Notions of Conformance Testing for Cyber-Physical Systems: Overview and Roadmap*

Narges Khakpour[1] and Mohammad Reza Mousavi[2]

1 **Department of Computer Science**
Linnaeus University, Sweden
narges.khakpour@lnu.se

2 **Centre for Research on Embedded Systems (CERES)**
School of Information Technology
Halmstad University, Sweden
m.r.mousavi@hh.se

Abstract

We review and compare three notions of conformance testing for cyber-physical systems. We begin with a review of their underlying semantic models and present conformance-preserving translations between them. We identify the differences in the underlying semantic models and the various design decisions that lead to these substantially different notions of conformance testing. Learning from this exercise, we reflect upon the challenges in designing an "ideal" notion of conformance for cyber-physical systems and sketch a roadmap of future research in this domain.

1998 ACM Subject Classification D.2.4 Software/Program Verification

Keywords and phrases Cyber-physical systems, hybrid systems, conformance testing, model-based testing, behavioral pre-orders, hybrid input-output conformance testing, (τ, ε) conformance, approximate simulation

Digital Object Identifier 10.4230/LIPIcs.CONCUR.2015.18

Category Invited Paper

1 Introduction

Cyber-physical systems (CPSs) are the result of the massive and tight interaction of computer systems with physical components and with each other. CPSs have gained importance due to the opportunities that they provide and the criticality of the applications in which they are used. Concerning this importance, model-based approaches are being studied and extended for cyber-physical systems, in order to lay a rigorous foundation for their design and engineering. One typical feature of models used for cyber-physical systems is the integration of discrete behavioral descriptions (often stemming from the "cyber" side), with continuous behavioral descriptions (often stemming from the "physical" side). This combination is well-known in the formal modeling literature and is often referred to as hybrid-systems modeling.[1]

* The work of M. R. Mousavi has been partially supported by the Swedish Research Council (Vetenskapsrådet) with award number 621-2014-5057 (Effective Model-Based Testing of Parallel Systems) and the Swedish Knowledge Foundation (Stiftelsen för Kunskaps- och Kompetensutveckling) in the context of the AUTO-CAAS project.

[1] Note that in addition to the interaction between the computer systems and the physical world, cyber-physical systems also feature complex patterns of communication and interaction among networked computer systems. This aspect is only mentioned in passing in the present paper.

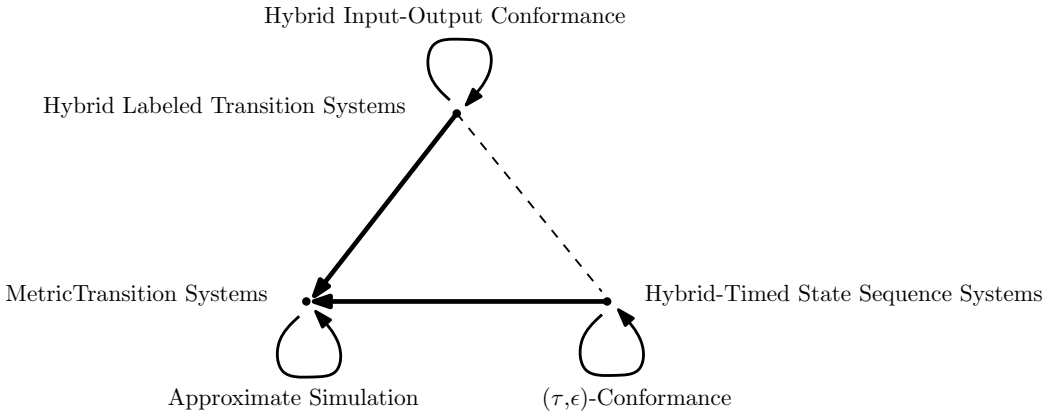

Figure 1 An overview of the semantic models and their mutual translations.

Myriad pieces of research work have been devoted to formal verification of cyber-physical systems and their hybrid-systems models; we refer to [8] for an overview. Model-based conformance testing is a lightweight verification technique, which aims at detecting faults or establishing a level of quality by generating test-cases from a model and applying it to the system under test [16, 38]. Conformance testing based on hybrid-systems models is a relatively recent subject matter [3, 4, 6, 7, 11, 14, 22, 23, 27, 32, 33, 34, 41, 59, 63, 64] and as we demonstrate in the remainder of this paper, still calls for more mature foundations and practical implementations.

A typical feature of hybrid systems research is the reliance on different approaches to system modeling and analysis that stem from different disciplines such as computer science (e.g., finite automata, process algebra, and Petri nets) and control theory (e.g., switching systems and bond graphs). Subsequently, existing approaches to conformance testing are based on different semantic models and assume different semantic properties of the model and the system under test. In this paper, we review the notions proposed in [3, 4, 22, 23, 33, 41, 59], which are, to our knowledge, the main existing proposals for a formal notion of conformance based on behavioral models. We compare these notions and reflect upon the results of this comparison. A summary of the semantic models studied in this paper, their corresponding notions of conformance and the devised translation relations are depicted in Figure 1. The three corners of the triangle denote the semantic models. The self-loops on the corners denote the notions of conformance. The solid lines on the sides of the triangle represent the translations that are presented in this paper. The dashed line is a missing translation; we refer to [46] for a related attempt in this context. We conclude the paper by describing some theoretical and practical challenges in model-based testing of cyber-physical systems.

Structure of the Paper

In Section 2, we give an overview of the three studied semantic models. In Section 3, we present mutual translations between the semantic models as outlined in Figure 1. In Section 4, we present the notions of conformance defined for these semantic models in the literature. Using these notions, in Section 5, we present full abstraction results regarding our translations; namely, we show that conforming models in the source semantic domain are projected into and reflected from conforming models in the target semantic domain. In Section 6, we reflect upon the results of our translations and comparisons and present several challenges for future research.

2 Semantic Models

Several behavioral semantic models exist for hybrid systems of which [18, Chapter2] and [19, 36] provide an overview. In this section, we review the following three of such fundamental semantic models:

- Hybrid labeled transition systems,
- Metric transition systems, and
- Hybrid-timed state sequence systems.

The choice of these models is motivated by the fact that they have been used in the context of the notions of conformance that are presented in Section 4.

2.1 Basic Definitions

The continuous behavior of a hybrid system is captured by the valuation of a set V of continuous variables. We assume that V is partitioned into disjoint sets of input variables, denoted by V_I, and output variables, denoted by V_O. A valuation of V is a function that assigns a value to each variable $v \in V$; here, only variables of type $\mathbb{R}$ (the set of real numbers) are considered. The set of all valuations of V is denoted by $Val(V)$ and is defined as the set of all functions $V \to \mathbb{R}$. To describe the piece-wise evolution of the system, we use the following notion of *trajectory* [43] (called activity in [9]).

▶ **Definition 1** (Trajectory). Let D be the set $\{(0,t] \mid t \in \mathbb{R}^{>0}\}$ of all left-open right-closed intervals.[2] A trajectory ϕ is a function of type $D \to Val(V)$, which maps each element in an interval in D to a valuation. The set of all trajectories for V is denoted by $Trajs(V)$.

Furthermore, we define the restriction operator on valuations and trajectories as follows.

▶ **Definition 2** (Valuation and Trajectory Restriction). Consider a valuation $val \in Val(V)$ and a set $V' \subseteq V$ of variables; the restriction of val to V', denoted by $val \downarrow V'$, is a valuation in $Val(V')$ such that for all $v' \in V'$, $(val \downarrow V')(v') = val(v')$.

Consider a trajectory $\phi : D \to Val(V)$; the restriction of ϕ to $V' \subseteq V$, denoted by (reusing the notation) $\phi \downarrow V'$, is the function of type $D \to Val(V')$, such that for each $d \in D$, $(\phi \downarrow V')(d) = \phi(d) \downarrow V'$.

2.2 Hybrid Labeled Transition System

A hybrid labeled transition system [19, 59] consists of a set of states with discrete (action) and continuous (trajectory) transitions between them. It is formally defined as follows:

▶ **Definition 3** (Hybrid Labeled Transition System (HLTS)). Let A be the union of disjoint sets of input actions A_I and output actions A_O. Assume that V is the union of disjoint sets of input variables V_I and output variables V_O. A hybrid labeled transition system $\mathcal{T}$ is a 5-tuple $(S, s_0, V, L, \to)$, where

- S is a (possibly infinite) set of states;
- $s_0 \in S$ is the initial state;
- V is a set of continuous variables;
- $L = A \uplus Trajs(V)$ is a set of (resp. action or trajectory) labels;

[2] The choice of "left-open right-closed" is arbitrary; we could just as well have chosen "left-closed right-open" intervals. The developments to come will be only slightly different in that case.

- $\rightarrow \subseteq S \times (L \cup \{\xi\}) \times S$ specifies the transition relation, where ξ denotes the internal action.[3]

We may write $s \xrightarrow{l} s'$ to mean $(s, l, s') \in \rightarrow$. We also write $s \xrightarrow{a}$ (respectively, $s \overset{\phi}{\rightsquigarrow}$) to mean that there exists an $s' \in S$ such that $s \xrightarrow{a} s'$ (and $\phi \in \mathit{Trajs}(V)$).

▶ **Definition 4** (Concrete HLTS). An HLTS is *concrete* if it does not have any ξ-labeled transition emanating from its reachable states.

Next, we define the notion of generalized transition relation for an HLTS, which allows us to "jump over" internal actions and to concatenate actions and trajectories to form traces.

▶ **Definition 5** (Generalized Transition Relation). Consider an HLTS $\mathcal{T} = (S, s_0, V, L, \rightarrow)$. The generalized transition relation for $\mathcal{T}$ is defined as the smallest relation $\Rightarrow \subseteq S \times L^* \times S$ where

- $s \overset{\epsilon}{\Rightarrow} s$, where ϵ denotes the empty sequence of labels;
- if $s \xrightarrow{\xi} s'$, then $s \overset{\epsilon}{\Rightarrow} s'$,
- $\forall l \in L$, if $s \xrightarrow{l} s'$, then $s \overset{l}{\Rightarrow} s'$;
- $\forall \alpha, \beta \in L^*$, if $s \overset{\alpha}{\Rightarrow} s''$ and $s'' \overset{\beta}{\Rightarrow} s'$, then $s \overset{\alpha\beta}{\Rightarrow} s'$;

We write $s \overset{\alpha}{\Rightarrow}$ to denote that there exists an $s' \in S$ such that $s \overset{\alpha}{\Rightarrow} s'$. The behavior of a system is specified by its set of traces, which are finite sequences of actions and trajectories.

▶ **Definition 6** (Trace). For HLTS $\mathcal{T}$, a trace is a finite sequence $\alpha \in L^*$ such that $s_0 \overset{\alpha}{\Rightarrow}$, where s_0 is the initial state of $\mathcal{T}$.

The length of a trace α is defined as the number of elements of the sequence and is represented by $|\alpha|$. We denote the set of all traces of $\mathcal{T}$ by $\mathit{Traces}(\mathcal{T})$. The restriction of a trace σ to a set $V' \subseteq V$ of variables, denoted by $\alpha \downarrow V'$, is defined by point-wise restriction of the trajectories in α while keeping the actions intact.

Let $\delta \in A_O$ be a special symbol to denote that a state has at least one emanating trajectory. We assume that HLTSs are normalized, i.e., for $s \in S$, if there exists $a \in \mathit{Trajs}(V)$ such that $s \overset{a}{\rightsquigarrow}$, then $s \xrightarrow{\delta} s$. This assumption is motivated by the notion of conformance on HLTSs.[4]

▶ **Definition 7** (Active Trajectory of a Trace). Consider an HLTS $\mathcal{T} = (S, s_0, V, L, \rightarrow)$, a trace $\alpha \in \mathit{Traces}(\mathcal{T})$, and a point $t \in \mathbb{R}^{>0}$ denoting time; the active trajectory of α at t, denoted by $\mathit{active}(\alpha, t)$ is defined to be a trajectory $\phi_i \in \mathit{Trajs}(V)$ when for each $j \leq i$ there exist $\phi_j \in \mathit{Trajs}(V)$, $\alpha_j \in A^*$, and $\alpha' \in (A \cup \mathit{Trajs}(V))^*$ such that the domain of each ϕ_j is $(0, t_j]$, $\alpha = \alpha_1\phi_1 ... \alpha_i\phi_i\alpha'$ and $\sum_{j=1}^{i-1} t_j < t \leq \sum_{j=1}^{i} t_j$. In that case the elapsed time of the active trajectory at t, denoted by $\mathit{elapsed}(\alpha, t)$, is defined as $t - \sum_{j=1}^{i-1} t_j$.

Moreover, we assume that HLTSs have the following three properties **A1**-**A3** [59]; we refer to [20] for some consequences of these assumptions:

- **A1** if $s \overset{\phi}{\rightsquigarrow} s'$ and $s \overset{\phi}{\rightsquigarrow} s''$, then $s' = s''$.
- **A2** if $s \overset{\phi' \frown \phi''}{\rightsquigarrow} s'$, then there exists s'' such that $s \overset{\phi'}{\rightsquigarrow} s''$ and $s'' \overset{\phi''}{\rightsquigarrow} s'$, where $\phi' \frown \phi''$ denotes concatenation of trajectories after shifting the domain of ϕ''.
- **A3** if $s \overset{\phi'}{\rightsquigarrow} s''$ and $s'' \overset{\phi''}{\rightsquigarrow} s'$, then $s \overset{\phi' \frown \phi''}{\rightsquigarrow} s'$.

[3] In the literature of concurrency theory, internal (unobservable) transitions are labeled by τ; in our context, however, τ denotes the conformance time bound and hence, we use ξ instead.

[4] We deviate from the notation commonly used in the literature in order to avoid clashing with the other notations used for conformance. In the literature, δ is used for lack of discrete output actions and ε for lack of trajectories.

Input-enabledness, defined below, is another constraint, which we in general do not require for all HLTSs. However, our full abstraction result does depend on input-enabledness, as we demonstrate in Section 5.1.

▶ **Definition 8** (Input-enabled HLTS). An HLTS $\mathcal{T} = (S, s_0, V, L, \rightarrow)$ is *input-enabled* if $\forall s \in S, \forall a \in A_I : s \stackrel{a}{\Rightarrow}$ and $\forall \phi \in \mathit{Trajs}(V_I) : \exists \phi' \in \mathit{Trajs}(V)$ such that $\phi' \downarrow V_I = \phi \wedge s \stackrel{\phi'}{\Rightarrow}$.

Finally, we define the notion of determinism, which again plays a role in our full abstraction results.

▶ **Definition 9** (Determinism). An HLTS $\mathcal{T} = (S, s_0, V, L, \rightarrow)$ is *deterministic* when $\forall s, s', s'' \in S$,

- $\forall a \in A$: if $s \stackrel{a}{\rightarrow} s'$ and $s \stackrel{a}{\rightarrow} s''$, then $s' = s''$, and
- $\forall \phi', \phi'' \in \mathit{Trajs}(V_I)$ if $\phi' \downarrow V_I = \phi'' \downarrow V_I$, if $s \stackrel{\phi'}{\rightarrow} s'$ and $s \stackrel{\phi'}{\rightarrow} s''$, then $\phi' = \phi''$ and $s' = s''$.

2.3 Metric Transition System

A metric is a function that defines the distance between different elements of a set:

▶ **Definition 10** (Metric). A metric on a set E is a function $d : E \times E \rightarrow \mathbb{R}_{\geq 0} \cup \{\infty\}$ such that for all $e_1, e_2, e_3 \in E$ the following properties hold:

- $d(e_1, e_3) \leq d(e_1, e_2) + d(e_2, e_3)$,
- $e_1 = e_2 \Leftrightarrow d(e_1, e_2) = 0$, and
- $d(e_1, e_2) = d(e_2, e_1)$.

Metric transition systems [31, 33, 56] are transition systems that are equipped with an observation function and one or more metrics. They are a generic formalism that can be used for specifying different sorts of dynamic behavior, such as discrete, continuous, and hybrid systems. In this paper, we use the metric transition systems whose metric is defined over observations on states [33], as quoted below.

▶ **Definition 11** (Metric Transition System (MTS)). A metric transition system $\mathcal{M}$ is a 7-tuple $(Q, Q_0, I, \rightarrow, O, \mathcal{B}, d)$, where:

- Q is a set of states,
- $Q_0 \subseteq Q$ is a set of initial states,
- I is the set of inputs,
- $\rightarrow \subseteq Q \times I \times Q$ is the transition relation,
- O is a set of outputs,
- $\mathcal{B} : Q \rightarrow O$ is an observation function, and
- $d = (d_1, d_2)$ is a pair of metrics where d_1 is defined over O and d_2 is defined over I.

Intuitively, the inputs in an MTS capture both continuous (system dynamics) and discrete behavior of the system.

2.4 Hybrid-Timed State Sequence System

Hybrid-timed state sequence systems (HSSs) [3, 4, 58] are another semantic model that assumes a discrete sampling of input and output variables.[5] The formal definition of HSS is quoted below.

[5] In [3, 4], Hybrid-Timed State Sequence Systems (HSSs) are simply called "hybrid systems"; we use the former term to be more specific and differentiate between different semantic models for hybrid systems.

▶ **Definition 12** (Hybrid-Timed State Sequence (TSS)). Consider $N \in \mathbb{N}$, $\mathbb{T} = \mathbb{R}_{\geq 0} \times \mathbb{N}$, and a set of variables V. A hybrid-timed state sequence (TSS) is defined as pair (x,σ), where $x \in (Val(V))^N$ and $\sigma \in \mathbb{T}^N$. The i'th element of a TSS (x,σ) is denoted by (x_i,σ_i), where $\sigma_i{=}(t_i, j_i) \in \mathbb{T}$.

We denote the set of all TSSs defined over the set of variables V, considering a specific sample size $N \in \mathbb{N}$, by $\mathbb{TSS}(V, N)$. The set of all TSSs over v regardless of sample size, denoted by $\mathbb{TSS}(V)$, is defined as $\bigcup_{N \in \in \mathbb{N}} \mathbb{TSS}(V, N)$.

The time domain $\mathbb{T} = \mathbb{R}_{\geq 0} \times \mathbb{N}$ is often called the "super-dense" time domain in the literature [44]. The ordering $<$ on the super-dense time domain is the lexicographical ordering by lifting the ordering on real and natural numbers.

Consider $x \in (Val(V))^N$ and $\sigma_x \in \mathbb{T}^N$; we refer to the i'th element of x and σ_x, respectively, by x_i and $\sigma_{x,i}$. Moreover for $t \in \mathbb{T}$, we write t^1 for the first (the real-valued) component of t and t^2 for the second (the natural-number-valued) component of t.

We assume for all $(x,\sigma) \in \mathbb{TSS}(V, N)$, $\sigma : \mathbb{N} \to \mathbb{T}$ is a strictly monotonic function with respect to the lexicographic ordering on $\mathbb{T}$. In other words, for each two consecutive points in σ, the real-valued component does not decrease and if the real-valued component remains the same, the natural-number-valued component increases.

▶ **Definition 13** (Hybrid-Timed State Sequence System (HSS)). A hybrid-timed state sequence system (HSS) $\mathcal{H}$ is a function $\mathcal{H} : H \times \mathbb{TSS}(V_I, N) \to \mathbb{TSS}(V_O, N)$, where $H \subseteq Val(V_I \cup V_O)$.

HSSs are assumed to be input enabled; this constraint is captured by the following formal definition:

▶ **Definition 14** (Input-enabled HSS). An HSS $\mathcal{H} : H \times \mathbb{TSS}(V_I, N) \to \mathbb{TSS}(V_O, N)$ is input-enabled, if it satisfies the following constraint:

$\forall h_0 \in H, \forall (x, \sigma_x) \in \mathbb{TSS}(V_I, N) : \exists (y, \sigma_y) \in \mathbb{TSS}(V_O, N)$ such that $(y, \sigma_y) = \mathcal{H}(h_0, (x, \sigma_x))$.

2.5 Informal Comparison of Semantic Models

The following list summarizes some of the fundamental differences among the three semantic models reviewed in this section.

- (Non-)Determinism: HLTSs and MTSs offer natural support for non-determinism. HSSs, to the contrary, are defined as a function and hence, do not provide support for non-determinism. Non-determinism is often useful in modeling abstraction from details that are either unavailable or irrelevant at the time of modeling.
- Dense-time or sampled trajectories: HLTSs explicitly assume dense-timed trajectories while HSSs assume a fixed sample size. MTSs are oblivious to this choice and can be used to model both dense- and sampled-time trajectories.
- Explicit discrete interactions: HLTSs provide an explicit means for modeling discrete observable actions that can be further used for synchronization between models and with the environment; these represent message passing synchronization among concurrent processes [10]. HSSs inherently lack this means; they could be exploited though to code discrete actions as changes in auxiliary variables' valuations. MTSs stay at a very abstract level and can be used to naturally model discrete actions as inputs.
- Input-enabledness: HLTSs and MTSs are not required to be input-enabled. HSSs as a semantic model may not be input-enabled, but the authors seem to have put it as a requirement on the semantic domain.

- Observations on states or transitions: HLTSs suggest that the observations should be placed on transitions and states do not carry any observable information. For HSSs the notions of states and transitions are a bit blurred; if one takes states to be valuations of variables and transitions to be labeled by the difference of super-dense time points, then HSSs carry observable information both on states and transitions [50]. Likewise, MTSs carry observable information both in states and transitions.
- Partial valuations: HLTSs assume that the valuation of all variables are defined by trajectories; HSSs, however, do not make such an assumption and allow for different sampling of input and output variable valuations. Similar to the previous cases MTSs are oblivious to this choice and can be used to model either of the two types of systems.

3 Translations among Semantic Models

In this section, we present translations between the semantic models introduced in Section 2, as illustrated in Figure 1.

3.1 From HLTS to MTS

The translation from HLTS to MTS defines as the states of the target MTS: the triples comprising (discrete) states, input trajectories leading to the state (or $\perp$ if none), and valuations of variables in the source HLTS. The transitions are then the union of discrete transitions and time transitions. Discrete transitions only update the discrete part of the state. In the case of trajectories, we update both the discrete state and the valuation with the state and the valuation at the upper bound of their domain.

▶ **Definition 15** (Translation from HLTS to MTS)**.** Consider an HLTS $\mathcal{T} = (S, s_0, V, L, \rightarrow)$; an MTS $\mathcal{M} = (Q, Q_0, I, \rightarrow, O, \mathcal{B}, d)$ is a translation of $\mathcal{T}$ when it satisfies the following constraints:

- $Q = S \times (\mathit{Trajs}(V_I) \cup \{\perp\}) \times \mathit{Val}(V)$,
- $Q_0 = \{\langle s_0, \phi, x\rangle \mid \phi \in \mathit{Trajs}(V_I) \cup \{\perp\}, x \in \mathit{Val}(V)\}$,
- $I = A \cup \mathbb{R}^{\geq 0}$,
- for each generalized transition $s \overset{\alpha}{\Rightarrow} s'$ of $\mathcal{T}$, and for all $x \in \mathit{Val}(V)$,
  - if $\alpha \in A$, then $\langle s, \phi, x\rangle \xrightarrow{\alpha} \langle s', \perp, x\rangle$,
  - if $\alpha \in \mathit{Trajs}(V)$, then $\langle s, \phi_\perp, x\rangle \xrightarrow{t} \langle s', \alpha \downarrow V_I, \alpha(t)\rangle$, where $\mathit{dom}(\alpha) = (0, t]$ and $\phi_\perp \in \mathit{Trajs}(V_I) \cup \{\perp\}$,
- $O \subseteq \mathit{Trajs}(V_I) \times \mathit{Val}(V)$,
- $\mathcal{B}(\langle s, \phi, x\rangle) = (\phi, x)$, for all $\langle s, \phi, x\rangle \in Q$,
- $d = (d_1, d_2)$, where

$$d_1((\phi_\perp, x), (\phi'_\perp, y)) = \begin{cases} ||x - y|| & \text{if } \phi_\perp = \phi'_\perp = \perp \text{ or} \\ & \quad \forall t \in \mathit{dom}(\phi_\perp) \cap \mathit{dom}(\phi'_\perp) : \\ & \quad\quad \phi_\perp(t) \downarrow V_I = \phi'_\perp(t) \downarrow V_I \\ \infty & \text{otherwise} \end{cases}$$

$$d_2(t, t') = |t - t'|$$

3.2 From HSS to MTS

In the translation from HSS to MTS, we define as the state of the target MTS: the initial condition, the input TSS, the output TSS and the current moment of time (in the super-dense time domain). (The initial state is an exception; it is denoted by $\perp$ and does not assume any initial state, input- or output TSS.) We define two types of transitions:

- initial transitions that define an initial condition and an input TSS for a state, and
- timed transitions that are labeled by relative time (a real number) denoting the amount of time need to reach a different output valuation in the past or in the future.

The metric on states compares the current valuations of output TSSs and the metric on transitions takes the difference in the amount of time passed.

▶ **Definition 16** (Translation from HSS to MTS). Consider HSS $\mathcal{H} : H_0 \times \mathbb{TSS}(V_I, N) \to \mathbb{TSS}(V_O, N)$; MTS $\mathcal{M} = (Q, Q_0, I, \to, O, \mathcal{B}, d)$ is a translation of $\mathcal{H}$ with respect to initial condition $h \in H_0$, if it satisfies the following constraints:

- $Q = \{\perp\} \cup$
  $\{\langle h_0, (x, \sigma_x), (y, \sigma_y), \sigma_{y,i}\rangle \mid$
  $h_0 \in H_0,$
  $(x, \sigma(x)) \in \mathbb{TSS}(V_I, N),$
  $(y, \sigma(y)) \in \mathbb{TSS}(V_O, N),$
  $(y, \sigma_y) = \mathcal{H}(h_0, (x, \sigma_x)),$
  $i \in [1, N]\}$
- $Q_0 = \{\perp\}$,
- $I = \mathbb{R}$,
- $\perp \xrightarrow{0} \langle h_0, (x, \sigma_x), (y, \sigma_y), \sigma_{y,1}\rangle$, for each $\langle h_0, (x, \sigma_x), (y, \sigma_y), \sigma_{y,1}\rangle \in Q$, and

  $\langle h_0, (x, \sigma_x), (y, \sigma_y), \sigma_{y,i}\rangle \xrightarrow{\sigma^1_{y,j} - \sigma^1_{y,i}} \langle h_0, (x, \sigma_x), (y, \sigma_y), \sigma_{y,j}\rangle$.
  Moreover, the transition relation $\longrightarrow$ is closed under the following deduction rule:

$$\frac{q_0 \xrightarrow{t_0} q_1 \quad q_1 \xrightarrow{t_1} q_2}{q_0 \xrightarrow{t_0+t_1} q_2}$$

- $O = \{\perp\} \cup H \times \mathbb{TSS}(V_I, N) \times Val(V_O)$,
- $\mathcal{B}(\perp) = \perp$,
  $\mathcal{B}(\langle h_0, (x, \sigma_x), (y, \sigma_y), \sigma_{y,i}\rangle) = (h_0, (x, \sigma_x), val_O)$ if and only if $y_i = val_O$,
- $d = (d_1, d_2)$, where

$$d_1(a, b) = \begin{cases} 0 & a = b = \perp \\ ||val_O - val'_O|| & a = (h_0, (x, \sigma_x), val_O) \\ & b = (h_0, (x, \sigma_x), val'_O) \\ \infty & \text{otherwise} \end{cases}$$

$$d_2(t, t') = |t - t'|$$

Next, we informally explain the definition of the metrics and the transition relation in the above-given translation.

Concerning metric d_1, the first case is self-explanatory. The second case compares the current valuations of output variables only in cases where the initial conditions and input

TSSs are identical. The last case covers both comparing the initial state with the rest and also comparing those states that have different initial conditions or input TSSs. Metric d_2 is also straightforward and only concerns the difference in relative time to reach a state.

Regarding the transition relation, the initial state is connected by a 0-labeled transition to all states in which a valid initial condition and an output produces an output TSS; moreover, in the target state the current time is set to the time of the first element in the output TSS. The time transitions are straightforward, once the initial condition and input and output TSSs are fixed, a time transition moves back and forth between the points of time (in the super-dense time domain) when outputs are produced. The last deduction rule in the definition of $\xrightarrow{t}$ is only needed to connect the initial state by a single transition to all reachable states. This is a technical requirement that pops up in the forthcoming proof of full abstraction.

4 Notions of Conformance

In this section, we review the different notions of conformance testing defined for the semantic models presented in Section 2.

4.1 Hybrid Input-Output Conformance (HIOCO)

We start with reviewing the notion of hybrid input-output conformance (*HIOCO*) [59, 60]. To this end, we first give some preliminary definitions. To avoid repetition, we assume in the remainder of this section that HLTS $\mathcal{T}$ is a 5-tuple $(S, s_0, V, L, \rightarrow)$.

▶ **Definition 17** (**after** Operator)**.** For HLTS $\mathcal{T}$ and trace $\alpha \in \mathit{Traces}(\mathcal{T})$, we define $\mathcal{T}$ **after** $\alpha = \left\{ s \,\middle|\, s_0 \stackrel{\alpha}{\Rightarrow} s \right\}$

▶ **Definition 18** (Trajectories of a State)**.** Consider HLTS $\mathcal{T}$ and state $s \in S$, then **trj**(s) is defined as $\mathbf{trj}(s) = \left\{ \sigma \in \mathit{Trajs}(V) \mid s \stackrel{\sigma}{\Rightarrow} \right\}$. This definition is extended to a set of states $S' \subseteq S$ as $\mathbf{trj}(S') = \bigcup_{s \in S'} \mathbf{trj}(s)$.

▶ **Definition 19** (Trajectory Infiltration)**.** Consider $\Sigma_{\mathcal{I}}, \Sigma_{\mathcal{S}} \subseteq \mathit{Trajs}(V)$ and $V_I \subseteq V$ as the input partition of V;

$$\mathbf{infilter}(\Sigma_{\mathcal{I}}, \Sigma_{\mathcal{S}}) = \{\sigma \in \Sigma_{\mathcal{I}} \mid \exists \sigma' \in \Sigma_{\mathcal{S}}.\ \sigma \downarrow V_I = \sigma' \downarrow V_I\}$$

Informally, $\mathbf{infilter}(\Sigma_{\mathcal{I}}, \Sigma_{\mathcal{S}})$ is the set of all trajectories $\sigma \in \Sigma_{\mathcal{I}}$ such that they find a counterpart in $\Sigma_{\mathcal{S}}$ that agrees with σ on input variables. In this definition $\mathcal{I}$ and $\mathcal{S}$ typically denote the implementation and the specification, respectively.

▶ **Definition 20** (State Output)**.** Consider $s \in S$; the output of s, denoted by $\mathbf{out}(s) \subseteq A_O$, is defined as $\mathbf{out}(s) = \{a \in A_O | s \stackrel{a}{\rightarrow}\}$, where A_O is assumed to contain δ. This definition is extended to a set of states $C \subseteq S$ as $\mathbf{out}(C) = \bigcup_{s \in C} \mathbf{out}(s)$.

Using the above-given definitions, we are now ready to define the notion of HIOCO.

▶ **Definition 21** (Hybrid I/O Conformance)**.** Consider an HLTS $\mathcal{S}$; an input-enabled HLTS $\mathcal{I}$ is said to be hybrid input-output conforming to $\mathcal{S}$, denoted by $\mathcal{I}$ *hioco* $\mathcal{S}$, if and only if for all traces $\alpha \in \mathit{Traces}(\mathcal{S})$:

$$\mathbf{out}\ (\mathcal{I}\ \mathbf{after}\ \alpha) \subseteq \mathbf{out}\ (\mathcal{S}\ \mathbf{after}\ \alpha) \wedge$$
$$\mathbf{infilter}\ (\mathbf{trj}(\mathcal{I}\ \mathbf{after}\ \alpha), \mathbf{trj}\ (\mathcal{S}\ \mathbf{after}\ \alpha)) \subseteq \mathbf{trj}\ (\mathcal{S}\ \mathbf{after}\ \alpha)$$

Informally, this notion states that for all traces α of specification $\mathcal{S}$, the discrete outputs of the implementation $\mathcal{I}$ after α must be a subset of that of the specification. Moreover after each trace α of specification $\mathcal{S}$, the set of trajectories in the implementation, with a corresponding trajectory in the specification (with equal input valuations), must be a subset of the trajectories of the specification.

4.2 Approximate Simulation

Approximate simulation [33, 32, 34, 52] is a notion of conformance that describes how well a system is approximated by another system in terms of its observable behavior. We assume below that MTSs $\mathcal{M}_i$ are defined as $(Q, Q_{0i}, I, \rightarrow, O, \mathcal{B}, d)$, i.e., have all components in common apart from the initial state.

▶ **Definition 22** (Approximate Simulation). Consider two MTSs $\mathcal{M}_1$ and $\mathcal{M}_2$; a relation $R_{\varepsilon,\tau} \subseteq Q \times Q$ is an *approximate simulation relation with precision* (ε, τ), for $\varepsilon \geq 0, \tau \geq 0$, when for all $(q_1, q_2) \in R_{\varepsilon,\tau}$,

- $d_1(\mathcal{B}(q_1), \mathcal{B}(q_2)) \leq \varepsilon$,
- if $q_1 \xrightarrow{i_1} q_1'$, then there exists a transition $q_2 \xrightarrow{i_2} q_2'$ such that $d_2(i_1, i_2) \leq \tau$ and $(q_1', q_2') \in R_{\varepsilon,\tau}$.

$\mathcal{M}_2$ *approximately simulates* $\mathcal{M}_1$ with precision (ε, τ), denoted by $\mathcal{M}_1 \preceq_{\varepsilon,\tau} \mathcal{M}_2$, when there exists a relation $R_{\varepsilon,\tau}$, such that for all $q_1 \in Q_{01}$, there exists $q_2 \in Q_{02}$ such that $(q_1, q_2) \in R_{\varepsilon,\tau}$.

$\mathcal{M}_1$ and $\mathcal{M}_2$ are *approximately bisimilar* with precision (ε, τ), denoted by $\mathcal{M}_1 \equiv_{\varepsilon,\tau} \mathcal{M}_2$, when there exists a symmetric simulation relation relating their sets of initial states.

4.3 (τ, ε)-Conformance

The notion of (τ, ε)-conformance relates two HSSs when for each sampled input, the sampled output of related pairs differ in time with the maximum value of τ and differ in valuations with the maximum value of ε.

▶ **Definition 23** ((τ, ε)-Conformance). Consider a test duration $T \in \mathbb{T}$ and $\tau, \varepsilon > 0$ and TSSs $(y, \sigma_y) \in \mathbb{TSS}(V, N)$ and $(y', \sigma_{y'}) \in \mathbb{TSS}(V, N')$; then (τ, ε)-*conforms to* $(y', \sigma_{y'})$, denoted by $(y, \sigma_y) \approx_{\tau,\varepsilon} (y', \sigma_{y'})$, if and only if

1. for all $i \in [1, N]$ such that $t_i \leq T$ there exists $k \in [1, N']$ such that $t_k \leq T$, $|t_i^1 - t_k'^1| \leq \tau$ and $||y_i - y_k'|| \leq \varepsilon$, and
2. for all $i \in [1, N']$ such that $t_i' \leq T$, there exists $k \in [1, N]$ such that $t_k \leq T$, $|t_i'^1 - t_k^1| \leq \tau$ and $||y_i' - y_k|| \leq \varepsilon$.

HSS $\mathcal{H}$ (τ, ε)-*conforms to* HSS $\mathcal{H}'$ (both with the same sets of input and output variables), denoted by $\mathcal{H} \approx_{\tau,\varepsilon} \mathcal{H}'$, when for each initial condition h_0 and each TSS (x, σ_x) on the common input variables V_I, $\mathcal{H}(h_0, (x, \sigma_x)) \approx_{\tau,\varepsilon,V_O} \mathcal{H}'(h_0, (x, \sigma_x))$.

We have made two slight modifications with respect to the original definition of (τ, ε)-conformance in [3, 4]: firstly, the original notion requires the number of discrete jumps (the natural-number-valued part of time) for related states to be identical. The main purpose for this choice is to be sensitive to Zeno behavior, but we consider this a too strong constraint for the purpose and hence, we removed this it. (We expect our results to hold in the original setting, as well.) Secondly, the original definition requires the difference of the timing and

valuations to be strictly less than the conformance bounds τ and ϵ, respectively. We have changed this to *less than or equal* to be consistent with the earlier definition of approximate simulation.

5 Full Abstraction for Translations

In this section, we show that the translations introduced in Section 3 preserve the notions of conformance for a designated subset of the source semantic domain. We also give some suggestions as to how to generalize these results to all semantic models.

5.1 Full Abstraction for the HLTS-MTS Translation

We start with proving that our translation from HLTS to MTS projects and reflects HIOCO-conforming pairs of models to approximately similar models with approximation bounds set to 0. To obtain this result, we need to focus on a very restricted subset of HLTSs, namely concrete input-enabled deterministic ones. After we present the proof, we sketch some initial ideas as to how to relax these restrictive assumptions. To avoid repetition, we assume that the HLTSs in the remainder of this section are of the form $(S_{\mathcal{T}}, s_{0\mathcal{T}}, V, L, \rightarrow)$, where $\mathcal{T}$ is the HLTS at hand. Note that we do not subscript the transition relation because it is always clear from the context and we do not subscript the set of variables and labels, because they are assumed to be the same throughout this section.

▶ **Theorem 24.** *Consider two concrete input-enabled HLTSs $\mathcal{I}$ and $\mathcal{S}$, where $\mathcal{S}$ is deterministic; assume that $trans(\mathcal{I})$ and $trans(\mathcal{S})$ denote the MTS translations of $\mathcal{I}$ and $\mathcal{S}$, respectively. Then, the following statement holds:*

$$\mathcal{I} \textbf{ hioco } \mathcal{S} \Rightarrow trans(\mathcal{I}) \preceq_{0,0} trans(\mathcal{S}).$$

Proof. We start with proving the following lemma. This lemma makes sure that the simulation relation between the states of $trans(\mathcal{I})$ and $trans(\mathcal{S})$ (given in the remainder of the proof) is well-defined.

▶ **Lemma 25.** *Consider concrete input-enabled and deterministic HLTSs $\mathcal{I}$ and $\mathcal{S}$ and an arbitrary trace $\alpha \in Traces(S)$; $\mathcal{I}$ **hioco** $\mathcal{S}$ implies that for all $s_1 \in \mathcal{I}$ **after** α, there exits $s_2 \in \mathcal{S}$ **after** α such that $\mathcal{I}_{s_1}$ **hioco** $\mathcal{S}_{s_2}$, where for an HLTS $\mathcal{T}$ and state s, $\mathcal{T}_s$ denotes the same HLTS with the initial state s.*

Proof. We prove the lemma by induction on the length of α.

Since both $\mathcal{I}$ and $\mathcal{S}$ are concrete (i.e., $s_{0\mathcal{I}}$ **after** $\epsilon = s_{0\mathcal{I}}$ and $s_{0\mathcal{S}}$ **after** $\epsilon = s_{0\mathcal{S}}$) and $\mathcal{I}_{s_{0\mathcal{I}}} = \mathcal{I}$ and $\mathcal{S}_{s_{0\mathcal{S}}} = \mathcal{S}$, the base case follows immediately.

Assume that for all $\alpha_n \in Traces(S)$ with $|\alpha_n| \leq n$ and all $s_1 \in \mathcal{I}$ **after** α_n, there exists $s_2 \in \mathcal{S}$ **after** α_n, such that $\mathcal{I}_{s_1}$ **hioco** $\mathcal{S}_{s_2}$.

Consider a trace $\alpha_{n+1} \in Traces(S)$; if $\mathcal{I}$ **after** $\alpha_{n+1} = \emptyset$, then the thesis follows vacuously.

For the case that $\mathcal{I}$ **after** $\alpha_{n+1} \neq \emptyset$, let l be the last element of α_{n+1}. Hence, there is an l-labeled transition $s_1 \xrightarrow{l} s_1'$, where $s_1 \in \mathcal{I}$ **after** α_n and $s_1' \in \mathcal{I}$ **after** α_{n+1}. It follows from the induction hypothesis that there exists a state $s_2 \in \mathcal{S}$ **after** α_n such that $\mathcal{I}_{s_1}$ **hioco** $\mathcal{S}_{s_2}$. If l is an input, since $\mathcal{S}$ is input-enabled, there exists a transition $s_2 \xrightarrow{l} s_2'$. If l is an output, it follows from the first condition of Definition 21 that $s_2 \xrightarrow{l} s_2'$. If l is a trajectory, it follows from input-enabledness of $\mathcal{S}$ that $\mathcal{I}$ affords an l' trajectory such that $l \downarrow V_I = l' \downarrow V_I$. Then, it follows from the second condition of Definition 21 that that $s_2 \xrightarrow{l} s_2'$.

Hence, it remains only to prove that $\mathcal{I}_{s'_1}$ **hioco** $\mathcal{S}_{s'_2}$; we proceed with a proof by contradiction. Suppose there exists $\alpha' \in \mathit{Traces}(S_{s'_2})$ such that one of the two conditions of Definition 21 does not hold for $\mathcal{I}_{s'_1}$ and $\mathcal{S}_{s'_2}$. It follows from $\alpha' \in \mathit{Traces}(S_{s'_2})$ that $\alpha_{n+1}\alpha' \in \mathit{Traces}(S)$ and hence, the same condition is violated for $\mathcal{I}$ and $\mathcal{S}$, which contradicts the assumption of the lemma. Hence, we conclude that $\mathcal{I}_{s'_1}$ **hioco** $\mathcal{S}_{s'_2}$. ◀

We define the following relation R and then prove that R is a witnessing approximate simulation relation for $\mathit{trans}(\mathcal{I}) \preceq_{0,0} \mathit{trans}(\mathcal{S})$:

$$R = \{(\langle s_1, \phi, x\rangle, \langle s_2, \phi, x\rangle) \mid \mathcal{I}_{s_1} \textbf{ hioco } \mathcal{S}_{s_2}, \phi \in \mathit{Trajs}(V_I) \cup \{\bot\}, x \in \mathit{Val}(V)\}$$

It follows from Definitions 10 and 15 (particularly, the definition of $\mathcal{B}$ in the latter) that for each two related states in R, $d_1(\mathcal{B}(\langle s_1, \phi, x\rangle), \mathcal{B}(\langle s_2, \phi, x\rangle)) = d_1((\phi, x), (\phi, x)) = 0$.

From $\mathcal{I}$ **hioco** $\mathcal{S}$, it follows that $(\langle s_{0,\mathcal{I}}, \bot, x\rangle, \langle s_{0,\mathcal{S}}, \bot, x\rangle) \in R$.

Hence, it only remains to show that if $\langle s_1, \phi, x\rangle \xrightarrow{\alpha} \langle s'_1, \phi', x'\rangle$, for some $\alpha \in L$, then there exists s'_2 such that $\langle s_2, \phi, x\rangle \xrightarrow{\alpha} \langle s'_2, \phi', x'\rangle$ and $(\langle s'_1, \phi', x'\rangle, \langle s'_2, \phi', x'\rangle) \in R$.

We distinguish the following cases based on whether α is an input action, output action, or a trajectory:

- $\alpha \in A_I$: Since $\mathcal{S}$ is input-enabled, $\alpha \in \mathit{Traces}(\mathcal{S}_{s_1})$. Hence, s_2 affords an α-labeled transition to some state s'_2. Since $\mathcal{S}$ is deterministic, this is the only α-labeled transition afforded by s_1. Hence, it follows from $\mathcal{I}_{s_1}$ **hioco** $\mathcal{S}_{s_2}$ and Lemma 25 that $\mathcal{I}_{s'_1}$ **hioco** $\mathcal{S}_{s'_2}$. Finally, it follows from the latter statement and the construction of R that $(\langle s'_1, \bot, x'\rangle, \langle s'_2, \bot, x'\rangle) \in R$, which was to be shown.
- $\alpha \in A_O$: It follows from $\mathcal{I}_{s_1}$ **hioco** $\mathcal{S}_{s_2}$ and Definition 21. Hence, s_2 affords an α-labeled transition to some state s'_2. With a similar reasoning as in the above item, we obtain that $(\langle s'_1, \bot, x'\rangle, \langle s'_2, \bot, x'\rangle) \in R$.
- $\alpha \in \mathit{Trajs}(V)$: Then, Since $\mathcal{S}$ is input-enabled, there exists some $\phi' \in \mathit{Traces}(\mathcal{S}_{s_1})$ such that $\phi' \downarrow V_I = \alpha \downarrow V_I$. It follows from $\mathcal{I}_{s_1}$ **hioco** $\mathcal{S}_{s_2}$ that s_2 affords an α' labeled trajectory to some state s'_2. With a similar reasoning as in the first item, we obtain that $(\langle s'_1, \phi' \downarrow V_I, x'\rangle, \langle s'_2, \phi' \downarrow V_I, x'\rangle) \in R$. ◀

The assumptions that both HLTSs are concrete and input-enabled and that the specification is deterministic are very restrictive. We envisage that concreteness and determinism assumptions can be relaxed. A possible proof technique could require a slightly modified translation that is reminiscent of the subset construction for transforming non-deterministic finite automata to determinstic ones; a similar transformation has been proposed in the setting of IOCO, see, e.g., [49, Definition 8]. For non-input-enabled HLTSs, we believe that a slightly different notion than approximate bisimulation needs to be employed; this notion should be similar to XY-bisimulation [2].

▶ **Theorem 26.** *Consider two concrete deterministic HLTSs $\mathcal{I}$ and $\mathcal{S}$, where $\mathcal{S}$ is also input-enabled, and assume that $\mathit{trans}(\mathcal{I})$ and $\mathit{trans}(\mathcal{S})$ denote their MTS translations, respectively. Then, the following statement holds:*

$$\mathit{trans}(\mathcal{I}) \preceq_{0,0} \mathit{trans}(\mathcal{S}) \Rightarrow \mathcal{I} \textbf{ hioco } \mathcal{S}$$

Proof. Consider the approximate bisimilarity relation $\preceq_{0,0}$ (i.e., the largest approximate bisimulation relation). We proceed with proving the following lemma.

▶ **Lemma 27.** *Consider two states* $\langle s_1, \phi_1, x_1\rangle$ *and* $\langle s_2, \phi_2, x_2\rangle$*, respectively, of trans*$(\mathcal{I})$ *and trans*$(\mathcal{S})$ *such that* $\langle s_1, \phi_1, x_2\rangle \preceq_{0,0} \langle s_2, \phi_2, x_2\rangle$ *; the following statements hold:*

$$\boldsymbol{out}\ (s_1) \subseteq \boldsymbol{out}\ (s_2) \tag{1}$$

and

$$\boldsymbol{trj}(s_1) \subseteq \boldsymbol{trj}\ (s_2) \tag{2}$$

Proof. We next proceed with the proofs of the two statements in the thesis:

- In order to prove that $\mathbf{out}\ (s_1) \subseteq \mathbf{out}\ (s_2)$, consider an arbitrary action $a \in \mathbf{out}\ (s_1)$. This can only be due to a transition $s_1 \xrightarrow{a} s_1'$ in $\mathcal{I}$. It follows from Definition 15 that $\langle s_1, \phi_1, x_1\rangle \xrightarrow{a} \langle s_1', \perp, x_2\rangle$. Due to $\langle s_1, \phi_1, x_2\rangle \preceq_{0,0} \langle s_2, \phi_2, x_2\rangle$, we obtain $\langle s_2, \phi_2, x_2\rangle \xrightarrow{a} \langle s_2', \phi_2, x_2\rangle$ for some $\langle s_2', \phi_2, x_2\rangle$ such that $\langle s_1', \perp, x_2\rangle \preceq_{0,0} \langle s_2', \phi_2, x_2\rangle$. It then follows from Definition 15 that $\phi_2 = \perp$ and $x_1 = x_2$. It also follows from Definition 15 that the transition of $\langle s_2, \phi_2, x_2\rangle$ is due to a transition $s_2 \xrightarrow{a} s_2'$ in $\mathcal{S}$. Hence, $a \in \mathbf{out}\ (s_2)$, which was to be shown.
- In order to prove that $\mathbf{trj}(s_1) \subseteq \mathbf{trj}\ (s_2)$ consider an arbitrary trajectory $\phi \in \mathbf{trj}(s_1)$. This statement can only be due to a transition $s_1 \xrightarrow{\phi} s_1'$ in $\mathcal{I}$. It follows from Definition 15 that $\langle s_1, \phi_1, x_1\rangle \xrightarrow{t} \langle s_1', \phi \downarrow V_I, \phi(t)\rangle$. Due to $\langle s_1, \phi_1, x_1\rangle \preceq_{0,0} \langle s_2, \phi_2, x_2\rangle$, we obtain $\langle s_2, \phi_2, x_2\rangle \xrightarrow{t} \langle s_2', \phi \downarrow V_I, \phi(t)\rangle$ for some s_2' such that $\langle s_1', \phi \downarrow V_I, \phi(t)\rangle \preceq_{0,0} \langle s_2', \phi \downarrow V_I, \phi(t)\rangle$. Hence, it follows from Definition 15 that s_2 affords a trajectory ϕ' such that $s_2 \xrightarrow{\phi'} s_2'$, $dom(\phi') = (0, t]$, $\phi'(t) = \phi(t)$, and $\phi' \downarrow V_I = \phi \downarrow V_I$. We claim that for each $t' \in (0, t]$ it holds that $\phi'(t') = \phi(t')$.

  Take an arbitrary $t' \in (0, t]$, it follows from assumption **A2** that $\langle s_1, \phi_1, x_1\rangle \xrightarrow{t'} \langle s_1'', \phi_3 \downarrow V_I, \phi(t')\rangle$ and $\langle s_1'', \phi_3 \downarrow V_I, \phi(t')\rangle \xrightarrow{t-t'} \langle s_1''', \phi_4 \downarrow V_I, \phi(t)\rangle$ for some $\phi_3, \phi_4 \in Traj(V)$ such that $\phi = \phi_3 \frown \phi_4$. Since $\phi = \phi_3 \frown \phi_4$, it follows from Definition 9 and assumption **A1** that $s_1''' = s_1'$.

  It follows from $\langle s_1, \phi_1, x_1\rangle \preceq_{0,0} \langle s_2, \phi_2, x_2\rangle$ that $\langle s_2, \phi_2, x_2\rangle \xrightarrow{t'} \langle s_2'', \phi_3' \downarrow V_I, \phi_3'(t')\rangle$ for some $\langle s_2'', \phi_3' \downarrow V_I, \phi_3'(t')\rangle$ such that $\langle s_1'', \phi_3 \downarrow V_I, \phi(t')\rangle \preceq_{0,0} \langle s_2'', \phi_3' \downarrow V_I, \phi_3'(t')\rangle$ and $\phi_3 \downarrow V_I = \phi_3' \downarrow V_I$. Also, from $\langle s_1'', \phi_3 \downarrow V_I, \phi(t')\rangle \preceq_{0,0} \langle s_2'', \phi_3' \downarrow V_I, \phi_3'(t')\rangle$, we obtain that $\phi(t') = \phi_3'(t')$. Likewise, we obtain from $\langle s_1'', \phi_3 \downarrow V_I, \phi(t')\rangle \preceq_{0,0} \langle s_2'', \phi_3' \downarrow V_I, \phi(t')\rangle$ that $\langle s_2'', \phi_3' \downarrow V_I, \phi(t')\rangle \xrightarrow{t-t'} \langle s_2''', \phi_4' \downarrow V_I, \phi(t)\rangle$ for some $\langle s_2''', \phi_4' \downarrow V_I, \phi(t)\rangle$ such that $\langle s_1''', \phi_4 \downarrow V_I, \phi(t)\rangle \preceq_{0,0} \langle s_2''', \phi_4' \downarrow V_I, \phi(t)\rangle$ and $\phi_4 \downarrow V_I = \phi_4' \downarrow V_I$. We have that $\phi = \phi_3 \frown \phi_4$, $\phi_3 \downarrow V_I = \phi_3' \downarrow V_I$, $\phi_4 \downarrow V_I = \phi_4' \downarrow V_I$ and $\phi \downarrow V_I = \phi' \downarrow V_I$; hence, due to **A1**, we obtain that $\phi' = \phi_3' \frown \phi_4'$. We already had that $\phi(t') = \phi_3'(t')$ and from $\phi' = \phi_3' \frown \phi_4'$, we obtain that $\phi(t') = \phi'(t')$, which was to be shown. ◀

In order to prove the theorem, we prove the following claim:

Consider a trace $\alpha_n \in Traces(\mathcal{S})$ with length n; for all $s_1 \in \mathcal{I}$ **after** α_n, there exists $s_2 \in \mathcal{S}$ **after** α_n such that $\langle s_1, \phi, x\rangle \preceq_{0,0} \langle s_2, \phi, x\rangle$ for all $\phi \in TrajV$, $x \in Val(V)$.

Once we prove the claim, by considering Lemma 27 the theorem follows.

We prove the claim by induction on n. For the base case, we have that $s_{0\mathcal{I}}\mathbf{after}\epsilon = s_{0\mathcal{I}}$ and $s_{0\mathcal{S}}\mathbf{after}\epsilon = s_{0\mathcal{S}}$ and it follows from Definition 15 and $\langle s_{0\mathcal{I}}, \phi, x\rangle \preceq_{0,0} \langle s_{0\mathcal{S}}, \perp, x\rangle$ and s_2 is $s_{0\mathcal{S}}$.

Assume that for all $k \leq n$, $\alpha_k \in Traces(\mathcal{S})$ and all $s_1 \in \mathcal{I}$ **after** α_k, there exists $s_2 \in \mathcal{S}$ **after** α_k such that $\langle s_1, \phi, x\rangle \preceq_{0,0} \langle s_2, \phi, x\rangle$.

Consider a trace $\alpha_{n+1} \in Traces(\mathcal{S})$; if $\mathcal{I}$ **after** $\alpha_{n+1} = \emptyset$, then the claim follows. For the case that $\mathcal{I}$ **after** $\alpha_{n+1} \neq \emptyset$, consider $l \in L$ to be the last element of α_{n+1} and assume that

$s_1 \in \mathcal{I}$ **after** α_n. According to the induction hypothesis, there exists $s_2 \in \mathcal{S}$ **after** α_n such that $\langle s_1, \phi, x\rangle \preceq_{0,0} \langle s_2, \phi, x\rangle$. We distinguish the following cases based on the type of the last element l in α_{n+1}:

- $l \in A$: Since $s_1 \xrightarrow{a} s_1'$, then $\langle s_1, \phi, x\rangle \xrightarrow{a} \langle s_1', \bot, x\rangle$ based on Definition 15. Considering $\langle s_1, \bot, x\rangle \preceq_{0,0} \langle s_2, \bot, x\rangle$, $\langle s_1, \phi, x\rangle \xrightarrow{a} \langle s_1', \bot, x\rangle$ and Definition 22, it follows that there exists a state s_2' such that $\langle s_2, \phi, x\rangle \xrightarrow{a} \langle s_2', \bot, x\rangle$ and $\langle s_1', \bot, x\rangle \preceq_{0,0} \langle s_2', \bot, x\rangle$.
  In order to show that $\langle s_1', \phi, x\rangle \preceq_{0,0} \langle s_2', \phi, x\rangle$, we prove the following more general claim:

▶ **Lemma 28.** *If* $\langle s_1', \phi, x\rangle \preceq_{0,0} \langle s_2', \phi, x\rangle$*, then* $\langle s_1', \phi', x'\rangle \preceq_{0,0} \langle s_2', \phi', x'\rangle$ *for any* $\phi' \in$ *Trajs*(V) *and* $x' \in$ *Val*(V)*.*

Proof. Consider the witnessing approximate simulation relation R such that $(\langle s_1', \phi, x\rangle\ , \langle s_2', \phi, x\rangle) \in R$; consider the extension R' of R with all pairs $(\langle s, \phi', x'\rangle, \langle s', \phi', x'\rangle)$ such that for some ϕ' and x', $(\langle s, \phi'', x''\rangle, \langle s', \phi'', x''\rangle) \in R$. It is straightforward to check that R' is an approximate simulation relation. ◀

Therefore, due to Lemma 28, $\langle s_2', \bot, x\rangle$ is the state that is approximately bisimilar to $\langle s_1', \bot, x\rangle$ where $s_2' \in \mathcal{S}$ **after** α_{n+1}, which was to be shown.

- $l \in$ *Trajs*(V): since $l \in$ **trj**(s_1) and $\langle s_1, \phi, x\rangle \preceq_{0,0} \langle s_2, \phi, x\rangle$, due to Lemma 27, we obtain that $l \in$ **trj**(s_2). It follows from Definition 15 that $\langle s_1, \phi, x\rangle \xrightarrow{t} \langle s_1', l \downarrow V_I, l(t)\rangle$, where $dom(l) = (0, t]$. Since $\mathcal{S}$ is deterministic and $l \in$ **trj**(s_2), the only possibility for $\langle s_2, \phi, x\rangle$ to mimic this transition is the transition: $\langle s_2, \phi, x\rangle \xrightarrow{t} \langle s_2', l \downarrow V_I, l(t)\rangle$, for some s_2' such that $\langle s_1', l \downarrow V_I, l(t)\rangle \preceq_{0,0} \langle s_2', l \downarrow V_I, l(t)\rangle$. It follows from Lemma 28 that $\langle s_1', \phi', x'\rangle \preceq_{0,0} \langle s_2', \phi', x'\rangle$, for each $\phi' \in$ *Trajs*(V) and $x' \in$ *Val*(V), which concludes the proof of this item and the theorem. ◀

Similar to Theorem 24, we conjecture that Theorem 26 also holds for non-deterministic HLTSs.

5.2 Full Abstraction for HSS-MTS Translation

In this section, we prove full abstraction results for the translation from HSS to MTS. In both the projection and the reflection theorems, we need to double the original conformance time bound τ. We need to stretch the conformance time bound, because the notion of (τ, ε)-conformance allows for matching valuations within τ time bounds both in the past and in the future, which is a neighborhood of width 2τ.

▶ **Theorem 29.** *Consider two HSSs* $\mathcal{I}$ *and* $\mathcal{S}$ *and* $\varepsilon > 0$*; assume that MTSs* $MTS(\mathcal{I})$ *and* $MTS(\mathcal{S})$ *are translations of* $\mathcal{I}$ *and* $\mathcal{S}$*, respectively. Then, the following statement holds:*

$$\mathcal{I} \approx_{\tau,\varepsilon} \mathcal{S} \Rightarrow MTS(\mathcal{I}) \equiv_{\varepsilon,2\tau} MTS(\mathcal{S}).$$

Proof. Consider the following relation between the states of $MTS(\mathcal{I})$ and $MTS(\mathcal{S})$:

$$\begin{aligned} R_{\varepsilon,2\tau} \;=\; & \{(\bot,\bot),\\ & (\langle h_0, (x,\sigma_x), (y,\sigma_y), \sigma_{y,i}\rangle, \langle h_0, (x,\sigma_x), (y',\sigma_{y'}), \sigma_{y',j}\rangle \mid \\ & \quad (y,\sigma_y) \approx_{\tau,\varepsilon} (y',\sigma_{y'}) \wedge \\ & \quad |\sigma^1_{y,i} - \sigma^1_{y',j}| \le \tau \\ & \quad ||y_i - y'_j|| \le \varepsilon\} \end{aligned}$$

The relation is symmetric and hence, once we prove that this relation is an approximate bisimulation relation, the theorem follows, because the initial states of $MTS(\mathcal{I})$ and $MTS(\mathcal{S})$ are related by $R_{\varepsilon,2\tau}$.

The fact that for each $(q_0, q_1) \in R_{\varepsilon,2\tau}$, $d_1(\mathcal{B}(q_0), \mathcal{B}(q_1)) \leq \varepsilon$ follows from the construction of $R_{\varepsilon,2\tau}$ and Definition 16.

It remains to prove the transfer conditions of Definition 22.

Regarding the transfer condition for $\perp$, assume that in $MTS(\mathcal{I})$, $\perp \xrightarrow{t} \langle h_0, (x, \sigma_x), (y, \sigma_y), \sigma_{y,i}\rangle$; we immediately have that $t = \sigma_{y,i} - \sigma_{y,1}$. It follows from $\mathcal{I} \approx_{\tau,\varepsilon} \mathcal{S}$ that there exists $(y', \sigma_{y'}) = \mathcal{S}(h_0, (x, \sigma_x))$ such that $(y, \sigma_y) \approx_{\tau,\varepsilon} (y', \sigma_{y'})$. In particular, it holds that there exists a $\sigma_{y',j}$ such that $|\sigma^1_{y,i} - \sigma^1_{y',j}| \leq \tau$ and $||y_i - y'_j|| \leq \varepsilon$. It hence, follows from Definition 16 that in $MTS(\mathcal{S})$, $\perp \xrightarrow{t'} \langle h_0, (x, \sigma_x), (y', \sigma_{y'}), \sigma_{y',j}\rangle$, where $t' = \sigma_{y',j} - \sigma_{y',1}$. We note that $|\sigma_{y,1} - \sigma_{y',1}| \leq \tau$, because otherwise, one of the two points does not find a counter part (within the time range of τ) in the other sequence. Hence, we obtain that and we have that $d_2(t, t') = |(\sigma_{y,i} - \sigma_{y',j}) - (\sigma_{y,1} - \sigma_{y',1})| \leq 2\tau$. We also have that $(\langle h_0, (x, \sigma_x), (y, \sigma_y), \sigma_{y,i}\rangle, \langle h_0, (x, \sigma_x), (y', \sigma_{y'}), \sigma_{y',j}\rangle) \in R_{\varepsilon,2\tau}$ and this concludes the transfer condition for $\perp$.

Take arbitrary states $(\langle h_0, (x, \sigma_x), (y, \sigma_y), \sigma_{y,i}\rangle, \langle h_0, (x, \sigma_x), (y', \sigma_{y'}), \sigma_{y',j}\rangle) \in R_{\varepsilon,2\tau}$. Consider a transition $\langle h_0, (x, \sigma_x), (y, \sigma_y), \sigma_{y,i}\rangle \xrightarrow{t} \langle h_0, (x, \sigma_x), (y, \sigma_y), \sigma_{y,k}\rangle$; we have that $t = \sigma_{y,k} - \sigma_{y,i}$. It follows from $(y, \sigma_y) \approx_{\tau,\varepsilon} (y', \sigma_{y'})$ that there exists $m \in \mathbb{N}$ such that $|\sigma^1_{y,k} - \sigma^1_{y',m}| \leq \tau$ and $||y_k - y'_m|| \leq \varepsilon$. Due to Definition 16, we have that $\langle h_0, (x, \sigma_x), (y', \sigma_{y'}), \sigma_{y',j}\rangle \xrightarrow{t'} \langle h_0, (x, \sigma_x), (y', \sigma_{y'}), \sigma_{y',m}\rangle$ and $t' = \sigma^1_{y',j} - \sigma^1_{y',m}$. It also follows from $|\sigma^1_{y,i} - \sigma^1_{y',j}| \leq \tau$ and $|\sigma^1_{y,k} - \sigma^1_{y',m}| \leq \tau$ that $d_2(t, t') = |(\sigma_{y,i} - \sigma_{y',j}) - (\sigma_{y,1} - \sigma_{y',1})| \leq 2\tau$. We also have that $(\langle h_0, (x, \sigma_x), (y, \sigma_y), \sigma_{y,k}\rangle, \langle h_0, (x, \sigma_x), (y', \sigma_{y'}), \sigma_{y',m}\rangle) \in R_{\varepsilon,2\tau}$ and this concludes the only remaining part of the transfer condition and the proof. ◀

▶ **Theorem 30.** *Consider two HSSs $\mathcal{I}$ and $\mathcal{S}$, and two approximation bounds $\tau \geq 0$ and $\varepsilon \geq 0$; assume that MTSs $MTS(\mathcal{I})$ and $MTS(\mathcal{S})$ are translations of $\mathcal{I}$ and $\mathcal{S}$, respectively. Then, the following statement holds:*

$$MTS(\mathcal{I}) \equiv_{\tau,\varepsilon} MTS(\mathcal{S}) \Rightarrow \mathcal{I} \approx_{\varepsilon,2\tau} \mathcal{S}.$$

Proof. In the proof to follow, we assume that $\mathcal{I}$ and $\mathcal{S}$ are of the form $\mathcal{I} : H \times \mathbb{TSS}(V_I, N) \to \mathbb{TSS}(V_O, N_{\mathcal{I}})$ and $\mathcal{S} : H \times \mathbb{TSS}(V_I, N) \to \mathbb{TSS}(V_O, N_{\mathcal{S}})$, respectively.

Consider $h_0 \in H_0$, $(x, \sigma_x) \in \mathbb{TSS}(V_I, N)$, and $(y, \sigma_y) \in \mathbb{TSS}(V_O, N)$ such that $(y, \sigma_y) = \mathcal{I}(h_0, (x, \sigma_x))$; in order to prove the theorem, we need to find a TSS $(y', \sigma_{y'}) \in \mathbb{TSS}(V_O, N_{\mathcal{I}})$ such that $(y', \sigma_{y'}) = \mathcal{I}(h_0, (x, \sigma_x))$ and $(y, \sigma_y) \approx_{\varepsilon,2\tau} (y', \sigma_{y'})$.

It follows from Definition 16 that in $MTS(\mathcal{I})$, we have $\perp \xrightarrow{t} \langle h_0, (x, \sigma_x), (y, \sigma_y), \sigma_{y,i}\rangle$, for each $i < N_{\mathcal{I}}$. It also follows from $MTS(\mathcal{I}) \equiv_{\tau,\varepsilon} MTS(\mathcal{S})$ that there exists an approximate bisimulation relation $R_{\varepsilon,\tau}$ such that $(\{\perp\}, \{\perp\}) \in R_{\varepsilon,\tau}$. Due to the latter statement and the t-labeled transition of $\perp$ in $MTS(\mathcal{I})$, we obtain in $MTS(\mathcal{S})$ that for some $(y', \sigma_{y'}) \in \mathbb{TSS}(V_O, N_{\mathcal{S}})$ and $j \leq N_{\mathcal{S}}$, $\perp \xrightarrow{t'} \langle h_0, (x, \sigma_x), (y', \sigma'_y), \sigma_{y',j}\rangle$ such that $|\sigma^1_{y,i} - \sigma^1_{y',j}| \leq \tau$, $||y_i - y'_j|| \leq \varepsilon$, and $(\langle h_0, (x, \sigma_x), (y, \sigma_y), \sigma_{y,i}\rangle, \langle h_0, (x, \sigma_x), (y', \sigma_{y'}), \sigma_{y',j}\rangle) \in R_{\varepsilon,\tau}$. In the remainder of the proof, without loss of generality, we assume that $\sigma^1_{y,i} - \sigma^1_{y',j}$ is non-negative; the proof for the case where $\sigma^1_{y,i} - \sigma^1_{y',j}$ is negative is symmetric and is hence, dispensed with. We claim that for any such (y', σ'_y), it holds that $(y, \sigma_y) \approx_{\varepsilon,\tau} (y', \sigma_{y'})$.

To see why this claim holds, take an arbitrary point $\sigma_{y',k}$ for $k < N_{\mathcal{S}}$. We have in $MTS(\mathcal{S})$ that $\langle h_0, (x, \sigma_x), (y', \sigma'_y), \sigma_{y',j}\rangle \xrightarrow{\sigma^1_{y',k} - \sigma^1_{y',j}} \langle h_0, (x, \sigma_x), (y', \sigma'_y), \sigma_{y',k}\rangle$. We have that $(\langle h_0, (x, \sigma_x), (y, \sigma_y), \sigma_{y,i}\rangle, \langle h_0, (x, \sigma_x), (y', \sigma_{y'}), \sigma_{y',j}\rangle) \in R_{\varepsilon,\tau}$ and hence, we obtain in

$MTS(\mathcal{I})$ that for some $m \leq N_{\mathcal{I}}$, $\langle h_0, (x, \sigma_x), (y, \sigma_y), \sigma_{y,i}\rangle \overset{\sigma^1_{y,m}-\sigma^1_{y,i}}{\longrightarrow} \langle h_0, (x, \sigma_x), (y, \sigma_y), \sigma_{y,m}\rangle$ such that $|(\sigma^1_{y,m} - \sigma^1_{y,i}) - (\sigma^1_{y',k} - \sigma^1_{y',j})| \leq \tau$ and $||y_k - y'_m|| \leq \varepsilon$. We only need to show that $|\sigma^1_{y,k} - \sigma^1_{y',m}| \leq 2tau$. We distinguish the following two cases based on whether $\sigma^1_{y,m} - \sigma^1_{y,i}$ is negative or not.

- $(\sigma^1_{y,m} - \sigma^1_{y,i}) - (\sigma^1_{y',k} - \sigma^1_{y',j}) \leq 0$, then we have:

$$\begin{array}{ll} (\sigma^1_{y,m} - \sigma^1_{y',k}) + \tau & \leq \\ (\sigma^1_{y,m} - \sigma^1_{y',k}) + (\sigma^1_{y',j} - \sigma^1_{y,i}) & = \\ (\sigma^1_{y,m} - \sigma^1_{y,i}) - (\sigma^1_{y',k} - \sigma^1_{y',j}) & \leq \\ 0 & \end{array}$$

  To conclude $(\sigma^1_{y,m} - \sigma^1_{y',k}) + \tau < 0$, that is $|\sigma^1_{y,m} - \sigma^1_{y',k}| < \tau < 2tau$, which was to be shown.
- $(\sigma^1_{y,m} - \sigma^1_{y,i}) - (\sigma^1_{y',k} - \sigma^1_{y',j}) > 0$, then we have:

$$\begin{array}{ll} (\sigma^1_{y,m} - \sigma^1_{y',k}) & = \\ (\sigma^1_{y,m} - \sigma^1_{y',k}) - (\sigma^1_{y',j} - \sigma^1_{y,i}) + (\sigma^1_{y',j} - \sigma^1_{y,i}) & = \\ ((\sigma^1_{y,m} - \sigma^1_{y,i}) - (\sigma^1_{y',k} - \sigma^1_{y',j})) + (\sigma^1_{y',j} - \sigma^1_{y,i}) & \leq \\ \tau + tau & = \\ 2\tau & \end{array}$$

  To summarize, $(\sigma^1_{y,m} - \sigma^1_{y',k}) \leq 2\tau$, and this concludes the proof. ◀

6 Challenges

6.1 Notions of Conformance

Partial models

Practical models are typically incomplete: firstly, they do not specify what the consequence of applying each and every intput at each and every state is (i.e., they may not be input-enabled). Secondly, they may not specify / allow to inspect the values of all variables at every state. Also, another aspects of partiality that may come in handy is to allow for different sets of variables in the specification and implementation: implementations tend to be defined in terms of far more variables than the specification and an abstraction mechanism should relate the implementation variables to the abstract variables in the specification.

HIOCO supports partiality in terms of non-input-enabled specifications, but other aspects of partiality are still missing in the theory. The notion of (τ, ε)-conformance does not support partial models and hence, its extension in this direction can be useful. Establishing a model of partiality for MTSs requires a generic framework for defining observations and metrics.

Partial models pose a further theoretical challenge for defining a notion of conformance that is a pre-order and pre-congruence; these aspects are discussed in the remainder of this section.

Non-determinism

At a high level of abstraction, many choices cannot be made or are irrelevant. Hence, non-deterministic models are natural phenomena at high levels of abstraction. It turns out that

testing based on non-deterministic models is significantly more complex and computationally more demanding than testing based on deterministic models. There are several proposals to allow for a restricted form of non-determinism [47, 62], which could be employed to reconcile non-determinism, e.g., with (τ, ε)-conformance.

It should also be noted that abstraction from internal actions is not supported by the current definitions of (τ, ε)-conformance and approximate simulation, which is useful when abstraction into internal actions is introduced in modeling and the design trajectory.

Pre-order (or Equivalence)

The notions of HIOCO and (τ, ε)-conformance are known not to be transitive for different reasons. Namely, HIOCO allows for partial models and hence, the set of traces tested in two pairs of conforming models may be different. (This is a straightforward extension of the original counter-example of transitivity for ioco [53].) For (τ, ε)-conformance the "conformance bounds" (τ and ε) add up when comparing two pairs of related models and hence, transitivity breaks. Finding reasonable conditions for establishing a pre-order conformance relation for partial models may help in exploiting such a notion in top-down design trajectory. Also regarding conformance bounds, calculating new bounds on the compositions of conformance relations is an interesting research direction.

Pre-congruence (compositionality)

Coming up with a notion of conformance testing that supports partial models and / or approximate comparison of behaviors and is also compositional is another non-trivial challenge. In addition to [57], there seems to be renewed interest in addressing this challenge in the context of IOCO [12, 21]. Also the authors of [13] suggest that it may be possible to achieve new results through the meta-theory of structural operational semantics.

Logical characterization

The notions of testing developed in concurrency theory are often motivated by a notion observer that can perform certain interactions with the system [5, 26]. This type of theoretical connection between the extensional and intensional notions seems to be open for the notion of conformance studied in this paper. Along the same lines, it is unclear what the logical characterization of these notions are, i.e., what class of properties (e.g., in extensions of modal μ-calculus or temporal logic [24, 25, 29]) are preserved under these notion of conformance. We refer to [4, 28] for some related results in this direction.

Distributed systems / testers

Conformance testing of asynchronous and distributed systems has been extensively studied; examples of recent work in adapting different notions of conformance to asynchronous and distributed settings are [37, 48, 51, 55, 61]. Distributed testing introduces several issues regarding nondeterminism and controllability in test-case generation, as well as reconstructing observation from partial distributed observations when checking conformance. This combination is known to be notoriously difficult and becomes even more challenging in the setting of cyber-physical systems.

Quiescence, agility, and Zeno behavior

Observing quiescence, i.e., lack of outputs and internal actions, has been a key aspect of the original IOCO theory [53], but it has been left out of HIOCO. In HIOCO, quiescence has been replaced with the agility observation which serves a different purpose, namely to note when time may or may not pass. A careful analysis of these two observations (i.e., quiescence and agility) and the different possible combinations of choices concerning them remains to be done.

Zeno behavior and chattering [42, 65] are modeling phenomenon, which bear resemblance to the divergence issue in the traditional notions of conformance testing. An ideal notion of conformance should be sensitive to these issues and distinguish systems featuring and lacking Zeno behavior. The notion of (τ, ε)-conformance counts the number of discrete steps in order to distinguish Zeno- and non-Zeno behavior, which is in our view somewhat ad-hoc. It is also contrary to the intuition that discrete jumps should be visible only through their observable effects. We believe, hence, that a more thorough treatment of Zeno behavior in the notions of conformance for cyber-physical systems is due.

6.2 Test-Case Generation

Robustness

In [35], a notion of robustness in test-case generation has been introduced for (τ, ε)-conformance. Robustness ensures the generated test-cases remain within the (τ, ε) boundaries of guard conditions (for transitions) and invariants of state space. This ensures that off-line test case can always be applied to the implementation without forcing it into an unintended state or violating the invariant. However, this turns out to be very challenging for generic hybrid systems, which requires solutions to Ordinary Differential Equations (ODEs) or Differential Algebraic Equations (DAEs). Finding a practical subset of hybrid system models for which robustness can be decided efficiently and building a test-case generation algorithm around it is a practical challenge.

Currently the notion of (τ, ε)-conformance assumes a single conformance bound for all variables; this needs to be further detailed, perhaps into a vector-valued conformance bound. Moreover, the useful notion of "conformance degree" [4], which determines the least conformance bounds can be adapted to this vector-valued setting, allowing for a Pareto analysis of conformance.

Coverage

Defining a model-based notion of coverage is still a relevant research question. In [15, 17, 40], some recent attempts have been made in consolidating traditional notions of coverage such as regularity and uniformity [30]; these notions are formalizations of category-partition techniques. Regarding continuous dynamics, various notions of coverage have been studied in [22, 23, 27, 41]. A combination of these ideas need to be considered for defining a satisfactory notion of coverage for hybrid systems.

Sampling

The choice of sampling in [4] is left unspecified and is mentioned as future work. This is a non-trivial problem and an appropriate choice of sampling requires knowing the solutions to the systems dynamics equations. Coverage-based choice of sampling is a possible solution to this issue [22, 23, 27, 41], which requires further investigation in the case of (τ, ε)-conformance.

6.3 Practical challenges

Tooling

We are aware of only few academic tools for conformance testing of cyber-physical systems, namely, HTG [22] and S-TaLiRo [39]. Also in [35] a prototype Matlab-based tool is reported which implements a test-case generation and conformance checking algorithm based on the notion of (τ, ε)-conformance. Developing tools and benchmarks for conformance testing of cyber-physical systems is certainly a challenge for the years to come.

Rigorous Models

Formal models are the starting point of model-based testing and yet, they often do not exist for industrial systems. This makes bars wide application of model-based testing in industrial practice. We see two possible solutions to this problem. In some domains, (domain-specific) languages are common in the design trajectory. For example, Matlab Simulink models are common, among others, in the automotive domain. Hence, building model-based testing tools that use such domain-specific models as their input language can be very beneficial. (Reactis and S-TaLiRo, respectively, are examples of commercial and academic model-based testing tools with such an input language.)

Model-learning techniques (see, e.g., [1, 2]) (also called learning-based testing [45] and test-based modeling [54]) may provide an alternative path to building or updating models for model-based testing.

Online testing

Applying on-line testing techniques to cyber-physical systems requires predictable real-time performance of test-case generation, test-case execution and conformance checking algorithms, none of which is trivial. Moreover, in the presence of non-determinism, conformance checking requires calculating the set of current states in a huge state-space, which becomes intractable in concurrent systems; employing on-the-fly reduction techniques in such a setting is inevitable.

Acknowledgments. We thank Pieter Cuijpers, Eugenio Moggi, Wojciech Mostowski, Michel A. Reniers, and Walid Taha for providing insightful comments on earlier drafts of this paper.

References

1 Fides Aarts, Bengt Jonsson, Johan Uijen, and Frits W. Vaandrager. Generating models of infinite-state communication protocols using regular inference with abstraction. *Formal Methods in System Design*, 46(1):1–41, 2015.

2 Fides Aarts and Frits W. Vaandrager. Learning I/O automata. In *Proceedings of the 21th International Conference on Concurrency Theory (CONCUR 2010)*, volume 6269 of *Lecture Notes in Computer Science*, pages 71–85. Springer, 2010.

3 Houssam Abbas, Bardh Hoxha, Georgios E. Fainekos, J. V. Deshmukh, James Kapinski, and Koichi Ueda. WiP abstract: Conformance testing as falsification for cyber-physical systems. In *Proceedings of the ACM/IEEE 5th International Conference on Cyber-Physical Systems (ICCPS 2014)*, page 211. IEEE CS, 2014. Available online: http://arxiv.org/abs/1401.5200.

4 Houssam Abbas, Hans Mittelmann, and Georgios E. Fainekos. Formal property verification in a conformance testing framework. In *12th ACM-IEEE International Conference on*

Formal Methods and Models for System Design (MEMOCODE 2014), pages 155–164. IEEE, 2014.

5 Samson Abramsky. Observation equivalence as a testing equivalence. *Theoretical Computer Science*, 53(2-3):225–241, 1987.

6 Bernhard K. Aichernig, Harald Brandl, Elisabeth Jöbstl, and Willibald Krenn. Model-based mutation testing of hybrid systems. In *Revised Selected Papers from the 8th International Symposium on Formal Methods for Components and Objects (FMCO 2009)*, volume 6286 of *Lecture Notes in Computer Science*, pages 228–249. Springer, 2010.

7 Bernhard K. Aichernig, Harald Brandl, and Franz Wotawa. Conformance testing of hybrid systems with qualitative reasoning models. *Electronic Notes in Theoretical Computer Science*, 253(2):53–69, 2009. Proceedings of Fifth Workshop on Model Based Testing (MBT 2009).

8 Rajeev Alur. Formal verification of hybrid systems. In *Proceedings of the 11th International Conference on Embedded Software (EMSOFT 2011)*, pages 273–278. ACM, 2011.

9 Rajeev Alur, Costas Courcoubetis, Nicolas Halbwachs, Thomas A. Henzinger, Pei-Hsin Ho, Xavier Nicollin, Alfredo Olivero, Joseph Sifakis, and Sergio Yovine. The algorithmic analysis of hybrid systems. *Theoretical Computer Science*, 138(1):3–34, 1995.

10 Jos C. M. Baeten. A brief history of process algebra. *Theoretical Computer Science*, 335(2-3):131–146, 2005.

11 Klaus Bender, Manfred Broy, István Péter, Alexander Pretschner, and Thomas Stauner. Model based development of hybrid systems: Specification, simulation, test case generation. In *Modelling, Analysis, and Design of Hybrid Systems*, volume 279 of *Lecture Notes in Control and Information Sciences*, pages 37–51. Springer, 2002.

12 Nikola Benes, Przemyslaw Daca, Thomas A. Henzinger, Jan Kretínský, and Dejan Nickovic. Complete composition operators for IOCO-testing theory. In *Proceedings of the 18th International ACM SIGSOFT Symposium on Component-Based Software Engineering (CBSE 2015)*, pages 101–110. ACM, 2015.

13 Harsh Beohar and Mohammad Reza Mousavi. A pre-congruence format for XY-simulation. In *Proceedings of the 6th International Conference on Fundamentals of Software Engineering (FSEN 2015)*, Lecture Notes in Computer Science, 2015.

14 Harald Brandl, Martin Weiglhofer, and Bernhard K. Aichernig. Automated conformance verification of hybrid systems. In *Proceedings of the 10th International Conference on Quality Software (QSIC 2010)*, pages 3–12. IEEE CS, 2010.

15 Cécile Braunstein, Anne Elisabeth Haxthausen, Wen-ling Huang, Felix Hübner, Jan Peleska, Uwe Schulze, and Linh Vu Hong. Complete model-based equivalence class testing for the ETCS ceiling speed monitor. In *Proceedings of the 16th International Conference on Formal Engineering Methods on Formal Methods and Software Engineering (ICFEM 2014)*, volume 8829 of *Lecture Notes in Computer Science*, pages 380–395. Springer, 2014.

16 Manfred Broy, Bengt Jonsson, Joost-Pieter Katoen, Martin Leucker, and Alexander Pretschner. *Model-Based Testing of Reactive Systems: Advanced Lectures*, volume 3472 of *Lecture Notes in Computer Science*. Springer, 2005.

17 Ana Cavalcanti and Marie-Claude Gaudel. Data flow coverage for circus-based testing. In *Proceedings 17th International Conference of the Fundamental Approaches to Software Engineering (FASE 2014)*, volume 8411 of *Lecture Notes in Computer Science*, pages 415–429. Springer, 2014.

18 Pieter Cuijpers. *Hybrid Process Algebra*. PhD thesis, Department of Computer Science, Eindhoven University of Technology, 2004.

19 Pieter Cuijpers, Michel Reniers, and Maurice Heemels. Hybrid transition systems. Technical Report CSR-02-12, Department of Computer Science, Eindhoven University of Technology, 2002.

20 Pieter J. L. Cuijpers and Michel A. Reniers. Lost in translation: Hybrid-time flows vs. real-time transitions. In *Proceedings of the 11th International Workshop on Hybrid Systems: Computation and Control (HSCC 2008)*, volume 4981 of *Lecture Notes in Computer Science*, pages 116–129. Springer, 2008.

21 Przemyslaw Daca, Thomas A. Henzinger, Willibald Krenn, and Dejan Nickovic. Compositional specifications for ioco testing. In *Proceedings of the 7th IEEE International Conference on Software Testing, Verification and Validation (ICST 2014)*, pages 373–382. IEEE CS, 2014.

22 Thao Dang. Model-based testing of hybrid systems. In Justyna Zander, Ina Schieferdecker, and Pieter J. Mosterman, editors, *Model-based Testing for Embedded Systems*, pages 383–424. CRC Press, 2011.

23 Thao Dang and Tarik Nahhal. Coverage-guided test generation for continuous and hybrid systems. *Formal Methods in System Design*, 34(2):183–213, 2009.

24 Jennifer M. Davoren. On hybrid systems and the modal μ-calculus. In *Hybrid Systems V*, volume 1567 of *Lecture Notes in Computer Science*, pages 38–69. Springer, 1999.

25 Jennifer M. Davoren, Vaughan Coulthard, Nicolas Markey, and Thomas Moor. Non-deterministic temporal logics for general flow systems. In *Proceedings of the 7th International Workshop on Hybrid Systems: Computation and Control (HSCC 2004)*, volume 2993 of *Lecture Notes in Computer Science*, pages 280–295. Springer, 2004.

26 Rocco De Nicola and Matthew Hennessy. Testing equivalences for processes. *Theoretical Computer Science*, 34:83–133, 1984.

27 Tommaso Dreossi, Thao Dang, Alexandre Donzé, James Kapinski, Xiaoqing Jin, and Jyotirmoy V. Deshmukh. Efficient guiding strategies for testing of temporal properties of hybrid systems. In *Proceedings of the 7th International NASA Formal Methods Symposium (NFM 2015)*, volume 9058 of *Lecture Notes in Computer Science*, pages 127–142. Springer, 2015.

28 Georgios E. Fainekos and George J. Pappas. Robustness of temporal logic specifications for continuous-time signals. *Theoretical Computer Science*, 410(42):4262–4291, 2009.

29 Carlo A. Furia and Matteo Rossi. On the expressiveness of MTL variants over dense time. In *Proceedings of the 5th International Conference on Formal Modeling and Analysis of Timed Systems (FORMATS 2007)*, volume 4763 of *Lecture Notes in Computer Science*, pages 163–178. Springer, 2007.

30 Marie-Claude Gaudel. Testing can be formal, too. In *Proceedings of the 6th International Joint Conference on Theory and Practice of Software Development (TAPSOFT 1995)*, volume 915 of *Lecture Notes in Computer Science*, pages 82–96. Springer, 1995.

31 Alessandro Giacalone, Chi-Chang Jou, and Scott A Smolka. Algebraic reasoning for probabilistic concurrent systems. In *Proceedings of the IFIP TC2 Working Conference on Programming Concepts and Methods (PROCOMET 1990)*, pages 443–458. North-Holland, 1990.

32 Antoine Girard, A. Agung Julius, and George J. Pappas. Approximate simulation relations for hybrid systems. *Discrete Event Dynamic Systems*, 18(2):163–179, 2008.

33 Antoine Girard and George J. Pappas. Approximation metrics for discrete and continuous systems. Technical Report MS-CIS-05-10, Dept. of CIS, University of Pennsylvania, 2005.

34 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.

35 Antoine Girard and George J. Pappas. A tool prototype for model-based testing of cyber-physical systems. Technical Report CST 2015.090, Contol Systems Group, Dept. of Mechanical Engineering, Eindhoven University of Technology, 2015.

36 Rafal Goebel, Ricardo G. Sanfelice, and Andrew R. Teel. Hybrid dynamical systems. *IEEE Control Systems*, 29(2):28–93, 2009.

37 Robert M. Hierons. Generating complete controllable test suites for distributed testing. *IEEE Trans. Software Eng.*, 41(3):279–293, 2015.

38 Robert M. Hierons, Kirill Bogdanov, Jonathan P. Bowen, Rance Cleaveland, John Derrick, Jeremy Dick, Marian Gheorghe, Mark Harman, Kalpesh Kapoor, Paul J. Krause, Gerald Lüttgen, Anthony J. H. Simons, Sergiy A. Vilkomir, Martin R. Woodward, and Hussein Zedan. Using formal specifications to support testing. *ACM Computing Surveys*, 41(2):9:1–9:76, 2009.

39 Bardh Hoxha, Houssam Abbas, and Georgios E. Fainekos. Using S-TaLiRo on industrial size automotive models. In *Proceedings of the Applied Verification for Continuous and Hybrid Systems (ARCH 2014)*, 2014.

40 Wen-ling Huang and Jan Peleska. Exhaustive model-based equivalence class testing. In *Proceedings of the 25th IFIP WG 6.1 International Conference of Testing Software and Systems (ICTSS 2013)*, volume 8254 of *Lecture Notes in Computer Science*, pages 49–64. Springer, 2013.

41 A. Agung Julius, Georgios E. Fainekos, Madhukar Anand, Insup Lee, and George J. Pappas. Robust test generation and coverage for hybrid systems. In *Proceedings of the 10th International Workshop on Hybrid Systems: Computation and Control (HSCC 2007)*, volume 4416 of *Lecture Notes in Computer Science*, pages 329–342. Springer, 2007.

42 Daniel Liberzon. *Switching in Systems and Control*. Systems & Control: Foundations and Application. Birkhäuser, 2003.

43 Nancy Lynch, Roberto Segala, and Frits Vaandrager. Hybrid I/O automata. *Information and Computation*, 185(1):105–157, 2003.

44 Oded Maler, Zohar Manna, and Amir Pnueli. From timed to hybrid systems. In *Proceedings of the REX Workshop on Real-Time: Theory in Practice*, volume 600 of *Lecture Notes in Computer Science*, pages 447–484. Springer, 1992.

45 Karl Meinke, Fei Niu, and Muddassar A. Sindhu. Learning-based software testing: A tutorial. In *International Workshops on Leveraging Applications of Formal Methods, Verification, and Validation*, volume 336 of *Communications in Computer and Information Science*, pages 200–219. Springer, 2012.

46 Morteza Mohaqeqi, Mohammad Reza Mousavi, and Walid Taha. Conformance testing of cyber-physical systems: A comparative study. In *Proceedings of the 14th International Workshop on Automated Verification of Critical Systems (AVOCS 2014)*, volume 70 of *Electronic Communications of the EASST*, 2014.

47 Neda Noroozi. *Improving Theories of Input Output Conformance Testing*. PhD thesis, Eindhoven University of Technology, The Netherlands, 2015.

48 Neda Noroozi, Ramtin Khosravi, Mohammad Reza Mousavi, and Tim A. C. Willemse. Synchrony and asynchrony in conformance testing. *Software and System Modeling*, 14(1):149–172, 2015.

49 Neda Noroozi, Mohammad Reza Mousavi, and Tim A. C. Willemse. Decomposability in input output conformance testing. In *Proceedings of the 8th Workshop on Model-Based Testing (MBT 2013)*, volume 111 of *Electronic Proceedings in Theoretical Computer Science*, pages 51–66, 2013.

50 Jan Willem Polderman and Jan C. Willems. *Introduction to Mathematical Systems Theory: A Behavioral Approach*, volume 26 of *Texts in Applied Mathematics*. Springer, 1998.

51 Adenilso Simao and Alexandre Petrenko. From test purposes to asynchronous test cases. In *Third International Conference on Software Testing, Verification, and Validation Workshops (ICSTW 2010)*, pages 1–10. IEEE CS, 2010.

52 Paulo Tabuada. Approximate simulation relations and finite abstractions of quantized control systems. In *Proceedings of the 10th International Workshop on Hybrid Systems:*

Computation and Control (HSCC 2007), volume 4416 of *Lecture Notes in Computer Science*, pages 529–542. Springer, 2007.

53 Jan Tretmans. *A formal Approach to conformance testing.* PhD thesis, University of Twente, The Netherlands, 1992.

54 Jan Tretmans. Model-based testing and some steps towards test-based modelling. In *Advanced Lectures of the 11th International School on Formal Methods for the Design of Computer, Communication and Software Systems (SFM 2011)*, volume 6659 of *Lecture Notes in Computer Science*, pages 297–326. Springer, 2011.

55 Jan Tretmans and Louis Verhaard. A queue model relating synchronous and asynchronous communication. In *Proceedings of the IFIP Symposium on Protocol Specification, Testing and Verification XII*, pages 131–145, Amsterdam, The Netherlands, The Netherlands, 1992. North-Holland Publishing Co.

56 Franck van Breugel, Claudio Hermida, Michael Makkai, and James Worrell. An accessible approach to behavioural pseudometrics. In *Proceedings of the 32nd International Colloquium on Automata, Languages and Programming (ICALP 2005)*, volume 3580 of *Lecture Notes in Computer Science*, pages 1018–1030. Springer, 2005.

57 Machiel van der Bijl, Arend Rensink, and Jan Tretmans. Compositional testing with IOCO. In *Proceedings of the 3rd International Workshop on Formal Approaches to Testing of Software (FATES 2003)*, volume 2931 of *Lecture Notes in Computer Science*, pages 86–100. Springer, 2004.

58 Arjan van der Schaft and Hans Schumacher. *An Introduction to Hybrid Dynamical Systems*, volume 251 of *Lecture Notes in Control and Information Sciences*. Springer, 2000.

59 Michiel van Osch. Hybrid input-output conformance and test generation. In *Formal Approaches to Software Testing and Runtime Verification*, volume 4262 of *Lecture Notes in Computer Science*, pages 70–84. Springer, 2006.

60 Michiel van Osch. *Automated Model-based Testing of Hybrid Systems.* PhD thesis, Eindhoven University of Technology, The Netherlands, 2009.

61 Louis Verhaard, Jan Tretmans, Pim Pars, and Ed Brinksma. On asynchronous testign. In *Protocol Test Systems*, volume C-11 of *IFIP Transaction*, pages 55–66, 1992.

62 Martin Weiglhofer. *Automated Software Conformance Testing.* PhD thesis, Graz University of Technology, Austria, 2009.

63 Matthias Woehrle, Kai Lampka, and Lothar Thiele. Segmented state space traversal for conformance testing of cyber-physical systems. In *Proceedings of the 9th International Conference on Formal Modeling and Analysis of Timed Systems (FORMATS 2011)*, volume 6919 of *Lecture Notes in Computer Science*, pages 193–208. Springer, 2011.

64 Matthias Woehrle, Kai Lampka, and Lothar Thiele. Conformance testing for cyber-physical systems. *ACM Transactions on Embedded Computing Systems*, 11(4):84:1–84:23, 2013.

65 Jun Zhang, Karl Henrik Johansson, John Lygeros, and Shankar Sastry. Zeno hybrid systems. *International Journal of Robust and Nonlinear Control*, 11(5):435–451, 2001.