Stability Verification of Self-Timed Control Systems using Model-Checking

Viktorio S. el Hakim[§]
Email: v.s.elhakim@utwente.nl
[§]Computer Architectures
for Embedded Systems,
University of Twente,
Enschede, The Netherlands

Marco J. G. Bekooij[§†]
Email: marco.bekooij@nxp.com
[†]Department of Embedded Software
and Signal Processing,
NXP Semiconductors
Eindhoven, The Netherlands

***Abstract*—Cyber-physical control systems are typically time-triggered because the Analog to Digital (A/D) and Digital to Analog (D/A) converters are triggered by a periodic clock. This enables analytical stability analysis of closed-loop models in case the plant and controller are linear. However ensuring periodic sampling requires that a sampling period is selected larger than the Worst-Case Execution Times (WCETs) of the digital control tasks. Unfortunately, low sampling rates must be selected as a result of loose WCET bounds especially if multi-processor hardware is applied with shared data caches and SDRAM memory.**

In this paper we propose a model-checking based stability analysis approach for control systems that sample aperiodically. Our approach is targetting *self-timed* systems, where the A/D and D/A converters are driven by internal events, such as the completion of a task, which also trigger subsequent task executions. During self-timed execution the average sampling period tends to be significantly smaller than possible in time-triggered mode, and as a result the control performance is improved significantly. Self-timed systems also allow the use of a running average workload characterization of the tasks which is tighter than a WCET characterization, and which can greatly improve the accuracy of the stability analysis results.

We show using two self-timed control systems that control performance in terms of stability and settling time improves for self-timed systems when the running average workload characterization is applied. Furthermore, we point at an essential feature that is needed for asymptotic stability analysis which is missing in well known model checkers such as SpaceEx.

I. Introduction

Digital control systems are commonly time-triggered because the A/D and D/A converters are driven by a clock (see Fig. 1a). This practice allows the use of well established analytical stability analysis techniques in case the plant and controller are linear. The approach is suitable for highly deterministic single core computer architectures. However modern control systems are increasingly being implemented on embedded multi-processor systems with shared data caches and shared SDRAM memory.

The use of multi-processor hardware tends to "push" the WCET bound to undesirable limits, even though the occurrence of large execution times may be rare. Since the design of time-triggered systems requires that the sampling period be larger than this WCET, the control behavior may then drastically deteriorate. In the most extreme case the system can even become unstable.

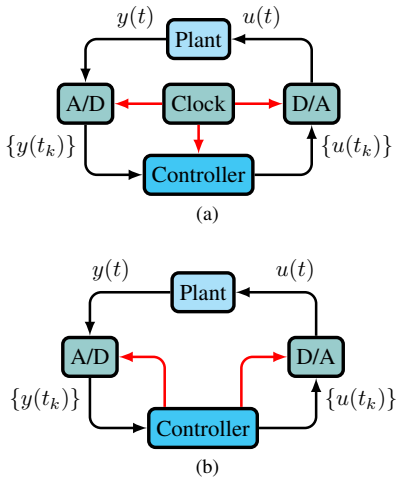


Fig. 1: Block diagrams of basic time-triggered (1a) and self-timed systems (1b).

In so-called *self-timed* systems the A/D and D/A converters are driven aperiodically by internal architectural events, as shown in Fig. 1b. In particular, the control task initiates sampling and actuation (S/A) when it completes an execution. This event also starts a new execution at the same time. As a result the controller enters a free-running mode with higher S/A rates on average compared to its periodic counterpart. However, since S/A is now aperiodic, the derived mathematical models are not always analyzable using classical control theory.

In this paper we propose a model-checking based approach for verifying stability of self-timed systems, using a specific class of hybrid automata models in combination with an average workload characterization [1]. More specifically, we introduce the Hybrid Automata with Clocked Linear Dynamics (HA-CLD) model which allows to jointly describe the temporal and functional behavior of linear plants and controllers under aperiodic S/A.

To demonstrate the applicability of our approach and to show that a self-timed system can perform better than its

978-1-5386-7377-5/18/$31.00 ©2018 IEEE
DOI 10.1109/DSD.2018.00062

time-triggered counterpart, we consider a practical example in our case study. Here various HA-CLD models are derived from the system's physical model and different workload characterizations, including the WCET. We show first that a model-checker, such as *SpaceEx* [2], can conclude stability of the system, but not whether it is asymptotic. Therefore we make use of our own model-checker, which is implemented in MATLAB, for asymptotic stability analysis. We then show that the considered self-timed system has an improved transient response.

The rest of this paper is organized as follows. Section II reviews and compares other state-of-the-art work with ours. Section III describes time-triggered and self-timed systems and the difficulties associated with their design. In Section IV we take a closer look at single-mode closed-loop control. In Section V we describe the modeling framework of our approach. In Section VI we prove that (asymptotic) stability of HA-CLD can be determined using reachability analysis. In Section VII we present the case studies. Finally, we state the conclusions in Section VIII.

II. Related Work

In this section we describe related stability analysis approaches and explain the differences with the approach described in this paper.

Many approaches make use of Lyapunov functions [3]–[6]. Such approaches first derive hybrid automata and/or switched system models of the closed-loop system, by analyzing its discrete-event behavior. A Common Quadratic Lyapunov Function (CQLF) [7] is then computed and used to verify the stability of the system. Average Dwell-Time (ADT) [6] and Multiple Lyapunov Functions (MLF) [4] theory can also be used in a similar way. In contrast our approach relies only on stability verification through reachability analysis. As such it can still be used to conclude stability in cases when a CQLF cannot be computed or does not exist.

Approaches which combine model-checking with Lyapunov functions are proposed in [8], [9]. Such approaches first compute simplified abstractions of the derived hybrid automaton. Reachability analysis is then used on the transformed models to compute critical regions of the system. Finally Lyapunov functions are derived for these regions to conclude local stability. Global stability is concluded if all regions are locally stable. However the key difference is that in our approach we don't consider generic hybrid automata models. Furthermore we don't make use of Lyapunov functions.

A method by Aminifar et al. [10] considers stability verification for networked control systems with variable delay and sampling jitter. Here the authors use a periodic task workload characterization, similar to [1], and the so-called jitter margin [11] curves to compute optimal sampling periods of the controller. However compared to our approach they don't consider systems with aperiodic execution. In contrast our approach can be used to verify the stability of systems with aperiodic S/A, which more accurately describes the behavior of the system.

The approach proposed by Frehse et al. [12] uses the hybrid model checker SpaceEx [2] to verify functional and temporal properties simultaneously of time-triggered systems. They propose the use of a typical worst-case response time [13] to characterize the delay introduced by the execution of the control task(s). The key difference with our work is that we provide an analytical proof that stability can be determined using a model checker for a specific class of self-timed hybrid systems. Furthermore, we consider a different workload/response-time characterization similar to the one introduced in [1].

The works by Khatib et al. [14], [15] propose a reachability analysis based approach for stability verification of a class of self-triggered control systems. The systems under consideration, similarly to our work, have non-deterministic S/A times. Specifically the works propose reachability algorithms to synthesize feasible schedules given a *timing contract* and verify stability. They consider a variant of the WCET characterization of the control task, which they define as the timing contract. However they do not consider systems with multiple modes. Additionally they do not consider workload characterizations other than WCET.

The work by Hausmans et al. [1] presents a two-parameter workload characterization to characterize the maximum total execution time in every window of n subsequent task executions. We make use of a workload characterization that is inspired by characterization of [1]. The work of Hausmans considers only the discrete event part of self-timed systems. An important difference is that in our work the temporal and functional behaviors are analyzed together, by encoding the workload characterization and system dynamics into a hybrid automaton model.

III. Basic Idea

In this section we give a basic overview of time-triggered and self-timed control systems, and discuss their design issues.

A. Time-triggered systems

Time-triggered systems are designed such that the S/A times are dictated by a clock (timer). By S/A times, we refer to the time instances when the A/D and D/A converters are triggered, respectively. For the sake of simplicity it is assumed that these times coincide. Graphically this scheme is shown in Fig. 1a, where $y(t)$ and $u(t)$ are continuous time measurement and actuation signals. The sequences $\{y(t_k)\}_{k\in\mathbb{N}}$ and $\{u(t_k)\}_{k\in\mathbb{N}}$ are their respective discrete-time equivalents.

Most commonly the converters are driven periodically such that $t_{k+1} - t_k = \tau$ for all $k \in \mathbb{N}$, where τ is the sampling period. This is so, because the derived mathematical model of the system has strong analytical properties in case that it is linear and time-invariant. In particular, if the sampling period can be assumed constant throughout the evolution of the system, then one can utilize static analysis techniques to analyze and optimize the controller, such that a desired behavior is achieved. Loosely speaking the closed-loop state-space model of the form $\phi_{k+1} = \Psi\phi_k$ is studied, where Ψ is a fixed transition matrix derived from the plant and controller,

given a certain sampling period, and ϕ_k is the closed-loop state at time t_k. Stability can be verified by showing that the spectral radius $\rho(\Psi)$ is smaller than one [16].

However a requirement for time-triggered systems is that the sampling period must be larger than the WCET of the control task. This WCET tends to be very large in case the processor on which the control task is executed uses a cache (in particular a data-cache), as is the case for most modern general purpose multiprocessor systems. The WCET usually becomes even larger if the data is fetched from shared memory. An important issue is that a large WCET results in a large S/A period, which in turn can result to a system with a poor controlled behavior and can even become unstable.

B. Self-timed systems

In this paper we propose a self-timed system setup, see Fig. 1b, in which the execution of the control task is started by the arrival of internal architectural events. These events are usually generated by the completion of task executions and are also used to trigger the A/D and D/A converters. In this setup S/A occur as soon as possible and aperiodically.

A key advantage of self-timed execution is that the effective sampling period may be significantly smaller than its time-triggered counterpart. This is often the case in practice when large execution times rarely occur. In such cases a so-called running average workload characterization can be used during analysis that is less pessimistic than the WCET of the control task, while still providing an upper bound on the total execution time of a sequence of executions.

However a drawback is that the functional behavior of the system changes dramatically, because the S/A times are non-deterministically selected. In this case the transition matrix becomes a function of the dwell-time $\tau_k = t_{k+1} - t_k$ of iteration k. Additionally workload characterizations other than the WCET introduce switching behavior, such that on each iteration k a new matrix function with mode index σ_k is non-deterministically selected. Formally the derived state-space model takes the form:

$$\phi_{k+1} = \Psi_{\sigma_k}(\tau_k)\phi_k, \quad \Psi_{\sigma_k}(\cdot) \in \mathcal{M}, \tag{1}$$

where $\mathcal{M} = \{\Psi_1(\cdot), ..., \Psi_m(\cdot)\}$ is a finite set of matrix valued functions $\Psi : t \rightarrow \mathbb{R}^{n\times n}$ for a system with m discrete modes. As a result methods such as the CQLF [7] cannot always be applied. The reason is that an infinite amount of matrix product combinations must be evaluated, which should be approximated by a CQLF. Other approaches like the Joint Spectral Radius (JSR) [16] are for the same reason not always applicable. A special case when this is not true, is when the matrices commute or they are triangular.

We show a system example in the case study section which, given a WCET bound $\hat{L}$, may become unstable during aperiodic S/A with each $\tau_k \leq \hat{L}$, despite that it is stable given periodic S/A with $\tau_k = L$ and $L \leq \hat{L}$ is constant $\forall k \in \mathbb{N}$. Fortunately given a running average workload characterization the opposite may often be the case, i.e. the system will be unstable during periodic sampling, but stable during aperiodic sampling.

Since classical Linear Time Invariant (LTI) models and theorems are not sufficient to verify stability, we model linear self-timed systems using a specific subclass of Hybrid Automata (HAs). This representation allows model-checking tools to analyze all of their possible trajectories and check for stability. This is not possible for general HAs, because a contraction towards a region of states cannot be shown for every possible initial state of the HAs using reachability analysis.

We will show later in this paper that the state-trajectories of the considered subclass of HA take the same form as in Eq. (1) and provide an algebraic proof that stability can be verified using model-checking methods for any initial state ϕ_0. In particular we will show that a self-timed system can be verified using a model checker given a specific work-load characterization, and that a finite number of iterations of the model-checker are needed to conclude stability. To the best of our knowledge, our approach is the first that can address this practically relevant stability analysis problem.

C. Drawbacks of the WCET task characterization

In order to understand the drawback of the WCET characterization more precisely, consider a single iterative control task with execution time τ_k per iteration, where k is an iteration index. Given that an upper and a lower bound, $\hat{L}$ and $\check{L}$ respectively, is derived, each execution must satisfy the relation:

$$\forall k \in \mathbb{N} : \check{L} \leq \tau_k \leq \hat{L}. \tag{2}$$

This characterization leads to a model with a single mode of operation. In both the time-triggered and self-timed cases, this model is unstable, if $\hat{L}$ becomes too large.

However in a practical application the maximum execution time case does not typically occur often. In this case a different characterization can be used, similar to [1], such that the upper bound is more relaxed. In the context of Eq. (2) this characterization also requires that:

$$\forall i = 1, ..., n-1 : \tau_{k+i} \leq \bar{L} \text{ if } \tau_k > \bar{L}, \tag{3}$$

with $\bar{L} = \frac{\hat{L}}{n}$.

Informally this characterization states that for every n consecutive executions at least one large execution $\bar{L} < \tau_k \leq \hat{L}$ is allowed. For example in the case that $n = 2$, it is stated that a long execution time is always followed by a short one, i.e., $\tau_{k+1} \leq \bar{L}$.

IV. Single mode closed-loop control

In this section we describe classical LTI models of the plant and controller. We consider the feedback interconnection between the controller and plant as shown in Figure 1.

A. Plant and controller modeling

Classical control systems theory relies on LTI models. Of particular importance is the Input-State-Output (I/S/O) representation, which describes the dynamical behavior of the

plant as a system of first order linear differential equations according to

$$\dot{x}(t) = Ax(t) + Bu(t), \quad (4)$$
$$y(t) = Cx(t) + Du(t),$$

where $x(t) \in \mathbb{R}^n$ is the state of the plant, $u(t) \in \mathbb{R}^m$ is the input and $y(t) \in \mathbb{R}^p$ is the output. Given an initial state $x(0)$, the state at time t is determined by

$$x(t) = \Phi(t)x(0) + \xi(t), \quad (5)$$

where $\Phi(t) = e^{At}$ is the transition matrix and $\xi(t) = \int_0^t \Phi(\tau)Bu(\tau)d\tau$ is the plant's input response at rest, i.e. $x(t) = 0, \forall t \leq 0$. Of particular importance is the step input response, where $u(\tau) = \mu$ is held constant for a time period $\tau \in [0, t)$. In this case $\xi(t) = \Gamma(t)u(0)$, where $\Gamma(t) = A^{-1}(\Phi(t) - I_n)B$ and $I_n \in \mathbb{R}^{n\times n}$ is the identity matrix assuming that A is non-singular.

Similarly to the plant, a *discrete* I/S/O representation in the form of first-order difference equations is derived for the controller

$$z(k+1) = Gz(k) + Fy(k), \quad (6)$$
$$u(k) = Hz(k) + Py(k),$$

where $z(k) \triangleq z(t_k) \in \mathbb{R}^q$ is the state of the controller, while $y(k) \triangleq y(t_k)$ and $u(k) \triangleq u(t_k)$ are its input and output respectively (connected to the plant). Note that $y(t)$ and $u(t)$ are held constant for $t_k \leq t < t_{k+1}$ and that $k \in \mathbb{N}$ is an iteration index. The control law described above is then implemented as a set of instructions on a computer and executed iteratively for each new sampled measurement.

B. Periodic time-triggered mode

The digital controller is implemented as a single task executed periodically, that applies the control law described earlier to the sampled measurements, $y(t_k)$, and computes appropriate actuation values, $u(t_k)$, to steer the plant. In addition the time difference $\tau_k = t_{k+1} - t_k = L$ between sampling/actuation times is assumed constant for every $k \in \mathbb{N}$. A discrete time model of the plant is then derived using Eq. (4) such that

$$x(k+1) = \Phi(L)x(k) + \Gamma(L)u(k), \quad (7)$$
$$y(k) = Cx(k) + Du(k). \quad (8)$$

This representation is then used to find an optimal value for the gain matrix F in (6).

Since the A/D and D/A converters are triggered at equidistant times, t_k, it is required to know the WCET of the control task, to ensure that it completes execution before the next sampling moment. A timer can be used to synchronize the task to the sampling period, once this upper bound is known.

C. Aperiodic self-timed mode

During aperiodic S/A the controller task does not need to wait for a periodic trigger from a timer. Instead it may do so as soon as the current iteration finishes execution.

However since the time difference, τ_k, between the sampling/actuation times becomes non-deterministic, the transition matrices Φ and Γ now change on every iteration with respect to the task execution time. The classical LTI representation is thus not practical for statically analyzing and tuning the closed-loop system, since the discrete-time behavior of the plant is no-longer deterministic.

V. Modeling Self-timed Systems

In this section we describe our modeling framework for self-timed systems using hybrid automata theory.

In particular we derive HA models of the single mode self-timed controller using the WCET and *running average* characterizations. However we point out that the same framework can be used to model any type of multi-mode controller, provided that the temporal behavior of each mode can be characterized. Hybrid automata are a very suitable model for self-timed systems because of their property to include non-determinism, along with multiple mode dependent discrete-event and continuous-time dynamics.

A. Hybrid Automata

The general hybrid automaton is a directed graph system model which is used to describe the evolution of a set of discrete and continuous state variables. The following definition is given in [17]:

Definition 1. *A hybrid automaton H is a tuple $H = (Q, X, f, Init, Dom, E, G, R)$ where*

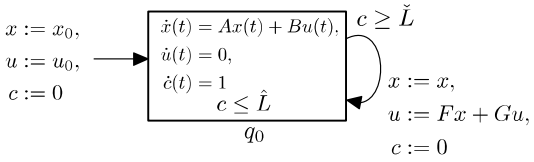
- *$Q = \{q_1, ..., q_N\}$ is a finite set of discrete states (modes);*
- *$X = \mathbb{R}^n$ is the set of continuous states;*
- *$f : \mathbb{R} \times Q \times X \to X$ is a vector field;*
- *$Init \subseteq Q \times X$ is the set of initial states;*
- *$Dom : Q \to 2^X$ is an invariant assignment map;*
- *$E \subseteq Q \times Q$ is a set of edges;*
- *$G : E \to 2^X$ is a transition guard assignment map;*
- *$R : E \times X \to 2^X$ is a reset map.*

Note that 2^X denotes the power set of X. At any moment in time the pair $s = (q, x) \in Q \times X$ describes the total state of H. The continuous state evolves over time according to $\dot{x}(t) = f(t, x, q), q \in Q$ as long as $x \in Dom(q)$. During this time the discrete state q stays constant. A transition from one discrete state q to another q' may occur, whenever $x(t) \in G((q, q')), (q, q') \in E$.

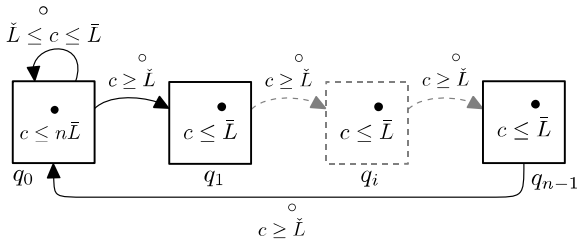
The automaton is expressive enough to model any event-driven and/or time-driven system. However computing the reachable state-space is not always a decidable problem. Some restricted definitions yield more tractable models, such as Timed-Automata, Rectangular Automata, etc, for which the reachability problem is known to be decidable [18].

B. Modeling using automata

In this paper we focus on the analysis of linear self-timed LTI control systems. In this case we use equations similar to Eq. (4) and (6) to describe the dynamic behavior of the plant and controller respectively. On the other hand *clock* variables, $c(t) \in \mathbb{R}^p$ are used to describe the temporal behavior. The



(a) WCET characterization model



(b) Running-average characterization model

Fig. 2: Hybrid automata of a single mode closed-loop controller according to WCET and running average characterizations.

guards and invariants are interval hulls which only constrain the set of clocks, similarly to Timed Automaton (TA). This way the discrete-event (temporal) behavior is specified using real-time task characterizations, independently of the physical and functional behavior of the plant and the controller.

An example of such a hybrid automaton model describing a single mode controller characterized by its WCET, $\hat{L}$, and a Best-Case Execution Time (BCET), $\check{L}$, is illustrated in Figure 2a. Here the continuous time behavior is specified in a single discrete state (mode) q_0. The discrete event control law applied by the controller is specified using the reset map with a single edge (q_0, q_0). Note that here it is assumed that the state $x(t)$ is observable and controllable.

On the other hand the running average characterization as defined in Eq. (3) is shown in Figure 2b. The reset maps and continuous-time dynamics are the same as the automaton in Figure 2a for all edges and locations, and for clarity are replaced with $\circ$ and $\bullet$ respectively. The model uses $n > 1$ modes to describe a running-average behavior, such that each mode $q_i, 0 < i < n$, has a single outgoing edge (q_i, q_{i+1}) with a guard $c \geq \check{L}$ and invariant $c \leq \bar{L}$. A large value of n indicates that one execution with a large peak in execution time of at most $n\bar{L}$ duration can occur, and is followed by n executions that have an execution time of at most $\bar{L}$ duration. As a consequence, the average execution time in every window of n consecutive executions is at most $\frac{(2n-1)\bar{L}}{n}$.

C. Hybrid Automata with Clocked Linear Dynamics

The automaton models described earlier share some common features: 1) the state-space is partitioned into clocks, continuous time and piece-wise constant variables, 2) the reset map and flow equations are linear relations, 3) the guards and invariants are restricted to the clocks only. We identify the class of all hybrid automata which share these features as HA-CLD and define it as follows:

Definition 2. *A Hybrid Automata with Clocked Linear Dynamics is a tuple* $\mathcal{T} = (Q, \mathcal{X}, \mathcal{Z}, \mathcal{C}, f, Init, Dom, E, G, R, R_c)$ *where*

- $Q = \{q_1, ..., q_N\}$ *is a finite set of discrete states (locations);*
- $\mathcal{X} = \mathbb{R}^n$ *is the set of continuous state variables;*
- $\mathcal{Z} = \mathbb{R}^m$ *is the set of piecewise constant variables;*
- $\mathcal{C} = \mathbb{R}^p$ *is the set of clocks;*
- $f : \mathbb{R} \times Q \times \mathcal{X} \times Z \to \mathcal{X}$ *is a vector field;*
- $Init \subseteq Q \times \mathcal{X} \times \mathcal{Z} \times \mathcal{C}$ *is the set of initial states;*
- $Dom : Q \to 2^{\mathcal{C}}$ *is a clock invariant assignment map;*
- $E \subseteq Q \times Q$ *is a set of edges;*
- $G : E \to 2^{\mathcal{C}}$ *is a transition guard assignment map;*
- $R : E \times \mathcal{X} \times \mathcal{Z} \to 2^{\mathcal{X} \times \mathcal{Z}}$ *is a state reset map;*
- $R_c : E \times \mathcal{C} \to 2^{\mathcal{C}}$ *is a clock reset map.*

Here the continuous state $x \in \mathcal{X}$ flows according to $\dot{x}(t) = f(t, q, x, z) = Ax(t) + Bz(t)$. Note that this is different, but similar to equation (4). $z \in \mathcal{Z}$ is a piece-wise constant state, such that $\dot{z}(t) = 0$ while in any $q \in Q$. The state pair $(x, z) = \phi \in \mathcal{X} \times \mathcal{Z}$ is used to model the interaction between the physical environment and the computer architecture. The clocks evolve according to $\dot{c}_i(t) = 1, i \in [1, p]$. For a mode q the clock invariant is an (possibly infinite) interval hull of the form $Dom(q) = [\underline{c}_1^q, \bar{c}_1^q] \times ... \times [\underline{c}_p^q, \bar{c}_p^q]$. Similarly for an edge $e \in E$ the transition guard is an interval hull of the form $G(e) = [\underline{c}_1^e, \bar{c}_1^e] \times ... \times [\underline{c}_p^e, \bar{c}_p^e]$. The reset map assigns a new value to the continuous variable pair ϕ at an edge e according to $\phi' := R(e, \phi) = C_e\phi$, where ϕ' is the new value after a transition takes place and $C_e \in \mathbb{R}^{(n+m)\times(n+m)}$ is a transition matrix.

VI. Semantics and analysis of HA-CLD

In this section we formalize the semantics of HA-CLD models and their use for stability verification of self-timed systems.

A. Evolution of the state

Let $\mathcal{T}$ be a HA-CLD as defined in Definition 2 with a total state $s = (q, x, z, c) \in Q \times \mathcal{X} \times \mathcal{Z} \times \mathcal{C}$. We use this notation throughout the rest of the paper. A discrete transition relation from a state s is then defined as

$$\rho_D(s) = \{s' \mid \forall e \in E : \phi' = C_e\phi \text{ and } c' \in R_c(e, c) \text{ such that } c \in G(e)\}. \quad (9)$$

where $\phi = (x, z)$ and $e = (q, q')$.

By solving the differential equations, a continuous transition relation is derived as

$$\rho_C(s) = \{s' \mid \exists \tau \geq 0 : c' = c + \mathbf{1}\tau \in Dom(q), x' = \Phi(\tau)x + \Gamma(\tau)z, (q', z') = (q, z)\}. \quad (10)$$

where $\Phi(\tau)$ and $\Gamma(\tau)$ are transition matrices as defined in (5) and $\mathbf{1} = (1 \ ... \ 1)^\intercal \in \mathbb{R}^p$.

To reason about the behavior of the automaton however, it is more useful to consider the sets of reachable states. Define the *successor* sets

$$\begin{aligned} S_0 &= Init \\ S_{k+1} &= \{s \in \rho_D(s') \mid \forall s' \in \rho_C(S_k)\}, \end{aligned} \tag{11}$$

then

$$\mathcal{R}_k = \bigcup_{j=0}^{k} S_j = S_k \cup \mathcal{R}_{k-1} \tag{12}$$

is the set of reachable states after k discrete transitions.

Using Eq. (11) we give the following definition for a trajectory of a HA-CLD:

Definition 3. *A trajectory (or trace) of the hybrid automaton $\mathcal{T}$ is a sequence $\{s_j\}_{j=0}^{k}$, such that $s_j \in S_j$, where s_j is the state at time t_j during which a discrete transition takes place.*

B. Temporal behavior

The temporal behavior in a HA-CLD is explicitly modeled by the set of clocks $\mathcal{C}$ and the associated guard and invariant constraints. However to simplify the analysis of the functional behavior, it is better to reason with *dwell* times. Specifically with dwell times we refer to the time intervals between two discrete transitions in a trajectory. Formally, given a trajectory $\{s_k = (q_k,\ x_k,\ z_k,\ c_k)\}_{k=0}^{N}$, a dwell time $\tau_k = t_{k+1} - t_k$, where t_k is a time when a k-th discrete transition takes place.

Furthermore we denote with $\check{\tau}(q,q') = \inf_\tau\{\tau \in \mathbb{R} \mid c + \tau\mathbf{1} \in G(q,q') \cap Dom(q)\}$ the lower bound on the dwell time for an edge (q,q'). Similarly with $\hat{\tau}(q) = \sup_\tau\{\tau \in \mathbb{R} \mid c + \tau\mathbf{1} \in Dom(q)\}$ we denote the upper bound. Finally we denote with $\mathfrak{T}(q,q') = [\check{\tau}(q,q'), \hat{\tau}(q)]$ the closed dwell time interval for an edge (q,q').

C. Functional behavior

Of particular interest are the sub-trajectories of the state pair $(x,z) \in \mathcal{X} \times \mathcal{Z}$ because they describe the functional behavior of the system. Let $\phi_k = (x_k, z_k) \in \mathbb{R}^{n+m}$ be the combined continuous-time and piece-wise constant state vector pair of the system at time t_k, $k \in \mathbb{N}$. We consider only the time of initialization of the automaton, t_0, and the times when a discrete transition takes place. Then ϕ_{k+1} is the value of the state after continuously evolving up-to, but not including time t_{k+1} from its initial value ϕ_k, and subsequently applying the reset map upon transitioning to a discrete state q_{k+1} at time t_{k+1}. Formally, from the transition relations defined in (9) and (10) we derive without loss of generality the recurrence relation

$$\begin{aligned} \phi_0 &\in Init \\ \phi_{k+1} &= \Psi(e,\tau)\phi_k, \quad k \in \mathbb{N} \end{aligned} \tag{13}$$

where

$$\Psi(e,\tau) = C_e \begin{pmatrix} \Phi(\tau) & \Gamma(\tau) \\ 0 & I_m \end{pmatrix}, \tag{14}$$

for some edge $e \in E$ and dwell time $\tau \in \mathfrak{T}(e)$.

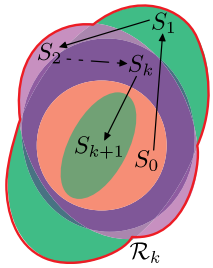


Fig. 3: Visual depiction of the reachable and successor sets. Here $\mathcal{R}_{k+1} = \mathcal{R}_k$ because $S_{k+1} \subset \mathcal{R}_k$.

From this relation a trajectory of the state ϕ up-to time t_k can be seen as a left-wise product of $k-1$ matrices, such that

$$\phi_k = \Psi(e_{k-1}, \tau_{k-1}) \cdots \Psi(e_0, \tau_0)\phi_0, \tag{15}$$

where $e_i \in E$, $\tau_i \in \mathfrak{T}_{e_i}, i \leq k$ and $\phi_0 \in Init$. Using this we give the following definition of stability:

Definition 4. *Let the sequence $\{s_j\}_{j=0}^{k}$ be any valid trajectory of the automaton $\mathcal{T}$ for some k as defined in Definition 3, and let $\{\phi_j\}_{j=0}^{k}$ be its corresponding sub-trajectory that satisfies Eq.* (13)*. We say that the automaton is stable if for all $k \in \mathbb{N}$ there exists a bounded constant $M \in \mathbb{R}$, such that $\|\phi_k\| \leq M$, for any $\phi_0 \in Init$. It is asymptotically stable if $\|\phi_k\| \to 0$ as $k \to \infty$. Here $\|\cdot\|$ is any vector norm.*

The automaton is unstable if it is not stable.

D. Stability verification

The process of verifying the stability of a self-timed system through model checking involves computing exact or tight over-approximations of the reachable and successor sets as defined in Eqs. (11) and (12) respectively, until a fixed-point is found. This condition is met when the set of reachable states, $\mathcal{R}_k$, stops expanding for some k, formally $\mathcal{R}_k = \mathcal{R}_{k+1}$. Additionally to verify asymptotic stability, it must hold that all states in S_k are inside S_0, where S_0 is a unit ball centered at the origin. Formally $\forall\phi \in S_k : \|\phi\| < 1$, given any vector norm $\|\cdot\|$. Visually this is shown in Figure 3.

For general HAs this is not possible. Fortunately for the HA-CLD model we can capitalize on the following properties to show global (asymptotic) stability using model-checking:

1) The temporal behavior is independent of the functional behavior. Removing the functional state variables yields a Linear Hybrid Automaton (LHA), for which the reachability problem is decidable [18].
2) The state recurrence relations are linear as observed from Eq. (13). Specifically the state after $k > 0$ discrete transitions is computed by a series of matrix multiplications from the initial state.

We formalize these notions with the following theorem:

Theorem 1. *Let $\mathcal{T}$ be an HA-CLD with corresponding successor and reachable sets of states, S_k and $\mathcal{R}_k$, defined according to Eq.* (11) *and* (12) *respectively. Furthermore*

let $S_0 = \mathcal{S}_0 = Init$ be an initial set of states, such that $\|\phi\|_\infty \leq 1, \forall \phi \in S_0$.

1) *The automaton is stable for any initial state ϕ_0 if and only if $\mathcal{R}_{k+1} = \mathcal{R}_k$ for some $k > 0$.*
2) *It is asymptotically stable if and only if $\forall \phi \in S_k$ holds that $\|\phi\|_\infty < 1$ and $S_k \subset S_0$.*

Proof:

1) $\Rightarrow$ Define the norm of a set S with respect to ϕ as $\|S\|_\phi = \max\{\|\phi\|_\infty \mid \phi \in S\}$. Assuming that the automaton is *stable*, then this implies that there exists a constant $M > 0$, such that $\forall k \in \mathbb{N}$ holds that $\|\mathcal{R}_k\|_\phi \leq M$. This implies that there exists a k, such that $\mathcal{R}_{k+1} = \mathcal{R}_k$.
   $\Leftarrow$ Assuming that $\mathcal{R}_{k+1} = \mathcal{R}_k$ for some k, then $\forall \phi \in \mathcal{R}_k : \exists \phi'$ such that $\phi' = \Psi(e_k, \tau_k)\phi$ and $\phi' \in \mathcal{R}_k$ for an edge e_k and dwell time $\tau_k \in \mathfrak{T}(e_k)$. Informally this states that any state ϕ in $\mathcal{R}_k$ is mapped back into $\mathcal{R}_k$. This in turn implies that $\forall j > k : \mathcal{R}_j = \mathcal{R}_k$.
   Thus $\forall k$ there exists a constant $M > 0$ such that $\|\mathcal{R}_k\|_\phi < M$. Since $\mathcal{R}_k$ per definition contains all the possible trajectories up-to and including k, we conclude that the automaton is *stable*.
2) $\Rightarrow$ Assuming that the automaton is asymptotically stable then $\forall \phi \in S_k : \|\phi\|_\infty \to 0$ as $k \to \infty$. By definition this implies that there exists a k such that $\|S_k\|_\phi < 1$ and $S_k \subset S_0$.
   $\Leftarrow$ Let $\|S_k\|_\phi < 1$ for some k, then $\forall \phi_k \in S_k$ there exists $\phi_0 \in S_0$ and a sequence $\{e_j\}_{j=0}^{k-1}$, $e_j = (q_j, q_{j+1}) \in E$, with associated sequence of dwell times $\{\tau_j\}_{j=0}^{k-1}$, $\tau_j \in \mathfrak{T}(e_j)$ such that

$$\phi_k = \Psi(e_{k-1}, \tau_{k-1}) \cdots \Psi(e_0, \tau_0)\phi_0 = \hat{\Psi}\phi_0$$

   and $\|\phi_k\|_\infty < \|\phi_0\|_\infty$. By definition of S_0 this implies that $\|\hat{\Psi}\|_\infty < 1$, where $\|\hat{\Psi}\|_\infty = \sup_{\|\phi\|_\infty = 1}\{\|\hat{\Psi}\phi\|_\infty\}$ is the induced infinity matrix norm. This implies that for any following transition $e_k = (q_k, q_{k+1})$ and dwell time $\tau_k \in \mathfrak{T}(e_k)$ holds that

$$\|\Psi(e_k, \tau_k)\hat{\Psi}\phi_0\|_\infty = M\|\Psi(e_k, \tau_k)\phi_0\|_\infty, \quad M < 1.$$

   Given that $\mathcal{R}_k \in \mathcal{R}_{k-1}$, then this implies that there exists an identical sequence of edges $\{e_j\}_{j=k}^{2k-1}$ such that

$$\Psi(e_{2k-1}, \tau_{2k-1}) \cdots \Psi(e_k, \tau_k) = \hat{\Psi},$$

   thus $\|\phi_{2k}\|_\infty = \|\hat{\Psi}^2\phi_0\|_\infty = M^2\|\phi_0\|_\infty$. By repeating this process j times then $\|\phi_{jk}\|_\infty = M^j\|\phi_0\|_\infty \to 0$ as $j \to \infty$. Thus the automaton is *asymptotically* stable.

To show that it is sufficient to check stability for any initial state $\phi_0 \in \mathbb{R}^{m+n}$ given S_0 as defined in Theorem 1, consider $\psi_0 = a\phi_0$ for some non-zero constant $a \in \mathbb{R}$ such that $\|\psi_0\|_\infty \leq 1$. This implies that after k discrete transitions

$$\psi_k = \Psi_k\psi_0 = a\Psi_k\phi_0 = a\phi_k.$$

Thus ϕ_k converges from ϕ_0 if and only if ψ_k converges from ψ_0. The same holds if ψ_k diverges.

Theorem 1 shows that if a HA-CLD has a stable fixed point, then during the model-checking process the condition $\mathcal{R}_{k+1} = \mathcal{R}_k$ is eventually met and the tool terminates. Conversely when a model-checker terminates execution, then this implies that there exists a stable fixed-point. Furthermore the fixed point is at the origin. However verifying asymptotic stability requires an additional finite amount of iterations.

VII. Case study

In this section we consider a closed-loop static gain control self-timed system. Given a WCET and a running average task workload characterization we derive HA-CLD models to verify the stability of the system and analyze the performance. We use MATLAB to verify asymptotic stability and to compute state envelopes to benchmark the performance. The SpaceEx [2] model-checker is also used to confirm the results.

A. Setup

The plant under consideration has a state $x \in \mathbb{R}^2$ and the following system matrices

$$A = \begin{pmatrix} -1 & 0.1 \\ -0.02 & -2 \end{pmatrix}, \quad B = \begin{pmatrix} 0 \\ 2 \end{pmatrix},$$

which are derived from a brushed DC motor model. The motor is controlled by a proportional feedback regulator with a state $u \in \mathbb{R}$, and update matrices $F = \begin{pmatrix} -K & 0 \end{pmatrix}$ and $G = 0$.

The total state of the closed-loop system is then $\phi = (x_1, x_2, u) \in \mathbb{R}^3$. The initial set of states is $\phi_{Init} = \{(x_1, x_2, u)^\intercal \mid x_1 \in [-1, 1] \wedge x_2 \in [-1, 1] \wedge u \in [-1, 1]\}$.

Given a WCET characterization, we consider the HA-CLD model shown in Figure 2a. Here $\hat{L} = 2.2$ and $\check{L} = 0.2$, such that the system is unstable. For the running average characterization the model shown in Figure 2b is used.

The SpaceEx tool uses the STC algorithm [19] with an absolute error of 0.001 and octagon template polyhedra. The time horizon is set to $5s$. An example 5-mode automaton is shown in Figure 4.

B. Results

The results from our MATLAB-based model-checker are shown in Figure 5, where the absolute maximum and minimum value of the state per iteration k is computed. The resulting graph forms the so-called envelopes, which encapsulate all possible state-trajectories of the plant. The state of the controller is not shown for clarity.

The 1-mode envelope (red) shows that the system is unstable given a WCET characterization. A running average characterization however is shown to improve the performance, as seen from the rest of the envelopes. Although the system is still unstable given a 2-mode approximation, refining the model by including more modes stabilizes the system and decreases the maximum overshoot.

The SpaceEx model checker also confirmed some of these results. For the 1-mode automaton the tool fails to find a fixed-point after 50 iterations. Repeating the same procedure for the running average characterization with 2, 3 and 4 modes results in a similar situation. The 5 mode model from Figure 4 with

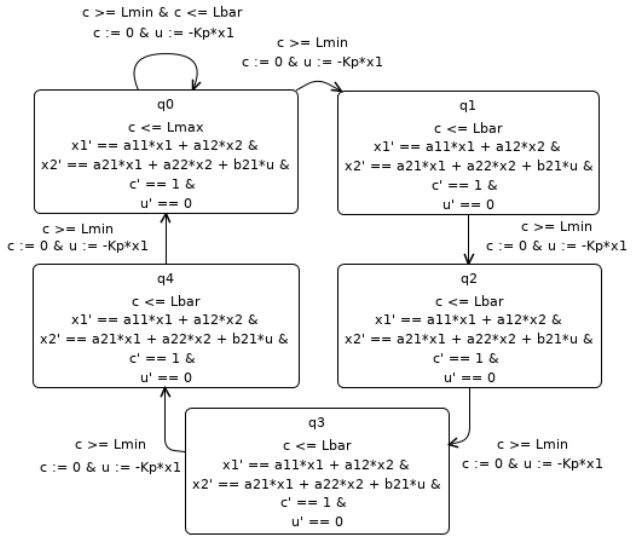
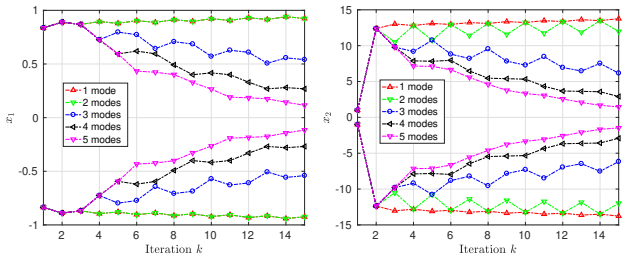


Fig. 4: SpaceEx automaton model of a 5-mode self-timed system.

$\bar{L} = 0.2$ and $\hat{L} = 0.44$ however converges after 38 iterations. We expect that the poor convergence behavior of SpaceEx is a result of that it is not optimized for handling of HA-CLD, which causes a larger over-approximation of the reachable state sets.



(a) x_1 (b) x_2

Fig. 5: State-envelopes of the plant.

VIII. Conclusion

In this paper we proposed a model-checking based approach for stability analysis of systems with aperiodic sampling and actuation. The approach is particularly suitable for analyzing self-timed systems, which typically have a much smaller average sampling and actuation period compared to their time-triggered counter-parts. As a result such systems have an improved control performance. Additionally they allow tighter task execution time characterizations to describe their discrete behavior which lead to more accurate stability analysis results.

To show that model-checking tools can be used to verify stability of aperiodic systems we first introduce the HA-CLD model. This model allows describing aperiodic systems with linear discrete-event and linear continuous-time dynamics, subjected to temporal constraints derived from the execution time characterization. We then prove analytically that a model checker can conclusively verify (asymptotic) stability of HA-CLD models.

Finally we demonstrate the applicability of our model checking approach with a practical closed-loop system. Specifically we use SpaceEx to show stability of the system in self-timed mode, of which the execution times are characterized by its WCET and a running average. Asymptotic stability is shown using our own MATLAB based model-checker, because existing model checkers do not provide the required state-space exploration stop criterion. Additionally we show with MATLAB that the control performance of the system with tighter task workload characterizations is improved in terms of maximum overshoot.

As future work we would like to extend our modeling framework of self-timed systems, such that more general workload characterizations of tasks can be handled.

References

[1] J. P. Hausmans and other, "Two parameter workload characterization for improved dataflow analysis accuracy," in *Proc. IEEE Real-Time and Embedded Technology and Applications Symposium (RTAS)*, 2013, pp. 117–126.

[2] G. Frehse *et al.*, "SpaceEx: Scalable verification of hybrid systems," in *Proc. Int'l Conf. on Computer Aided Verification (CAV)*, 2011.

[3] P. Kumar *et al.*, "A hybrid approach to cyber-physical systems verification," in *Proc. Design Automation Conference (DAC)*, 2012, pp. 688–696.

[4] J. Lu and L. J. Brown, "A multiple Lyapunov functions approach for stability of switched systems," in *American Control Conference (ACC), 2010*, 2010, pp. 3253–3256.

[5] R. Shorten *et al.*, "Stability criteria for switched and hybrid systems," *SIAM review*, vol. 49, no. 4, pp. 545–592, 2007.

[6] G. Zhai *et al.*, "Qualitative analysis of discrete-time switched systems," in *Proc. American Control Conference*, vol. 3, 2002, pp. 1880–1885.

[7] R. N. Shorten and K. S. Narendra, "On common quadratic Lyapunov functions for pairs of stable LTI systems whose system matrices are in companion form," *IEEE Transactions on automatic control*, vol. 48, no. 4, pp. 618–621, 2003.

[8] W. Hagemann, E. Möhlmann, and O. Theel, "Hybrid tools for hybrid systems: Proving stability and safety at once," *Formal Modeling and Analysis of Timed Systems*, 2015.

[9] A. Podelski and S. Wagner, "Model checking of hybrid systems: From reachability towards stability," *Proc. Int'l Conf. on Hybrid systems*, pp. 507–521, 2006.

[10] A. Aminifar *et al.*, "Bandwidth-efficient controller-server co-design with stability guarantees," in *Proc. Design, Automation and Test in Europe Conference and Exhibition (DATE)*, 2014, pp. 1–6.

[11] A. Cervin *et al.*, "The jitter margin and its application in the design of real-time control systems," in *Proc. IEEE Real-Time and Embedded Technology and Applications Symposium (RTAS)*, 2004, pp. 1–9.

[12] G. Frehse *et al.*, "Formal analysis of timing effects on closed-loop properties of control software," in *Proc. Real-Time Systems Symposium*, 2014, pp. 53–62.

[13] S. Quinton, M. Hanke, and R. Ernst, "Formal analysis of sporadic overload in real-time systems," in *Proc. Design Automation Conference (DAC)*, 2012, pp. 515–520.

[14] M. Al Khatib, A. Girard, and T. Dang, "Stability verification of nearly periodic impulsive linear systems using reachability analysis," *IFAC-PapersOnLine*, vol. 48, no. 27, pp. 358–363, 2015.

[15] ——, "Self-triggered control for sampled-data systems using reachability analysis," *IFAC-PapersOnLine*, vol. 50, no. 1, pp. 7881–7886, 2017.

[16] D. J. Hartfiel, *Nonhomogeneous matrix products*. World Scientific, 2002.

[17] J. Lygeros, "Lecture notes on hybrid systems," in *Notes of ENSIETA workshop*, 2004.

[18] T. A. Henzinger, P. W. Kopke, A. Puri, and P. Varaiya, "What's decidable about hybrid automata?" in *Proc. annual ACM Symposium on Theory of computing*, 1995, pp. 373–382.

[19] G. Frehse, R. Kateja, and C. Le Guernic, "Flowpipe approximation and clustering in space-time," in *Proc. Int'l Conf. on Hybrid systems*, 2013, pp. 203–212.