Differential Analysis of Interacting Automata with Immediate Actions

Luca Bortolussi
Department of Mathematics and Geosciences
University of Trieste, Italy
ISTI-CNR, Pisa, Italy
luca@dmi.units.it

Mirco Tribastone
Electronics and Computer Science
University of Southampton, United Kingdom
m.tribastone@soton.ac.uk

ABSTRACT

The stochastic modelling of software systems with activities of durations that are separated by many orders of magnitude typically leads to numerical complications, due to *stiffness*. To avoid explicit state-space generation—a prerequisite to tackle this problem via suitable manipulations or aggregations—in this paper we present an accurate and scalable *fluid* approximation. It is expressed as a compact piecewise linear system of ordinary differential equations, which have discontinuous right-hand sides as a result of the incorporation of immediateness. We study the nature of this approximation in a general high-level framework of interacting automata. On a case study of client/server interaction, our approach is ca two times faster than the analysis conducted on the stiff equations where immediate actions are explicitly modelled.

Categories and Subject Descriptors

H.4 [**Information Systems Applications**]: Miscellaneous; D.2.8 [**Software Engineering**]: Metrics—*complexity measures, performance measures*

General Terms

Theory

Keywords

Immediate actions, differential equations, time-scale separation

1. INTRODUCTION

For accurate quantitative models of software systems, it is often necessary to describe the occurrence of logically crucial computations taking place over a time scale which can however be considered negligible with respect to all the other activities. This is the case, for instance, in fork/join synchronisations, where the process of spawning a new thread is much smaller than the thread's computational demand. Another example is exclusive access, for instance via a mutex, whereby the checks to be performed are negligible with respect to the time spent in the critical region.

If the system under scrutiny can be reasonably described as a continuous-time Markov chain (CTMC), two possible solutions can be considered in order to deal with such *fast* activities. The modeller could think of simply assigning to those a very large rate, separated by many orders of magnitude from the *slow* activities in the system. Unfortunately, this approach may introduce severe computational problems in the numerical solution, which are usually referred to as *stiffness*: the modeller is interested in the behaviour over time scales comparable to the slow rates, but the solution algorithms need to take step sizes of the order of the much faster rates, thus significantly slowing down the overall computation. Several authors have considered suitable model simplifications to eliminate stiffness (see, e.g., [3] and references therein).

Another alternative is to assume that the fast activities take no time whatsoever, and devise (stochastic) models which explicitly separate *immediate* activities from ordinary delayed ones. Among these approaches are generalised stochastic Petri nets (GSPNs) [14] and stochastic process algebra [9]. In all the aforementioned cases, the analysis technique essentially involves the generation of the full state space of the model containing slow states and immediate ones. A removal process transforms such state space into a smaller one that can be readily interpreted as a CTMC for the analysis. The very generation of the state space, unfortunately, makes these techniques difficult to apply to models of systems with a high degree of parallelism, because the cardinalities of both the original state space and the aggregated one grow very quickly with increasing populations (i.e., number of tokens or interacting processes).

Recent research has introduced so-called *fluid* models for the scalable analysis of software systems [19, 11]. The main computational advantage comes from the approximation of the time-course evolution of a (possibly huge) population of identical components with a single ordinary differential equation (ODE). Thus, the problem becomes only linearly dependent on the number of distinct classes of components. Crucially, fluid models are not currently available if there are immediate activities. To cope with this problem, one might introduce a *delay-only approximation*, whereby the ideally immediate transitions are replaced with very fast rates in the model. However, this approach causes computational complications due to stiffness also in the ODE realm [18].

In this paper we present a fluid model of interacting automata where immediate activities are first-class citizens. Our model is expressive enough to capture synchronous request/response between independent processes, exclusive access to a common resource, and fork/join synchronisation. Given a model, we define an algorithm to construct the set of fluid ODEs associated with it, working under some mild restriction of the synchronisation patterns. Importantly, the introduction of immediateness leads to an ODE system whose vector field does not satisfy certain classical conditions of regularity, as it turns out to be piecewise linear, with discontinuities that account for how the behaviour of an immediate transition is influenced by the other activities that enable that transition. Using recent results from the theory of dynamical systems with discontinuous right-hand sides [5, 6], the ODE systems are shown to have, however, a meaningful solution.

We prove that the trajectories solutions of the ODE obtained from the model with immediate actions coincide asymptotically with the ones associated with the ODEs obtained from the delay-only approximation, when all the very fast rates used to approach immediateness tend to infinity. In this sense, the ODEs associated with the model with immediate actions explicitly describe the different working regimes of the system, caused by the interaction of fast and slow activities. We also study the computational implication of this theoretical result. Whilst the delay-only model can only be integrated with a costly implicit scheme, which requires the solution of a system of nonlinear equations at each step (*backward differentiation formulae*), our model with discontinuities can be solved with an explicit solver which only requires the knowledge of the solution at the current type point (see, e.g., [1] for a discussion on numerical ODE integration). Experimentally, by comparing the runtimes registered on 400 different models for both solvers to escape an initially stiff regime, we show that our model with immediate actions yields an average speed-up of ca. 2 over the corresponding delay-only model. Although fluid approaches are known to yield relatively small absolute runtimes for moderately compact systems, this computational gain may be exploited to reduce the cost of problems that require expensive explorations of a model's parameter space.

Structure of the paper. Sect. 2 presents the model and introduces a running example used to explain all the derivation steps to obtain the underlying fluid approximation. Sect. 3 presents a simple model of a synchronous client/server request/response scenario with explicit, but immediate, message passing. We use this model to numerically compare the solution runtimes against those of its delay-only approximation. The models amenable to analysis with immediate actions must satisfy a condition of acyclicity. In Sect. 4, we provide an explanation of this condition by studying a model that violates it. Using this model, we also hint at how an analysis with immediate actions is still possible in an ad-hoc, case-by-case basis. Sect. 5, instead, presents a comparison between our approach and quasi-steady-state assumptions [16], a notable class of approximations for dynamical systems with pronounced time-scale decomposition. Finally, Sect. 6 concludes the paper.

2. MODEL

In this section we formally define and study our model framework. We will first define syntactically the model and then show how, under additional restrictions on the synchronisation of immediate actions, we can construct a system of ODEs. We will not explicitly provide a semantics of the model in terms of a CTMC, nor a definition of the ODEs as a fluid limit.

2.1 Model

We begin with introducing our model as a set of automata classes.

DEFINITION 1 (MODEL OF INTERACTING AUTOMATA). *Our model of interacting automata is a tuple $\Omega = (\mathbf{S}, \mathcal{L}, \lambda)$ where:*

- $\mathbf{S} = (\mathbf{S}_i)_{1 \le i \le N}$ *are the N classes of automata. Each $\mathbf{S}_i$ is a finite labelled transition system with N_i states, denoted by S_{ij}, $1 \le j \le N_i$.*
- $\mathcal{L}$ *is the set of transition labels, partitioned into $\mathcal{D}$ (delayed) and $\mathcal{Z}$ (immediate actions). We range over these sets with d and z, respectively. We adopt the usual notation for a transition, that is $S_{ij} \xrightarrow{l} S_{ik}$, $l \in \mathcal{L}$.*
- $\lambda : \mathcal{D} \to \mathbb{R}_{>0}$ *is the* rate function, *which associates a positive real with each delayed action. The rate is denoted by λ_d, $d \in \mathcal{D}$.*

REMARK 1. *As discussed, we do not consider a stochastic interpretation here. However, we have in mind delayed actions with exponentially distributed firing times. In our model, the rates will be considered as the deterministic rates of change of the overall populations of the N automata classes.*

Throughout the paper, we assume to be working with well-formed models, according to the following.

DEFINITION 2 (WELL-FORMEDNESS). *A model of interacting automata Ω is said to be well-formed if the following conditions hold:*

i) For each $l \in \mathcal{L}$, there are at most two distinct automata states with an outgoing transition labelled with l.

ii) No $\mathbf{S}_i$ contains a cycle through immediate actions only.

iii) For each S_{ij}, $S_{ij} \xrightarrow{d}$ implies that there is no z such that $S_{ij} \xrightarrow{z}$.

iv) For each S_{ij}, $S_{ij} \xrightarrow{z}$ implies that there is no $z' \neq z$ such that $S_{ij} \xrightarrow{z'}$.

REMARK 2. *Assumption* i) *enforces* binary *synchronisation for both immediate and delayed actions. This is not overly restricting the expressiveness of our model. Indeed,* multi-way *synchronisation for immediate actions can be implemented as a suitable sequence of binary synchronisations; for delayed actions, an extension of our model would be straightforward but would impinge on the succinctness of the forthcoming exposition;* ii) *requires that, at some stage, for each class of automata there must be a state where populations accumulate by means of non-zero delays;* iii) *requires no competition between immediate and delayed actions at a state, whereas* iv) *requires no competition between immediate transitions.*

Based on these assumptions, we call a state S_{ij} *immediate* if it has one outgoing transition with an immediate action label; otherwise S_{ij} will be called *delayed*.

In order to define a system of ODEs underlying our model of interacting automata, we make use of the following.

DEFINITION 3 (SEMANTICAL OBJECT). *The semantics of Ω is defined by the tuple $\Sigma(\Omega) = (\mathbf{x}, \mathcal{M}, \mathbf{r})$, where:*

- $\mathbf{x} = (\mathbf{x}_1, \ldots, \mathbf{x}_N)$, *with $\mathbf{x}_i = (x_{i1}, \ldots, x_{iN_i})$, is a tuple of $K = \sum_{i=1}^{N} N_i$* variables. *These will be interpreted as the ODE variables.*
- $\mathcal{M}$ *is a set of tuples in the form $(l, \mathbf{u}, f(\mathbf{x}))$, where:*
  - *l is an action in $\mathcal{L}$;*
  - $\mathbf{u}$ *is the* update tuple, *of length K, which records the changes in the populations of automata due to the execution of action l. The notation $\mathbf{u}\langle x_{ij}, x_{i'j'}, \ldots\rangle = \langle m_{ij}, m_{i'j'}, \ldots\rangle$ indicates that u has zero components except at the coordinates of $x_{ij}, x_{i'j'}, \ldots$ where it takes values $m_{ij}, m_{i'j'}, \ldots$, respectively. The coordinate for x_{ij} will be denoted by u_{ij}.*

  - $f(\mathbf{x})$ *is the* output flux*, an expression in the variables* $\mathbf{x}$*. It denotes the total deterministic rate at which the populations of automata change state as an effect of an execution of the* l*-th action.*

- $\mathbf{r}$ *is the* input-flux *tuple, of length* K*, which records an expression in* $\mathbf{x}$ *for each ODE variable. The input flux for* x_{ij} *is denoted by* $\mathbf{r}_{ij}$*.*

We now formally define how to construct $\Sigma(\Omega)$. We start by initially setting $\mathbf{r}_{ij} = 0$ for all i and j. Then, $\mathcal{M}$ and $\mathbf{r}$ are populated by exhaustively applying the rules presented in Fig. 1, where we adopt the following *update* notation: $\mathbf{r}_{ij} \overset{+}{\longleftarrow} e(\mathbf{x})$ means that, for some expression $e(\mathbf{x})$ with variables $\mathbf{x}$, $\mathbf{r}_{ij}$ is updated to be equal to $\mathbf{r}_{ij} + e(\mathbf{x})$, which is again a valid expression in $\mathbf{x}$. A similar notation applies to updates of $f(\mathbf{x})$ and $\mathcal{M}$. The operations performed via $\overset{+}{\longleftarrow}$ are assumed to be *symbolic*.

Informal description. The rules provide the aggregate rate at which a population of identical copies of a certain class S_{ij} of discrete-state components evolve, given the description of a representative element, characterised by the automaton $\mathbf{S}_i$. This is analogous in spirit to the *promotion rule* of [19] for population-based process algebra.

Rule ID considers the case where an automaton in local state S_{ij} may perform a d-action, with rate λ_d, in isolation, i.e., without synchronisation. In a population of x_{ij} components in state S_{ij}, a d-action will performed with the proportional rate $\lambda_d x_{ij}$. The update for $\mathbf{r}_{ik}$ captures the fact that $\lambda_d x_{ij}$ is also the rate at which the population of automata in state S_{ik} increase, thus contributing to S_{ik}'s total input flux.

The rule SD is similarly defined for synchronised agents over a delayed action. If λ_d is the delay of the single action, then $\phi(\mathbf{x})$ is the total rate across all possible pairs of automata that can synchronise at the same time. This semantics covers the behaviour of multi-server queueing networks (where synchronisation occurs between the customers in the queue and the number of servers), stochastic Petri nets with multiple-server semantics (cf., [13], where synchronisation is between the tokens of different incoming places into the same arc), and can be shown to be a special class of the stochastic process algebra PEPA [10].

The rules II and SI are instead peculiar to our model, since they deal with immediate activities, and can be best explained by means of the illustrations in Fig. 2. For rule II it is necessary to consider only the top diagram. The flux $\mathbf{r}_{ij}$, incoming into the immediate state S_{ij}, is redirected to S_{ik}. The output rate at S_{ij} is set equal to $\mathbf{r}_{ij}$ so that the average net flux is zero, capturing the fact non-zero populations of automata at state S_{ij} cannot be observed.

We now turn to an intuitive explanation of rule SI. Let us assume that, at some point, the population level of *only one* of S_{ij} and $S_{i'j'}$ is zero. For instance, let us consider the case $x_{ij} = 0$ and $x_{i'j'} > 0$. Here, $\psi(\mathbf{x})$ will evaluate to $\mathbf{r}_{ij}$. The reason is that the net rate at which the synchronisation occurs is dominated by the synchronising state with zero population. As soon as it receives an incoming flux $\mathbf{r}_{ij}$, this will be immediately moved to S_{ik} and to $S_{i'k'}$, because these are the two states reached by the immediate synchronisation. A similar argument holds for the symmetric case $x_{ij} > 0$ and $x_{i'j'} = 0$. Instead, in the case $x_{ij} = x_{i'j'} = 0$, $\psi(\mathbf{x})$ will evaluate to $\min\{\mathbf{r}_{ij}, \mathbf{r}_{i'j'}\}$. This captures the desired behaviour for synchronisation, as the total mass involved in the immediate rendezvous is constrained by the slowest of the two fluxes. In every case, both S_{ik} and $S_{i'k'}$ will witness the same incoming flux.

- For each $d \in \mathcal{D}$:

  - If $S_{ij} \xrightarrow{d} S_{ik}$ and there exists no $S_{i'j'}$, $i' \neq i$, $j' \neq j$ such that $S_{i'j'} \xrightarrow{d}$, then:

$$\mathcal{M} \overset{+}{\longleftarrow} (d, \mathbf{u}, f(\mathbf{x})), \quad \mathbf{r}_{ik} \overset{+}{\longleftarrow} \lambda_d x_{ij},$$

  with

$$\mathbf{u}\langle x_{ij}, x_{ik}\rangle = \langle -1, 1\rangle, \; f(\mathbf{x}) \overset{+}{\longleftarrow} \lambda_d x_{ij}.$$

[ID: Independent delayed action]

  - If $S_{ij} \xrightarrow{d} S_{ik}$ and $S_{i'j'} \xrightarrow{d} S_{i'k'}$, with $i' \neq i$ then:

$$\mathcal{M} \overset{+}{\longleftarrow} (d, \mathbf{u}, f(\mathbf{x})), \quad \mathbf{r}_{ik} \overset{+}{\longleftarrow} \phi(\mathbf{x}), \quad \mathbf{r}_{i'k'} \overset{+}{\longleftarrow} \phi(\mathbf{x}),$$

  with

$$\mathbf{u}\langle x_{ij}, x_{i'j'}, x_{ik}, x_{i'k'}\rangle = \langle -1, -1, 1, 1\rangle,$$
$$f(\mathbf{x}) \overset{+}{\longleftarrow} \phi(\mathbf{x}),$$
$$\phi(\mathbf{x}) := \lambda_d \min\{x_{ij}, x_{i'j'}\}.$$

[SD: Synchronised delayed action]

- For each $z \in \mathcal{Z}$:

  - If $S_{ij} \xrightarrow{z} S_{ik}$ and there exists no $S_{i'j'}$, $i' \neq i$, $j' \neq j$ such that $S_{i'j'} \xrightarrow{z}$, then:

$$\mathcal{M} \overset{+}{\longleftarrow} (z, \mathbf{u}, f(\mathbf{x})), \quad \mathbf{r}_{ik} \overset{+}{\longleftarrow} \mathbf{r}_{ij},$$

  with

$$\mathbf{u}\langle x_{ij}, x_{ik}\rangle = \langle -1, 1\rangle,$$
$$f(\mathbf{x}) \overset{+}{\longleftarrow} \mathbf{r}_{ij}.$$

[II: Independent immediate action]

  - If $S_{ij} \xrightarrow{z} S_{ik}$ and $S_{i'j'} \xrightarrow{z} S_{i'k'}$, with $i' \neq i$ then:

$$\mathcal{M} \overset{+}{\longleftarrow} (z, \mathbf{u}, f(\mathbf{x})), \quad \mathbf{r}_{ik} \overset{+}{\longleftarrow} \psi(\mathbf{x}), \quad \mathbf{r}_{i'k'} \overset{+}{\longleftarrow} \psi(\mathbf{x}),$$

  with

$$\mathbf{u}\langle x_{ij}, x_{i'j'}, x_{ik}, x_{i'k'}\rangle = \langle -1, -1, 1, 1\rangle,$$
$$f(\mathbf{x}) \overset{+}{\longleftarrow} \psi(\mathbf{x}),$$

$$\psi(\mathbf{x}) := \min\{\mathbb{H}(x_{ij})\mathbf{r}_{ij}, \mathbb{H}(x_{i'j'})\mathbf{r}_{i'j'}\}$$
$$\mathbb{H}(x) := \begin{cases} 1 & \text{if } x = 0, \\ \infty & \text{otherwise.} \end{cases}$$

[SI: Synchronised immediate action]

Figure 1: Rules for the construction of the semantical object $\Sigma(\Omega)$. To ease layout, we do not show the obvious ranges for the subscripts of S.

After all the rules of Fig. 1 are applied, some of the input fluxes $\mathbf{r}_{ij}$ may be defined in terms of some other $\mathbf{r}_{i'j'}$ (cf., rules II and SI). We capture the dependency amongst the input fluxes through the following.

Definition 4 (Dependency Graph). *Let* Ω *be a model of*

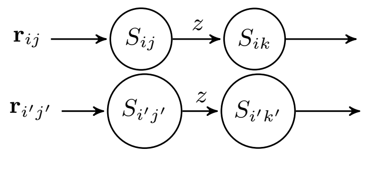


Figure 2: Illustration of the rules II and SI.

interacting automata and $\Sigma(\Omega) = (\mathbf{x}, \mathcal{M}, \mathbf{r})$ *its underlying semantical object. The* dependency graph *of* Ω*, written* $D(\Omega)$*, is a directed graph with vertices* $\mathbf{r}_{ij}$*, for all i and j, and arcs denoted by the relation* $\rightarrow$ *on* $\mathbf{r}$ *such that* $\mathbf{r}_{ij} \rightarrow \mathbf{r}_{i'j'}$ *iff* $\mathbf{r}_{i'j'}$ *appears in the expression of* $\mathbf{r}_{ij}$ *(i.e., iff the evaluation of* $\mathbf{r}_{ij}$ depends *on that of* $\mathbf{r}_{i'j'}$*).*

For instance, by applying rule II one would obtain that $\mathbf{r}_{ik} \rightarrow \mathbf{r}_{ij}$ in the dependency graph. The symbolic manipulations induced by our semantics are meaningful *if there is no cyclic dependence for any* $\mathbf{r}_{ij}$. That is, it is not possible to find a path such that $\mathbf{r}_{ij} \rightarrow \cdots \rightarrow \mathbf{r}_{ij}$ for any $\mathbf{r}_{ij}$. This introduces, *a fortiori*, a restriction on the synchronisation of immediate actions, as will be further discussed in Sect. 4. We notice that input fluxes that are defined solely in terms of the variables $\mathbf{x}$ (for instance, if some S_{ik} has only one incoming delayed transition, cf. rule ID) lead to vertices in $D(\Omega)$ with no outgoing transitions. Thus, if for each $\mathbf{r}_{ij}$ all the paths starting from $\mathbf{r}_{ij}$ are of finite length then $\mathbf{r}_{ij}$ may be evaluated from expressions which only depend on the variables $\mathbf{x}$. This suggests the following definition.

DEFINITION 5. *Let* Ω *be a well-formed model of interacting automata. Then,* Ω *is said* acyclic *iff* $D(\Omega)$ *is an acyclic graph.*

We postpone a discussion on this condition to Sect. 4. Now, we define the system of ODEs underlying our object according to the following.

DEFINITION 6. *The ODEs of a acyclic model of interacting automata* Ω *are given by:*

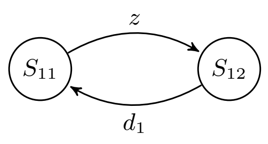
$$\dot{x}_{ij} = \sum_{(l,\mathbf{u},f(\mathbf{x}))\in\mathcal{M}} u_{ij} f(\mathbf{x}), \qquad 1 \le i \le N, 1 \le j \le N_i,$$

with some initial condition $\mathbf{x}(0)$ *such that* $x_{ij}(0) \ge 0$ *if* S_{ij} *is a delayed state,* $x_{ij}(0) = 0$ *if* S_{ij} *is an immediate state for a non-synchronised action z, and either* $x_{ij}(0) = 0$ *or* $x_{i'j'}(0) = 0$*, for* S_{ij} *and* $S_{i'j'}$ *immediate states synchronising on* $z \in \mathcal{Z}$*.*

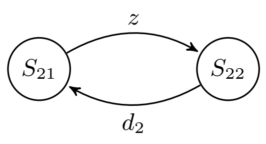
The choice of the initial conditions requires some further explanation. Essentially, we are excluding all those situations in which immediate transitions would be enabled at time 0. In fact, if S_{ij} is an immediate state for a non-synchronised action z, then z is enabled as soon as $x_{ij}(0) > 0$. Hence, z would fire immediately at time zero, emptying the counter of S_{ij} states and bringing the system into a new configuration in which $x_{ij}(0) = 0$. A similar behaviour can be observed if both $x_{ij}(0) > 0$ and $x_{i'j'}(0) > 0$, for S_{ij} and $S_{i'j'}$ immediate states synchronising on the shared activity z. It follows that we can always replace a generic set of initial conditions with an equivalent one conforming to our restrictions, after applying all enabled immediate transitions.

2.2 Worked Example

Let us consider the model in Fig. 3, where the interacting automata are depicted in the usual graphical style, in Figures 3(a) and 3(b). An immediate synchronisation occurs over label z between



(a) Automaton $\mathbf{S}_1$



(b) Automaton $\mathbf{S}_2$

Figure 3: Worked example.

two automata $\mathbf{S}_1$ and $\mathbf{S}_2$, after which both proceed independently with delayed actions d_1 and d_2.

Let us first apply rule SI to S_{11} and S_{21}, which yields the following:

$$\mathcal{M} = \big\{\big(z, (-1,+1,-1,+1), \min\{\mathbb{H}(x_{11})\mathbf{r}_{11}, \mathbb{H}(x_{21})\mathbf{r}_{21}\}\big)\big\},$$
$$\mathbf{r}_{12} = \mathbf{r}_{22} = \min\{\mathbb{H}(x_{11})\mathbf{r}_{21}, \mathbb{H}(x_{21})\mathbf{r}_{11}\},$$

where the notation for the jump vector follows the convention that the coordinates are ordered for x_{11}, x_{12}, x_{21}, and x_{22}. We note that $\mathbf{r}_{12}$ and $\mathbf{r}_{22}$ are both defined in terms of $\mathbf{r}_{21}$ and $\mathbf{r}_{11}$, which are in turn not yet fully defined. In $D(\Omega)$, we would have, for instance, $\mathbf{r}_{12} \rightarrow \mathbf{r}_{21}$ and $\mathbf{r}_{12} \rightarrow \mathbf{r}_{11}$. Now, applying rule ID to both S_{12} and S_{22} yields the following final $\Sigma(\Omega)$, where all $\mathbf{r}_{ij}$ may be expressed in terms of $\mathbf{x}$ directly:

$$\begin{aligned}
\mathbf{r}_{11} &= \lambda_{d_1} x_{12},\\
\mathbf{r}_{21} &= \lambda_{d_2} x_{22},\\
\mathbf{r}_{12} &= \min\{\mathbb{H}(x_{11})\mathbf{r}_{11}, \mathbb{H}(x_{21})\mathbf{r}_{21}\}\\
&= \min\{\mathbb{H}(x_{11})\lambda_{d_1}x_{12}, \mathbb{H}(x_{21})\lambda_{d_2}x_{22}\} = \mathbf{r}_{22},\\
\mathcal{M} &= \big\{\big(z, (-1,+1,-1,+1), \min\{\mathbb{H}(x_{11})\lambda_{d_1}x_{12}, \mathbb{H}(x_{21})\lambda_{d_2}x_{22}\}\big),\\
&\quad \big(d_1, (1,-1,0,0), \lambda_{d_1}x_{12}\big), \big(d_2, (0,0,1,-1), \lambda_{d_2}x_{22}\big)\big\}\,.
\end{aligned}$$

In components, the ODE system for the model in Fig. 3 is given by

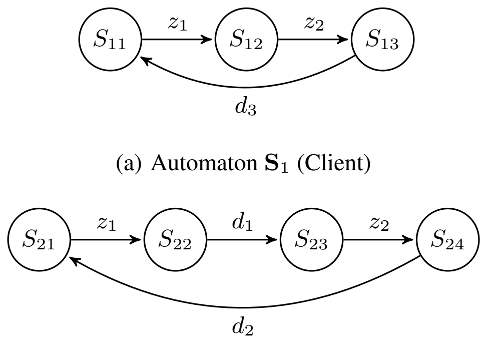
$$\begin{aligned}
\dot{x}_{11} &= -\min\{\mathbb{H}(x_{11})\lambda_{d_1}x_{12}, \mathbb{H}(x_{21})\lambda_{d_2}x_{22}\} + \lambda_{d_1}x_{12},\\
\dot{x}_{12} &= +\min\{\mathbb{H}(x_{11})\lambda_{d_1}x_{12}, \mathbb{H}(x_{21})\lambda_{d_2}x_{22}\} - \lambda_{d_1}x_{12},\\
\dot{x}_{21} &= -\min\{\mathbb{H}(x_{11})\lambda_{d_1}x_{12}, \mathbb{H}(x_{21})\lambda_{d_2}x_{22}\} + \lambda_{d_2}x_{22},\\
\dot{x}_{22} &= +\min\{\mathbb{H}(x_{11})\lambda_{d_1}x_{12}, \mathbb{H}(x_{21})\lambda_{d_2}x_{22}\} - \lambda_{d_2}x_{22}.
\end{aligned} \tag{1}$$

2.3 Properties

In this section we state the main results of the paper. We begin by considering the problem of existence and uniqueness of solutions of the ODEs of Definition 6, which is nontrivial due to their discontinuous right hand side. The following result is proved in Appendix A.

THEOREM 1. *For any well-formed and acyclic model of interacting automata* Ω*, the solution to the system of ODEs of Definition 6 exists and is unique.*

We turn now to considering the relationship between the fluid semantics of a model of interacting automata Ω, with label set $\mathcal{L} = \mathcal{D} \cup \mathcal{Z}$, that may exhibit immediate transitions, and that of a *delay-only* version Ω' constructed as follows. Assume without loss of generality that of $d^z \notin \mathcal{L}$. The model Ω' is defined as Ω except that every immediate transition z is replaced by a "fast" delayed action $d^z \in \mathcal{D}$, where $\lambda_{d^z} = q \gg 0$ for all d^z. For instance, let us consider the example discussed in Sect. 2.2. Then, its corre-



(a) Automaton $\mathbf{S}_1$ (Client)

(b) Automaton $\mathbf{S}_2$ (Server)

Figure 4: Case study.

sponding delay-only model has a system of ODEs given by

$$\begin{aligned}
\dot{x}_{11} &= -q\min\{x_{11}, x_{21}\} + \lambda_{d_1} x_{12},\\
\dot{x}_{12} &= +q\min\{x_{11}, x_{21}\} - \lambda_{d_1} x_{12},\\
\dot{x}_{21} &= -q\min\{x_{11}, x_{21}\} + \lambda_{d_2} x_{22},\\
\dot{x}_{22} &= +q\min\{x_{11}, x_{21}\} - \lambda_{d_2} x_{22}.
\end{aligned}$$

Intuitively, we expect that if q is taken to infinity, its solution should correspond to that obtained from the set of ODEs with immediate actions given in (1).

To prove this, let us first fix the notation by calling $F_q(\vec{x})$ the vector field obtained from the delay-only model Ω', and $F(\vec{x})$ the vector field obtained from the model Ω with immediate transitions (according to Definition 6). Then, call $\vec{x}_q(t)$ the solution of the ODE $\dot{\vec{x}} = F_q(\vec{x})$, starting from a point $\vec{x_0}$ admissible according to Definition 6, and $\vec{x}(t)$ the solution of the ODE $\dot{\vec{x}} = F(\vec{x})$, starting from the same initial point. For technical reasons, we restrict our attention to trajectories $\vec{x}(t)$ such that each, for each pair of states with synchronising fast variables S_{ij} and $S_{i'j'}$, it holds that x_{ij} and $x_{i'j'}$, $x_{ij}(t) = x_{i'j'}(t) = 0$ for at most countably many time instants, i.e. such that the trajectory never slides in the surface $\mathcal{H}_c = \{x_{ij} = 0\} \cap \{x_{i'j'} = 0\}$. Notice that this assumption rules out sliding motion in $\mathcal{H}_c$. What happens in this case is still an open problem, although we believe that convergence holds also in this case. Showing this, however, seems to require a radically different proof machinery.

We can now state the following result, whose proof is reported in Appendix B.

THEOREM 2. *Under the notation and assumptions stated above,*

$$\lim_{q\to\infty} \vec{x}_q(T) = \vec{x}(T), \ \forall T > 0.$$

3. CASE STUDY

In this section we discuss the computational implications of ODE analysis with immediate actions. We show how this approach may tackle numerical issues due to a pronounced time-scale separation in certain quantitative models of software systems, as has been required by the approximation of immediate transitions with fluid techniques using delay-only models.

Our scenario consists of clients (automaton $\mathbf{S}_1$) which call server instances (automaton $\mathbf{S}_2$) synchronously by means of an immediate action z_1; upon invocation, the server performs a computation with a delay with mean rate λ_{d_1}, after which it responds to the client by means of a synchronous immediate action z_2. The service interposes a *second phase* (borrowing terminology from layered queueing networks [7]) whereby it may process another request only after a mean delay λ_{d_2}. After the client receives the response with z_2, it interposes some think time at rate λ_{d_3}, before invoking the server again cyclically. The overall model is illustrated in Fig. 4.

$$\begin{aligned}
\dot{x}_{11} &= -\phi_{z_1} + \lambda_{d_3} x_{13}\\
\dot{x}_{12} &= +\phi_{z_1} - \phi_{z_2}\\
\dot{x}_{13} &= +\phi_{z_2} - \lambda_{d_3} x_{13}\\
\dot{x}_{21} &= -\phi_{z_1} + \lambda_{d_2} x_{24}\\
\dot{x}_{22} &= +\lambda_{d_2} x_{24} - \lambda_{d_1} x_{22}\\
\dot{x}_{23} &= +\lambda_{d_1} x_{22} - \phi_{z_2}\\
\dot{x}_{24} &= +\phi_{z_2} - \lambda_{d_2} x_{24}\\
\phi_{z_1} &= \min\{\mathbb{H}(x_{11})\lambda_{d_3} x_{13},\\
&\qquad \mathbb{H}(x_{21})\lambda_{d_2} x_{24}\}\\
\phi_{z_2} &= \min\{\mathbb{H}(x_{12})\phi_{z_1},\\
&\qquad \mathbb{H}(x_{23})\lambda_{d_1} x_{22}\}
\end{aligned}$$

(a)

$$\begin{aligned}
\dot{x}_{11} &= -\psi_{q_1} + \lambda_{d_3} x_{13}\\
\dot{x}_{12} &= +\psi_{q_1} - \psi_{q_2}\\
\dot{x}_{13} &= +\psi_{q_2} - \lambda_{d_3} x_{13}\\
\dot{x}_{21} &= -\psi_{q_1} + \lambda_{d_2} x_{24}\\
\dot{x}_{22} &= +\psi_{q_1} - \lambda_{d_1} x_{22}\\
\dot{x}_{23} &= +\lambda_{d_1} x_{22} - \psi_{q_2}\\
\dot{x}_{24} &= +\psi_{q_2} - \lambda_{d_2} x_{24}\\
\psi_{q_1} &= q\min\{x_{11}, x_{21}\}\\
\psi_{q_2} &= q\min\{x_{12}, x_{23}\}
\end{aligned}$$

(b)

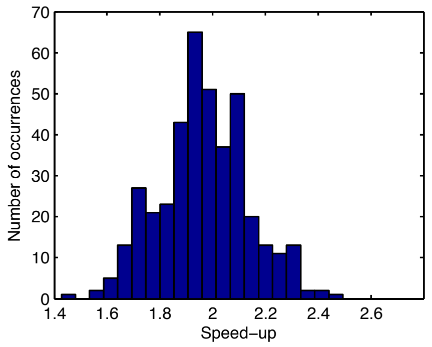
Figure 5: Model equations. (a) ODE system with immediate actions; (b) approximation by replacing z_1 and z_2 with synchronous delays q_1 and q_2.

Despite its simplicity, this model does not enjoy a tractable analytical form because of the presence of the second phase at the server—otherwise it would be in product form. Recent work has tackled this problem by considering a fluid approximation with a delay-only model [18]. Using the transformation considered in Sect. 2.3, the immediate actions z_1 and z_2 are replaced with synchronised delayed actions d^{z_1} and d^{z_2}. This model is easily seen to be acyclic, so that we can construct its fluid ODEs with the method of Sect. 2. The model equations for the original (discontinuous) system and the (Lipschitz continuous, but stiff) delay-only one are shown in Figures 5(a) and 5(b), respectively.

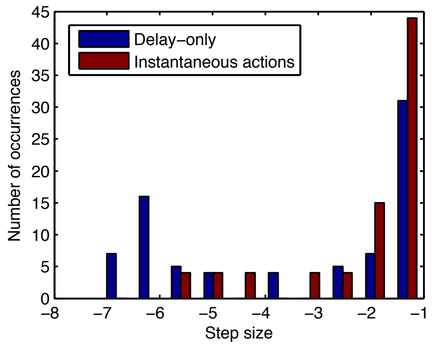
The delay-only model can be shown to be an accurate approximation of a many-server/many-client layered queueing network when the actual value of q is separated by many orders of magnitude from λ_{d_1} and λ_{d_2} [17, 18]. Therefore, here we do not consider the relationship between the fluid model and the stochastic process (with immediate activities). Rather, we numerically investigate the computational advantages gained by using the model of Fig. 5(a) in place of that of Fig. 5(b).

Next, we describe the evaluation methodology that we followed.

Experimental setup. All numerical computations reported in this section were performed on a MATLAB 7.9.0 environment running on an ordinary laptop computer. We generated 400 model instances with different initial conditions and rate values as follows. We fixed the values $\lambda_{d_2} = \lambda_{d_3} = 1.0$ and considered values for λ_{d_1} in the interval $[0.1, 10.0]$ with 0.5 step, in order to study varying ratios between the originally delayed actions in the system. For each such configuration of rates we considered varying ratios of initial conditions by fixing $x_{24}(0) = 10$ and letting $x_{13}(0)$ vary between 1 and 100 with step 5. All the remaining initial values were set to 0 in all cases. For each configuration we numerically integrated the ODEs of Fig. 5(a) and Fig. 5(b). The former were solved using the *ode45* routine, which implements the well-known Dormand-Prince integration scheme for non-stiff models; the latter were solved with *ode15s*, an implicit solver for stiff models. We furthermore set $q = 1\text{E}8$ in order to well express near-immediate



(a) Speed-up distribution



(b) Step-size distribution (log scale)

Figure 6: Evaluation results.

synchronisation. Each solver run was repeated 20 times to filter out the variance in the measurements. In all cases we set absolute and relative tolerances of the solvers to 1E-8 and 1E-5, respectively. Every solution was then computed over a time interval of 2 time units, of which we collected solution points with step size 0.02. This interval of time was found to be in all cases sufficient for the solution to escape the regime of stiffness.

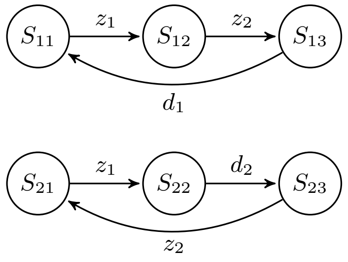
Evaluation. We first assessed the quality of the model with immediate actions with respect to the delay-only one with a finite (but large) q, according to the following notion of distance between the two numerical solutions. Let t_0, t_1, ..., t_N be the time instants returned by the solver (i.e., $t_0 = 0.00$, $t_1 = 0.02, \dots, t_N = 2.00$) and let $x_{ij}(t_k)$ and $\hat{x}_{ij}(t_k)$ be solutions of the two models. For each instance, we computed the distance

$$d = \max_{i,j,k} |x_{ij}(t_k) - \hat{x}_{ij}(t_k)|.$$

We numerically found that the maximum distance across all experiments was less than 4E-4, thus demonstrating the correspondence between the two models.

We measured the advantage in using the model with immediate actions by defining a notion of speed-up whereby, for each model, the runtime of the delay-only ODE system solution is divided by that with immediate actions. Figure 6(a) shows the overall distribution of the measured speed-up. The average speed-up, 1.97, together with the minimum (1.42) and maximum (2.49) ones, indicate an excellent behaviour in all cases.

The improvement can be explained by measuring the distribution of the step sizes used by *ode45* and *ode15s* for solving the same model. Figure 6(b) shows the distribution (in logarithmic scale) of first 80 step sizes for the model with the highest speed-up ratio (i.e., $x_{11}(0) = 6$ and $\lambda_{d_1} = 2.1$) in order to better highlight a phenomenon which is however typical of all the instances considered in this study. The delay-only model shows a significant number of small step sizes (left part of the figure) which are needed to tackle the initial stiffness of the problem. Both distributions are skewed to the right because, once stiffness is avoided, larger step sizes may be taken in both cases.



$$\begin{tikzcd} S_{11} \arrow[r, "z_1"] & S_{12} \arrow[r, "z_2"] & S_{13} \arrow[ll, bend left, "d_1"] \end{tikzcd}$$

$$\begin{tikzcd} S_{21} \arrow[r, "z_1"] & S_{22} \arrow[r, "d_2"] & S_{23} \arrow[ll, bend left, "z_2"] \end{tikzcd}$$

Figure 7: Cyclic model.

4. ON MODEL ACYCLICITY

This section proposes a discussion of the condition of acyclicity in Definition 5. Although our results hold in general for acyclic models, here we discuss that acyclicity is not a necessary condition for permitting ODE analysis with immediate rates. However, the analysis appears to be model-dependent. Here, we hint at how to deal with cyclicity using an example. We leave it to future work further investigation of a more general analysis of acyclic models.

Our model of interest is obtained by a simple modification of the case study shown in Fig. 4, namely by removing d_2, the second delayed phase of the servers. The so obtained automata are depicted in Fig. 7.

Applying the procedure to construct ODEs of Sect. 2, the input flow $\mathbf{r}_{12}$ due to transition z_1 is

$$\phi_{z_1} = \min\{\mathbb{H}(x_{21})\phi_{z_2}, \mathbb{H}(x_{11})\lambda_{d_1}x_{13}\},$$

since $\mathbf{r}_{11} = \lambda_{d_1}x_{13}$, while the input flow in states S_{13} and S_{21}, determined by transition z_2, is

$$\phi_{z_2} = \min\{\mathbb{H}(x_{12})\phi_{z_1}, \mathbb{H}(x_{23})\lambda_{d_2}x_{22}\},$$

since $\mathbf{r}_{23} = \lambda_{d_2}x_{22}$. If a state is such that $x_{21} = x_{12} = 0$ and $x_{11} > 0$, $x_{23} > 0$, the previous equations both simplify to $\phi_{z_1} = \phi_{z_2}$: the value of ϕ_{z_1} and ϕ_{z_2} is therefore undefined. Indeed, the method introduced in the previous sections does not apply here, as the dependency graph $D(\Omega)$ has a cycle. This is shown in Fig. 8, in which we annotated the graph with further information: the shape of the vertices distinguishes between *ground flows* (depicted as double circles), i.e. input flows coming from a delayed transition, and *non-ground flows* (single circles), coming from immediate events. Furthermore, each edge is annotated with the corresponding action and with the conditions on the variables for the edge to be used in the computation of the flow (i.e., the conditions such that $\mathbb{H}$ equals one).

As we can see, there is a cycle between vertices $\mathbf{r}_{12}$ and $\mathbf{r}_{21}$, but this is in force if and only if both x_{12} and x_{21} are zero. Now, it is easy to check that this situation corresponds to a deadlocked state in the model. Indeed, when x_{12} and x_{21} are zero, all servers are in states S_{22} or S_{23}, with all clients being in states S_{11} or S_{13}. Hence, transition z_1 is disabled as there are no servers in state S_{21}, and transition z_2 is also disabled due to the absence of clients in

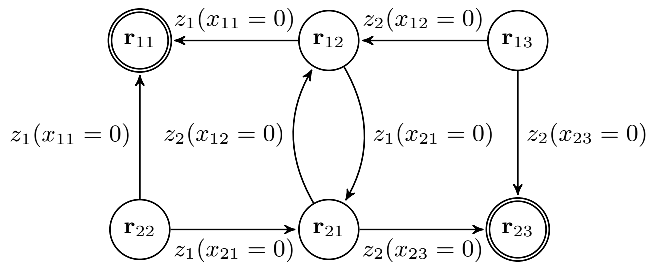

$$\begin{tikzcd}
\boxed{\mathbf{r}_{11}} & \mathbf{r}_{12} \arrow[l, "z_1(x_{11}=0)"'] \arrow[d, bend right, "z_2(x_{12}=0)"'] & \mathbf{r}_{13} \arrow[l, "z_2(x_{12}=0)"'] \arrow[d, "z_2(x_{23}=0)"] \\
\mathbf{r}_{22} \arrow[u, "z_1(x_{11}=0)"] \arrow[r, "z_1(x_{21}=0)"'] & \mathbf{r}_{21} \arrow[u, bend right, "z_1(x_{21}=0)"'] \arrow[r, "z_2(x_{23}=0)"'] & \boxed{\mathbf{r}_{23}}
\end{tikzcd}$$

Figure 8: Flow graph for the example of Section 4.

state S_{12}. However, such a deadlocked configuration is easily seen to be unreachable if all clients and servers are initially in state S_{11} and S_{21}!

Therefore, the idea to define a vector field for this example is to impose a condition forcing the absence of such a deadlock, namely assuming that either $x_{12} > 0$ or $x_{21} > 0$. When writing the ODEs solving the dependency between $\mathbf{r}_{ij}$'s, we further need to distinguish which specific constraint we are enforcing (i.e. either $x_{12} > 0$ or $x_{21} > 0$). For this example, it is easy to obtain the following "ground" definitions of $\mathbf{r}_{12}$ (i.e. ϕ_{z_1}) and $\mathbf{r}_{21}$ (i.e. ϕ_{z_2}):

$$\phi_{z_1} = \begin{cases} \lambda_{d_1} x_{13} & \text{if } x_{11} = 0 \wedge x_{21} > 0 \\ \min\{\mathbb{H}(x_{11})\lambda_{d_1} x_{13}, \lambda_{d_2} x_{22}\} & \text{if } x_{11} \geq 0 \wedge x_{21} = 0 \\ & [\text{as } x_{23} = 0 \wedge x_{12} > 0] \end{cases}$$

$$\phi_{z_2} = \begin{cases} \lambda_{d_2} x_{22} & \text{if } x_{12} > 0 \wedge x_{23} = 0 \\ \min\{\mathbb{H}(x_{23})\lambda_{d_2} x_{22}, \lambda_{d_1} x_{13}\} & \text{if } x_{12} = 0 \wedge x_{23} \geq 0 \\ & [\text{as } x_{11} = 0 \wedge x_{21} > 0] \end{cases}$$

This leads to the following set of fluid differential equations:

$$\begin{aligned} \dot{x}_{11} &= -\phi_{z_1} + \lambda_{d_3} x_{13} \\ \dot{x}_{12} &= +\phi_{z_1} - \phi_{z_2} \\ \dot{x}_{13} &= +\phi_{z_2} - \lambda_{d_3} x_{13} \\ \dot{x}_{21} &= -\phi_{z_1} + \phi_{z_2} \\ \dot{x}_{22} &= +\phi_{z_1} - \lambda_{d_2} x_{22} \\ \dot{x}_{23} &= +\lambda_{d_2} x_{22} - \phi_{z_2} \end{aligned}$$

In order for this construction of the vector field to be meaningful, we need to be sure that the ODE trajectories can never enter into the region $B = \{x_{21} = 0 \wedge x_{12} = 0\}$, where the vector field is undefined, i.e. that the deadlocked state is unreachable also in the fluid model. This is indeed the case, as can be seen observing that $\dot{x}_{12} = -\dot{x}_{21}$. Hence, if we are in a point close to B, when x_{21} decreases, x_{12} increases, and viceversa. Hence we cannot reach a state in which both variables are zero. Staten otherwise, the vector field points outwards B in a neighbourhood of B, so that the solutions of the fluid ODE can never enter B.

For validation purposes, in Figure 9, we compare the solution of the system of ODEs presented above with the solution of the system of ODEs obtained by replacing instantaneous transitions with fast exponential ones. As we can see, the agreement is excellent in this case. As discussed, extending the scope of this analysis to more general models will be subject of future work.

5. RELATED WORK

Time-scale separation in ODE systems has received considerable attention outside the context of computer science and engineering due to the occurrence of many natural phenomena exhibiting such a behaviour. In biochemistry, for instance, approaches to tackle this problem are known as *quasi steady-state approximations* (QSSAs) [16]. They are applicable in cases when there are variables that reach equilibrium after a very short transient; these are approximated to have zero derivatives at all time points. In our notation, this translates into assuming that $\dot{x}_{ij} = \sum_{(l,\mathbf{u},f(\mathbf{x}))\in\mathcal{M}} u_{ij} f(\mathbf{x}) = 0$, for some i and j. From this, one separates x_{ij} from the right-hand side of the equation, and plugs the expression in the rest of the ODE system.

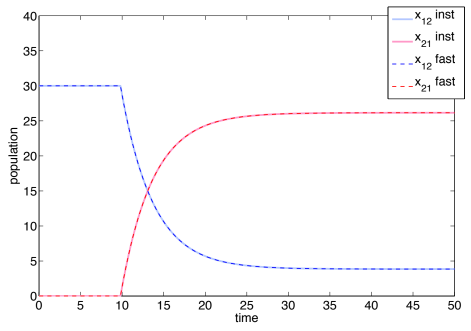


Figure 9: Comparison between the ODE systems of the model of Sect. 4 with immediate (light continuous lines) and fast (dark dashed lines) rates, for variables x_{12} and x_{21}. Parameters are $\lambda_{d_1} = 0.01$ and $\lambda_{d_2} = 0.25$, for 100 clients and 30 servers.

Here we wish to show that QSSA is significantly less robust than our approach, for the ODE systems generated according to the semantics presented in Sect. 2.1. For this exercise, let us consider a model in the form of Fig. 5(b). Firstly, we observe that QSSA does not constructively hint at the candidate variables to simplify. Here, we circumvent this problem by looking at the derivatives of the ODE solution. For $x_{13}(0) = 1$, $\lambda_{d_1} = 0.1$, and all other parameters set as in Sect. 3, the results of the numerical integration are shown in Fig. 10. This allows us to identify x_{11} and x_{23} as the variables to simplify.

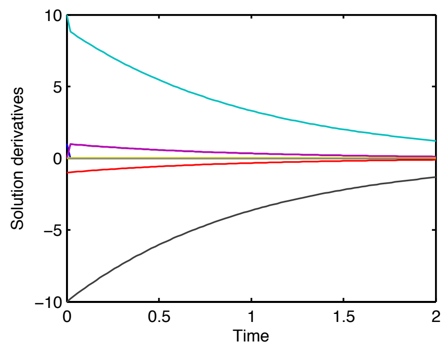
A problem now arises by setting $\dot{x}_{11} = 0$ (and, analogously, $\dot{x}_{23} = 0$) due to the non-linearity of the right hand side which gives

$$\lambda_{q_1} \min\{x_{11}, x_{21}\} = \lambda_{d_3} x_{13}.$$

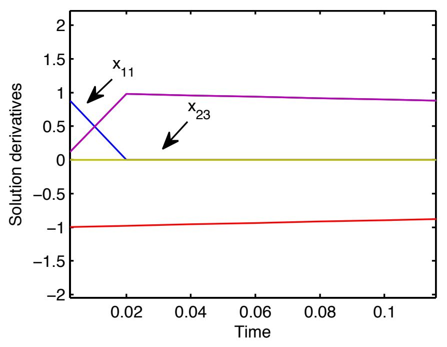
This expression cannot be written explicitly in terms of x_{11}, as would be required by QSSA. However, we find that, in these cases, it also holds that $x_{11}(t) \approx 0$ and $x_{23}(t) \approx 0$ for all t (not graphically shown for reasons of space). By setting both variables equal to 0, the ODE system of Fig. 5(b) is greatly simplified into:

$$\begin{aligned} \dot{x}_{11} &= 0, \\ \dot{x}_{12} &= -\lambda_{d_1} x_{22} + \lambda_{d_3} x_{13}, \\ \dot{x}_{13} &= +\lambda_{d_1} x_{22} - \lambda_{d_3} x_{13}, \\ \dot{x}_{21} &= -\lambda_{d_3} x_{13} + \lambda_{d_2} x_{24}, \\ \dot{x}_{22} &= +\lambda_{d_3} x_{13} - \lambda_{d_1} x_{22}, \\ \dot{x}_{23} &= 0, \\ \dot{x}_{24} &= \lambda_{d_1} x_{22} - \lambda_{d_2} x_{24}. \end{aligned}$$

It can be shown that the solution of this ODE system corresponds to that of Fig. 5(b), up to numerical precision of the numerical integrator. However, as tempting as it may appear—the ODE system size is reduced and, in this case, it even admits a closed-form



(a)



(b)

Figure 10: Derivatives of the ODE solution of the model in Fig. 5(b) with $x_{13}(0) = 1$ and $\lambda_{d_1} = 0.1$. (a) Integration between over time interval $[0, 2]$; (b) zoomed-in plot.

solution—this approximation is very frail. Essentially, it has transformed the original non-linear model of Fig. 5(b) into a linear one: the latter will only be an accurate approximation so long as the solution does traverse the hyperplanes of non-linearity induced by the minimum expressions. That is, with the initial conditions of Fig. 10, the solution coincides *exactly* with that of the original model. However, it will be grossly inaccurate by setting another initial condition, for instance, $x_{13}(0) = 20$ and $\lambda_{d_1} = 0.1$ (cf. Fig. 11). Notice that both cases were considered in the assessment study of Sect. 3, demonstrating the robustness of our method with respect to the QSSA discussed above.

6. CONCLUSION

Although the treatment of immediate activities is well understood in stochastic models with discrete state spaces, the explicit state-space exploration may impede the applicability of the technique to very large scale systems. On the other hand, in these cases fluid models are particularly accurate, but they do not incorporate immediate activities, if not with approximations which lead computational problems due to stiffness.

We have started to fill the gap between these two worlds by presenting an accurate and scalable fluid model for interactive automata with immediate activities, under suitable restrictions on the synchronisation and competition of immediate actions (the condition of acyclicity).

In order to make the modelling framework more general, building on the example of Sect. 4, we want to provide a definition of the vector field also for models which are cyclic. Furthermore, some

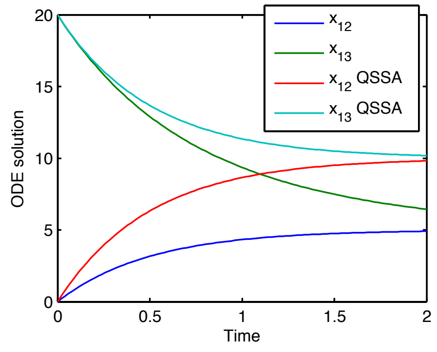


Figure 11: Representative traces comparing the ODE solutions of Fig. 5(b) and the QSSA approximation when the initial conditions, i.e., $x_{13}(0) = 20$ and $\lambda_{d_1} = 0.1$, push the trajectory to a different region of non-linearity.

forms of interaction are not currently allowed in our models. A compelling restriction that we intend to relax in the future is the impossibility at a given state to allow two or more kinds of immediate actions, as this prevents the modelling of certain multi-class situations, e.g., two kinds of clients requiring distinct demands on the same server.

An interesting research question raised by this paper, which we intend to explore in the future, is to understand whether the underlying ODE system with discontinuities can be shown to correspond to the deterministic limit behaviour of a suitable sequence of stochastic processes (with immediate delays). To this end, we plan to exploit recent advances in fluid approximation theorems, extending the range of applicability of standard techniques [12, 2] to ODE systems with discontinuities [4] or differential inclusions [8]. The idea is to describe the CTMC associated with our model by removing vanishing states and combining immediate transitions with the triggering stochastic ones, using guard predicates (e.g., in the format of [4]) on variables to restrict to the non-vanishing state space.

Acknowledgement. This work has been partially sponsored by the EU project QUANTICOL, 600708, and by the DFG SPP-1593 project DAPS.

7. REFERENCES

[1] U. M. Ascher and L. R. Petzold. *Computer Methods for Ordinary Differential Equations and Differential-Algebraic Equations*. SIAM, 1988.

[2] M. Benaïm and J.-Y. Le Boudec. A class of mean field interaction models for computer and communication systems. *Performance Evaluation*, 65(11-12):823–838, 2008.

[3] A. Bobbio and K. Trivedi. An aggregation technique for the transient analysis of stiff Markov chains. *IEEE Transactions on Computers*, 35:803–814, 1986.

[4] L. Bortolussi. Hybrid limits of continuous time markov chains. In *Proceedings of Eighth International Conference on the Quantitative Evaluation of Systems, QEST 2011*, pages 3–12. IEEE Computer Society, 2011.

[5] J. Cortes. Discontinuous dynamical systems: A tutorial on solutions, nonsmooth analysis, and stability. *IEEE Control Systems Magazine*, pages 36–73, 2008.

[6] A. Filippov. *Differential Equations with discontinuous right-hand sides*. Mathematics and Its Applications. Kluwer Academic, 1988.

[7] G. Franks, T. Omari, C. M. Woodside, O. Das, and S. Derisavi. Enhanced modeling and solution of layered queueing networks. *IEEE Trans. Software Eng.*, 35(2):148–161, 2009.

[8] N. Gast and B. Gaujal. Mean field limit of non-smooth systems and differential inclusions. *SIGMETRICS Perform. Eval. Rev.*, 38:30–32, October 2010.

[9] H. Hermanns. *Interactive Markov Chains: The Quest for Quantified Quality*, volume 2428 of *Lecture Notes in Computer Science*. Springer, 2002.

[10] J. Hillston. *A compositional approach to performance modelling*. Cambridge University Press, New York, NY, USA, 1996.

[11] J. Hillston. Fluid flow approximation of PEPA models. In *Proceedings of the Second International Conference on the Quantitative Evaluation of Systems*, pages 33–43, Torino, Italy, Sept. 2005. IEEE Computer Society Press.

[12] T. Kurtz. *Approximation of population processes*. SIAM, 1981.

[13] M. A. Marsan, G. Balbo, G. Conte, S. Donatelli, and G. Franceschinis. *Modelling with Generalised Stochastic Petri Nets*. John Wiley and Sons, 1995.

[14] M. A. Marsan, G. Conte, and G. Balbo. A class of generalized stochastic Petri nets for the performance evaluation of multiprocessor systems. *ACM Trans. Comput. Syst.*, 2(2):93–122, 1984.

[15] W. Rudin. *Principles of Mathematical Analysis*. McGraw-Hill, 1976.

[16] L. Segel and M. Slemrod. The quasi-steady-state assumption: A case study in perturbation. *SIAM Review*, 31(3):446–477, 1989.

[17] M. Tribastone. Approximate mean value analysis of process algebra models. In *MASCOTS: IEEE 19th International Symposium on Modelling, Analysis and Simulation of Computer and Telecommunication Systems*, pages 369–378, Singapore, July 2011. IEEE Computer Society Press.

[18] M. Tribastone. A fluid model for layered queueing networks. *IEEE Transactions on Software Engineering*, 39(6):744–756, 2013.

[19] M. Tribastone, S. Gilmore, and J. Hillston. Scalable differential analysis of process algebra models. *IEEE Trans. Software Eng.*, 38(1):205–219, 2012.

APPENDIX

A. PROOF OF THEOREM 1

In this appendix, we sketch the proof of Theorem 1. However, a fully formal argument can be constructed easily.

First, we observe that the model is piecewise linear. If the solution lives in a region in which the system is Lipschitz continuous, there is nothing to prove: existence and uniqueness follow from classical results in analysis [15]. Hence, we need to focus on the surfaces in which discontinuities are present. As these are induced by an immediate synchronisation, we consider an arbitrary one, and we let $z \in \mathcal{Z}$ be the associated action label. This event has a flow, according to the rules in Figure 1, which is equal to $\psi(\mathbf{x}) := \min\{\mathbb{H}(x_{ij})\mathbf{r}_{ij}, \mathbb{H}(x_{i'j'})\mathbf{r}_{i'j'}\}$, where $\mathbf{r}_{ij}$ and $\mathbf{r}_{i'j'}$ are the input flows into states S_{ij} and $S_{i'j'}$, from which the immediate transition z departs. For the moment, suppose that these flows are uniquely defined in any possible state of the system and lead to unique solutions. This is the case, for instance, if states S_{ij} and $S_{i'j'}$ have no incoming immediate transitions.

First, we observe that the motion of solutions of the limit vector field is confined in the subset of the positive octant (variables are always greater or equal than zero by construction) given by the hyper-surface $\mathcal{H} = \{x_{ij} = 0\} \cup \{x_{i'j'} = 0\}$: we cannot have both $x_{ij} > 0$ and $x_{i'j'} > 0$, otherwise the system will jump immediately back to $\mathcal{H}$ (as the flow $\psi(\mathbf{x})$ will be infinite).

We can partition this surface into three regions: $\mathcal{H}_a = \{x_{ij} > 0\} \cap \{x_{i'j'} = 0\}$, $\mathcal{H}_b = \{x_{ij} = 0\} \cap \{x_{i'j'} > 0\}$, and $\mathcal{H}_c = \{x_{ij} = 0\} \cap \{x_{i'j'} = 0\}$. In the first and second region, the flow component of the vector field due to z equals respectively $\mathbf{r}_{i'j'}$ and $\mathbf{r}_{ij}$. In particular, in $\mathcal{H}_a$, the ODE for $x_{i'j'}$ is identically zero, while that for x_{ij} is $\mathbf{r}_{ij} - \mathbf{r}_{i'j'}$, which can be positive, negative, or zero. If it is negative and remains so for long enough, the solution will eventually enter the surface $\mathcal{H}_c$. Notice that the vector field in $\mathcal{H}_a$ is Lipschitz continuous, hence solutions exist and are unique in this region. By symmetry, a similar argument applies to $\mathcal{H}_b$.

To deal with $\mathcal{H}_c$, observe that, in that situation, the flow depends on the relative ordering of $\mathbf{r}_{i'j'}$ and $\mathbf{r}_{ij}$. In particular, if $\mathbf{r}_{i'j'} < \mathbf{r}_{ij}$, the vector field immediately leaves $\mathcal{H}_c$ heading for $\mathcal{H}_a$, as the ODE for x_{ij} is $\mathbf{r}_{ij} - \mathbf{r}_{i'j'} > 0$. This corresponds essentially to a transversal crossing, in the piecewise smooth systems jargon [5]. A similar behaviour happens when $\mathbf{r}_{i'j'} > \mathbf{r}_{ij}$. Finally, when $\mathbf{r}_{i'j'} = \mathbf{r}_{ij}$, the solution remains in $\mathcal{H}_c$ until equality is broken. This is a stable sliding motion [5]. As we have covered all possibilities and ruled out unstable sliding motion, i.e. the possibility of leaving $\mathcal{H}_c$ simultaneously towards $\mathcal{H}_a$ and $\mathcal{H}_b$, the solutions of the ODE exists and are unique also from $\mathcal{H}_c$. In particular, the vector field in $\mathcal{H}_c$ is uniquely identified by the input flows $\mathbf{r}_{ij}$ and $\mathbf{r}_{i'j'}$.

This argument relied on the fact that those input flows are uniquely defined and determine a unique solution. This is the case, as remarked above, if the states S_{ij} and $S_{i'j'}$ have no incoming immediate transitions. To deal with the general case, we exploit the fact that there are no loops of immediate transitions, as per Definition 2, hence we can reason inductively on sequences of immediate transitions (or more precisely, on the length of the longest incoming path made only from immediate transitions), using the previous argument to deal both with the base and the inductive cases.

B. PROOF OF THEOREM 2

We provide here a proof of Theorem 2. Instead of giving a fully formal proof in the epsilon-delta style of analysis, we give a more intuitive explanation, which can be turned into a precise argument by taking care of all the details involved.

Recall that we indicate with F_q the vector field of the all-delayed model with fast rate q, and with F the vector field of the model with immediate transitions. In the following, we will use the subscript or superscript q whenever appropriate for distinguishing the two models.

Secondly, we observe that the syntactic restrictions of Definition 2 guarantee that we can observe only finite sequences of immediate transitions, as no loops are possible, and that there is further no competition between immediate transitions (which avoids to deal with devices to solve the non-deterministic behaviour probabilistically) or immediate and delayed transitions.

Now, recall that states S_{ij}, and consequently the variables x_{ij}, can be split in two classes: immediate, if there is a fast or immediate transition exiting from S_{ij}, and delayed otherwise. In the following, we will refer to these two classes also as fast and slow, respectively.

We will first prove that F_q converges to F as $q \to \infty$, *assuming that slow variables are fixed*, by reasoning inductively on the path of the dependency graph (Definition 4) of the limit model, inverting the direction of its edges. In this way, top nodes in the graph correspond to states with no incoming immediate transitions. This is the base case of the induction.

To deal with top nodes, focus the attention on a single state S_{ij}. We can have two cases: either it is involved in a synchronised transition, or it is not. In this latter case, we have locally a situation like the one depicted below:

$$\begin{tikzcd}
\mathbf{r}_{ij} \arrow[r] & \boxed{S_{ij}} \arrow[r, "z"] & f_z^q
\end{tikzcd}$$

In this case, $\mathbf{r}_{ij}$ contains only slow rates (i.e. rates of slow transitions), which are assumed fixed. Hence we have that in F_q, the ODE for x_{ij} is

$$\dot{x}_{ij} = \mathbf{r}_{ij} - f_z^q,$$

where $f_z^q = qx_{ij}$. Now, assuming $q \gg 0$ and much bigger than all slow rates, the dynamics involving x_{ij} will be very fast and reach equilibrium almost instantaneously (in particular, much before slow variables are modified sensibly, which justifies heuristically the fact that we consider them fixed). Now, the assumption that $\mathbf{r}_{ij}$ is constant in turns implies that the equation for x_{ij} is linear and converges to the steady state $\tilde{x}_{ij} = \mathbf{r}_{ij}/q$. Furthermore, the speed of convergence to equilibrium is inversely proportional to q, and goes to zero as q goes to infinity. Hence, for $q \to \infty$, we have that $f_z^q \to \tilde{f}_z^q = \mathbf{r}_{ij}$, which is exactly the x_{ij} component of the vector field of the limit vector field F.

The base case where z is synchronised corresponds to a situation like the one below:

$$\begin{tikzcd}
\mathbf{r}_{ij} \arrow[r] & \boxed{S_{ij}} \arrow[r, "z"] & f_z^q \\
\mathbf{r}_{i'j'} \arrow[r] & \boxed{S_{i'j'}} \arrow[r, "z"] & f_z^q
\end{tikzcd}$$

Now $f_z^q = q\min\{x_{ij}, x_{i'j'}\}$. Thus, we have the following ODE equations for x_{ij} and $x_{i'j'}$:

$$\begin{cases} \dot{x}_{ij} = \mathbf{r}_{ij} - f_z^q \\ \dot{x}_{i'j'} = \mathbf{r}_{i'j'} - f_z^q \end{cases}$$

We recall the the flow associated with z in the limit vector field (i.e. the outgoing flow from x_{ij} and $x_{i'j'}$) is

$$f_z = \begin{cases} \mathbf{r}_{i'j'} & \text{if } x_{ij} > 0 \wedge x_{i'j'} = 0 \qquad \text{(a)} \\ \mathbf{r}_{ij} & \text{if } x_{ij} = 0 \wedge x_{i'j'} > 0 \qquad \text{(b)} \\ \min\{\mathbf{r}_{ij}, \mathbf{r}_{i'j'}\} & \text{if } x_{ij} = 0 \wedge x_{i'j'} = 0 \qquad \text{(c)} \end{cases}$$

To prove the convergence of f_z^q to f_z we reason separately in cases (a)–(c) above. In case (a), observe that $\min\{x_{ij}, x_{i'j'}\} = x_{i'j'}$, so that the ODE for $x_{i'j'}$ is similar to the non-synchronised case. Notice that in the q-model we just need that the condition $\min\{x_{ij}, x_{i'j'}\} = x_{i'j'}$ to be satisfied, we do not require $x_{i'j'} = 0$, which may not hold. Reasoning as in the non-synchronised case, we obtain that $f_z^q \to \mathbf{r}_{i'j'} = f_z$, as required. The case (b) is clearly symmetric. Finally, to deal with (c), observe that this is the case in which $x_{ij} = x_{i'j'}$, and so both the equation for x_{ij} and for $x_{i'j'}$ assume the form of the non-synchronised case. Assuming they could evolve independently, they would reach the steady states $\tilde{x}_{ij} = \mathbf{r}_{ij}/q$ and $\tilde{x}_{i'j'} = \mathbf{r}_{i'j'}/q$. So we obtain

$$\tilde{f}_z^q = q\min\{\tilde{x}_{ij}, \tilde{x}_{i'j'}\} = \min\{\mathbf{r}_{ij}, \mathbf{r}_{i'j'}\} = f_z.$$

Now, when $q \to \infty$, the time to approach steady state is zero, hence the previous independence assumption holds in the limit and gives $f_z^q \to f_z$.

In order to prove the inductive step, i.e. the convergence of F_q to F, fixing slow variables, for those components corresponding to fast variables whose input flow is at depth greater than or equal to one in the reversed dependency graph. The generic situation for one such a node is depicted in the figure below, where $\mathbf{r}_{ij}^s$ is the component of the input flow due to delayed incoming events, while $f_{z_j}^q$ are the components of the input flow due to instantaneous events. The fact that the input flow $\mathbf{r}_{ij}$ is of this form follows from Definition 3.

$$\begin{tikzcd}
\mathbf{r}_{ij}^s \arrow[dr] & & \\
f_{z_1}^q \arrow[r, "z_1"] & \boxed{S_{ij}} \arrow[r, "z"] & f_z^q \\
\vdots & & \\
f_{z_k}^q \arrow[uur, "z_k"] & &
\end{tikzcd}$$

In case z is not synchronised, the ODE for x_{ij} is

$$\dot{x}_{ij} = \mathbf{r}_{ij}^s + f_{z_1}^q + \ldots + f_{z_k}^q - qx_{ij},$$

whose steady state is given by $\tilde{f}_z^q = \mathbf{r}_{ij}^s + \tilde{f}_{z_1}^q + \ldots + \tilde{f}_{z_k}^q = \mathbf{r}_{ij} + f_{z_1} + \ldots + f_{z_k} = f_z$. Hence, as $q \to \infty$, we have that $f_z^q \to f_z$. In proving the limit, we use the inductive hypothesis to show that $f_{z_i}^q \to f_{z_i}$. The synchronisation case then follows by combining this line of reasoning with a similar argument as the one for the base case.

Up to now, we have proved convergence of vector fields as $q \to \infty$, assuming the slow variables to have a fixed value. In order to extend this result to the convergence of trajectories up to time T, we can reason by first splitting the time interval $[0, T]$ into small intervals of length h. We further assume that all the variables for the limit model are constant in each such a small interval, while only the slow variables are constant for the fast model. We further approximate the trajectories by piecewise constant functions, according to the standard Euler scheme of step size h. Now, in each such an interval, we can apply the previous arguments to show that the vector fields will be equal, excluding an initial phase in which the fast model equilibrates. However, the duration of this phase tends to zero as q increases, which allows to prove that the total error caused by this equilibration will also go to zero. Furthermore, the error due to the discretisation step is independent of the error due to the equilibration phase, hence both can be made arbitrary small at the same time. This shows convergence of trajectories.

The argument we presented relies also on another implicit assumption, i.e. that for q large enough, and for each pair of synchronising fast variables, the trajectory of F_q visits the surfaces $\mathcal{H}_a$ and $\mathcal{H}_b$ in the same order as the limit system (see proof of Theorem 1 for a definition of those surfaces). However, if the solution of the limit vector field enters $\mathcal{H}_c$ always in a configuration falling into cases (a) and (b) above, then this property holds. Informally, this follows because the limit system will stay in $\mathcal{H}_c$ only in a single time point, and then enter immediately $\mathcal{H}_a$ or $\mathcal{H}_b$. Furthermore, because in $\mathcal{H}_c$ one input flow is smaller than the other, this will also hold (with the same ordering) for q large enough. In turn, this implies uniform converge of the vector fields along solutions, a part from a set of times of measure vanishing in the limit, as we assumed that the limit trajectory visits $\mathcal{H}_c$ only a countable number of times. As solutions are formally defined as integrals along the vector field, they are insensitive to changes in the values of the vector field in a set of measure zero.