efficient syntax-driven lumping of differential equations. In *TACAS*, 2016.

[8] H. Conzelmann, J. Saez-Rodriguez, T. Sauter, B. Kholodenko, and E. Gilles. A domain-oriented approach to the reduction of combinatorial complexity in signal transduction networks. *BMC Bioinformatics*, 7(1):34, 2006.

[9] H. Conzelmann, D. Fey, and E. Gilles. Exact model reduction of combinatorial reaction networks. *BMC Systems Biology*, 2(1):78, 2008.

[10] V. Danos and C. Laneve. Formal molecular biology. *TCS*, 325(1): 69–110, 2004.

[11] V. Danos, J. Feret, W. Fontana, R. Harmer, and J. Krivine. Abstracting the differential semantics of rule-based models: Exact and automated model reduction. In *LICS*, 2010.

[12] J. Desharnais, V. Gupta, R. Jagadeesan, and P. Panangaden. Metrics for labeled Markov systems. In *CONCUR*, 1999.

[13] J. Desharnais, A. Edalat, and P. Panangaden. Bisimulation for Labelled Markov Processes. *Information and Computation*, 179(2):163–193, 2002.

[14] J. Desharnais, V. Gupta, R. Jagadeesan, and P. Panangaden. Metrics for labelled markov processes. *TCS*, 318(3):323–354, 2004.

[15] D. Doty. Timing in chemical reaction networks. In *SODA*, 2014.

[16] J. Feret, T. Henzinger, H. Koeppl, and T. Petrov. Lumpability abstractions of rule-based systems. *TCS*, 431:137–164, 2012.

[17] S. Gay, S. Soliman, and F. Fages. A graphical method for reducing and relating models in systems biology. *Bioinformatics*, 26(18):i575–i581, 2010.

[18] M. Lakin, A. Phillips, and D. Stefanovic. Modular verification of DNA strand displacement networks via serializability analysis. In *DNA Computing and Molecular Programming*, volume 8141 of *LNCS*, pages 133–146, 2013.

[19] M. Lakin, D. Stefanovic, and A. Phillips. Modular verification of chemical reaction network encodings via serializability analysis. *TCS*, 2015. In press.

[20] K. G. Larsen, R. Mardare, and P. Panangaden. Taking it to the limit: Approximate reasoning for markov processes. In *MFCS*, 2012.

[21] G. Li and H. Rabitz. A general analysis of exact lumping in chemical kinetics. *Chemical Engineering Science*, 44(6):1413 – 1430, 1989.

[22] A. Regev and E. Shapiro. Cellular abstractions: Cells as computation. *Nature*, 419(6905):343–343, 9 2002.

[23] C. T. Seung Woo Shin and E. Winfree. Verifying chemical reaction network implementations: A pathway decomposition approach. In *VEMDP*, Vienna Summer of Logic, 2014.

[24] D. Soloveichik, G. Seelig, and E. Winfree. DNA as a universal substrate for chemical kinetics. *PNAS*, 107(12):5393–5398, 2010.

[25] J. J. Tyson and B. Novák. Functional motifs in biochemical reaction networks. *Annual review of physical chemistry*, 61:219, 2010.

[26] F. van Breugel and J. Worrell. Approximating and computing behavioural distances in probabilistic transition systems. *TCS*, 360(1–3): 373–385, 2006.

[27] A. van der Schaft. Equivalence of dynamical systems by bisimulation. *IEEE Transactions on Automatic Control*, 49(12):2160–2172, 2004.

[28] G. Zavattaro and L. Cardelli. Termination problems in chemical kinetics. In *CONCUR*, 2008.