Scalable Differential Analysis of Process Algebra Models

Tribastone, M., Gilmore, S., & Hillston, J. (2012). Scalable Differential Analysis of Process Algebra Models. IEEE Transactions on Software Engineering, 38(1), 205-219. 10.1109/TSE.2010.82

Scalable Differential Analysis of Process Algebra Models

Mirco Tribastone, Stephen Gilmore, and Jane Hillston

Abstract—The exact performance analysis of large-scale software systems with discrete-state approaches is difficult because of the well-known problem of state-space explosion. This paper considers this problem with regard to the stochastic process algebra PEPA, presenting a deterministic approximation to the underlying Markov chain model based on ordinary differential equations. The accuracy of the approximation is assessed by means of a substantial case study of a distributed multi-threaded application.

Index Terms—Modeling and Prediction, Ordinary Differential Equations, Markov processes.

1 Introduction

Continuous-time Markov chains (CTMCs) are an established tool for the quantitative analysis of systems, and a vast body of research in this area has resulted in a wide variety of techniques for their evaluation, such as efficient and numerically robust methods for the computation of transient and steady-state probability distributions [1], [2], fast simulation algorithms [3], and stochastic model checking [4]. However, as with most discrete-state analysis techniques, the major drawback is the well-known problem of state-space explosion (i.e., the number of reachable states of the chain grows combinatorially with the number of individuals in the system), which is only partially alleviated by ingenious research devoted to exploiting symmetries in the model in order to obtain a smaller (*lumped*) CTMC which still preserves most of the information on the stochastic behaviour of the original process (e.g., [5]).

An alternative technique for performance evaluation may be offered by *deterministic models*, which use ordinary differential equations (ODEs) as the underlying mathematical structure. Here the temporal evolution of the population of inherently discrete entities is approximated in a continuous fashion. As a result, large-scale models are much easier to handle because the actual population size of the system under study does not impact on the ODE representation. Despite their apparently contrasting modelling approach, in many circumstances it is possible to establish a very useful relationship of convergence between the stochastic and deterministic models, where the ODE is interpreted as the fluid limiting behaviour of a *family* of CTMCs associated with the model under evaluation and parametrised by a system factor such as density or concentration [6], [7].

• *The authors are with the Laboratory for Foundations of Computer Science, School of Informatics, The University of Edinburgh, Edinburgh, UK. E-mail:{mtribast,stg,jeh}@inf.ed.ac.uk*

The main contribution of this paper is a novel operational semantics for the process calculus PEPA, which allows us to demonstrate this deterministic convergence for population models expressed in the language. The semantics gives rise to a compact symbolic representation of the CTMC of the model, from which it is possible to infer the corresponding ODE representing its fluid limit. This semantics provides a formal account of earlier approaches to deterministic interpretations of PEPA [8], and substantially extends their scope of applicability by incorporating all the operators of the language and removing earlier assumptions on the syntactical structure of the models amenable to analysis.

Population-based modelling is particularly suitable for capturing the dynamics of large-scale distributed systems, because these usually consist of many independent copies of components with the same behaviour, e.g., tens or hundreds of threads or processes which deal with requests from potentially many customers. This paper presents an example of a three-tier distributed application modelled in PEPA, on which we conduct a large numerical study to assess the accuracy of the differential analysis.

Structure of this paper: Section 2 gives an overview of PEPA, with focus on the main issues regarding large-scale modelling. Section 3 discusses the population-based operational semantics, and the result of deterministic convergence is presented in Section 4. Section 5 presents the case study and a validation study on the accuracy of the approximation. Section 6 discusses related work and Section 7 gives concluding remarks.

2 Background and Motivation

Using a running example, this section introduces the notions of PEPA which will be used extensively throughout the remainder of the paper. The reader is referred to [9] for a formal definition. Then, the problem of state space explosion in the context of PEPA is discussed and the method of fluid-flow approximation is informally

TABLE 1
Notation

Symbol	*Meaning*
$\alpha, \beta, \ldots \in \mathcal{A}$	Action types of a PEPA model
$P, P', Q, \ldots$	PEPA components
$r, r_1, s, \ldots \in \mathbb{R}$	Rates of a PEPA activity
$r_\alpha(P)$	Apparent rate of α in P (Markovian semantics)
$ds(P)$	Derivative set of P
$C_{i,j}$	j-th derivative of the i-th component in the numerical vector form, $1 \leq i \leq N_C, 1 \leq j \leq N_i$
$\xi_{i,j}$	Coordinate in $\xi \in \mathbb{N}_0^d$ for population level $C_{i,j}$
$r_\alpha^\star(P, \xi)$	Parametric apparent rate of α in P
$ds^\star(P)$	Parametric derivative set of P
$\varphi_\alpha(\cdot, l)$	Generating function for action α and jump $l \in \mathbb{Z}^d$
$\delta \in \mathbb{N}_0^d$	Density vector (initial state of the CTMC)
$\{X_n(t)\}$	Family of CTMCs with initial state $X_n(0) = n\delta$
$x(t)$	Dependent variable of the ODE
$F(\cdot)$	Vector field of the underlying ODE

introduced. Table 1 summarises the main notation and terminology used in this paper.

2.1 Overview of PEPA

PEPA is a CSP-like process algebra extended with the notion of stochastically timed activities. It supports the following operators.

Prefix: $(\alpha, r).P$ denotes a process which performs an action of type α and behaves as P subsequently. The activity is associated with an exponential distribution with mean duration $1/r, r \in \mathbb{R}_{>0}$. (This paper does not deal directly with *passive* activities, however a discussion on their use is given in Section 4.3.) The set of all the activities (α, r) in a PEPA model is denoted by $\mathcal{A}ct$ and the set of all action types is denoted by $\mathcal{A}$.

Choice: $P + Q$ specifies a component which behaves either as P or as Q. The activities of both operands are enabled and the choice will behave as the component which first completes. For instance, $(\alpha, r).P + (\beta, s).Q$ behaves as P (resp., Q) with probability $r/(r+s)$ (resp., $s/(r+s)$).

Constant: $A \stackrel{def}{=} P$ is used for recursion. Cyclic definitions are central in the characterisation of the underlying CTMC of a PEPA model.

Cooperation: $P \underset{L}{\bowtie} Q$ is the compositional operator of PEPA. Components P and Q synchronise over the set of action types in set L; other actions are performed independently. For example, $(\alpha, r_1).(\beta, s).P \underset{\{\alpha\}}{\bowtie} (\alpha, r_2).(\gamma, t).Q$ is a composition of two processes which execute α cooperatively. Then, they perform actions β and γ independently and behave as P and Q, respectively. Cooperating components need not have a common view of the duration of shared actions. The semantics of PEPA specifies that the rate of a shared action is the slowest of the individual rates of the synchronising components, e.g., $\min(r_1, r_2)$ in the example above. This corresponds to the assumption of *unbounded capacity* [10]. The operator $\|$ is sometimes used as shorthand notation for a cooperation over an empty set, i.e., $\underset{\emptyset}{\bowtie}$. The notation $S[N]$ indicates N independent copies of a sequential component S and will be extensively used in the reminder of this paper as the abbreviated form of $\underbrace{S \parallel S \parallel \cdots \parallel S}_{N}$.

Hiding: P/L relabels the activities of P with the *silent action* τ for all types in L. Thus, $\left((\alpha, r_1).P/\{\alpha\}\right) \underset{\{\alpha\}}{\bowtie} (\alpha, r_2).Q$ does not cooperate over action α because the process in the left-hand side of the cooperation performs a transition (τ, r_1) to P.

An interesting class of PEPA models comprises those which can be generated by the two-level grammar

$$S ::= (\alpha, r).S \quad | \quad S + S$$
$$C ::= S \quad | \quad C \underset{L}{\bowtie} C \quad | \quad C/L$$

where the former production defines *sequential components* and the latter produces *model components*. The *system equation* designates the model component that defines the environment which embraces all of the behaviour of the system under study. In the remainder of this paper system equations are denoted with constants such as $System$. Models from this grammar give rise to a finite underlying CTMC. We consider these models throughout the remainder of this paper.

Example 1 (PEPA model with cooperation)**.**

$$P \stackrel{\text{def}}{=} (\alpha, p).P'$$
$$P' \stackrel{\text{def}}{=} (\beta, p').P$$
$$Q \stackrel{\text{def}}{=} (\alpha, q).Q'$$
$$Q' \stackrel{\text{def}}{=} (\gamma, q').Q$$
$$System_1 \stackrel{\text{def}}{=} P[N_P] \underset{\{\alpha\}}{\bowtie} Q[N_Q]$$

The model comprises two arrays of components, with initial state P and Q, where each pair (P, Q) can cooperate over the action type α. There are N_P instances of P and N_Q instances of Q. P and Q carry out independent actions β and γ, respectively, before returning to the state in which α may be performed.

Without loss of generality, we assume $N_P, N_Q > 1$. The derivations that follow in this paper are also valid for $N_P = N_Q = 1$, although this case leads to less insightful and simpler derivation trees and recursion stacks.

Definition 1. *The* apparent rate *of action α in process P, denoted by $r_\alpha(P)$, indicates the overall rate at which α can be performed by P. It is recursively defined as follows:*

$$r_\alpha((\beta, r).P) = \begin{cases} r & \text{if} \quad \beta = \alpha \\ 0 & \text{if} \quad \beta \neq \alpha \end{cases}$$

$$r_\alpha(P + Q) = r_\alpha(P) + r_\alpha(Q)$$

$$r_\alpha\left(P \underset{L}{\bowtie} Q\right) = \begin{cases} \min(r_\alpha(P), r_\alpha(Q)) & \text{if } \alpha \in L \\ r_\alpha(P) + r_\alpha(Q) & \text{if } \alpha \notin L \end{cases}$$

$$r_\alpha(P/L) = \begin{cases} r_\alpha(P) & \text{if} \quad \alpha \notin L \\ 0 & \text{if} \quad \alpha \in L \end{cases}$$

According to this definition, for the array of sequential components $P[N_P]$ the apparent rate of α is

$$r_\alpha\left(P[N_P]\right) = N_P\, r_\alpha\left(P\right), \tag{1}$$

and this holds for any $\alpha \in \mathcal{A}$ and any N_P because all the cooperation sets amongst such components are empty. Similarly, this additivity holds in general for parallel compositions of distinct arrays,

$$r_\alpha\left(P_1[N_1] \,\|\, P_2[N_2] \,\|\cdots\|\, P_K[N_K]\right) = \sum_{i=1}^{K} N_i r_\alpha\left(P_i\right) \tag{2}$$

The semantics of PEPA is shown in Table 2. Given a PEPA component P, the operational semantics induces the *derivative set*[1], denoted by $ds(P)$, which is the set of the possible states reachable from P. We use the term *local derivative* to indicate a state reachable from a sequential component (which is itself a sequential component, as can be seen from the two-level grammar). A *derivation graph* whose nodes are in $ds(P)$ and arcs in $ds(P) \times \mathcal{A}ct \times ds(P)$ indicates all the transitions between each pair of derivatives of P. Arcs are taken with multiplicity, corresponding to the number of distinct inference trees which give the same transition. Ultimately, the derivation graph can be directly mapped onto a CTMC for performance analysis.

Using Example 1, we now consider how the presence of arrays of sequential components affects the derivation tree for a transition and the derivative set of a model component. By rules S_0 and S_1 the following two transitions can be inferred for P and Q:

$$P \xrightarrow{(\alpha,p)} P' \tag{3}$$

$$Q \xrightarrow{(\alpha,q)} Q' \tag{4}$$

The dynamic behaviour of the leftmost component P of the array can be collected by $N_P - 1$ applications of rule C_0. The first application has the form:

$$\frac{P \xrightarrow{(\alpha,p)} P'}{P \parallel P \xrightarrow{(\alpha,p)} P' \parallel P}$$

Then, for $1 \leq i \leq N_P - 2$, the other $N_P - 2$ applications are of type

$$\frac{P \parallel P[i] \xrightarrow{(\alpha,p)} P' \parallel P[i]}{P \parallel P[i] \parallel P \xrightarrow{(\alpha,p)} P' \parallel P[i] \parallel P}$$

For $i = N_P - 2$, the conclusion of this rule may be written as

$$P[N_P] \xrightarrow{(\alpha,p)} P' \parallel P[N_P - 1] \tag{5}$$

The behaviour of the leftmost component Q can be collected in a similar way, leading to a transition in the form

$$Q[N_Q] \xrightarrow{(\alpha,q)} Q' \parallel Q[N_Q - 1] \tag{6}$$

1. The term *derivative* is used in PEPA to denote a reachable state of a component. It is not to be confused with the notion of derivative in calculus, which will be used later in this paper for the deterministic interpretation of the stochastic process underlying a PEPA model.

TABLE 2
Markovian semantics of PEPA (from [9]).

Prefix

$$S_0 : \frac{}{(\alpha, r).P \xrightarrow{(\alpha,r)} P}$$

Choice

$$S_1 : \frac{P \xrightarrow{(\alpha,r)} P'}{P + Q \xrightarrow{(\alpha,r)} P' + Q} \qquad S_2 : \frac{Q \xrightarrow{(\alpha,r)} Q'}{P + Q \xrightarrow{(\alpha,r)} P + Q'}$$

Cooperation

$$C_0 : \frac{P \xrightarrow{(\alpha,r)} P'}{P \underset{L}{\bowtie} Q \xrightarrow{(\alpha,r)} P' \underset{L}{\bowtie} Q}, \alpha \notin L$$

$$C_1 : \frac{Q \xrightarrow{(\alpha,r)} Q'}{P \underset{L}{\bowtie} Q \xrightarrow{(\alpha,r)} P \underset{L}{\bowtie} Q'}, \alpha \notin L$$

$$C_2 : \frac{P \xrightarrow{(\alpha,r_1)} P' \qquad Q \xrightarrow{(\alpha,r_2)} Q'}{P \underset{L}{\bowtie} Q \xrightarrow{(\alpha,R)} P' \underset{L}{\bowtie} Q'}, \alpha \in L$$

$$R = \frac{r_1}{r_\alpha(P)} \frac{r_2}{r_\alpha(Q)} \min\left(r_\alpha(P), r_\alpha(Q)\right)$$

Hiding

$$H_0 : \frac{P \xrightarrow{(\alpha,r)} P'}{P/L \xrightarrow{(\alpha,r)} P'/L}, \alpha \notin L \qquad H_1 : \frac{P \xrightarrow{(\alpha,r)} P'}{P/L \xrightarrow{(\tau,r)} P'/L}, \alpha \in L$$

Constant

$$A_0 : \frac{P \xrightarrow{(\alpha,r)} P'}{A \xrightarrow{(\alpha,r)} P'}, A \stackrel{def}{=} P$$

Finally, by applying rule C_2 to (5) and (6) we obtain

$$P[N_P] \underset{\{\alpha\}}{\bowtie} Q[N_Q] \xrightarrow{(\alpha,R)} P' \parallel P[N_P - 1] \underset{\{\alpha\}}{\bowtie} Q' \parallel Q[N_Q - 1] \tag{7}$$

where, by rule C_2,

$$\begin{aligned} R &= \frac{p}{r_\alpha\left(P[N_P]\right)} \frac{q}{r_\alpha\left(Q[N_Q]\right)} \min\left(r_\alpha\left(P[N_P]\right), r_\alpha\left(Q[N_Q]\right)\right) \\ &= \frac{p}{N_P\, r_\alpha\left(P\right)} \frac{q}{N_Q\, r_\alpha\left(Q\right)} \min\left(N_P\, r_\alpha\left(P\right), N_Q r_\alpha\left(Q\right)\right) \\ &= \frac{p}{N_P\, p} \frac{q}{N_Q\, q} \min\left(N_P\, p, N_Q\, q\right) = \frac{\min\left(N_P\, p, N_Q\, q\right)}{N_P N_Q} \end{aligned} \tag{8}$$

The conclusion of (7) is not the only transition enabled by the initial state, because each individual component P can be paired with each component Q to carry out the shared activity. Hence, $P[N_P] \underset{\{\alpha\}}{\bowtie} Q[N_Q]$ enables $N_P \cdot N_Q$ transitions to distinct states of type

$$\underbrace{P \parallel \cdots \parallel P \parallel P' \parallel P \parallel \cdots \parallel P}_{N_P \text{ sequential components}} \underset{\{\alpha\}}{\bowtie} \underbrace{Q \parallel \cdots \parallel Q \parallel Q' \parallel Q \parallel \cdots \parallel Q}_{N_Q \text{ sequential components}}$$

which only differ in the locations of the components P' and Q'. Since each transition occurs at rate R, the exit rate from $P[N_P] \underset{\{\alpha\}}{\bowtie} Q[N_Q]$ is $N_P \cdot N_Q \cdot R = \min(N_P\, p, N_Q q)$, and the factor $1/(N_P \cdot N_Q)$ is the probability that one pair of components makes the transition.

2.2 Aggregation and State Representation

If we consider a representative state for all the possible configurations in which there is one component P' and $N_P - 1$ components P, this state is isomorphic to each configuration [9]. Isomorphism was exploited in [11] for exact aggregation of the underlying CTMC using a state representation (called *canonical*) in which the local derivatives within an array are arranged in lexicographical order. In this aggregated CTMC the state $P[N_P] \underset{\{\alpha\}}{\bowtie} Q[N_Q]$ has one transition to the representative state $P' \parallel P[N_P - 1] \underset{\{\alpha\}}{\bowtie} Q' \parallel Q[N_Q - 1]$ with rate $\min(N_P\, p, N_Q q)$.

Further refinement on state-space representation aims at reducing storage requirements by relying upon two properties. First, models defined according to the two-level grammar described above have a static structure of the compositional operators. As such, this structure needs not be recorded in the state descriptor. Second, if the number of copies of an array is larger than the size of the derivative set of the replicated component, then a state representation with *counter variables*, called the *numerical vector form* (NVF) [8], leads to a more parsimonious data structure for storage. In the NVF, the state is represented as a vector of integers; each coordinate is associated with a distinct local derivative of the sequential components in the system and it records the number of components exhibiting that derivative. The state descriptor in the NVF is denoted by $\xi \in \mathbb{N}_0^d$, where d is the total number of local derivatives in the system. For instance, a possible NVF-representation for the transition in (7) is

$$(N_P, 0, N_Q, 0) \xrightarrow{(\alpha, \min(N_P p, N_Q q))} (N_P - 1, 1, N_Q - 1, 1) \quad (9)$$

where, from left to right, the coordinates are assigned to the local derivatives P, P', Q, and Q'.

It is important to point out that a CTMC in the NVF is not aggregated any more than the CTMC from the same PEPA model in the canonical form is, as the NVF only operates at the level of a single state descriptor. In addition, aggregation via isomorphism only reduces the problem of state-space explosion—although in most cases it can achieve dramatic reductions in size; with increasing population levels the derivation of the transition system and the solution of the underlying CTMC will eventually become impractical.

2.3 Deterministic Approximation

State-space explosion may be tackled more effectively by shifting the focus to an approximate representation in which the dynamics of inherently discrete components is given in a continuous fashion. The underlying mathematics used here is that of ODEs, in which the dependent function $x(t)$ has values in vectors of reals and each of its coordinates corresponds to the (continuous) population level of a sequential component in the system (hence, it is the deterministic counterpart of ξ in the NVF).

Clearly, a fundamental requisite for such an approximation is that the ODE be inferred from the PEPA model statically, i.e. without the explicit enumeration of the entire state space of the underlying CTMC. Most important, the deterministic interpretation must be made compatible with the original stochastic treatment, so as to justify the approximation and reason about its accuracy. The former condition is satisfied by the earlier work on this topic [8], as the ODE is derived only by inspection of the model description via the construction of the *activity graph*, a structure which records the rates of change of the population levels when an activity is carried out. In contrast, the relationship between the ODE and the CTMC is not investigated, but numerical evidence showing good agreement is given.

Here we establish this relationship by developing a structured operational semantics of PEPA by which the ODE is inferred from a *symbolic* representation of an aggregated CTMC in the NVF. The operational semantics leads to the derivation of *generating functions* of the CTMC, i.e., functions of the state descriptor which give the transition rates to all the reachable states of the system. These functions are parametrised by action types to keep track of the additional information about which action type is associated with a transition. Let $l \in \mathbb{Z}^d$ be the *transition jump*, i.e., the transition moves from state ξ to $\xi + l$. The generating functions are denoted by $\varphi_\alpha(\xi, l) : \mathbb{R}^d \to \mathbb{R}$ and give the transition rate for a jump l and an activity of type $\alpha \in \mathcal{A}$. Thus, the entry in the generator matrix corresponding to the transition from ξ to $\xi + l$, denoted by $q_{\xi,\xi+l}$, can be written as

$$q_{\xi,\xi+l} = \sum_{\alpha \in \mathcal{A}} \varphi_\alpha(\xi, l).$$

The summation across $\mathcal{A}$ captures the fact that distinct action types may contribute to a transition to the same target state, e.g., $(\alpha, p).P + (\beta, s).P$. These transitions are kept distinct in the labelled transition system of PEPA, because it records the action type as well as the transition rate, but they collapse onto the same entry in the underlying generator matrix. We use the notation

$$\varphi(\xi, l) \equiv \sum_{\alpha \in \mathcal{A}} \varphi_\alpha(\xi, l)$$

to indicate the overall contribution to the transition. The extraction of the generating functions from the PEPA model usually presents very little computational challenge because the environment collected via the inference rules in our operational semantics abstracts away from the (potentially very large) actual population levels of the system under study. Using terminology and

notation from Kurtz (cfr. [6]), from $\varphi(\xi, l)$ it is possible to construct a vector field $F : \mathbb{R}^d \to \mathbb{R}^d$ defined as

$$F(x) = \sum_{l \in \mathbb{Z}^d} l\varphi(x, l) \tag{10}$$

and an associated ODE

$$\frac{dx(t)}{dt} = F(x(t)). \tag{11}$$

This formulation makes it possible to establish a property of asymptotic convergence for PEPA models. The result used here states that the solution to a properly defined initial value problem with (11) is the fluid limiting behaviour of a family of CTMCs in the sense of the following theorem.

Theorem 1 (cfr. [6], Theorem 3.1)**.** *Let $\{X_n(t)\}$ be a family of* density dependent *CTMCs, i.e., a sequence of chains with parameter $n \in \mathbb{N}$ taking values in $\mathbb{Z}^d$ such that the infinitesimal generator entries for $X_n(t)$, denoted by $q_{\xi,\xi+l}$, can be described as*

$$q_{\xi,\xi+l} = n\,\varphi(\xi/n, l). \tag{12}$$

Suppose that:

1) *The functions $\varphi(x, l)$ are continuous.*
2) *There exists an open set $E \subset \mathbb{R}^d$ and a constant $L_E \in \mathbb{R}$ such that:*
   a) $\|F(x) - F(y)\| < L_E \|x - y\|, \quad x, y \in E$
   b) $\sup_{x \in E} \sum_{l \in \mathbb{Z}^d} \|l\| \varphi(x, l) < \infty$
   c) $\lim_{k \to \infty} \sup_{x \in E} \sum_{\|l\| > k} \|l\| \varphi(x, l) = 0$

Then, for every solution to the initial value problem of (11) subject to

$$x(0) = \delta \quad \textit{and} \quad x(t) \in E, \quad 0 \le t \le T$$

the family $\{X_n(t)\}$ converges to $x(t)$ in the sense that

$$\lim_{n \to \infty} X_n(0)/n = \delta \Longrightarrow$$
$$\forall \varepsilon > 0 \lim_{n \to \infty} \mathbb{P}\left(\sup_{t \le T} \|X_n(t)/n - x(t)\| > \varepsilon \right) = 0. \tag{13}$$

2.4 Motivating Example

Let us now use Example 1 to illustrate the rationale behind the approach and give an intuitive interpretation of the result of convergence. The parametric rates are obtained by reducing $System_1$ to a much smaller model component $System_1'$ which disregards the information about the multiplicities of the replicated components:

$$System_1' \stackrel{def}{=} P \underset{\{\alpha\}}{\bowtie} Q$$

The sequential components P and Q in this equation are not interpreted as single entities, but as representatives of classes of behaviour. The state descriptor in the NVF is formed by computing the local derivatives of each sequential component in $System_1'$ — this procedure is of negligible computational cost because the behaviour of such components is usually simple, and the state space growth arises from the interleaving of their concurrent behaviours.

The symbolic population-dependent transitions are inferred from the individual transition rates of a single component. The operational semantics gives the following generating function for α:

$$\varphi_\alpha(\xi, (-1, 1, -1, 1)) = \min(p\xi_1, q\xi_3) \tag{14}$$

which intuitively means: if there are ξ_1 components P and ξ_3 components Q, each being able to perform the shared action at rate p and q, respectively, then the overall rate of execution for a shared activity is the minimum (by the cooperation rule) of the two rates at which the action can be performed by the populations of the synchronising components (by additivity of apparent rate calculation). This generating function can be used to derive the transition in (9) for $\xi = (N_P, 0, N_Q, 0)$. Similarly, since P' and Q' are independent, their behaviour is described by the generating functions

$$\varphi_\beta(\xi, (1, -1, 0, 0)) = p'\xi_2 \tag{15}$$

and

$$\varphi_\gamma(\xi, (0, 0, 1, -1)) = q'\xi_4. \tag{16}$$

The non-zero elements of the jump vector indicate which classes are involved in the transition. With regard to the shared actions, all sequential components are subjected to change in their population levels, because of the transitions of the single components (3) and (4) which record a decrease of P and Q and a corresponding increase of P' and Q'.

The generating functions (14), (15) and (16) are used to compute the vector field

$$F(x) = \sum_{l \in \mathbb{Z}^d} l\varphi(x, l) = (-1, 1-1, 1)\min(p\,x_1, q\,x_3) + (1, -1, 0, 0)p'x_2 + (0, 0, 1, -1)q'x_4$$

The associated ODE model (11), in components, is

$$\begin{aligned}
\frac{dx_1(t)}{dt} &= -\min\big(p\,x_1(t), q\,x_3(t)\big) + p'x_2(t) \\
\frac{dx_2(t)}{dt} &= \min\big(p\,x_1(t), q\,x_3(t)\big) - p'\,x_2(t) \\
\frac{dx_3(t)}{dt} &= -\min\big(p\,x_1(t), q\,x_3(t)\big) + q'\,x_4(t) \\
\frac{dx_4(t)}{dt} &= \min\big(p\,x_1(t), q\,x_3(t)\big) - q'x_4(t)
\end{aligned} \tag{17}$$

The intuition behind this approach can be built by comparing, for example, the generating function (14) and the summands of (17) of kind $\min\big(p\,x_1(t), q\,x_3(t)\big)$. Eq. (14) states that, in any given state of the chain, there is a unitary decrease in the population levels of components P and Q every $1/\min\big(p\,\xi_1, q\,\xi_3\big)$ time units on average. It is possible to approximate such a discrete change in a continuous fashion. Letting $x(t)$ be the state descriptor in this continuous state-space, the change in the population count of P over a finite time interval Δt is

$$x_1(t + \Delta t) = x_1(t) - \min\big(p\,x_1(t), q\,x_3(t)\big)\Delta t.$$

Rearranging this equation and taking the limit $\Delta t \to 0$ gives the ODE

$$\frac{dx_1}{dt} = -\min\big(p\,x_1(t), q\,x_3(t)\big).$$

This equation gives only a partial account of the system dynamics (and it corresponds to the first summand of $dx_1(t)/dt$ in (17)). A similar equation may be written for $x_3(t)$, corresponding to the first summand of $dx_3(t)/dt$ in (17). The increase in the population counts of P' and Q' by the same value is expressed with the same rate being present in $dx_2(t)/dt$ and $dx_4(t)/dt$, but with opposite sign. The other summands in (17) may be obtained from analogous interpretations of the generating functions (15) and (16).

A family of CTMCs $\{X_n(t)\}$ can be systematically associated with a PEPA model by taking a density vector, denoted by $\delta \in \mathbb{N}_0^d$, which is interpreted as giving the relative proportions between the distinct sequential components. By letting $\delta = (N_P, 0, N_Q, 0)$, the sequence of CTMCs is such that the initial population levels are multiples of δ, i.e.,

$$X_n(0) = n\delta, \quad \text{for all } n.$$

This corresponds to increasingly large initial population levels as a function of n. For instance, $X_1(t)$ represents the original aggregated CTMC, $X_2(t)$ is the CTMC underlying the model with initial state $P[2N_P] \underset{\{\alpha\}}{\bowtie} Q[2N_Q]$, and so on. Since by construction $\lim_{n\to\infty} X_n(0)/n = \delta$, the result of convergence (13) intuitively states that, asymptotically, a sample path of the CTMC $X_n(t)$ may be well approximated by $n\,x(t)$, over any finite time interval, where $x(t)$ is the solution to the initial value problem of the ODE (17) with $x(0) = \delta$. A pictorial representation of this result is given in Fig. 1, which shows that the ODE is a closer approximation to sample paths of $X_n(t)/n$ for increasingly large n, with excellent accuracy at $n = 1000$.

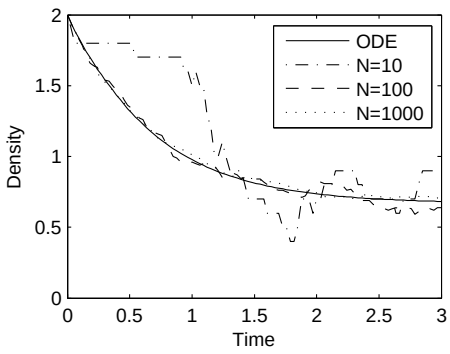

Fig. 1. $x_1(t)$, i.e., the solution to the ODE (17) giving the time-course evolution of the density of component P in Example 1. One realisation of the scaled Markov chain $X_n(t)/n$ over the first three time units becomes closer to the deterministic estimate as n increases. Parameter set: $p = 1.0, p' = 0.5, q = 2.0, q' = 4.0, \delta = (2, 0, 1, 0)$.

3 Population-Based Semantics

3.1 Preliminary Definitions

The interpretation of a PEPA model against the population-based structured operational semantics begins with considering a system equation which does not record the multiplicities of independent replicated sequential components. Any PEPA component may be compacted in such a way. Here we use isomorphism to establish whether two distinct sequential components are equivalent — this notion requires that the derivation graphs of the two components be equal.

Definition 2 (Reduced Context)**.** *The* reduced context *of a PEPA component P, denoted by $red(P)$, is recursively defined as follows:*

$$\begin{aligned}
red\left((\alpha, r).P\right) &= (\alpha, r).P \\
red\left(P + Q\right) &= P + Q \\
red(A \stackrel{\text{def}}{=} P) &= red(P) \\
red(P \underset{L}{\bowtie} P') &= \begin{cases} red(P), \\ \quad \textit{if } L = \emptyset \wedge P = P' \\ \quad \wedge P, P' \textit{ are sequential components} \\ red(P) \underset{L}{\bowtie} red(P'), \textit{otherwise} \end{cases} \\
red(P/L) &= red(P)/L
\end{aligned}$$

The reduced context considers one representative single sequential component P in place of the cooperation $P \parallel P'$ if the two cooperating processes are isomorphic sequential components. Thus, because of this equivalence relation between these components, the first case for the cooperation operator in Definition 2 could also read $red(P')$. Clearly, the two arrays $P[N_P]$ and $Q[N_Q]$ in Example 1 are recursively reduced to single sequential components P and Q, respectively and

$$red(System_1) = P \underset{\{\alpha\}}{\bowtie} Q, \tag{18}$$

as illustrated above. Notice that the same context reduction (18) would be obtained if the system equation was replaced with

$$\big(P[N_P - K_P] \,\|\, P'[K_P]\big) \underset{\{\alpha\}}{\bowtie} \big(Q[N_Q - K_Q] \,\|\, Q'[K_Q]\big), \tag{19}$$

for any $1 \le K_P \le N_P$ and $1 \le K_Q \le N_Q$. Indeed the $(N_P - K_P)$ P components would be reduced to P as before. Furthermore, the cooperation $P \parallel P'$ would be reduced to P as well, since P and P' are isomorphic because they are two local derivatives of the same sequential component. Similar arguments hold for the isomorphism between Q and Q'. Therefore, the two model equations will give rise to the same underlying ODE although with two different initial value problems, as determined by the population levels specified in the equations.

It is worthwhile pointing out that Definition 2 also allows for two or more instances of a sequential component to appear in the reduced context of a PEPA model. For example, we have that

$$red\big(P[N_P] \underset{\{\alpha\}}{\bowtie} Q[N_Q] \parallel P[N'_P]\big) = P \underset{\{\alpha\}}{\bowtie} Q \parallel P$$

This supports the intuitive observation that the leftmost array of P components will behave differently from the rightmost array. In this instance, the action α of the former array is executed in cooperation with a Q component, whereas it is an independent action with regard to the latter array because of the empty cooperation set.

In the remainder of this paper we consider a PEPA model for which the reduced context, hereinafter denoted by $\mathcal{M}$, is already known. This minimal form contains the necessary information to determine the state descriptor in NVF, and is analogous to a Petri net without any marking.

Definition 3 (Numerical Vector Form)**.** *Let N_C be the number of distinct sequential components in $\mathcal{M}$. Let $\mathcal{C}_i$ be the derivative set of the i-th component, $i = 1, 2, \ldots, N_C$ and let N_i be its size, i.e., $N_i = |\mathcal{C}_i|$. Let $C_{i,j}$ denote the j-th derivative of the i-th component, $j = 1, 2, \ldots, N_i$. The state descriptor in the NVF, denoted by $\xi \in \mathbb{N}_0^d, d = \sum_{i=1}^{N_C} N_i$, assigns a coordinate, denoted by $\xi_{i,j}$, to each local derivative $C_{i,j}$ and indicates the number of copies in the system which exhibit that derivative.*

Definition 4 (Initial State of the CTMC)**.** *The initial state of the CTMC is denoted by $\delta \in \mathbb{N}_0^d$ and gives an initial population level $\delta_{i,j} \geq 0$ to each local derivative $C_{i,j}$. Without loss of generality we exclude the case in which all the derivatives of a sequential component are set to 0, by subjecting δ to the condition $\sum_{k=1}^{N_i} \delta_{i,k} > 0$, for all i.*

Sometimes the element $\xi_{i,j}$ is conveniently referred to by a single subscript ξ_k, i.e., we assume an implicit mapping of each sequential component $C_{i,j}$ to a coordinate $1 \leq k \leq d$ in the population vector. For instance, with regard to Example 1, $N_C = 2$, $\mathcal{C}_1 = \{P, P'\}$, and $\mathcal{C}_2 = \{Q, Q'\}$. Furthermore, we use the following mappings: $C_{1,1} \mapsto P$, $C_{1,2} \mapsto P'$, $C_{2,1} \mapsto Q$, $C_{2,2} \mapsto Q'$. Assuming the same ordering as in (9), the initial state in Example 1 is $(N_P, 0, N_Q, 0)$ whereas it is $(N_P - K_P, K_P, N_Q - K_Q, K_Q)$ in (19).

As with the Markovian interpretation, at the core of this semantics is the notion of apparent rate. Here this concept is modified to take into account the interpretation of the reduced context described above.

Definition 5 (Parametric Apparent Rate)**.** *Consider a process P composed of sequential components $C_{i,j}$. The* parametric apparent rate *of action type α in component P, denoted by $r_\alpha^\star(P, \xi)$, defines the overall rate at which the action type α can be performed by component P as a function of the population sizes ξ of the sequential components of the system:*

$$r_\alpha^\star\left(P \underset{L}{\bowtie} Q, \xi\right) = \begin{cases} \min\left(r_\alpha^\star(P,\xi), r_\alpha^\star(Q,\xi)\right) & \text{if } \alpha \in L \\ r_\alpha^\star(P,\xi) + r_\alpha^\star(Q,\xi) & \text{if } \alpha \notin L \end{cases}$$

$$r_\alpha^\star(P/L, \xi) = \begin{cases} r_\alpha^\star(P,\xi) & \text{if } \alpha \notin L \\ 0 & \text{if } \alpha \in L \end{cases}$$

$$r_\alpha^\star(C_{i,j}, \xi) = \textstyle\sum_{k=1}^{N_i} r_\alpha(C_{i,k})\xi_{i,k}$$

The first two cases are structurally and syntactically similar to their counterparts in the Markovian semantics, $r_\alpha(P \underset{L}{\bowtie} Q)$ and $r_\alpha(P/L)$. For a sequential component of the reduced context, the definition of parametric apparent rate exploits the property in (1) that it can be expressed as the product of the population size and the apparent rate of a single sequential component. In addition, the behaviour of the other derivatives in the same derivative set of $C_{i,j}$ is taken into account because of the interpretation of $\mathcal{M}$. As already discussed, each sequential component in $\mathcal{M}$ represents an array of identical components, evolving through the local derivatives $C_{i,k}$, $1 \leq k \leq N_i$. In any state of the CTMC there may be one or more components exhibiting each such derivative. These components will *compete* to participate in a shared action α, and the probability that the action is completed by each derivative will be proportional to the population level of that derivative and the individual rate of execution. Thus, the apparent rate calculated in this manner reflects the potential contribution to the action by any concurrent sequential component. As observed in (2) the summation is legitimate due to the property of additivity which holds for the apparent rates for non-cooperating components.

The set of functions generated by $r_\alpha^\star(\cdot, \xi)$ is denoted by $\mathcal{F} = [\mathbb{R}^d \longrightarrow \mathbb{R}_{\geq 0}]$, a function space with values in the non-negative reals because passive actions are not allowed.

3.2 Structured Operational Semantics

The population-based parametric structured operational semantics for PEPA is shown in Table 3. Let $\mathcal{C}$ be the set of PEPA processes composed by $C_{i,j}$. Let $\mathcal{L}$ be the labelling alphabet, i.e., $\mathcal{L} = \mathcal{A} \times \mathcal{F}$. The rules induce a *parametric multi-transition system*, $(\mathcal{C}, \mathcal{L}, \rightarrow_\star), \rightarrow_\star \subseteq \mathcal{C} \times \mathcal{L} \times \mathcal{C}$, which records the multiplicity of a transition between two components. As with the Markovian semantics of PEPA, this requirement is necessary in order to calculate the transition rates correctly.

The rule for sequential components $\mathrm{S}_0^\star$ constructs the relationship between the two semantics. The premise is a transition of the Markovian semantics for a single sequential component. By construction of $\mathcal{C}$ the right hand side of the transition is in the same derivative set, i.e., $C_{i,j} \xrightarrow{(\alpha,r)} C_{i',j'} \Rightarrow i = i'$. Such a transition is said to be *promoted* to an inference for the population-based semantics — the premise describes the behaviour

TABLE 3
Population-based parametric structured operational semantics of PEPA. Transitions are denoted by the symbol $\longrightarrow_\star$ to distinguish them from the Markovian transitions in PEPA which carry reals instead of functions.

Sequential Component (Promotion Rule)

$$\mathrm{S}_0^\star : \frac{C_{i,j} \xrightarrow{(\alpha,r)} C_{i,j'} \quad C_{i,j} \in \mathcal{C}_i}{C_{i,j} \xrightarrow{(\alpha, r\xi_{i,j})}_\star C_{i,j'}}$$

Cooperation

$$\mathrm{C}_0^\star : \frac{P \xrightarrow{(\alpha,r(\xi))}_\star P'}{P \underset{L}{\bowtie} Q \xrightarrow{(\alpha,r(\xi))}_\star P' \underset{L}{\bowtie} Q}, \alpha \notin L$$

$$\mathrm{C}_1^\star : \frac{Q \xrightarrow{(\alpha,r(\xi))}_\star Q'}{P \underset{L}{\bowtie} Q \xrightarrow{(\alpha,r(\xi))}_\star P \underset{L}{\bowtie} Q'}, \alpha \notin L$$

$$\mathrm{C}_2^\star : \frac{P \xrightarrow{(\alpha,r_1(\xi))}_\star P' \qquad Q \xrightarrow{(\alpha,r_2(\xi))}_\star Q'}{P \underset{L}{\bowtie} Q \xrightarrow{(\alpha,r(\xi))}_\star P' \underset{L}{\bowtie} Q'}, \alpha \in L$$

$$\text{and} \quad r(\xi) = \frac{r_1(\xi)}{r_\alpha^\star(P,\xi)} \frac{r_2(\xi)}{r_\alpha^\star(Q,\xi)} \min\left(r_\alpha^\star(P,\xi), r_\alpha^\star(Q,\xi)\right)$$

Hiding

$$\mathrm{H}_0^\star : \frac{P \xrightarrow{(\alpha,r(\xi))}_\star P'}{P/L \xrightarrow{(\alpha,r(\xi))}_\star P'/L}, \alpha \notin L$$

$$\mathrm{H}_1^\star : \frac{P \xrightarrow{(\alpha,r(\xi))}_\star P'}{P/L \xrightarrow{(\tau,r(\xi))}_\star P'/L}, \alpha \in L$$

Constant

$$\mathrm{A}_0^\star : \frac{P \xrightarrow{(\alpha,r(\xi))}_\star P'}{A \xrightarrow{(\alpha,r(\xi))}_\star P'}, A \stackrel{def}{=} P$$

of a single sequential component, whereas the conclusion gives the collective dynamics of the population of components $C_{i,j}$. This population evolves at an overall rate which is the product of the individual rate and the number of components exhibiting this local derivative.

The other rules are syntactically similar to their counterparts in the Markovian semantics. However, in all cases the derivations carry as rates functions of $\mathcal{F}$ instead of reals. The following derivation tree gives a transition for the shared activity with regard to the reduced context of Example 1.

$$\frac{\dfrac{P \xrightarrow{(\alpha,p)} P'}{P \xrightarrow{(\alpha,p\xi_{1,1})}_\star P'}\mathrm{S}_0^\star \quad \dfrac{Q \xrightarrow{(\alpha,q)} Q'}{Q \xrightarrow{(\alpha,q\xi_{2,1})}_\star Q'}\mathrm{S}_0^\star}{P \underset{\{\alpha\}}{\bowtie} Q \xrightarrow{\left(\alpha,\min\left(p\xi_{1,1},q\xi_{2,1}\right)\right)}_\star P' \underset{\{\alpha\}}{\bowtie} Q'}\mathrm{C}_2^\star \tag{20}$$

The following two examples present cases which could not be handled by the deterministic interpretation of [8]. The rules for cooperation can be used to derive the rate for shared actions which can be performed by two distinct local derivatives of the same sequential component, as shown by P in the following

Example 2 (Distinct local states enabling the same activity type)**.**

$$\begin{array}{lcll} \xi_{1,1} & P & \stackrel{\text{def}}{=} & (\alpha,p).P' \\ \xi_{1,2} & P' & \stackrel{\text{def}}{=} & (\beta,p').P'' \\ \xi_{1,3} & P'' & \stackrel{\text{def}}{=} & (\alpha,p'').P \\ \xi_{2,1} & Q & \stackrel{\text{def}}{=} & (\alpha,q).Q' \\ \xi_{2,2} & Q' & \stackrel{\text{def}}{=} & (\gamma,q').Q \\ & System_2 & \stackrel{\text{def}}{=} & P[N_P] \underset{\{\alpha\}}{\bowtie} Q[N_Q] \end{array}$$

(Alongside the process definitions are the corresponding coordinates in the population vector.) The local derivatives P and P'' perform the shared action at parametric rate $\xi_{1,1}p$ and $\xi_{1,3}p''$, respectively. Similarly, the parametric rate for Q is $\xi_{2,1}q$. Rule $\mathrm{C}_2^\star$ says that each local state evolves at a rate which is weighted by their relative probabilities of execution, i.e., $\xi_{1,1}p/(\xi_{1,1}p + \xi_{1,3}p'')$ and $\xi_{1,3}p''/(\xi_{1,1}p + \xi_{1,3}p'')$.

Rules $\mathrm{C}_0^\star$ and $\mathrm{C}_1^\star$ allow two distinct sequential components not to cooperate over the set of shared action types, as illustrated by the following example.

Example 3 (Implicit Choice)**.**

$$\begin{array}{lcll} \xi_{1,1} & P & \stackrel{\text{def}}{=} & (\alpha,p).P' \\ \xi_{1,2} & P' & \stackrel{\text{def}}{=} & (\beta,p').P \\ \xi_{2,1} & R & \stackrel{\text{def}}{=} & (\alpha,r).R' \\ \xi_{2,2} & R' & \stackrel{\text{def}}{=} & (\delta,r').R \\ \xi_{3,1} & Q & \stackrel{\text{def}}{=} & (\alpha,q).Q' \\ \xi_{3,2} & Q' & \stackrel{\text{def}}{=} & (\gamma,q').Q \\ & System_3 & \stackrel{\text{def}}{=} & \left(P[N_P] \parallel R[N_R]\right) \underset{\{\alpha\}}{\bowtie} Q[N_Q] \end{array}$$

Components P and R may both perform an activity of type α, although the system equation does not enforce synchronisation between them because their cooperation set is empty. In our semantics, two deduction trees for α can be inferred which represent the interactions between components P and Q, and R and Q. The deduction tree for the interaction between P and Q is:

$$\frac{\dfrac{\dfrac{P \xrightarrow{(\alpha,p)} P'}{P \xrightarrow{(\alpha,p\xi_{1,1})}_\star P'}\mathrm{S}_0^\star}{P \parallel R \xrightarrow{(\alpha,p\xi_{1,1})}_\star P' \parallel R}\mathrm{C}_0^\star \quad \dfrac{Q \xrightarrow{(\alpha,q)} Q'}{Q \xrightarrow{(\alpha,q\xi_{3,1})}_\star Q'}\mathrm{S}_0^\star}{(P \parallel R) \underset{\{\alpha\}}{\bowtie} Q \xrightarrow{(\alpha,r'(\xi))}_\star (P' \parallel R) \underset{\{\alpha\}}{\bowtie} Q'}\mathrm{C}_2^\star,$$

where

$$\begin{aligned} r'(\xi) &= \frac{p\xi_{1,1}}{r_\alpha^\star(P \parallel R,\xi)} \frac{q\xi_{3,1}}{r_\alpha^\star(Q,\xi)} \min\left(r_\alpha^\star(P \parallel R,\xi), r_\alpha^\star(Q,\xi)\right) \\ &= \frac{p\xi_{1,1}}{p\xi_{1,1} + r\xi_{2,1}} \min\left(p\xi_{1,1} + r\xi_{2,1}, q\xi_{3,1}\right) \end{aligned}$$

The deduction tree for the transition

$$(P \parallel R) \underset{\{\alpha\}}{\bowtie} Q \xrightarrow{(\alpha,r''(\xi))}_\star (P \parallel R') \underset{\{\alpha\}}{\bowtie} Q'$$

can be similarly inferred in the obvious way, where

$$r''(\xi) = \frac{p\xi_{2,1}}{p\xi_{1,1} + r\xi_{2,1}} \min\left(p\xi_{1,1} + r\xi_{2,1}, q\xi_{3,1}\right)$$

Notice that $r'(\xi)+r''(\xi) = \min\left(p\xi_{1,1} + r\xi_{2,1}, q\xi_{3,1}\right)$, which represents the total activity rate for α.

3.3 Parametric Derivation Graph

All the inference trees presented in the previous section are concerned with the derivation of transitions from the initial state $\mathcal{M}$. However, this information is not sufficient to obtain the behaviour of the entire system under consideration, because the derivatives of the initial state under the Markovian semantics only give the first-step behaviour of the process. The collective behaviour of the system is represented by the notions of derivative set and derivation graph of $\mathcal{M}$ in the population-based semantics, which are defined in a similar way to their counterparts in the Markovian semantics.

Definition 6 (Parametric Derivative Set)**.** *The* parametric derivative set *of $\mathcal{M}$, denoted by $ds^\star(\mathcal{M})$, is the smallest set of PEPA components which satisfies the following conditions:*

- $\mathcal{M} \in ds^\star(\mathcal{M})$
- *If $P \in ds^\star(\mathcal{M})$ and there exists $P \xrightarrow{(\alpha, r(\xi))}_\star P'$ then $P' \in ds^\star(\mathcal{M})$*

Notice that the indicator function can be applied to each $P \in ds^\star(\mathcal{M})$ because it is a composition through the combinators of PEPA of sequential components $C_{i,j}$, each of which has the coordinate (i, j) in the NVF by Definition 3. We use the following notion of *indicator function* to obtain the local states exhibited by a derivative in $ds^\star(\mathcal{M})$.

Definition 7 (Indicator Function)**.** *Let $1_{i,j} \in \mathbb{N}_0^d$ denote a vector whose elements are all zero except for the coordinate corresponding to the derivative $C_{i,j}$, which is set to one. Let $P \in ds^\star(\mathcal{M})$. The* indicator *of P, denoted by $ind(P)$, returns a vector whose non-zero elements correspond to the indices in the population vector of the sequential components in P. It is defined as follows:*

$$\begin{aligned} ind(C_{i,j}) &= 1_{i,j} \\ ind(A \stackrel{\text{def}}{=} P) &= ind(P) \\ ind(P \underset{L}{\bowtie} Q) &= ind(P) + ind(Q) \\ ind(P/L) &= ind(P) \end{aligned}$$

For instance, in Example 1 we have that $ind(P \underset{\{\alpha\}}{\bowtie} Q) = (1, 0, 1, 0)$.

The derivative set $ds^\star(\mathcal{M})$ is of crucial importance for the development of the population-based semantics. Each derivative $P \in ds^\star(\mathcal{M})$ identifies a specific kind of behaviour, i.e., the interactions amongst the sequential components when they exhibit the local states indicated by $ind(P)$. For instance, in Example 1 the semantics will give transitions for the generic state

$$\left(P[\xi_{1,1}] \parallel P'[\xi_{1,2}]\right) \underset{\{\alpha\}}{\bowtie} \left(Q[\xi_{2,1}] \parallel Q'[\xi_{2,2}]\right) \tag{21}$$

although the component $P \underset{\{\alpha\}}{\bowtie} Q$ subsumes information only about the transitions between the $\xi_{1,1}$ components in state P and the $\xi_{2,1}$ components in state Q. As observed above (cfr. (8)), the transition between each such pair of sequential components can be expressed parametrically as a function of their population levels and the behaviour of the individual sequential components involved. The other kinds of behaviour which are simultaneously enabled by (21) are obtained by the other elements of $ds^\star(\mathcal{M})$.

In Example 1, the inference tree in (20) implies $P' \underset{\{\alpha\}}{\bowtie} Q' \in ds^\star(\mathcal{M})$. The transitions from this component are concerned with the interactions between the $\xi_{1,2}$ components exhibiting state P' and the $\xi_{2,2}$ components in state Q'. These can be obtained from the following two inference trees:

$$\frac{\dfrac{P' \xrightarrow{(\beta, p')} P}{P' \xrightarrow{(\beta, p'\xi_{1,2})}_\star P}}{P' \underset{\{\alpha\}}{\bowtie} Q' \xrightarrow{(\beta, p'\xi_{1,2})}_\star P \underset{\{\alpha\}}{\bowtie} Q'} \tag{22}$$

$$\frac{\dfrac{Q' \xrightarrow{(\gamma, q')} Q}{Q' \xrightarrow{(\gamma, q'\xi_{2,2})}_\star Q}}{P' \underset{\{\alpha\}}{\bowtie} Q' \xrightarrow{(\gamma, q'\xi_{2,2})}_\star P' \underset{\{\alpha\}}{\bowtie} Q} \tag{23}$$

The construction of the parametric derivative set is completed by the inference of the transitions for $P \underset{\{\alpha\}}{\bowtie} Q'$ and $P' \underset{\{\alpha\}}{\bowtie} Q$:

$$\frac{\dfrac{Q' \xrightarrow{(\gamma, q')} Q}{Q' \xrightarrow{(\gamma, q'\xi_{2,2})}_\star Q}}{P \underset{\{\alpha\}}{\bowtie} Q' \xrightarrow{(\gamma, q'\xi_{2,2})}_\star P \underset{\{\alpha\}}{\bowtie} Q} \tag{24}$$

$$\frac{\dfrac{P' \xrightarrow{(\beta, p')} P}{P' \xrightarrow{(\beta, p'\xi_{1,2})}_\star P}}{P' \underset{\{\alpha\}}{\bowtie} Q \xrightarrow{(\beta, p'\xi_{1,2})}_\star P \underset{\{\alpha\}}{\bowtie} Q} \tag{25}$$

Finally, the notion of *parametric derivation graph* encompasses the complete behaviour of the system.

Definition 8 (Parametric Derivation Graph)**.** *Given a parametric derivative set $ds^\star(\mathcal{M})$, the* parametric derivation graph *of $\mathcal{M}$, denoted by $\mathcal{D}^\star(\mathcal{M})$ is a labelled directed multi-graph (V, A) with vertices $V \in ds^\star(\mathcal{M})$ and arcs $A \in ds^\star(\mathcal{M}) \times \mathcal{L} \times ds^\star(\mathcal{M})$ where the number of occurrences of an arc, denoted by m, is equal to the number of distinct inference trees for a transition.*

The inference trees (20), (22), (23), (24), and (25) give rise to the parametric derivation graph depicted in Fig. 2 (each arc has multiplicity one).

3.4 Extraction of the Generating Functions

The arcs of the parametric derivation graph can be used to construct the generating functions of the underlying

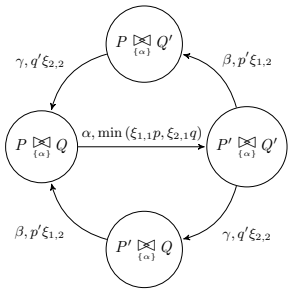

Fig. 2. Parametric derivation graph of Example 1.

population-based CTMC, as straightforwardly as the derivation graph in the original semantics gives rise to the underlying Markov process. An arc $P \xrightarrow{(\alpha, r(\xi))}_\star P' \in A$ implies a generating function in the form $\varphi_\alpha(\xi, l) = m \cdot r(\xi)$, where m is the multiplicity of the arc and the jump vector l indicates the sequential components whose population levels change due to the transition. The jump vector is taken from the inspection of the source and target components of the transition. The population levels of sequential components in the source component are subjected to a decrease by one (cfr. (9)). Correspondingly, the population levels in the target component are increased by the same quantity. This is captured by the following definition.

Definition 9 (Extraction of the Generating Functions)**.** *Let $\mathcal{M}$ be a PEPA model with parametric derivative graph $\mathcal{D}^\star(\mathcal{M})$. The generating functions of the underlying population-based CTMC are as follows:*

$$\varphi_\alpha(\xi, l) = \begin{cases} m \cdot r(\xi) & \text{if } \exists\ P \xrightarrow{(\alpha, r(\xi))}_\star P' \in A \text{ and} \\ & \quad l = 0^d - ind(P) + ind(P') \\ 0 & \text{otherwise} \end{cases}$$

where 0^d is the zero-vector in $\mathbb{Z}^d$.

It is possible to verify that the generating functions derived according to this definition coincide with those formulated in (14–16) for Example 1. Notice that two distinct transitions in the parametric derivation graph may give rise to the same generating functions. For instance, (22) and (25) imply the generating function (15). However, both transitions express the same kind of behaviour, i.e., the possibility for components of kind P' to perform action β, regardless of the states of the components in the right hand side of the cooperation. As discussed in Section 2 the fact that the components exhibiting states Q and Q' are not involved in this transition is reflected by their corresponding elements in the jump vector being equal to zero. This property emerges from the calculation of the jump vector in Definition 9, as any sequential component which is present in both sides of a transition is such that the negative entry -1 (due to the presence in the lhs) cancels out the positive entry $+1$ (due to the presence in the rhs) in the component's corresponding coordinate. (A similar remark can be applied to the symmetric case of (23) and (24), which define the same function (16).)

4 Fluid Limit of the CTMC

This section is concerned with verifying that the population-based semantics satisfies the conditions of Theorem 1.

4.1 Density Dependency

In order to prove (12) we begin by proving the following property for parametric apparent rates.

Lemma 1. *Let $r^\star_\alpha(P, \xi)$ be the parametric apparent rate of action type α in process P. For any $n \in \mathbb{N}$ and $\alpha \in \mathcal{A}$,*

$$r^\star_\alpha(P, \xi) = n \cdot r^\star_\alpha(P, \xi/n)$$

Proof: We proceed by structural induction over Definition 5. For the base case, we have that

$$r^\star_\alpha(C_{i,j}, \xi) = \sum_{k=1}^{N_i} r_\alpha(C_{i,k})\xi_{i,k} = n \sum_{k=1}^{N_i} r_\alpha(C_{i,k})\xi_{i,k}/n = \\ = n \cdot r^\star_\alpha(P, \xi/n)$$

The inductive step follows by observing that density dependency is preserved by the functions $\min$ and summation. □

This lemma is used to prove that the same property is enjoyed by the parametric rates which label the transitions in the new semantics.

Lemma 2. *If $P \xrightarrow{(\alpha, r(\xi))}_\star Q$ then, for any $n \in \mathbb{N}$, $r(\xi) = n \cdot r(\xi/n)$*

Proof: We prove this by structural induction over the structured operational semantics in Table 3. The base case $S^\star_0$ is obvious. The less straightforward case is that of rule $C^\star_2$ where the rate function does not carry over to the conclusion. Combining the induction hypothesis on $r_1(\xi)$ and $r_2(\xi)$ and the previous lemma for $r^\star_\alpha(P, \xi)$ and $r^\star_\alpha(Q, \xi)$,

$$\begin{aligned} r(\xi) &= \frac{r_1(\xi)}{r^\star_\alpha(P, \xi)} \frac{r_2(\xi)}{r^\star_\alpha(Q, \xi)} \min\left(r^\star_\alpha(P, \xi), r^\star_\alpha(Q, \xi)\right) \\ &= \frac{n \cdot r_1(\xi/n)}{n \cdot r^\star_\alpha(P, \xi/n)} \frac{n \cdot r_2(\xi/n)}{n \cdot r^\star_\alpha(Q, \xi/n)} \\ &\quad \cdot \min\left(n \cdot r^\star_\alpha(P, \xi/n), n \cdot r^\star_\alpha(P, \xi/n)\right) \\ &= \frac{r_1(\xi/n)}{r^\star_\alpha(P, \xi/n)} \frac{r_2(\xi/n)}{r^\star_\alpha(Q, \xi/n)} \\ &\quad \cdot n \cdot \min\left(r^\star_\alpha(P, \xi/n), r^\star_\alpha(P, \xi/n)\right) = nr(\xi/n) \end{aligned}$$

□

Observing that $\varphi(x, l)$ is a summation of functions which satisfy the previous lemma, the following proposition holds.

Proposition 1. *Let $\mathcal{M}$ be a PEPA model with generating functions $\varphi(x, l)$ derived according to Definition 9. The elements of the generator matrix are such that they verify (12).*

4.2 Lipschitz Continuity

Observing that Lipschitz continuity is preserved by summation, in order to verify that the vector field (10) is Lipschitz it suffices to prove that any parametric rate generated by the semantics is Lipschitz. This property will be used to satisfy condition 2a) of Theorem 1. As with density dependence, we check that the property holds for apparent rates.

Lemma 3. *Let $r_\alpha^\star(P,\xi)$ be the parametric apparent rate of action type α in process P. There exists a constant $L \in \mathbb{R}$ such that for all $x, y \in \mathbb{R}^d, x \neq y$,*

$$\frac{\|r_\alpha^\star(P,x) - r_\alpha^\star(P,y)\|}{\|x-y\|} \leq L$$

Proof: This is proven by using the supremum norm $\|x\| = \max_i |x_i|$ and structural induction over the Definition 5.
Base case

$$\begin{aligned}\|r_\alpha^\star(C_{i,j},x) - r_\alpha^\star(C_{i,j},y)\| &= \left\|\sum_{k=1}^{N_i} r_\alpha(C_{i,k})(x_{i,k}-y_{i,k})\right\| \\ &\leq \sum_{k=1}^{N_i} r_\alpha(C_{i,k})\,\|x-y\|\end{aligned}$$

Inductive Step
Case $r_\alpha^\star\left(P \underset{L}{\bowtie} Q, \cdot\right) = \min\left(r_\alpha^\star(P,\cdot), r_\alpha^\star(Q,\cdot)\right)$, $\alpha \in L$ follows because the minimum of two Lipschitz functions (by the induction hypothesis) is also Lipschitz.
Case $r_\alpha^\star\left(P \underset{L}{\bowtie} Q, \cdot\right) = r_\alpha^\star(P,\cdot) + r_\alpha^\star(Q,\cdot), \alpha \notin L$. This is Lipschitz with constant $L = L_P + L_Q$, where L_P and L_Q are the Lipschitz constants of P and Q, respectively, which exist by the induction hypothesis.
Case $r_\alpha^\star(P/L,\cdot)$. The function 0 is Lipschitz. The other case follows by the induction hypothesis. □

Lemma 4. *If $P \xrightarrow{(\alpha, r(x))}_\star P'$ then $r(x) \leq r_\alpha^\star(P,x)$*

Proof: We prove this by structural induction. The most interesting case is that of cooperation.
Rule $C_0^\star$ (Rule $C_1^\star$ is symmetric)

$$\begin{aligned}r(x) =& r_1 \leq r_\alpha^\star(P,x) \\ &\leq r_\alpha^\star(P,x) + r_\alpha^\star(Q,x) \equiv r_\alpha^\star\left(P \underset{L}{\bowtie} Q, x\right)\end{aligned}$$

Rule $C_2^\star$

$$\begin{aligned}r(x) =& \frac{r_1(x)}{r_\alpha^\star(P,x)}\frac{r_2(x)}{r_\alpha^\star(Q,x)}\min\left(r_\alpha^\star(P,x), r_\alpha^\star(Q,x)\right) \\ &\leq 1 \cdot 1 \cdot \min\left(r_\alpha^\star(P,x), r_\alpha^\star(Q,x)\right) \equiv r_\alpha^\star\left(P \underset{L}{\bowtie} Q, x\right)\end{aligned}$$

□

By combining Lemma 3 and 4, by structural induction over the semantic rules,

Proposition 2. *If $P \xrightarrow{(\alpha, r(x))}_\star P'$ then $r(x)$ is Lipschitz continuous.*

These results establish that the parametric rates are *globally* Lipschitz in $\mathbb{R}^d$. Thus, in Theorem 1, Condition (1) is satisfied and (2a) holds for any open $E \subset \mathbb{R}^d$.

Theorem 2. *Let $x(t), 0 \leq t \leq T$ satisfy the initial value problem $\dot{x} = F(x(t)), x(0) = \delta$, specified from a PEPA model according to (10) and Definition 4. Let $\{X_n(t)\}$ be a family of CTMCs with parameter $n \in \mathbb{N}$ generated according to Definition 9 and let $X_n(0) = n \cdot \delta$. Then,*

$$\forall \varepsilon > 0 \lim_{n\to\infty} \mathbb{P}\left(\sup_{t\leq T} \|X_n(t)/n - x(t)\| > \varepsilon\right) = 0.$$

Proof: This proof is only sketched here because it relies heavily on the theory developed in [6], which is not fully reported here due to space limitations. This proof is based on verifying that the conditions of Theorem 1 are satisfied for any PEPA model. The main technical difficulty consists in finding an open set $E \subset \mathbb{R}^d$ such that the hypotheses of the theorem are verified, in particular that it holds $x(t) \in E, 0 \leq t \leq T$. However, according to the proof of Theorem 1 provided in [6], this is a sufficient condition to prove that there exists $\eta > 0$ such that for every n

$$E_n \cap \left\{y \in \mathbb{R}^d : \inf_{t\leq T} \|y - x(t)\| \leq \eta\right\} \in E, \qquad (26)$$

where $E_n = \{\xi/n : \xi \text{ is a state of } X_n(t)\}$ (cfr. [6], Theorem 2.11, of which Theorem 1 is a specialisation for density-dependent chains). However, observe that for any PEPA model the trajectory of $X_n(t)$ is bounded because, for all n, the sum of the population levels of all components in all states is constant, in particular, it is equal to the sum of the initial population levels, i.e.,

$$\sum_{k=1}^{d} \xi_k = n \sum_{k=1}^{d} \delta_k$$

Letting $\Delta = \sum_{k=1}^{d} \delta_k$ and recalling that $\xi_i \geq 0, 1 \leq i \leq d$, we have that

$$0 \leq \frac{\xi_i}{n} \leq \frac{\sum_{k=1}^{d} \xi_k}{n} \equiv \Delta$$

Thus, for any PEPA model, (26) is verified for any η if E is the smallest open set which contains $\left\{y \in \mathbb{R}^d : 0 \leq y_i \leq \Delta, 0 \leq i \leq d\right\}$ □

4.3 Discussion

The result of convergence discussed above only holds for models with active synchronisation. The original semantics of PEPA also allows *passive* activities, whose rate is denoted by the symbol $\top$. Informally, the meaning of a passive component is that the rate is determined by some other (active) cooperating component. For instance, replacing the definition of Q with $Q \stackrel{def}{=} (\alpha, \top).Q'$ in Example 1 yields a model in which the rate of α is determined by P only. According to the arithmetic of

passive rates presented in [9], the analogue of (8) in this case is

$$R_{pas} = \frac{p}{N_P p}\frac{\top}{N_Q \top}\min(N_P p, N_Q \top) = \frac{p}{N_Q},$$

which would suggest a similar transition to (9) in the NVF of type

$$(N_P, 0, N_Q, 0) \xrightarrow{(\alpha, N_P p)} (N_P - 1, 1, N_Q - 1, 1) \qquad (27)$$

However, unlike (9), this transition is enabled if $N_Q = 0$, which leads to a meaningless state of the chain because one component is negative. Instead, the presence of passive components can be correctly captured by the following generating function (see also [12] for a similar treatment):

$$\varphi_\alpha(\xi, (-1, 1, -1, 1)) = \begin{cases} \xi_1 p & \text{if } \xi_2 \neq 0 \\ 0 & \text{if } \xi_2 = 0 \end{cases}$$

However, such a function is clearly discontinuous, hence it does not satisfy one condition for the applicability of Kurtz's theorem (in fact, the existence and uniqueness of the solution is not even guaranteed by the condition of Lipschitz continuity on the vector field).

Our semantics can be extended in order to accommodate passive rates. With respect to this example, the strategy consists in using a continuous generating function $\varphi_\alpha(\xi, (-1, 1, -1, 1)) = \xi_1 p$ and defining an *exit time* for the ODE, i.e. by setting T of Theorem 2 as $T = \inf\{t : x_1(t) > 0 \wedge x_3(t) = 0\}$. Thus, solutions to the ODE are accepted until the deterministic process is in such a state that there are active components capable of carrying out the shared actions but there are no cooperating passive components (notice that if $x_1(t) = 0$ the shared activity is not enabled regardless of the population level $x_3(t)$).

Our approach can also incorporate the alternative treatment presented in [13], in which a model with passive cooperation is translated into an equivalent one with active synchronisation, yielding better results with regard to the agreement with the underlying Markov process. Thus, a model with passive synchronisation may be subjected to this transformation process before our deterministic semantics is applied.

5 Case Study

In this section we apply the population-based semantics of PEPA to a more complex PEPA model. We carry out numerical tests to assess the agreement between the deterministic approximation and the stochastic process.

5.1 Three-Tier Distributed Application

The model, shown in Fig. 3, describes a three-tier distributed application. The process definitions prefixed with $Cl:$ indicate the client behaviour, which performs a synchronous request to the system and interposes some thinking time between successive requests. Clients

$$\begin{aligned}
Cl : Request &\stackrel{def}{=} (request, r_{c:request}).Cl : Wait \\
Cl : Wait &\stackrel{def}{=} (reply, r_{c:reply}).Cl : Think \\
Cl : Think &\stackrel{def}{=} (think, r_{c:think}).Cl : Request \\
Sr : Wait &\stackrel{def}{=} (request, p_{fresh} r_{s:request}).Sr : Fresh \\
&\quad + (request, (1 - p_{fresh}) r_{s:request}).Sr : Reply \\
&\quad + (fail, r_{s:fail}).Sr : Repair \\
Sr : Fresh &\stackrel{def}{=} (read, p_{ok} r_{s:read}).Sr : Reply \\
&\quad + (read, (1 - p_{ok}) r_{s:read}).Sr : Force \\
Sr : Force &\stackrel{def}{=} (force, r_{s:force}).Sr : Write \\
Sr : Write &\stackrel{def}{=} (write, r_{s:write}).Sr : Reply \\
Sr : Reply &\stackrel{def}{=} (reply, r_{s:reply}).Sr : Wait \\
Sr : Repair &\stackrel{def}{=} (read, r_{s:read}).Sr : Wait \\
Db : Wait &\stackrel{def}{=} (read, r_{d:read}).Db : Update \\
&\quad + (write, r_{d:write}).Db : Update \\
Db : Update &\stackrel{def}{=} (update, r_{d:update}).Db : Wait \\
Rb : Gather &\stackrel{def}{=} (crawl, r_{r:crawl}).Rb : Write \\
Rb : Write &\stackrel{def}{=} (write, r_{r:write}).Rb : Gather \\
System_{App} &\stackrel{def}{=} Cl : Request[N_c] \underset{\{request, reply\}}{\bowtie} \\
&\qquad \Big(\big(Sr : Wait[N_s] \parallel Rb : Gather[N_r] \big) \underset{\{read, write\}}{\bowtie} \\
&\qquad\quad Db : Wait[N_d] \Big) / \{read, write\}
\end{aligned}$$

Fig. 3. PEPA model of a three-tier distributed application.

communicate with server components, denoted by the prefix $Sr:$, over the shared action types *request* and *reply*. The component $Sr : Wait$ illustrates two classes of request. Upon receiving a request, the information is retrieved via a database query with probability p_{fresh}; conversely, the server uses some cached data with probability $1 - p_{fresh}$, modelled as a reply without access to the database. A server may also incur some recoverable error, which requires retrieving information from the database in order to be able to accept further requests. When a database query is executed, the server checks whether the information is up-to-date. With probability $1 - p_{ok}$ this check fails and the server *forces* an update of the dataset, by performing the action *write*. A database server thread, denoted by the prefix $Db:$, is modelled as a two-state component. The state $Db : Wait$ exposes the two operations provided to the clients, while the state $Db : Update$ models some internal action which needs to be taken after every operation. The system also comprises a *robot* component, denoted by the prefix $Rb :$, describing the behaviour of a program which routinely writes to the database after gathering some data (modelled via the state $Rb : Gather$).

This model employs all of the operators of the language and features forms of interactions which were not allowed in earlier approaches to deterministic approximation. In particular, the following features were not supported in [8]:

- Sequential components participating in shared activities may specify distinct local rates (e.g., $r_{c:request}$ and $p_{fresh}r_{s:request}$).
- Two distinct local derivatives of the same sequential component may perform the same action type (e.g., $Sr : Fresh$ and $Sr : Repair$).
- Two distinct sequential component may compete for the same shared activitiy (e.g., $Sr : Write$ and $Rb : Write$).
- Support for hiding (e.g., here, $read$ and $write$ need not be seen by the client components).

The use of large population levels in models of this kind is justified by interpreting each distinct sequential component as a distinct process or thread of execution. Thus, $Cl : Request[N_c]$ indicates the total workload on the system, and the use of parallel composition expresses independence amongst the clients. $Sr : Wait[N_s]$ is the thread pool instantiated for the application server. Similarly, $Db : Wait[N_d]$ is the thread pool provided by the database. Note that this model of concurrency is in agreement with actual policies implemented by most web and database servers.

In practice, it is not unusual to have applications with hundreds of clients or multi-threaded servers with large pool sizes. However, such large-scale systems are difficult to analyse due to usually rapid state space growth. For instance, Table 4 shows the state space sizes in the NVF up to a maximum population size of ten. Clearly, explicit state-space enumeration makes the analysis intractable for scenarios with larger population sizes. An alternative approach in order to avoid onerous storage requirements consists of employing stochastic simulation. However, if on the one hand this reduces memory complexity dramatically, on the other it usually involves long execution times to compute a statistically significant number of samples.

5.2 Numerical Results

The validation tests were conducted on the following reduced model $\mathcal{M}_{App}$ obtained from $System_{App}$:

$$\mathcal{M}_{App} = Cl : Request \underset{\{request,reply\}}{\bowtie} \Big(\big(Sr : Wait \parallel Rb : Gather \big) \underset{\{read,write\}}{\bowtie} Db : Wait \Big) / \{read, write\}$$

A set of 200 randomly generated instances was constructed by drawing the values of the rate parameters from uniform distributions in $]0, 50]$ and the values of the probabilities p_{fresh} and p_{ok} from uniform distributions in $]0, 1[$. The initial densities of the local derivatives which do not appear in $\mathcal{M}_{App}$ were set to zero. The remaining densities were chosen at random between one and eight. Each model instance implies a family of CTMCs $\{X_n(t)\}$ and the corresponding ODE. The dynamics of the Markov processes at $n = 1$, $n = 10$, $n = 50$ and $n = 100$ were compared against the solution to the ODE. As an indicative measure of the quality of the approximation, the percentage relative errors between the expected value of the scaled Markov process $X_n(t)/n$ and the deterministic trajectory $x(t)$ were calculated for each coordinate i of the NVF at any given time point, according to the following equation:

$$\%\text{Error}_n^i(t) = \left| \frac{\mathbb{E}\left[X_n^i(t)/n\right] - x_i(t)}{\mathbb{E}\left[X_n^i(t)/n\right]} \right| \times 100 \qquad (28)$$

The results discussed in this section are provided for $t = 20.0$, arbitrarily chosen as a representative time point of the process since similar behaviour can also be observed for other time points. The analyses were conducted using the *Pepato* library, available from the *PEPA Eclipse Plug-in* software package [14]. For the sake of consistency, Gillespie's stochastic simulation algorithm (cfr. [3]) was employed for all values of n, although in principle the CTMCs for $n = 1$ could be solved numerically given their relatively small state space sizes. The simulations were terminated when the 95% confidence intervals were within 10% of the statistical averages. The ODEs were numerically integrated using a fifth-order Runge-Kutta solver.

The validation results are reported in Table 5. Each coordinate of the population vector behaves quantitatively differently. For instance, the deterministic estimates of the database and robot components are significantly more precise than the other sequential components. Nevertheless, in general the average approximation errors as well as their variance across the validation set decrease with n. These results also indicate that the deterministic approximation is sufficiently accurate for most practical purposes even at relatively low population levels. In particular, the scale factors $n \geq 10$ correspond to model instances with realistically sized pool sizes, i.e., hundreds of clients and server threads. In these cases the ODE solutions behave very well on average, with worst-case situations which give acceptable errors. Furthermore, as already observed in [8], ODE analysis is much less expensive than CTMC analysis—in this study the numerical integration of the ODE was found to be about four orders of magnitude faster, executing in tens of milliseconds on average.

As with any empirical assessment of this kind, the accuracy and the execution runtimes are specific to the validation data set under consideration. The ranges of the model parameters were arbitrarily chosen in this study. Different ranges (or different model structures)

TABLE 4
State-space sizes for the three-tier application model.

N_c, N_s, N_r, N_d	1	2	4	8	9	10
State-space size	32	315	7350	382239	800800	1574573

TABLE 5
Comparison between the expected value of the Markov process and the ODE solution at time $t = 20.0$. For each value of n and each coordinate in the NVF are listed the average percentage relative errors and the 5% and 95% percentiles across the validation set of 200 randomly generated model instances.

	$n = 1$			$n = 10$			$n = 50$			$n = 100$		
Component	5%	Avg.	95%	5%	Avg.	95%	5%	Avg.	95%	5%	Avg.	95%
$Cl : Request$	0.09%	19.62%	74.20%	0.01%	5.15%	29.09%	0.01%	1.87%	8.73%	0.01%	1.16%	4.85%
$Cl : Wait$	0.22%	17.09%	59.36%	0.03%	1.97%	7.57%	0.02%	0.76%	2.60%	0.02%	0.55%	1.70%
$Cl : Think$	0.70%	31.13%	87.57%	0.09%	2.96%	9.92%	0.06%	1.71%	6.00%	0.07%	1.62%	5.16%
$Sr : Wait$	0.31%	13.02%	50.49%	0.06%	2.46%	9.66%	0.05%	1.24%	4.56%	0.05%	1.23%	4.14%
$Sr : Fresh$	0.56%	20.21%	60.54%	0.09%	3.74%	12.81%	0.03%	2.09%	7.03%	0.06%	1.82%	5.68%
$Sr : Force$	1.20%	31.02%	85.57%	0.29%	4.39%	11.49%	0.22%	3.63%	9.17%	0.21%	3.27%	7.80%
$Sr : Write$	0.95%	27.68%	80.39%	0.21%	4.14%	12.38%	0.12%	2.91%	9.26%	0.10%	2.64%	8.91%
$Sr : Reply$	0.26%	24.69%	71.60%	0.07%	3.70%	13.10%	0.04%	1.69%	4.70%	0.05%	1.48%	5.44%
$Sr : Repair$	0.16%	13.19%	50.63%	0.01%	2.77%	11.37%	0.01%	1.32%	5.32%	0.02%	0.90%	3.92%
$Db : Wait$	0.01%	3.64%	20.21%	0.01%	0.77%	3.66%	0.01%	0.43%	1.70%	0.01%	0.38%	1.33%
$Db : Update$	0.04%	4.04%	17.08%	0.03%	1.07%	4.33%	0.01%	0.79%	2.93%	0.01%	0.81%	2.76%
$Rb : Gather$	0.05%	4.00%	16.56%	0.02%	1.09%	3.54%	0.02%	0.95%	3.23%	0.02%	0.89%	3.52%
$Rb : Write$	0.03%	2.82%	15.60%	0.02%	1.03%	3.12%	0.02%	0.91%	3.01%	0.01%	0.89%	3.00%

may lead to particularly problematic (or particularly good) cases. Moreover, some parameter sets may give rise to *stiff* differential equations which are difficult to integrate with explicit numerical solvers. Future work shall be concerned with these issues. A promising line of research seems to be, in the context of PEPA, the use of theoretical error probability bounds of density-dependent chains [15]. For a numerical investigation into the quality of the approximation of stiff large-scale models, the interested reader is referred to [16].

6 Related Work

The earlier work on the deterministic approximation of Markovian PEPA is due to Hillston [8]. In addition to the syntactical restrictions to the language discussed in Section 5.1, another major difference between our work and [8] concerns methodological aspects. Instead of an operational semantics for the language, [8] presents an algorithm for automatic generation of the ODE based on static inspection of the PEPA description which is not related to a corresponding CTMC. An alternative deterministic interpretation of PEPA in the style of [8] has been proposed in [17] for applications to epidemiology, albeit still with the aforementioned restrictions. The use of differential equations as the underlying mathematics of PEPA first appeared in the context of computational system biology in [18], where the authors present a methodology for the extraction of differential equations using a semantics called the *reagent-centric* view. This approach deviates significantly from the original interpretation of PEPA—most notably, the semantics of synchronisation captures the biologically interesting mass action kinetics as opposed to the standard notion of *bounded capacity*. Furthermore, unlike our approach, a sequential component does not represent a single entity of the system; rather, it is the abstraction of a concentration level of a species. This accomplishes an orthogonal goal with respect to ours, as the semantics associates a sequential component with an entire population instead of a single entity. Bounded capacity and mass action kinetics semantics for fluid models are merged in [19], in which a slight extension to the cooperation operation is provided to model signalling pathways which exhibit both kinds of reaction laws. An important contribution of this work is that the Markov process generated from the pathway-centric style of [20] is shown to converge in the limit to the underlying ODE by using Kurtz's results [6]. Density dependency and the interpretation of the ODE as the fluid limit of Markov process is also investigated in [21], in which some assumptions on the syntactical structure of the PEPA models are not removed and the generation of ODE is described algorithmically.

Apart from the context of PEPA, Cardelli has investigated the relationship between the deterministic and the discrete-state representation of the *Chemical Ground Form*, a process algebra designed for the modelling of chemical reactions [22]. In [23], Bortolussi and Policriti investigate the differential approximation of models of biochemical models using the *stochastic Concurrent Constraint Programming* process algebra.

A general modelling framework which exploits results of asymptotic convergence of stochastic processes to a differential-equation model is that of mean-field analysis (cfr. [24] and the bibliography therein). Similarly to our approach, it infers the collective (continuous) dynamics of a system from the description of a single participating object, which evolves through a (discrete) set of states. Mean-field analysis lends itself well to situations in which all objects have the same behaviour—indeed, it has been employed in performance studies of peer-to-peer protocols [25], [26]—and it easily allows for the modelling of communication between components of the same kind. Such a form of interaction is not available within the fluid-flow framework of PEPA because at its core is the notion of independence among components of an array. However, the semantics of our approach can be more readily used in cases where distinct kinds of interacting objects are to be considered.

7 Conclusions

This paper has presented a formal semantic account of the deterministic approximation of the stochastic process algebra PEPA, encompassing previous work on this topic and substantially extending the applicability to models with arbitrary structure. The modelling paradigm of PEPA is suited to capturing the behaviour of software systems consisting of interactions between replicated components, as may be the case in multi-process applications serving many customers. The result of asymptotic convergence relates the (computationally easy) solution of an underlying ODE to the Markov process obtained from the same process algebraic description, guaranteeing that the differential trajectory is an exact approximation. Furthermore, numerical tests on a model of a large three-tier distributed application gave confidence on the applicability of this analysis in realistically-sized large-scale models.

Acknowledgment

This work has been partially supported by the EU-funded project SENSORIA, IST-2005-016004. Jane Hillston is supported by the EPSRC ARF EP/c543696/01.

References

[1] W. Stewart, *Introduction to the Numerical Solution of Markov Chains*. Princeton University Press, 1994.

[2] A. L. Reibman and K. S. Trivedi, "Numerical transient analysis of Markov models," *Comput. Opns Res.*, vol. 15, no. 1, pp. 19–36, 1988.

[3] D. T. Gillespie, "Exact stochastic simulation of coupled chemical reactions," *The Journal of Physical Chemistry*, vol. 81, no. 25, pp. 2340–2361, 1977.

[4] M. Z. Kwiatkowska, G. Norman, and D. Parker, "Stochastic model checking," in *SFM*, ser. LNCS, M. Bernardo and J. Hillston, Eds., vol. 4486. Springer, 2007, pp. 220–270.

[5] P. Buchholz, "Exact and Ordinary Lumpability in Finite Markov Chains," *Journal of Applied Probability*, vol. 31, no. 1, pp. 59–75, 1994.

[6] T. G. Kurtz, "Solutions of ordinary differential equations as limits of pure Markov processes," *J. Appl. Prob.*, vol. 7, no. 1, pp. 49–58, April 1970.

[7] M. Mitzenmacher, "The power of two choices in randomized load balancing," Ph.D. dissertation, University of California at Berkeley, Department of Computer Science, Berkeley, CA, 1996.

[8] J. Hillston, "Fluid flow approximation of PEPA models," in *Proceedings of the Second International Conference on the Quantitative Evaluation of Systems*. Torino, Italy: IEEE Computer Society Press, Sep. 2005, pp. 33–43.

[9] ——, *A Compositional Approach to Performance Modelling*. Cambridge University Press, 1996.

[10] ——, "The nature of synchronisation," in *Proceedings of the Second International Workshop on Process Algebras and Performance Modelling*, U. Herzog and M. Rettelbach, Eds., Erlangen, Nov. 1994, pp. 51–70.

[11] S. Gilmore, J. Hillston, and M. Ribaudo, "An efficient algorithm for aggregating PEPA models," *IEEE Transactions on Software Engineering*, vol. 27, no. 5, pp. 449–464, May 2001.

[12] J. Bradley, S. Gilmore, and J. Hillston, "Analysing distributed internet worm attacks using continuous state-space approximation of process algebra models," *Journal of Computer and System Sciences*, vol. 74, no. 6, pp. 1013–1032, September 2008.

[13] R. A. Hayden and J. T. Bradley, "Evaluating fluid semantics for passive stochastic process algebra cooperation," *Performance Evaluation*, vol. 67, no. 4, pp. 260 – 284, 2010.

[14] M. Tribastone, A. Duguid, and S. Gilmore, "The PEPA Eclipse Plug-in," *Performance Evaluation Review*, vol. 36, no. 4, pp. 28–33, March 2009.

[15] R. Darling and J. Norris, "Differential equation approximations for Markov chains," *Probability Surveys*, vol. 5, pp. 37–79, 2008.

[16] M. Tribastone, "Relating layered queueing networks and process algebra models," in *WOSP/SIPEW '10: Proceedings of the first joint WOSP/SIPEW international conference on Performance engineering*. New York, NY, USA: ACM, 2010, pp. 183–194.

[17] S. Benkirane, J. Hillston, C. McCaig, R. Norman, and C. Shankland, "Improved continuous approximation of PEPA models through epidemiological examples," *Electronic Notes in Theoretical Computer Science*, vol. 229, no. 1, pp. 59–74, 2009.

[18] M. Calder, S. Gilmore, and J. Hillston, "Automatically deriving ODEs from process algebra models of signalling pathways," in *Computational Methods in Systems Biology*, 2005.

[19] N. Geisweiller, J. Hillston, and M. Stenico, "Relating continuous and discrete PEPA models of signalling pathways," *Theor. Comput. Sci.*, vol. 404, no. 1-2, pp. 97–111, 2008.

[20] M. Calder, S. Gilmore, and J. Hillston, "Modelling the Influence of RKIP on the ERK Signalling Pathway Using the Stochastic Process Algebra PEPA," *T. Comp. Sys. Biology*, vol. 4230, pp. 1–23, 2006.

[21] J. Ding and J. Hillston, "Convergence of the Fluid Approximation of PEPA Models," in *Seventh Workshop on Process Algebra and Stochastically Timed Activities (PASTA)*, Edinburgh, UK, 2008.

[22] L. Cardelli, "On process rate semantics," *Theor. Comput. Sci.*, vol. 391, no. 3, pp. 190–215, 2008.

[23] L. Bortolussi and A. Policriti, "Stochastic concurrent constraint programming and differential equations," *Electr. Notes Theor. Comput. Sci.*, vol. 190, no. 3, pp. 27–42, 2007.

[24] M. Benaïm and J.-Y. L. Boudec, "A class of mean field interaction models for computer and communication systems," *Performance Evaluation*, vol. 65, no. 11-12, pp. 823 – 838, 2008.

[25] A. Chaintreau, J.-Y. L. Boudec, and N. Ristanovic, "The age of gossip: spatial mean field regime," in *SIGMETRICS/Performance*, J. R. Douceur, A. G. Greenberg, T. Bonald, and J. Nieh, Eds. ACM, 2009, pp. 109–120.

[26] R. Bakhshi, L. Cloth, W. Fokkink, and B. Haverkort, "Mean-field analysis for the evaluation of gossip protocols," in *QEST '09: Proceedings of the 2009 Sixth International Conference on the Quantitative Evaluation of Systems*. Washington, DC, USA: IEEE Computer Society, 2009, pp. 247–256.

Mirco Tribastone received his degree in Computer Engineering from the University of Catania, Italy in 2005. He is currently a PhD candidate with the School of Informatics at the University of Edinburgh. His principal interests are in the quantitative evaluation of computer systems using analytical models. He is the lead developer of the *PEPA Eclipse Plug-in Project* which supports a range of qualitative and quantitative analysis methods for PEPA.

Stephen Gilmore joined the University of Edinburgh as a Lecturer in October 1990. He was then promoted to Senior Lecturer and is now a Reader in the School of Informatics. He was the Edinburgh site leader for the Sensoria project which developed performance models of large-scale systems and service-oriented computing. He is a co-investigator with Hillston on the Signal project using stochastic process algebra for biochemical signalling pathway analysis.

Jane Hillston received the PhD degree in computer science in 1994 from the University of Edinburgh where she is currently Professor of Quantitative Modelling. Her principal research interests are in the use of stochastic process algebras to model and analyse dynamic systems. She was awarded the first Roger Needham Award in 2004 by the British Computer Society in recognition of her work on PEPA. She was elected to fellowship of the Royal Society of Edinburgh in 2007.