Validated Simulation-Based Verification of Delayed Differential Dynamics *

Mingshuai Chen[1], Martin Fränzle[2], Yangjia Li[1], Peter N. Mosaad[2], and Naijun Zhan[1]

[1] State Key Lab. of Computer Science, Institute of Software, CAS, China
{chenms,yangjia,znj}@ios.ac.cn,
[2] Dpt. of Computing Science, C. v. Ossietzky Universität Oldenburg, Germany
{fraenzle,peter.nazier.mosaad}@informatik.uni-oldenburg.de

Abstract. Verification by simulation, based on covering the set of time-bounded trajectories of a dynamical system evolving from the initial state set by means of a finite sample of initial states plus a sensitivity argument, has recently attracted interest due to the availability of powerful simulators for rich classes of dynamical systems. System models addressed by such techniques involve ordinary differential equations (ODEs) and can readily be extended to delay differential equations (DDEs). In doing so, the lack of validated solvers for DDEs, however, enforces the use of numeric approximations such that the resulting verification procedures would have to resort to (rather strong) assumptions on numerical accuracy of the underlying simulators, which lack formal validation or proof. In this paper, we pursue a closer integration of the numeric solving and the sensitivity-related state bloating algorithms underlying verification by simulation, together yielding a safe enclosure algorithm for DDEs suitable for use in automated formal verification. The key ingredient is an on-the-fly computation of piecewise linear, local error bounds by nonlinear optimization, with the error bounds uniformly covering sensitivity information concerning initial states as well as integration error.

1 Introduction

Delayed coupling between state variables of dynamic systems occurs in many domains. Prominent examples include population dynamics, where birth rate follows changes in population size with a delay related to reproductive age, spreading of infectious diseases, where delay is induced by the incubation period, exhaust gas control in internal combustion engines, where relevant sensors, like the λ probe, are located downstream the exhaust system such that gas transport induces a delay between the controlled combustion processes and sensing their effect, or networked control systems with their associated transport delays when forwarding data through the communication network,

* The first, third and fifth authors are supported partly by "973 Program" under grant No. 2014CB340701, by NSFC under grant 91418204, by CDZ project CAP (GZ 1023), and by the CAS/SAFEA International Partnership Program for Creative Research Teams. The second and fourth authors are supported partly by Deutsche Forschungsgemeinschaft within the Research Training Group "SCARE - System Correctness under Adverse Conditions" (DFG GRK 1765).

to name just a few. Most examples feature feedback dynamics and it should be obvious that the presence of feedback delays reduces controllability due to the impossibility of immediate reaction and enhances likelihood of transient overshoot or even oscillation in the feedback system. In fact, the introduction of delays into a feedback system may reduce stabilization rates of or even destabilize an otherwise stable system, it may provoke overshoot and drive the system to otherwise unreachable states, it is likely to stretch dwell times, and it may induce residual error that never cancels. As this implies that safety or stability certificates obtained on idealized, delay-free models of systems prone to delayed coupling may be erratic, automated methods for system verification ought to address models of system dynamics reflecting delays, rendering verification tools only addressing ordinary differential equations (ODE) and their derived models, like hybrid automata, vastly insufficient. It can well be argued that such tools should better address delay differential equations (DDE), as introduced in [2].

Generalizing techniques developed for ODE to DDE is not as straightforward as it may seem at first glance. The reason is that the future evolution of a DDE is no longer governed by the current state instant only, but depends on a chunk of its past trajectory, such that introducing a delay immediately renders a system with finite-dimensional state into an infinite-dimensional dynamical system. Consequently, approximate numerical methods for solving DDEs as well as methods for stability analysis have well been developed in the field of control, while in automatic verification, hitherto only few approaches address the effects of delays due to the immediate impact of delays on the structure of the state spaces to be traversed by state-exploratory methods.

In this paper, we address this problem by suitably adapting the paradigm of verification by simulation to delay differential equations. Verification by simulation provides bounded-time verification of dynamical systems based on covering the full set of time-bounded trajectories of a dynamical system evolving from the initial state set by means of a finite sample of initial states plus a sensitivity argument. To achieve this, a sufficiently dense sample of initial states is drawn from the set of all possible initial states, numeric simulation is then used for obtaining the trajectories originating from the sample points, and finally a quantitative sensitivity argument permits to pessimistically over-approximate the "tube" of trajectories originating from arbitrary start states by means of "bloating" the individual simulated trajectories into a neighborhood of the radius given by the bound on sensitivity on the start state, see e.g. [7,19,8,14]. If a validated numerical solver is used for the simulations, the above procedure will immediately yield a safe over-approximation of the set of possible trajectories; else, more aggressive bloating additionally covering the possible inaccuracies of numeric integration of differential equations has to be employed to obtain a sound, validated method.

The class of systems we approach features delayed differential dynamics governed by DDE of the following form:

$$\begin{cases} \dot{\mathbf{x}}(t) = \boldsymbol{f}(\mathbf{x}(t), \mathbf{x}(t-r_1), \ldots, \mathbf{x}(t-r_k)), & t \in [0,\infty) \\ \mathbf{x}(t) = \boldsymbol{g}(t), & t \in [-r_{\max}, 0] \end{cases} \tag{1}$$

It thus involves a combination of ODE and DDE with multiple constant delays $r_i > 0, i = 1, \ldots, k$. Here, $r_{\max} = \max\{r_1, \ldots, r_k\}$ is the maximal delay, $\mathbf{x} : \mathbb{R}_{\geq -r_{\max}} \mapsto \mathbb{R}^n$ is a trajectory, $\boldsymbol{f} : (\mathbb{R}^n)^{k+1} \mapsto \mathbb{R}^n$ a vector field, and $\boldsymbol{g} : [-r_{\max}, 0] \mapsto \mathbb{R}^n$ is

a continuous function providing the initial condition. This form of equations has been successfully used to model various real world systems in the fields of, e.g., biology, control theory, and economics.

Generally speaking, formal verification of temporally unbounded reachability properties of system dynamics governed by Eq. (1) inherits undecidability from similar properties for ODE. Therefore, and also due to our wish to use simulation as an underlying mechanism of system analysis, we restrict ourselves to time-bounded reachability problems. Such a time-bounded reachability problem for a given model of the form (1) is parameterized by a temporal horizon (i.e., a time bound) set by the user, a set of initial states which in the case of DDE generalizes to constant functions over the time frame $[-r_{\max}, 0]$ immediately preceding system start, and a set of unsafe states that system dynamics is expected to avoid. The proof obligation is to determine whether there exists a *trajectory* of the model starting in some initial state which reaches any unsafe state within the time bound. In our approach, we first trigger a set of numerical approximations of the behaviours from a finite sampling of the initial states. Such a simulation does not yield a trajectory, but rather a timed trace, i.e., a sequence of time-stamp value pairs. Along each simulation run, we bloat each snapshot, i.e., each time-stamp value pair by a distance determined via an *error bound* computed automatically on-the-fly, where the error bound incorporates coverage and sensitivity information concerning the sampled start states as well as the integration error incurred by numerical solving. The union of these bloatings covers all time-bounded trajectories possibly evolving from all initial states, and thus yields an over-approximation of the states reachable from the initial set within the time bound. If this over-approximation proves safety in the sense that the cover of the reachable states is disjoint from the unsafe states, or conversely if the simulation produces a valid counter-example in the sense that it can prove that a trajectory inevitably hits the unsafe states, then the algorithm generates the corresponding verdict. Otherwise, it refines the sample drawn from the initial states, thus requiring less aggressive bloating of simulation runs, and computes a more precise over-approximation.

Our approach is distinguished from competing approaches by providing a validated verification-by-simulation paradigm for DDE. Given that validated methods for DDE enclosure are not readily available, it achieves this by pursuing a closer than traditional integration of the numeric solving and the sensitivity-related state bloating algorithms underlying verification by simulation, together yielding a safe enclosure algorithm for DDE guaranteed to contain the true solution. The key ingredient is an on-the-fly computation of piecewise linear, local error bounds by nonlinear optimization, which provides an alternative to established methods computing discrepancy bounds from Lipschitz constants and Jacobians, as employed in [12].

To illustrate our approach, some experimental results obtained on several benchmark systems involving delayed differential dynamics are demonstrated.

Related work. Zou, Fränzle *et al*. proposed in [26] a procedure for generating stability and safety certificates for the simplest class of DDEs of the form $\dot{\mathbf{x}}(t) = \boldsymbol{f}(\mathbf{x}(t-r))$. This is achieved by iterating interval-based Taylor over-approximations of the time-wise segments of the solution to a DDE, which depends essentially on the fact that the interval coefficients of the solution over the time interval $(n, n+1]$ can be represented as a function of those of the solution over $(n-1, n]$. Extracting the operator mapping

coefficients at one time frame to those of the next, one obtains a time-invariant discrete-time dynamical system. Thus, stability analysis and safety verification of the original DDE is reduced to appropriate counterparts encoding these properties on the resulting time-invariant discrete-time dynamical system. This approach does not immediately generalize to mixed ODE-DDE forms as in Eq. (1), as the delayed parts of the dynamics would there function as inputs to an ODE with input, rendering the above operator time-variant. Though this is doable in principle, we have herein opted for the more immediate approach of verification by simulation.

In [21], Pola *et al.* proposed an approach abstracting incrementally input-to-state stable (δ-ISS) nonlinear control systems with constant and known delays to finite-state symbolic models, and establish approximate bisimilarity between them. In [20], they extended the work in [21] to incrementally-input-delay-to-state stable (δ-IDSS) nonlinear control systems with time-varying and unknown delays, and proved that the original δ-IDSS nonlinear control systems and the corresponding symbolic models are alternating approximately bisimilar. The crucial differences between their work and ours lie in, firstly, their approach being confined to δ-ISS nonlinear control systems, while our approach being applicable to any kind of nonlinear control systems with constant and known time delays. So, our method relaxes a problematic applicability condition. Second, their approach can do unbounded verification of time-delay systems, while our approach currently can only conduct bounded verification. Third, their approach can be applied to δ-IDSS nonlinear control systems with time-varying and unknown delays, while our approach cannot yet. It is a crucial aspect of our future work to extend our approach to nonlinear control systems with time-varying and unknown delays, without sacrificing its applicability beyond δ-IDSS systems.

Verifying delayless dynamical systems, in particular ODE, using numerical simulations has well been studied, e.g., in [7,19,8,14], where similar concepts based on sensitivity information provided by discrepancy functions or simulation functions, respectively, have been presented to bloat the traces obtained from simulations to "trajectory tubes" over-approximating time-bounded reach sets. While the first settings resorted to user-supplied sensitivity information, Fan and Mitra in [12] proposed an algorithm for automatically computing piecewise exponential discrepancy functions. This algorithm pessimistically estimates the sensitivity of the ODE on its initial value, but also takes assumed error bounds of the numerical simulation, which in that case is Matlab's `ode45` solver, into account. This, however, renders the soundness of this algorithm dependent on the assumption that Matlab's built-in ODE solver can always guarantee those numerical error bounds, while it is possible to find extremely stiff ODEs as follows for which the solver returns very inaccurate results.

$$\dot{x}(t) = 1 + \delta_a(x - \sqrt{2}), \text{ with } \delta_a(y) = \frac{1}{a\sqrt{\pi}} \mathrm{e}^{-y^2/a^2} \tag{2}$$

$\delta_a(y)$ approximates the *Dirac δ function* [6] modelling a tall narrow spike around $y = 0$, where the spike shrinks as $a \to 0$. When Eq. (2) is simulated with $a = 10^{-3}$ by Matlab's ODE solver `ode45`, results show that the solver can detect the sharp increment of the derivative with a user-specified `MaxStep` as 0.01, while not the case with 0.1. Furthermore, adjusting the simulation step width could not essentially cure the problem, yet just shifts it to a smaller a for which the solver fails to identify the leaping trajectory

and instead follows straight-line dynamics. This motivates us to address the issue of numerical errors in discrepancy computation. Moreover, the method in [12] requires computations of a global Lipschitz constant as well as a bound on the eigenvalues of the Jacobians within a region, which may not be feasible in some dynamical systems.

2 Problem Formulation

Notations. For a vector $\mathbf{x} \in \mathbb{R}^n$, x_i refers to its ith component, and $\|\mathbf{x}\|$ denotes the ℓ^2-norm. The notation $\|\cdot\|$ extends to an $n \times n$ real matrix $A \in \mathbb{R}^{n\times n}$ with $\|A\| = \sqrt{\lambda_{\max}(A^{\mathrm{T}}A)}$, where $\lambda_{\max}(A)$ is the largest eigenvalue of A. For $\mathbf{x}, \mathbf{x}' \in \mathbb{R}^n$, $\|\mathbf{x}'-\mathbf{x}\|$ is the Euclidean distance between the points, and we define for $\delta \geq 0$, $\mathcal{B}_\delta(\mathbf{x}) = \{\mathbf{x}' \in \mathbb{R}^n | \|\mathbf{x}' - \mathbf{x}\| \leq \delta\}$ as the closed ball of radius δ centered at $\mathbf{x}$. For a set $S \subseteq \mathbb{R}^n$, $\mathcal{B}_\delta(S) = \cup_{\mathbf{x}\in S}\mathcal{B}_\delta(\mathbf{x})$. The diameter of a compact set S is $dia(S) = \sup_{\mathbf{x},\mathbf{x}'\in S}\|\mathbf{x}-\mathbf{x}'\|$, and a δ-*cover* of S is a finite collection of points $\mathcal{X}$ such that $S \subseteq \cup_{\mathbf{x}\in\mathcal{X}}\mathcal{B}_\delta(\mathbf{x})$. For a set $S \subseteq \mathbb{R}^n$, its convex hull is denoted as $conv(S)$.

Delayed dynamical systems. We consider a timed-bounded delayed dynamical system of the form

$$\begin{cases} \dot{\mathbf{x}}(t) = \boldsymbol{f}(\mathbf{x}(t), \mathbf{x}(t-r_1), \ldots, \mathbf{x}(t-r_k)), & t \in [0,\infty) \\ \mathbf{x}(t) \equiv \mathbf{x}_0 \in \Theta, & t \in [-r_k, 0] \end{cases}^{3}, \tag{3}$$

where $\mathbf{x}$ is the time-dependent *state* vector in $\mathbb{R}^n$, $\dot{\mathbf{x}}$ denotes its temporal derivative $\mathrm{d}\mathbf{x}/\mathrm{d}t$, and t is a real variable modelling time. The discrete delays are assumed to be ordered as $r_k > \ldots > r_1 > 0$, and the initial states are generalized to a constant function over $[-r_k, 0]$ taking values from a compact set Θ.

Let the vector-valued function $\boldsymbol{f} : (\mathbb{R}^n)^{k+1} \mapsto \mathbb{R}^n$ be continuous and continuously differentiable in the first argument, which implies that the system has a unique maximal *solution* (or *trajectory*) from each constant initial condition valued $\mathbf{x}_0 \in \mathbb{R}^n$, denoted as $\xi_{\mathbf{x}_0}(t) : [-r_k, \ell) \mapsto \mathbb{R}^n$, where $\ell = \infty$ holds if $\boldsymbol{f}$ is Lipschitz.

Example 1 (Gene Regulation [11,23]). The control of gene expression in cells is often modelled with time delays in equations of the form

$$\begin{cases} \dot{x_1}(t) = g(x_n(t-r_n)) - \alpha_1 x_1(t) \\ \dot{x_j}(t) = g(x_{j-1}(t-r_{j-1})) - \alpha_j x_j(t), 1 < j \leq n \end{cases}, \tag{4}$$

where the gene is transcribed producing mRNA (x_1), which is translated into enzyme x_2 that turn produces another enzyme x_3 and so on. The end product x_n acts to repress the transcription of the gene by $\dot{g} < 0$. Time delays are introduced to account for time involved in transcription, translation, and transport. The $\alpha_j > 0$ represent decay rates of the species. The dynamic described in Eq. (4) falls exactly into the scope of systems considered in this paper, and in fact, it instantiates a more general family of systems known as monotone cyclic feedback systems (MCFS) [18], which includes neural networks, testosterone control, and many other effects in systems biology.

[3] In general, the initial condition is represented by $\mathbf{x}(t) = \xi_0(t)$, for $t \in [-r_k, 0]$, where $\xi_0 \in \mathcal{X} \subseteq C^0([-r_k, 0], \mathbb{R}^n)$, $C^0([-r_k, 0], \mathbb{R}^n)$ stands for all continuous functions mapping from $[-r_k, 0]$ to $\mathbb{R}^n$, $\mathcal{X}$ is compact and bounded. So, we can let $\Theta = \cup_{\xi\in\mathcal{X}}\xi([-r_k, 0])$. Clearly, Θ is compact and bounded.

Safety verification problem. Given a set $\mathcal{U} \subseteq \mathbb{R}^n$ of unsafe or otherwise bad states, a delayed dynamical system of shape (3) is said to be (time-bounded) *safe* iff all the trajectories originating from any $\mathbf{x}_0 \in \Theta$ do not intersect with $\mathcal{U}$ (within the given time bound T), otherwise it is called *unsafe*.

3 Verification of Delayed Dynamical Systems via Simulation

Generating formal guarantees for DDEs of the form (3) tends to be challenging due to unavailability of guaranteed for solving them. We are trying to alleviate that problem by adopting approximate numeric methods, enhancing them with methods for rigorous error tracking, thus rendering them validated numerical methods, and adding sensitivity information for being able to cover sets of initial states based on simulating and bloating the trajectories originating from finitely many samples. This approach has been inspired by similar approaches for ODE, in particular the discrepancy functions of [12].

We will now expose in detail the overall procedure of simulation by verification, which hinges on the validated simulation of DDE that we will turn to in Sect. 4. For the sake of simplifying the exposition, we first consider the special case of delayed dynamical systems featuring a single delay, as in

$$\begin{cases} \dot{\mathbf{x}}(t) = \boldsymbol{f}(\mathbf{x}(t), \mathbf{x}(t-r)), & t \in [0, \infty) \\ \mathbf{x}(t) \equiv \mathbf{x}_0 \in \Theta, & t \in [-r, 0] \end{cases} . \tag{5}$$

In this case, the differential dynamics is a function $\boldsymbol{f}(\mathbf{x}, \mathbf{u})$ of two states, namely the current state $\mathbf{x}$ and the past state $\mathbf{u}$.

The basic idea of simulation-based verification of a DDE (5), as implemented by Algorithm 1, can be sketched as follows:
First, we build on a validated simulation procedure Simulation, whose design is shown in Sect. 4. Given a delayed dynamical system as above, a subset $\mathcal{X}_0 \subset \Theta$ of the initial states, and a time bound T, Simulation yields a *simulation trace* $(t_0, \mathbf{y}_0), \ldots, (t_n, \mathbf{y}_n)$ consisting of pairs of time stamps $t_i \in [0, T]$ and states $\mathbf{y}_i \in \mathbb{R}^n$ with $\mathbf{y}_0 = \mathbf{x}_0$, as well as a sequence of *local error bounds* $d_0, d_1, \ldots, d_n \geq 0$ providing a *validation* of this trace observing the following two properties:

P1: $0 = t_0 < t_1 < \ldots < t_n = T$, i.e., the time stamps in the trace are ascending and cover the temporal horizon of interest.

P2: For each of the trajectories $\xi_{\mathbf{x}_0}(t)$ of (5) starting from any point $\mathbf{x}_0 \in \mathcal{X}_0$, the validation property

$$(\xi_{\mathbf{x}_0}(t), t) \in \mathit{conv}\left((\mathcal{B}_{\mathbf{d}_i}(\mathbf{y}_i) \times \{t_i\}) \cup (\mathcal{B}_{\mathbf{d}_{i+1}}(\mathbf{y}_{i+1}) \times \{t_{i+1}\}) \right) \tag{6}$$

holds for each $t \in [t_i, t_{i+1}]$, $i = 0, 1, \ldots, n-1$. I.e., the reported error bounds $\mathbf{d}_i$ span a piecewise linear tube around the points $(\mathbf{y}_i, t_i)$ in the simulation trace such that $\xi_{\mathbf{x}_0}(t)$ is properly enclosed for any $\mathbf{x}_0 \in \mathcal{X}_0$ and any $t \in [0, T]$.

Then, time-bounded safety verification of system (5) can be obtained as follows:

Algorithm 1: Simulation-based Verification for Delayed Dynamical Systems

input : The dynamics $\boldsymbol{f}(\mathbf{x},\mathbf{u})$, delay term r, initial set $\mathcal{X}_0$, unsafe set $\mathcal{U}$, time bound T, precision ϵ.

```
/* initialization */
1  R ← ∅; δ ← dia(X_0)/2; τ ← τ_0;
2  X ← δ-Partition(X_0);
3  while X ≠ ∅ do
4      if δ < ε then
5          return (UNKNOWN, R);
6      for B_δ(x_0) ∈ X do
7          ⟨t, y, d⟩ ← Simulation(B_δ(x_0), f(x, u), r, τ, T);
8          T ← ⋃_{n=0}^{N-1} conv(B_{d_n}(y_n) ∪ B_{d_{n+1}}(y_{n+1}));
9          if T ∩ U = ∅ then
10             X ← X\B_δ(x_0);  R ← R ∪ T;
11         else if ∃i. B_{d_i}(y_i) ⊆ U then
12             return (UNSAFE, T);
13         else
14             X ← X\B_δ(x_0);  X ← X ∪ (δ/2)-Partition(B_δ(x_0));
15     δ ← δ/2;
16 return (SAFE, R);
```

1. At the beginning, we cover the given initial set $\mathcal{X}_0$ by a finite set of balls of radius δ; so, δ-Partition$(\mathcal{X}_0)$ in line 2 of Algorithm. 1 returns a finite δ-cover of the compact set $\mathcal{X}_0$. We then call Simulation to each of these balls. For each ball B, we collect all states contained in the bloating of the N-step simulation trace $\mathbf{y}$ as $\mathcal{B}_{\mathbf{d}}(\mathbf{y}) = \bigcup_{n=0}^{N-1} \mathit{conv}(\mathcal{B}_{\mathbf{d}_n}(\mathbf{y}_n) \cup \mathcal{B}_{\mathbf{d}_{n+1}}(\mathbf{y}_{n+1}))$, cf. line 8. This yields an over-approximation of the states reachable from B following (5) within time up to T.
2. If the over-approximation of the reachable set thus obtained is disjoint to the unsafe set (line 9), then (5) is *safe* when starting in B; otherwise, if there exists a sampling point in the simulation which has its full bloating with the corresponding local error bound being contained in the unsafe set (line 11), then (5) is definitely *unsafe*. If none of these two conditions applies, we compute a finer partition of B (line 14), and we repeat the above procedure until the granularity of the partition becomes finer than the given threshold. In this case, we cannot give an answer whether or not (5) is safe and terminate with the inconclusive result *unknown*.

Obviously, our approach is different from existing approaches providing simulation-based verification for dynamical systems modeled by ordinary differential equations, like [7,8]. In our approach, the simulation procedure provides a rigorous validation of the above property P2, rather than relying on assumptions concerning numerical accuracy of the underlying simulator. Second, our approach covers rigorous simulation-based formal verification of DDE rather than just ODE. The correctness of the resulting algorithm is captured by the following theorem:

Theorem 1 (Correctness). *If* Simulation *satisfies above properties* P1 *and* P2 *(which will be verified in the next section), then Algorithm 1 terminates and its outputs are guaranteed to satisfy the following soundness properties:*

- *it reports (*SAFE*,*$\mathcal{R}$*) only if the system is safe.*
- *it reports (*UNSAFE*,* $\mathcal{T}$*) only if the system is unsafe and* $\mathcal{T}$ *is a counter-example.*

The general case of multiple different delays in Eq. (3) can be dealt with analogously to the case (5) of a single delay: we only need to allow $\mathbf{u}$ to have more components, meanwhile, we need to revise Algorithm 2 accordingly by introducing multiple different m_i as $m_1 \leftarrow r_1/\tau, \ldots, m_k \leftarrow r_k/\tau$. Thus, the delayed states y_{n-m_i}s can be exactly located when computing y_{n+1} by $\boldsymbol{f}(y_n, y_{n-m_1}, \ldots, y_{n-m_k})$ (line 6 in Algorithm 2) as well as when finding the minimal e (line 7 in Algorithm 2).

4 Validated Simulation

In this section, we elaborate on simulation and on computation of rigorous local error bounds to guarantee the *enclosure property* P2. Instead of directly computing the error bounds $d_0, \ldots, d_n$ accompanying the simulation trace $(t_0, \mathbf{y}_0), \ldots, (t_n, \mathbf{y}_n)$, we compute an initial error bound d_0 and a sequence $e_1, \ldots, e_n$ of error slopes recursively defining error bounds $E(t)$ for each $t \in [0, T]$ — and thus not only for time stamps in the simulated trace— as follows:

$$E(t) = \begin{cases} d_0, & \text{if } t = 0, \\ E(t_i) + (t - t_i)e_{i+1}, & \text{if } t \in [t_i, t_{i+1}]. \end{cases} \tag{7}$$

The validation property (P2) can thus be rewritten as

P2': For each of the trajectories $\xi_{\mathbf{x}_0}(t)$ of system (5) starting from any point $\mathbf{x}_0 \in \mathcal{X}_0$, the validation property

$$\xi_{\mathbf{x}_0}(t) \in \mathcal{B}_{E(t)}\left(\frac{(t - t_i)\mathbf{y}_i + (t_{i+1} - t)\mathbf{y}_{i+1}}{t_{i+1} - t_i}\right) \tag{8}$$

holds for each $t \in [t_i, t_{i+1}]$.

I.e., the e_i's provide the slopes of piecewise conic enclosures around the linear interpolations between the points $(t_i, \mathbf{y}_i)$ in the simulation trace.

The Simulation Algorithm. Inferring formal proofs from simulations essentially attributes to a validated numerical solver which can produce rigorous error bounds on the generated sampling points. We present in Algorithm 2 a procedure[4] Simulation that provides a trace of sampling points bundled with their local error bounds thus giving an over-approximation of the reachable set in terms of an initial state space.

The algorithm is provided with an initial ball $\mathcal{B}_\delta(\mathbf{x}_0)$ and it proceeds with a discrete simulation starting from $\mathbf{x}_0$ paced by a fixed stepsize τ. Three *list* structures (denoted as $[\![\cdot]\!]$) with the same length are introduced respectively as (1) $\mathbf{t}$: storing a sequence of time stamps on which the approximations are computed, (2) $\mathbf{y}$: keeping a sequence of sampling points that approximates the trajectory starting from $\mathbf{x}_0$, and (3) $\mathbf{d}$: capturing the corresponding sequence of local error bounds. Due to the nature of DDEs where the evolving of states may refer to those ahead of time $t_0 = 0$, we index the lists beginning

[4] For ease of presentation, we demonstrate the approach on DDEs with one single delay, and it readily extends to that with multiple delays as discussed in Sect. 3.

Algorithm 2: Simulation: a validated DDE solver producing rigorous bounds

input : The initial set $\mathcal{B}_\delta(\mathbf{x}_0)$, dynamics $\boldsymbol{f}(\mathbf{x},\mathbf{u})$, delay term r, stepsize τ, time bound T.
output: A triple $\langle \mathbf{t},\mathbf{y},\mathbf{d}\rangle$, where the components represent lists, with the same length, respectively for the time points, numerical approximations (possibly multi-dimensional), and the rigorous local error bounds.

/* initializing the lists, whose indices start from -1 */
1 $\mathbf{t} \leftarrow [\![-\tau, 0]\!]$; $\mathbf{y} \leftarrow [\![\mathbf{x}_0, \mathbf{x}_0]\!]$; $\mathbf{d} \leftarrow [\![0, \delta]\!]$;
/* r has to be divisible by τ (in FP numbers) */
2 $n \leftarrow 0$; $m \leftarrow r/\tau$;
3 **while** $\mathbf{t}_n < T$ **do**
4 $\quad t_{n+1} \leftarrow \mathbf{t}_n + \tau$;
$\quad$ /* approximating y_{n+1} using *forward Euler method* */
5 $\quad y_{n+1} \leftarrow \mathbf{y}_n + \boldsymbol{f}(\mathbf{y}_n, \mathbf{y}_{n-m}) * \tau$;
$\quad$ /* computing error slope by constrained optimization, where σ is a positive slack constant */
6 $\quad e_n \leftarrow$ **Find** minimum e s.t.

$$\begin{cases} \|\boldsymbol{f}(\mathbf{x}+t*\mathbf{f}, \mathbf{u}+t*\mathbf{g}) - \boldsymbol{f}(\mathbf{y}_n, \mathbf{y}_{n-m})\| \le e - \sigma, \text{ for} \\ \forall t \in [0,\tau] \\ \forall \mathbf{x} \in \mathcal{B}_{\mathbf{d}_n}(\mathbf{y}_n) \\ \forall \mathbf{u} \in \mathcal{B}_{\mathbf{d}_{n-m}}(\mathbf{y}_{n-m}) \\ \forall \mathbf{f} \in \mathcal{B}_e(\boldsymbol{f}(\mathbf{y}_n, \mathbf{y}_{n-m})) \\ \forall \mathbf{g} \in \mathcal{B}_{e_{n-m}}(\boldsymbol{f}(\mathbf{y}_{n-m}, \mathbf{y}_{n-2m})); \end{cases} \tag{9}$$

7 $\quad d_{n+1} \leftarrow \mathbf{d}_n + \tau e_n$;
$\quad$ /* updating the lists by appending the extrapolation */
8 $\quad \mathbf{t} \leftarrow [\![\mathbf{t}, t_{n+1}]\!]$; $\mathbf{y} \leftarrow [\![\mathbf{y}, y_{n+1}]\!]$; $\mathbf{d} \leftarrow [\![\mathbf{d}, d_{n+1}]\!]$;
9 $\quad n \leftarrow n+1$;
10 **return** $\langle \mathbf{t},\mathbf{y},\mathbf{d}\rangle$;

from -1 and assume that all the evaluations of $\mathbf{y}$ and $\mathbf{d}$ with a negative index return the element at -1, namely $\mathbf{y}_{<0} = \mathbf{y}_{-1}$ [5], and analogously for $\mathbf{d}$.

At $t_0 = 0$, the corresponding local error is initialized with the radius of the initial set $d_0 = \delta$ (line 1). An offest m is computed in line 2 such that $\mathbf{y}_{n-m}$ locates the delayed approximation at $t_n - r$. In each iteration of the simulation loop, the state is extrapolated in line 6 using the well-known *forward Euler method*, which computes y_{n+1} explicitly from previous points y_n and y_{n-m}. Higher-order *Runge-Kutta methods* [1] could be employed here to obtain more precise approximations. Line 7 derives a local error bound d_{n+1} based on the local error slope e_n satisfying the enclosure property (P2'). The computation of e_n is reduced to a constrained optimization problem (line 6).

Correctness of Simulation. Note that the constrained optimization problem (9) need not have a finite solution, in which case our algorithm fails to provide a useful enclosure. Straightforward continuity arguments do, however, show that for small enough stepsize τ, it will always have a solution, which motivated us to implement stepsize control, as discussed below. When being able to compute useful, i.e., finite error slopes, the simulation delivers a safe enclosure satisfying (P2):

Theorem 2 (Correctness). *Suppose the maximum index of the lists generated by Algorithm 2 is N, then $\forall t \in [0,T]$ and $\forall \mathbf{x} \in \mathcal{B}_\delta(\mathbf{x}_0)$,*

$$\xi_{\mathbf{x}}(t) \subseteq \bigcup_{n=0}^{N-1} \mathit{conv}(\mathcal{B}_{\mathbf{d}_n}(\mathbf{y}_n) \cup \mathcal{B}_{\mathbf{d}_{n+1}}(\mathbf{y}_{n+1})).$$

[5] For a general initial condition $\boldsymbol{g}(t)$, $\mathbf{y}$ is initialized as $\mathbf{y} \leftarrow [\![\boldsymbol{g}(-r), \boldsymbol{g}(-r+\tau), \ldots, \boldsymbol{g}(0)]\!]$.

Algorithm 3: Simulation: a simulation procedure with local stepsize control

input : $\mathcal{B}_\delta(\mathbf{x}_0)$, $\boldsymbol{f}(\mathbf{x},\mathbf{u})$, r, τ_0, T.
output: $\langle \mathbf{t}, \mathbf{y}, \mathbf{d}\rangle$

1 $\mathbf{t} \leftarrow [\![-\tau_0, 0]\!]$; $\mathbf{y} \leftarrow [\![\mathbf{x}_0, \mathbf{x}_0]\!]$; $\mathbf{d} \leftarrow [\![0, \delta]\!]$;
2 $n \leftarrow 0$;
3 **while** $\mathbf{t}_n < T$ **do**
4 $\tau \leftarrow \tau_0$; $m \leftarrow r/\tau$;
  /* relocating the bias m by a backward search */
5 **for** $j \leftarrow$ Length($\mathbf{t}$); $j \geq 1$; $j--$ **do**
6 **if** $\mathbf{t}_n - r \in (\mathbf{t}_{j-1}, \mathbf{t}_j]$ **then**
7 $m \leftarrow n - j$;
8 Break;
9 **while** True **do**
10 $t_{n+1} \leftarrow \mathbf{t}_n + \tau$; $y_{n+1} \leftarrow \mathbf{y}_n + \boldsymbol{f}(\mathbf{y}_n, \mathbf{y}_{n-m}) * \tau$;
11 **if** minimal e satisfying Eq. (10) under the constraints of (9) is found **then**
12 $e_n \leftarrow e$; $d_{n+1} \leftarrow \mathbf{d}_n + \tau e_n$;
13 Break;
14 **else**
15 $\tau \leftarrow \tau/2$;
  /* Smaller e, tighter the bloating. */
16 $\mathbf{t} \leftarrow [\![\mathbf{t}, t_{n+1}]\!]$; $\mathbf{y} \leftarrow [\![\mathbf{y}, y_{n+1}]\!]$; $\mathbf{d} \leftarrow [\![\mathbf{d}, d_{n+1}]\!]$;
17 $n \leftarrow n + 1$;
18 **return** $\langle \mathbf{t}, \mathbf{y}, \mathbf{d}\rangle$;

The completeness result can be formally stated as follows:

Theorem 3 (Completeness). *Suppose the function $\boldsymbol{f}$ in Eq. (5) is continuously differentiable in both arguments and the dynamical system is solvable for time interval $[0, T]$, then for any $\varepsilon > 0$, there exists δ, τ and σ such that the optimization problem (9) has a solution e_n for all $n \leq \frac{T}{\tau}$, and moreover $\mathbf{d}_n \leq \varepsilon$.*

Extension to variable stepsize. Local stepsize control reducing the current stepsize whenever Eq. (9) has no finite solution seems natural. An improved simulation procedure with flexible stepsize control is presented in Algorithm 3, where in each step of simulation, the procedure first tries to find a finite upper bound e satisfying Eq. (9) with an initial stepsize τ_0. If it fails, the current interval is split into two (line 15) and the above operations repeat. Termination of refining the stepsize is guaranteed by the continuous differentiability of $\boldsymbol{f}$ in both of its arguments. Along with variation of τ, the bias locating the delayed state within the list of sampling points need to be recomputed in each step by a backward search (line 8). This may generate extra error, as the nearest sampling point $\mathbf{y}_{n-m}$ may not feature exactly the desired delay. This additional error is accounted for by modifying the first line of the constrained optimization (9) into

$$\|(\mathbf{x} + t_1 * \mathbf{f}, \mathbf{u} + t_2 * \mathbf{g}) - \boldsymbol{f}(\mathbf{y}_n, \mathbf{y}_{n-m})\| \leq e - \sigma \tag{10}$$

for any $t_1, t_2 \in [0, \tau]$. The correctness and completeness arguments for Algorithm 3 are akin to Theorem 2.

5 Implementation and Experimental Results

To evaluate the approach of verification along simulations, we have implemented the proposed algorithms with local stepsize control as a prototype[6] in Matlab. It takes a time-bounded safety verification problem of delayed dynamical systems as input, and it terminates with one of the three results SAFE, UNSAFE, or UNKNOWN, reflecting the fact that a fine enough over-approximation has been found to prove the system safe or unsafe, respectively, or that the maximum permitted density of covering the initial set was insufficient for obtaining a definite answer.

As our algorithm relies on solving the constrained optimization problems (9) or (10), resp., for determining validated bounds, we have tried different solvers for discharging that optimization problem, namely the numerical (and thus devoid of formal guarantees concerning completeness and soundness) procedure `fmincon` provided by Matlab and the optimization-modulo-theory procedure offered by the nonlinear SAT-modulo theory solver `HySAT II`[7] [15]. The constrained optimization problems (9) and (10) involve a universally quantified constraint of the shape

$$\text{find } \min\{e \geq 0 \mid \forall x : \phi(x,e) \implies \psi(x,e)\} \ , \tag{11}$$

which is outside the scope of the above solving procedures, as these handle existential constraints only. We therefore have substituted (11) by the existentially constrained optimization problem

$$\text{find } \max\{e \geq 0 \mid \exists x : \phi(x,e) \wedge \neg\psi(x,e)\} \ . \tag{12}$$

Due to the linear ordering on $\mathbb{R}_{\geq 0}$, problem (12) is guaranteed to yield an upper bound on the solution of (11), which is safe in our context. Both `fmincon` and `HySAT II` proved to be able to efficiently solve (9) and (10) in the formulation (12), with `HySAT II` being able to provide a validated solution due to global search based on a combination of interval constraint propagation with optimization-modulo-theory solving.

`HySAT II` [15] is a sat-modulo-theory (SMT) solver accepting formulas containing arbitrary boolean combinations of theory atoms involving linear, polynomial and transcendental functions. It internally rewrites these formulae into an equi-satisfiable conjunctive normal form by means of a definitional translation introducing auxiliary propositional and numeric variables representing the truth values of sub-formulae and the numeric values of subexpressions, resp., thus generalizing the well-known Tseitin transformation [24]. `HySAT II` then solves the resulting CNF through a tight integration of the Davis-Putnam-Logemann-Loveland (DPLL) algorithm [5] in its conflict-driven clause learning (CDCL) variant with interval constraint propagation (ICP) [3]. Details of the algorithm, which operates on interval valuations for both the Boolean and the numeric variables and alternates between choice steps splitting such intervals and deduction steps narrowing them based on logical deductions computed through ICP or Boolean constraint propagation (BCP), can be found in [13]. Implementing a branch-and-prune search in interval lattices and conflict-driven learning of clauses comprising

[6] Available from `http://lcs.ios.ac.cn/~chenms/tools/DDEChecker_v1.0.tar.bz2`

[7] Available from `https://www.uni-oldenburg.de/en/hysat/`

irreducible atoms in those lattices, it can be classified as an early implementation of abstract conflict-driven clause learning (ACDCL) [4].

By this ACDCL proof search, `HySAT II` will successively construct a cover of the actual solution set of the constraint problem by tiny interval boxes, a sequence of so-called candidate solution boxes together enclosing all solutions. Optimization then is based on a branch-and-prune search over the candidate solution boxes, which is straightforward to integrate into the ACDCL proof search by biasing the ACDCL splitting rule to better values when splitting along the variable representing the optimization criterion, plus learning bounds that impose blocking on any solutions worse than the best value up-to-now found.

The soundness of this procedure for solving the optimization problems (9) or (10) in the formulation (12) follows immediately from the soundness properties of ICP, which narrows the search space by chopping off regions not containing any solution, but will never remove solutions [3]. It consequently is an invariant of the iSAT algorithm's proof search, as implemented in `HySAT II`, that its residual search space internally represented by interval boxes plus the already reported solution boxes together safely over-approximate the actual solution space [13]. This in turn implies that the maximum found by `HySAT II` always is a safe upper bound of the actual maximum, irrespective of possible non-convexity of the optimization problem at hand. We can conclude that solving the optimization problems (9) or (10) in the formulation (12) with `HySAT II` will provide a safe upper bound on the actual optimal value of (12), which in turn is an upper bound on (9) or (10), resp., in the original form (11). As any upper bound renders the enclosure in Algorithm 2 or 3, resp., correct, we can conclude that `HySAT II`'s optimization procedure guarantees soundness of the overall algorithm. The possible failure of `HySAT II`'s optimization procedure in determining a sharp over-approximation of the optimal value will at most impact performance, as it may enforce an unnecessarily dense cover by simulation traces due to overly pessimistic bloating of the original traces.

In the following, we demonstrate our approach by verification of some quintessential DDEs.

Delayed Logistic Equation. In 1948, G. Hutchinson [16] introduced the delayed logistic equation

$$\dot{n}(t) = a[1 - n(t-T)/K]n(t)$$

to model a single population whose percapita rate of growth at time t

$$\dot{n}(t)/n(t) = a[1 - n(t-T)/K]$$

depends on the population size T times units in the past. This would be a reasonable model for a population that features a significant minimum reproductive age or depends on a resource, like food, needing time to grow and those to recover its availability. If we let $N(t) = n(t)/K$ and rescale time, then we get the discrete-delay logistic equation

$$\dot{N}(t) = N(t)[1 - N(t-r)], t \geq 0. \tag{13}$$

Arguments in [23] established that for any initial function $N_0 > 0$, there exists a unique non-negative solution $N(\phi, t)$ defined for all $t > 0$. Wright's conjecture [25], still unsolved, is that if $r \leq \pi/2$ then $N(\phi, t) \to 1$ as $t \to \infty$ for all solutions of Eq. (13) satisfying $N_0 > 0$.

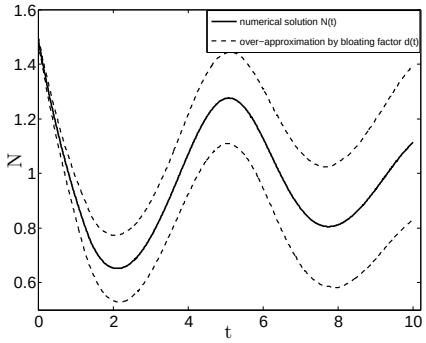


Fig. 1: Over-approximation of the solutions of Eq. (13) originating from region $\mathcal{B}_{0.01}(1.49)$ under delay $r = 1.3$. Initial stepsize $\tau_0 = 0.01$, time bound $T = 10$s.

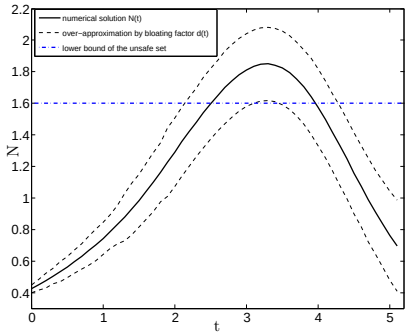
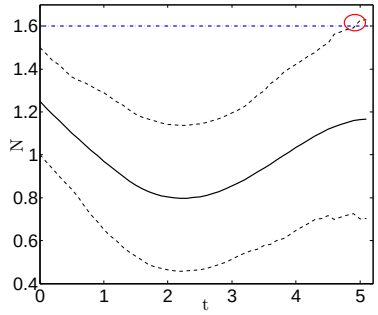
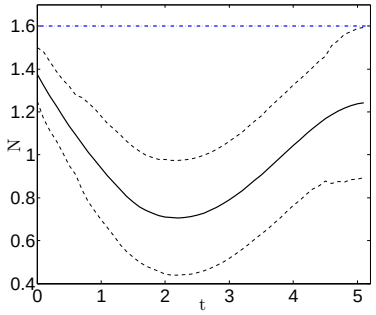


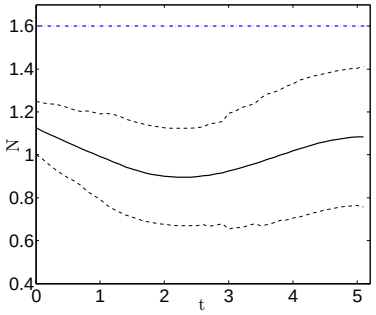
Fig. 2: Over-approximation rigorously proving Eq. (13) unsafe, with $r = 1.7$, $\mathcal{X}_0 = \mathcal{B}_{0.025}(0.425)$, $\tau_0 = 0.1$, $T = 5$s and $\mathcal{U} = \{N | N > 1.6\}$.



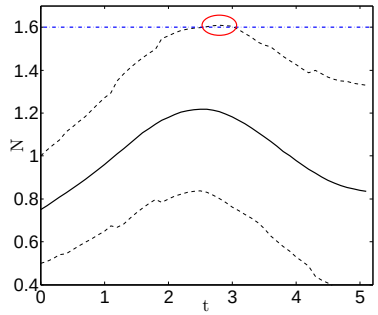
(a) An initial over-approximaion of trajectories starting from $\mathcal{B}_{0.225}(1.25)$. It overlaps with the unsafe set (s. circle). Initial set is consequently split (cf. Figs. 3b, 3c).



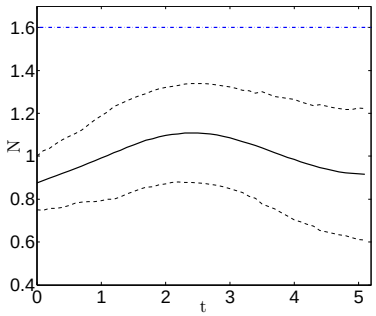
(b) All trajectories starting from $\mathcal{B}_{0.125}(1.375)$ are proven safe within the time bound, as the over-approximation does not intersect with the unsafe set.



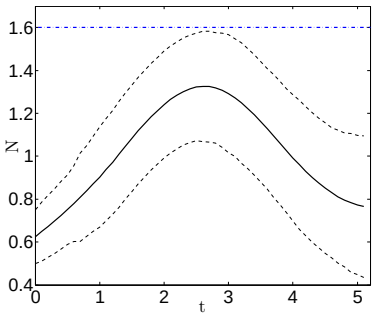
(c) Initial state set $\mathcal{B}_{0.125}(1.125)$ is verified to be safe as well.



(d) $\mathcal{B}_{0.25}(0.75)$ yields overlap w. unsafe; the ball is partitioned again (Figs. 3e, 3f).



(e) All trajectories originating from $\mathcal{B}_{0.125}(0.875)$ are provably safe.



(f) All trajectories originating from $\mathcal{B}_{0.125}(0.625)$ are provably safe as well.

Fig. 3: The logistic system (13) is proven safe through 6 rounds of simulation with base stepsize $\tau_0 = 0.1$. Delay $r = 1.3$, initial state set $\mathcal{X}_0 = \{N | N \in [0.5, 1.5]\}$, time bound $T = 5$s, unsafe set $\{N | N > 1.6\}$.

Fig. 1 illustrates an over-approximation of trajectories of Eq. (13) in terms of a specific initial set. It provides an intuitive description of our simulation approach equipped with computation of on-the-fly linear local error bounds. To investigate Wright's conjecture, we further explore the safety verification framework based upon validated simulations with a delay $r = 1.3 < \pi/2$, for which the trajectories are expected to converge within a time interval. The detailed verification process is elaborated in Fig. 3. Meanwhile, we also successfully falsified an unsafe case with $r = 1.7$ where the over-approximation of a diverging trajectory can be rigorously shown to violate the safety property (see Fig. 2).

Delayed Microbial Growth. Ellermeyer *et al.* [9,10] introduced a delay in the standard bacterial growth model in a chemostat which, after scaling time and the dependent variables, can be written as

$$\begin{aligned}\dot{S}(t) &= 1 - S(t) - f(S(t))x(t)\,,\\ \dot{x}(t) &= \mathrm{e}^{-r} f(S(t-r))x(t-r)\\ &\quad -x(t)\,,\end{aligned} \tag{14}$$

where $f(S) = \alpha S/(\beta + S)$, and $S(t)$ denotes the substrate (food for bacteria) concentration, while $x(t)$ is the biomass concentration of bacteria. The delay r reflects the assumption that whereas cellular absorption of substrate is assumed to be an instantaneous process, a resulting increase in microbial biomass reflecting assimilation is assumed to lag by a fixed amount of time r. A specific verification problem of Eq. (14) is shown in Fig. 4, where different rounds of simulation are depicted together in the phase space of S and x, and for a clear presentation, we only sketch the over-approximations around those numerically computed sampling points.

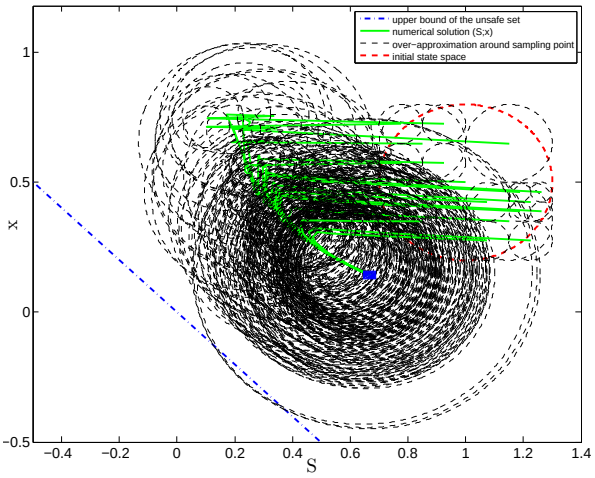


Fig. 4: Eq. (14) is proven safe by 17 rounds of simulation w. $\tau_0 = 0.45$. The simulated trajectories start from within a cover of $\mathcal{X}_0$ (the red dashed circle on the right) and converge eventually to a *basin of attraction* (marked by a small blue rectangle). Here, $\alpha = 2\mathrm{e}$, $\beta = 1$, $r = 0.9$, $\mathcal{X}_0 = \mathcal{B}_{0.3}((1;0.5))$, $\mathcal{U} = \{(S;x)|S + x < 0\}$, $T = 8\mathrm{s}$.

Gene Regulation. To further investigate the scalability of our approach to high dimensions, we recall an instantiation of Example 1 by setting $n = 5$, namely with 5 state components $\mathbf{x} = (x_1; x_2; \ldots; x_5)$ and 5 delay terms $\mathbf{r} = (r_1; r_2; \ldots; r_5)$ involved. This essentially yields, in each step of simulation, an optimization procedure of the form (10) with 23 scalar variables, i.e., e, t_1, t_2 and $\mathbf{x}, \mathbf{u}, \mathbf{f}, \mathbf{g} \in \mathbb{R}^5$. By further setting $\mathbf{r} = (0.1; 0.2; 0.4; 0.8; 1.6)$, $\mathcal{X}_0 = \mathcal{B}_{0.2}((1;1;1;1;1))$, $\mathcal{U} = \{\mathbf{x}|x_1 < 0\}$, and $T = 2\mathrm{s}$, the system of Eq. (4) is rigorously proven unsafe, which means that the dosage of mRNA might degrade to negative in this hypothetical setting.

As an intuitive observation, the verification time consumed by our prototype is fairly sensitive to the specific setting of the verification problem, including the initial set $\mathcal{X}_0$,

the delays $\mathbf{r}$, the unsafe set $\mathcal{U}$, and the time bound T as well. However, the optimization routine proved well scalable to high dimensions, and particularly, verifications of the above benchmark systems all completed successfully in a handful of minutes.

6 Conclusion and Future Work

We have exposed an approach for automated formal verification of time-bounded reachability properties of a class of systems that feature delayed differential dynamics governed by delay differential equations (DDE) with multiple different delays (including 0, i.e., direct feedback). This class of system models has successfully been used to model various real-world systems in the field of biology, control theory, economics, and other domains. Our approach is based on adapting the paradigm of verification-by-simulation to DDEs. It provides bounded-time verification by covering the full set of time-bounded trajectories of a dynamical system evolving from the initial state set by means of investigating a finite sample of initial states plus generalization via a sensitivity argument. Initially, it triggers a finite set of numerically approximate simulations of the dynamic behaviors, thereby generating a finite set of approximate simulation traces originating from a finite sample of the initial states. As the sample does not cover all initial states, and as simulation is only approximate, we bloat each time-stamp value pair returned from the simulation by a distance determined via an error bound computed automatically on-the-fly during simulation. This error bound incorporates both sensitivity information concerning start states and rigorous bounds on integration error incurred by numerical solving. Hence, the union of the state sets reached by all the individual bloated trajectories provides a safe over-approximation of the states actually reachable from the initial set within the time bound. If this over-approximation proves safety in the sense that the reachable states do not intersect the unsafe states, or conversely if the simulation produces a valid counter-example in the sense that it can prove that a trajectory hits the unsafe states, then the algorithm generates the corresponding verdict. Otherwise, our algorithm refines its sample of initial states and repeats the previous steps to compute a more precise over-approximation.

Based on that approach, we have implemented a prototype of a validated solver for DDE. Using it, we have successfully demonstrated the method on several benchmark systems involving delayed differential dynamics.

As a future work, we plan to replace Euler's direct method by high-order Runge-Kutta methods [1] in order to obtain more precise approximations. Furthermore, the method of Zou *et al*. [26] can be extended to provide a safe enclosure algorithm for the class of systems (3) suitable for use in unbounded formal verification, based on the fact that the iSAT constraint solver [13] used therein supports unbounded verification by means of Craig interpolation. In addition, it could be quite interesting to investigate how to combine the technique of conformance testing for hybrid systems [22,17] with our approach. The potential merits of such combination is twofold: on the one hand, it can extend the conformance testing technique to deal with hybrid systems with delays; on the other hand, it may improve the efficiency of the conformance testing technique by using simulation-based approach to over-approximate the reachable set instead of directly computing.

References

1. Alfredo Bellen and Marino Zennaro. *Numerical methods for delay differential equations*. Numerical Mathematics and Scientific Computation. Clarendon Press, Oxford, 2003.
2. Richard Ernest Bellman and Kenneth L. Cooke. Differential-difference equations. Technical Report R-374-PR, RAND Corporation, Santa Monica, California, January 1963.
3. F. Benhamou and L. Granvilliers. Continuous and interval constraints. In F. Rossi, P. van Beek, and T. Walsh, editors, *Handbook of Constraint Programming*, Foundations of Artificial Intelligence, chapter 16, pages 571–603. Elsevier, Amsterdam, 2006.
4. Martin Brain, Vijay D'Silva, Alberto Griggio, Leopold Haller, and Daniel Kroening. Interpolation-based verification of floating-point programs with abstract CDCL. In Francesco Logozzo and Manuel Fähndrich, editors, *Static Analysis - 20th International Symposium, SAS 2013, Seattle, WA, USA, June 20-22, 2013. Proceedings*, volume 7935 of *Lecture Notes in Computer Science*, pages 412–432. Springer, 2013.
5. M. Davis, G. Logemann, and D. Loveland. A Machine Program for Theorem Proving. *Communications of the ACM*, 5:394–397, 1962.
6. P.A.M. Dirac. *The Principles of Quantum Mechanics*. Clarendon Press, 1981.
7. Alexandre Donzé and Oded Maler. Systematic simulation using sensitivity analysis. In *Hybrid Systems: Computation and Control*, pages 174–189. Springer, 2007.
8. Parasara Sridhar Duggirala, Sayan Mitra, and Mahesh Viswanathan. Verification of annotated models from executions. In *Proceedings of the Eleventh ACM International Conference on Embedded Software*, page 26. IEEE Press, 2013.
9. S. F. Ellermeyer. Competition in the chemostat: Global asymptotic behavior of a model with delayed response in growth. *SIAM Journal on Applied Mathematics*, 54(2):456–465, 1994.
10. S. F. Ellermeyer, Jerald Hendrix, and Nariman Ghoochan. A theoretical and empirical investigation of delayed growth response in the continuous culture of bacteria. *Journal of Theoretical Biology*, 222(4):485–494, 2003.
11. Christopher P. Fall, Eric S. Marland, John M. Wagner, and John J. Tyson, editors. *Computational Cell Biology*, volume 20. Springer-Verlag New York, 2002.
12. Chuchu Fan and Sayan Mitra. Bounded verification with on-the-fly discrepancy computation. In Bernd Finkbeiner, Geguang Pu, and Lijun Zhang, editors, *Automated Technology for Verification and Analysis*, volume 9364 of *Lecture Notes in Computer Science*, pages 446–463. Springer International Publishing, 2015.
13. Martin Fränzle, Christian Herde, Stefan Ratschan, Tobias Schubert, and Tino Teige. Efficient solving of large non-linear arithmetic constraint systems with complex Boolean structure. *Journal on Satisfiability, Boolean Modeling and Computation*, 1:209–236, 2007.
14. Antoine Girard and George J. Pappas. Approximate bisimulation: A bridge between computer science and control theory. *European Journal of Control*, 17(5–6):568–578, 2011.
15. Christian Herde. *Efficient Solving of Large Arithmetic Constraint Systems with Complex Boolean Structure*. Vieweg+Teubner, 2011.
16. G Evelyn Hutchinson. Circular causal systems in ecology. *Annals of the New York Academy of Sciences*, 50(4):221–246, 1948.
17. Narges Khakpour and Mohammad Reza Mousavi. Notions of conformance testing for cyber-physical systems: Overview and roadmap (invited paper). In *CONCUR'15*, volume 42 of *LIPIcs*, pages 18–40, 2015.
18. John Mallet-Paret and George R. Sell. The Poincaré-Bendixson theorem for monotone cyclic feedback systems with delay. *Journal of Differential Equations*, 125:441–489, 1996.
19. Tarik Nahhal and Thao Dang. Test coverage for continuous and hybrid systems. In *CAV 2007*, volume 4590 of *Lecture Notes in Computer Science*, pages 449–462. Springer, 2007.

20. Giordano Pola, Pierdomenico Pepe, and Maria Domenica Di Benedetto. Symbolic models for time-varying time-delay systems via alternating approximate bisimulation. *International Journal of Robust and Nonlinear Control*, 25:2328C2347, 2015.
21. Giordano Pola, Pierdomenico Pepe, Maria Domenica Di Benedetto, and Paulo Tabuada. Symbolic models for nonlinear time-delay systems using approximate bisimulations. *Systems & Control Letters*, 59(6):365–373, 2010.
22. Hendrik Roehm, Jens Oehlerking, Matthias Woehrle, and Matthias Althoff. Reachset conformance testing of hybrid automata. In *HSCC '16*, pages 277–286, 2016.
23. Hal Smith. *An Introduction to Delay Differential Equations with Applications to the Life Sciences*, volume 57. Springer-Verlag New York, 2011.
24. Grigori S. Tseitin. On the complexity of derivations in propositional calculus. In A. Slisenko, editor, *Studies in Constructive Mathematics and Mathematical Logics*. 1968.
25. E. M. Wright. A non-linear difference-differential equation. *J. Reine Angew. Math.*, 1955:66–87, 1955.
26. Liang Zou, Martin Fränzle, Naijun Zhan, and Peter N. Mosaad. Automatic verification of stability and safety for delay differential equations. In Daniel Kroening and Corina S. Păsăreanu, editors, *Computer Aided Verification*, volume 9207 of *Lecture Notes in Computer Science*, pages 338–355. Springer International Publishing, 2015.

Appendix: Proofs for Theorems and Lemmas

The following is the proof for Theorem 2.

Proof. We only need to prove that for any $t \in [0, \tau]$,

$$\|\mathbf{x}(t_n + t) - \mathbf{x}(t_n) - t\boldsymbol{f}(\mathbf{y}_n, \mathbf{y}_{n-m})\| \le te_n, \tag{15}$$

as this directly implies the enclosure property

$$\begin{aligned} &\|\mathbf{x}(t_n + t) - \mathbf{y}_n - t\boldsymbol{f}(\mathbf{y}_n, \mathbf{y}_{n-m})\| \\ \le\ &\|\mathbf{x}(t_n) - \mathbf{y}_n\| + \|\mathbf{x}(t_n + t) - \mathbf{x}(t_n) - t\boldsymbol{f}(\mathbf{y}_n, \mathbf{y}_{n-m})\| \le d_n + te_n. \end{aligned}$$

basically equivalent to (P2'). From that, Theorem 2, i.e. (P2), immediately follows from the fact

$$\mathcal{B}_{d_n+te_n}(\mathbf{y}_n + t\boldsymbol{f}(\mathbf{y}_n, \mathbf{y}_{n-m})) = (1 - \frac{t}{\tau})\mathcal{B}_{d_n}(\mathbf{y}_n) \oplus \frac{t}{\tau}\mathcal{B}_{d_n+\tau e_n}(\mathbf{y}_n + \tau\boldsymbol{f}(\mathbf{y}_n, \mathbf{y}_{n-m})),$$

where $\oplus$ stands for Minkowski sum, i.e., $A \oplus B = \{a + b \mid a \in A, b \in B\}$.

In order to prove Eq. (15), assume that there exists some $t \in [0, \tau]$ which violates inequality (15). Let $\mathcal{T}_n = \{t \in [0, \tau] \mid \|\mathbf{x}(t_n + t) - \mathbf{x}(t_n) - t\boldsymbol{f}(\mathbf{y}_n, \mathbf{y}_{n-m})\| \ge te_n\}$. Clearly, $\mathcal{T}_n$ is a nonempty compact set, so we denote by $t_0 = \min \mathcal{T}_n$. Then Eq. (15) holds for $t \in [0, t_0)$, and

$$\|\mathbf{x}(t_n + t_0) - \mathbf{x}(t_n) - t_0\boldsymbol{f}(\mathbf{y}_n, \mathbf{y}_{n-m})\| = t_0e_n \tag{16}$$

due to the continuity of $\mathbf{x}$. Applying the mean value theorem,

$$\mathbf{x}(t_n + t_0) - \mathbf{x}(t_n) = t_0\dot{\mathbf{x}}(t_n + \varsigma) = t_0\boldsymbol{f}(\mathbf{x}(t_n + \varsigma), \mathbf{x}(t_n + \varsigma - r)), \tag{17}$$

for some $\varsigma \in (0, t_0)$. Note that $\mathbf{x}(t_n) \in \mathcal{B}_{\mathbf{d}_n}(\mathbf{y}_n)$, and from Eq. (15),

$$\boldsymbol{f}(\mathbf{x}(t_n + \varsigma), \mathbf{x}(t_n + \varsigma - r)) \in \mathcal{B}_e(\boldsymbol{f}(\mathbf{y}_n, \mathbf{y}_{n-m})).$$

Similarly, we have

$$\mathbf{x}(t_n + \varsigma - r) = \mathbf{u} + \varsigma\mathbf{g}$$

for some $\mathbf{u} \in \mathcal{B}_{\mathbf{d}_{n-m}}(\mathbf{y}_{n-m})$, and $\mathbf{g} \in \mathcal{B}_{e_{n-m}}(\boldsymbol{f}(\mathbf{y}_{n-m}, \mathbf{y}_{n-2m}))$. Then from Eq. (9), we have

$$\|\boldsymbol{f}(\mathbf{x}(t_n + \varsigma), \mathbf{x}(t_n + \varsigma - r)) - \boldsymbol{f}(\mathbf{y}_n, \mathbf{y}_{n-m})\| < e_n, \tag{18}$$

which contradicts to Eq. (16) and Eq. (17). Consequently, Eq. (15) holds. □

In order to prove Theorem 3, we first assume that the function $\boldsymbol{f}$ and its partial derivatives satisfies the boundness property, i.e., there are constants c, c_1, c_2 such that $\|\boldsymbol{f}(\mathbf{x}_1, \mathbf{x}_2)\| \le c$, $\|\frac{\partial \boldsymbol{f}}{\partial \mathbf{x}_1}(\mathbf{x}_1, \mathbf{x}_2)\| \le c_1$, and $\|\frac{\partial \boldsymbol{f}}{\partial \mathbf{x}_2}(\mathbf{x}_1, \mathbf{x}_2)\| \le c_2$, for all possible values $(\mathbf{x}_1, \mathbf{x}_2)$ in our discussion. An obvious consequence of this assumption is that

$$\|\boldsymbol{f}(\mathbf{x}_1, \mathbf{x}_2) - \boldsymbol{f}(\mathbf{y}_1, \mathbf{y}_2)\| \le c_1\|\mathbf{x}_1 - \mathbf{y}_1\| + c_2\|\mathbf{x}_2 - \mathbf{y}_2\|, \tag{19}$$

for all $(\mathbf{x}_1, \mathbf{x}_2)$ and $(\mathbf{y}_1, \mathbf{y}_2)$. With this assumption, the feasibility of optimization problem (9) and controllability of $\mathbf{d}_n$ is from the following lemma, and then the completeness theorem can be proved by making the boundness assumption without loss of generality.

Lemma 1. *Suppose the boundess property holds and $c_1\tau < 1$. Then, the optimization problem (9) has a solution with $e_n \leq \frac{\sigma+c(c_1+c_2)\tau+(c_1+c_2)d_n}{1-c_1\tau}$ and $d_n \leq (\delta + c\tau + \frac{\sigma}{c_1+c_2})\mathrm{e}^{\frac{c_1+c_2}{1-c_1\tau}T}$.*

Proof. We first prove that the constraint of Eq. (9) is satisfied by

$$e = \frac{\sigma + c(c_1+c_2)\tau + (c_1+c_2)d_n}{1-c_1\tau},$$

then the solution e_n of the optimization problem exsits and satisfies $e_n \leq e$. By invoking Eq. (19), the inequality

$$\|\boldsymbol{f}(\mathbf{x}+t*\mathbf{f}, \mathbf{u}+t*\mathbf{g}) - \boldsymbol{f}(\mathbf{y}_n, \mathbf{y}_{n-m})\| \leq e - \sigma$$

can be deduced from

$$e - \sigma \geq c_1\|\mathbf{x}+t*\mathbf{f}-\mathbf{y}_n\| + c_2\|\mathbf{u}+t*\mathbf{g}-\mathbf{y}_{n-m})\|,$$

which can be verified from the following facts:

$$\begin{aligned}
&\|\mathbf{x}+t*\mathbf{f}-\mathbf{y}_n\| \leq \|\mathbf{x}-\mathbf{y}_n\| + t\|\mathbf{f}\| \\
&\leq \mathbf{d}_n + t(\|\mathbf{f}-\boldsymbol{f}(\mathbf{y}_n,\mathbf{y}_{n-m})\| + \|\boldsymbol{f}(\mathbf{y}_n,\mathbf{y}_{n-m})\|) \\
&\leq \mathbf{d}_n + t(e+c) \leq \mathbf{d}_n + c\tau + e\tau, \\
&\|\mathbf{u}+t*\mathbf{g}-\mathbf{y}_{n-m}\| \leq \|\mathbf{u}-\mathbf{y}_{n-m}\| + t\|\mathbf{g}\| \\
&\leq \mathbf{d}_{n-m} + t(\|\mathbf{g}-\boldsymbol{f}(\mathbf{y}_{n-m},\mathbf{y}_{n-2m})\| + \|\boldsymbol{f}(\mathbf{y}_{n-m},\mathbf{y}_{n-2m})\|) \\
&\leq \mathbf{d}_{n-m} + t(e_{n-m}+c) \leq \mathbf{d}_{n-m} + \tau(e_{n-m}+c) \\
&= \mathbf{d}_{n-m+1} + c\tau \leq \mathbf{d}_n + c\tau
\end{aligned}$$

and

$$\begin{aligned}
e - \sigma &= c_1\tau e + c(c_1+c_2)\tau + (c_1+c_2)\mathbf{d}_n \\
&= c_1(\mathbf{d}_n + c\tau + e\tau) + c_2(\mathbf{d}_n + c\tau).
\end{aligned}$$

To show the upper bound of d_n, we note that

$$\begin{aligned}
d_{n+1} &= d_n + \tau e_n \\
&\leq \frac{1+c_2\tau}{1-c_1\tau}d_n + \frac{\sigma\tau + c(c_1+c_2)\tau^2}{1-c_1\tau}
\end{aligned}$$

for all $n \geq 0$. Then it is easily verified that

$$\begin{aligned}
d_n &\leq (\frac{1+c_2\tau}{1-c_1\tau})^n d_0 + \frac{\sigma\tau + c(c_1+c_2)\tau^2}{1-c_1\tau}\frac{(\frac{1+c_2\tau}{1-c_1\tau})^n - 1}{\frac{1+c_2\tau}{1-c_1\tau} - 1} \\
&= (\frac{1+c_2\tau}{1-c_1\tau})^n\delta + (c\tau + \frac{\sigma}{c_1+c_2})((\frac{1+c_2\tau}{1-c_1\tau})^n - 1).
\end{aligned}$$

Note that

$$(\frac{1+c_2\tau}{1-c_1\tau})^n \leq \mathrm{e}^{\frac{c_1+c_2}{1-c_1\tau}n\tau},$$

and $n\tau \leq T$, thus,

$$\begin{aligned} d_n &\leq \delta \mathrm{e}^{\frac{c_1+c_2}{1-c_1\tau}T} + (c\tau + \frac{\sigma}{c_1+c_2})(\mathrm{e}^{\frac{c_1+c_2}{1-c_1\tau}T} - 1) \\ &\leq (\delta + c\tau + \frac{\sigma}{c_1+c_2})\mathrm{e}^{\frac{c_1+c_2}{1-c_1\tau}T}. \end{aligned}$$

□

Now, Theorem 3 can be proved as follows:

Proof. (Sketch) Let $R \triangleq \{\xi_{\mathbf{x}_0}(t) \mid 0 \leq t \leq T\}$ be the set of points in the trajectory between time interval $[0, T]$, and $R_\varepsilon \triangleq \{\mathbf{x} \mid \inf_{\mathbf{y}\in R} \|\mathbf{x} - \mathbf{y}\| \leq \epsilon\}$ its ε-inflation. Obviously, both R and R_ε are compact sets. Since $\boldsymbol{f}$, $\frac{\partial \boldsymbol{f}}{\partial \mathbf{x}_1}$ and $\frac{\partial \boldsymbol{f}}{\partial \mathbf{x}_2}$ are continuous functions, we can put $c = \max_{\mathbf{x}_1,\mathbf{x}_2\in R_\varepsilon} \|\boldsymbol{f}(\mathbf{x}_1, \mathbf{x}_2)\|$, $c_1 = \max_{\mathbf{x}_1,\mathbf{x}_2\in R_\varepsilon} \|\frac{\partial \boldsymbol{f}}{\partial \mathbf{x}_1}(\mathbf{x}_1, \mathbf{x}_2)\|$, and $c_2 = \max_{\mathbf{x}_1,\mathbf{x}_2\in R_\varepsilon} \|\frac{\partial \boldsymbol{f}}{\partial \mathbf{x}_2}(\mathbf{x}_1, \mathbf{x}_2)\|$.

Now we claim that the result holds for δ, σ and τ satisfying $c_1\tau < 1$ and

$$(\delta + c\tau + \frac{\sigma}{c_1+c_2})\mathrm{e}^{\frac{c_1+c_2}{1-c_1\tau}T} \leq \varepsilon.$$

In fact we can prove by induction on n that $\|\mathbf{y}_n - \xi_{\mathbf{x}_0}(n\tau)\| \leq \varepsilon$ during the execution of Algorithm 2, and thus all the involved states would be in R_ε (and the details can be checked by following the proof of Lemma 1). It further implies that the boundness property holds for such c, c_1 and c_2, then the result is straightforward from Lemma 1.

□