Model Checking Two Layers of Mean-Field Models

Anna Kolesnichenko, Anne Remke, Pieter-Tjerk de Boer and Boudewijn R. Haverkort

Abstract Recently, many systems that consist of a large number of interacting objects have been analysed using the mean-field method, which allows a quick and accurate analysis of such systems, while avoiding the state-space explosion problem. To date, the mean-field method has primarily been used for classical performance evaluation purposes. In this chapter, we discuss model-checking mean-field models. We define and motivate two logics, called *Mean-Field Continuous Stochastic Logic* (MF-CSL) and *Mean-Field Logic* (MFL), to describe properties of systems composed of many identical interacting objects. We present model-checking algorithms and discuss the differences in the expressiveness of these two logics and their combinations.

1 Introduction

Present-day computational technologies are massive and can cope with a huge amount of data. However, for modelling or simulation of a large system of interacting objects this computational power is often not enough. The mean-field method can be used to model such large systems efficiently. This method [1, 2] does not consider the state of each individual object separately. Instead, only their average

A. Kolesnichenko (✉)
UL Transaction Security Division, De Heyderweg 2, 2314 XZ Leiden, The Netherlands
e-mail: anna.kolesni4enko@gmail.com

A. Remke
Department of Computer Science, University of Münster, Einsteinstrasse 62, 48149 Münster, Germany
e-mail: Anne.Remke@wwu.de

P.-T. de Boer · B.R. Haverkort
Department of Computer Science, University of Twente, P.O. Box 217, 7500 AE Enschede, The Netherlands
e-mail: p.t.deboer@utwente.nl

B.R. Haverkort
e-mail: b.r.h.m.haverkort@utwente.nl

L. Fiondella and A. Puliafito (eds.), *Principles of Performance and Reliability Modeling and Evaluation*, Springer Series in Reliability Engineering,
DOI 10.1007/978-3-319-30599-8_13

behaviour, i.e. the fraction of objects in each possible state at any time, is considered. The model obtained possesses two levels: (i) the *local* level describes the behaviour of a random individual; (ii) the *global* level addresses the overall model behaviour. On the global level, the mean-field model computes the exact limiting behaviour of an infinite population of identical objects, which is a good approximation when the number of objects is not infinite but sufficiently large. Examples of systems for which the mean-field method has been successfully applied include gossiping protocols [3], disease spread between islands [4], peer-to-peer botnets spread [5] and many more.

Thus far, the mean-field method was primarily used for classical system performance measures, however, also more involved measures might be of interest. Therefore, methods for efficient and automated *model-checking* of such non-trivial measures (or properties) are essential and non trivial. One challenge lies in the fact that the model has two layers. Therefore, it is desirable to be able to formulate properties on both levels. Another challenge is that the local model is a *time-inhomogeneous* Markov chain (ICTMC). Therefore, the results of the model-checking procedure depend on time. Finally, the state-space of the global mean-field model is infinite. Hence, finding the satisfaction set is difficult.

In this chapter we discuss two logics, namely *Mean-Field Continuous Stochastic Logic* (MF-CSL) and *Mean-Field Logic* (MFL) together with the corresponding model-checking algorithms. While MF-CSL was proposed in Ref. [6], MFL is a valuable addition, since it enables reasoning about timed properties on the global level. MF-CSL first expresses the property of a random node in a system (including timed properties) and then lifts this to the system level using *expectation* operators. In contrast, MFL expresses the property of the whole system directly and it does not take into account the behaviour of the individual objects.

The contribution of this chapter is to adapt the existing logic *Signal Temporal Logic* (STL) [7] to mean-field models. This is done by defining *global atomic properties* of the mean-field model, which define binary trajectories based on the real-valued behaviour of the mean-field model. Moreover, three ways to approximate the full satisfaction set of an MFL formula are discussed. One of the methods is specifically tailored to MFL and may be very computationally expensive. The two other methods are based on existing algorithms, proposed in Refs. [8, 9], where sensitivity analysis is used to partition the parameter set, which results in more efficient algorithms. We adapt these methods to approximate the satisfaction set of an MFL formula. This chapter compares the three methods and discusses available tools. Moreover, we compare the previously proposed logic MF-CSL to the logic MFL and motivate the existence of both. The possible combination of both logics is also discussed and illustrated by an example.

This chapter is further organized as follows. Section 2 discusses related work on model-checking mean-field models. In Sect. 3, a brief overview of the mean-field model and mean-field analysis is provided. The logic MF-CSL is described in Sect. 4, whereas Sect. 5 introduces MFL. The two logics are compared in Sect. 6. Section 7 concludes the chapter.

2 Related Work

Model-checking analyses whether a system state satisfies certain properties. It was initially introduced for finite deterministic models, for the validation of computer and communication systems, and later extended to stochastic models and models with continuous time. Checking models of large systems is complicated by the state-space explosion problem. Hence, model-checking mean-field models is considered a valuable continuation. For an overview, we refer the reader to Ref. [10].

The first work on model-checking mean-field models was presented in Refs. [6, 11]. Reference [11] presented first steps towards approximate model-checking of bounded CSL properties of an individual object (or group of objects) in a large population model. The Fast Simulation Theorem is used to characterise the behaviour of a single object via the average system behaviour, as defined by mean-field approximation. The proposed method is called *fluid model-checking*, which has been supplemented with next and steady-state operators in Ref. [12].

In contrast to fluid model checking, cf. [6], focuses on the properties of the whole population and proposes the logic MF-CSL. Such formulas consist of two layers, namely the local CSL formula, which describes the property of an individual object and a global *expectation* formula describing the fraction of objects, satisfying the local formula. The algorithm to check the until operator on the local level is based on the algorithm presented in Ref. [11]. An extra layer on top of local CSL properties allows the description of global properties of the whole system. We discuss this logic in more details later in this chapter.

Another two-layer logic is introduced in Ref. [13]. The time-bounded local properties are described using 1-global-clock Deterministic Timed Automata. Next, the global probability operator is introduced to estimate the probability that a certain fraction of local objects satisfies the local property, instead of the fractions of objects satisfying a certain property as in Ref. [6].

A different approach for model-checking mean-field models was proposed in Ref. [14]. There, the authors focus on the properties of an individual object, which is modelled as a discrete-time model in contrast to previously mentioned works. The, so-called, *on-the-fly* model-checking approach examines only those states that are required for checking a given property instead of constructing the entire state space.

Another way of looking at the mean-field model is to consider the behaviour of the whole system without addressing its local behaviour. Having in mind the representation of the global behaviour as a real-valued signal, one can use STL-like properties [7], as will be discussed further in this chapter. Moreover, a great effort has recently been made to enhance temporal properties with a real value, which is addressed as *quantitative semantics* or *robustness degree* [9, 15–17]. These methods can be successfully applied to mean-field models as well. The robustness of stochastic systems (including mean-field or fluid models) has been discussed in Ref. [18]. The design problem has also been addressed, where the parameters of the model can be optimized in order to maximize robustness. Several tools are available which allow calculating the robustness degree [19–21].

3 Mean-Field Models

The main idea of mean-field analysis is to describe the overall behaviour of a system that is composed of many similar objects via the average behaviour of the individual objects. In Sect. 3.1 we briefly recall the definition of the *local model*, which describes the behaviour of each individual object as well as how to build the *overall model* that describes the complete system. In Sect. 3.2, we then describe how to compute transient and steady-state occupancy measures using mean-field analysis.

3.1 Model Definition

Let us start with a random individual object which is part of a large population. We assume that the size N of the population is constant; furthermore, we do not distinguish between classes of individual objects for simplicity of notation. However, these assumptions can be relaxed, see, e.g., [22].

The behaviour of a single object can be described by defining the state space $S^l = \{s_1, s_2, \ldots, s_K\}$ that contains the states or "modes" this object may experience during its lifetime, the labelling of the state space $L : S^l \to 2^{LAP}$ that assigns *local atomic propositions* from a fixed finite set of Local Atomic Properties (LAP) to each state and the transitions between these states.

In the following we will consider a large population of N objects, where each individual is modelled as described above, and denoted as $\mathscr{M}_i$ for $i \in \{1, \ldots, N\}$. Let us first try to preserve the identity of each object and build the model, describing the behaviour of N objects individually. It is easy to see that when the population grows linearly the size of the state space of the model grows exponentially. Fortunately, the mean-field approach allows modelling such a system of indistinguishable objects and avoids exponential growth of the state space (*state-space explosion*).

Given the large number of objects, where each individual is modelled by $\mathscr{M}$, we proceed to build the overall model of the whole population. We assume that all objects behave identically, therefore, we can move from the *individual* representation to a *collective* one, that does not reason about each object separately, but gives the number (or fraction) of individual objects in a given state of the model $\mathscr{M}$. It is done by taking the following steps:

Step 1. Lump the state space. When preserving the identity of the objects in a population $(\mathscr{M}_1, \mathscr{M}_2, \ldots, \mathscr{M}_N)$ the sequence of the models of individual objects can be considered as a model of the population. However, the size of such sequence depends on N. Due to the identical and unsynchronized behaviour of the individual objects, a *counting abstraction* (or, stated differently, a transition from the individual to a collective representation) is used to find a smaller stochastic process, denoted as $\mathbf{M}^{(N)}$, of which the states capture the number of the individual objects across the states of the local model $\mathscr{M}$:

$$\mathbf{M}_j^{(N)} = \sum_{i=1}^{N} \mathbb{1}\{\mathscr{M}_i = j\}.$$

The state of $\mathbf{M}^{(N)}$ at time t is a counting vector $\overline{M}(t) = (M_1(t), M_2(t), \ldots, M_K(t))$, where $M_i \in \{0, \ldots, N\}$, and $\sum_{i=1}^{K} M_i = N$. The initial state is denoted as $\overline{M}(0)$.

Step 2. Defining transition rates. Given $\mathbf{M}^{(N)}$ and $\overline{M}(0)$ as defined above the Continuous-Time Markov Chain (CTMC) $\mathscr{M}^{(N)}(t)$ can be easily constructed. The transition rates are defined as follows [23]:

$$\mathbf{Q}_{i,j}(\overline{M}(t)) = \begin{cases} \lim_{\Delta \to 0} \frac{1}{\Delta} \text{Prob}\Big\{\mathscr{M}(t+\Delta)) = j \,|\, \\ \qquad\qquad \mathscr{M}(t) = i, \overline{M}(t)\Big\}, & \text{if } M_i(t) > 0, \\ 0, & \text{if } M_i(t) = 0, \\ -\sum_{h \in S^l, h \neq i} \mathbf{Q}_{i,h}(\overline{M}(t)), & \text{for } i = j, \end{cases}$$

where $\mathscr{M}(t)$ indicates the state of the individual object at time t. The transition matrix depends on time via $\mathscr{M}(t)$.

Step 3. Normalize the population. For the construction of the mean-field model, which does not depend on the size of the population, the state vector is normalized as follows:

$$\overline{m}(t) = \frac{\overline{M}(t)}{N},$$

where $0 \leq \overline{m}_i(t) \leq 1$ and $\sum_{i=1}^{K} m_i = 1$.
When normalizing, first we have to make sure that the related transition rates are scaled appropriately. The transition rate matrix for the normalized population is given by:

$$Q(\overline{m}(t)) = \mathbf{Q}(N \cdot \overline{m}(t)).$$

Secondly, the initial conditions have to scale appropriately. This is commonly called *convergence of the initial occupancy vector* [24, 25]:

$$\overline{m}(0) = \frac{\overline{M}(0)}{N}.$$

The overall mean-field model can then be constructed as follows:

Definition 1 (*Overall mean-field model*) An overall mean-field model $\mathscr{M}^{\mathscr{O}}$ describes the limit behaviour of $N \to \infty$ identical objects and is defined as a tuple (S^o, Q), that consists of an infinite set of states:

$$S^o = \left\{ \overline{m} = (m_1, m_2, \ldots, m_K) \,|\, \forall j \in \{1, \ldots, K\}, m_j \in [0,1] \wedge \sum_{j=1}^{K} m_j = 1 \right\},$$

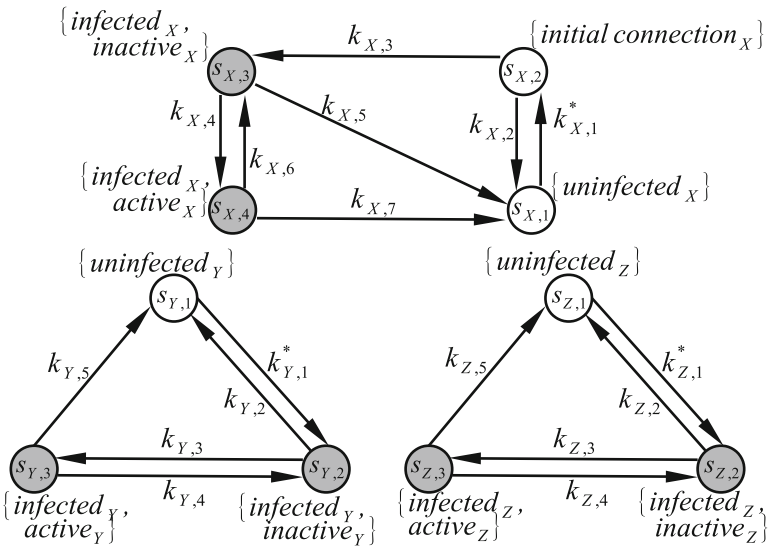

Fig. 1 The model describing computer virus spread in three groups of computers

where $\overline{m}$ is called *occupancy vector*, and $\overline{m}(t)$ is the value of the occupancy vector at time t; m_j denotes the fraction of the individual objects that are in state s_j of the model $\mathscr{M}$. The transition rate matrix $Q(\overline{m}(t))$ consists of entries $Q_{s,s'}(\overline{m}(t))$ that describe the transition rate of the system from state s to state s'.

Note that for any finite N the occupancy vector $\overline{m}$ is a discrete distribution over K states, taking values in $\{0, \frac{1}{N}, \frac{2}{N}, \ldots, 1\}$, while for infinite N, the m_i are real numbers in $[0, 1]$.

Example 1 We describe a model of virus spread in a system of interacting computers (see Fig. 1). We divide the whole computer system into three groups (e.g. different departments or geographical locations). We name these groups X, Y and Z. Each group has a fixed number of nodes (computers) N_X, N_Y and N_Z, respectively, where $N = N_X + N_Y + N_Z$. Communication between the groups is possible, but less probable than within a group. The system we model has three different locations and two different types of hardware (or software), where for each type the same computer virus would behave differently, i.e. computers in group X differ from the computers in groups Y and Z. The model of computer behaviour must take into account the possibility of being (i) in each of the three groups and (ii) being in each of the states of infection. Let us now consider the spread of the infection.

States represent the modes of an individual computer, which can be *uninfected*, *infected and active* or *infected and inactive* in all three groups; in addition, in group X a computer must first stay in the *initially connected* state before the virus is fully embedded in the computer system and the computer is infected. An *active* infected computer spreads the virus, while an *inactive* computer does not. Given this

information, the state-space of the local model must be extended to incorporate the possibility of being in each of the three groups. This results in the finite local state-space $S^l = \{s_{X,1}, s_{X,2}, s_{X,3}, s_{X,4}, s_{Y,1}, s_{Y,2}, s_{Y,3}, s_{Z,1}, s_{Z,2}, s_{Z,3}\}$, with $|S^l| = K = 10$ states, which are labelled as $\mathit{infected}_X$, $\mathit{uninfected}_X$, $\mathit{initial\ connection}_X$, active_X and $\mathit{inactive}_X$, etc., as indicated in Fig. 1.

The transition rates for computers in the first group are as follows: the infection rate $k^*_{X,1}$ is the rate for the initial connection; after that a computer must first try to pass the initially connected state with rate $k_{X,3}$ or return to the uninfected state with rate $k_{X,2}$. The recovery rate for an inactive infected computer is $k_{X,5}$, the recovery rate for an active infected computer is $k_{X,7}$, the rate with which computers become active is $k_{X,4}$ and they return to the inactive state with rate $k_{X,6}$. Rates $k_{X,2}$, $k_{X,3}$, $k_{X,4}$, $k_{X,5}$, $k_{X,6}$ and $k_{X,7}$ are specified by the individual computer and the properties of the computer virus and *do not* depend on the overall system state. The infection rate $k^*_{X,1}$ *does* depend on the rate of attack $k_{X,1}$, the fraction of computers that is infected and active and, possibly, the fraction of uninfected computers. The dependence on the overall system state is intended to reflect real-world scenarios and might be different for different situations. We discuss the infection rate of the current model further in this example.

Given a system of N computers, we can model the average behaviour of the whole system through the global mean-field model, which has the same underlying structure as the individual model (see Fig. 1), however, with state-space $S^o = \{m_{X,1}, m_{X,2}, m_{X,3}, m_{X,4}, m_{Y,1}, m_{Y,2}, m_{Y,3}, m_{Z,1}, m_{Z,2}, m_{Z,3}\}$, where $m_{X,1}$ denotes the fraction of uninfected computers in group $X \cdot m_{X,2}$ represents the fraction of computers in the initially connected state, and $m_{X,3}$ and $m_{X,4}$ denote the fraction of active and inactive infected computers in group X, etc.

The infection rate can then be seen as the number of attacks performed by all active infected computers in group X, which is uniformly distributed over all uninfected computes in a chosen group, that is, $k_{X,1} \cdot \frac{m_{X,4}(t)}{m_{X,1}(t)}$. Note that we assume here that computer viruses are "smart enough" to only attack computers which are not yet infected, see [5, 26]. As discussed above, the computers from the different groups might interact with a certain probability. In the context of our virus spread model, this interaction plays a role when infected computers from groups Y and Z might contact uninfected computers in group X and vice versa. In this example model, we describe a virus which chooses one of the groups of computers with probability $p_{X,X}$, $p_{X,Y}$, $p_{X,Z}$ (for an infected computer from group X). The complete infection rates are composed by multiplying the above rates for one group with the probability of choosing a given group and accumulating all possible interactions between the three groups. Given the reasoning above, the infection rate in group X is

$$k^*_{X,1} = p_{X,X} \cdot k_{X,1} \cdot \frac{m_{X,4}(t)}{m_{X,1}(t)} + p_{Y,X} \cdot k_{Y,1} \cdot \frac{m_{Y,3}(t)}{m_{X,1}(t)} + p_{Z,X} \cdot k_{Z,1} \cdot \frac{m_{Z,3}(t)}{m_{X,1}(t)}.$$

The infection rates of groups Y and Z are constructed in a similar way.

For example, a completely healthy system without infected computers would be in state $\overline{m} = \left(\frac{N_1}{N}, 0, 0, 0, \frac{N_2}{N}, 0, 0, \frac{N_3}{N}, 0, 0\right)$. A system with 50 % *uninfected* computers and 40 and 10 % of *inactive* and *active* computers in group X, and no infected computers in groups Y and Z would be in state:

$$\overline{m} = \left(0.5 \cdot \frac{N_1}{N}, 0, 0.4 \cdot \frac{N_1}{N}, 0.1 \cdot \frac{N_1}{N}, \frac{N_2}{N}, 0, 0, \frac{N_3}{N}, 0, 0\right).$$

3.2 Mean-Field Analysis

Here we express a reformulation of Kurtz's theorem [27] which relates the behaviour of the sequence of models $\mathscr{M}_1, \mathscr{M}_2, \ldots, \mathscr{M}_N$ with increasing population sizes to the limit behaviour. We reformulate the theorem to make it more suitable for our purposes.

Before the theorem can be applied one has to check whether the overall mean-field model satisfies the following two conditions:

1. the model preserves the so-called *density dependence* condition in the limit $N \to \infty$ for all $N > 1$. This means that transition rates scale together with the model population, so that in the normalized models they are independent of the size of the population.
2. The rate functions are required to be Lipschitz continuous (informally it means that rate function are not too step).

When the three steps for constructing the mean-field model are taken and the above-mentioned conditions are satisfied, Kurtz's theorem can be applied, which can be reformulated as follows: *For increasing values of the system size* ($N \to \infty$) *the sequence of the individual models* converges almost surely [28] *to the occupancy vector* $\overline{m}$, *assuming that functions in* $Q(\overline{m}(t))$ *are Lipschitz continuous, and for increasing values of the system size, the initial occupancy vectors converge to* $\overline{m}(0)$. The above statement can be formally rewritten as in Ref. [23].

Theorem 1 (Mean-field convergence theorem) *The normalized occupancy vector* $\overline{m}(t)$ *at time* $t < \infty$ *tends to be deterministic in distribution and satisfies the following differential equations when* N *tends to infinity:*

$$\frac{\mathrm{d}\overline{m}(t)}{\mathrm{d}t} = \overline{m}(t) \cdot Q(\overline{m}(t)), \text{ given } \overline{m}(0). \tag{1}$$

The ODE (1) is called the *limit ODE*. It provides the results for the population of size $N \to \infty$, which is often an unrealistic assumption for real-life systems. However, when the number of objects in the population is finite but sufficiently large, the limit ODE provides an accurate approximation and the mean-field method can be successfully applied.

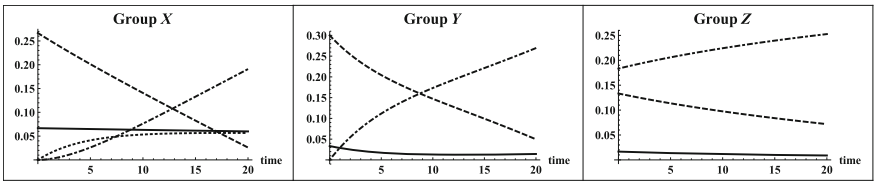

Fig. 2 Distribution of the computers over the states of the model for each group. The *dashed*, *dotted*, *dash-dotted* and *solid lines* show the fraction of uninfected, initially infected, infected inactive and infected active computers respectively

The transient analysis of the overall system behaviour can be performed using the above system of differential equations (1), i.e. the fraction of objects in each state of $\mathscr{M}$ at every time t is calculated, starting from some given initial occupancy vector $\overline{m}(0)$, as illustrated in the following example.

Example 2 We provide an example, applying the mean-field method to the virus spread model, as in Example 1. We describe how to construct the mean-field model and obtain the performance results. The model below will be used throughout the chapter as a running example. The distribution of objects over three groups in this example is uniform, where $N_1 = N_2 = N_3 = \frac{N}{3}$.

Theorem 1 is used to derive the system of ODEs (1) describing the mean-field model. These ODEs are solved, given the following initial vector[1] $\overline{m}(0) = \frac{1}{3}(\{0.8, 0, 0, 0.2\}, \{0.9, 0, 0.1\}, \{0.4, 0.55, 0.05\})$, and the following parameters:

$$
\begin{array}{llll}
k_{X,1} = 0.2, & k_{Y,1} = 0.9, & k_{Z,1} = 0.25, & p_{X,X} = 0.93, \\
k_{X,2} = 0.01, & k_{Y,2} = 0.005, & k_{Z,2} = 0.001, & p_{Y,Y} = 0.94, \\
k_{X,3} = 0.2, & k_{Y,3} = 0.01, & k_{Z,3} = 0.001, & p_{Z,Z} = 0.97, \\
k_{X,4} = 0.0001, & k_{Y,4} = 0.1, & k_{Z,4} = 0.05, & p_{X,Y} = 0.05, \\
k_{X,5} = 0.0001, & k_{Y,5} = 0.06, & k_{Z,5} = 0.001, & p_{X,Z} = 0.02. \\
k_{X,6} = 0.005, & p_{Y,X} = 0.05, & p_{Z,X} = 0.02, & \\
k_{X,7} = 0.005, & p_{Y,Z} = 0.01, & p_{Z,Y} = 0.01, &
\end{array}
$$

The distribution of the objects between the states of the model over time (see Fig. 2) is obtained using Wolfram Mathematica [29].

In the following we discuss a couple of topics which lie beyond the convergence theorem discussed above. We first explain how the behaviour of individual objects within the overall population can be modelled. Second, possible relaxations of the assumptions made for this theorem are discussed. Then the behaviour in the stationary regime is briefly recalled.

Local model. The rates of the model for an individual object within the population may depend on the overall system state, which means that the local model is a *Time-Inhomogeneous Continuous-Time Markov Chain* (ICTMC). To formally describe

[1] Note that for ease of interpretation, we group the elements of the vector according to the three submodels.

the behaviour of a single individual in the population the *asymptotic decoupling* of the system is used, and the result is often referred to as *Fast Simulation* [25, 30]. The main idea of this method lies in the fact that every single object (or group of objects) behaves independently from other objects, and can only sense the *mean* of the system behaviour, which is described by $\overline{m}(t)$. The model of one object within the population is called "local mean-field model" in the following and is defined as:

Definition 2 (*Local model*) A local model $\mathscr{M}^l$ describing the behaviour of one object is defined as a tuple $(S^l, \mathbf{Q}, L)$ that consists of a finite set of K local states $S^l = \{s_1, s_2, ..., s_K\}$; an infinitesimal generator matrix $\mathbf{Q} : (S^l \times S^l) \to \mathbb{R}$; and the labelling function $L : S^l \to 2^{LAP}$ that assigns *local atomic propositions* from a fixed finite set of Local Atomic Properties (LAP) to each state.

Relaxing the assumptions. For models considered in practice the assumption of density dependence may be too restrictive [25]. Furthermore, also the assumption of (global) Lipschitz continuity of transition rates can be unrealistic [31]. Therefore, these assumptions can be relaxed and a more general version of the mean-field approximation theorem, having less strict requirements and which is applied to *prefixes* of trajectories rather than to full model trajectories, can be obtained. We will not focus on this reformulation of the convergence theorem here, instead we refer to [2].

Moreover, the mean-field approach has recently been expanded to a class of models with both Markovian and deterministically timed transitions, as introduced for *generalized semi-Markov processes* in Ref. [32]; and generally distributed timed transitions for *population generalized semi-Markov processes* [33]. In addition, an extension towards hybrid Markov population models has recently been made in Refs. [34, 35].

Stationary behaviour. The convergence theorem does not explicitly cover the asymptotic behaviour, i.e. the limit for $t \to \infty$. However, when certain assumptions hold, the mean-field equations allow to perform various studies including steady-state analysis. In the following we briefly recall how to assess the steady-state behaviour of mean-field models as in [36].

The stochastic process $(\mathbf{M}^{(N)})$, which was approximated by the mean-field model, has to be studied in order to find out whether the stationary distribution exists. It has been shown that, if the stochastic process is reversible, the fixed point approximation addressing the limiting behaviour of the overall mean-field model is indeed valid. Fixed point is an approximation of the stationary behaviour of the stochastic process by the stationary points of the mean-field (fluid) limit [36]. The reversibility of the stochastic process implies that any limit point of its stationary distribution is concentrated on stationary points of the mean-field limit. If the mean-field limit has a unique stationary point, it is an approximation of the stationary distribution of the stochastic process. The stationary distribution $\widetilde{m} = \lim_{t\to\infty} \overline{m}(t)$, if it exists, then is the solution of:

$$\widetilde{m} \cdot Q(\widetilde{m}) = 0. \tag{2}$$

For some models the above equation can not be applied straightforwardly and more advanced methods are required in order to approximate the stationary distribution or its bounds [37]. This, however, lies outside of the scope of this chapter.

4 Overall System Properties. The Logic MF-CSL

In this section, we briefly recall the logic MF-CSL for checking properties of mean-field models, as introduced in Ref. [6]. MF-CSL is a two-layer logic, where (i) on the local level the property of a random object is specified; (ii) on the global level the fraction of objects satisfying this local property is expressed.

4.1 CSL and MF-CSL

Let us first recall how the properties of a random object can be expressed and checked. As discussed in Sect. 3, the model of one object in a mean-field system is an ICTMC. Therefore, the logic CSL [38] can be used to describe properties on the local layer.

Definition 3 (*CSL Syntax*) Let $p \in [0, 1]$ be a real number, $\bowtie \in \{\leq, <, >, \geq\}$ a comparison operator, $I \subseteq \mathbb{R}_{\geq 0}$ a non-empty time interval and *LAP* a set of atomic propositions with $lap \in LAP$. CSL **state formulas** Φ are defined by:

$$\Phi ::= tt \mid lap \mid \neg\Phi \mid \Phi_1 \wedge \Phi_2 \mid \mathscr{S}_{\bowtie p}(\Phi) \mid \mathscr{P}_{\bowtie p}(\phi),$$

where ϕ is a **path formula** defined as:

$$\phi ::= \chi^I \Phi \mid \Phi_1 \, U^I \, \Phi_2.$$

When the local model is time homogeneous, the semantics of CSL formulas is well known. However, in any non-trivial mean-field model, the transition rates of the local model $\mathscr{M}^l$ are not constant. According to Definition 2, the rates of the local model $\mathscr{M}^l$ may depend on the state of the global model $\mathscr{M}^O$, which changes with time. There are two ways to formalize this: the local rates depend on (i) the *current state* $\overline{m}$, which changes with time, or (ii) on *global* time. While the first is more intuitive, it does not allow transition rates to depend explicitly on global time. For ease of notation we restrict ourselves to models that only depend on the overall state. Note that this approach can easily be extended to models that explicitly depend on global time.

Since the local model changes with the overall system state, the satisfaction relation for a local state or path depends on the global state $\overline{m}$. Therefore, the satisfaction relation $\models^{\overline{m}}$, was introduced in Ref. [6]. Due to the lack of space we do not provide the extended semantics of CSL for the local model and refer to [6] for more details.

In order to reason at the level of the overall model in terms of fractions of objects we introduce an extra layer "on top of CSL" that defines the logic *CSL for mean-field models*, denoted MF-CSL.

Definition 4 (*MF-CSL Syntax*) Let $p \in [0, 1]$ be a real number, and $\bowtie \in \{\leq, <, >, \geq\}$ a comparison operator. MF-CSL formulas Ψ are defined as follows:

$$\Psi ::= tt \mid \neg\Psi \mid \Psi_1 \wedge \Psi_2 \mid \mathbb{E}_{\bowtie p}(\Phi) \mid \mathbb{ES}_{\bowtie p}(\Phi) \mid \mathbb{EP}_{\bowtie p}(\phi),$$

where Φ is a **CSL state formula** and ϕ is a **CSL path formula**.

This definition introduces three expectation operators: $\mathbb{E}_{\bowtie p}(\Phi)$, $\mathbb{ES}_{\bowtie p}(\Phi)$ and $\mathbb{EP}_{\bowtie p}(\phi)$, with the following interpretation: $\mathbb{E}_{\bowtie p}(\Phi)$ denotes whether the fraction of objects that are in a (local) state satisfying a general CSL state formula Φ fulfils $\bowtie p$; $\mathbb{ES}_{\bowtie p}(\Phi)$ denotes whether the fraction of objects that satisfy Φ in steady state, fulfils $\bowtie p$; $\mathbb{EP}_{\bowtie p}(\phi)$ denotes whether the probability that an object chosen uniformly at random satisfies path formula ϕ fulfils $\bowtie p$.

Example 3 In the following, we provide a couple of example MF-CSL formulas.

- No more than 5 % of all computers are $infected_Y$ (i.e. these computers belong to the group Y and are infected), is expressed as $\mathbb{E}_{\leq 0,05}\ infected_Y$.
- The percentage of all computers which happen to belong to group X and have a probability of less than 10 % of going from an uninfected to an active infected state within 3 h is greater than 40 %, i.e.

$$\mathbb{E}_{>0,4}\left(\mathbb{P}_{<0.1}(uninfected_X\ U^{[0,3]}\ active_X)\right).$$

- $\mathbb{EP}_{<0.5}(\mathrm{tt}\ U^{[0,2]}\ infected_Z)$ ensures a probability below 50 % that a machine, chosen uniformly at random, is in group Z and becomes infected within two hours from now.
- In the long run a very low proportion (less than 2 %) of all computer should be infected group Z machines: $\mathbb{ES}_{<0.02}\ infected_Z$.

Definition 5 (*MF-CSL Semantics*) The satisfaction relation $\models$ for MF-CSL formulas and states $\overline{m} = (m_1, m_2, \ldots, m_K) \in S^o$ of the overall mean-field model is defined by:

$$\begin{aligned}
&\overline{m} \models tt && \forall\ \overline{m} \in S^o,\\
&\overline{m} \models \neg\Psi && \text{iff } \overline{m} \not\models \Psi,\\
&\overline{m} \models \Psi_1 \wedge \Psi_2 && \text{iff } \overline{m} \models \Psi_1 \wedge \overline{m} \models \Psi_2,\\
&\overline{m} \models \mathbb{E}_{\bowtie p}(\Phi) && \text{iff } \left(\sum_{j=1}^{K} m_j \cdot Ind_{(s_j \models \overline{m}\Phi)}\right) \bowtie p,\\
&\overline{m} \models \mathbb{ES}_{\bowtie p}(\Phi) && \text{iff } \left(\sum_{j=1}^{K} m_j \cdot \pi^{\mathscr{M}^l}(s_j, Sat(\Phi, \overline{m}))\right) \bowtie p,\\
&\overline{m} \models \mathbb{EP}_{\bowtie p}(\phi) && \text{iff } \left(\sum_{j=1}^{K} m_j \cdot Prob^{\mathscr{M}^l}(s_j, \phi, \overline{m})\right) \bowtie p,
\end{aligned}$$

where $Sat(\Phi, \overline{m})$, $\pi^{\mathscr{M}}(s, Sat(\Phi, \overline{m}))$, $Prob^{\mathscr{M}^l}(s, \phi, \overline{m})$ are defined in the semantics of CSL formula [6]; and $Ind_{(s_j \models^{\overline{m}} \Phi)}$ is an indicator function, which shows whether a local state $s_j \in S^l$ satisfies formula Φ for a given overall state $\overline{m}$:

$$Ind_{(s_j \models^{\overline{m}} \Phi)} = \begin{cases} 1, & \text{if } s_j \models^{\overline{m}} \Phi, \\ 0, & \text{if } s_j \not\models^{\overline{m}} \Phi. \end{cases}$$

Although $\overline{m}$ is referred to as the $\overline{m}$ vector at time 0, this is only for ease of discussion, without loss of generality.

To check an MF-CSL formula at the global level, the local CSL formula has to be checked first, and the results are then used at the global level. As discussed above, the local model $\mathscr{M}^l$ is a time-inhomogeneous CTMC, i.e. transition rates vary with the state of the overall model $\mathscr{M}^O$, which makes model-checking at the local level non-trivial. The full algorithms for checking CSL properties of the local model are based on the methods presented in Ref. [11], and have been adapted for the MF-CSL semantics in Ref. [6] according to Definition 5. These algorithms enable the identification of the satisfaction set of a CSL formula for a given initial occupancy vector (i.e. at time $t = 0$), as well as the time-dependent satisfaction set of a CSL formula for a given initial occupancy vector and time bound θ.

Traditionally, the satisfaction set of a given formula is the set of states which satisfies that formula. In the context of MF-CSL model-checking, this would result in the set of all occupancy vectors $\overline{m}$ that satisfy a given formula. While such a set can be built for *time-independent* MF-CSL operators, it is not a trivial task for *time-dependent* operators, since model-checking on the local model $\mathscr{M}^l$ must be done without knowing the initial occupancy vector. Theoretically speaking, partitioning of the whole state-space into sets which satisfy a given MF-CSL formula and the rest is possible, but it is computationally very expensive, as model-checking the local time-inhomogeneous CTMC is very demanding.

Since obtaining the full satisfaction set of an MF-CSL formula is not practically feasible, the notion of *Time Validity Set* of the MF-CSL formula for a given initial occupancy vector and time interval is defined in Ref. [6] as follows:

Definition 6 (*Time Validity Set*) Let $\theta > 0$ be a predefined time bound, Ψ be an MF-CSL formula, and initial conditions of the mean-field model $\mathscr{M}^O$ be an occupancy vector $\overline{m}(0) = \overline{m}$. The *T*ime *V*alidity *S*et (TVS) contains all time intervals, where Ψ holds, for a given $\overline{m}$, and θ:

$$\text{TVS}(\Psi, \overline{m}, \theta) = \{t \in [0, \theta] \mid \overline{m}(t) \models \Psi\}.$$

The time validity set is a set of time intervals, where a given MF-CSL formula holds for fixed initial conditions. We will use this notion later in the chapter when discussing MFL and comparing the two logics. For the details of model-checking MF-CSL we refer to [6]. However, we provide an example that illustrates the general procedure of model-checking two-layer properties.

Example 4 We show how to check whether a given occupancy vector satisfies an MF-CSL formula (Case A), and how to compute the corresponding TVS (Case B) for the model provided in Example 1.3.2.

Case A. Let us consider the following formula

$$\Psi = \mathbb{EP}_{<0.2}(\mathit{uninfected}_Y \; U^{[0,3]} \; \mathit{infected}_Y)$$

and the occupancy vector $\overline{m} = \frac{1}{3}(\{0.8, 0, 0, 0.2\}, \{0.9, 0, 0.1\}, \{0.4, 0.55, 0.05\})$. In order to check the satisfaction relation $\overline{m} \models \Psi$, we first compute the satisfaction set of the until formula on the local level, and then check the satisfaction relation $\overline{m} \models \Psi$ using Definition 5. We use algorithms from [6] as follows.

To find the probability that the until formula holds, the following reachability problem must be solved: Is state $\mathit{infected}_Y$ reached within 3 time units by only visiting $\mathit{uninfected}_Y$ states? This is done by calculating transient probabilities on the modified local model, where all states satisfying $\mathit{infected}_Y$, and states in groups X and Z are made absorbing. The Kolmogorov equation is used to calculate the transient probability matrix of the modified local model. The transient (reachability) probabilities for state $s_{Y,1}$ to reach states $s_{Y,1}$ and $s_{Y,2}$ are equal to 0.77 and 0.23, respectively. The probability to reach any state s starting at this state is, obviously, one. All other transient probabilities are equal to zero.

To compute the probability that the until formula $\phi = \mathit{uninfected}_Y \; U^{[0,3]} \; \mathit{infected}_Y$ holds for each starting state, we have to accumulate the probabilities to start at this state and end at the $\mathit{infected}_Y$ state. Therefore, the probability that the until formula holds equals 0.23, since for all states, but $s_{Y,1}$, this probability equals zero. According to Definition 5, the weighted sum of the entries of the occupancy vector $\overline{m}$ and the respective probabilities in the local model define the expected probability $\mathbb{EP}(\phi)$ as follows:

$$\sum_{j=1}^{K} m_j \cdot Prob^{\mathscr{M}'}(s_j, \phi, \overline{m}) = m_{Y,1} \cdot 0.23 + m_{Y,1} \cdot 1.0 = 0.3 \cdot 0.23 + \frac{1}{30} \cdot 1 = 0.102.$$

Therefore the occupancy vector $\overline{m}$ satisfies $\mathbb{EP}_{<0.2}(\mathit{uninfected}_Y \; U^{[0,3]} \; \mathit{infected}_Y)$.

Case B. Let us consider the same formula Ψ and occupancy vector $\overline{m}$ and compute $\mathrm{TVS}(\Psi, \overline{m}, 15)$ for $\theta = 15$. The calculation of the time-dependent probabilities $Prob^{\mathscr{M}'}(s, \mathit{uninfected}_Y \; U^{[0,3]} \mathit{infected}_Y, \overline{m}, t)$ requires the initial transient probability (for time $t = 0$), as done in Case A. Next, the ODEs describing the time-dependent transient probability of the modified local model are constructed using both forward and backward Kolmogorov equations [6]. These equations are solved using the initial transient probability (as computed in Case A) as initial condition. The solution of these ODEs defines the required time-dependent reachability probabilities. The time-dependent probability that state $s_{Y,1}$ satisfies the until formula is depicted in Fig. 3. The probability equals zero for all other starting states, since these states do not satisfy $\mathit{uninfected}_Y$. To calculate $\mathrm{TVS}(\Psi, \overline{m}, 15)$ the time-dependent expected probability is calculated using the time-dependent probability that the until formula holds (see Fig. 3). Then, the time points where the expected probability equals 0.2

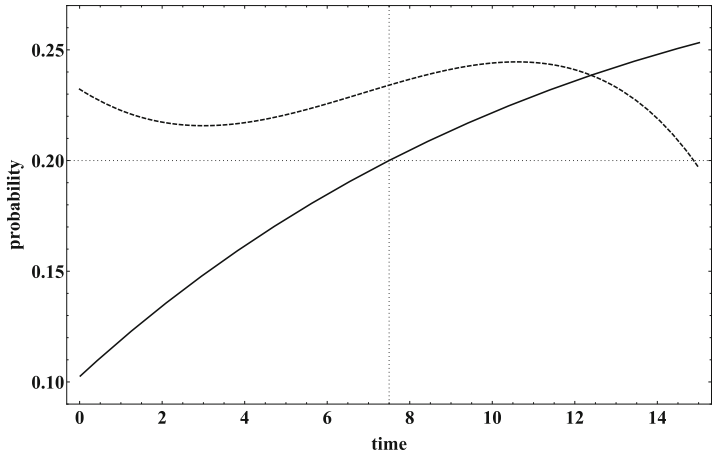

Fig. 3 The *dashed line* (*top*) shows $Prob^{\mathscr{M}^l}(s_{Y,1}, \text{uninfected}_Y\ U^{[0,1]}\text{infected}_Y, \overline{m}, t)$. The (increasing) *solid line* shows the time-dependent expected probability that the formula $\mathbb{EP}_{<0.2}(\text{uninfected}_Y\ U^{[0,1]}\ \text{infected}_Y)$ holds. The *horizontal* and *vertical dotted lines* delimit the period for which the probability is below 0.2, and hence the formula $\mathbb{EP}_{<0.2}(\text{uninfected}_Y\ U^{[0,1]}\ \text{infected}_Y)$ holds

are found. In the current example, this is only $t = 6.228$. The time validity set of the formula $\mathbb{EP}_{<0.2}(\mathit{uninfected}_Y\ U^{[0,3]}\ \mathit{infected}_Y)$ then consists of all time intervals, where the probability $\mathbb{EP}(\mathit{uninfected}_Y\ U^{[0,3]}\mathit{infected}_Y)$ is less than 0.2, i.e. $\text{TVS}(\Psi, \overline{m}, 15) = [0, 7.45)$.

5 Timed Properties of the Global Model: MFL

In the previous section, we introduced a way to express and check properties of the global mean-field model $\mathscr{M}^O$ via properties of a random local object. The properties which MF-CSL can describe include CSL properties (possibly temporal) of the local model and the expected number of objects for a global model to satisfy this property. However, timed properties of the global model can not be expressed using MF-CSL, but can be beneficial for understanding the behaviour of large systems. For that purpose, the new *Mean-Field Logic* (MFL) is introduced in Sect. 5.1. Algorithms for model-checking MFL properties are presented in Sects. 5.2 and 5.3.

5.1 MFL Syntax and Semantics

To be able to express timed properties of the overall model, we adapt the existing Signal Temporal Logic (STL) which was developed to monitor discrete temporal properties of continuous signals [7, 39]. In order to do so, we address the solution

of the ODEs (1) for given initial conditions $\overline{m}(0)$ as a signal or a *trajectory* of the global/overall mean-field model $\mathscr{M}^O$. In the following, we express properties of real-valued trajectories of mean-field models $\overline{m}(t)$ using STL-like properties. We first introduce the mapping of the model trajectory to a Boolean signal.

Definition 7 (*Global atomic property*) An atomic property *GAP* of the global model is a characteristic function (Boolean predicate) $S^o \to \{0, 1\}$, from occupancy vector $\overline{m}$ to a Boolean value.

Applying the concept of *GAP* to a given trajectory of a mean-field model $\overline{m}(t)$ results in a Boolean function of time $GAP(\overline{m}(t))$. In order to guarantee decidability, we require that the output Boolean trajectory $GAP(\overline{m}(t))$ has finite variability, i.e. the number of time points where $GAP(\overline{m}(t))$ changes value is finite; the output trajectory is a Boolean robust (cadlag[2]) function [7]. For simplicity, in the following we use inequalities of the form $\sum_{i \in N} a_i \cdot m_i \bowtie p$ as global atomic properties of mean-field models, where a_i is an indicator factor equal to 1 or 0. However, more advanced functions, satisfying the above requirements, can be used as *GAP*. Given the definition of a global atomic property, the syntax of *Mean-Field Logic* (MFL) can be introduced.

Definition 8 (*Syntax of MFL*) Let $I = [a, b]$, where $0 \leq a < b$, be a non-empty bounded time interval and function *GAP* defining global atomic properties. MFL formulas Υ are defined as:

$$\Upsilon ::= tt \mid GAP \mid \Upsilon_1 \wedge \Upsilon_2 \mid \neg\Upsilon \mid \Upsilon_1 \mathscr{U}^I \Upsilon_2.$$

We can define not only a property of the global model at a given time point but also the evolution of the model over time, as shown in the following example.

Example 5 We first start with the properties of the global model at a given time point (time-independent properties). To define such a property, GAPs are combined with the time-independent operators $\neg$ and $\wedge$.

The following property describes a system in which the fraction of computers that belong to group Y and that are infected is smaller than 0.2:

$$\Upsilon_1 = m_{Y,2} + m_{Y,3} < 0.2,$$

where $m_{Y,2}$ and $m_{Y,3}$ denote the number of infected computers in group Y (inactive and active respectively). The same property can be expressed using atomic properties of the local model as follows: $\text{infected}_Y < 0.2$. Here and in the following, we use labels of the local model instead of fractions in GAP to ease the interpretation of the formula.[3] A more involved property can be defined by a conjunction of GAPs.

[2] A function is called *cadlag* if it is defined on the real numbers (or a subset of them), if it is everywhere right-continuous and if it has left limits everywhere.

[3] Note, however, a global atomic property is not always connected to the properties of the local model (unlike the expectation operator in MF-CSL).

For example, the property that a system has more than 20 % infected computers and less than 1 % active infected group-Z computers is formalized as:

$$\Upsilon_2 = \text{infected} < 0.2 \ \wedge \ \text{active}_Z < 0.01.$$

Note that the first part of the property Υ_2 includes all infected computers in all three groups. The timed properties of the global system are constructed by combining *GAP*s (or other MFL formulas) using the until operator. The following property describes the system in which the fraction of computers which belong to group X and are infected is smaller than 0.1 at all times **until** in the time interval between 3 and 5 time units the fraction of computers that are members of group Z and active exceeds 0.4:

$$\Upsilon_3 = (\text{infected}_Y < 0.1) \ \mathscr{U}^{[3,5]} \ (\text{active}_Z > 0.4).$$

Definition 9 (*Semantics of MFL*) The satisfaction relation $\models$ for MFL state formulas and state $\overline{m} \in S^o$ is defined as:

$$\begin{array}{lll}
\overline{m} \models tt & & \forall \ \overline{m} \in S^o, \\
\overline{m} \models GAP & \text{iff } GAP(\overline{m}) = 1, & \\
\overline{m} \models \Upsilon_1 \wedge \Upsilon_2 & \text{iff } \overline{m} \models \Upsilon_1 \text{and } \overline{m} \models \Upsilon_2, & \\
\overline{m} \models \neg\Upsilon & \text{iff } \overline{m} \not\models \Upsilon, & \\
\overline{m} \models \Upsilon_1 \mathscr{U}^I \Upsilon_2 & \text{iff } \exists t \in I : (\overline{m}(t) \models \Upsilon_2) \wedge (\forall t' \in [0,t] \, \overline{m}(t') \models \Upsilon_1), &
\end{array}$$

where $\overline{m} = \overline{m}(0)$ at time $t = 0$, and $\overline{m}(t')$ is a solution of the ODEs (1) at time $t = t'$ with $\overline{m}$ as the initial condition.

The definition of the until formula is different from the usual representation [40], because it requires both Υ_1 and Υ_2 to hold at time t, in order to guarantee closure [7].

As was explained in the previous section, in this chapter we discuss mean-field models, where the dependency on time is only implicit (via $\overline{m}(t)$). Therefore, the entire model trajectory (the solution of the ODEs (1)) is defined through the current system state; the time when this state is reached does not influence the future behaviour of the system. The occupancy vector for which the satisfaction relation is checked is therefore denoted $\overline{m}(0)$, and the time intervals in the until formulas are relative. However, all the arguments and algorithms presented in this section can be generalized to models with an explicit time dependence. The next section overviews the algorithms used for checking MFL properties of mean-field models.

5.2 Checking an MFL Property

To check an MFL formula, its parse tree has to be built and all sub-formulas have to be checked recursively. Therefore, the algorithms for checking each individual operator have to be introduced. Checking a given occupancy vector $\overline{m}$ against time-

independent operators is straightforward, which follows directly from Definition 9. However, MFL formulas containing the Until operator can not be checked that easily, since the behaviour of the system (trajectory) influences the result. Therefore, we introduce the notion of the *time validity set* for a given MFL formula, mean-field model and an occupancy vector, as done in the previous section for MF-CSL formulas (see Definition 6). It is easy to see that if the $\mathrm{TVS}(\Upsilon, \overline{m}, \theta)$ contains $t = 0$, the formula holds for $\overline{m}$. The TVS of a general MFL formula is again built recursively by finding the TVSs of sub-formulas. The computation of the TVS for the time-independent operators is straightforward:

$$\begin{aligned}
&\mathrm{TVS}(tt, \overline{m}, \theta) &&= [0, \theta],\\
&\mathrm{TVS}(GAP, \overline{m}, \theta) &&= \{t \in [0, \theta] \mid GAP(\overline{m}(t)) = 1\},\\
&\mathrm{TVS}(\neg\Upsilon, \overline{m}, \theta) &&= [0, \theta] \setminus \mathrm{TVS}(\Upsilon, \overline{m}, \theta),\\
&\mathrm{TVS}(\Upsilon_1 \wedge \Upsilon_2, \overline{m}, \theta) &&= \mathrm{TVS}(\Upsilon_1, \overline{m}, \theta) \cap \mathrm{TVS}(\Upsilon_2, \overline{m}, \theta).
\end{aligned}$$

Computing the TVS for the until operator $(\Upsilon_1 \mathscr{U}^{[a,b]} \Upsilon_2)$ (with $0 \leq a < b$) is more challenging. The algorithm described in the following is based on the method of monitoring temporal properties as in Refs. [7, 39]; we refer to these papers for more details and proofs.

To compute the TVS for the until formula $\Upsilon = \Upsilon_1 \mathscr{U}^{[a,b]} \Upsilon_2$ we first find the sets of time intervals where the sub-formulas Υ_1 and Υ_2 hold. Note that both sets may contain multiple intervals. Therefore we denote them as $\mathrm{TVS}(\Upsilon_1, \overline{m}, \theta) = \upsilon_1^1 \cup \upsilon_1^2 \cup \ldots \cup \upsilon_1^{n_1}$, and $\mathrm{TVS}(\Upsilon_2, \overline{m}, \theta) = \upsilon_2^1 \cup \upsilon_2^2 \cup \ldots \cup \upsilon_2^{n_2}$, respectively.

To calculate $\mathrm{TVS}(\Upsilon, \overline{m}, \theta)$ one must obtain all time intervals where $\Upsilon = \Upsilon_1 \mathscr{U}^{[a,b]} \Upsilon_2$ holds. Hence, we search for time intervals where both Υ_1 and Υ_2 hold, since these are the time intervals, where the validity of the until formula can be confirmed, in case at least one such time interval lies between a and b. Recall that the time interval in the until formula is relative to the starting point. Therefore, to check whether a given vector fulfils the until formula, one must check whether the intersection interval can be reached from the vector within the predefined time interval $[a, b]$. Hence, to directly compute the set of all time points from which the formula can be fulfilled, we shift $\mathrm{TVS}(\Upsilon_1 \cap \Upsilon_2, \overline{m}, \theta)$ backwards, i.e. move the left interval bound back with b and the right with a time units. This is defined for each pair of sub-intervals in $\mathrm{TVS}(\Upsilon_1, \overline{m}, \theta)$ and $\mathrm{TVS}(\Upsilon_2, \overline{m}, \theta)$ as a *backwards shift*, denoted as $\mathscr{BS}^{[a,b]}(\upsilon_1^i; \upsilon_2^j)$. For each pair of the intervals $(\upsilon_1^i; \upsilon_2^j)$, the backward shift is computed as follows:

$$\mathscr{BS}^{[a,b]}(\upsilon_1^i; \upsilon_2^j) = ((\upsilon_1^i \cap \upsilon_2^j) \ominus [a, b]) \cap \upsilon_1^i, \tag{3}$$

where $[x_1, x_2] \ominus [a, b] := [x_1 - b, x_2 - a] \cap [0, \infty)$. This backward shift can be understood as follows (from left to right):

1. The intersection $(\upsilon_1^i \cap \upsilon_2^j)$ defines all time points where both Υ_1 and Υ_2 are valid.
2. The $\ominus$-operation (or backwards shift) ensures that:

a. the earliest starting point is taken such that after at most b time units one can reach a state where Υ_2 holds;
b. the latest starting point is taken such that one can still switch to a state in which Υ_2 holds for at least a time units.

3. The intersection with $\upsilon_1^{(i)}$ ensures that on the way to the state where Υ_2 holds, Υ_1 always holds.

After the backwards shift is applied to each pair $(\upsilon_1^i; \upsilon_2^j)$, the resulting intervals are then combined to find the TVS of the overall until formula:

$$\mathrm{TVS}(\Upsilon_1 \mathscr{U}^{[a,b]} \Upsilon_2, \overline{m}, \theta) = \bigcup_{i=1}^{n_1} \bigcup_{j=1}^{n_2} \mathscr{B}\mathscr{S}^{[a,b]}(\upsilon_1^i; \upsilon_2^j). \tag{4}$$

In practice, only the pairs of intervals which actually intersect must be considered. Given the above, the TVS of any MFL formula can be found. After the TVS of the formula is found, we can validate whether a formula holds for a given initial occupancy vector $\overline{m}$ by checking whether $t = 0$ lies in the TVS. Note that the TVS can also be seen as an independent measure of interest, if one is looking for the time slots where the system satisfies a given property, for a given initial state (as in the previous section). In the following, model-checking MFL formulas is illustrated by an example.

Example 6 We again address the model of Example 1, with the same parameters as given in Example 2. We explain in detail how to calculate the time validity set $\mathrm{TVS}(\Upsilon, \overline{m}(0), \theta)$ for both time-independent and time-dependent formulas, given $\overline{m}(0) = \frac{1}{3}(\{0.8, 0, 0, 0.2\}, \{0.9, 0, 0.1\}, \{0.4, 0.55, 0.05\})$, and $\theta = 25$. Next, we check whether $0 \in \mathrm{TVS}(\Upsilon, \overline{m}(0), \theta)$, which would indicate that the initial occupancy vector $\overline{m}(0)$ satisfies the formula.

Case A. We first consider the time-independent property, describing the situation in which the fractions of active computers in groups Y and Z are "sufficiently small", i.e. the fraction of active infected computers in group Y is bounded by 0.015, and the fraction of active infected computers in group Z is at most 0.01:

$$\Upsilon^A = (\mathrm{active}_Y \leq 0.015) \wedge (\mathrm{active}_Z \leq 0.01).$$

To check this property the following steps are taken:

1. The trajectory of the model is obtained by solving the ODEs given $\overline{m}(0)$.
2. The TVS of the sub-formulas $\Upsilon_1^A = (\mathrm{active}_Y \leq 0.015)$ and $\Upsilon_2^A = (\mathrm{active}_Z \leq 0.01)$ are calculated using a root finding procedure for $m_{Y,3}(t) = 0.015$ and $m_{Z,3}(t) = 0.01$:
   - $\mathrm{TVS}(\Upsilon_1^A, \overline{m}(0), \theta) = [6.79; 21.69]$;
   - $\mathrm{TVS}(\Upsilon_2^A, \overline{m}(0), \theta) = [15.14, 25]$ (see Fig. 4a).
3. The TVS of the whole formula then consists of all intervals, where both sub-formulas hold: $\mathrm{TVS}(\Upsilon^A, \overline{m}(0), \theta) = [15.14; 21.69]$ (see Fig. 4b).

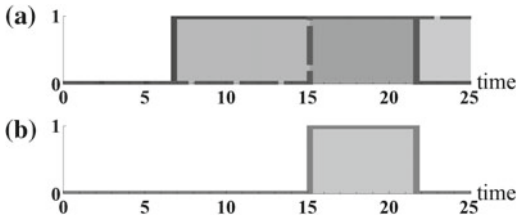

Fig. 4 **a** TVS of $\Upsilon_1^A = (\text{active}_Y \leq 0.015)$ (*solid line*) and $\Upsilon_2^A = (\text{active}_Z \leq 0.01)$ (*dashed line*). **b** Intersection of $\text{TVS}(\Upsilon_1^A, \overline{m}(0), \theta)$ and $\text{TVS}(\Upsilon_2^A, \overline{m}(0), \theta)$

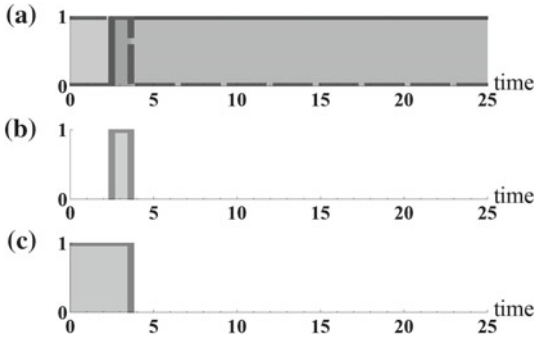

Fig. 5 **a** TVSs of $\Upsilon_1^B = \text{active} \leq 0.05$ (*dashed line*) and $\Upsilon_2^B = (\text{uninfected} \leq 0.6)$ (*solid line*). **b** Intersection of $\text{TVS}(\Upsilon_1^B, \overline{m}(0), \theta)$ and $\text{TVS}(\Upsilon_2^B, \overline{m}(0), \theta)$. **c** TVS of $\Upsilon^B = (\text{active} \leq 0.05)\ \mathscr{U}^{[0,3]}\ (\text{uninfected} \leq 0.6)$

4. The validity of the formula for a given initial vector is checked by verifying whether $t = 0$ lies in $\text{TVS}(\Upsilon^A, \overline{m}(0), \theta)$. It is easy to see that $0 \notin [15.14, 21.69]$, therefore $\overline{m}(0) \not\models \Upsilon^A$.

Case B. We now consider a time-dependent property of the global model, which describes that the fraction of active infected computers in all three groups together (denoted as *active*) remains smaller or equal to 0.05 until within 3 time units the fraction of all uninfected computers becomes less or equal to 0.6. This property ensures that the virus is "quiet enough" and will not be detected until a sufficient number of computers in the system is infected:

$$\Upsilon^B = (\text{active} \leq 0.05)\ \mathscr{U}^{[0,3]}\ (\text{uninfected} \leq 0.6).$$

We first find the time validity sets for this formula and check whether the initial occupancy vector $\overline{m}(0)$ satisfies this property:

1. The time validity sets of the sub-formulas $\Upsilon_1^B = \text{active} \leq 0.05$ and $\Upsilon_2^B = \text{uninfected} \leq 0.6$ are calculated using a root finding procedure for $m_{X,3} + m_{Y,3} + m_{Z,3} = 0.05$ and $m_{X,1} + m_{Y,1} + m_{Z,1}$=0.6; and are given as $\text{TVS}(\Upsilon_1^B, \overline{m}(0), \theta) = [0; 3.64]$ and $\text{TVS}(\Upsilon_2^B, \overline{m}(0), \theta) = [2.5; 25]$ (see Fig. 5a).
2. The intersection of $\text{TVS}(\Upsilon_1^B, \overline{m}(0), \theta)$ and $\text{TVS}(\Upsilon_2^B, \overline{m}(0), \theta)$ is $[2.5; 3.64]$ (Fig. 5b).
3. To find $\text{TVS}(\Upsilon^B, \overline{m}(0), \theta)$ the backwards shift is performed as in Eq. (4) $\text{TVS}(\Upsilon^B, \overline{m}(0), \theta) = [2.5 - 3; 3.64 - 0] = [0; 3.64]$ (see Fig. 5c). Note that the behaviour of the system in the past (before time $t = 0$) is not relevant. Therefore, the lower bound of the TVS is set to zero.

4. Formula $\Upsilon^B = (\text{active} \leq 0.05)\ \mathscr{U}^{[0,3]}\ (\text{uninfected} \leq 0.6)$ holds for $\overline{m}(0)$, since $0 \in \text{TVS}(\Upsilon^B, \overline{m}(0), \theta)$.

Wolfram Mathematica [29] was used to calculate the results above. We compared them with results obtained from the Breach toolbox [19], which has been built to check STL properties, confirming that the results coincided.

5.3 Satisfaction Set of an MFL Formula

In this section, we discuss how to compute the complete satisfaction set of an MFL formula, which is formally defined as follows:

Definition 10 (*Satisfaction Set*) Given an MFL formula Υ and a mean-field model $\mathscr{M}^O$, the **satisfaction set** of an MFL formula consists of *all* occupancy vectors $\overline{m}$ that satisfy Υ:

$$\text{Sat}(\Upsilon) = \{\overline{m} \mid \overline{m} \models \Upsilon\}.$$

The mean-field model has an infinite state-space. Therefore, the computation of the satisfaction set is not straightforward. The ultimate goal is to partition the state-space of the model S^O into two parts: (i) states satisfying a given formula, i.e. $\text{Sat}(\Upsilon)$ and (ii) states which do not satisfy Υ. Since exact methods for computing the satisfaction set of an MFL formula are not available, numerical (approximate) algorithms will be discussed in the following, which means that it might not always be possible to partition the state-space into two sets. Hence, a third set, namely, a set of *uncertain states*, may be necessary. In such cases, this third set should be as small as possible. The satisfaction set of a given formula is constructed recursively, by building and combining the satisfaction sets of sub-formulas.

5.3.1 Time-Independent Operators

The computation of satisfaction sets for operators which are not time dependent, is straightforward and does not imply any additional computations. It follows directly from Definition 9 and is formalized as follows:

$$\begin{aligned}
\text{Sat}(tt) &= S^o, \\
\text{Sat}(GAP) &= \{\overline{m} \mid GAP(\overline{m}) = 1\}, \\
\text{Sat}(\Upsilon_1 \wedge \Upsilon_2) &= \text{Sat}(\Upsilon_1) \cap \text{Sat}(\Upsilon_2), \\
\text{Sat}(\neg\Upsilon) &= S^o \backslash \text{Sat}(\Upsilon).
\end{aligned}$$

5.3.2 The Until Operator

The computation of the satisfaction set of the time-bounded until operator $\Upsilon_1 \mathscr{U}^{[a,b]} \Upsilon_2$ is not trivial and involves additional methods. We discuss three ways that can be used to calculate this set. Two of these methods are directly applicable to the complete MFL formula, and one is only suitable for a single until operator. Hence, the satisfaction sets of the time-independent sub-formulas must be computed separately (see Sect. 5.3.1).

Discretization of the state-space. One of the ways to approximate the satisfaction set of an MFL formula is to discretize the continuous state space and check the MFL formula for each point of the discrete state space obtained using the standard method (see Sect. 5.2). The discretization can be done, for example, by a grid-based approach. However, this approach is computationally intensive and produces only an approximation of the satisfaction set. Moreover, the quality of such an approximation and the computational demand depend directly on the granularity of the grid. Moreover, the complexity of the problem grows with the number of dimensions (local states). Although the method is applicable to any MFL formula, applying it to a model with a large local state-space to obtain high quality approximations is simply not feasible.

Solving two reachability problems. Another way to numerically develop the satisfaction set of a given until formula would be to divide the formula $\Upsilon_1 \mathscr{U}^{[a,b]} \Upsilon_2$ into two *reachability problems*, similar to standard methods, as e.g. in Ref. [38]: (i) starting from the states which satisfy Υ_1, the trajectory evolves such that only states satisfying Υ_1 are visited during time interval $[0, a]$; (ii) starting from the states which satisfy Υ_1 and are reachable during the first step, the trajectory evolves such that only states satisfying Υ_1 are visited until a state satisfying Υ_2 is reached during time interval $[0, b - a]$. The reachability problems for mean-field models can be solved using techniques proposed in Ref. [8]. Their method partitions the parameter set of the ODE-based model into three sets, namely, (i) S_{goal}, which comprises all states that satisfy the reachability problem, (ii) S_{bad}, including all states which do not satisfy it, and (iii) S_{unc}, which combines all states for which reachability can not (yet) be decided. Instead of partitioning the parameter space, we use the proposed methods to partition the state-space of the mean-field model. Note, however, that this approach is only applicable for a single until operator. To compute the satisfaction set of an until formula, we need to solve the two reachability problems in reverse order. We first find all states from which we can reach an Υ_2 state in at most $(b - a)$ time units, while visiting only Υ_1 states. We denote the set of all these states as S'_{goal}. We then find the set of all states, denoted as S_{goal}, from which we can reach S'_{goal}, while visiting only Υ_1 states. The satisfaction set of the overall until formula then equals S_{goal}.

The general approach of the method given in Ref. [8] is as follows. The reachable set is found using *sensitivity analysis* and the satisfaction set is obtained by using a parameter synthesis algorithm based on *refining partition*, which iteratively refines the state-space of the mean-field model and assigns the subsets obtained to one of the three sets, namely: S_{goal}, S_{bad}, or S_{unc} by checking the reachability problem for this set. Each refinement introduces only subsets that are strictly smaller than the refined

set to guarantee that the process ends. The algorithm is designed to stop when the uncertain set is either empty or smaller than a predefined value.

The Breach Toolbox [19] was used to implement the above algorithm. However, a tool for the automated solution of the reachability problem is not available. Although the numerical algorithms in Ref. [8] can not provide formal guarantees on the correctness of the results, asymptotic guarantees exist. Therefore, results can always be improved by decreasing the tolerance factor in the numerical computations.

Robustness-based method. Another method to obtain the satisfaction set of an arbitrary MFL formula, including nested until operators, partitions the state-space based on refining partition algorithms. However, this approach requires introducing a *quantitative semantics* of MFL. This allows both Boolean and real values as a result of a model-checking algorithm ($\mathbb{R} \cup \{\top, \bot\}$). The result of the model-checking procedure shows that a given occupancy vector satisfies an MFL formula (in case the obtained value is greater than zero), and also estimates the quality of satisfaction. We introduce the quantitative semantics of MFL, similarly to [15], where a quantitative semantics was introduced for STL. For simplicity of notation, we use global atomic properties in the form $f(m_1, m_2 \dots m_K) \geq c$, where $c \in \mathbb{R}$.

Definition 11 (*Quantitative semantics*) Given an MFL formula Υ, a mean-field model $\mathscr{M}$, and initial occupancy vector $\overline{m}$, the quantitative semantics $\rho(\Upsilon, \overline{m})$ is defined as follows:

$$
\begin{aligned}
-\top &= \bot, \\
\rho(tt, \overline{m}) &= \top, \\
\rho(GAP, \overline{m}) &= f(m_1, m_2 \dots m_K) - c, \\
\rho(\Upsilon_1 \wedge \Upsilon_2, \overline{m}) &= \min(\rho(\Upsilon_1, \overline{m}), \rho(\Upsilon_2, \overline{m})), \\
\rho(\neg\Upsilon, \overline{m}) &= -\rho(\Upsilon, \overline{m}), \\
\rho(\Upsilon_1 \mathscr{U}^I \Upsilon_2, \overline{m}) &= \sup_{t' \in I} \min\Big(\rho(\Upsilon_2, \overline{m}(t')),\ \inf_{t'' \in [0,t']} \rho(\Upsilon_1, \overline{m}(t''))\Big),
\end{aligned}
$$

where $\overline{m} = \overline{m}(0)$ at time $t = 0$, and $\overline{m}(t')$ is a solution of the ODEs (1) at time $t = t'$ with $\overline{m}$ as the initial condition.

Time and space-time robustness of satisfaction for a quantitative semantics is discussed in Ref. [15], where $\rho(\Upsilon, \overline{m})$ is called a *robustness estimate*. The robustness estimate is found using an inductive procedure for model-checking MFL formulas. Efficient algorithms to find robust estimates are described in Ref. [9], and the Breach toolbox [19] can be directly used for that purpose. Given algorithms to find the robust satisfaction of an MFL formula, the satisfaction set of such formula, $\mathrm{Sat}(\Upsilon)$, can be calculated by partitioning the state-space S^O of the mean-field model. The latter can be done using refining partition, as proposed in Refs. [8, 15].

The robustness analysis can also be performed with tools like S-TaLiRo [20] and BIOCHAM [21]. Note also that here no formal guarantees on the correctness of results can be provided. The advantage of the robustness-based method lies in the fact that the procedure is ununiform for any MFL formula, unlike reachability-based methods. Moreover, there are tools available for robustness analysis of the

time series, numerical solutions of the ODEs, or even measured data in the context of satisfaction set development. Finally, more advanced numerical methods may be applied to partition the state space.

6 Relation Between the Two Logics

As discussed in the previous sections, there are two ways to describe properties of the overall mean-field model. One way is reasoning about the fraction of objects satisfying a given local property by checking whether this number meets a given threshold using the logic MF-CSL. Another way is to describe the properties of the whole system, including timed properties, which can be done with the logic MFL. In Sect. 6.1, we discuss the difference between these two logics and argue that both possess value. The possibility of combining both logics is discussed in Sect. 6.2.

6.1 Comparison of MFL and MF-CSL

Table 1 depicts the main differences between the MFL and MF-CSL logics. As previously discussed, both logics are used in order to describe (and check) properties of mean-field models. Moreover, the time validity set can be defined for MF-CSL, while model-checking MFL properties require the computation of TVSs.

All properties in MF-CSL are based on the structure and labelling of the local model, and *expectation operators* lift these properties to the global level. MFL expresses properties of the global mean-field model independently from the labelling and structure of the local model. A *GAP* can be defined both on the local model structure and labelling, as well as via labelling-independent functions of the occupancy vector $\overline{m}$. For example, properties such as "there are infected computers in all three groups" can easily be described by MFL, while MF-CSL needs "workarounds" by

Table 1 The MF-CSL logic versus the MFL logic

Property	MF-CSL	MFL
Applicable for mean-field models	+	+
Operates on both local and global levels	+	−
Timed property on the local level	+	−
Timed property on the global level	−	+
Depends on the local labelling	+	−
Uses expectation relations	+	−
Has notion of TVS	+	+
Satisfaction set can be obtained	−/+	+

either introducing different labelling on the local level, or expressing the *infected* properties for each group separately on the local level, then on the global level, and finally combining these properties by concatenation and/or negation.

The largest difference lies in the application of timed properties. Both logics may use the until operator. However, in MFL the until operator is used on the global level, while in MF-CSL only the evolution of an individual random object can be described with the until operator. Approximating the full satisfaction set is possible for MFL properties, as defined in Sect. 5. The calculations on the local level of MF-CSL are, however, quite demanding and the partitioning of the state-space using MF-CSL property as an indicator function is impractical.

For some models, such as models of chemical reactions [41], the behaviour of a random individual (one molecule) is not of interest. Therefore, MF-CSL may not be of interest and only the logic MFL would be applicable. Despite this, there are many systems that can be modelled using the mean-field method, where the behaviour of a random object would still be important, for example in the virus spread models, as discussed in this chapter. Clearly, both logics can be of interest, albeit for different users and different systems. Some properties can be expressed in both logics. However, the majority of properties can only be described using one of the two logics, which explains the necessity of introducing both these logics separately.

6.2 Combination of the Two Logics

We now discuss the combination of the two logics to achieve the greatest expressivity. As described in Sect. 5, a *GAP* can be defined by any Boolean function which, when applied to the model trajectory, produces as output a robust cadlag function. As MF-CSL properties can be interpreted as a Boolean function $S^o \to \{0, 1\}$, combined properties can be expressed and analysed.

The above can be generalize as follows: in order to describe the combined property of the global mean-field model, a linear combination of the MF-CSL properties is used as a global atomic property:

$$\sum_j a_j \cdot \mathbb{1}_{\Psi_j} \bowtie p, \tag{5}$$

where a_j is a real number and $\mathbb{1}_{\Psi_j}$ is the indicator function of the MF-CSL property Ψ_j being satisfied.

Due to the cadlag restriction on the *GAP* function, the set of MF-CSL formulas that can be used as *GAP* is restricted in order to guarantee decidability, e.g. properties which are only valid at one time point are not allowed for the combination. Apart from this, any MF-CSL formula whose TVS consists of a finite number of sets can be used as *GAP* in MFL. As an example of such a property, we consider the following formula:

$$\Upsilon = (\mathbb{E}_{\leq 0.1} active_Y)\ \mathscr{U}^{[0,5]}\ (\mathbb{EP}_{\leq 0.1} tt\ U^{[0,3]}\ infected_Y)$$

This property is useful for a system administrator, who wants to be sure that with the current activity of the anti-malware software not more than 10 % of the computers in group Y are infected and active until within 5 time units a system state is reached, where the probability that a random computer will become infected within 3 time units is less or equal than 0.1.

The example below provides a more detailed explanation on how to check such properties. Note that, the calculation of the satisfaction set of such combined properties may not be practically feasible, due to the high computational costs on the local level of MF-CSL sub-formula.

Example 7 The virus spread model is used in the following example with the parameters as given in Example 4. We will construct a combined property using both logics. Then we find the time validity set for the property obtained given a predefined time interval $\theta = [0, 20]$, and initial distribution

$$\overline{m}(0) = \frac{1}{3}(\{0.8, 0, 0, 0.2\}, \{0.9, 0, 0.1\}, \{0.4, 0.55, 0.05\}).$$

Finally, we check the combined property against the initial occupancy vector $\overline{m}(0)$. For simplicity, we use the MF-CSL property, described in Example 4 as one of global atomic properties in the combined property:

$$\Upsilon_1 = \mathbb{EP}_{<0.3}(uninfected_Y\ U^{[0,3]}\ infected_Y).$$

We combine Υ_1 with $\Upsilon_2 = \text{active} \leq 0.1$ using the until operator and obtain the following combined formula:

$$\Upsilon = \mathbb{EP}_{<0.3}(uninfected_Y\ U^{[0,3]}\ infected_Y)\ \mathscr{U}^{[1,2]}\ (active \leq 0.1).$$

This property describes a system where the expected probability that a random computer is from group Y and becomes infected within 3 time units is less than 0.3, at all times until within time interval [1, 2] the number of active infected computers in all three groups is less or equal than 0.1.

To check such a combined property we must first find the TVSs of all sub-formulas, including the time validity set of the MF-CSL formula Υ_1, which equals $\text{TVS}(\Upsilon_1, \overline{m}, \theta) = [0, 11.86]$ (see Example 4). The time validity set of the MFL sub-formula $\text{TVS}(\Upsilon_2, \overline{m}, \theta)$ is found by solving $m_{X,3} + m_{Y,3} + m_{Z,3} = 0.1$; and equals $\text{TVS}(\Upsilon_2, \overline{m}, \theta) = [9.925, 20]$. Figure 6a displays the time validity sets of sub-formulas Υ_1 and Υ_2. The time validity set of the combined formula is computed as described in Sect. 5.2. First, the intersection of $\text{TVS}(\Upsilon_1, \overline{m}, \theta)$ and $\text{TVS}(\Upsilon_2, \overline{m}, \theta)$ is found: $\text{TVS}(\Upsilon_1 \cap \Upsilon_2, \overline{m}, \theta) = [9.925, 11.86]$. Then, $\text{TVS}(\Upsilon_1 \cap \Upsilon_2, \overline{m}, \theta)$ is shifted backwards: $\text{TVS}(\Upsilon, \overline{m}, \theta) = [9.925 - 2, 11.86 - 1] = [7.925, 10.86]$ (see Fig. 6b). As one can see, the occupancy vector $\overline{m}$ does not satisfy the combined property Υ, since $0 \notin [7.925, 10.86]$.

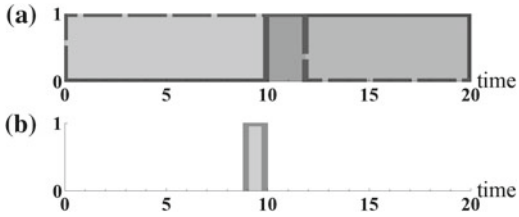

Fig. 6 **a** TVSs of Υ_1 (*dashed line*) and Υ_2 (*solid line*). **b** TVS of Υ

7 Conclusions

Over the last decade, many systems that consist of a large number of interacting objects have been analysed using the mean-field method. This method avoids the well-known state-space explosion problem, which is encountered in many classical Markovian analysis techniques. However, the mean-field method has primarily been used for classical performance evaluation purposes. In this chapter, we have discussed model-checking algorithms for mean-field models, in order to be able to address non-standard system properties.

In this chapter, we define and motivate two logics, called *Mean-Field Continuous Stochastic Logic* (MF-CSL) and *Mean-Field Logic* (MFL), to describe properties of systems composed of many identical interacting objects. The logic MF-CSL [6] uses local CSL properties as a basis for the global expectation operator. Therefore, it is fully dependent on the structure of the local model. The logic MFL, on the other hand, does not take into account properties of the individual objects and only reasons on the global level. Therefore, time-dependent properties of the global model can be described using MFL, while MF-CSL allows only time-dependent properties on the local level.

The algorithms to check MFL properties against a given occupancy vector and to find the so-called time validity set are based on the methods of monitoring temporal properties as in Refs. [7, 39]. We adapt these methods to check global properties of the mean-field model, and illustrate these algorithms in examples. All computations were done in Wolfram Mathematica and compared against results produced by the Breach Toolbox, which were consistent with one another.

Furthermore, three possible ways to calculate the satisfaction set of an MFL formula were discussed. One of these methods relies on the Boolean semantics of MFL presented and a discretization of the continuous state space. The second method makes use of the existing technique to find the parameters of the model, which satisfy a given *reachability* problem [8] using *sensitivity analysis*. The third method adapts an existing notion of *robustness* [15] of a temporal logic and *sensitivity analysis*. This technique is based on defining a *quantitative semantics* of MFL. The resulting robust estimate is then used as an indicator in order to guide the partitioning algorithm.

The expressivity and applicability of the two logics were also compared in this chapter. Despite the fact that both logics are applicable to mean-field models, both are clearly of interest and can not be fully replaced by the other. Another interesting

insight related to the use of the logics presented is that they can be combined if the global atomic property of the mean-field model is represented by one of the expectation operators. This allows the combination of MF-CSL and MFL properties on both levels, including timed properties. Such properties can be easily checked for a given occupancy vector. However, the satisfaction set development may be even more challenging.

Acknowledgments The work in this chapter has been performed when Anna Kolesnichenko was still at the University of Twente. She has been supported through NWO grant 612.063.918, MATMAN (Mean-Field Approximation Techniques for Markov Models), as well as the FP7 Sensation project (see below). Anne Remke has been supported through an NWO VENI grant on Dependability Analysis of Fluid Critical Infrastructures using Stochastic Hybrid Models. Boudewijn Haverkort and Pieter-Tjerk de Boer have been supported through FP7 STREP 318490, Sensation (Self Energy-Supporting Autonomous Computation).

References

1. Kurtz TG (1970) Solutions of ordinary differential equations as limits of pure jump Markov processes. J Appl Probab 7(1):49–58
2. Bortolussi L, Hillston J, Latella D, Massink M (2013) Continuous approximation of collective systems behaviour: a tutorial. Perform Eval 70(5):317–349
3. Bakhshi R, Cloth L, Fokkink W, Haverkort BR (2009) Mean-field analysis for the evaluation of Gossip protocols. In: QEST, IEEE CS Press, pp 247–256
4. Bakhshi R, Endrullis J, Endrullis S, Fokkink W, Haverkort BR (2010) Automating the mean-field method for large dynamic gossip networks. In: QEST, IEEE CS Press, pp 241–250
5. Kolesnichenko A, Remke A, de Boer PT, Haverkort BR (2011) Comparison of the mean-field approach and simulation in a peer-to-peer botnet case study. In: EPEW, vol 6977. LNCS, Springer, pp 133–147
6. Kolesnichenko A, Remke A, de Boer PT, Haverkort BR (2013) A logic for model-checking mean-field models. In: DSN/PDF, IEEE CS Press, pp 1–12
7. Maler O, Nickovic D (2004) Monitoring temporal properties of continuous signals. In: FORMATS, vol 3253. LNCS, Springer, pp 152–166
8. Donzé A, Clermont G, Legay A, Langmead CJ (2010) Parameter synthesis in nonlinear dynamical systems: application to systems biology. J Comput Biol 17(3):325–336
9. Donzé A, Ferrère T, Maler O (2013) Efficient robust monitoring for STL. In: CAV, vol 8044. LNCS, Springer, pp 264–279
10. Hillston J (2014) The benefits of sometimes not being discrete. In: CONCUR, vol 8704. LNCS, Springer, pp 7–22
11. Bortolussi L, Hillston J (2012) Fluid model checking. In: CONCUR, vol 7454. LNCS, Springer, pp 333–347
12. Bortolussi L, Hillston J (2013) Checking individual agent behaviours in markov population models by fluid approximation. In: SFM, vol 7938. LNCS, Springer, pp 113–149
13. Bortolussi L, Lanciani R (2013) Model checking Markov population models by central limit approximation. In: QEST, vol 8054. LNCS, Springer, pp 123–138
14. Latella D, Loreti M, Massink M (2014) On-the-fly fast mean-field model-checking. In: TGC, LNCS, Springer, pp 297–314
15. Donzé A, Maler O (2010) Robust satisfaction of temporal logic over real-valued signals. In: FORMATS, vol 6246. LNCS, Springer, pp 92–106
16. Fainekos GE, Pappas GJ (2009) Robustness of temporal logic specifications for continuous-time signals. Theor Comput Sci 410(42):4262–4291

17. Rizk A, Batt G, Fages F, Soliman S (2008) On a continuous degree of satisfaction of temporal logic formulae with applications to systems biology. In: CMSB, vol 5307. LNCS, Springer, pp 251–268
18. Bartocci E, Bortolussi L, Nenzi L, Sanguinetti G (2013) On the robustness of temporal properties for stochastic models. In: HSB, vol 125. EPTCS, Open Publishing Association, pp 3–19
19. Donzé A, Breach A (2010) Toolbox for verification and parameter synthesis of hybrid systems. In: CAV, vol 6174. LNCS, Springer, pp 167–170
20. Annpureddy Y, Liu C, Fainekos G, Sankaranarayanan S (2011) S-TaLiRo: a tool for temporal logic falsification for hybrid systems. In: TACAS, vol 6605. LNCS, Springer, pp 254–257
21. Calzone L, Fages F, Soliman S (2006) Biocham: an environment for modeling biological systems and formalizing experimental knowledge. Bioinformatics 22(14):1805–1807
22. Chaintreau A, Le Boudec JY, Ristanovic N (2009) The age of gossip: spatial mean field regime. In: ACM SIGMETRICS/Performance, pp 109–120
23. Bobbio A, Gribaudo M, Telek M (2008) Analysis of large scale interacting systems by mean field method. In: QEST, IEEE Computer Society, pp 215–224
24. Darling RWR, Fluid limits of pure jump Markov processes: a practical guide, ArXiv mathematics e-prints arXiv:arXiv:math/0210109
25. Darling RWR, Norris JR (2008) Differential equation approximations for Markov chains. Probab Surv 5:37–79
26. van Ruitenbeek E, Sanders WH (2008) Modeling peer-to-peer botnets. In: QEST, IEEE CS Press, pp 307–316
27. Kurtz TG (1970) Solutions of ordinary differential equations as limits of pure jump Markov processes. J Appl Probab 7(1):49–58
28. Billingsley P (1995) Probability and measure. 3rd edn. Wiley-Interscience
29. Wolfram Research Inc (2010) Mathematica tutorial. http://www.wolfram.com/mathematica/
30. Gast N, Gaujal B (2010) A mean field model of work stealing in large-scale systems. In: ACM SIGMETRICS, ACM, pp 13–24
31. Bortolussi L (2011) Hybrid limits of continuous time Markov chains. In: QEST, IEEE Computer Society, pp 3–12
32. Hayden RA (2012) Mean field for performance models with deterministically-timed transitions. In: QEST, IEEE Computer Society, pp 63–73
33. Hayden RA, Horvàth I, Telek M (2014) Mean field for performance models with generally-distributed timed transitions. In: QEST, vol 8657. LNCS, Springer, pp 90–105
34. Stefanek A, Hayden RA, Gonagle MM, Bradley JT (2012) Mean-field analysis of Markov models with reward feedback. In: ASMTA, vol 7314. LNCS, pp 193–211
35. Stefanek A, Hayden RA, Bradley JT (2014) Mean-field analysis of hybrid Markov population models with time-inhomogeneous rates. Ann Oper Res 1–27
36. Le Boudec JY (2010) The stationary behaviour of fluid limits of reversible processes is concentrated on stationary points. Technical report
37. Benaïm M, Le Boudec JY (2008) A class of mean field interaction models for computer and communication systems. Perform Eval 65(11–12):823–838
38. Baier C, Haverkort BR, Hermanns H, Katoen JP (2003) Model-checking algorithms for continuous-time Markov chains. IEEE Trans Softw Eng 29(7):524–541
39. Nickovic D, Maler O (2007) AMT: a property-based monitoring tool for analog systems. In: FORMATS, vol 4763. LNCS, Springer, pp 304–319
40. Pnueli A (1977) The temporal logic of programs. In: SFCS, IEEE computer society, pp 46–57
41. Gómez-Marn AM, Hernndez-Ortz JP (2013) Mean field approximation of Langmuir-Hinshelwood CO-surface reactions considering lateral interactions. J Phys Chem 117(30):15716–15727