On-the-fly Fluid Model Checking via Discrete Time Population Models*

Diego Latella[1], Michele Loreti[2], and Mieke Massink[1]

[1] Istituto di Scienza e Tecnologie dell'Informazione 'A. Faedo', CNR, Italy
[2] Dip. di Statistica, Informatica, Applicazioni 'G. Parenti', Università di Firenze, and IMT Advanced Studies Lucca, Lucca, Italy

Abstract. We show that, under suitable convergence and scaling conditions, fluid model checking bounded CSL formulas on selected individuals in a continuous *large* population model can be approximated by checking equivalent bounded PCTL formulas on corresponding objects in a discrete time, time synchronous Markov population model, using an *on-the-fly* mean field approach. The proposed technique is applied to a benchmark epidemic model and a client-server case study showing promising results also for the challenging case of nested formulas with time dependent truth values. The on-the-fly results are compared to those obtained via global fluid model checking and statistical model-checking.

1 Introduction

Model checking has been widely recognised as a powerful approach to the automatic verification of concurrent and distributed systems, including aspects of their performance. It consists of an efficient procedure that, given an abstract model $\mathcal{M}$ of the system, decides whether $\mathcal{M}$ satisfies a logical formula Φ, typically drawn from a temporal logic. Recently, the integration of mean field and fluid approximation techniques, that originate in statistical physics, with formal modelling techniques has received increasing attention as a way to obtain *highly scalable* approximate model checking techniques, such as fluid model checking [3, 4, 12] and mean field model checking [16]. These approaches are *independent* of the population size, as long as this is large enough. Such extreme scalability is a prerequisite for the verification of large scale collective adaptive systems, of which performance aspects and emerging behaviour are an essential feature.

Traditional model checking approaches do not scale up to such large systems due to the well-known state space explosion problem. Statistical model-checking [21] is in general performing much better in this respect. It avoids the generation of the state space and approximates the results by a statistical analysis of a number of randomly generated finite executions of the model. This leads to better scalability, but the complexity is still linear in the number of objects that the system is composed of. Furthermore, depending on the accuracy

* This work is partially supported by the EU project QUANTICOL (nr. 600708), and the IT MIUR project CINA

required and the particular property of interest, it may be necessary to take a large number of samples into consideration.

Fluid model checking [3, 4, 12] relies on a global model checking approach for time-inhomogeneous Continuous Time Markov Chains (ICTMC) representing a typical individual object in the context of a large CTMC population model. The rates of the individual may depend on the fraction of the population that is in a particular state. The algorithm relies on the deterministic approximation of the average stochastic behaviour of the system in continuous time, i.e. a fluid approximation. An alternative approach is the one we refer to in this paper by *on-the-fly mean field model checking* [16]. In this approach only as much of the state space as strictly needed to verify the given formula is generated from a high-level specification of the individual behaviour and the population. The algorithm relies on the deterministic approximation of the probabilistic behaviour of the population in discrete time and can be used to verify formulas of the bounded Probabilistic Computation Tree Logic (PCTL)[9].

The main contribution of the present paper is to show that, under suitable convergence and scaling assumptions[3], and for models that are not too stiff[4], fluid model checking can be performed exploiting on-the-fly mean field model checking techniques [16] applied on a time-inhomogeneous *Discrete Time* Markov Chain (IDTMC) model and bounded PCTL formula that are derived from a corresponding ICTMC model and bounded Continuous Stochastic Logic (CSL) [1] formula via a transformation presented in Sect. 2.3. This approach is interesting and differs from other approaches in several respects: 1) the mean field model checking algorithm is implemented as a particular instantiation of an *on-the-fly probabilistic model-checking algorithm* [16]; 2) the latter is *parametric w.r.t. the semantics interpretation* of the model specification language and in this case we instantiate it on the mean-field approximation of a simple probabilistic population description language; 3) the transformation presented in Sect. 2.3 allows one to reuse the implementation once more for a class of CTMC population models; 4) the global fluid model checking algorithm in [3, 4] requires the *a priori* calculation of discontinuity points, i.e. points in time in which the truth values of time-dependent (sub)-formulas of an until formula change. This is a non-trivial task and consists in finding all zeros of an analytic function. In the on-the-fly setting such points are detected automatically during the computation of the probabilities, upto a difference that is in the order of a small discrete step size; 5) on-the-fly approaches are particularly efficient when verifying conditional reachability properties because in that case much fewer states need to be generated. Ultimately, however, the on-the-fly mean field algorithm is based on an Euler method to solve differential equations. This poses certain limitations on the continuous time models that can be analysed efficiently this way, in particular they should not be too stiff. For non stiff models the results are promising as shown by the available benchmark models for which also some results for global fluid model checking and statistical model checking are available in the literature.

[3] See Theorem 5 of [4].

[4] Stiff models are those whose rates differ several orders of magnitude.

The outline of the paper is as follows. Sect. 2 introduces discrete and continuous time Markov population models. The relevant temporal logics are recalled in Sect. 3. Sect. 4 presents the model and logic transformation functions and the correctness results w.r.t. fluid model-checking. Sect. 5 provides a comparison with benchmark examples from the literature. Related work and conclusions are presented in Sect. 6 and Sect. 7, respectively. Basic knowledge on Markov chains and related model checking algorithms is assumed.

2 Population Models

We consider two types of Markov population models: continuous time models and discrete time models. In both models we assume that the number of objects in the population is N and that this size remains constant during execution.

2.1 Continuous Time Population Models

For CTMC population models we adopt the notation following [3]. Let $Y_i^{(N)}(t) \in \mathcal{S}$ be the random variable representing the state of object i at time t, where $\mathcal{S} = \{1, 2, \ldots, n\}$ represents the local state space of each object. Multiple classes of agents are represented by partitioning $\mathcal{S}$ into disjoint subsets and allowing state changes only within a single class. Let $\mathbf{Q}^{(N)}(\boldsymbol{x})$ denote the $n \times n$ infinitesimal generator matrix that depends on the fraction of objects $\boldsymbol{x} \in [0,1]^n$ that are in each state. The latter quantity can be computed from $Y_i^{(N)}$ as $\hat{X}_i^{(N)}(t) = \frac{1}{N}\sum_{j=1}^{N} \mathbf{1}\{Y_j^{(N)}(t) = i\}$[5]. $\langle \hat{X}_1^{(N)}(t), \hat{X}_2^{(N)}(t), \ldots, \hat{X}_n^{(N)}(t)\rangle$ is a CTMC $\hat{\mathcal{X}}^{(N)}$ [2] on the state space $[0,1]^n$, also called the *occupancy measure*, with initial state $\boldsymbol{x}_0^{(N)} \in [0,1]^n$. The average infinitesimal variation of $\hat{\mathcal{X}}^{(N)}$, given that it is in state $\boldsymbol{x}$ is $F^{(N)}(\boldsymbol{x}) = \boldsymbol{x}^T\mathbf{Q}^{(N)}(\boldsymbol{x})$, also called the drift[6]. If, for $N \to \infty$, $\mathbf{Q}^{(N)}(\boldsymbol{x})$ converges uniformly to the Lipschitz continuous generator matrix $\mathbf{Q}(\boldsymbol{x})$, and $\boldsymbol{x}_0^{(N)}$ to $\boldsymbol{x}_0$, and, furthermore, if $\boldsymbol{x}(t)$ is the solution of the ODE $\frac{d\boldsymbol{x}}{dt} = F(\boldsymbol{x}) = \boldsymbol{x}^T\mathbf{Q}(\boldsymbol{x})$ for initial condition $\boldsymbol{x}(0) = \boldsymbol{x}_0$, then in the limit the two processes behave almost surely the same for a finite time horizon T [7, 13][7].

It is possible to decouple the analysis of a single object from the analysis of the global system by letting the behaviour of the single object depend on the other objects only through the solution of the fluid ODE. This result is known as *fast simulation* [7, 18]. The stochastic behaviour of a single object can be defined as $Z^{(N)} = Y_1^{(N)}$ on state space $\mathcal{S}$, assuming, without loss of generality, we are interested in the behaviour of the first object. Note that $Z^{(N)}$ is an ICTMC. Let $z(t)$ be the ICTMC of an individual object with states in $\mathcal{S}$ such that $\Pr\{z(t+dt) = j | z(t) = i\} = q_{i,j}(\boldsymbol{x}(t))dt$, and let $\mathbf{Q}_z(\boldsymbol{x}(t)) = (q_{i,j}(\boldsymbol{x}(t)))$.

[5] $\mathbf{1}\{x = y\}$ yields 1 if $x = y$ and 0 otherwise.

[6] $\boldsymbol{x}^T$ denotes the transpose of vector $\boldsymbol{x}$.

[7] The conditions on uniform convergence and Lipschitz continuity automatically hold for PEPA population models because in that case the rate functions are all piecewise linear [20].

We then have that for any finite horizon T and $t \leq T$ the behaviour of the single object $Z^{(N)}(t)$ tends to the behaviour of the object that senses the rest of the system only through its limit behaviour given by $\boldsymbol{x}$, i.e. $z(t)$.

Running example: Consider the simple PEPA specification of processors and resources that synchronise on a common task [11]:

$$\begin{array}{ll} Proc0 := (task1, r_1).Proc1 & Res0 := (task1, r_1).Res1 \\ Proc1 := (task2, r_2).Proc0 & Res1 := (reset, s).Res0 \end{array}$$
$$Proc0[N_p] \underset{task1}{\bowtie} Res0[N_q]$$

where $Proc0[N_p]$ is a shorthand notation for N_p instances of process $Proc0$ in parallel, and $Res0[N_q]$ denotes N_q instances of process $Res0$ in parallel. Such population oriented PEPA specifications have been given a formal semantics based on ODE by Hillston in [11] and by Tribastone et al. in [20]. In particular, the ODE associated to the example specification can be given as:

$$\begin{array}{l} \frac{d\, proc0(t)}{dt} = -r_1.min(proc0(t), res0(t)) + r_2.proc1(t) \\ \frac{d\, proc1(t)}{dt} = -r_2.proc1(t) + r_1.min(proc00(t), res0(t)) \\ \frac{d\, res0(t)}{dt} = -r_1.min(proc0(t), res_0(t)) + s.res1(t) \\ \frac{d\, res1(t)}{dt} = -s.res_1(t) + r_1.min(proc0(t), res0(t)) \end{array}$$

where $proc0(t)$, $proc1(t)$, $res0(t)$ and $res1(t)$ denote the limit occupancy measure at time t for each local state respectively. The function min denotes the minimum function and originates from the specific definition of action synchronisation of the semantics of PEPA [11].

The infinitesimal $\mathbf{Q}$-matrix of an individual object that depends on the behaviour of the global system via its limit occupancy measure can be retrieved as follows (see [3]). From the PEPA semantics of the synchronisation (cooperation) operator we know that the total rate of a shared $task1$ action is given by $min(r_1.proc_0(t), r_1.res_0(t))$. The rate of an *individual process* performing a $task1$ action is then this global rate divided by the fraction of objects present in the system at time t, i.e. $proc_0(t)$. The rate of an individual process performing a $task_2$ action is simply r_2 because this action does not depend on the limit occupancy measure $\boldsymbol{x}$, where $\boldsymbol{x}^T(t) = (proc_0(t), proc_1(t), res_0(t), res_1(t))$. Similar reasoning applies to the rates of a resource object. So, we obtain the following rate functions for the $\mathbf{Q}$-matrix:

$$\begin{array}{l} \mathbf{Q}_{proc0,proc1}(\boldsymbol{x}(t)) = r_1.min(proc0(t), res0(t))/proc0(t) \\ \mathbf{Q}_{proc1,proc0}(\boldsymbol{x}(t)) = r_2 \end{array}$$

$$\begin{array}{ll} \mathbf{Q}_{res0,res1}(\boldsymbol{x}(t)) & = r_1.min(proc0(t), res0(t))/res0(t) \\ \mathbf{Q}_{res1,res0}(\boldsymbol{x}(t)) & = s \end{array}$$

The rate functions used in the $\mathbf{Q}$-matrix are all continuous and bounded, at least as long as we do not divide by zero.

2.2 Discrete Time Population Models

For DTMC population models we consider again a system of N interacting objects. Let $W_i^{(N)}(k) \in \mathcal{S}$ be the random variable representing the state of object i at step k, where $\mathcal{S} = \{1, 2, \ldots, n\}$ represents the local state space of each object. Let $\mathbf{K}^{(N)}(\boldsymbol{m})$ denote the $n \times n$ one step transition probability matrix that depends on the fraction of objects $\boldsymbol{m} \in [0,1]^n$ that are in each state of $\mathcal{S}$. This fraction can be computed as $\hat{M}_i^{(N)}(k) = \frac{1}{N}\sum_{j=1}^{N} \mathbf{1}\{W_j^{(N)}(k) = i\}$. It is easy to see that the process $\langle \hat{M}_1^{(N)}(k), \hat{M}_2^{(N)}(k), \cdots, \hat{M}_n^{(N)}(k)\rangle$ is a DTMC $\hat{\mathcal{M}}^{(N)}$ on the state space $[0,1]^n$, with initial state $\boldsymbol{m}_0^{(N)} \in [0,1]^n$.

The average variation of $\hat{\mathcal{M}}^{(N)}$, given that it is in state $\boldsymbol{m}$ is $F^{(N)}(\boldsymbol{m}) = \boldsymbol{m}^T\mathbf{K}^{(N)}(\boldsymbol{m})$. Suppose that, for all i, j and for $N \to \infty$, the elements $\mathbf{K}_{i,j}^{(N)}(\boldsymbol{m})$ converge uniformly in $\boldsymbol{m}$ to some $\mathbf{K}_{i,j}(\boldsymbol{m})$, which is a continuous function of $\boldsymbol{m}$, and $\boldsymbol{m}_0^{(N)}$ converges almost surely to $\boldsymbol{m}_0$, and furthermore define $\boldsymbol{m}(k)$ as follows: $\boldsymbol{m}(0) = \boldsymbol{m}_0$ and $\boldsymbol{m}(k+1) = \boldsymbol{m}(k)^T \cdot \mathbf{K}(\boldsymbol{m}(k))$; then, for any fixed step t, almost surely $\hat{\mathcal{M}}^{(N)}$ converges to function $\boldsymbol{m}(k)$ [18]. As for CTMC population models, it is possible to decouple the analysis of the single object from the analysis of the global system using a fast simulation approach involving the solution of a difference equation rather than an ODE.

Example: Taking probabilities α_i for the rates r_i in the processes and resources example, we obtain the following difference equations for $\boldsymbol{m}^T(k) = (m_{p0}(k), m_{p1}(k), m_{r0}(k), m_{r1}(k)))$:

$$
\begin{aligned}
m_{p0}(k+1) &= m_{p0}(k) - \alpha_1.min(m_{p0}(k), m_{r0}(k)) + \alpha_2.m_{p1}(k)\\
m_{p1}(k+1) &= m_{p1}(k) + \alpha_1.min(m_{p0}(k), m_{r0}(k)) - \alpha_2.m_{p1}(k)\\
m_{r0}(k+1) &= m_{r0}(k) - \alpha_1.min(m_{p0}(k), m_{r0}(k)) + \alpha_s.m_{r1}(k)\\
m_{r1}(k+1) &= m_{r1}(k) + \alpha_1.min(m_{p0}(k), m_{r0}(k)) - \alpha_s.m_{r1}(k)
\end{aligned}
$$

where $m_{pj}(k)$ and $m_{rj}(k)$ denote the limit occupancy measure at step k for processes and resources. We can also retrieve the one step probability matrix for each individual process and resource object using a similar reasoning as in the CTMC case:

$$
\begin{aligned}
\mathbf{K}_{p0,p1}(\boldsymbol{m}(k)) &= \alpha_1.min(m_{p0}(k), m_{r0}(k))/m_{p0}(k)\\
\mathbf{K}_{p1,p0}(\boldsymbol{m}(k)) &= \alpha_2\\
\\
\mathbf{K}_{r0,r1}(\boldsymbol{m}(k)) &= \alpha_1.min(m_{p0}(k), m_{r0}(k))/m_{r0}(k)\\
\mathbf{K}_{r1,r0}(\boldsymbol{m}(k)) &= \alpha_s
\end{aligned}
$$

The difference equations can be obtained from $\mathbf{K}$ by $\boldsymbol{m}(k+1) = \boldsymbol{m}(k)^T.\mathbf{K}(\boldsymbol{m}(k))$.

2.3 Relationship Between the Models

First note that we can interpret the difference equations obtained from a discrete time population model as an instance of the Euler forward method for approximating the solution of a set of ODEs. The set of ODEs we are interested

in solving are those of a corresponding continuous population model. To obtain an acceptable approximation of the solution we need to find a step size for the difference equations such that absolute stability of the method, to avoid that the global error grows exponentially, and a sufficient accuracy [19] are guaranteed. This, in turn, means that we need to derive suitable values for the probabilities α from the rates in the continuous model. What we would be even more interested in is to transform an ICTMC model of an individual (from which the ODEs can be derived) into an IDTMC model, with the same local states and jump structure as the ICTMC, from which we get exactly the set of difference equations that can be used to approximate the solution of the ODEs. We proceed as follows. Using a feature of CTMC uniformisation[8] we can obtain a DTMC with probability matrix $\mathbf{K}$ such that $\mathbf{K} = \mathbf{I} + \frac{1}{q} \cdot \mathbf{Q}$, where $\mathbf{Q}$ is the infinitesimal rate matrix and q the uniformisation rate that is *at least as large* as the maximal exit rate of the states in the original CTMC. This DTMC preserves the local states and the jumps of the original CTMC apart from additional self-loops. Note that in our case the rates in $\mathbf{Q}$ may depend on the occupancy measure $\boldsymbol{m}$. However, $0 \leq m_i \leq 1$ for all $i \in |\mathcal{S}|$, so assuming rate-functions that include minimum functions and linear combination[9] (but not rational functions) that we derive from PEPA specifications we can easily find a suitable q.

At this point we need to satisfy also the requirements of absolute stability and obtain a satisfactory accuracy following standard procedures [19]. If we are lucky, q is already large enough so that these requirements are fulfilled, otherwise we need to increase q, which is allowed because q is only a lower bound (see above). For linear systems of l differential equations where $u(t) \in \mathbb{R}^l$ and $d\,u(t)/dt = A \cdot u(t)$ where A is an $l \times l$ matrix a necessary condition is that $h\lambda$ is in the stability region of the Euler method for each eigenvalue λ of matrix A and step size h. So, for each eigenvalue λ we need that $-2 \leq h\lambda \leq 0$ [19]. For non-linear systems we need to determine the range of each eigenvalue and make sure that the step size h is taken small enough so that $h\lambda$ stays within the region of absolute stability for its complete range. Note that $h = 1/q$.

Example: For the running example, with $r_1 = 10, r_2 = 3.0$ and $s = 7.0$, we obtain uniformisation rate $q = 10$ and eigenvalues $\lambda_1 = 0$ or $-r_1 - r_2 \leq \lambda_1 \leq -r_2$ and $\lambda_2 = 0$ or $-r1 - s \leq \lambda_2 \leq -s$, showing that all eigenvalues are in a bounded range, with a maximum absolute value of 17. So when taking $h = 1/q$ we get that $0 \leq 17 * 1/q \leq 2$. This implies that $q = 10$ guarantees stability of the method, but a higher value may be preferred for better accuracy.

3 Properties of Individual Objects

Properties of the behaviour of individuals in the context of a large population model can be expressed as formulas of a suitable temporal logic. For the purpose

[8] I.e. transforming a CTMC into one where each state has the same exit rate by adding self-loops where needed, which is an operation that does not alter the transient and steady state properties of the CTMC.

[9] I.e. piecewise linear functions leading to the class of split-free PEPA models [10].

$s,t \models_{\mathcal{C}} a$	$\Leftrightarrow a \in \ell(s)$
$s,t \models_{\mathcal{C}} \neg\Phi$	$\Leftrightarrow$ not $s,t \models_{\mathcal{C}} \Phi$
$s,t \models_{\mathcal{C}} \Phi_1 \vee \Phi_2$	$\Leftrightarrow s,t \models_{\mathcal{C}} \Phi_1$ or $s,t \models_{\mathcal{C}} \Phi_2$
$s,t \models_{\mathcal{C}} \mathcal{P}_{\bowtie p}(\varphi)$	$\Leftrightarrow \Pr\{\sigma \in Paths_{\mathcal{C}}(s,t) \mid \sigma,t \models_{\mathcal{C}} \varphi\} \bowtie p$
$\sigma,t \models_{\mathcal{C}} \Phi_1 \mathcal{U}^{\leq\tau} \Phi_2$	$\Leftrightarrow \exists\, \tau_2$ s.t. $0 \leq \tau_2 \leq \tau, \sigma@\tau_2, t+\tau_2 \models_{\mathcal{C}} \Phi_2 \ \wedge$ $\forall\, 0 \leq \tau_1 < \tau_2,\ \sigma@\tau_1, t+\tau_1 \models_{\mathcal{C}} \Phi_1$

Table 1. Satisfaction relation: CSL fragment.

of this paper, properties of continuous time models are expressed in bounded CSL, and properties of discrete time models are expressed in bounded PCTL. Both are briefly recalled in this section, where we assume set $\mathscr{P}$ of atomic propositions is given and $a \in \mathscr{P}$, $\tau \in \mathbb{Q}_{\geq 0}$, $k \in \mathbb{N}$ and $\bowtie \in \{>, <\}$ and $p \in [0,1] \cap \mathbb{Q}$.

Continuous Stochastic Logic for ICTMC. The syntax of the fragment of bounded CSL we consider is defined below:

$$\Phi ::= a \mid \neg\Phi \mid \Phi \vee \Phi \mid \mathcal{P}_{\bowtie p}(\varphi) \text{ where } \varphi ::= \Phi\, \mathcal{U}^{\leq\tau}\, \Phi.$$

CSL formulas[10] are interpreted over state labelled ICTMCs $\langle \mathcal{C}, \ell \rangle$, where $\mathcal{C}$ is an ICTMC with state set $\mathcal{S}$ and $\ell : \mathcal{S} \to 2^{\mathscr{P}}$ associates each state with a set of atomic propositions. We define the satisfaction relation on $\mathcal{C}$ and the logic in Table 1. We abbreviate $\langle \mathcal{C}, \ell \rangle$ with $\mathcal{C}$, when no confusion can arise, with $\mathbf{Q}$ its infinitesimal generator matrix. A path σ over $\mathcal{C}$ is a non-empty sequence $s_0 \xrightarrow{t_0} s_1 \xrightarrow{t_1} \cdots$ such that the probability of going from state s_i to s_{i+1} at time $T_i = \sum_{j=0}^{i} t_i$ is positive for all $i \geq 0$. We let $Paths_{\mathcal{C}}(s,t)$ denote the set of all infinite paths over $\mathcal{C}$ starting from state s at time t. We require that all subsets of paths considered are measurable. We let $\sigma@t$ denote the state s_k in σ such that k is the maximum i such that $\sum_{j=0}^{i} t_i \leq t$, i.e. the state reached on path σ at time t. Finally, in the sequel we will consider ICTMCs equipped with an initial state s_0, i.e. the probability mass is initially all in s_0.

Probabilistic Logic for DTMC.[11] The syntax of the fragment of bounded PCTL we consider is defined below:

$$\Phi ::= a \mid \neg\Phi \mid \Phi \vee \Phi \mid \mathcal{P}_{\bowtie p}(\varphi) \text{ where } \varphi ::= \Phi\, \mathcal{U}^{\leq k}\, \Phi.$$

PCTL formulas are interpreted over *state labelled* DTMCs $\mathcal{D}$ in a similar way as for CTMCs. We assume $\mathbf{P}$ to be the one step probability matrix for $\mathcal{D}$. A path σ over $\mathcal{D}$ is a non-empty sequence of states $s_0, s_1, \cdots$ where $\mathbf{P}_{s_i,s_{i+1}} > 0$ for all $i \geq 0$. We let $Paths_{\mathcal{D}}(s)$ denote the set of all infinite paths over $\mathcal{D}$ starting from

[10] For simplicity the time bounds in the formulas are of the form $[0,\tau]$ instead of the more general $[\tau_1,\tau_2]$.

[11] Note that, by making time explicit, the structures used by FlyFast are DTMCs rather than IDTMCs (see Sect. 4).

$s \models_{\mathcal{D}} a$	$\Leftrightarrow a \in \ell(s)$
$s \models_{\mathcal{D}} \neg\Phi$	$\Leftrightarrow$ not $s \models_{\mathcal{D}} \Phi$
$s \models_{\mathcal{D}} \Phi_1 \vee \Phi_2$	$\Leftrightarrow s \models_{\mathcal{D}} \Phi_1$ or $s \models_{\mathcal{D}} \Phi_2$
$s \models_{\mathcal{D}} \mathcal{P}_{\bowtie p}(\varphi)$	$\Leftrightarrow \Pr\{\sigma \in Paths_{\mathcal{D}}(s) \mid \sigma \models_{\mathcal{D}} \varphi\} \bowtie p$
$\sigma \models_{\mathcal{D}} \Phi_1 \mathcal{U}^{\leq k} \Phi_2$	$\Leftrightarrow \exists\, 0 \leq h \leq k \text{ s.t. } \sigma[h] \models_{\mathcal{D}} \Phi_2 \ \wedge\ \forall\, 0 \leq i < h \,.\, \sigma[i] \models_{\mathcal{D}} \Phi_1$

Table 2. Satisfaction relation: PCTL fragment.

state s. By $\sigma[i]$ we denote the element s_i of path σ. We will consider DTMCs equipped with an initial state s_0. We define the satisfaction relation on $\mathcal{D}$ and the logic in Table 2.

4 Fluid Model Checking via Discrete Time Models

We first define two transformation functions. Function $\mathcal{T}_M$ takes an ICTMC $z(t)$ with infinitesimal generator matrix $\mathbf{Q}(t)$ and initial state s_0. It takes a step size $d \in \mathbb{Q}$ and a time bound $b > d$. It returns a DTMC with state set $\mathcal{S} \times \{0, \ldots, \lfloor \frac{b}{d} \rfloor\}$, initial state $(s_0, 0)$ and one step transition probability matrix $\mathbf{U}$, as follows:

Definition 1. *For all $0 < d \in \mathbb{Q}, b \in \mathbb{R}$ with $b > d$, and infinitesimal generator matrix $\mathbf{Q}(t)$, $\mathcal{T}_M(\mathbf{Q}(t), d, b)$ is the one step transition probability matrix $\mathbf{U}$:*

$$\mathbf{U}_{(s,i),(s',i')} = \begin{cases} [\mathbf{I} + d{\cdot}\mathbf{Q}(i{\cdot}d)]_{s,s'}, \textit{ if } i' = i{+}1, \mathbf{Q}(i{\cdot}d)_{s,s} \neq 0, \\ 1, \textit{ if } i' = i, s' = s, \mathbf{Q}(i{\cdot}d)_{s,s} = 0, \\ 0, \textit{ otherwise} \end{cases}$$

where the indexes of $\mathbf{U}$ are assumed to be ordered as follows:

$$(s_0, 0), ..., (s_n, 0), (s_0, 1), ..., (s_n, 1), ..., (s_0, \lfloor \frac{b}{d} \rfloor), ..., (s_n, \lfloor \frac{b}{d} \rfloor).$$

Function $\mathcal{T}_F$ below transforms bounded CSL into bounded PCTL formulas:

Definition 2. *For atomic propositions a, bounded CSL formulas Φ, Φ_1 and Φ_2, and $d \in \mathbb{Q}$, function $\mathcal{T}_F$ is defined as follows:*

$\mathcal{T}_F[\![a]\!]_d = a$ $\quad$ $\mathcal{T}_F[\![\Phi_1 \vee \Phi_2]\!]_d = \mathcal{T}_F[\![\Phi_1]\!]_d \vee \mathcal{T}_F[\![\Phi_2]\!]_d$

$\mathcal{T}_F[\![\neg\Phi]\!]_d = \neg\mathcal{T}_F[\![\Phi]\!]_d$ $\quad$ $\mathcal{T}_F[\![\mathcal{P}_{\bowtie p}(\Phi_1 \,\mathcal{U}^{\leq \tau}\, \Phi_2)]\!]_d = \mathcal{P}_{\bowtie p}(\mathcal{T}_F[\![\Phi_1]\!]_d \,\mathcal{U}^{\leq \lfloor \frac{\tau}{d} \rfloor}\, \mathcal{T}_F[\![\Phi_2]\!]_d)$

The definition for the basic state formulas is straightforward. Bounded CSL until formulas translate to bounded PCTL until formulas with the same probability bound and structure, but with a time bound $\frac{\tau}{d}$ where τ was the original time bound in the CSL formula. In the sequel, we let $|\Phi|$ denote the *duration* of Φ, i.e. the length of time to which it refers, as follows:

Definition 3. *For any bounded CSL formula Φ the duration of Φ, $|\Phi|$ is defined as follows:*

$|a| = 0$ $\quad$ $|\Phi_1 \vee \Phi_2| = \max\{|\Phi_1|, |\Phi_2|\}$

$|\neg\Phi| = |\Phi|$ $\quad$ $|\mathcal{P}_{\bowtie p}(\Phi_1 \,\mathcal{U}^{\leq \tau}\, \Phi_2)| = \tau + \max\{|\Phi_1|, |\Phi_2|\}$

Model	CSL Property	FlyFast ($N > 500$)	Fluid MC ($N > 500$)	SMC (N=100)	SMC (N=1000)
SEIR	$s\mathcal{U}^{\leq T} r$	$\sim$ 0.005 s	$\sim$0.05 s	$\sim$ 5 s	$\sim$ 20 s
SEIR	tt $\mathcal{U}^{\leq T} \mathcal{P}_{\leq 0.01}(true\ \mathcal{U}^{\leq 10}\ i\ )$	$\sim$ 6.3 s	N.A.	N.A.	N.A.
CS	tt $U^T(P_{\leq 0.167}[\text{tt}\ U^{50} CR])$	$\sim$ 63.9 s	N.A.	N.A.	N.A.

Table 3. Comparison of model checking times. Times for Statistical MC (SMC) based on 10000 runs. Data for SMC and Fluid MC from [5]. Time in seconds.

Recall that we assume that time bounds in until formulas are rational numbers. For formula Φ, we let $\boldsymbol{\tau}_\Phi = (\tau_1, \ldots, \tau_l)$, denote the vector of all time bounds occurring in the (until subformulae of) Φ; furthermore we define d_Φ and D_Φ as follows: $d_\Phi = \max\{d \in \mathbb{Q} \mid \frac{\tau_j}{d} \in \mathbb{N}, \text{ for } j = 1 \ldots l\}$ and $D_\Phi = \{d \in \mathbb{Q} \mid \text{ there exists } w \in \mathbb{N} \text{ s.t. } d = \frac{d_\Phi}{w}\}$. Note that d_Φ is well defined since $\tau_j \in \mathbb{Q}$, for $j = 1 \ldots l$; actually, letting $\tau_i = \frac{a_i}{b_i}$, s.t. $\gcd(a_i, b_i) = 1$, it is easy to see that $d_\Phi = \frac{1}{\text{lcm}(b_1, \ldots, b_l)}$. We are now ready to state the main Theorem for robust CSL formulas[12]:

Theorem 1. *Let $\mathcal{X}^{(N)}$ be a sequence of CTMC population models, with deterministic fluid limit $\boldsymbol{x}(t)$ for any fixed time $t < T$, under initial condition $\boldsymbol{x}(0) = \boldsymbol{x}_0$, and let $z = z(t)$ be the stochastic process defined from $\mathcal{X}^{(N)}$ as in Sect. 2.1. Let Φ be a robust CSL formula for z. There exists $N_0 \in \mathbb{N}$, such that, for all $d \in D_\Phi$, with $d \leq \frac{1}{q}$ as in Sect. 2.3 and for all $N \geq N_0$ and $b > \lceil \frac{|\Phi|}{d} \rceil$ the following holds:*

$$s, t \models_z \Phi \text{ iff } (s, \lfloor \frac{t}{d} \rfloor) \models_{\mathcal{T}_M(z,d,b)} \mathcal{T}_F[\![\Phi]\!]_d$$

The proof is by induction on the structure of the CSL formula Φ. One is usually interested in the result for $t = 0$. The proof is available in [14]. The result of Theorem 1 shows that it is indeed possible, under suitable scaling and convergence conditions, to use PCTL and a discrete time Markov population model to obtain similar results as by global fluid model checking CSL formulas on ICTMCs. The advantages and limitations of this approach were already outlined in the introduction.

Complexity. For what concerns the complexity of the approach, this depends on the complexity of the underlying on-the-fly probabilistic model-checking algorithm that consists of two phases, an expansion phase and a computation phase, both phases are linear in the number of states and transitions [17] for the time bounded fragment of PCTL. Furthermore it depends on the length and type of the formula, e.g conditional reachability is more likely to lead to the generation of fewer states, the time bounds in the formula and the uniformisation rate needed.

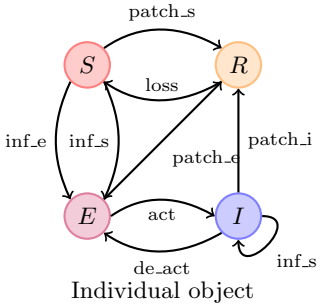
[12] We refer to [4] for the definition of formula robustness and to [13, 6] for constraints on time horizon T.

5 Benchmark Examples and Comparison

Fluid model-checking is a young field of research and to the best of our knowledge, the global fluid model checking algorithm has not been fully implemented as yet and is not publicly available; consequently we will only use the few benchmark examples available in the literature. More complex examples can be found in [15]. For those we compare results of global fluid model-checking, on-the-fly fluid model checking and statistical model checking for a computer epidemic model and a client-server model [5, 4, 3]. Our experiments were conducted with a 1.8 GHz Core i7 Intel processor and 4 GB RAM running Mac OS X 10.7.5. The results are summarised in Table 3.

5.1 A Computer Worm Epidemic Model

The computer worm epidemic model consists of a large number of nodes, each with four local states; susceptible (S), exposed (E), infected (I) and recovered (R) (see Fig. 1). Susceptible nodes can be infected by an external source (inf_e) or by other nodes that are already infected (inf_s) or, rarely, be patched (patch_s). Exposed nodes can be activated (act) and become infected, or they can be patched (patch_e). Infected nodes can be de-activated (de_act) or patched (patch_i), or infect other nodes (inf_s) while remaining infected. Recovered nodes can loose (loss) their protection and become susceptible.



Individual object

Probability functions:

action inf_e :: k_{ext}/q;
action inf_s :: $(k_{inf}/q) * (\mathsf{frc}\, I)$;
action act :: k_{act}/q;
action de_act :: k_{deact}/q;
action patch_i :: k_{high}/q;
action patch_s :: k_{low}/q;
action patch_e :: k_{low}/q;
action loss :: k_{loss}/q;

system worm = $< S[10000] >$;

Fig. 1. Computer Worm Epidemic Process and related rates: $k_{ext} = 0.01$, $k_{inf} = 5$, $k_{act} = k_{deact} = 0.1$, $k_{low} = 0.005$, $k_{high} = 0.1$, $k_{loss} = 0.005$ (left) and derived probability functions using uniformisation rate $q = 10$ (right).

Following the procedure outlined in Sect. 2.3 the model shown in Fig. 1 is transformed into a discrete time model using a suitable rate which guarantees absolute stability and sufficient accuracy of the Euler forward method. The highest exit rate is that of state S, namely 5.2, if we assume that in theory all nodes could be infected at some stage, such that $\mathsf{frc}\, I = 1$. However, to facilitate comparison with results in the literature we take $q = 10$, at the cost of being slightly

less efficient. Fig. 1 (right) shows the probabilities for the actions in the discrete time probabilistic model. Note that additional self-loops are added implicitly to the model to make sure that the total outgoing probability for each state is 1.

Fig. 2 shows the correspondence for model checking results (see also [5]) for all three model checking methods for the CSL path formula $s\,\mathcal{U}^{\leq T}\,r$, where s (r, resp.) denote the atomic propositions of being in state S (R, resp.), for T ranging from 0 to 20 showing the probability that a node is patched before being infected within T time steps. This corresponds to an equivalent PCTL formula with T ranging from 0 to 200. Model checking times for all methods for this formula are shown in Table 3. FlyFast generated 601 states. Note that this holds for any large number of nodes. FlyFast is faster than global fluid model-checking in this case. This is likely due to the fact that we are dealing with a *conditional* reachability property, so not all states need to be generated, showing the advantage of an on-the-fly approach. Furthermore, the algorithm uses memoization, meaning that probabilities computed once are preserved for later use. Note that both fluid model-checking approaches are several orders of magnitude faster than statistical model checking for a large population size, providing a scalability compatible with their use for analysing properties of individuals in the context of large scale collective systems, which is the main aim of the current work.

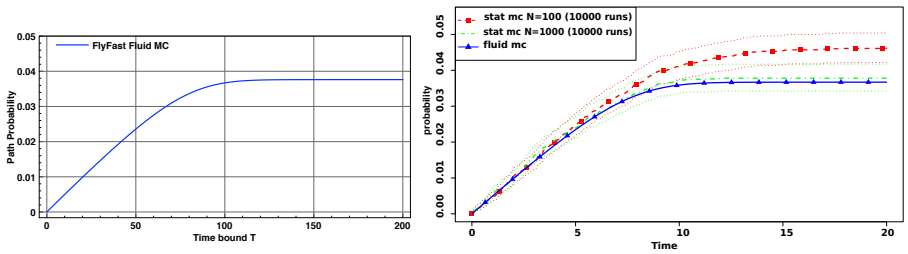


Fig. 2. Results for FlyFast (left) and Stat. MC and global Fluid MC (right).

An example of a nested path formula is $true\ \mathcal{U}^{\leq T}\,\mathcal{P}_{\leq 0.01}(true\ \mathcal{U}^{\leq 10}\ i\ )$, where i denotes to be in state I. It says that eventually, within time $T \in [0, 100]$, a state will be reached in which the probability to get infected within 10 time units is less than 0.01. In PCTL the formula is $true\ \mathcal{U}^{\leq T'}\,\mathcal{P}_{\leq 0.01}(true\ \mathcal{U}^{\leq 100}\ i\ )$, for T' ranging from 0 to 1000, given $q = 10$. Figures are omitted due to space limitations, but a comparison of results for a similar formula are shown in Fig. 4 for a more complex example. The FlyFast model-checking time is approximately 6.3 sec. and the number of states generated is 4000. No data on efficiency is available for the other two techniques for this formula.

5.2 A Client-Server Model

A larger model involving synchronisation is the Client-Server model [4, 3]. This model is composed of two populations of processes that synchronise on request and reply actions. A Client process (see Fig. 3) has initial state (CQ) in which it can only perform a request (rq) to the server and then waits (CW) for either a timeout (to) or a reply (rp) from the server to happen. After a timeout it goes to a state to recover (CR), and then returns to the initial state when recovery is completed (rc). After receiving a reply (rp) the Client enters a thinking state (CT) after which it returns to the initial state upon completing thinking (th). The Server process (see Fig. 3) is initially (SQ) ready to receive a request (rq) from a Client. If it receives it, either a timeout (top) may occur or it may process the request (pr) moving to the reply state (SR). From the latter it may either produce a timeout (tor) or deliver a reply (rp) to the client and in both cases the server moves to a log-state (SL) and afterwards returns to the initial state (SQ) upon completion of logging (lg). So, the behaviour of Clients and Servers are synchronising via actions rq and rp, using a PEPA-based interaction paradigm based on a minimum rate principle [4, 3]. The various timeouts are occurring independently. The ratio between the number of Clients and Servers is 2 to 1.

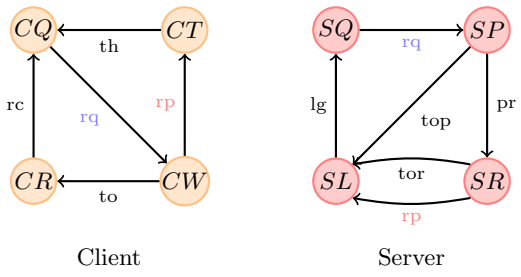


Fig. 3. Client and Server process. *Client* rates: th=1, rp=100, rq=1, rc=1, to=0.01; *Server* rates: rq=1, pr=0.1, top=0.005, tor=0.005, rp=1, lg=10.

As before, we proceed with the transformation of the model shown in Fig. 3 into a discrete time one, by finding a suitable rate. Note that the rate of an action shared by two types of objects can never be higher than the rate of the objects that contribute to the synchronisation, and will also be proportional to the (normalised) population size of the model. Therefore we can choose a rate equal to the maximum total exit rate of any of the states of the objects. In the client-server case this maximum is 100.01 (the sum of rates for Client actions rp and to). This is a large overestimation of the maximal exit rates, since the reply action of the Client is synchronised with that of the much slower Server (with reply rate 1). Therefore $q = 10$ is sufficient, given that the next highest rate is that of action lg which is 10. Table 4 shows the translation of the continuous time model into a discrete time model in the probabilistic input language of FlyFast. Actions of

different objects must be distinct which is achieved by appropriate prefixing ($c\_$ for Client actions and $s\_$ for Server ones). Fig. 4 (left) shows results concerning

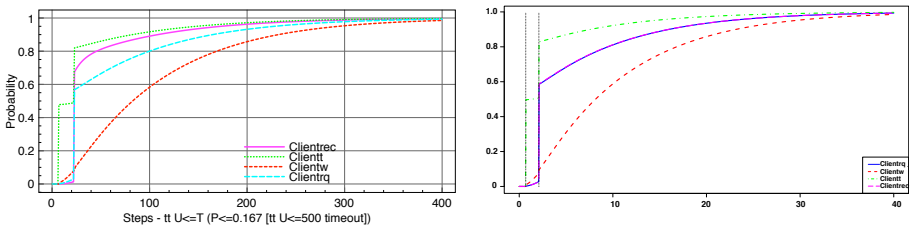


Fig. 4. FlyFast results (left) and global fluid model checking (right) from [3].

the nested, time-dependent PCTL (path) formula tt $U^T(P_{\leq 0.167}[\text{tt } U^{500}\, CR])$, concerning a Client timeout; the FlyFast results are given for T varying from 0 to 400 steps. Nested formulas are computationally the most complex to analyse. The property is the PCTL version of the corresponding CSL nested property analysed using the global fluid model checking approach in [4, 3]. Model checking times for FlyFast are shown in Table 3. This time is the cumulative time for the 400 different values for T considered, while it needed to generate 1598 states.

In the original model in [3], the recovery rate of the Client is 100 and not 10. This only affects the curve shown as a solid line in Fig. 4 (left), therefore the other curves continue to show close correspondence to ones on the right of Fig. 4 showing the fluid model checking results for the original model. The latter is a stiff model in which the parameter values differ five orders of magnitude. It would require a uniformisation rate of at least 100 and consequently a time bound of 5000 steps in the PCTL formula. The FlyFast results would still correspond well to the original global fluid model checking results, but model-checking times increase considerably. Global fluid model checking might be a viable alternative in that case because it can exploit existing highly optimised adaptive transient analysis methods, once the points in which the truth values change have been established.

6 Related Work

Closest to our work is that by Bortolussi and Hillston [4, 3] presenting a technique for global fluid model checking. We have briefly recalled some of the elements of this technique and compared their model checking results with those obtained by our on-the-fly technique and to statistical model checking where available. Work on fluid model checking can also be found in [12] which uses in part similar techniques as [4]. On-the-fly probabilistic model checking for bounded PCTL has also been developed by Della Penna et al. [8] but they do not consider its use for

action c_rq : $min(\alpha_{c\_rq} * \mathsf{frc}(CQ), \alpha_{s\_rq} * \mathsf{frc}(SQ))/\mathsf{frc}(CQ)$	state CQ $\{c\_rq.CW\}$
action c_to : α_{c_to}	state CW $\{c\_to.CR + c\_rp.CT\}$
action c_rc : α_{c_rc}	state CR $\{c\_rc.CQ\}$
action c_th : α_{c_th}	state CT $\{c\_th.CQ\}$
action c_rp : $min(\alpha_{c\_rp} * \mathsf{frc}(CW), \alpha_{s\_rp} * \mathsf{frc}(SR))/\mathsf{frc}(CW)$	
action s_rq : $min(\alpha_{c\_rq} * \mathsf{frc}(CQ), \alpha_{s\_rq} * \mathsf{frc}(SQ))/\mathsf{frc}(SQ)$	state SQ $\{s\_rq.SP\}$
action s_pr : α_{s_pr}	state SP $\{s\_pr.SR + s\_top.SL\}$
action s_tor : α_{s_tor}	state SR $\{s\_tor.SL + s\_rp.SL\}$
action s_top : α_{s_top}	state SL $\{s\_lg.SQ\}$
action s_rp : $min(\alpha_{c\_rp} * \mathsf{frc}(CW), \alpha_{s\_rp} * \mathsf{frc}(SR))/\mathsf{frc}(SR)$	
action s_lg : α_{s_lg}	
system $clientServerSystem =< CQ[1000], SQ[500] >$	

Table 4. Client and Server specification in FlyFast with $q = 10$ and the following values for the probabilities: $\alpha_{c\_th} = 1/q$, $\alpha_{c\_rp} = 100/q$, $\alpha_{c\_rq} = 1/q$, $\alpha_{c\_rc} = 1/q$, $\alpha_{c\_to} = 0.01/q$, $\alpha_{s\_rq} = 1/q$, $\alpha_{s\_pr} = 0.1/q$, $\alpha_{s\_top} = 0.005/q$, $\alpha_{s\_tor} = 0.005/q$, $\alpha_{s\_rp} = 1/q$, $\alpha_{s\_lg} = 10/q$.

on-the-fly *fast mean field* model checking as we did in [16]. Stochastic population models can also be analysed using statistical model checking methods based on simulation (see for example [21]). The computational complexity of the latter increase linearly with the number of objects N, whereas on-the-fly fast mean-field model checking and fluid model checking do not depend on N.

7 Conclusions

We have illustrated an alternative way to perform fluid model checking of bounded CSL properties of individual entities in the context of large CTMC population models. The framework makes use of a prototype implementation of the on-the-fly fast mean field model checker FlyFast to check bounded PCTL formulas of individuals in the context of synchronous, discrete time DTMC population models. We have provided a correctness result and shown promising verification results compared to those available in the literature. Future work will consist in integrating the method and related transformation functions into FlyFast, looking for further optimisations and investigating the possibility of generating error bounds along with the analysis results. We will also consider the extension of the fragment of the logic with intervals and further operators.

References

1. Baier, C., Haverkort, B., Hermanns, H., Katoen, J.P.: Model-Checking Algorithms for Continuous Time Markov Chains. IEEE Transactions on Software Engineering. IEEE CS 29(6), 524–541 (2003)
2. Benaïm, M., Le Boudec, J.: A class of mean field interaction models for computer and communication systems. Performance Evaluation 65(11-12), 823–838 (2008)
3. Bortolussi, L., Hillston, J.: Fluid model checking. CoRR abs/1203.0920 (2012), version 2 of Jan. 2013.

4. Bortolussi, L., Hillston, J.: Fluid model checking. In: Koutny, M., Ulidowski, I. (eds.) CONCUR. LNCS, vol. 7454, pp. 333–347. Springer-Verlag (2012)
5. Bortolussi, L., Hillston, J.: Checking individual agent behaviours in Markov population models by fluid approximation. In: Bernardo, M., de Vink, E.P., Di Pierro, A., Wiklicky, H. (eds.) SFM. Lecture Notes in Computer Science, vol. 7938, pp. 113–149. Springer (2013)
6. Bortolussi, L., Hillston, J., Latella, D., Massink, M.: Continuous approximation of collective system behaviour: A tutorial. Performance Evaluation 70(5), 317 – 349 (2013), http://www.sciencedirect.com/science/article/pii/S0166531613000023
7. Darling, R., Norris, J.: Differential equation approximations for Markov chains. Probability Surveys 5, 37–79 (2008)
8. Della Penna, G., Intrigila, B., Melatti, I., Tronci, E., Zilli, M.V.: Bounded probabilistic model checking with the mur*alpha* verifier. In: Hu, A.J., Martin, A.K. (eds.) FMCAD 2004. LNCS, vol. 3312, pp. 214–229. Springer (2004)
9. Hansson, H., Jonsson, B.: A logic for reasoning about time and reliability. Formal Aspects of Computing 6, 512–535 (1994)
10. Hayden, R.: Scalable Performance Analysis of Massively Parallel Stochastic Systems. Ph.D. thesis, Imperial College London (April 2011), http://pubs.doc.ic.ac.uk/hayden-thesis/
11. Hillston, J.: Fluid flow approximation of PEPA models. In: Proceedings of the Second International Conference on the Quantitative Evaluaiton of Systems (QEST 2005). pp. 33–43 (2005)
12. Kolesnichenko, A., de Boer, P.T., Remke, A., Haverkort, B.R.: A logic for model-checking mean-field models. In: DSN. pp. 1–12. IEEE (2013)
13. Kurtz, T.: Solutions of ordinary differential equations as limits of pure jump Markov processes. Journal of Applied Probability 7, 49–58 (1970)
14. Latella, D., Loreti, M., Massink, M.: On-the-fly fluid model checking via discrete time population models: Extended version. QUANTICOL TR-QC-08-2014, www.quanticol.eu (2014)
15. Latella, D., Loreti, M., Massink, M.: On-the-fly PCTL Fast Mean-Field Model-Checking for Self-organising Coordination. SCP (2015), http://dx.doi.org/10.1016/j.scico.2015.06.009
16. Latella, D., Loreti, M., Massink, M.: On-the-fly fast mean-field model-checking. In: Abadi, M., Lluch-Lafuente, A. (eds.) Trustworthy Global Computing - 8th International Symposium, TGC 2013, Buenos Aires, Argentina, August 30-31, 2013, Revised Selected Papers. LNCS, vol. 8358, pp. 297–314. Springer (2013), http://dx.doi.org/10.1007/978-3-319-05119-2_17
17. Latella, D., Loreti, M., Massink, M.: On-the-fly probabilistic model-checking. In: Proceedings 7th Interaction and Concurrency Experience ICE 2014. EPTCS, vol. 166 (2014)
18. Le Boudec, J.Y., McDonald, D., Mundinger, J.: A generic mean field convergence result for systems of interacting objects. In: QEST07. pp. 3–18. IEEE Computer Society Press (2007), iSBN 978-0-7695-2883-0
19. LeVeque, R.J.: Finite Difference Methods for Ordinary and Partial Differential Equations. SIAM (2007)
20. Tribastone, M., Gilmore, S., Hillston, J.: Scalable differential analysis of process algebra models. IEEE Trans. Software Eng. 38(1), 205–219 (2012)
21. Younes, H., Kwiatkowska, M., Norman, G., Parker, D.: Numerical vs. statistical probabilistic model checking: An empirical study. In: Jensen, K., Podelski, A. (eds.) Proc. 10th International Conference on Tools and Algorithms for the Construction and Analysis of Systems (TACAS'04). LNCS, vol. 2988, pp. 46–60. Springer (2004)