Quantitative Abstractions for Collective Adaptive Systems

Andrea Vandin and Mirco Tribastone

IMT School for Advanced Studies Lucca, Italy

Abstract. Collective adaptive systems (CAS) consist of a large number of possibly heterogeneous entities evolving according to local interactions that may operate across multiple scales in time and space. The adaptation to changes in the environment, as well as the highly dispersed decision-making process, often leads to emergent behaviour that cannot be understood by simply analysing the objectives, properties, and dynamics of the individual entities in isolation.
As with most complex systems, modelling is a phase of crucial importance for the design of new CAS or the understanding of existing ones. Elsewhere in this volume the typical workflow of formal modelling, analysis, and evaluation of a CAS has been illustrated in detail. In this chapter we treat the problem of efficiently analysing *large-scale* CAS for quantitative properties. We review algorithms to automatically reduce the dimensionality of a CAS model preserving modeller-defined state variables, with focus on descriptions based on systems of ordinary differential equations. We illustrate the theory in a tutorial fashion, with running examples and a number of more substantial case studies ranging from crowd dynamics, epidemiology and biological systems.

1 Introduction

Distinctive features of collective adaptive systems (CAS) are the presence of a large number of entities with their own properties, objectives, and behaviour, that interact with each other and with the environment in such a way that the resulting global dynamics arises as an emergent property that cannot be directly inferred from the study of individuals in isolation. To ensure that a CAS design meets the desired properties, or to accurately understand the behaviour of existing CAS, it is of crucial importance to be able to reason about a (possibly huge) system as a whole. In this context, the modelling phase clearly plays an important role, as it does with any system characterised by high complexity.

Quantitative abstractions. The focus of this chapter is on quantitative modelling of CAS. Due to their heterogeneity and scale, CAS introduce a number of difficult challenges, the most notable of which is the problem of state space explosion that is typically incurred when analysing large collectives of entities. Elsewhere in this volume are contributions to a prototypical design and modelling workflow for CAS which take scalability and accuracy of the analysis into account.

The process algebra CARMA (cf. [9]) is explicitly designed to study collectives of agents evolving stochastically according to a continuous-time Markov chain (CTMC) model [57]; approximate analysis techniques based on hybrid or differential equation approximations are presented in [10]; effective approaches for dealing with the spatial dimension of CAS are reviewed in [38]; finally, model checking for spatial and temporal properties are discussed in [40]. Here, instead, we focus on techniques that crosscut the above phases of the modelling workflow. Namely, we consider the problem of obtaining suitable *abstractions* of dynamical models for CAS. We are motivated by the fact that, in real-world scenarios, the inherent system's complexity is so high that it may even defeat typically compact and effective model descriptions, such as those based on ordinary differential equations (ODEs).

Let us consider, for instance, the case of a bike-sharing system (BSS). This is a prototypical CAS [30], an instance of which has been also used as running example of [57]. Its quantitative analysis may be based on a CTMC model, which will however grow unfeasibly large in realistic settings since the state space has to cover (at least) all of the possible combinations of bike availabilities at each station. Deterministic approximations based on ODEs may come to the rescue, by more compactly associating one equation for each station and each possible link between two stations. In this case they would capture an estimate of the average number of bikes available at each station as well as of those in transit [37]. Clearly, if instantiated to a real-world large BSS such as London's, with over 700 stations, it would yield an ODE system of many thousands of equations, which is likely to drastically impact on the practical feasibility of the analysis. Furthermore, the analysis would become prohibitive if the modeller wished to track higher order moments than the averages, since the ODE system size grows polynomially with the number of variables of the original system (e.g., [32]).

Abstraction techniques may help tackle the dimensionality problem further. The basic idea is to obtain a representation of the original model projected onto a lower dimensional state space so as to allow a more efficient analysis. Due to the large scale involved in CAS models, there are four main desirable properties for an effective method:

P1. The abstraction should come with formal guarantees on the relationship between the abstract dynamics and the original one. This enables the modeller to use the abstract model with full confidence in the results of the analysis.
P2. The construction of the abstract model should be fully automatic, since the original model is likely to be unintelligible due to size.
P3. The method should be generic in order to be applicable to as a wide range of CAS models as possible.
P4. The abstract model should preserve user-defined observables of the original system. For instance, it should be possible to fully recover the dynamics of selected variables of the original model.

In this chapter we consider abstraction techniques that satisfy the above requirements for quantitative CAS models based on ODEs. However they can be

applicable also to CTMC models by studying the CTMC's equations of motion, which is a linear ODE system (e.g., [61]). In fact, we will discuss that these techniques can be somewhat seen as a generalisation of aggregation algorithms specific to CTMCs, based on the well-known notion of lumpability [14]. Languages and equivalences have been extensively studied for models based on CTMC semantics (e.g., [27]).

Since ODEs are a universal dynamical model, featuring in many diverse scientific branches including organic and inorganic chemistry, ecology, economics, epidemiology, systems biology, and control theory, reducing large scale ODEs has also a long-standing tradition (e.g., [2,48,62]). Here we offer a specific computer-science viewpoint on this subject, looking at ODE reduction as the problem of finding an appropriate equivalence relation over the ODE's state variables, borrowing ideas from the programming languages community and concurrency theory. Most of the results discussed here, summarised from [19,21,20], concern exact notions of aggregation. These may be lossy in that the dynamics of some original variables cannot be recovered in the abstract model, yet all the information in the abstract model is exactly related to the original variables; approximate notions of aggregation are an exciting future research direction.

Differential equivalences. The problem of minimising ODEs is interpreted as a quotienting up to some equivalence, akin to more classical models of computation based on labelled transition systems (LTS). We put forward the analogy between states of an LTS and ODE variables. The starting point is that of *differential equivalences*, relations between ODE variables that preserve their corresponding solutions in some appropriate sense. Here we consider two variants of differential equivalence, as first presented in [21].

In *forward differential equivalence* (FDE), an ODE system can be written for the variables that represent the equivalence classes, giving the sum of the solutions of its members at all time points t. Let us consider the example:

$$\dot{x}_1 = -x_1, \qquad \dot{x}_2 = k_1 \cdot x_1 - x_2, \qquad \dot{x}_3 = k_2 \cdot x_1 - x_3, \tag{1}$$

where k_1 and k_2 are constants and the 'dot' operator denotes the derivative.[1] It can be shown that $\{\{x_1\}, \{x_2, x_3\}\}$ is an FDE quotienting. Indeed, exploiting basic properties one gets

$$\dot{x}_1 = -x_1, \qquad \dot{(x_2 + x_3)} = \dot{x}_2 + \dot{x}_3 = (k_1 + k_2) \cdot x_1 - (x_2 + x_3). \tag{2}$$

By the change of variable $y = x_2 + x_3$, this is equivalent to writing

$$\dot{x}_1 = -x_1 \qquad\qquad \dot{y} = (k_1 + k_2) \cdot x_1 - y.$$

This quotient ODE model recovers the sum of the solutions of the variables in each equivalence class. Thus, setting the *initial condition* $y(0) = x_2(0) + x_3(0)$ yields that the solution satisfies $y(t) = x_2(t) + x_3(t)$ at all time points t.

[1] Throughout the paper we will work with *autonomous* ODE systems, which are not explicitly dependent on time.

Backward differential equivalence (BDE) equates variables that have the same solutions at all time points. In (1), $\{\{x_1\},\{x_2,x_3\}\}$ is also a BDE provided that $k_1 = k_2$. In this case, we obtain a quotient ODE by removing either equation between x_2 and x_3, say x_3, and rewriting every occurrence of x_3 as x_2:

$$\dot{x}_1 = -x_1 \qquad\qquad \dot{x}_2 = k_1 x_1 - x_2.$$

Both FDE and BDE satisfy P1, since a differential equivalence will yield an abstract model that can be exactly related to the original one. However, we observe that BDE is lossless, because every variable in the same equivalence class has the same solution. Therefore, the original model solution can be fully recovered at the expense of the side condition that equivalent variables have to be initialised equally. Instead, with FDE one cannot recover the original solutions in general; on the other hand FDE has no restrictions on the initial conditions.

When the ODE comes from a CTMC model, in [21] it is shown that FDE and BDE correspond to ordinary and exact lumpability of CTMCs [14], respectively. Incidentally, this also implies that FDE and BDE are not comparable in general. The terms "forward" and "backward" are motivated by a rather established tradition in the literature to call these two notions of CTMC lumpability (e.g., [70,36,19]), due to the fact that they involve conditions on the outgoing and incoming arcs of the CTMC state transition diagram, respectively.

Differential equivalence can be in principle defined for any ODE system. However, in order to satisfy P2 and obtain a minimisation algorithm, it is necessary to impose some restrictions on the kind of admissible ODE systems. In this chapter we review two alternatives that trade off expressiveness for scalability.

Symbolic minimisation algorithms. The first approach, presented in [21], interprets each ODE variable directly as a real function. Establishing an equivalence between two variables thus amounts to relating two functions for all their possible assignments, which involves reasoning over uncountable state spaces. The first step in [21] is to encode the equivalence conditions into logical formulae containing ODE variables, and check them *symbolically* through a satisfiability modulo theories (SMT) solver [4]. Actually, it turns out that differential equivalences can be encoded into the quantifier-free fragment of first-order logic. By appropriately restricting the admissible ODE systems to those for which an SMT solver — in our implementation, the well-known Z3 [26] — is a decision procedure for such formulae, we obtain a rigorous way of checking the existence of a differential equivalence. The language IDOL (Intermediate Drift-oriented Language) of [21] does so by essentially excluding trigonometric functions. On the other hand, it can encode polynomials of any degree, rational expressions, minima and maxima, enough to cover affine systems, chemical reaction networks with frequently used kinetics such as the law of mass action and Hill's, and the deterministic semantics of process algebra. Thus, it can satisfy P3 to some extent.

The SMT checks can be embedded into an algorithm that finds the coarsest refinement of a given input partition up to a differential equivalence. This exploits the ability of the SMT solver to produce a *witness*, i.e., a variable as-

signment that falsifies the hypothesis that the current partition is a differential equivalence. The partition is then refined iteratively until a fixed point is found.

We note that the algorithm meets the requirement of property P4. Indeed, suppose that the modeller wishes to keep track of an ODE variable x in the abstract model. Then, starting the FDE algorithm with the trivial partition where all variables are in the same block might clearly lead to an equivalence where x is related to other variables. As a result, x's individual solution cannot be recovered. However, since the input partition may be chosen arbitrarily, it is possible to isolate the desired observable variables into singleton initial blocks. Similarly, to be able to fully reconstruct the original model from the abstract one when using BDE, it is necessary to construct an initial partition *consistent* with the initial conditions of the original model (that is, two variables are in the same initial block if their initial conditions are the same).

Syntax-driven minimisation. The second approach takes a different perspective that offers a trade off between expressiveness of the language and efficiency of the minimisation algorithm. It is based on a finitary representation of an ODE system by means of a so-called *reaction network* (RN) [20]. This is a slight extension of a formal chemical reaction network (CRN) which allows rate parameters to be also negative. Assuming *elementary* reactions only, i.e., reactions with at most two reagents, a reaction network gives rise to an ODE system with derivatives that are multivariate polynomials of degree at most two. The advantage in using this construction is that it is possible to use bisimulation-style equivalences for model reduction, originally developed in [19] for CRNs, over a state space that is discrete because it only concerns finitely many "species" (corresponding to the ODE variables) and reactions (each representing a monomial in the ODE's right-hand side, as discussed in Section 2.3).

The notions of bisimulation for RNs are closely related to the differential equivalences in [21]. In particular, *forward bisimulation* (FB) is a partition of an RN's set of species which represents a sufficient condition for an FDE of the corresponding ODE variables. Instead, *backward bisimulation* (BB) fully characterises BDE (for multivariate polynomials of degree at most two). The main contribution of [20] is to exploit the fact that FB and BB can be written in the Larsen-Skou style of probabilistic bisimulation [55]. This enables us to cast the computation of the largest FB/BB into Paige and Tarjan's famous coarsest refinement problem [63]. In particular, in [20] a partition refinement algorithm is developed along the lines of efficient analogues for Markov chain lumping such as [31] and [77], and for probabilistic transition systems [3].

Tool support. Both families of symbolic and syntactic minimisation techniques are tool supported. The former has been implemented in ERODE, a tool offering SMT-based automatic Exact Reduction of Ordinary Differential Equations. The tool is available at `http://sysma.imtlucca.it/tools/erode/`, together with installation and usage instructions. ERODE is a Java tool which interacts with Z3 to perform automatic minimisation of IDOL programs up to FDE and BDE. More details on the implemented procedures are provided in Sections 2.1

and 2.2. ERODE currently supports the continuous-state semantics based on the law of mass action of CRNs given in the .net format generated with the well-established tool BioNetGen [8], version 2.2.5-stable. This allowed us to validate our differential equivalences against a wide set of existing models in the literature. Support for the entire IDOL language in under development.

The syntactic minimisation techniques have been implemented in CRNReducer, a Java tool offering automatic exact reduction of (chemical) reaction networks. It is available at `http://sysma.imtlucca.it/tools/crnreducer/`. CRNReducer performs the syntactic checks necessary to minimize an input RN up to forward and backward bisimulations. More details on the implemented algorithms are given in Sections 2.3 and 2.4. CRNReducer currently supports CRNs given in the BioNetGen's .net format, and CTMCs in the .tra/.lab format of the state-of-the-art model checker MRMC [50]. In addition, it accepts a compact CSV-like representation of linear systems of equations in the form $A \cdot x = b$, where x is the vector of unknowns. Stationary iterative methods such as Jacobi's can be seen as discrete-time dynamical system that converges to the solution. To such a system, CRNReducer can apply FB/BB (see [20] for details and benchmarks). Support for other languages which can be encoded as reaction networks is currently under development.

As part of a larger effort, a new tool collecting both symbolic and syntactic minimization techniques is currently under development. The tool will be provided with a modern integrated development environment, will offer full support for the IDOL language, and will be equipped with importing capabilities from a number formats.

Paper structure. The paper is organized as follows. Section 2 presents our symbolic (Sections 2.1 and 2.2) and syntactic (Sections 2.3 and 2.4) reduction techniques. Then, Section 3 shows how they can be applied to crowd dynamics models (Section 3.1), to multi-community epidemiology models (Section 3.2), as well as to models from the realm of evolutionary biology (Section 3.3) and biochemistry (Section 3.4). Finally, Section 4 discusses related works, while Section 5 concludes the paper.

2 Background

2.1 Differential Equivalences

Although differential equivalences can be in principle defined for a larger class of ODE models, here we consider a fragment, identified by a formal kernel language called Intermediate Drift-oriented Language (IDOL), which guarantees decidability for the problem of computing a differential equivalence.

Definition 1 (IDOL syntax). *The syntax of programs of the* intermediate drift oriented language *(IDOL) is given by*

$$p ::= \varepsilon \mid \dot{x}_i = f,\ p$$
$$f ::= n \mid x_i \mid f + f \mid f \cdot f \mid f^{\frac{1}{m}}$$

where $x_i \in \mathbb{V}$ *and* $n, m \in \mathbb{Z}$ *and* $m \neq 0$.

The set $\mathbb{V}$ represents ODE variables. A program is a list of elements $\dot{x}_i = f$ where each element gives the drift f for ODE of the variable x_i. The "dot" operator indicates the derivative with respect to time. Given an IDOL program p, we define $\mathcal{V}_p = \{x_1, \ldots, x_n\}$ as the set of variables in p. We say that p is *well-formed* if for every $x_i \in \mathcal{V}_p$ there exists a unique term $\dot{x}_i = f$ in p. We denote its drift by f_i. Throughout this paper we will consider well-formed programs only.

We remark that IDOL can cover frequently used dynamics such as:

- the law of mass action for CRNs, using drifts such as $x_1 \cdot x_2$;
- the Hill kinetics for CRNs, with drifts such as $x_1^2/(1 + x_1^2)$;
- and the *minimum* function for threshold based drifts, where

$$\min(x_1, x_2) := \frac{1}{2}(x_1 + x_2 - |x_1 - x_2|), \quad \text{with } |x| := (x \cdot x)^{\frac{1}{2}}.$$

The semantics of IDOL is given denotationally through the ODE solution of an initial value problem, starting from an initial condition $\hat{\sigma}$. For an IDOL program p, we denote by $\Theta(p)$ the logical formula that encodes the appropriate domain where the solution lives (which must be regular enough, cf. [21] for details). Furthermore, we slightly ease notation with respect to [21] by representing the solution for variable x_i simply by $x_i(t)$.

FDE is a partition over IDOL variables satisfying the property that sums of variables can be factored out from the cumulative derivatives that sum across the drifts of all variables belonging to each block, e.g. (2). This property can be captured by replacing each variable as a scaled sum of the corresponding variables of its block, such that all scaling factors are non-negative and sum to one; in the example (1), we would keep x_1 as is (it is a singleton block), and replace x_2 with $s_1 \cdot (x_2 + x_3)$ and x_3 with $s_2 \cdot (x_2 + x_3)$, where s_1 and s_2 are the scaling factors. Then, FDE amounts to proving that the aggregated drifts do not depend on the assignments of the scaling factors. For instance, in (1) we would rewrite the aggregated drift $f_2 + f_3$ as follows

$$\begin{aligned} f_2 + f_3 &= k_1 \cdot x_1 - x_2 + k_2 \cdot x_1 - x_3 \\ &= k_1 \cdot x_1 - s_1 \cdot (x_2 + x_3) + k_2 \cdot x_1 - s_2 \cdot (x_2 + x_3) \\ &= (k_1 + k_2) \cdot x_1 - (s_1 + s_2) \cdot (x_2 + x_3) \\ &= (k_1 + k_2) \cdot x_1 - (x_2 + x_3) \end{aligned}$$

Indeed it does not depend on the choice of s_1 and s_2, since $s_1 + s_2 = 1$.

We now appeal to a fundamental result from [72], which shows that it is enough to check this for a particular choice. For technical reasons discussed in [21], FDE checks this through a *uniform* scaling (for instance $s_1 = s_2 = 1/2$ in the example).

Definition 2 (FDE). *Let p be an IDOL program and $\mathcal{Z}$ a partition of $\mathcal{V}_p$. Then, $\mathcal{Z}$ is a* forward differential equivalence *if the following formula is valid:*

$$\Theta(p) \to \bigwedge_{H \in \mathcal{Z}} \Big(\sum_{x_i \in H} f_i = \sum_{x_i \in H} f_i \Big[x_j \Big/ \frac{\sum_{x_k \in H'} x_k}{|H'|} : H' \in \mathcal{Z},\ x_j \in H' \Big] \Big) \quad (\mathbf{\Phi}^{\mathcal{Z}})$$

As usual, we have denoted by $\psi[t/s]$ the term where each occurrence of t in ψ is replaced by s.

Definition 3 (FDE Quotient). *Let p be an IDOL program and $\mathcal{Z}$ an FDE partition. Then, the* forward quotient *of p with respect to $\mathcal{Z}$, denoted by $\overrightarrow{p}_{\mathcal{Z}}$, is:*

$$\dot{y}_H = \sum_{x_i \in H} f_i\Big[x_j \Big/ \frac{y_{H'}}{|H'|} : H' \in \mathcal{Z},\ x_j \in H'\Big], \quad \text{for all } H \in \mathcal{Z}.$$

We now state a crucial *dynamical characterization* theorem: A partition of IDOL variables is FDE if and only if the ODEs of the quotient program preserve the sums of the original trajectories in each equivalence class. Hence the largest FDE represents the best possible aggregation that can be obtained in this sense.

Theorem 1. *Let p be an IDOL program with initial condition $\hat{\sigma}$, $\mathcal{Z}$ a partition of $\mathcal{V}_p$. Then, $\mathcal{Z}$ is an FDE partition with forward quotient $\overrightarrow{p}_{\mathcal{Z}}$ if and only if*

$$y_H(t) = \sum_{x_i \in H} x_i(t)$$

for all t for which the solutions exist and for an initial condition of the quotient program $\hat{\sigma}_{\mathcal{Z}}$ that satisfies $\hat{\sigma}_{\mathcal{Z}}(y_H) = \sum_{x_i \in H} \hat{\sigma}(x_i)$ for all $H \in \mathcal{Z}$.

One extra step is needed to make FDE usable in a minimisation algorithm. We need to be able to refer FDE to properties enjoyed by the single variables, as opposed to blocks of variables in the original definition. If a candidate partition is not FDE, the algorithm needs to "split" the partition blocks in such a way that it isolates such variables that prevent the partition from being an FDE. For this we consider an alternative characterisation of FDE in terms of *binary* conditions.

Theorem 2 (Binary FDE characterization). *Let p be an IDOL program, $\mathcal{R}$ be an equivalence relation on $\mathcal{V}_p$, and $\mathcal{Z} = \mathcal{V}_p/\mathcal{R}$. Then $\mathcal{Z}$ is an FDE if and only if for all distinct $x_i, x_j \in \mathcal{V}_p$ we have that $(x_i, x_j) \in \mathcal{R}$ implies that the following formula is valid:*

$$\Theta(p) \to \bigwedge_{H \in \mathcal{Z}} \Big(\sum_{x_k \in H} f_k = \sum_{x_k \in H} f_k\big[x_i/s{\cdot}(x_i + x_j), x_j/(1-s){\cdot}(x_i + x_j)\big]\Big) \quad (\mathbf{\Phi}^{\mathcal{Z}}_{\mathbf{x_i},\mathbf{x_j}})$$

We now turn to BDE. The fact that IDOL variables have the same solutions at all time points is characterized by the property that variables with the same assignment are mapped to equal drifts.

Definition 4 (BDE). *Let p be an IDOL program and $\mathcal{Z}$ a partition of $\mathcal{V}_p$. Then $\mathcal{Z}$ is a* backward differential equivalence *if the following formula is valid:*

$$\Theta(p) \to \Big(\bigwedge_{H \in \mathcal{Z}} (x_{H,1} = \ldots = x_{H,|H|}) \to \bigwedge_{H \in \mathcal{Z}} (f_{H,1} = \ldots = f_{H,|H|})\Big) \qquad (\mathbf{\Psi}^{\mathcal{Z}})$$

Now, similarly to FDE it is possible to define a notion of quotient IDOL program, and state the dynamical characterization theorem.

Definition 5 (BDE Quotient). *Let p be an IDOL program and $\mathcal{Z}$ a BDE partition of $\mathcal{V}_p$. The* backward quotient *of p with respect to $\mathcal{Z}$, denoted by $\overleftarrow{p_{\mathcal{Z}}}$, is given by*

$$\dot{y}_H = f_{H,1}\big[x_{H',1}/y_{H'}, \ldots, x_{H',|H'|}/y_{H'} : H' \in \mathcal{Z}\big], \text{ for } H \in \mathcal{Z}.$$

Theorem 3 (Dynamical BDE Characterization). *Let p be an IDOL program and $\mathcal{Z}$ a partition of $\mathcal{V}_p$. Then, $\mathcal{Z}$ is a BDE partition with backward quotient $\overleftarrow{p_{\mathcal{Z}}}$ if and only if $\hat{\sigma}_{\mathcal{Z}}(y_H) = \hat{\sigma}(x_{H,1}) = \ldots = \hat{\sigma}(x_{H,|H|})$ for all $H \in \mathcal{Z}$ implies*

$$y_H(t) = x_{H,1}(t) = \ldots = x_{H,|H|}(t)$$

for all $H \in \mathcal{Z}$ and all t for which the solutions exist.

2.2 Symbolic Minimisation

The first step toward a symbolic minimisation algorithm is to be able to check whether a candidate partition is a differential equivalence. The problem amounts to establishing the validity of the (quantifier-free) formulae $\Phi^{\mathcal{Z}}$, $\Phi^{\mathcal{Z}}_{x_i,x_j}$ and $\Psi^{\mathcal{Z}}$, which are decidable by Tarski's famous result. To check them, we encode the problem into the unsatisfiability of their negations, i.e., by computing $\mathsf{sat}(\neg\Phi^{\mathcal{Z}})$, $\mathsf{sat}(\neg\Phi^{\mathcal{Z}}_{x_i,x_j})$, and $\mathsf{sat}(\neg\Psi^{\mathcal{Z}})$. These can be decided using the decision procedure nlsat [49], which is implemented in Z3 v4.0 [26]. Thus, a partition $\mathcal{Z}$ is FDE (resp., BDE) if and only if $\mathsf{sat}(\neg\Phi^{\mathcal{Z}})$ (resp., $\mathsf{sat}(\neg\Psi^{\mathcal{Z}})$) returns "unsatisfiable".

Example 1. Consider the ODE system given in Equation (1), the partition of its species $\mathcal{Z}_1 = \{\{x_1\}, \{x_2, x_3\}\}$, and $\neg\Psi^{\mathcal{Z}_1}$, i.e., the formula to check if $\mathcal{Z}_1$ is a BDE. Listing 1 provides the encoding of $\neg\Psi^{\mathcal{Z}_1}$ in the standard SMT-LIB v2.0 [5]. Given that Equation (1) is parametric with respect to two real variables k_1 and k_2, we declare them in Lines 2-3. We consider two cases: either $k_1 = k_2 = 1$ (Line 6), or $k_1 = 1$ and $k_2 = 2$ (commented out in Line 7). We have three ODE variables: x_1, x_2 and x_3, declared as real variables in Lines 10-12, paired with the three corresponding drifts f_1, f_2 and f_3, defined as functions in Lines 20-28. The three functions implicitly take k_1, k_2 and the three ODE variables as arguments, and evaluate in a real number. In this example we assume that the domain Θ of interest is $\mathbb{R}^3_{\geq 0}$, as encoded in Lines 15-17. After having specified the ODE system of interest, in Lines 31-32 we can provide the actual encoding of $\neg\Psi^{\mathcal{Z}_1}$. By applying simple transformations we can rewrite $\neg\Psi^{\mathcal{Z}_1} \equiv \neg((x_2 = x_3) \implies (f_2 = f_3))$ as $(x_2 = x_3) \wedge (f_2 \neq f_3)$. The first conjunct $(x_2 = x_3)$ imposes that the ODE variables are *constant on* $\mathcal{Z}_1$ (i.e., the ODE variables in the same block have same value). Instead, the second conjunct $(f_2 \neq f_3)$ imposes that the drifts are not constant on $\mathcal{Z}_1$. Note that the given SMT-encoding has 3 free variables: x_1, x_2 and x_3. If there exists an assignment for them that satisfies Listing 1, then $\mathcal{Z}_1$ is not a BDE. The command to check the satisfiability is given in Line 35, while Line 36 asks the solver to return one of the satisfying assignments (if any).

```
;Declare a real constant per parameter k1 and k2
(declare-const k1 Real)
(declare-const k2 Real)

;We consider k1 = 1, and either k2 = 1 or k2 = 2
(assert (= k1 1)) (assert (= k2 1))
;(assert (= k1 1)) (assert (= k2 2))

;Declare a real constant per ODE variable
(declare-const x1 Real)
(declare-const x2 Real)
(declare-const x3 Real)

;We assume to have R^3_{>=0} as domain Θ.
(assert (>= x1 0))
(assert (>= x2 0))
(assert (>= x3 0))

;Define the drift f_i of each ODE variable x_i.
(define-const f1 Real
  (* -1 x1)
)
(define-const f2 Real
  (+ (* k1 x1) (* -1 x2))
)
(define-const f3 Real
  (+ (* k2 x1) (* -1 x3))
)

;We encode ¬Ψ^{Z_1} in the equivalent form (x_2 = x_3) ∧ (f_2 ≠ f_3)
(assert (= x2 x3))
(assert (not (= f2 f3)))

;Check if the formula is satisfiable, and return a witness if so
(check-sat)
(get-model)
```

Listing 1. SMT-LIB v2.0 encoding of $\neg\Psi^{\mathcal{Z}_1}$ to check that $\mathcal{Z}_1$ is a BDE for (1)

Listing 1 can be solved using any of the SMT solvers supporting the SMT-LIB v2.0 standard. The executable Z3 encoding of Listing 1 for both the cases $k_1 = k_2$ and $k_1 \neq k_2$ is available via the *rise4fun* web interface at `http://rise4fun.com/Z3/lW7d1`. For the case $k_1 = k_2$ we obtain "unsatisfiable", because $\mathcal{Z}_1$ is a BDE partition if $k_1 = k_2$, as discussed. Instead, for the case $k_1 \neq k_2$ we obtain "satisfiable", and the assignment $\sigma_w = \{x_1 = 1, x_2 = 0, x_3 = 0\}$. In fact, we have $[\![f_2]\!](\sigma_w) = 1$ and $[\![f_3]\!](\sigma_w) = 2$, where by $[\![f]\!](\sigma)$ we have denoted the interpretation of f as a real function, evaluated with the assignment σ.

The steps $\mathsf{sat}(\Phi^{\mathcal{Z}}_{x_i,x_j})$ and $\mathsf{sat}(\Psi^{\mathcal{Z}})$ can be embedded into an algorithm that computes the coarsest FDE/BDE refinement of a given input partition, shown in Algorithm 1, and parametrised by the differential equivalence of interest (by setting $\chi = F$ and $\chi = B$ for FDE and BDE, respectively).

The refinement step for FDE (Algorithm 2) exploits its binary characterization, relating two variables whenever they do not prevent the current partition from being an FDE.

Algorithm 1 Construction of the largest FDE and BDE.

Require: Program p, partition $\mathcal{G}$ of $\mathcal{V}_p$ and $\chi \in \{F, B\}$.
  $\mathcal{Z} \leftarrow \mathcal{G}$
  while true do
    $\mathcal{Z}' \leftarrow \mathit{refine}_\chi(\mathcal{Z})$
    if $\mathcal{Z}' = \mathcal{Z}$ **then**
      return $\mathcal{Z}$
    else
      $\mathcal{Z} \leftarrow \mathcal{Z}'$
    end if
  end while

Algorithm 2 Routine refine_F

Require: Program p and a partition $\mathcal{Z}$ of $\mathcal{V}_p$.
  $\mathcal{Z}' \leftarrow \emptyset$
  for all $H \in \mathcal{Z}$ **do**
    $\mathcal{R} \leftarrow \{(x_i, x_j) : x_i, x_j \in H \text{ and } (x_i = x_j \text{ or } \Phi^{\mathcal{Z}}_{x_i,x_j} \text{is valid})\}$
    $\mathcal{Z}' \leftarrow \mathcal{Z}' \cup (H/\mathcal{R})$
  end for
  return $\mathcal{Z}'$

Algorithm 3 Routine refine_B

Require: Program p and a partition $\mathcal{Z}$ of $\mathcal{V}_p$.
  if $\Psi^{\mathcal{Z}}$ is valid **then**
    $\mathcal{Z}' \leftarrow \mathcal{Z}$
  else
    $\sigma_w \leftarrow \mathit{getWitness}(\mathsf{sat}(\neg\Psi^{\mathcal{Z}}))$
    $\mathcal{Z}' \leftarrow \emptyset$
    for all $H \in \mathcal{Z}$ **do**
      $\mathcal{R} \leftarrow \{(x_i, x_j) : x_i, x_j \in H \text{ and } [\![f_i]\!](\sigma_w) = [\![f_j]\!](\sigma_w)\}$
      $\mathcal{Z}' \leftarrow \mathcal{Z}' \cup (H/\mathcal{R})$
    end for
  end if
  return $\mathcal{Z}'$

The refinement step for BDE (Algorithm 3) exploits the fact that, when the current partition is not a BDE, i.e., $\neg\Psi^{\mathcal{Z}}$ is satisfiable, then the SMT solver can produce a witness assignment, σ_w, as shown in Example 1. This can be interpreted as a counterexample with respect to BDE, since it provides evidence that an equal assignment of variables within the same block of the candidate partition gives different values of the corresponding drifts, denoted by $[\![f_i]\!](\sigma_w)$ and $[\![f_j]\!](\sigma_w)$ in the algorithm. The idea of the refinement is to preserve variables in the same block whenever the corresponding drifts are not distinguished by the witness assignment.

Example 2. Let us consider the following IDOL program:

$$\begin{aligned}\dot{x}_1 &= -\min(x_1, x_3) + x_2\\ \dot{x}_2 &= -\min(x_2, x_3) + x_1\\ \dot{x}_3 &= -\min(x_1, x_3) - \min(x_2, x_3)\end{aligned}$$

We show that $\{\{x_1, x_2\}, \{x_3\}\}$ is the coarsest BDE that refines the initial partition $\mathcal{Z} = \{\{x_1, x_2, x_3\}\}$. Indeed, by applying the partition refinement algorithm, at the first iteration the formula $\Psi^{\mathcal{Z}}$ reads

$$\begin{aligned}&x_1 = x_2 = x_3 \rightarrow\\ &\quad -\min(x_1, x_3) + x_2 = -\min(x_2, x_3) + x_1 = -\min(x_1, x_3) - \min(x_2, x_3)\end{aligned}$$

where we have omitted the encoding of the domain $\Theta(p)$ since we assume the whole of $\mathbb{R}^3$. Its negation $\neg\Psi^{\mathcal{Z}}$ is satisfiable. Indeed, a witness assignment is $\sigma_w = \{x_1 = 1, x_2 = 1, x_3 = 1\}$, which yields a drift evaluation $[\![f_1]\!](\sigma_w) = [\![f_2]\!](\sigma_w) = 0$, and $[\![f_3]\!](\sigma_w) = -2$. This triggers a new iteration with a refined partition that preserves variables whenever their corresponding drifts evaluated for the witness are equal. In this case, we obtain the partition $\mathcal{Z}' = \{\{x_1, x_2\}, \{x_3\}\}$. Then, at the next iteration $\Psi^{\mathcal{Z}'}$ reads

$$x_1 = x_2 \rightarrow -\min(x_1, x_3) + x_2 = -\min(x_2, x_3) + x_1$$

Now, its negation is unsatisfiable, thus terminating the algorithm.

2.3 Reaction Networks

An RN (S, R) is a pair of a finite set of *species* S and a finite set of *reactions* R. A reaction is a triple written in the form $\rho \xrightarrow{k} \pi$, where ρ and π are multisets of species, called *reactants* and *products*, respectively, and $k \neq 0$ is the *reaction rate*. We restrict to *elementary* reactions where $|\rho| \leq 2$ (while no restriction is posed on the products). We denote by $\rho(X)$ the multiplicity of species X in the multiset ρ, and by $\mathcal{MS}(S)$ the set of finite multisets of species in S. The operator $+$ denotes multiset union, e.g., $X+Y+Y$ (or just $X+2Y$) is the multiset $\{\!|X, Y, Y|\!\}$. We also use X to denote either the species X or the singleton $\{\!|X|\!\}$.

The semantics of an RN (S, R) is given by the ODE system $\dot{V} = f(V)$, with $f : \mathbb{R}^S \rightarrow \mathbb{R}^S$, where each component f_X, with $X \in S$ is defined as:

$$f_X(V) := \sum_{\rho \xrightarrow{\alpha} \pi \in R} (\pi(X) - \rho(X)) \cdot \alpha \cdot \prod_{Y \in S} V_Y^{\rho(Y)} \ .$$

This ODE satisfies a unique solution $V(t) = (V_X(t))_{X \in S}$ for any initial condition $V(0)$. The restriction to elementary reactions ensures that the monomials are of degree at most 2. A standard CRN with mass-action semantics (where reaction speeds are proportional to the product of the concentrations of the reactants) is recovered by restricting to positive reaction rates and non-negative initial conditions. Instead, an arbitrary ODE system with multivariate polynomials can be encoded according to the following.

Lemma 1. *Consider the ODE system $\dot{y} = G(y)$ with components*

$$\dot{y}_k = G_k(y) := \sum_{1 \le i,j \le n} \alpha_{i,j}^{(k)} \cdot y_i \cdot y_j + \sum_{1 \le i \le n} \alpha_i^{(k)} \cdot y_i + \beta^{(k)}, \quad 1 \le k \le n, \tag{3}$$

and with $\alpha_{i,j}^{(k)}, \alpha_i^{(k)}, \beta^{(k)} \in \mathbb{R}$. Then, then RN (S_G, R_G), with $S_G := \{1, \ldots, n\}$ and

$$R_G := \left\{ i + j \xrightarrow{\alpha_{i,j}^{(k)}} i + j + k \mid \alpha_{i,j}^{(k)} \neq 0 \right\} \cup \left\{ i \xrightarrow{\alpha_i^{(k)}} i + k \mid \alpha_i^{(k)} \neq 0 \right\} \cup \left\{ \emptyset \xrightarrow{\beta^{(k)}} k \mid \beta^{(k)} \neq 0 \right\},$$

has ODEs $\dot{V}_k = G_k(V)$, for $1 \le k \le n$.

This encoding gives one reaction for each monomial in the ODE.

FB and BB are relations over the species of an RN defined only through properties that concern the reactions in which they are involved. Thus we say that they are *syntax-based* in that the ODE system is never analysed directly, in contrast to the symbolic checks performed with IDOL. FB is a sufficient condition for FDE, defined in terms of *reaction* and *production* rates.

Definition 6 (Reaction and Production rates). *Let (S, R) be an RN, $X, Y \in S$, and $\rho \in S \cup \{\emptyset\}$. The* ρ-reaction rate *of X, and the* ρ-production rate *of Y-elements by X are defined respectively as*

$$\mathbf{crr}[X, \rho] := (\rho(X) + 1) \sum_{X + \rho \xrightarrow{k} \pi \in R} k, \quad \mathbf{pr}(X, Y, \rho) := (\rho(X) + 1) \sum_{X + \rho \xrightarrow{k} \pi \in R} k \cdot \pi(Y)$$

Finally, for $H \subseteq S$ we define $\mathbf{pr}[X, H, \rho] := \sum_{Y \in H} \mathbf{pr}(X, Y, \rho)$.

Definition 7. *Let (S, R) be an RN, $\mathcal{R}$ an equivalence relation over S and $\mathcal{Z} = S/\mathcal{R}$. Then, $\mathcal{R}$ is a forward RN bisimulation (FB) if for all $(X, Y) \in \mathcal{R}$, all $\rho \in S \cup \{\emptyset\}$, and all $H \in \mathcal{Z}$ it holds that*

$$\mathbf{crr}[X, \rho] = \mathbf{crr}[Y, \rho] \quad \textit{and} \quad \mathbf{pr}[X, H, \rho] = \mathbf{pr}[Y, H, \rho] \tag{4}$$

This definition, originally proposed in [19] for chemical reaction networks, carries over to RNs. An important observation that is instrumental for the development of an efficient partition refinement algorithm is that, as discussed, FB is in the Larsen-Skou style of probabilistic bisimulation, whereby species are related with respect to their aggregate behaviour toward the equivalence classes, parametrised by a further object ρ which plays a role akin to "action labels" in probabilistic transition systems.

Example 3. Consider the RN with species $\{X_1, X_2, X_3, X_4, X_5\}$ and reactions

$$\begin{aligned} X_1 &\xrightarrow{1} X_2 & X_1 + X_3 &\xrightarrow{3} X_4 \\ X_2 &\xrightarrow{1} X_1 & X_2 + X_3 &\xrightarrow{3} X_5 \\ X_4 &\xrightarrow{1} X_1 + X_3 & X_5 &\xrightarrow{1} X_2 + X_3 \\ X_4 + X_3 &\xrightarrow{3} X_5 & X_5 + X_3 &\xrightarrow{3} X_4 \end{aligned}$$

Then, it holds that $\{\{X_1, X_2\}, \{X_3\}, \{X_4, X_5\}\}$ is an FB. For instance, we have

$$\begin{aligned} \mathbf{crr}[X_1, \emptyset] = \mathbf{crr}[X_2, \emptyset] = 1 \\ \mathbf{crr}[X_4, X_3] = \mathbf{crr}[X_2, X_3] = 3 \end{aligned}$$

As regards **pr**, we have

$$\begin{aligned} \mathbf{pr}[X_1, \emptyset, \{X_1, X_2\}] = \mathbf{pr}[X_1, \emptyset, \{X_1, X_2\}] = 1 \\ \mathbf{pr}[X_4, \emptyset, \{X_3\}] = \mathbf{pr}[X_5, \emptyset, \{X_3\}] = 1 \end{aligned}$$

We now provide a version of BB developed in [20] in the same style.

Definition 8 (Cumulative splitter flux rate). *Let (S, R) be an RN, $X, Y \in S$, $\mathcal{Z}$ a partition of S, $H \in \mathcal{Z}$ and $H' \in \mathcal{Z} \cup \{\{\emptyset\}\}$. We define*

$$\mathbf{sr}(X, Y, H') := \sum_{\rho' \in H'} \sum_{\substack{\rho \xrightarrow{\alpha} \pi \in R \\ \rho = Y + \rho'}} (\pi(X) - \rho(X)) \cdot \alpha', \quad \mathbf{sr}[X, H, H'] := \sum_{Y \in H} \mathbf{sr}(X, Y, H').$$

with $\alpha' = \frac{\alpha}{2}$ if $Y \neq \rho'$ and $Y \in H'$, or $\alpha' = \alpha$ otherwise. We call the quantity $\mathbf{sr}[X, H, H']$ the cumulative (H, H')-splitter flux rate *of X.*

Note that we account for summands that are counted twice due to the two summations over H' in $\mathbf{sr}[X, H', H']$ by choosing $\alpha' \in \{\alpha, \frac{\alpha}{2}\}$ in the above definition.

Theorem 4. *Let (S, R) be an RN, $\mathcal{R}$ an equivalence relation over S and $\mathcal{Z} = S/\mathcal{R}$. Then $\mathcal{R}$ is a BB if and only if for all $(X, Y) \in \mathcal{R}$, all $H \in \mathcal{Z}$ and all $H' \in \mathcal{Z} \cup \{\{\emptyset\}\}$ it holds that $\mathbf{sr}[X, H, H'] = \mathbf{sr}[Y, H, H']$.*

Example 4. The partition $\{\{X_1, X_2\}, \{X_3\}, \{X_4, X_5\}\}$ of Example 3 is also a BB. For instance, due to the reactions $X_1 \xrightarrow{1} X_2$ and $X_2 \xrightarrow{1} X_1$ we have

$$\mathbf{sr}[X_1, \{X_1, X_2\}, \emptyset] = \mathbf{sr}(X_1, X_1, \emptyset) + \mathbf{sr}(X_1, X_2, \emptyset) = -1 + 1 = 0$$

Similarly, we have

$$\mathbf{sr}[X_2, \{X_1, X_2\}, \emptyset] = \mathbf{sr}(X_2, X_1, \emptyset) + \mathbf{sr}(X_2, X_2, \emptyset) = 1 - 1 = 0$$

2.4 Partition-refinement algorithms for RNs

The minimisation algorithms for FB and BB are partition-refinement algorithms based on Paige and Tarjan's approach, iteratively refining an input partition based on a *splitter* block that tells apart the behaviour of two species toward that block. We omit the technical details of the minimisation algorithm, which can be found in [20]. Here we remark that the coarsest FB and BB partitions of an arbitrary polynomial ODE system can be computed in $\mathcal{O}(r \cdot s \cdot \log s)$ time and $\mathcal{O}(r \cdot s)$ space, where r is the number of monomials and s is the number of species. Instead, here we provide a step-by-step illustration of the algorithms on a simple example.

For FB, we first observe that the **crr**-condition of FB can be implemented as an initialization step that pre-partitions the species according to the values of **crr**. This is because **crr** is a "global" property of the RN, i.e., it does not depend on the current partition. Then, as discussed, the conditions on **pr** require the iterative partition-refinement treatment, where ρ plays the role of the label as discussed. An important property is that, at each iteration, the blocks of the current partition are used as potential splitters. This ensures that the list of splitters can be updated at no additional cost while splitting the blocks.

Example 5. Let us consider again the RN in Example 3 and compute the coarsest FB refinement of the trivial partition $\{\{X_1, X_2, X_3, X_4, X_5\}\}$. The initialization step that computes the pre-partitioning with respect to the values of **crr** leads to the refinement $\{\{X_1, X_2, X_4, X_5\}, \{X_3\}\}$. Now, both blocks $\{X_1, X_2, X_4, X_5\}$ and $\{X_3\}$ will be considered as potential splitters. The former does not cause any splitting because, for any species X_i and any label ρ, the values of $\mathbf{pr}[X_i, \{X_1, X_2, X_4, X_5\}, \rho]$ are the same. Instead, $\{X_3\}$ will split the block $\{X_1, X_2, X_4, X_5\}$ into two blocks $\{X_1, X_2\}$ and $\{X_4, X_5\}$, because, e.g., it holds

$$\mathbf{pr}[X_4, \{X_3\}, \emptyset] = \mathbf{pr}[X_5, \{X_3\}, \emptyset] = 1$$
$$\mathbf{pr}[X_1, \{X_3\}, \emptyset] = \mathbf{pr}[X_2, \{X_3\}, \emptyset] = 0$$

Since $\{X_1, X_2, X_4, X_5\}$ has already been used as a splitter, following the principle of *ignoring the largest part* [63], the sub-block with maximal size is not added to the list of potential splitters. In this case, the algorithm will add $\{X_4, X_5\}$, which remains the only splitter to be considered. Since it does not refine any of the existing blocks, the algorithm terminates with the partition $\{\{X_1, X_2\}, \{X_3\}, \{X_4, X_5\}\}$ being the coarsest FB refinement.

For BB, instead, the third argument of **sr** can be seen as a label. However, while in FB this ranges over the set of species (together with the distinguished species $\emptyset$ to indicate unary reactions), in BB it ranges over blocks of the candidate BB partition to be checked (again, together with the distinguished set $\{\emptyset\}$ for unary reactions). When used within the partition refinement algorithm, splitting a partition block leads to a refinement of the BB labels. In other words, unlike for FB the set of labels must be updated at every iteration. However, it can be shown that this incurs no additional computational cost [20].

Example 6. Let us consider once more the RN in Example 3 and compute the coarsest BB refinement of the trivial partition $\{\{X_1, X_2, X_3, X_4, X_5\}\}$. In the first iteration the block $H = \{X_1, X_2, X_3, X_4, X_5\}$ is used to split itself, computing $\mathbf{sr}[X_i, H, \emptyset]$ and $\mathbf{sr}[X_i, H, H]$ for all $i \in \{1, 2, 3, 4, 5\}$. This leads to the partition $\{\{X_1, X_2\}, \{X_3\}, \{X_4, X_5\}\}$. Similarly to the FB case, since H has already been used as a splitter, only $\{X_4, X_5\}$ and $\{X_3\}$ are added as potential splitters, while $\{X_1, X_2\}$ is ignored. The two candidate splitters do not lead to any refinement, and thus the previously computed partition is returned.

3 Case Studies

This section presents four case studies of CAS models. We begin in Section 3.1 with a crowd dynamics scenario, where the emergent behaviour of a population arises from decisions made locally by individuals. Then, in Section 3.2 we consider an epidemiological model, where the emergent phenomenon of an infection spreading is the result of individual opportunistic contacts between agents. Incidentally, these two case studies feature space and locality as first-class citizen, with increasing complexity. In the crowd dynamics model, individuals do not have an internal status, and dynamics are restricted only to movements among locations. The epidemiological model, instead, does account for individuals' internal states, affected by *local interactions* with other individuals in the same location. In both cases, we start from specifications given by co-authors of this volume in two formal languages, namely BioPEPA [22] and PALOMA [33], from which (together with PEPA [44] and SCEL [28]), originates the CARMA language described elsewhere in this volume.

Sections 3.3 and 3.4 present case studies of biological relevance. Specifically, Section 3.3 deals with adaptation in biological systems through evolution of simple structures into more complex ones that retain some of the original behaviour. This is formally captured by means of suitable differential equivalences between CRNs. Section 3.4 presents reductions of a number of CRN models of protein interaction networks presented in the literature, which are well known to the problem of ODEs with combinatorial complexity (e.g., [23]; see also Section 4 for further related work).

3.1 Crowd Dynamics

Our first case study regards a crowd scenario in which individuals move among the squares of a city according to certain policies. Our starting point is the famous "El Botellón" model [66], used to describe the spontaneous self-organization of drinking parties in the squares of Spanish cities. The model considers four squares connected in a ring by streets. The movements of a single individual are dictated by a simple rule: if no *friend* (or partner to talk to) can be found in its current square, the individual randomly moves to one of the two connected squares. The model assumes that an individual in square i moves with probability $(1 - c)^{s_i - 1}$, where s_i is the number of people currently in square i, and c

is the *chat probability*, i.e. the probability that an individual finds a friend. The model has been also studied in [58] by co-authors of this volume using related analysis techniques.

More recently, a variant of El Botellón has been proposed in [12] and further analysed in [11], where the chat probability is not a constant, but it depends on two parameters: (i) The *socialisation factor* of the population (*soc*), i.e. the average number of friends of each individual; (ii) The total number of considered individuals (N). The socialisation-driven chat probability is then given by $c = soc/N$. The intuition is that people tend to have a limited number of friends, *soc*, hence the larger is the considered population, the lower is the probability of meeting a friend.

Inspired by the El Botellón model and its socialisation-based variant, we hereby propose a sort of dual scenario where individuals do not move across the squares on their own, e.g. because streets are not safe, but move only if they are able to meet a friend to share the path with. Also, we assume that movements follow a biologically-inspired dynamics: movements from a square i to a square j happen with a rate proportional to the power of the number of people in square i (s_i^2), modelling the probability of two individuals to meet, multiplied by the socialisation-driven chat probability. This is reminiscent of the already discussed law of mass action, which states that the firing rate of a chemical reaction $X_i + X_j \xrightarrow{k} Y_r + Y_l$ is proportional to the concentration of the reacting species of the reaction (X_i, X_j), times the kinetic constant k. Considering n squares, we assume to have an $n \times n$ routing matrix Q, where each $Q_{i,j}$ entry stores the probability that an individual moves from square i to square j. The evolution of the population of each square is governed by an ODE system defined as, for all $i \in \{1 \dots n\}$:

$$\dot{s}_i = -\sum_{1 \leq j \leq n} 2 \cdot c \cdot Q_{i,j} \cdot s_i^2 + \sum_{1 < j \leq n} 2 \cdot c \cdot Q_{j,i} \cdot s_j^2 \tag{5}$$

For example, the ODE system for the case of four cities ($n = 4$) is

$$\begin{aligned}
\dot{s}_1 &= 2 \cdot c \cdot \big(-\sum_{1 \leq j \leq 4} Q_{1,j} \cdot s_1^2 + \sum_{1 < j \leq 4} Q_{j,1} \cdot s_j^2 \big) \\
\dot{s}_2 &= 2 \cdot c \cdot \big(-\sum_{1 \leq j \leq 4} Q_{2,j} \cdot s_2^2 + \sum_{1 < j \leq 4} Q_{j,2} \cdot s_j^2 \big) \\
\dot{s}_3 &= 2 \cdot c \cdot \big(-\sum_{1 \leq j \leq 4} Q_{3,j} \cdot s_3^2 + \sum_{1 < j \leq 4} Q_{j,3} \cdot s_j^2 \big) \\
\dot{s}_4 &= 2 \cdot c \cdot \big(-\sum_{1 < j \leq 4} Q_{4,j} \cdot s_4^2 + \sum_{1 < j \leq 4} Q_{j,4} \cdot s_j^2 \big)
\end{aligned}$$

The same dynamics can be expressed also in terms of a reaction network defined as, for all $i, j \in \{1 \dots n\}$ such that $i \neq j$:

$$s_i + s_i \xrightarrow{Q_{i,j} \cdot c} s_j + s_j \tag{6}$$

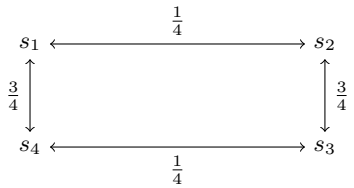

Fig. 1. Probabilities of movements among squares in the crowd dynamics model.

This reaction models the fact that two individuals in square i meet and move together to a target square j. The deterministic firing rate of the reaction is $[s_i] \cdot [s_i] \cdot Q_{i,j} \cdot c$, where $[s_i]$ is the number of individuals in square i. The term $[s_i] \cdot [s_i]$ accounts for the number of meetings [2], while c restricts to the successful ones (i.e. those among friends), and $Q_{i,j}$ for those leading to movements towards square j. For example, the reaction network for the case of four cities ($n = 4$) is

$$
\begin{array}{ll}
s_1 + s_1 \xrightarrow{Q_{1,2} \cdot c} s_2 + s_2 & \qquad s_3 + s_3 \xrightarrow{Q_{3,4} \cdot c} s_4 + s_4 \\
s_1 + s_1 \xrightarrow{Q_{1,4} \cdot c} s_4 + s_4 & \qquad s_3 + s_3 \xrightarrow{Q_{3,2} \cdot c} s_2 + s_2 \\
s_2 + s_2 \xrightarrow{Q_{2,3} \cdot c} s_3 + s_3 & \qquad s_4 + s_4 \xrightarrow{Q_{4,1} \cdot c} s_1 + s_1 \\
s_2 + s_2 \xrightarrow{Q_{2,1} \cdot c} s_1 + s_1 & \qquad s_4 + s_4 \xrightarrow{Q_{4,3} \cdot c} s_3 + s_3
\end{array}
$$

This model shows an interesting property in case Q is symmetric, i.e. $Q_{i,j} = Q_{j,i}$: independently from how the individuals are initially distributed among the squares, on the long run they will be evenly distributed. The same property is found also in the models of [66] and [12]. To show this, Figure 3.1 depicts the evolution of 200000 individuals among the squares (s_1, s_2, s_3, and s_4) with symmetric routing matrix Q defined such that $Q_{1,2} = Q_{2,1} = \frac{1}{4}$, $Q_{1,4} = Q_{4,1} = \frac{3}{4}$, $Q_{2,3} = Q_{3,2} = \frac{3}{4}$ and $Q_{3,4} = Q_{4,3} = \frac{1}{4}$, as depicted in Figure 1. The socialisation factor *soc* is set to 2. In the left plot all individuals are initially located in square s_1, while in the right plot they are evenly divided among s_1 and s_2. After some time, in both plots individuals equi-distribute in the four squares, as expected. We notice that more time is required in the case in which all individuals are initially located in the first square.

Figure 3.1 shows a further interesting property of the crowd scenario. From Figure 3.1 (right) we note that the populations in squares s_1 and s_2, as well as those in s_3 and s_4, evolve in the same way if individuals are initially evenly distributed in s_1 and s_2. Instead, such symmetries do not appear in Figure 3.1 (left). This can be proven using our backward reductions. The model has the following property:

[2] Note that $[s_i] \cdot [s_i]$ should actually be $[s_i] \cdot ([s_i] - 1)$, since an individual cannot meet itself. However, this is irrelevant for large populations, and hence, as for existing ODE-based semantics in the biological context [78], we approximate it to $[s_i] \cdot [s_i]$.

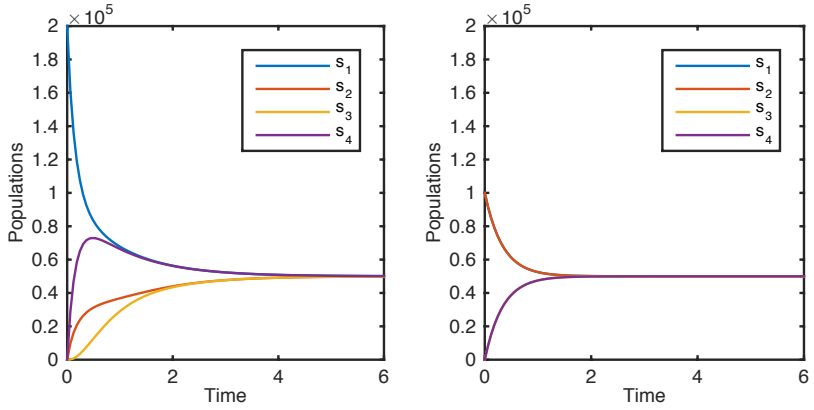

Fig. 2. Crowd scenario with 4 squares and 200000 individuals. All individuals initially in square 1 (left) or evenly divided among squares 1 and 2 (right).

> Whenever the initial number of individuals in s_1 and s_3 is equal that of s_2 and s_4, respectively, then the number of individuals in s_1 and s_3 will be equal to that of s_2 and s_4, respectively, at any point in time.

This property can be verified by reducing the model up to BDE (or equivalently up to BB) using a pre-partition coherent with the required initial conditions, i.e., $\mathcal{Z}_{BDE} = \{\{s_1, s_2\}\ ,\ \{s_3, s_4\}\}$. The algorithm returns $\mathcal{Z}_{BDE}$ itself, confirming that it is a BDE. Instead, by using an initial partition coherent with Figure 3.1 (left), i.e. $\{\{s_1\}, \{s_2, s_3, s_4\}\}$ we obtain no reduction, as expected.

The model also allows for forward reductions, even though they are less interesting. It can be shown that the only forward differential equivalence of the model is $\mathcal{Z}_{FDE} = \{\{s_1, s_2, s_3, s_4\}\}$. This one-block partition is typical of *mass-preserving systems*, i.e. where the total number of entities does not change. In fact, the corresponding FDE-reduced model is $\dot{s}_{FDE} = 0$, meaning that the cumulative population $s = s_1 + s_2 + s_3 + s_4$ is an invariant of the system. No reduction can be instead computed using FB. This is because, as discussed in [19], FB distinguishes among incoming and outgoing flow.

3.2 Multi-community Epidemiology

Our second case study is inspired from the well-known epidemiology model SIR [51], describing the spreading of an infection in a population from *infected* individuals (I), to *susceptible* individuals (S), considering the possibility of *recovering* (R) from infection after some time. We hereby consider a multi-community SIR model extended with spatial features as considered in [33]. Intuitively, individuals move among a number of *communities*, similarly to our crowd model. In addition, individuals in the same location might interact spreading the infection.

More in particular, the authors of [33] use PALOMA (the Process Algebra of Located Markovian Agents), a predecessor of the CARMA language described

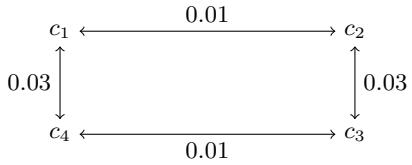

Fig. 3. Rates of movements between communities in the multi-community SIR model.

in this volume, to formalize a simplified model of the 1918-1919 flu epidemic in central Canada originally described in [68]. The model consists of m communities, with a routing matrix Q used to store the rates (rather than probabilities as in the previous crowd model) at which individuals travel between communities. While moving between communities, individuals might interact with locals, spreading the flu. Interactions might happen at different rates in each community (e.g., to distinguish among residential and business areas). Each community c is thus associated with three populations of susceptible (S_c), infected (I_c) and recovered (R_c) individuals. Upon a *contact* between an S_c and I_c individual, the former gets infected with a given probability p. In $1/\gamma$ days on average, an I_c will recover, becoming immune from the flu. The two parameters p and γ are system-dependent, while the rate of contact might change in each community.

In the rest of this section we will consider two variants of the model. In the first one we assume that contacts happen with rate 0.03 in all communities, while in the second variant we have contact rate equal to 0.03 in communities c_1 and c_2, and to 0.04 in the others. In both models we have $p = 0.5$, $\gamma = 0.2$, and, as for the crowd protocol we consider four locations (i.e., communities) connected in a ring by streets according to the symmetric routing matrix Q defined as in Figure 3. Also, we assume that each community initially has 150000 susceptible individuals, 11000 infected ones and 12000 recovered ones.

The actual PALOMA specification (up to slight changes in the parameters) can be found at `http://groups.inf.ed.ac.uk/paloma/SIR.paloma`. We refer the interested reader to [33] for more details about the considered PALOMA specification, as well as PALOMA's syntax and tool support. Thanks to the tool support of PALOMA, it is possible to generate an ODE system whose solution gives an approximation of the expected values and the variances of the three populations (S, I and R) in each of the four locations, for a total of 24 measures of interest [32]. In total, 90 ODEs are generated.

The obtained ODE system belongs to the IDOL language, allowing us to apply our symbolic reduction techniques.The coarsest BDE of the model variant with homogeneous contact rates consists of 27 blocks, 6 of which contain all and only the 24 measures of interest, while the other 21 blocks contain the additional variables. In particular, we have three blocks containing the expected values of the three populations in each community: $\{y_{E[S_{c_1}]}, y_{E[S_{c_2}]}, y_{E[S_{c_3}]}, y_{E[S_{c_4}]}\}$, $\{y_{E[I_{c_1}]}, y_{E[I_{c_2}]}, y_{E[I_{c_3}]}, y_{E[I_{c_4}]}\}$, $\{y_{E[R_{c_1}]}, y_{E[R_{c_2}]}, y_{E[R_{c_3}]}, y_{E[R_{c_4}]}\}$. This tells us that (the approximation of) each population evolves in the same way in all

communities if initialized equally. This might be expected in a sense, due to the fact that interaction rates do not depend on the community of residence. However, it is interesting to note that populations remain evenly distributed among communities despite having different inter-community transition rates. This can be explained using similar arguments to those of the crowd scenario. The other three blocks are similar, but refer to the second-order moments. Hence, not just the expected values, but also the variances of the populations evolve equally. The obtained BDE partition does not change even if starting with an initial partition coherent with the discussed initial populations. Hence, when the populations of each of S, I and R are initially evenly divided among the four communities, we have that the same information contained in the original ODE system can be recovered from one with 30% of its original size. Similarly to the crowd scenario, FDE does not produce notable reductions.

We now focus on the model variant having 0.03 as contact rate in communities c_1 and c_2, and 0.04 in c_3 and c_4. By applying BDE starting from the trivial partition with one block only, or from the one coherent with the initial populations, we obtain a partition of 48 blocks. This is actually a refinement of the BDE partition obtained from the homogeneous model variant. In particular, the 6 blocks of interest are split to separate the populations of the communities c_1 and c_2 from those of c_3 and c_4; e.g., the 3 blocks about the average populations are split in $\{y_{E[S_{c_1}]}, y_{E[S_{c_2}]}\}$,$\{y_{E[S_{c_3}]}, y_{E[S_{c_4}]}\}$, $\{y_{E[I_{c_1}]}, y_{E[I_{c_2}]}\}$, $\{y_{E[I_{c_3}]}, y_{E[I_{c_4}]}\}$, $\{y_{E[R_{c_1}]}, y_{E[R_{c_2}]}\}$, $\{y_{E[R_{c_3}]}, y_{E[R_{c_4}]}\}$. As a result, an ODE system of size of about 50% the original one can be used to study the measures of interest of the model.

3.3 Evolutionary Biology

A major subject of investigation in evolutionary biology is to understand how simple structures may evolve into more complex ones as a result of their adaptation to the environment. It has been argued, for instance, that basic cellular switches have evolved in order to increase robustness in their capacity to perform certain functionality by reducing sensitivity to noise [18].

Recently, Cardelli has proposed the notion of *emulation* as a formal way of comparing two CRN models of biological systems in order to postulate an evolutionary path between them [17]. A simpler CRN, i.e. a CRN with fewer species, is said to emulate a larger CRN if in the latter it is possible to find appropriate initial conditions such that the trajectories exactly correspond to those of the simpler CRN. Since the CRN semantics associates an ODE variable with each species, the presence of an emulation will imply that in the larger CRN two or more species' ODE trajectories will overlap, and match one of the simpler CRN as well whenever the initial conditions are equal. The intuitive interpretation given to this dynamical property is that the more complex CRN might possess richer behaviour than the simpler CRN from which it descends, but that the evolution is *conservative* in the sense that under special initial conditions it may collapse onto the original one.

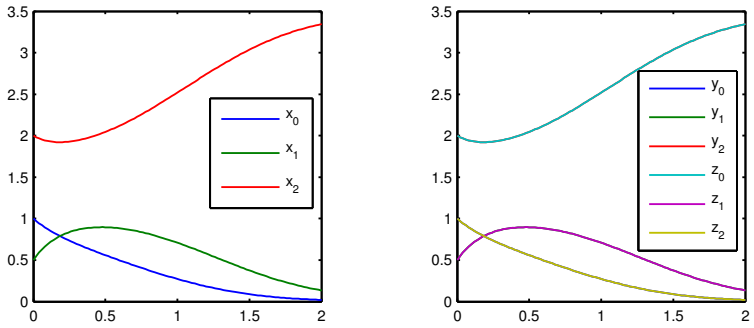

Fig. 4. ODE solutions of AM (left) and MI (right), showing equivalent trajectories with equal initial conditions.

Example 7. The following two mass-action CRNs describe the behaviour of AM, a basic biological switch (left) and MI, a mutual inhibition mechanism (right) [17]:

$$
\begin{array}{l|l}
 & Y_0 + Z_0 \xrightarrow{\alpha_1} Z_0 + Y_1 \\
 & Y_1 + Z_0 \xrightarrow{\alpha_2} Z_0 + Y_2 \\
X_0 + X_2 \xrightarrow{\alpha_1} X_2 + X_1 & Y_2 + Y_0 \xrightarrow{\alpha_3} Y_0 + Y_1 \\
X_1 + X_2 \xrightarrow{\alpha_2} X_2 + X_2 & Y_1 + Y_0 \xrightarrow{\alpha_4} Y_0 + Y_0 \\
X_2 + X_0 \xrightarrow{\alpha_3} X_0 + X_1 & Z_2 + Z_0 \xrightarrow{\alpha_1} Z_0 + Z_1 \\
X_1 + X_0 \xrightarrow{\alpha_4} X_0 + X_0 & Z_1 + Z_0 \xrightarrow{\alpha_2} Z_0 + Z_0 \\
 & Z_0 + Y_0 \xrightarrow{\alpha_3} Y_0 + Z_1 \\
 & Z_1 + Y_0 \xrightarrow{\alpha_4} Y_0 + Z_2
\end{array}
$$

Consider the following mappings:

- Trajectories of Y_0 and Z_2 correspond to that of X_0;
- Trajectories of Y_1 and Z_1 correspond to that of X_1;
- Trajectories of Y_2 and Z_0 correspond to that of X_2.

Indeed, it can be shown that if one sets equal initial conditions for related species (e.g., by setting equal initial conditions for Y_0, Z_2, and X_0) then the trajectories will coincide at all time points (see Figure 4). It is clear that emulation is closely related to BDE — and to BB since it has been considered for mass-action CRNs. In fact, it can be shown that an emulation is an appropriate BDE on the "union CRN" [21]. For instance, in the example above the BDE is given by $\mathcal{Z}_{EMU} = \{\{X_0, Y_0, Z_2\}, \{X_1, Y_1, Z_1\}, \{X_2, Y_2, Z_0\}\}$. This can be checked using the executable Z3 encoding available at `http://rise4fun.com/Z3/bgVv`, which is similar to Listing 1, but regards the nine ODEs of (the union CRN of) AM and MI. Note that $\{\{X_0\}, \{X_1\}, \{X_2\}\}$ is a BDE form AM, while $\{\{Y_0, Z_2\}, \{Y_1, Z_1\}, \{Y_2, Z_0\}\}$ is a BDE for MI.

Here we show how to exploit the expressiveness of IDOL to strengthen the idea of an evolutionary relationship between networks, by studying whether it carries over to non-mass-action kinetics as well. The possibility of reasoning using different hypotheses for the reaction kinetics is of biological relevance because in different situations one may find mass-action mechanisms (e.g., phosphotransfers) or Hill-type mechanisms (e.g., enzymes) [78]. For instance, much of the utility of Hill kinetics is owed to supporting non-integer exponents. Famously, this ranges in 2.3-3.0 for haemoglobin. Furthermore, biologists often consider exponents less than 1 in order to describe "anticooperative" behaviour [60]. Since any rational exponent can be expressed in IDOL we consider the question whether the mappings are preserved by a BDE for CRNs with Hill semantics.

We discuss an IDOL encoding of CRNs according to the Hill kinetics (e.g., [78]) in the case of *catalytic* reactions, i.e., reactions which are in the form $B + C \xrightarrow{l} D + C$ with $B \neq D$. We remark that both AM and MI are in this form. Here, C plays the role of a catalyst, a species promoting the reaction but which is not affected by it. Species B is the *substrate* that is modified, becoming D, when the reaction occurs. Each reaction is labelled with a triple $(\beta_1, \beta_2, \nu) \in \mathbb{Q}^3_{>0}$.

Definition 9 (see [21]). *A Hill CRN is a pair (S, R_S) where R_S is a finite set of catalytic reactions with $R_S \subseteq \mathbb{N}_0^S \times \mathbb{N}_0^S \times \mathbb{Q}^3_{>0}$.*

Definition 10. *The IDOL program p_S of a Hill CRN is*

$$\dot{x}_A = h_A := \sum_{\substack{\rho \xrightarrow{(\beta_1,\beta_2,\nu)} \pi \in R_S \\ \rho = B+C, \pi = D+C}} (\pi_A - \rho_A) \frac{\beta_1 x_B^\nu}{\beta_2 + x_B^\nu}, \quad \textit{for all } A \in S.$$

By replacing equal mass-action rates with equivalent Hill triplets, it can be shown that the BDE carries over. The following are the Hill CRNs obtained from AM and MI by replacing each original mass action reaction $\rho \xrightarrow{\alpha} \pi$ with the corresponding Hill reaction $\rho \xrightarrow{\alpha,\alpha,\nu} \pi$:

$$\begin{array}{l|l}
 & Y_0 + Z_0 \xrightarrow{\alpha_1,\alpha_1,\nu} Z_0 + Y_1 \\
 & Y_1 + Z_0 \xrightarrow{\alpha_2,\alpha_2,\nu} Z_0 + Y_2 \\
X_0 + X_2 \xrightarrow{\alpha_1,\alpha_1,\nu} X_2 + X_1 & Y_2 + Y_0 \xrightarrow{\alpha_3,\alpha_3,\nu} Y_0 + Y_1 \\
X_1 + X_2 \xrightarrow{\alpha_2,\alpha_2,\nu} X_2 + X_2 & Y_1 + Y_0 \xrightarrow{\alpha_4,\alpha_4,\nu} Y_0 + Y_0 \\
X_2 + X_0 \xrightarrow{\alpha_3,\alpha_3,\nu} X_0 + X_1 & Z_2 + Z_0 \xrightarrow{\alpha_1,\alpha_1,\nu} Z_0 + Z_1 \\
X_1 + X_0 \xrightarrow{\alpha_4,\alpha_4,\nu} X_0 + X_0 & Z_1 + Z_0 \xrightarrow{\alpha_2,\alpha_2,\nu} Z_0 + Z_0 \\
 & Z_0 + Y_0 \xrightarrow{\alpha_3,\alpha_3,\nu} Y_0 + Z_1 \\
 & Z_1 + Y_0 \xrightarrow{\alpha_4,\alpha_4,\nu} Y_0 + Z_2
\end{array}$$

For example, the first reaction of the Hill variant of MI introduces the terms $\frac{\alpha_1 \cdot Y_0}{\alpha_1 + Y_0}$ and $-\frac{\alpha_1 \cdot Y_0}{\alpha_1 + Y_0}$ in the drifts of Y_1 and Y_0, respectively. It can be shown that

the coarsest BDE partition of the above Hill variant of MI remains $\{\{Y_0, Z_2\}, \{Y_1, Z_1\}, \{Y_2, Z_0\}\}$. Also, $\psi_{EMU} = \{\{X_0, Y_0, Z_2\}, \{X_1, Y_1, Z_1\}, \{X_2, Y_2, Z_0\}\}$ is an emulation among the two Hill CRNs. Similarly to the mass action case, we provide an executable Z3 encoding available at `http://rise4fun.com/Z3/f90U` to confirm this.

3.4 Protein interaction networks

We hereby consider three of the biochemical networks considered in our previous work [19] (and also in [20]): a model of pheromone signalling (M1, [71]); a model of a tumour suppressor protein (M2, [6]); and a MAPK model (M3, [52]). These are three biologically meaningful chemical reaction networks taken from the literature given in .net format of the widely used BioNetGen tool [8], version 2.2.5-stable. In [19] and [20] we have proved that FB and BB can be successfully applied to these and other BioNetGen models, providing the reduction times, the size of the obtained reduced models, and the speed-up obtained by analysing them. For each RN (S, R), in the case of FB reductions we considered the trivial partition $\{S\}$ (thus yielding the largest bisimulation). Instead, for BB an initial partition coherent with the initial conditions was chosen, due to the side condition of BB: two species were put in the same initial block in case of equal initial conditions, read from the original model specification.

The BioNetGen tool allows the modeller to specify *observables* of interest, given in the form of sums of species. When solving the ODEs underlying the considered model, a plot containing a line per observable is generated, showing the evolution of the specified cumulative concentrations. Differently from [19] and [20], we now study the FB partitions obtained when using initial partitions coherent with the user-specified observables. These are partitions which guarantee that the information of interest to the modeller is preserved, hence de facto obtaining "lossless" FB reductions. We remark that BioNetGen observables might not specify a partition of the species because: (i) Some species might not appear in the observables; (ii) Others might appear in more than one observable. However, it is easy to obtain an observables-preserving initial partition as follows:

1. All species not appearing in any observable are put in a single (sink) partition block;
2. For each observable, its subset of species not appearing in any other observable is turned into a partition block;
3. The set of species appearing in more than one observable is partitioned in blocks of species appearing in (all and only) the same blocks.

As shown in column $|Prep.|$ of Table 3.4, the 14 531 species of M1 are pre-partitioned in 1 345 blocks, the 796 species of M2 are pre-partitioned in 18 blocks, while the 85 species of M3 are pre-partitioned in 4 blocks. From the table we also note that, both in terms of reduction time and size of the reduced model, the pre-partitioning does not affect the FB reduction of M2, while it affects that of the other two models only slightly.

Original model				*FB reduction*			*FB reduction with prep.*			
Id	Ref.	$\|R\|$	$\|S\|$	*Red.(s)*	$\|R\|$	$\|S\|$	$\|Prep.\|$	*Red.(s)*	$\|R\|$	$\|S\|$
M1	[71]	194 054	14 531	3.88E–1	142 165	10 855	1 345	3.28E–1	147 797	12 037
M2	[6]	5 797	796	1.90E–2	4 210	503	18	4.10E–2	4 210	503
M3	[52]	487	85	2.00E–3	264	56	4	2.00E–3	362	69

4 Related Work

FDE/FB are special cases of exact ODE lumpability [62], which concerns ODE aggregations through a linear projection of the state space. While the general theory is well-established, in particular for ODEs arising from mass-action CRNs [72], there are no algorithms for computing these projections, unlike with the partition refinement algorithms of FDE/FB. As discussed, when the ODE represents the forward equations of motion of a CTMC, both FDE and FB correspond to ordinary CTMC lumpability [14]. In addition, in that case the partition refinement algorithm of FB yields the same time and space complexity of state-of-the-art algorithms for CTMCs [31,77]. FDE/FB are also related to a recently proposed notion of equivalence called *differential bisimulation* [47]. This is developed for a fragment of Hillston's PEPA process algebra that is equipped with an ODE semantics with non-linear minimum-based drifts that approximate the average evolution of underlying CTMCs with massively parallel computations [45,43,73]. Differential bisimulation is a relation over the set of constants of a PEPA model, defined in terms of conditions on the sequential behaviour and on the compositional structure of processes. It can be shown that differential bisimulation is a special case of FDE for the ODEs induced by a PEPA process.

BDE/BB are generalisations of the notion of *label equivalence* for process algebra with fluid semantics [74]. It relates processes that are equivalent whenever their ODE solutions are equal at all time points. Label equivalence is only a sufficient condition for ODE reduction since it works at a coarser level of granularity. Indeed, it relates sets of ODE variables, each corresponding to the behaviour of a sequential process. Instead, BDE/BB relate individual ODE variables. In addition, no algorithm for computing label equivalence is available. Analogously to FDE/FB, for ODEs that represent a CTMC we have that BDE and BB correspond to exact CTMC lumpability [14].

Model reductions have been extensively studied for CRNs in systems biology. In particular, for protein interaction networks, the combinatorial explosion of the state space has motived considerable research, e.g., [24,23,35,25,16,34,15,19]. The fragmentation approach for the rule-based language κ identifies a coarse-grained ODE system for models with mass-action semantics through sums of variables; this is weaker than an equivalence relation over species, because one variable may appear in more than one block (a *fragment*) [35,25]. Using the terminology of [62], fragmentation is a form of *improper lumping*, as opposed to the notions of equivalence presented here where a species belongs to a single block.

SMT has become a cornerstone in the programming languages and in the verification community, with contributions to program synthesis [41], constraint programming [53], and symbolic optimization [56]. The combination of SMT and equivalence relations has been the subject of recent investigations. In [7] partition-refinement algorithms are proposed to compute equivalences between terms over arbitrary theories inferred from a set of axioms. Applied to equivalences presented here, these partition-refinement algorithms could be used to check if a candidate partition is a differential equivalence, but not to compute the largest equivalence for an IDOL program. In [29] the authors present an SMT-based approach for the computation of the coarsest ordinary lumpable partition of a Markov chain, but for a fragment of the PRISM language [54].

Finally, links between ODEs and SMT are established in the formal verification community, especially for hybrid systems (e.g., [39,67,59]); however none of these works considers ODE comparisons and minimizations through equivalence relations. Bisimulation for dynamical systems have been studied by Pappas and co-authors [64] and van der Schaft [69]. These works are similar in spirit to ours, but the setting is different because the focus is on *control systems*, i.e., dynamical systems with internal states, external inputs, and output maps. In that context, bisimulation relates internal states mapped to the same output, i.e., they cannot be told apart by an external observer. The largest bisimulation is therefore related to the maximal unobservability subspace of a control system (e.g., [69, Corollary 6.4]) while our largest differential equivalences provide the coarsest partition of ODE variables that preserves the dynamics.

5 Conclusion

This paper has presented a number of techniques for the automatic reduction of systems of ordinary differential equations (ODEs), motivated by their popularity in the modelling and analysis of large-scale dynamical systems such as collective adaptive systems. The symbolic approach of differential equivalences and the syntax-driven minimisation through reaction-network (RN) bisimulations offer a trade-off between expressiveness and efficiency.

Differential equivalences support a rather rich class of non-linear ODE, which can be analysed by using satisfiability solvers as the underlying engine. In general, it is well known that such solvers are more efficient in providing a positive "sat" result than a negative "unsat", which is however required to check that a candidate partition is a differential equivalence. Nevertheless, the current technology allows us to analyse models of realistic size (see also [21] for further examples). In the current prototype implementation the SMT solver is used as a black-box; it would be interesting in the future to consider the development of domain-specific heuristics that improve the search.

RN bisimulations are particularly efficient since the partition-refinement algorithms run in polynomial time and space; however, currently they support ODE with derivatives given by multivariate polynomials of order at most two. Nevertheless, they cover an interesting class of systems, including CRNs and affine

systems (see also [20] for experiments in large-scale benchmarks). To further improve efficiency it would be interesting to consider parallelisation techniques; on a more theoretical viewpoint, an obvious direction for future research is to extend the bisimulations of higher-order multivariate polynomials.

The forward and backward variants of the presented equivalences are not comparable in general. This suggests a possible combined use, which has however not been investigated so far. A better understanding of the relationship between these two variants may help achieve further reductions.

Much of the efficiency in computing ODE reductions is owed to the fact that the largest differential equivalences and bisimulations exist and can be computed via partition-refinement. We argue, however, that there are situations of practical interest that cannot be cast into this framework. For example, the notion of emulation that is instrumental to investigate evolutionary aspects of CRNs, amounts to finding a particular backward bisimulation where each equivalence class contains exactly one species of the small CRN and at most one species of the larger CRN. This condition cannot be expressed as a suitable initial partition to be refined; hence, one is left with having to enumerate all possible partitions that satisfy these conditions in order to find emulations automatically. However, this is feasible only for very simple models. Further research is needed to develop algorithms that aggregate according to more liberal constraints on the desired equivalence classes.

Finally, we remark that all the techniques presented in this paper are concerned with exact aggregations. In some cases, these may be too strong because even small perturbations may discriminate ODE variables that have nearby trajectories in practice. This has motivated a large body of work into approximate notions of equivalence [65,1,13,42]. Preliminary work for models based on ODE semantics has been carried out in [76] in the case of process algebra; more general ODE systems are treated in [46,75]. However all these approaches still lack an algorithm for automatic reduction. Furthermore, they provide a priori bounds on the approximate aggregation that tend to grow fast with time. Future research work will be aimed at tackling these two issues.

Acknowledgement

This work was partially supported by the EU project QUANTICOL, 600708. The authors thank Luca Cardelli and Max Tschaikowski who co-authored the papers [19,20,21] used as background material in this chapter.

References

1. Aldini, A., Bravetti, M., Gorrieri, R.: A process-algebraic approach for the analysis of probabilistic noninterference. Journal of Computer Security 12(2), 191–245 (2004)
2. Aoki, M.: Control of large-scale dynamic systems by aggregation. IEEE Trans. Autom. Control 13(3), 246–253 (1968)

3. Baier, C., Engelen, B., Majster-Cederbaum, M.E.: Deciding bisimilarity and similarity for probabilistic processes. J. Comput. Syst. Sci. 60(1), 187–231 (2000)
4. Barrett, C., Sebastiani, R., Seshia, S.A., Cesare Tinelli Clark Barrett, Roberto Sebastiani, S.A.S., Tinelli, C.: Handbook of Satisfiability: Volume 185 Frontiers in Artificial Intelligence and Applications, chap. Satisfiability Modulo Theories. IOS Press (2009)
5. Barrett, C., Stump, A., Tinelli, C.: The SMT-LIB Standard: Version 2.0. Tech. rep., Department of Computer Science, The University of Iowa (2010), available at `www.SMT-LIB.org`
6. Barua, D., Hlavacek, W.S.: Modeling the effect of APC truncation on destruction complex function in colorectal cancer cells. PLoS Comput Biol 9(9), e1003217 (09 2013)
7. Berdine, J., Bjørner, N.: Computing all implied equalities via SMT-based partition refinement. In: Automated Reasoning, LNCS, vol. 8562, pp. 168–183. Springer (2014)
8. Blinov, M.L., Faeder, J.R., Goldstein, B., Hlavacek, W.S.: BioNetGen: software for rule-based modeling of signal transduction based on the interactions of molecular domains. Bioinformatics 20(17), 3289–3291 (2004)
9. Bortolussi, L., De Nicola, R., Galpin, V., Gilmore, S., Hillston, J., Latella, D., Loreti, M., Massink, M.: CARMA: Collective Adaptive Resource-sharing Markovian Agents. In: QAPL. pp. 16–31 (2015)
10. Bortolussi, L., Gast, N.: Scalable quantitative analysis: Fluid and hybrid approximations. In: SFM (2016)
11. Bortolussi, L., Latella, D., Massink, M.: Stochastic process algebra and stability analysis of collective systems. In: Coordination Models and Languages, 15th International Conference, COORDINATION 2013, Held as Part of the 8th International Federated Conference on Distributed Computing Techniques, DisCoTec 2013, Florence, Italy, June 3-5, 2013. Proceedings. pp. 1–15 (2013)
12. Bortolussi, L., Le Boudec, J.Y., Latella, D., Massink, M.: Revisiting the Limit Behaviour of "El Botellon". In: Tech. rep., http://infoscience.epfl.ch/record/179935/ (2012)
13. van Breugel, F., Worrell, J.: Approximating and computing behavioural distances in probabilistic transition systems. Theoretical Computer Science 360(1–3), 373–385 (8 2006)
14. Buchholz, P.: Exact and Ordinary Lumpability in Finite Markov Chains. Journal of Applied Probability 31(1), 59–75 (1994)
15. Camporesi, F., Feret, J.: Formal reduction for rule-based models. Electronic Notes in Theoretical Computer Science 276, 29–59 (2011)
16. Camporesi, F., Feret, J., Koeppl, H., Petrov, T.: Combining model reductions. Electronic Notes in Theoretical Computer Science 265, 73–96 (2010)
17. Cardelli, L.: Morphisms of reaction networks that couple structure to function. BMC Systems Biology 8(1), 84 (2014)
18. Cardelli, L., Csikász-Nagy, A., Dalchau, N., Tribastone, M., Tschaikowski, M.: Noise reduction in complex biological switches. Scientific Reports 6, 20214 EP – (02 2016), `http://dx.doi.org/10.1038/srep20214`
19. Cardelli, L., Tribastone, M., Tschaikowski, M., Vandin, A.: Forward and backward bisimulations for chemical reaction networks. In: CONCUR (2015)
20. Cardelli, L., Tribastone, M., Tschaikowski, M., Vandin, A.: Efficient syntax-driven lumping of differential equations. In: TACAS (2016), to appear.
21. Cardelli, L., Tribastone, M., Tschaikowski, M., Vandin, A.: Symbolic computation of differential equivalences. In: POPL (2016)

22. Ciocchetta, F., Hillston, J.: Bio-PEPA: A framework for the modelling and analysis of biological systems. TCS 410(33-34), 3065–3084 (2009)
23. Conzelmann, H., Fey, D., Gilles, E.: Exact model reduction of combinatorial reaction networks. BMC Systems Biology 2(1), 78 (2008)
24. Conzelmann, H., Saez-Rodriguez, J., Sauter, T., Kholodenko, B., Gilles, E.: A domain-oriented approach to the reduction of combinatorial complexity in signal transduction networks. BMC Bioinformatics 7(1), 34 (2006)
25. Danos, V., Feret, J., Fontana, W., Harmer, R., Krivine, J.: Abstracting the differential semantics of rule-based models: Exact and automated model reduction. In: LICS. pp. 362–381 (2010)
26. De Moura, L., Bjørner, N.: Z3: An efficient SMT solver. In: TACAS. pp. 337–340 (2008)
27. De Nicola, R., Latella, D., Loreti, M., Massink, M.: Rate-based transition systems for stochastic process calculi. In: ICALP (2). pp. 435–446 (2009)
28. De Nicola, R., Loreti, M., Pugliese, R., Tiezzi, F.: A formal approach to autonomic systems programming: The SCEL language. TAAS 9(2), 7:1–7:29 (2014)
29. Dehnert, C., Katoen, J.P., Parker, D.: SMT-based bisimulation minimisation of Markov models. In: VMCAI. LNCS, vol. 7737, pp. 28–47 (2013)
30. DeMaio, P.: Bike-sharing: History, impacts, models of provision, and future. Journal of Public Transportation 12(4) (2009)
31. Derisavi, S., Hermanns, H., Sanders, W.: Optimal state-space lumping in Markov chains. Inf. Process. Lett. 87(6), 309–315 (2003)
32. Feng, C., Hillston, J.: Automatic moment-closure approximation of spatially distributed collective adaptive systems. ACM Transactions on Modeling and Computer Simulation. To appear.
33. Feng, C., Hillston, J.: PALOMA: A process algebra for located Markovian agents. In: QEST. pp. 265–280 (2014)
34. Feret, J.: Fragments-based model reduction: Some case studies. Electronic Notes in Theoretical Computer Science 268, 77–96 (2010)
35. Feret, J., Danos, V., Krivine, J., Harmer, R., Fontana, W.: Internal coarse-graining of molecular systems. Proceedings of the National Academy of Sciences 106(16), 6453–6458 (2009)
36. Feret, J., Henzinger, T., Koeppl, H., Petrov, T.: Lumpability abstractions of rule-based systems. Theoretical Computer Science 431(0), 137–164 (2012)
37. Fricker, C., Gast, N.: Incentives and redistribution in homogeneous bike-sharing systems with stations of finite capacity. EURO Journal on Transportation and Logistics pp. 1–31 (2014)
38. Galpin, V.: Spatial representations and analysis techniques. In: SFM (2016)
39. Gao, S., Kong, S., Clarke, E.: Satisfiability modulo ODEs. In: FMCAD. pp. 105–112 (2013)
40. Grosu, R., Bartocci, E.: Spatio-temporal model checking. In: SFM (2016)
41. Gulwani, S., Jha, S., Tiwari, A., Venkatesan, R.: Synthesis of loop-free programs. In: PLDI. pp. 62–73 (2011)
42. Gupta, V., Jagadeesan, R., Panangaden, P.: Approximate reasoning for real-time probabilistic processes. Logical Methods in Computer Science 2(1) (2006)
43. Hayden, R.A., Bradley, J.T.: A fluid analysis framework for a Markovian process algebra. Theoretical Computer Science 411(22-24), 2260–2297 (2010)
44. Hillston, J.: A Compositional Approach to Performance Modelling. CUP (1996)
45. Hillston, J.: Fluid flow approximation of PEPA models. In: QEST. pp. 33–43 (Sep 2005)

46. Iacobelli, G., Tribastone, M.: Lumpability of fluid models with heterogeneous agent types. In: DSN. pp. 1–11 (2013)
47. Iacobelli, G., Tribastone, M., Vandin, A.: Differential bisimulation for a Markovian process algebra. In: MFCS. pp. 293–306 (2015)
48. Iwasa, Y., Andreasen, V., Levin, S.: Aggregation in model ecosystems. I. Perfect aggregation. Ecological Modelling 37(3-4), 287–302 (1987)
49. Jovanovic, D., de Moura, L.M.: Solving non-linear arithmetic. In: IJCAR. pp. 339–354 (2012)
50. Katoen, J., Khattri, M., Zapreev, I.: A Markov Reward Model Checker. In: QEST. pp. 243–244 (2005)
51. Kermack, W.O., McKendrick, A.: A Contribution to the Mathematical Theory of Epidemics. Proceedings of the Royal Society of London. Series A, Containing Papers of a Mathematical and Physical Character 115(772), 700–721 (Aug 1927)
52. Kocieniewski, P., Faeder, J.R., Lipniacki, T.: The interplay of double phosphorylation and scaffolding in MAPK pathways. Journal of Theoretical Biology 295, 116–124 (2012)
53. Köksal, A.S., Kuncak, V., Suter, P.: Constraints as control. In: POPL. pp. 151–164 (2012)
54. Kwiatkowska, M., Norman, G., Parker, D.: Prism 4.0: Verification of probabilistic real-time systems. In: CAV. pp. 585–591 (2011)
55. Larsen, K.G., Skou, A.: Bisimulation through probabilistic testing. Information and Computation 94(1), 1–28 (1991)
56. Li, Y., Albarghouthi, A., Kincaid, Z., Gurfinkel, A., Chechik, M.: Symbolic optimization with SMT solvers. In: POPL. pp. 607–618 (2014)
57. Loreti, M., Hillston, J.: Modeling and analysis of collective adaptive systems with CARMA and its tools. In: SFM (2016)
58. Massink, M., Latella, D., Bracciali, A., Hillston, J.: Modelling non-linear crowd dynamics in Bio-PEPA. In: FASE. pp. 96–110 (2011)
59. Mover, S., Cimatti, A., Tiwari, A., Tonetta, S.: Time-aware relational abstractions for hybrid systems. In: EMSOFT. pp. 1–10 (2013)
60. Nelson, D.L., Cox, M.M.: Lehninger Principles of Biochemistry. Palgrave Macmillan, 6th edn. (2013)
61. Norris, J.: Markov Chains. Cambridge Series in Statistical and Probabilistic Mathematics, Cambridge University Press (1998)
62. Okino, M.S., Mavrovouniotis, M.L.: Simplification of mathematical models of chemical reaction systems. Chemical Reviews 2(98), 391–408 (1998)
63. Paige, R., Tarjan, R.: Three partition refinement algorithms. SIAM Journal on Computing 16(6), 973–989 (1987)
64. Pappas, G.J.: Bisimilar linear systems. Automatica 39(12), 2035–2047 (2003)
65. Pierro, A.D., Hankin, C., Wiklicky, H.: Quantitative relations and approximate process equivalences. In: CONCUR. pp. 498–512 (2003)
66. Rowe, J.E., Gomez, R.: El Botellon: Modeling the movement of crowds in a city. Complex Systems 14, 363–370 (2003)
67. Sankaranarayanan, S., Tiwari, A.: Relational abstractions for continuous and hybrid systems. In: CAV. pp. 686–702 (2011)
68. Sattenspiel, L., Herring, D.A.: Simulating the effect of quarantine on the spread of the 1918–19 flu in Central Canada. Bulletin of Mathematical Biology 65(1), 1–26, http://dx.doi.org/10.1006/bulm.2002.0317
69. van der Schaft, A.J.: Equivalence of dynamical systems by bisimulation. IEEE Transactions on Automatic Control 49 (2004)

70. Sproston, J., Donatelli, S.: Backward bisimulation in Markov chain model checking. IEEE Trans. Software Eng. 32(8), 531–546 (2006)
71. Suderman, R., Deeds, E.J.: Machines vs. ensembles: Effective MAPK signaling through heterogeneous sets of protein complexes. PLoS Comput Biol 9(10), e1003278 (10 2013)
72. Toth, J., Li, G., Rabitz, H., Tomlin, A.S.: The effect of lumping and expanding on kinetic differential equations. SIAM Journal on Applied Mathematics 57(6), 1531–1556 (1997)
73. Tribastone, M., Gilmore, S., Hillston, J.: Scalable differential analysis of process algebra models. IEEE Trans. Software Eng. 38(1), 205–219 (2012)
74. Tschaikowski, M., Tribastone, M.: Exact fluid lumpability for Markovian process algebra. In: CONCUR (2012)
75. Tschaikowski, M., Tribastone, M.: Approximate reduction of heterogenous nonlinear models with differential hulls. IEEE Transactions on Automatic Control (2015), to appear
76. Tschaikowski, M., Tribastone, M.: A unified framework for differential aggregations in markovian process algebra. Journal of Logical and Algebraic Methods in Programming 84(2), 238–258 (2015)
77. Valmari, A., Franceschinis, G.: Simple $O(m \log n)$ time Markov Chain lumping. In: TACAS. pp. 38–52 (2010)
78. Voit, E.O.: Biochemical systems theory: A review. ISRN Biomathematics 2013, 53 (2013), http://dx.doi.org/10.1155/2013/897658%]897658