Scientific Annals of Computer Science vol. 27 (1), 2017, pp. 77–109
doi: 10.7561/SACS.2017.1.77

Model Checking Delay Differential Equations Against Metric Interval Temporal Logic

Peter Nazier Mosaad,[1] Martin Fränzle,[1] Bai Xue[1]

Abstract

Delay differential equations (DDEs) play an important role in the modeling of dynamic processes. Delays arise in contemporary control schemes like networked distributed control and can cause deterioration of control performance, invalidating both stability and safety properties. This induces an interest in DDE especially in the area of modeling and verification of embedded control. In this article, we present an approach aiming at automatic safety verification of a simple class of DDEs against requirements expressed in a linear-time temporal logic. As requirements specification language, we exploit metric interval temporal logic (MITL) with a continuous-time semantics evaluating signals over metric spaces. We employ an over-approximation method based on interval Taylor series to enclose the solution of the DDE and thereby reduce the continuous-time verification problem for MITL formulae to a discrete-time problem over sequences of Taylor coefficients. We encode sufficient conditions for satisfaction as SMT formulae over polynomial arithmetic and use the `iSAT3` SMT solver in its bounded model-checking mode for discharging the resulting proof obligations, thus proving satisfaction of time-bounded MITL specifications by the trajectories induced by a DDE. In contrast to our preliminary work in [44], we can verify arbitrary time-bounded MITL formulae, including nesting of modalities, rather than just invariance properties.

Keywords: Delay Differential Equations; Automated Formal Verification; Interval Taylor Over-Approximation; Metric Interval Temporal Logic

[1] Department of Computer Science, Carl-von-Ossietzky Universität, Ammerländer Heerstraße 114-118, 26111 Oldenburg, Germany, E-mail: {peter.nazier.mosaad,fraenzle,bai.xue}@informatik.uni-oldenburg.De

1 Introduction

> *"Indecision and delays are the parents of failure."*
>
> [attributed to George Canning, 1770–1827]

Ordinary differential equations (ODEs) are traditionally used to model the continuous behavior within continuous- or hybrid-state feedback control systems. Significant research has consequently been pursued to achieve automatic verification for such dynamical systems, among it seamless integration of safe numeric ODE solving with satisfiability-modulo-theory solving [9, 14]. In practice, delay is introduced into the feedback loop if components are spatially or logically distributed. Such delays may significantly alter the system dynamics and unmodeled delays in a control loop consequently have the potential to invalidate any stability and safety certificate obtained on the delay-free model. An appropriate generalization of ODE able to model delays within the framework of differential equations is provided by delay differential equations (DDEs), as suggested by [3].

DDEs play an important role in the modeling of natural or artificial processes with time delays in biology, physics, economics, engineering, etc. As a consequence, attention has gone to developing tools permitting their mechanical analysis. However, such tools still are mostly confined to numeric simulation, e.g. by Matlab's `dde23` algorithm. Numerical simulation, despite being extremely useful in system analysis, fails to present reliable certificates of system properties due to numeric approximation. Techniques for safely enclosing set-based initial value problems of ODEs, be it safe interval enclosures [30, 40, 25], Taylor models [4, 31], or flow-pipe approximations based on polyhedra [7], zonotopes [15], ellipsoids [22], or support functions [24], consequently need to be lifted to DDEs.

A safe enclosure method using Taylor series with coefficients in interval form was presented in [44]. To avoid dimension explosion incurred by the ever-growing degree of the Taylor series along the time axis, the method depends on fixing the degree for the Taylor series and moving higher-degree terms into the parametric uncertainty permitted by the interval form of the Taylor coefficients. By using this data structure to iterate bounded degree Taylor over-approximations of the time-wise segments of the solution to a DDE, the approach identifies the operator that yields the parameters of the Taylor over-approximation for the next temporal segment from the current one. Employing constraint solving to analyze the properties of this operator, an automatic procedure is obtained to provide stability and safety verification for a simple class of DDEs of the form

$$\frac{\mathrm{d}}{\mathrm{d}t}\vec{x}(t) = f(\vec{x}(t-\delta)) \tag{1}$$

with linear or polynomial vector field $f : \mathbb{R}^N \to \mathbb{R}^N$, where the derivative at t is a function of the trajectory at $t - \delta$, i.e., the signal value determines the future evolution with delay of δ. The limitation of the method proposed in [44] is that its coverage of safety properties is confined to the verification of state invariants only. Improving on the previous work in [44], the contribution of the current article lies in verifying a class of safety requirements specified using linear-time temporal logic.

The method proposed in this article again addresses DDEs in the form of Eq. (1) and builds upon the safe enclosure method for DDEs presented in [44], yet addresses verification of behavioral properties expressed in metric interval temporal logic (MITL) [1, 11, 33]. MITL is a linear temporal logic that is meaningful when the states evolve in metric spaces, an assumption met by continuous-state systems as in Eq. (1). It is considered as a real-time extension of linear temporal logic (LTL), where the modalities of LTL are constrained with timing bounds. In particular, given a continuous dynamical system (1) with its initial condition(s) and a temporal logic specification expressed in time-constrained MITL, we employ the interval-based Taylor over-approximation method to enclose the solution of the given DDE. This facilitates effective reduction of the signal-based, continuous-time and continuous-state MITL verification problem to a related discrete-time MITL verification problem expressible in terms of timed state sequences. By using any bounded model checking (BMC) tool built on top of an arithmetic SMT solver being able to address polynomial arithmetic, we obtain a procedure able to provide safety certificates for DDE relative to temporal logic specifications. In our case, we use the `iSAT3`[2] implementation of the iSAT algorithm [13] that provides techniques for bounded and unbounded verification problems like k-induction [39] and Craig interpolation [29].

For dealing with temporal properties expressed in MITL, the key step is to safely determine truth values of atomic propositions, i.e., to generate sufficient conditions for their validity over a time frame based on the Taylor over-approximation of the DDE (1). Based on this, the solver is able to verify more complex formulae of temporal logic also involving Boolean connectives

[2] `http://projects.informatik.uni-freiburg.de/projects/isat3/`

and temporal modalities, like the *(bounded) until* operator. Our approach is characterized by the soundness guarantees obtained due to the over-approximation of the DDE and the sufficient conditions used for substituting the exact MITL semantics. The accuracy of approximation can be selected; an automatic refinement method dynamically adapting the accuracy in case of a negative verdict, however, remains to be developed. We demonstrate how our approach works in practice by presenting verification of temporal properties on example systems.

Structure of the article. After discussing related work in Section 2, we formulate the temporal verification problem on DDE in the form of Eq. (1) by defining syntax and continuous-time, continuous-state signal-based semantics of MITL formulae, our requirements specification language (Sect. 3). Section 4 develops interval-based Taylor over-approximation as a safe time-wise discretization to the solution of the DDEs, providing a time-invariant operator generating a timed state sequence on Taylor coefficients. In Section 5, we adapt the interpretation of MITL to the timed state sequence such that it safely recovers the original semantics on the actual solution of the DDE in terms of conditions on the Taylor coefficients of the time-discrete model. Finally, we conclude our paper in Section (6) presenting some ideas for further refinement and future directions of our work.

2 Related Work

Driven by the demand for safety cases (in a broad sense) for safety-critical control systems, we have over the past decades seen a rapidly growing interest in automatic verification procedures for system models involving continuous quantities and dynamics described by, a.o., differential equations. Traditionally ordinary differential equations (ODEs) are used for describing system dynamics and safety properties have been specified in terms of a set of unsafe states formulating the verification problem as reachability analysis. Reachability analysis, which involves computing appropriate approximations of the reachable state sets, plays a fundamental role in addressing such safety verification challenges. Consequently, significant research has been invested in reachability analysis of such dynamical systems [30, 4, 31, 7, 15, 22, 24]. In contrast to this extensive literature on computing the reachable state space for ODEs, the reachability analysis to dynamic systems modelled by delay differential equations (DDEs) is in its infancy and thus provides an open area of research.

Zou, Fränzle et al. proposed in [44] a safe enclosure method using interval-based Taylor over-approximation to enclose a set of functions by a parametric Taylor series with parameters in interval form for a simple class of DDEs in the form of Eq. (1). This method dealt only with simple invariants as safety properties. In [35], Prajna et al. extended the barrier certificate methodology for ODEs to the polynomial time-delay differential equations setting, in which the safety verification problem is formulated as a problem of solving sum-of-square programs. The work in [18] presents a technique for simulation-based time-bounded invariant verification of nonlinear networked dynamical systems with delayed interconnections by computing bounds on the sensitivity of trajectories (or solutions) to changes in initial states and inputs of the system. A similar simulation method integrating error analysis of the numeric solving and the sensitivity-related state bloating algorithms was proposed in [6] to obtain safe enclosures of time-bounded reach sets for systems modelled by DDEs.

Confining safety properties to a set of unsafe states, as in the aforementioned work, considerably restricts the ability of designers to adequately express the desired safe behavior of the system that may involve a number of critical properties such as timing requirements and bounded response. Metric temporal logic (MTL), introduced by Koymans [20], is popular formalism for expressing such properties as a real-time extension of linear temporal logic (LTL) [28] to specify real-time properties. Then, Alur et al. in [1] introduced metric interval temporal logic (MITL) to address the undecidability problem of MTL by relaxing the punctuality of the temporal operators. The bounded-time verification or falsification of such properties has been studied for continuous/hybrid systems in [10, 11, 37, 34, 27], yet DDEs are not handled. In this paper, however, we present an approach aiming at automatic safety verification of a class of DDEs in the form of Eq. (1) against requirements expressed in MITL formulae. In contrast to our preliminary work in [44], yet we can verify arbitrary time-bounded MITL formulae including nesting of modalities rather than just invariance properties.

3 Problem Formulation

In this section, we formulate the verification problem of a simple class of DDEs in the form of Eq. (1) against a class of safety requirements specified using an appropriate linear-time temporal logic. As we deal with continuous state and time, we adopt metric interval temporal logic (MITL) [1] for the purpose of requirements specification language. In this section, we review its syntax and its continuous-time, signal-based semantics.

Let $\mathbb{R}$ be the set of the real numbers. Our time domain is the set of nonnegative real numbers $\mathbb{R}_{\geq 0}$. Also, the trajectory of the DDE of Eq. (1) on an initial condition $x([0,\delta]) \equiv c \in \mathbb{R}$ is a function $x(t)$ such that $x : \mathbb{R}_{\geq 0} \to \mathbb{R}^N$ satisfies the initial condition and $\forall t \geq \delta : \frac{\mathrm{d}}{\mathrm{d}t}\vec{x}(t) = f(\vec{x}(t-\delta))$, where the positive integer N denotes the dimension of the state space. In order to specify the temporal properties of interest, we exploit MITL with continuous semantics, as meaningful when the states evolve in metric spaces like in Eq. (1). We say that $\mathcal{P}(\mathcal{C})$ denotes the powerset of a set $\mathcal{C}$ and assume that AP is a set of atomic propositions. Then, the predicate mapping $\mathcal{M} : AP \to \mathcal{P}(\mathbb{R}^N)$ is a set valued function that assigns to each atomic proposition $\rho \in AP$ a set of states $\mathcal{M}(\rho) \subseteq \mathbb{R}^N$. In this paper, we take the set of atomic propositions AP to be bound constraints $e \sim c$ on state expressions, where e is an expression formed over the state variables, like $x_1 x_2 - 2\sin x_3$, and being compared via a relation $\sim \in \{<, \leq, >, \geq\}$ to a constant $c \in \mathbb{Q}$. Such atomic propositions come equipped with their natural semantics.

3.1 Metric Interval Temporal Logic

Metric interval temporal logic (MITL) [1] is a linear-time temporal logic designed for capturing properties of signals evolving over quantitative and thus metric rather than qualitative time, an assumption met by continuous-state systems as in Eq. (1). It is a real-time extension of linear temporal logic (LTL), where the modalities of LTL are constrained with quantitative timing bounds. Metric temporal logic (MTL) was first introduced by Koymans [20] to specify real-time properties. In order to address the undecidability problem of MTL, Alur et al. in [1] relaxed the punctuality of the temporal operators s.t. they cannot constrain to singleton intervals. We employ MITL to formally characterize the desired behavior of DDEs. Along the following lines, we review and suitably adapt the syntax and the continuous-time and continuous-space semantics of MITL as presented in [1, 33].

Definition 1 ***(Syntax of MITL).*** *An MITL formula φ is built from a set of atomic propositions AP using Boolean connectives and timed-constrained versions of the until operator. It is inductively defined according to the* *grammar*

$$\varphi ::= \top \mid \rho \mid \neg\varphi_1 \mid \varphi_1 \wedge \varphi_2 \mid \varphi_1 \; \mathcal{U}_{\mathcal{I}} \; \varphi_2$$

where $\rho \in AP$, $\top$ is the Boolean constant true and $\mathcal{I} \subseteq \mathbb{Q}_{\geq 0}$ is a nonsingular interval imposing timing bounds on the temporal operators, where $\mathbb{Q}_{\geq 0}$ is the set of non-negative rational numbers.

We can derive the constant *false* by $\bot \equiv \neg\top$. Also, we can define additional, time-constrained version of, temporal operators such as *release* $\mathcal{R}_{\mathcal{I}}$, *eventually* $\diamond_{\mathcal{I}}$, and *always* $\square_{\mathcal{I}}$ as follows:

$$\begin{aligned} \varphi_1 \; \mathcal{R}_{\mathcal{I}} \; \varphi_2 &\equiv \neg((\neg\varphi_1) \; \mathcal{U}_{\mathcal{I}} \; (\neg\varphi_2)), \\ \diamond_{\mathcal{I}} \; \varphi &\equiv \top \; \mathcal{U}_{\mathcal{I}} \; \varphi, \text{ and} \\ \square_{\mathcal{I}} \; \varphi &\equiv \bot \; \mathcal{R}_{\mathcal{I}} \; \varphi \equiv \neg\diamond_{\mathcal{I}} \; \neg\varphi. \end{aligned}$$

Notice that the *release* operator is a temporal modality that is dual to the *until* operator. A formula $\varphi_1 \; \mathcal{R}_{\mathcal{I}} \; \varphi_2$ holds if φ_2 always holds, a requirement that is released as soon as φ_1 becomes valid w.r.t. the time bounds $\mathcal{I}$.

Also, note that MITL has no *next* operator as the time domain is dense. For $\mathcal{I} = [0, \infty]$, we can remove the subscript $\mathcal{I}$ from the temporal operators, obtaining the traditional modalities of LTL. Finally, we would like to point out that the decidability problem of MITL in the continuous semantics for both model checking and satisfiability problems is out of the scope of this paper. For details about the decidability problem, we refer the reader to [1, 32]. Furthermore, it is an open issue whether the model property of DDE w.r.t. MITL formulae is decidable.

3.1.1 Negation Normal Form

We consider MITL formulae in *negation normal form (NNF)*, which can be achieved by pushing all negations inside into the atoms [36]. If we admit the release modality and disjunction in our syntax, then every formula φ has a semantically equivalent negation normal form $nnf(\varphi)$. Such an NNF can be obtained by applying *De Morgan's laws* as well as the dualities between until and release in order to push negations inwards, and thereafter eliminating double negations. This is done by exploiting the following equivalences as rewrite rules from left to right:

$$\begin{aligned}\neg\neg\varphi_1 &\equiv \varphi_1,\\ \neg(\varphi_1 \wedge \varphi_2) &\equiv \neg\varphi_1 \vee \neg\varphi_2,\\ \neg(\varphi_1 \vee \varphi_2) &\equiv \neg\varphi_1 \wedge \neg\varphi_2,\\ \neg(\varphi_1 \ \mathcal{U}_\mathcal{I} \ \varphi_2) &\equiv \neg\varphi_1 \ \mathcal{R}_\mathcal{I} \ \neg\varphi_2,\\ \neg(\varphi_1 \ \mathcal{R}_\mathcal{I} \ \varphi_2) &\equiv \neg\varphi_1 \ \mathcal{U}_\mathcal{I} \ \neg\varphi_2.\end{aligned}$$

These rewrite rules can also be lifted to the derived operators as follows:

$$\begin{aligned}\neg\Diamond_\mathcal{I} \ \varphi &\equiv \Box_\mathcal{I} \ \neg\varphi,\\ \neg\Box_\mathcal{I} \ \varphi &\equiv \Diamond_\mathcal{I} \ \neg\varphi.\end{aligned}$$

3.1.2 Continuous-Time, Continuous-State Semantics of MITL

The continuous semantics of MITL formulae is used to express specifications on the desired temporal evolution to the solutions of DDEs in the form of Eq. (1). This semantics is based on real-valued signals $x : \mathbb{R}_{\geq 0} \to \mathbb{R}^N$ over time. We say that expression e over the state variables x satisfies atomic formula $e \sim c$ at time $t \geq 0$, denoted $e, t \models e \sim c$, iff $e(t) \sim c$ holds. Based on this, semantics of arbitrary MITL formulae is defined inductively, with the semantics of Boolean connectives $\neg$ and $\wedge$ as well as the constant $\top$ being standard. The semantics of the time-constrained *until* operator is defined as follows: $e, t \models \varphi_1 \ \mathcal{U}_\mathcal{I} \ \varphi_2$ iff for some $t' \in \mathcal{I}$, $e, t + t' \models \varphi_2$ holds and furthermore $e, t \models \varphi_1$ for all $t \in (t, t + t')$.

By convention, we say that the DDE of Eq. (1) with an initial value $x([0, \delta]) \equiv c$ satisfies an MITL formula φ if the expression $e(t)$ over its solution trajectory satisfies φ in the sense of $e, 0 \models \varphi$. In what follows, we employ the interval-based Taylor over-approximation method from [44] to enclose the solution of such a DDE. As this method factually generates a discrete sequence of Taylor coefficients rather than a continuous trajectory, we are thus able to reduce a correctness problem over continuous time into a corresponding problem of a time-invariant operator over discrete time. Therefore, it is however necessary to recover the continuous semantics on the actual solution of the DDE from the timed state sequence semantics on the Taylor coefficients.

4 Computing Enclosures for DDEs by Taylor Models

In this section, we review the bounded degree interval-based Taylor over-approximation method for a simple class of DDEs first presented in [44][3]. In order to compute an enclosure for the trajectory $x(t)$ defined by an initial value problem of the DDE (1), a template interval Taylor form of fixed degree k is defined as

$$f_n(t) = a_{n_0} + a_{n_1} t + \cdots + a_{n_k} t^k, \tag{2}$$

where f_n encloses the trajectory for time interval $[n\delta, (n+1)\delta]$, the constant δ is the feedback delay from Eq. (1), and $a_{n_0}, \ldots, a_{n_k}$ are interval-vector parameters. The trajectory induced by DDE (1) can be represented by a piece-wise function, with the duration of each piece being the feedback delay δ. To compute the enclosure for the whole solution of the DDE, we need to calculate the relation between the interval Taylor coefficients in successive time steps as pre-post-constraints on these interval parameters. For notational convenience, we denote the interval parameters $[a_{n_0}, \ldots, a_{n_k}]$ by a matrix $A(n)$ in $\mathbb{R}^{N\times(k+1)}$. The relation between $A(n)$ and $A(n+1)$ can be computed, exploiting different orders of Lie derivatives $f^{(1)}_{n+1}, f^{(2)}_{n+1}, \ldots, f^{(k)}_{n+1}$, as follows:

$$f^{(1)}_{n+1}(t) = g(f_n(t)),\ f^{(2)}_{n+1}(t) = \frac{d\, f^{(1)}_{n+1}(t)}{d\, t}, \ldots, f^{(k)}_{n+1}(t) = \frac{d\, f^{(k-1)}_{n+1}(t)}{d\, t}, \tag{3}$$

i.e., the first order is obtained directly from the given DDE (1) and the $(i+1)$-st order is computed from the i-th order by symbolic differentiation. Then, the Taylor expansion of $f_{n+1}(t)$ with fixed degree k is derived as follows:

$$f_{n+1}(t) = f_n(\delta) + \frac{f^{(1)}_{n+1}(0)}{1!} t + \cdots + \frac{f^{(k-1)}_{n+1}(0)}{(k-1)!} t^{k-1} + \frac{f^{(k)}_{n+1}(\xi_n)}{k!} t^k\,, \tag{4}$$

where ξ_n is a vector ranging over $[0,\delta]^N$.

[3] The corresponding prototype implementation of the interval Taylor over-approximation method for DDEs as well as some examples are available for download from `https://github.com/liangdzou/isat-dde`.

From Eq. (4), by comparing the coefficients of monomials with the same degree at the two sides and by replacing ξ_n by the interval vector $[0,\delta]^N$, we can obtain a time-invariant operator which represents the relation between $A(n)$ and $A(n+1)$. The details of this construction can be found in [44] or retrieved from the example underneath. Hence, we safely enclose the trajectory induced by the DDE (1) by a discrete-time model providing a timed state sequence on a state space $\mathcal{S} \subseteq \mathbb{R}^{N\times(k+1)}$.

4.1 Time-Wise Discretization of DDEs into Timed State Sequences

We demonstrate on a running example taken from [44] how to provide the discrete-time model that encloses the solution of a DDE like Eq. (1). The running example is the DDE

$$\dot{x}(t) = -x(t-1) \tag{5}$$

with the initial condition $x([0,1]) \equiv 1$. Fig. 1 shows the solution of ODE $\dot{x} = -x$ without delay (the dashed line) and with 1 second delay (the solid line). Obviously, the difference between the ODE and the DDE is substantial and necessitates analysing the behaviour of the DDE.

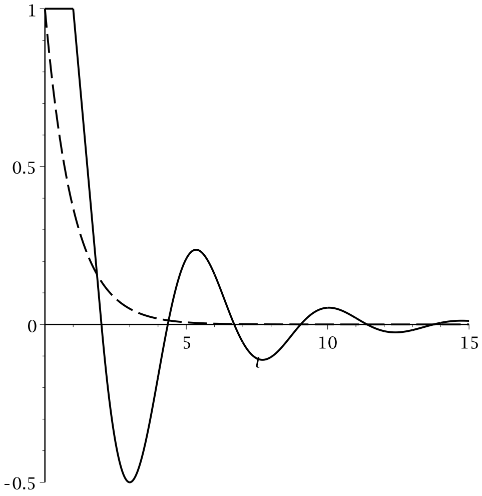

Figure 1: Solutions to the ODE $\dot{x} = -x$ (dashed graph) and the related DDE $\dot{x}(t) = -x(t-1)$ (solid line), both on similar initial conditions $x(0) = 1$ and $x([0,1]) \equiv 1$, respectively.

The method provided in [44] aims at over-approximating the solution of DDE (5) by iterating bounded degree interval-based Taylor over-approximations of the time-wise segments of the solution to the DDE. That way, we identify the operator that yields the parameters of the Taylor over-approximation for the next temporal segment from the current one. For instance, suppose we are trying to over-approximate the solution of DDE (5) by polynomials of degree 2. Then we can predefine a template Taylor form $f_n(t) = a_{n_0} + a_{n_1}t + a_{n_2}t^2$ on interval $[n, n+1]$, where a_{n_0}, a_{n_1}, and a_{n_2} are interval parameters able to incorporate the approximation error eventually necessarily incurred by bounding the degree of the polynomial to (in this example) 2. Here, $f_n(t)$ corresponds to the solution x of DDE (5) at time $n+t$, i.e., $f_n(t)$ over-approximates $x(n+t)$ in the sense of $x(n+t) \in f_n(t)$.

In order to compute the Taylor model, the first and second derivative $f^{(1)}_{n+1}(t)$ and $f^{(2)}_{n+1}(t)$ of solution segment $n+1$ based on the preceding segment (where both segments are of duration 1 each) have to be calculated. The first derivative $f^{(1)}_{n+1}(t)$ is computed directly from Eq. (5) as

$$f^{(1)}_{n+1}(t) = -f_n(t) = -a_{n_0} - a_{n_1}t - a_{n_2}t^2 \,.$$

The second derivative $f^{(2)}_{n+1}(t)$ is computed based on $f^{(1)}_{n+1}(t)$ by

$$f^{(2)}_{n+1}(t) = \frac{d\,(f^{(1)}_{n+1}(t))}{d\,t} = -a_{n_1} - 2a_{n_2}t\,.$$

By using a Lagrange remainder with fresh variable $\xi_n \in [0, 1]$, we obtain

$$\begin{aligned} f_{n+1}(t) &= f_n(1) + \frac{f^{(1)}_{n+1}(0)}{1!}t + \frac{f^{(2)}_{n+1}(\xi_n)}{2!}t^2 \\ &= (a_{n_0} + a_{n_1} + a_{n_2}) - a_{n_0}t - \frac{a_{n_1} + 2a_{n_2}\xi_n}{2}t^2. \end{aligned}$$

Then, the operator expressing the relation between Taylor coefficients in the current and the next step can be derived by replacing both $f_n(t)$ and $f_{n+1}(t)$ with their parametric forms $a_{n_0} + a_{n_1}t + a_{n_2}t^2$ and $a_{n+1_0} + a_{n+1_1}t + a_{n+1_2}t^2$ in the above equation and pursuing coefficient matching. As a result, one obtains the operator

$$\begin{bmatrix} a_{n+1_0} \\ a_{n+1_1} \\ a_{n+1_2} \end{bmatrix} = \begin{bmatrix} 1 & 1 & 1 \\ -1 & 0 & 0 \\ 0 & -\frac{1}{2} & -\xi_n \end{bmatrix} \begin{bmatrix} a_{n_0} \\ a_{n_1} \\ a_{n_2} \end{bmatrix} \tag{6}$$

mapping the coefficients of the Taylor form at step f_n to the coefficients of the Taylor form of f_{n+1}. The coefficients change at every δ time units (every second in the given example) according to the above operator, which therefore defines a discrete-time dynamical system corresponding to the DDE. The discrete-time operator can be rendered time-invariant, yet interval-valued by substituting the uncertain time varying parameter ξ_n with its interval $[0, \delta]$. Hence, we can safely enclose the solution of DDE (5) by a sequence of parametric Taylor series with parameters in interval form. In the case of system (5), as well as for any other linear DDE, the operator generating this sequence is a set-valued linear operator definable by an effectively computable interval matrix.

4.2 Bounded Model Checking Mode in iSAT3

In order to encode the Taylor model corresponding to a DDE, we use bounded model checking (BMC) mode in iSAT3 [38]. The iSAT3 solver is a satisfiability checker for Boolean combinations of arithmetic constraints over real- and integer-valued variables as well as a bounded model-checker for transition systems over the same fragment of arithmetic. It is a stable version implementation of the iSAT algorithm [13]. The solver can efficiently solve bounded verification problems that involve polynomial (and, if needed, transcendental) arithmetic. Hence, it is a good option to solve our proposed problem due to the Taylor forms involved. Also, it allows us to verify/falsify a variety of MITL formulae built on atomic predicates defined over simple bounds, linear, and nonlinear constraints [21]. Bounded model checking (BMC) of a transition system aims at finding a run of bounded length k_{depth} which

- starts in an initial state of the system,
- complies with the system's transition relation, and
- ends in a state in which a certain (un)desired property holds.

The bounded model checking engine then constructs a formula which is satisfiable if and only if a trace with above properties exists.[4] In case of satisfiability, any satisfying valuation of this formula corresponds to such a

[4] It should be noted that this semantic property does not imply that the solver engine subsequently checking that formula for satisfiability can exactly determine its satisfiability. In the case of iSAT, a sound, yet incomplete unsatisfiability check is implemented, as necessitated by the undecidable fragment of arithmetic addressed.

trace. For encoding the discrete transition system on Taylor model in BMC mode, iSAT3 has an input file format consisting of four sections:

- DECL: This part contains declaration of all variables (i.e., variables of the dynamic system, Taylor coefficients of the Taylor over-approximation solution, the duration of each segment $t \in [0, \delta]$, the uncertain time-varying parameter $\xi \in [0, \delta]$).
- INIT: This part is a formula describing the initial state(s) of the system to be investigated.
- TRANS: This formula describes the transition relation in symbolic form; in our case the evolution of the time-discrete Taylor model. We encode a template interval Taylor form of fixed degree k, i.e., $f_n(t)$, and the relation between interval Taylor coefficients in the current and the next step. Variables may occur in primed (e.g., a') or unprimed (e.g., a) form. A primed variable represents the value of that variable in the successor step, i.e., after the transition has taken place.
- TARGET: This formula characterises the state(s) whose reachability is to be checked; in our case it represents satisfaction of the given MITL formula.

The solver unwinds the transition relation k_{depth} times, conjoins the resulting formula with the formulae describing the initial state(s) and the target state(s), and then solves the obtained formula. For our transition relation in terms of Taylor coefficients, the solver recursively for each time frame $[0, k_{depth}\delta]$ constructs the following formula:

$$\mathit{init}\ (\vec{\mathbf{a}}^0) \wedge \bigwedge_{i=0}^{k_{depth}-1} \mathit{trans}\ (\vec{\mathbf{a}}^i, \vec{\mathbf{a}}^{i+1}) \wedge \mathit{target}\ (\vec{\mathbf{a}}^{k_{depth}}),$$

where $\vec{\mathbf{a}}$ is the interval-vector of the Taylor coefficients of the fixed-degree Taylor polynomial $f_n(t)$.

Back to our running example in Sect. 4.1, the iSAT3 encoding for this example is as shown in Listing (1). In the *DECL* part, we declare all variables; the variables of the dynamic system, i.e., x, the Taylor coefficients of degree 2, i.e., a_0, a_1, and a_2, the duration of each segment $t \in [0, 1]$, and the uncertain time varying parameter $\xi \in [0, 1]$. Notice that the range of each variable has to be bounded in iSAT3. We initialize the system variable

x and the Taylor coefficients in *INIT* part according to the given initial condition(s) in our example. Then, in the *TRANS* part, we state the interval Taylor form of degree 2, i.e., $f_n(t)$ corresponds to the solution x of DDE (5), as shown in line 22, and the relation between Taylor coefficients in the current (unprimed variables) and the next segment (primed variables) according to the generated operator (6), where the segments are of duration 1 each.

```
 1  DECL
 2  -- the range of each variable has to be bounded
 3    float [-1000, 1000] a0, a1, a2, x;
 4    float [0,1] t, xi;
 5
 6  INIT
 7  -- initial value of solution
 8    x = 1;
 9
10  -- initialize Taylor coefficients
11    a0 = x;
12    a1 = 0;
13    a2 = 0;
14
15  TRANS
16  -- relation betw. Taylor coefficients in current and next step
17    a0' = a0 + a1 + a2;
18    a1' = -a0;
19    a2' = -0.5*a1 - xi*a2;
20
21  -- x(t) is given by a Taylor form of degree 2
22    x' = a0' + a1'*t + a2'*(t^2);
23
24  TARGET
25  -- state to be reached, e.g.
26    x = -0.25;
```

Listing 1: The encoding in `iSAT3` of the running example in Sect. 4.1.

4.3 Proving Continuous-Time Properties on the Time Discretization

Operator (6) straightaway defines a safe temporal discretization of the DDE system in Eq. (1), i.e., an operator generating a classical timed state sequence in the sense of [1, 10]. We can, however, not simply apply the discrete-time interpretation of MITL to this timed state sequence, as it ranges over a different state space —namely the Taylor coefficients— than the requirements specification in terms of the original state variables. Therefore, we have to translate forth and back between the different state spaces and time models. In detail, the iterated execution of operator (6), starting from an initial vector $a_{0_0}, \ldots, a_{0_k}$ of Taylor coefficients encoding the initial solution segment $x([0,\delta])$, generates a timed state sequence over (interval) Taylor coefficients, with time stamps $t_i = i\delta$, rather than a continuous signal over the state variables $x_1, \ldots, x_N$. Reflecting this encoding, we need a translation step generating conditions over the timed sequence of Taylor coefficients from which we are able to recover the original continuous-time, continuous-state signal-based semantics on the actual solution x of the DDE, as defined in Sect. 3.1.

As has already been observed in [44], such a mapping is straightforward when invariance properties are to be dealt with, for which a sufficient —yet, in the light of over-approximation of the solution, obviously not necessary— condition can be obtained as follows. For an invariance requirement $\Box x \in \mathit{Safe}$, where *Safe* is a set of safe states, the requirement in the n-th segment is translated to the stronger condition

$$\forall t \in [0,\delta]\, \forall \xi \in [0,\delta]\, \forall a_0 \in A_{n,0}, \ldots, a_k \in A_{n,k} : f_n(t) \in \mathit{Safe}, \qquad (7)$$

where f_n is the underlying Taylor form and $A_{n,0}$ to $A_{n,k}$ are the intervals Taylor coefficients stemming from the n-th iteration of the operator (6). As this Taylor form provides an over-approximation of the solution x over time frame $[n\delta, (n+1)\delta]$, the condition (7) implies $\forall t \in [n\delta, (n+1)\delta] : x(t) \in \mathit{Safe}$. Consequently, the continuous-time safety property $\Box x \in \mathit{Safe}$ for system (5) is translated into a sufficient condition according to Eq. (7) for $t, \xi \in [0,1]$ over the sequence of Taylor coefficients of Taylor polynomial of degree 2. As its violation is an existential statement both instantiations of Taylor coefficients within given intervals and existentially quantified time points t and ξ, a solver for satisfiability modulo theory over the existential theory of polynomial arithmetic can be used to solve the safety verification problem. It requires polynomial constraint solving due to the Taylor forms, i.e., polynomial expressions involved in the statement $f_n(t) \in \mathit{Safe}$.

Different proof schemes can be implemented using such a solver: using k-induction [39] or interpolation-based unbounded proof schemes [29], absence of any time point in the sequence of valuations generated by operator (6) satisfying $\exists n \in \mathbb{N}, \exists t \in [0,1], \exists \xi \in [0,1], \exists a_0 \in A_{n,0}, \ldots, a_k \in A_{n,k} : f_n(t) \notin \mathit{Safe}$ can be shown, thereby rigorously showing safety of the DDE system under investigation. Bounded model checking of the same system could, on the other hand, generate counterexamples to safety, which may however be spurious due to the over-approximation involved in the Taylor enclosure.

5 Solving Continuous-Time MITL Formulae by Reduction to Time-Discrete Taylor Approximations

We extend the above idea of generating sufficient conditions for MITL specifications on DDEs in terms of the sequences of enclosing (interval) Taylor coefficients. The aim is to cover a large fragment of MITL, expanding well beyond the invariance properties addressed in [44]. As explained in the previous section, we have obtained a generator for a timed state sequence —the operator (6)— representing the solution of the DDE, yet ranging over a different state space, namely the Taylor coefficients. Hence, the continuous interpretation of the MITL formulae over DDE solutions has to be translated into a semantically appropriate discrete interpretation on a timed state sequence with time stamps $t_i = i\delta$. This translation needs to restore, in the sense of providing sufficient conditions for the solution being a counterexample (i.e., a witness of violation of the property), the continuous semantics of the MITL formulae over the discrete model of the timed state sequence. We do so by first transforming the MITL formula into negation normal form, then generating a sufficient condition by adding the appropriate conditions to the Taylor model that meet the semantics of the property for searching for (possibly spurious) counterexamples with the help of an SMT solver.

5.1 Atomic Proposition

According to the MITL syntax of Sect. 3, atomic propositions are of the form $e \sim c$, where e is an expression over the state variables, c is a constant and $\sim$ an inequational relational operator, i.e., one of $<, \leq, >, \geq$. Using bounded model-checking based on SMT solving, we attempt to find a counterexample of the MITL formula, or, in other words, look for a witness for the negation of the MITL formula. As we transform that negated formula into NNF, atomic propositions occur in positive context only. Then, sufficient conditions for truth of such propositions throughout a time frame $[i\delta, (i+1)\delta]$ can, as already observed in Eq. (7)), obviously be expressed as follows:

$$\forall t \in [0, \delta]\, \forall \xi \in [0, \delta]\, \forall a_0 \in A_{i,0}, \ldots, a_k \in A_{i,k} : \bigwedge_{i=1}^{n} x_i = f_i(t) \wedge e \sim c. \quad (8)$$

As mentioned in Sect. 4.3, when using SMT solving for finding violations of condition (8), we use the negation of the universally quantified condition Eq. (8). As this is an existential formula, it is amenable to standard SMT solving.

5.2 Boolean Connectives

For solving complex-structured formulae, we use a Tseitin-like definitional translation [43], where we introduce a fresh Boolean helper variable $\langle \psi \rangle_i$ for each subformula ψ and each index i of a time frame $[i\delta, (i+1)\delta]$. The intuition is that $\langle \psi \rangle_i$ being true implies that ψ holds for each time point $t \in [i\delta, (i+1)\delta]$. Note that this is a one-sided implication, as we cannot decide properties exactly.

Note that we have in the previous section already obtained appropriate definitions for the case that ψ is an atomic formula, such that we can define

$$\neg \langle e \sim c \rangle_i \Rightarrow \left(\begin{array}{l} \exists t \in [0, \delta]\, \exists \xi \in [0, \delta]\, \exists a_0 \in A_{i,0}, \ldots, a_k \in A_{i,k} : \\ \quad \bigwedge_{i=1}^{n} x_i = f_i(t) \wedge e \not\sim c \end{array} \right) \quad (9)$$

as sufficient condition for validity of an atomic formula $e \sim c$, where e is an expression over the state variables and $\not\sim$ is the converse of the relation $\sim$.

Given a compound formula of the form $\psi_1 = \varphi_1 \wedge \varphi_2$ or $\psi_2 = \varphi_1 \vee \varphi_2$, the encoding for the compound formula is obtained by conjoining to the "axiomatisations" of φ_1 and φ_2 the following definitional translations:

$$\langle \varphi_1 \wedge \varphi_2 \rangle_i \Leftrightarrow \langle \varphi_1 \rangle_i \wedge \langle \varphi_2 \rangle_i$$
$$\langle \varphi_1 \vee \varphi_2 \rangle_i \Leftrightarrow \langle \varphi_1 \rangle_i \vee \langle \varphi_2 \rangle_i$$

Note that a single-sided implication "$\Leftarrow$" from right to left would actually suffice, as we target sufficient conditions only.

5.3 Unary Temporal Operators

Assume we have an MITL formula $\psi_1 = \Diamond_{\mathcal{I}}\, \varphi$ or $\psi_2 = \Box_{\mathcal{I}}\, \varphi$ featuring a time-constrained *eventually* or *always* temporal operator as its outermost operator. Let the lower and upper bound of $\mathcal{I}$ for simplicity be integer multiplies $l\delta$ and $u\delta$ of δ. For each time frame, the value of a given MITL formula is encoded with the help of new Boolean variables for the truth values of its subformulae in particular time instants. The encoding of ψ_1 and ψ_2 can be recursively understood as follows:

$$\begin{aligned}
\langle \Diamond_{[l\delta,u\delta]} \rangle_i &\Leftrightarrow \langle \Diamond_{[0,(u-l)\delta]} \varphi \rangle_{i+l}, && \text{if } 1 \leq l < u \\
\langle \Diamond_{[0,u\delta]} \varphi \rangle_i &\Leftrightarrow \langle \varphi \rangle_i \vee \langle \Diamond_{[0,(u-1)\delta]} \varphi \rangle_{i+1}, && \text{if } 1 < u \\
\langle \Diamond_{[0,\delta]} \varphi \rangle_i &\Leftrightarrow \langle \varphi \rangle_i \\
\langle \Box_{[l\delta,u\delta]} \rangle_i &\Leftrightarrow \langle \Box_{[0,(u-l)\delta]} \varphi \rangle_{i+l}, && \text{if } 1 \leq l < u \\
\langle \Box_{[0,u\delta]} \varphi \rangle_i &\Leftrightarrow \langle \varphi \rangle_i \wedge \langle \Box_{[0,(u-1)\delta]} \varphi \rangle_{i+1}, && \text{if } 1 < u \\
\langle \Box_{[0,\delta]} \varphi \rangle_i &\Leftrightarrow \langle \varphi \rangle_i
\end{aligned}$$

Single-sided implications "$\Leftarrow$" from right to left would again suffice for a sound definitional translation.

In the case of the eventually modality, the condition for detecting satisfaction of φ is somewhat stronger than necessary, actually requiring it to hold throughout a full time frame rather than just once inside.

5.4 Binary Temporal Operators

Assume we have a subformula of shape $\psi_1 = \varphi_1 \; \mathcal{U}_{\mathcal{I}} \; \varphi_2$ or $\psi_2 = \varphi_1 \; \mathcal{R}_{\mathcal{I}} \; \varphi_2$ featuring a time-constrained *until* or *release* operator as its outermost connective. For simplicity, we assume that the lower bound of $\mathcal{I}$ is 0. Such a form can always be achieved by prepending the modality with a unary temporal operator. Then the encoding of a sufficient condition for validity of ψ_1 or ψ_2, resp., over time frame $[i\delta, (i+1)\delta]$ is as follows:

$$\begin{aligned}
\langle \varphi_1 \; \mathcal{U}_{[0,u\delta]} \; \varphi_2 \rangle_i &\Leftrightarrow \langle \varphi_2 \rangle_i \vee (\langle \varphi_1 \rangle_i \wedge \langle \varphi_1 \; \mathcal{U}_{[0,(u-1)\delta]} \; \varphi_2 \rangle_{i+1}), && \text{if } 1 < u \\
\langle \varphi_1 \; \mathcal{U}_{[0,\delta]} \; \varphi_2 \rangle_i &\Leftrightarrow \langle \varphi_2 \rangle_i \\
\langle \varphi_1 \; \mathcal{R}_{[0,u\delta]} \; \varphi_2 \rangle_i &\Leftrightarrow \langle \varphi_2 \rangle_i \wedge (\langle \varphi_1 \rangle_i \vee \langle \varphi_1 \; \mathcal{R}_{[0,(u-1)\delta]} \; \varphi_2 \rangle_{i+1}), && \text{if } 1 < u \\
\langle \varphi_1 \; \mathcal{R}_{[0,\delta]} \; \varphi_2 \rangle_i &\Leftrightarrow \langle \varphi_2 \rangle_i
\end{aligned}$$

As in the case of the eventually modality, the condition for detecting φ_2 in the case of until and of φ_1, resp., in the case of release again is somewhat stronger than necessary, requiring it to hold throughout the respective time frame instead of just once inside.

5.5 Correctness

Let ψ be an MITL formula and $[\psi]_0$ be the definitional translation of ψ obtained by recursively unfolding and conjoining the above definitions of $\langle\psi\rangle_0$ and all $\langle\varphi\rangle_j$ occurring therein. Let $\frac{d\vec{x}}{dt}(t+\delta) = f(\vec{x}(t))$ be a DDE with initial value $\vec{x}([0,\delta]) \equiv \vec{i}$, and let A be the interval matrix obtained from it due to Eq. (6). Let k be the highest index j of any Tseitin variable $\langle\varphi\rangle_j$ occurring in $[\psi]_0$.

Lemma 1 *If $\vec{a}_0 \mathrel{\hat{=}} \vec{i} \wedge \bigwedge_{i=0}^{k} \vec{a}_{i+1} = A\vec{a}_i \wedge [\psi]_0 \wedge \neg\langle\psi\rangle_0$ is unsatisfiable then $\vec{x}$ satisfies ψ, where $\vec{a}_0 \mathrel{\hat{=}} \vec{i}$ denotes the appropriate initialisation of the Taylor coefficients and $\vec{x}$ is the exact solution of the DDE.*

Proof: The sequence $\vec{a}_0 \mathrel{\hat{=}} \vec{i} \wedge \bigwedge_{i=0}^{k} \vec{a}_{i+1} = A\vec{a}_i$ of interval Taylor forms generates an over-approximation of x. The construction of $[\psi]_0$ is such that $\vec{a}_0 \mathrel{\hat{=}} \vec{i} \wedge \bigwedge_{i=0}^{k} \vec{a}_{i+1} = A\vec{a}_i \wedge [\psi]_0 \models \langle\psi\rangle_0$ if all trajectories y enclosed by the sequence of interval Taylor forms, and thus also x itself, satisfies ψ. Satisfiability of $\vec{a}_0 \mathrel{\hat{=}} \vec{i} \wedge \bigwedge_{i=0}^{k} \vec{a}_{i+1} = A\vec{a}_i \wedge [\psi]_0 \wedge \neg\langle\psi\rangle_0$ consequently is a necessary condition for violation of ψ by x. □

Note that $\vec{a}_0 \mathrel{\hat{=}} \vec{i} \wedge \bigwedge_{i=0}^{k} \vec{a}_{i+1} = A\vec{a}_i \wedge [\psi]_0 \wedge \neg\langle\psi\rangle_0$ is a purely existential statement and thus amenable to standard SAT-modulo-theory solving by removing the explicit existential quantifiers in each instant of Eq. (9) by introducing fresh variables.

5.6 Verification Examples

In this section, we use the `iSAT3` SMT solver to discharge the above proof obligations. In order to be able to present the encodings in a compact form suitable for manual inspection and for publication in print, we slightly deviate from a strict implementation of the above scheme, and instead employ the bounded model checking (BMC) mode of iSAT and symbolic counter variables as abbreviation mechanisms whenever appropriate in the search for witnesses as counterexamples of the MITL formulae. The results are, however, the same and the logics behind the encodings is equivalent to the point it can be in a BMC encoding. In particular, the method for using existential arithmetic constraints as sufficient conditions to determine the truth values of propositional (sub)formulae based on the over-approximation model of the DDE and thus recover the continuous semantics of the MITL formula on the actual solution x of the DDE from the timed state sequence semantics is exactly as in Eq. (9).

We demonstrate the approach based on illustrative examples of DDEs in the form of Eq. (1). In our examples, we first consider the DDE (5) presented in Sect. 4.1 with different conjectured MITL formulae to be verified. Thereafter, we apply our method to an adaptation of Gustafson's model of nutrient flow in an aquarium (three dimensional example) [17, p. 589f].

Example 1 *We consider the linear DDE $\dot{x}(t) = -x(t-1)$ with initial condition $x([0,1]) \equiv 1$ and the conjectured safety property $\Box_{[0,10]}\ (x \leq 1.2)$.*

The bounded degree interval-based sequence of Taylor forms can be generated by the operator Eq. (6)). Adopting degree 2 Taylor forms, we can encode this generator in the `iSAT3` input language as shown in lines 24–26 of Listing 2. The encoding is a discrete-time dynamic system over the variables x representing (snapshots of) the DDE solution, Taylor coefficients of the Taylor over-approximation solution, i.e., $a_0, a_1,$, and a_2, a time point in each segment $t \in [0,1]$, and the uncertain time varying parameter $\xi \in [0,1]$. Also, we declare a *counter* to observe the timing bound on the temporal operator.

In order to solve the given MITL formula $\Box_{[0,10]}\ (x \leq 1.2)$ in `iSAT3` in the sense of trying to construct a counterexample, we

1. in accordance with Eq. (9) search for a time frame within which x, being defined as the image of the Taylor polynomial for some $t \in [0,1]$, $\xi \in [0,1]$ in line 29 of the listing, exceeds 1.2, as encoded by condition $x > 1.2$ in the target (line 37), and
2. enforce the count of the time frame to be at most 9 (target, line 38), as time frame n ranges from time n to $n+1$.

For verifying the property at hand, it obviously suffices to check this formula up to unwinding depth 9.[5] Such bounds on unwinding depths can in `iSAT3` be set with the `--start-depth` and `--max-depth` command line options.

In our example, the solver outputs that the system is *safe* for unwinding depth 10, i.e., no state satisfying the target property could be reached within

[5] `iSAT` counts unwindings starting from 0 such that an unwinding of depth 9 yields a trace comprising 10 time instants.

the relevant depth. This constitutes a rigorous proof that the system actually satisfies the MITL formula.

```
DECL
-- the range of each variable has to be bounded
  float [-1000, 1000] a0, a1, a2, x;
  float [0,1] t, xi;

-- define counter for the bounded verification problem
  int [0,9] counter;

INIT
-- initial value of x over [0,1]
  x = 1;

-- initialize Taylor coefficients
  a0 = 1;
  a1 = 0;
  a2 = 0;

-- initialize the counter observing the time interval
-- covered by the bounded always
  counter = 0;

TRANS
-- relation between Taylor coefficients current and next step
  a0' = a0 + a1 + a2;
  a1' = -a0;
  a2' = -0.5*a1 - xi*a2;

-- x(t) is given by a Taylor form of degree 2
  x' = a0' + a1'*t + a2'*(t^2);
-- note the implicit existential quantification of t

-- increment the counter by 1 after each time frame
  counter' = counter + 1;

TARGET
-- state to be reached in bounded time
  x > 1.2 and
  counter <= 9;
```

Listing 2: The encoding of Example 1 in `iSAT3`.

Example 2 *Consider the same DDE equation as Example* (1) *with the same initial condition, but for solving the conjectured safety property of (bounded) until operator* $(x \leq 1.2)\ \mathcal{U}_{[0,10]}\ (x \leq 1.0)$.

This time, we employ four Boolean helper variables, of which `iSAT`'s BMC mode will instantiate a fresh copy in each step:

1. Boolean state variable `b` records a sufficient condition for $x \leq 1.2$ being true throughout the current time frame in the sense that `b` is true only if $x(t) \leq 1.2$ holds for each time instant t in the current time frame (cf. lines 28 and 50 in Listing 3);
2. Boolean state variable `c` records a sufficient condition for $x \leq 1.0$ being true throughout the current time frame (cf. lines 31 and 52);
3. the Boolean state variable `u` records a sufficient condition for the temporal property $(x \leq 1.2)\ \mathcal{U}_{[0,10-n]}\ (x \leq 1.0)$ being true in the current step, with n being the number of the current step (cf. line 54);
4. Boolean state variable `done` is a helper variable necessitated by the confined expressiveness of the BMC mode, which permits reference to current and next states only. It records whether the termination condition $x \leq 1.0$ has already been true in the past (lines 34 and 55).

```
 1  DECL
 2  -- the range of each variable has to be bounded
 3    float [-1000, 1000] a0, a1, a2, x1, x2;
 4    float [0,1] t1, t2, xi;
 5  -- each of the atomic subformulae needs its own
 6  -- fresh copy of the state variable x and the time
 7  -- instant t due to the quantifier elimination
 8
 9  -- define counter for bounded verification problem
10    int [0,9] counter;
11
12  -- define Boolean helper variables
13    boole b, c, u, done;
14  -- b records sufficient condition for x <= 1.2
15  -- c records sufficient condition for x <= 1.0
16  -- u records sufficient condition for until
17  -- done records whether c has been true in the past
18
19  INIT
20    x1  = 1;
21    x2  = 1;
22  -- initialize Taylor coefficients
23    a0  = 1;
24    a1  = 0;
```

```
  a2  = 0;

--initialize b, the sufficient condition of everywhere x <= 1.2
  (not b)  -> (x1 > 1.2);

--initialize c, the sufficient condition of everywhere x <= 1.0
  (not c)  -> (x2 > 1.0);

-- initialize done, the variable memoizing x <= 1.0
  done  <-> c;

-- counter observes the time interval
  counter = 9;

TRANS
-- description of the transition system of DDE model
  a0’ = a0 + a1 + a2;
  a1’ = -a0;
  a2’ = -0.5*a1 - xi *a2;

-- tracing the bounded until
-- find witness points
  x1’ = a0’ + a1’*t1 + a2’*(t1^2);
  x2’ = a0’ + a1’*t2 + a2’*(t2^2);
-- b is sufficient condition for x <= 1.2 throughout time frame
  (not b’) -> (x1’ > 1.2);
-- c is sufficient condition for x <= 1.0 throughout time frame
  (not c’) -> (x2’ > 1.0);
-- recurrence rules for until
  u  <-> done or (b and u’);
  done’ <-> (c’ and counter > 0) or done;  -- remembers c
  (counter > 0) -> (counter’ = counter -1);
  (counter = 0) -> (counter’ = 0);

TARGET
-- for constructing a counterexample, the until formula ought
-- to be violated in the initial time instant
  (not u) and (counter = 9);
```

Listing 3: The encoding of Example 2 in iSAT3.

Checking above example for an appropriate unwinding depth of at least 9, iSAT will report unsatisfiable, which approves absence of a counterexample and thus proves the property to be satisfied.

Example 3 *This example (taken from [44]) is an adaptation of Gustafson's model of nutrient flow in an aquarium [17, p. 589f]. It deals with using* *a radioactive tracer for the food chain consisting of two aquatic plankton varieties drifting with the currents. The variables in this three-dimensional system reflect the isotope concentrations in the water, a phytoplankton species, and a zooplankton species, respectively. The original model was an ODE model; a concise model would presumably have to use PDE (partial differential equations) to model spacial variations and the necessary drifts of species in the predator-prey part of the food chain; our DDE model here is a compromise between these two extremes. Therefore consider the three-dimensional linear DDE*

$$\dot{\vec{x}}(t) = \begin{bmatrix} -3 & 6 & 5 \\ 2 & -12 & 0 \\ 1 & 6 & -5 \end{bmatrix} \vec{x}(t - \frac{1}{100}) \tag{10}$$

with initial condition $\vec{x}([0,1]) \equiv [10,0,0]$ *and a conjectured MITL formula specifying the distance between the isotope concentrations of two aquatic plankton varieties always stays below* 10 *in a bounded time* $[0,50]$*, i.e.,* $\Box_{[0,50]} \mid x_2 - x_3 \mid \leq 10$*.*

Using Taylor models of degree 1, we calculate the operator relating successive parameter vectors to be

$$\boldsymbol{A}(n+1) = \begin{bmatrix} 1 & \frac{1}{100} & 0 & 0 & 0 & 0 \\ -3 & -3\xi_1 & 6 & 6\xi_1 & 5 & 5\xi_1 \\ 0 & 0 & 1 & \frac{1}{100} & 0 & 0 \\ 2 & 2\xi_2 & -12 & -12\xi_2 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & \frac{1}{100} \\ 1 & \xi_3 & 6 & 6\xi_3 & -5 & -5\xi_3 \end{bmatrix} \boldsymbol{A}(n)\,,$$

In this example, the solver outputs that the system is *safe*, which means that any state satisfying the target property is *unreachable* within depth 50 w.r.t. the over-approximation model of the DDE. This constitutes a rigorous proof that the system actually satisfies the given property.

```
DECL
-- the range of each variable has to be bounded
  float [-1000, 1000] a01, a11, a02, a12, a03, a13, x1, x2, x3;
  float [0,1/100] t, xi1, xi2, xi3;

-- define counter for the bounded verification problem
  int [0,49] counter;

INIT
-- initial values for the three components of the state
  x1  = 10;
  x2  = 0;
  x3  = 0;
-- initialize Taylor coefficients
  a01 = 10;
  a11 = 0;
  a02 = 0;
  a12 = 0;
  a03 = 0;
  a13 = 0;
-- initialize the counter
  counter = 0;

TRANS
--description of the transition system of DDE model
  x1' = a01' + a11'*t;
  x2' = a02' + a12'*t;
  x3' = a03' + a13'*t;
  a01' = a01 + (1/100)*a11;
  a11' = ((-3)*a01) - ((3*xi1)*a11) + (6*a02)
                    + ((6*xi1)*a12) + (5*a03) + ((5*xi1)*a13);
  a02' = a02 + (1/100)*a12;
  a12' = (2*a01) + ((2*xi2)*a11) - (12*a02) - ((12*xi2)*a12);
  a03' = a03 + (1/100)*a13;
  a13' = a01 + (xi3*a11) + (6*a02) + ((6*xi3)*a12)
                    - (5*a03) - ((5*xi3)*a13);

-- increment the counter by 1 for each time frame
  counter' = counter + 1;

TARGET
-- state to be reached in bounded time
  abs(x2-x3) > 10 and counter <= 49;
```

Listing 4: The encoding of Example 3 in iSAT3.

Along these lines, we should point out that the above verification procedure for temporal specifications could obviously fail, in the sense of providing false negatives and corresponding counter-examples, due to excessive over-approximation of the DDE's solution or due to the conditions used for MITL satisfaction not being necessary, but just sufficient. As the former problem would be induced by selecting an insufficient bound on the degree of the Taylor forms, one could simply select a higher degree. Therefore it should, however, be clear that the negative verdict actually is spurious due excessive over-approximation. Automatic methods to check whether the reported counterexample is spurious or not remain to be developed. One solution to this problem could be by using *counter-example guided abstraction refinement* (CEGAR) [8] for enhancing the over-approximation model. Another solution could be by refining the over-approximation Taylor model using sensitivity analysis, and hence enhancing the constructed conditions on the model. That latter solution aims at eliminating the wrapping effect due to the dependency issue in interval arithmetic. A pertinent algorithm based on this idea is currently under development and will be exposed in future work.

6 Conclusion and Future Work

In this paper, we have elaborated a method to verify/falsify temporal specifications of time-delay systems modeled by a simple class of delay differential equations (DDEs) with a single constant delay. Several dynamical systems can be modeled by DDEs with a single constant delay as in biology [16, 26], optics [19], economics [41, 42], ecology [12], to name just a few. As requirements specification language, we have exploited metric interval temporal logic (MITL) [1] with continuous-time semantics on the solutions of the DDEs. We have built our method around a fixed degree interval-based Taylor over-approximation technique [44] in order to provide a safe enclosure method for DDEs, thereby obtaining timed state sequences spanned by the piecewise valid Taylor coefficients. In this way, the continuous semantics of the MITL formulae is reduced to a time-discrete problem on timed state sequences in terms of Taylor coefficients. Then, we have devised sufficient conditions on these timed state sequences recovering the continuous-time interpretation of MITL on the actual solutions of the DDEs. To achieve this, we have first built sufficient conditions for validation of the atomic predicates over time frames of the Taylor over-approximation model of DDE. We have then extended this approach to arbitrary bounded MITL formulae in negation normal form. Exploiting this as a tableaux or using a related encoding as a bounded model checking (BMC) problem, we could employ arithmetic SMT solver addressing (a.o.) polynomial arithmetic as a tool able to automatically provide certificates of temporal properties for DDEs. In our case, we have used the `iSAT3` solver, which is the third implementation of the iSAT algorithm [13]. In very first experiments on simple DDEs, the `iSAT3` solver proved able to solve the temporal properties expressed in MITL formulae, thereby safely determining satisfaction of the formulae in an over-approximation setting. We were able to verify formulae of temporal logic also involving Boolean connectives and temporal modalities, like the *(bounded) until* operator.

We have presented some examples to demonstrate our method. The soundness of the method is guaranteed due to the over-approximation employed in DDE enclosure by Taylor forms and the sufficient conditions of determining the truth values of the atomic propositions over the time frames. Such over-approximation may, however, provide spurious counterexamples in case of a failing verification attempt, which ought to be disambiguated from true counterexamples. To resolve that ambiguity in case of a negative verdict, as a future work, further techniques remain to be developed. We may build our idea on the general *counter-example guided abstraction refinement* (CEGAR) technique [8]. Another solution could be by refinement based on sensitivity analysis. A pertinent algorithm based on sensitivity analysis is currently under development and will be exposed in future work.

In control applications, one may furthermore want to combine delayed feedback, as imposed a.o. by networked control, with immediate state feedback modeled by ordinary differential equations (ODEs). We may investigate in more detail some algorithms to handle such cases in the near future. The main idea is based on a layered combination of Taylor-model computation for ODE, e.g., [31], with the ideas imposed in [44] for DDE. In this way, we may extend our method exposed herein to verify the temporal properties of dynamical systems modeled by the combination of ODE and DDE. In subsequent steps, we plan to extend the method even further to more general kinds of DDE, like DDE with multiple different discrete delays, DDE with randomly distributed delay, or DDE with time-dependent or more generally state-dependent delay [23]. Finally, we would like to point out that in this paper, we essentially have presented a verification method based on model checking to design time-delay continuous systems modeled by a simple class of DDEs. This method may also be used in interactive proofs and stepwise refinement of hybrid systems featuring delayed feedback, akin to the methods developed for traditional hybrid systems [2, 5].

Acknowledgement

This research is funded by the Deutsche Forschungsgemeinschaft as part of the Research Training Group DFG GRK 1765 SCARE. The authors would like also to thank Ahmed Mahdi from University of Oldenburg for helpful discussions.

References

[1] R. Alur, T. Feder, and T.A. Henzinger. The benefits of relaxing punctuality. *Journal of the ACM*, 43(1):116–146, 1996. doi:10.1145/227595.227602.

[2] G. Babin, Y.A. Ameur, S. Nakajima, and M. Pantel. Refinement and proof based development of systems characterized by continuous functions. In X. Li, Z. Liu, and W. Yi, editors, *Dependable Software Engineering: Theories, Tools, and Applications - First International Symposium, SETTA 2015, Nanjing, China, November 4-6, 2015, Proceedings*, volume 9409 of *Lecture Notes in Computer Science*, pages 55–70. Springer, 2015. doi:10.1007/978-3-319-25942-0_4.

[3] R. Bellman and K.L. Cooke. Differential-difference equations. Technical Report R-374-PR, The RAND Corporation, Santa Monica, California, January 1963.

[4] M. Berz and K. Makino. Verified integration of ODEs and flows using differential algebraic methods on high-order Taylor models. *Reliable Computing*, 4(4):361–369, 1998. doi:10.1023/A:1024467732637.

[5] M.J. Butler, J.-R. Abrial, and R. Banach. Modelling and refining hybrid systems in Event-B and Rodin. In L. Petre and E. Sekerinski, editors, *From Action Systems to Distributed Systems - The Refinement Approach.*, pages 29–42. Chapman and Hall/CRC, 2016. doi:10.1201/b20053-5.

[6] M. Chen, M. Fränzle, Y. Li, P. Nazier Mosaad, and N. Zhan. Validated simulation-based verification of delayed differential dynamics. In J.S. Fitzgerald, C.L. Heitmeyer, S. Gnesi, and A. Philippou, editors, *FM 2016: Formal Methods - 21st International Symposium, Limassol, Cyprus, November 9-11, 2016, Proceedings*, volume 9995 of *Lecture Notes in Computer Science*, pages 137–154, 2016. doi:10.1007/978-3-319-48989-6_9.

[7] A. Chutinan and B.H. Krogh. Computing polyhedral approximations to flow pipes for dynamic systems. In *Decision and Control, 1998. Proceedings of the 37th IEEE Conference on*, volume 2, pages 2089–2094. IEEE, 1998. doi:10.1109/CDC.1998.758642.

[8] E.M. Clarke, O. Grumberg, S. Jha, Y. Lu, and H. Veith. Counterexample-guided abstraction refinement. In E.A. Emerson and A.P. Sistla, editors, *Computer Aided Verification, 12th International Conference, CAV 2000, Chicago, IL, USA, July 15-19, 2000, Proceedings*, volume 1855 of *Lecture Notes in Computer Science*, pages 154–169. Springer, 2000. doi:10.1007/10722167_15.

[9] A. Eggers, M. Fränzle, and C. Herde. SAT modulo ODE: A direct SAT approach to hybrid systems. In S.D. Cha, J.-Y. Choi, M. Kim, I. Lee, and M. Viswanathan, editors, *Automated Technology for Verification and Analysis, 6th International Symposium, ATVA 2008, Seoul, Korea, October 20-23, 2008. Proceedings*, volume 5311 of *Lecture Notes in Computer Science*, pages 171–185. Springer, 2008. doi:10.1007/978-3-540-88387-6_14.

[10] G.E. Fainekos, A. Girard, and G.J. Pappas. Temporal logic verification using simulation. In E. Asarin and P. Bouyer, editors, *Formal Modeling and Analysis of Timed Systems, 4th International Conference, FORMATS 2006, Paris, France, September 25-27, 2006, Proceedings*, volume 4202 of *Lecture Notes in Computer Science*, pages 171–186. Springer, 2006. doi:10.1007/11867340_13.

[11] G.E. Fainekos and G.J. Pappas. Robustness of temporal logic specifications for finite state sequences in metric spaces. Technical report, Technical Report MS-CIS-06-05, Dept. of CIS, Univ. of Pennsylvania, 2006.

[12] J. Fort and V. Méndez. Time-delayed theory of the neolithic transition in Europe. *Physical review letters*, 82(4):867–870, 1999. doi:10.1103/PhysRevLett.82.867.

[13] M. Fränzle, C. Herde, T. Teige, S. Ratschan, and T. Schubert. Efficient solving of large non-linear arithmetic constraint systems with complex boolean structure. *Journal on Satisfiability, Boolean Modeling and Computation*, 1(3-4):209–236, 2007. URL: https://satassociation.org/jsat/index.php/jsat/article/view/16/12.

[14] S. Gao, S. Kong, and E.M. Clarke. Satisfiability modulo ODEs. In *Formal Methods in Computer-Aided Design, FMCAD 2013, Portland, OR, USA, October 20-23, 2013*, pages 105–112. IEEE, 2013. doi:10.1109/FMCAD.2013.6679398.

[15] A. Girard. Reachability of uncertain linear systems using zonotopes. In M. Morari and L. Thiele, editors, *Hybrid Systems: Computation and Control, 8th International Workshop, HSCC 2005, Zurich, Switzerland, March 9-11, 2005, Proceedings*, volume 3414 of *Lecture Notes in Computer Science*, pages 291–305. Springer, 2005. doi:10.1007/978-3-540-31954-2_19.

[16] L. Glass and M.C. Mackey. *From clocks to chaos: the rhythms of life*. Princeton University Press, 1988.

[17] G.B. Gustafson. Systems of differential equations. In *Manuscript for Course Eng Math 2250-1 Spring 2014*, chapter 11. Dpt. of Mathematics, University of Utah, 2014.

[18] Z. Huang, C. Fan, and S. Mitra. Bounded invariant verification for time-delayed nonlinear networked dynamical systems. *Nonlinear Analysis: Hybrid Systems*, 23:211–229, 2017. doi:10.1016/j.nahs.2016.05.005.

[19] K. Ikeda and K. Matsumoto. High-dimensional chaotic behavior in systems with time-delayed feedback. *Physica D: Nonlinear Phenomena*, 29(1-2):223–235, 1987. doi:10.1016/0167-2789(87)90058-3.

[20] R. Koymans. Specifying real-time properties with metric temporal logic. *Real-Time Systems*, 2(4):255–299, 1990. doi:10.1007/BF01995674.

[21] S. Kupferschmid and B. Becker. Craig interpolation in the presence of non-linear constraints. In U. Fahrenberg and S. Tripakis, editors, *Formal Modeling and Analysis of Timed Systems - 9th International Conference, FORMATS 2011, Aalborg, Denmark, September 21-23, 2011. Proceedings*, volume 6919 of *Lecture Notes in Computer Science*, pages 240–255. Springer, 2011. doi:10.1007/978-3-642-24310-3_17.

[22] A.B. Kurzhanski and P. Varaiya. Ellipsoidal techniques for reachability analysis. In N.A. Lynch and B.H. Krogh, editors, *Hybrid Systems: Computation and Control, Third International Workshop, HSCC 2000, Pittsburgh, PA, USA, March 23-25, 2000, Proceedings*, volume 1790 of *Lecture Notes in Computer Science*, pages 202–214. Springer, 2000. doi:10.1007/3-540-46430-1_19.

[23] M. Lakshmanan and D.V. Senthilkumar. *Dynamics of nonlinear time-delay systems*. Springer Science & Business Media, 2011. doi:10.1007/978-3-642-14938-2.

[24] C. Le Guernic and A. Girard. Reachability analysis of linear systems using support functions. *Nonlinear Analysis: Hybrid Systems*, 4(2):250–262, 2010. doi:10.1016/j.nahs.2009.03.002.

[25] R. Lohner. *Einschließung der Lösung gewöhnlicher Anfangs- und Randwertaufgaben*. PhD thesis, Fakultät für Mathematik der Universität Karlsruhe, Karlsruhe, 1988.

[26] M.C. Mackey and L. Glass. Oscillation and chaos in physiological control systems. *Science*, 197(4300):287–289, 1977. doi:10.1126/science.267326.

[27] O. Maler and D. Nickovic. Monitoring temporal properties of continuous signals. In Y. Lakhnech and S. Yovine, editors, *Formal Techniques, Modelling and Analysis of Timed and Fault-Tolerant Systems, Joint International Conferences on Formal Modelling and Analysis of Timed Systems, FORMATS 2004 and Formal Techniques in Real-Time and Fault-Tolerant Systems, FTRTFT 2004, Grenoble, France, September 22-24, 2004, Proceedings*, volume 3253 of *Lecture Notes in Computer Science*, pages 152–166. Springer, 2004. doi:10.1007/978-3-540-30206-3_12.

[28] Z. Manna and A. Pnueli. *The temporal logic of reactive and concurrent systems - specification*. Springer, 1992. doi:10.1007/978-1-4612-0931-7.

[29] K.L. McMillan. Interpolation and SAT-based model checking. In W.A. Hunt Jr. and F. Somenzi, editors, *Computer Aided Verification, 15th International Conference, CAV 2003, Boulder, CO, USA, July 8-12, 2003, Proceedings*, volume 2725 of *Lecture Notes in Computer Science*, pages 1–13. Springer, 2003. doi:10.1007/978-3-540-45069-6_1.

[30] R.E. Moore. Automatic local coordinate transformation to reduce the growth of error bounds in interval computation of solutions of ordinary differential equations. In L. B. Ball, editor, *Error in Digital Computation*, volume II, pages 103–140. Wiley, New York, 1965.

[31] M. Neher, K.R. Jackson, and N.S. Nedialkov. On Taylor model based integration of ODEs. *SIAM Journal on Numerical Analysis*, 45(1):236–262, 2007. doi:10.1137/050638448.

[32] J. Ouaknine and J. Worrell. On the decidability of metric temporal logic. In *20th IEEE Symposium on Logic in Computer Science (LICS 2005), 26-29 June 2005, Chicago, IL, USA, Proceedings*, pages 188–197. IEEE Computer Society, 2005. doi:10.1109/LICS.2005.33.

[33] J. Ouaknine and J. Worrell. Some recent results in metric temporal logic. In F. Cassez and C. Jard, editors, *Formal Modeling and Analysis of Timed Systems, 6th International Conference, FORMATS 2008, Saint Malo, France, September 15-17, 2008. Proceedings*, volume 5215 of *Lecture Notes in Computer Science*, pages 1–13. Springer, 2008. doi:10.1007/978-3-540-85778-5_1.

[34] E. Plaku, L.E. Kavraki, and M.Y. Vardi. Falsification of LTL safety properties in hybrid systems. In S. Kowalewski and A. Philippou, editors, *Tools and Algorithms for the Construction and Analysis of Systems, 15th International Conference, TACAS 2009, Held as Part of the Joint European Conferences on Theory and Practice of Software, ETAPS 2009, York, UK, March 22-29, 2009. Proceedings*, volume 5505 of *Lecture Notes in Computer Science*, pages 368–382. Springer, 2009. doi:10.1007/978-3-642-00768-2_31.

[35] Stephen Prajna and Ali Jadbabaie. Methods for safety verification of time-delay systems. In *Proceedings of the 44th IEEE Conference on Decision and Control*, pages 4348–4353. IEEE, 2005. doi:10.1109/CDC.2005.1582846.

[36] A. Robinson and A. Voronkov. *Handbook of automated reasoning*, volume 1. Elsevier, 2001. doi:10.1016/B978-0-444-50813-3.50032-1.

[37] S. Sankaranarayanan and G.E. Fainekos. Falsification of temporal properties of hybrid systems using the cross-entropy method. In T. Dang and I.M. Mitchell, editors, *Hybrid Systems: Computation and Control (part of CPS Week 2012), HSCC'12, Beijing, China, April 17-19, 2012*, pages 125–134. ACM, 2012. doi:10.1145/2185632.2185653.

[38] K. Scheibler. *iSAT3 Manual*, December 2016. Available at https://projects.avacs.org/attachments/download/671/isat3_manual-0.03-20161213.pdf.

[39] M. Sheeran, S. Singh, and G. Stålmarck. Checking safety properties using induction and a SAT-solver. In W.A. Hunt Jr. and S.D. Johnson, editors, *Formal Methods in Computer-Aided Design, Third International Conference, FMCAD 2000, Austin, Texas, USA, November 1-3, 2000, Proceedings*, volume 1954 of *Lecture Notes in Computer Science*, pages 108–125. Springer, 2000. doi:10.1007/3-540-40922-X_8.

[40] O. Stauning. *Automatic Validation of Numerical Solutions*. PhD thesis, Technical University of Denmark, Lyngby, 1997. URL: http://www2.imm.dtu.dk/documents/ftp/phdliste/phd36_97.ps.

[41] M. Szydłowski and A. Krawiec. The stability problem in the Kaldor–Kalecki business cycle model. *Chaos, Solitons & Fractals*, 25(2):299–305, 2005. doi:10.1016/j.chaos.2004.11.012.

[42] M. Szydłowski, A. Krawiec, and J. Toboła. Nonlinear oscillations in business cycle model with time lags. *Chaos, Solitons & Fractals*, 12(3):505–517, 2001. doi:10.1016/S0960-0779(99)00207-6.

[43] G.S. Tseitin. On the complexity of derivation in propositional calculus. In *Automation of reasoning*, pages 466–483. Springer, 1983. doi:10.1007/978-3-642-81955-1_28.

[44] L. Zou, M. Fränzle, N. Zhan, and P. Nazier Mosaad. Automatic verification of stability and safety for delay differential equations. In D. Kroening and C.S. Pasareanu, editors, *Computer Aided Verification - 27th International Conference, CAV 2015, San Francisco, CA, USA, July 18-24, 2015, Proceedings, Part II*, volume 9207 of *Lecture Notes in Computer Science*, pages 338–355. Springer, 2015. doi:10.1007/978-3-319-21668-3_20.